MASTEROPPGAVE

Masterstudium i skolerettet utdanningsvitenskap med fordypning i Matematikk

Mai 2022

Programmering med et matematisk læringspotensial

En studie om til hvilken grad lærere knytter arbeidet med programmering opp mot det matematikkfaglige læringsutbytte i undervisningen

Mischa van der Linden

Kandidatnr. 400

OsloMet – storbyuniversitetet

Fakultet for lærerutdanning og internasjonale studier

Institutt for grunnskole- og faglærerutdanning

Innholdsfortegnelse

Summary

In this master`s thesis, we are looking at the implementation of programming in mathematics education. The issue we try to answer is: To which degree do teachers make a connection between the mathematical learning outcome and working with programming?

To be able to answer this question three sub-issues have been formulated. These are 1) what is the mathematical learning potential in programming? 2) What do the teachers see as the mathematical learning outcome? And 3) What do teachers do to implement programming on behalf of the mathematics curriculum?

The theory review shows that there is a huge potential for working with programming and learning mathematics. Both overall concepts like computational thinking and problem solving, and specific concepts like geometry, coordinate systems, number sense, algebra and knowledge of measurement.

Using interviews and observation as qualitative research methods, the teachers in this study inform us about a variety of mathematical learning outcomes connected to the use of programming in education. What overall gives the biggest impact on a mathematical outcome is the use of pedagogical strategies, that makes the pupils use and develop mathematical skills. A second factor making the connection is the motivational aspect of programming. By making the work with programming motivating for the pupils, the pupils get more engaged in the mathematical concepts. The teachers role is essential in making the connection between mathematics and programming. Mainly by being the one who has to make the mathematical concepts explicit, by connecting it to the pupils prior knowledge. This requires well planning by the teacher. This planning needs to consist of targeted look for the mathematical potential in the task desired.

Keywords: Programming, mathematics education, computational thinking, mathematical concepts.

Sammendrag

I denne oppgaven har det blitt sett på implementering av programmering i matematikkfaget. Utgangspunktet for oppgaven er problemstillingen: Til hvilken grad knytter lærere det matematikkfaglige læringsutbyttet til arbeidet med programmering på mellomtrinnet?

For å kunne besvare denne har det i tillegg blitt utformet tre delproblemstillinger. Disse er henholdsvis 1) Hva er det matematiske læringspotensialet i arbeidet med programmering? 2) Hva anser lærerne at det matematiske læringsutbyttet er? og 3) Hva gjør lærerne for å implementere programmeringen på matematikkfagets premisser?

En omfattende teorigjennomgang, viser at det er et stort potensial for å bruke programmering som metode for å arbeide med matematiske konsepter. Både overordnede konsepter, som algoritmisk tenking og problemløsing, samt spesifikke konsepter, som geometri, koordinatsystemer, tallforståelse, algebra og måling.

En kvalitativ undersøkelse ved bruk av intervjuer og observasjoner, viser at lærerne i denne undersøkelsen, i likhet med teorien, ser mange muligheter for matematikkfaglig sammenheng til arbeidet med programmering. Det er likevel først og fremst de pedagogiske strategiene som blir brukt i undervisningen som skaper sammenhengen til matematikken ved at disse fører til at elevene arbeider med matematisk kompetanse. Elevenes motivasjon for arbeidet, ser videre ut til å spille en viktig rolle for det matematikkfaglige læringen, da det er muligheter for at elevene arbeider med matematiske konsepter i et motivasjonspreget medium. Det konkluderes med at lærerens rolle er svært sentral i å skape sammenhengen mellom oppgavene elevene gjør i en programmeringskontekst og det matematiske innholdet. Først og fremst ved å knytte det matematiske innholdet til elevenes forkunnskaper og gjøre dette eksplisitt i undervisningen. Dette krever god planlegging, og bevisst tilpasning av oppgaver, for å kunne utnytte potensialet som ligger i disse.

Nøkkelord: Matematikk, Programmering, Pedagogiske strategier, læringspotensial, Algoritmisk tenking.

Forord

Høsten 2019 begynte jeg på masterstudiet ved OsloMet. Det samme året gikk jeg inn i mitt fjerde år som lærer, og har de siste tre årene kombinerte jobb og studier. Dette har vært krevende, men lærerikt. Disse årene har gitt meg et godt faglig påfyll, som jeg opplever å få bruk for i jobben som lærer. Jeg er derfor takknemlig for at jeg har fått muligheten til å gjøre dette. Dette siste året har jeg jobbet med denne oppgaven som du sitter og leser nå.

Da jeg i 2015 skrev bacheloroppgaven min, snublet jeg ved en tilfeldighet over emnet programmering i en undervisningssammenheng. På dette tidspunktet var det få lærere som hadde hørt om, og som arbeidet med temaet programmering. Selv hadde jeg heller ikke tenkt på programmering som et tema som hadde en plass i matematikkfaget. Da programmering i undervisningen var noe nytt og fremmed, valgte jeg å fordype meg i temaet programmering og matematikk for grunnskolen. Jeg leste flere artikler som argumenterte for at programmering burte inn i skolen, og etter hvert som jeg fordypet meg i oppgaven fant jeg flere paralleller mellom den digitale verdenen med programmering og matematikkens undersøkende verden (Linden, 2015).

Da jeg nå videre skulle velge et tema for masteroppgaven min, vinglet jeg mellom videre forskning på bruken av programmering og temaet problemløsing og undersøkende aktiviteter i matematikkfaget. Etter hvert landet jeg på programmering som en del av matematikkfaget. Jeg opplever at temaet er relevant og spennende, og kan gi meg en innsikt i hva jeg kan bruke programmering til i min matematikkundervisning fremover.

Arbeidet med denne oppgaven har på bydd på en berg- og dalbane av følelser og bekymringer. Der jeg det ene øyeblikket har jublet fordi jeg endelig fikk tak i informanter, til det neste øyeblikket der hele oppgaven oppleves som et håpløst prosjekt uten fremgang. Andre ganger har ordene rast ned på papiret (les: PC-skjermen) i en kjempefart, der hodet var overfylt av tanker og ideer, men med total mangel på struktur. Veileders beskrivende ord «Snøras», var det eneste som kunne forklare det han nettopp hadde lest. Så i arbeidet med dette «snøraset», har jeg, forhåpentligvis, etter hvert maktet å dra ut det som var viktig. Det jeg ville frem til. For etter mange tårer, mye frustrasjon, og mange, lange dager kan det se ut til at jeg endelig har kommet i mål. Her følger altså en oppgave som interesserer meg, og som jeg håper interesserer deg som leser.

De siste tre årene, og kanskje særlig det siste året, hadde ikke vært mulig uten veldig gode støttespillere. Takk til informantene som sa seg villige til å hjelpe meg med informasjon til

denne oppgaven. Takk til veileder, Henrik Forssell, som har lest det ene snøraset av en tekst etter den andre, og hjulpet meg med å rydde frem det som var viktig. Takk til familien, som har støttet og dyttet meg over målstreken, og som har lest og gitt tilbakemeldinger underveis. En spesiell takk til samboeren min, Robin, som har vært en enorm støtte disse tre årene.

1 Innledning

Programmering har på kort tid fått en stadig større rolle i undervisningssektoren både i Norge og internasjonalt. Noen land har kommet lenger i utviklingen og bruken, andre har akkurat startet. Norge implementerte for kort tid siden programmering i læreplanen for fag som matematikk, naturfag, musikk og kunst og håndverk. I denne oppgaven vil jeg se på implementeringen av programmering fra et matematikkfaglig ståsted. Jeg vil også se på det som en implementert del av grunnskolens mellomtrinn, altså 5.-7.trinn. Innledningsvis vil jeg i dette kapittelet se på hva som har satt i gang bruken av programmering i norsk skole, og hvilke offentlige føringer som ligger til grunn for implementeringen. I tillegg vil jeg se på hvordan denne implementering kan oppfattes og hvilke konsekvenser det kan ha for lærerne som skal bruke programmeringen i sin undervisning. I senere kapitler vil jeg se på det matematikkfaglige læringspotensialet som ligger i programmeringen, og hvordan det skal legges til rette for arbeid med programmering i en matematikkfaglig kontekst. Basert på en kvalitativ studie med data innhentet fra både observasjoner og intervjuer, tar denne oppgaven sikte på å få et innblikk i bruken av programmering på matematikkfagets premisser, samt utnyttelsen av det matematikkfaglige læringspotensialet som ligger i programmeringsarbeidet.

1.1 Offentlige føringer

Innføringen av programmering i norsk skole er offentlig bestemt. Prosessen startet allerede i 2015, da leverte Ludvigsen-utvalget leverte NOU 2015:8 *Fremtidens skole – Fornyelse av fag og kompetanser*. Denne handlet om hvordan skolen og skolefagene bør endres for å tilpasses fremtidens behov. Det samme året kom NOU 2015:13 *Digital sårbarhet - sikkert samfunn. Beskytte enkeltmennesker og samfunn i en digitalisert verden*. Denne anbefalte å konkret innføre programmering i norsk skole som et ledd i utviklingen av å øke kompetansen i IKT. Dette bygget Meld.St. 28 (2015-2016) videre på og anbefalte å videreutvikle forståelsen av digital kompetanse og uttrykker at programmering bør være integrert i dette.

I Januar 2016 oppnevnte utdanningsdirektoratet en ekstern arbeidsgruppe, som skulle se på opplæringen og bruken av teknologi i grunnskolen (Sanne et al., 2016). I august samme året kom det en anbefaling om å opprette et eget fag som skulle inneholde både teknologi og programmering (Sanne et al., 2016). Senere kom Senter for IKT i utdanningen med et notat, basert på arbeidsgruppas rapport, *Teknologi og programmering – et nytt fag i skolen,* der de fremhevet at programmering bør implementeres i skolen, enten som et eget fag, i fagene eller som en overordnet kompetanse på tvers av fag (Sevik & m.fl., 2016).

Fire år senere, skjer implementeringen av programmering i norsk skole ved innføringen av fagfornyelsen i 2020. Etter et prøveprosjekt der flere ungdomskoler har hatt programmering som valgfag (Meld.St. 28 (2015-2016)), gikk nå læreplanen, LK20, inn for å legge programmering inn under kompetansemålene i blant annet matematikk, musikk, naturfag og kunst og håndverk. I tillegg til at det nå kan velges som et eget valgfag i ungdomsskolen (Kunnskapsdepartementet, 2019). Denne beslutningen om implementering i eksisterende fag ble da altså tråd i kraft til tross for at ekspertgruppen, oppnevnt av utdanningsdirektoratet, anbefalte å innføre programmering som et eget fag i grunnskolen (Kaufmann, Stenseth & Holone, 2018; Sanne et al., 2016; Sevik & m.fl., 2016).

I faget matematikk er arbeidet med algoritmisk tenkning og programmering direkte og indirekte nevnt i Læreplanen for hele grunnskolen. De grunnleggende ferdighetene i tilknytning til matematikkfaget, trekker spesifikt frem programmering som et av verktøyene elevene skal kunne ta i bruk for å arbeide med «utforsking og løse matematiske problemer» (Kunnskapsdepartementet, 2019, s. 5). Læreplanens kjerneelementer nevner vider algoritmisk tenking som «viktig i prosessen med å utvikle strategier og framgangsmåter for å løse problemer» (Kunnskapsdepartementet, 2019, s. 2) Videre nevnes programmering i kompetansemålene, der det i kompetansemålene for 4.-7.trinn heter at elevene skal kunne:

- «lage algoritmer og uttrykke dem ved bruk av variabler, vilkår og løkker» (etter 4.trinn) (Kunnskapsdepartementet, 2019, s. 8)
- «lage og programmere algoritmer med bruk av variabler, vilkår og løkker» (etter 5.trinn) (Kunnskapsdepartementet, 2019, s. 9)
- «bruke variabler, løkker, vilkår og funksjoner i programmering til å utforske geometriske figurer og mønstre» (etter 6. trinn) (Kunnskapsdepartementet, 2019, s. 10).
- «bruke programmering til å utforske data i tabeller og datasett» (etter 7. trinn) (Kunnskapsdepartementet, 2019, s. 11)

I kompetansemålene har Kunnskapsdepartementet (2019) lagt noen føringer for enkelte matematiske konsepter som skal knyttes til bruken av programmering. Blant disse finner vi geometri og tabeller og datasett. I tillegg til implementeringen i læreplanen vektlegges det i Utdanningsdirektoratets egen kompetansepakke for opplæring for bruken av programmering, at «Programmering i matematikk skal skje på fagets premisser» (Utdanningsdirektoratet, 2021a, Kjerneelementer og nøkkelbegrep, avsn.3). Videre legges det vekt på at

implementeringen skal fremme læring av matematikk både gjennom utforsking, men også gjennom matematikkfremmende samtaler. Med dette presiseres det at arbeidet med programmering skal knyttes til matematikkfaget og matematikkfaglig innhold. Dette samsvarer med det Sanne et al. (2016) beskriver i rapporten *Teknologi og programmering for alle*. I denne rapporten oppfordres det til å innføre et eget fag i skolen for arbeid med teknologi, med særlig vekt på programmering. Selv om det her fremmes innføring av et eget fag, påpekes det i rapporten at en innføring av blant annet programmering vil føre til endringer i faginnholdet i enkeltfag også, deriblant matematikkfaget. Tanken var at innlæringen av programmering gjennom et eget fag, kunne lede til bruken av programmering som metode i ulike fag. Derfor må implementeringen også «gjøres med respekt for egenarten i det enkelt faget samtidig som det faglige innholdet ses i lys av mulighetene som ligger i teknologi og programmering» (Sanne et al., 2016, s. 76). Det kan se ut til at arbeidet med programmering i tilknytning til matematikkfaget, skal bidra til en dypere forståelse for faget.

1.2 Begrunnelser for implementering

På et generelt grunnlag begrunnes implementeringen av programmering med at det er et viktig verktøy for å arbeide med kompetanser som blir viktige å kunne beherske i fremtiden. Disse ferdighetene blir kalt «kompetanser og ferdigheter for det 21.århundre» eller «21st Century Skills» (Sevik & m.fl., 2016, s. 10). Ferdighetene består blant annet i metakognisjon; «å lære seg å lære», «kritisk tenking og problemløsing», «kommunikasjon og samarbeid» og «kreativitet og innovasjon» for å nevne noen (NOU 2014:7, s. 127). Det er altså et bredt spekter av kompetanser. Innføringen av programmering i skolen begrunnes med er at læring av dette vil føre til en kompetansene som er relevante i fremtidens arbeidsliv (Utdanningsdirektoratet, 2020b). I andre Europeiske land begrunnes implementeringen av programmering i Læreplaner ved at skal fremme logisk tenking, problemløsing, engasjere elever til å ta i bruk IKT, utvikle elevenes ferdigheter i å kode, bidra til bedre rekrutering i arbeidsmarkedet samt andre kompetanser (European Schoolnet, 2015)

Utdanningsdirektoratet (2021b) begrunner valget av å implementere programmering i skolen med at det vil bli et viktig pedagogisk verktøy. Arbeidet stimulerer til problemløsing og skaperglede og åpner for ulike måter å lære på. I tillegg har arbeidet med programmering en motiverende rolle i form av at det kan skape en opplevelse av mestring

1.3 Nytt for lærerne

På relativt kort tid har programmering gått fra å være noe omdiskutert, og forholdsvis ukjent for de fleste lærere til å bli innført som metode i flere fag, deriblant matematikkfaget (Sevik & m.fl., 2016). På bakgrunn av dette kan en anta at mange lærere har lite erfaring med å implementere programmering i undervisningen. Det vil også være nærliggende å anta at programmering er et fagfelt som muligens få lærere har en formell kompetanse i. Begge antakelsene kan bekreftes. Berggren og Jom (2019) viser til at halvparten av lærere i deres spørreundersøkelse ikke hadde kjennskap til noe programmeringsspråk. I tillegg svarte omtrent halvparten av de spurte lærere «vet ikke» på spørsmål om hva programmering innebærer. Samtidig finner vi i rapporten *Teknologi og programmering for alle*, at programmering i eksisterende fag kunne føre til nedprioritering av bruken av programmering som metode, blant annet fordi lærere opplever at de ikke innehar nok kompetanse til å kunne ta det i bruk (Moreau, 2021; Sevik & m.fl., 2016). Det tar tid å lære seg noe nytt, samtidig som det krever kunnskaper å kunne implementere noe nytt i sin egen undervisning. Dette er noe jeg kan relatere til som lærer selv. Tiden det tar å sette seg inn i og å lære programmering, kan muligens for enkelte bli for overveldende. Dermed vil implementering i eksisterende fag kunne føre til at en velger å se bort i fra det som er nytt og krevende og velger å prioritere det som er kjent fra før.

Ved å implementere programmering og teknologi i faget, legger læreplanen føringer for at lærere ikke kun skal kunne grunnleggende programmeringsferdigheter, men også skal inneha en dypere kunnskap om både matematikk og programmering ved at læreren skal kunne knytte det matematikkfaglige innholdet, og den matematikkfaglige læringen til arbeidet. Dette kan, forståelig nok, bli svært krevende i en allerede hektisk arbeidshverdag. Særlig hvis det kreves at en skal sette seg inn i de nye ferdighetene og programmene selvstendig uten organisert kursing.

1.4 Tilstrekkelig kunnskap og utfordringer ved implementering

Det vil ved implementeringen av programmering i skolens eksisterende fag, være nærliggende å se på hvordan andre land i Norden har gjennomført denne prosessen og hvordan de arbeider med programmering i skolen og hvordan læreplanene legger føringer for bruken av programmering. Norge ligger forholdsvis langt bak, da flere andre land i Europa innførte programmering i skolene for I Sverige ble programmering innført i læreplanen i 2018, og en naturlig del av dette er at forskere ved universiteter og høyskoler allerede er i gang med å forske på kunnskapen blant lærere. Blant annet har Sverige i likhet med Norge også

implementert programmering i matematikkfaget Her har de besluttet at programmeringsarbeidet knyttet opp mot arbeidet med algebra og problemløsing i læreplanen. (Bråting & Kilhamn, 2021; Kilhamn, Bråting & Rolandsson, 2021; Stigberg & Stigberg, 2020).

I Sverige fikk svært få av de ansatte i skolen opplæring i bruken av programmering da dette ble innført i læreplanen. En konsekvens av dette var at lærerne følte seg usikre og nervøse i forbindelse med det å undervise i programmering (Stigberg & Stigberg, 2020). Dermed uttrykte en stor andel av disse lærerne at de ikke var godt nok forberedt for å undervise programmering i matematikkundervisningen. De innehar dermed ikke tilstrekkelig «Subject matter knowledge» om programmering for å kunne bruke dette som en ressurs i undervisningen (Stigberg & Stigberg, 2020). Mangelen på opplæring og kompetanse hos lærere kan resultere i at metodeaspektet ved programmeringen muligens kan forsvinne i arbeidet med å selv forstå og lære seg programmering og dens begreper og tankesett. Dette fordi lærerne ikke ser sammenhengen mellom de to komponentene programmering og matematikk (Misfeldt, Szabo & Helenius, 2019). Likevel virker det som at lærere er positive til innføringen av programmering i matematikkfaget og positive til å lære (Berggren & Jom, 2019; Misfeldt et al., 2019).

For at effekten av programmering som metode i matematikk skal være god, og for at dette skal implementeres i faget på en god måte, blir det viktig at lærerens kompetanse også er deretter (Misfeldt et al., 2019). Sannsynligvis vil kompetansen også medføre en større bevissthet rundt mulighetene for å implementere programmering som en metode på matematikkfagets premisser. Dette vil være en utfordring for mange lærere, da mange ikke har kunnskapen som kreves for å kunne undervise i programmering fra tidligere, og fordi mange lærere ikke vil klare å holde tritt med elevenes utvikling av programmerings ferdigheter (Forsstøm & Kaufmann, 2018; Haraldsrud, Sveinsson & Løvold, 2020). Likevel, vil jeg tro at dette er en utfordring som ikke kun gjelder programmering, men alle digitale hjelpemidler. Det kan være både vanskelig og krevende å legge til rette for undervisning i matematikk der de digitale hjelpemidlene bidrar til verdifull læring for elevene, særlig fordi en til en hver tid må være oppdatert på hvile hjelpemidler som er tilgjengelige og hvordan de kan brukes (Norstein & Haara, 2018).

Det å implementere programmering i læreplanen og dermed i skolen, er i likhet med endringer i skoleverket og i likhet med innføringer av nye læreplaner tidligere ikke være noen enkel prosess. Det er heller ikke slik at innføringen og endringen skjer over natten. Derfor vil det

være naturlig at det medbringer skepsis, motstand, usikkerhet og utfordringer i arbeidet. Norstein og Haara (2018) legger vekt på vanlige utfordringer i arbeidet med programmering. Blant disse finner vi utfordringer knyttet til differensiering, kunnskaper og å gjøre feil. Noen av disse er utfordringer vi møter på uavhengig av fag, mens andre utfordringer er spesifikke for arbeidet med programmering. Noen elever mestrer programmeringen raskt og har behov for rask progresjon til vanskeligere oppgaver mens andre elever ikke forstår programmeringskonseptet like raskt. I tillegg vil det å ikke se på feil som dukker opp som et nederlag, men heller se på feilsøking som en ressurs være en utfordring som krever mental omjustering. Da programmering er en relativt ny-innført ferdighet for allmenheten, og mange elever kanskje har vokst opp med kjennskap til og kunnskaper om programmering, vil lærere som skal lære dette i voksen alder gjerne ligge noe bak elevere i utviklingen av ferdigheter. Dette siste punktet er en utfordring som kan være ganske unik for programmering og bruken av digitale hjelpemidler. Som lærere er vi vant til å sitte med kunnskapen, og det å miste denne tyngden i arbeidet med programmering kan være vanskelig å godta for enkelte. Likevel vil sannsynligvis læreren sitte med den matematikkfaglige tyngden, som igjen bidrar til en balanse i kunnskapsforholdet mellom lærere og elever. De to første punktene er utfordringer en vil møte på i likhet med utforingene vi finner i alle fag, også matematikkfaget. Det er derfor utfordringer vi kjenner til fra før, og vi kan derfor finne løsninger.

2 Problemstilling

Implementeringen av programmering blir begrunnet med henvisning til utvikling av kompetanser som regnes som viktige for fremtiden. Da en i Norge har valg å implementere det i eksisterende fag, skal implementeringen skje på fagets premisser. Dette vekker en nysgjerrighet for hvordan programmeringen får innpass i faget blant lærere. I forskningen til både Misfeldt et al. (2019), Kilhamn et al. (2021) og Bråting og Kilhamn (2021) ser en at innføringen i Sverige ikke nødvendigvis medfører at lærerne ser sammenhengen mellom matematikk som fag og programmering. Det er mulig dette også kan gjelde i Norge. I denne oppgaven ønsker jeg å se på hvilket matematisk potensial som ligger i bruken av programmering, og hvordan lærere benytter seg av mulighetene programmeringen kan gi. Jeg ønsker derfor i denne oppgaven å undersøke problemstillingen:

- Til hvilken grad knytter lærere det matematiske læringsutbytte til arbeidet med programmering i undervisningen på mellomtrinnet?

Dette vil fungere som en overordnet problemstilling. Innunder denne problemstillingen dukker det opp flere spørsmål som blir relevante å kunne besvare. Det vil blant annet kunne oppstå et skille mellom litteratur og lærerens anseelse av læringsutbyttet i arbeidet med programmering. Derfor vil det være relevant å se på hva som i teorien blir ansett som det matematiske læringspotensialet i arbeidet med programmering, samt hva lærerne anser som læringsutbyttet. Det blir også relevant å se på hva som faktisk gjøres for å implementere programmeringsarbeidet på matematikkfagets premisser. Forskningsspørsmålene vil derfor være:

1) Hva er det matematiske læringspotensialet i arbeidet med programmering?
2) Hva anser lærerne at det matematiske læringsutbyttet er?
3) Hva gjør lærerne for å implementere programmeringen på matematikkfagets premisser?

Den første av de tre delproblemstillingene baserer seg på hva litteraturen ser på som det matematiske læringspotensialet og vil derfor kunne besvares gjennom litteraturen som tas i bruk i denne oppgaven. Den vil fungere som en veiledende problemstilling for å kunne si noe om hva det matematikkfaglige læringsutbyttet kan være. Når det kommer til de to siste delproblemstillingene, vil disse være basert på den innsamlede dataen fra intervjuer og observasjoner. Sammen vil disse bidra til å kunne besvare den overordnede problemstillingen.

3 Teoretisk bakgrunn

I dette kapittelet vil jeg se på hva eksisterende forsking sier om sammenhengen mellom programmering og matematikk. Dette kapittelet er delt inn i seks delkapitler, med tilhørende underkapitler. Innledningsvis, i Kap. 3.1 Tilbake i tid, vil jeg se på hvordan programmeringen har fått sin innpass i skolen, med et historisk perspektiv. Kap. 3.2 Koding og programmering er en kort avklaring av begrepene programmering og koding, og Kap. 3.3 Programmering og matematikk ser på hva eksisterende forskning sier om sammenhengen mellom programmering og matematikk i skolen. Etter dette følger Kap. 3.4 Matematisk kompetanse, som baserer seg på Niss et al. (2002) sin definisjon av begrepet matematisk kompetanse. Videre viser Kap. 3.5 Matematisk læringspotensial til overordnede og spesifikke konsepter som kan knyttes til bruken av programmering i undervisningen. Det siste kapittelet, Kap. 3.6 Lærerens rolle, tar for seg lærerens rolle ved implementering av programmering i undervisningen. I dette kapittelet utdypes blant rammeverket 5E-er (Benton, Hoyles, Kalas & Noss, 2016; Benton,

Hoyles, Kalas & Noss, 2017) og teorier om overføring av kunnskap (Salomon & Perkins, 1989).

3.1 Tilbake i tid og programmering inn i skolen

Utviklingen av matematikken og bruken av algoritmer, slik vi kjenner den i dag, startet for flere tusen år siden. For eksempel ble Rihnd-papirusen skrevet rundt 2000-1800 f.Kr. og Egypterne utviklet divisjon-, multiplikasjon og dobblingsalgoritmen og den indiske matematikeren Bhranagupta fant opp en løsningsalgoritme for andregradsligninger (Bueie, 2019). Den teoretiske forståelsen av begrepet «algoritme», og tilhørende begreper som «beregning» og «beregnbarhet», skjøt fart i begynnelsen av forrige århundre, blant annet utviklingen av matematiske modeller av beregning, som den såkalte Turing maskinen. Dette la grunnlaget for moderne informatikk og moderne datamaskiner (Copeland, 2020; Horsten, 2022). Fra et mer anvendt perspektiv finner vi også at «Den opprinnelige motivasjonen for å bygge datamaskiner var for å automatisere utførelsen av matematiske algoritmer» (Sanne et al., 2016, s. 89). Historisk kan en dermed se at datamaskinen i seg selv og måten den fungerer på har utspring fra matematikken og ideene er basert på ideen om beregnbarhet.

Etter hvert har datamaskinen fått sin plass i skolesystemet, og ut over 1960-tallet utviklet matematikeren Seymour Papert, ved MIT Artificial inteligence Laboratory, Logo programming (Foundation, 2015; Papert, 1980). Logo var et programmeringsverktøy der en skulle programmere en «skilpadde» til å bevege seg rundt på skjermen ved hjelp av enkle koder, og egnet seg godt for konstruksjon av geometriske figurer (Foundation, 2015). Tanken bak denne utviklingen av Logo var Piagets konstruktivistiske teori. Denne teorien handler om at læring er en konstant konstruksjon av kunnskap, der for eksempel et barn lærer gjennom sine egne erfaringer med verden de agerer i. Gjennom nye erfaringer, vil barnet tilegne seg ny kunnskap (Feurzeig, Papert & with a preface by Bob Lawler, 2011).

Mot slutten av 1970-tallet ble det innført et prosjekt blant skoler i New York der en skulle ta i bruk Logo i undervisningen. I løpet av 1980-tallet var arbeidet i fullgang med å få programmering inn i skolen og Seymour Papert fremmet «turtle-programming» som et relevant verktøy for bruk i matematikkundervisningen (Foundation, 2015). Arbeidet med Logo skulle bidra til at elevene kunne ta i bruk «powerfull ideas» fra blant annet matematikkfaget gjennom, for dem, et naturlig språk (Papert, 1980). Norge fulgte utviklingen ved å legge programmering inn som et element i Mønsterplanen for Grunnskolen (M87). I følge Bueie (2019) beskrev Mønsterplanen «(…) at arealberegninger, ligningsløsninger,

tilnærmingsmetoder og simulering passet godt for datamaskinen» (s. 25). På denne måten la den konkret føringer for hvilke matematiske konsept en kunne arbeide med i arbeidet med en datamaskin.

Logo har stadig blitt utviklet videre, ved blant annet utviklingen av LEGO Logo på midten av 1980-tallet av blant annet Mitchel Resnick ved MIT media Lab. Til grunn for Resnicks ideer lå en inspirasjon hentet fra Seymour Paperts tanker om læring og datamaskiner (Evans, 2017). Denne gangen ble programmeringen tatt enda et skritt videre ved motorisering av legobygd teknologi. I 2004 utviklet Life Long Kindergarten Group ved MIT Media Lab et Logo programmeringsprogram kalt Scratch. Som i dag blir brukt av elever og skoler rundt i verden. Programmet Scratch blir fremmet som et interaktivt programmeringsspråk som er godt egnet for arbeid med og læring av blant annet viktige matematiske konsepter, der ideen er at elevene lærer dette gjennom arbeidet med meningsfulle prosjekter (Resnick et al., 2009).

I 2014 utviklet BBC i samarbeid med flere store bedrifter Micro:bit (Micro:Bit, Undated). Dette er en liten firkantet datamaskin, som enkelt kan programmeres til å utfylle ulike funksjoner. Micro:Bit-en ble i 2016 ført inn i skolen i Storbritannia, ved at flere millioner elever fikk tildelt hver sin Micro:Bit (Micro:Bit, Undated). I Norge har det som en del av innføringen av programmering i læreplanen blitt i gang et prosjekt. *Super:Bit* er et samarbeid mellom *NRK, Vitensentret* og *Lær Kidsa Koding*, og satsingen tar sikte på å gi alle 6.klassinger en innføring i bruken av programmering, gjennom programmering av en Micro:Bit (Super:Bit, Undated).

Historisk sett, kan det altså se ut til at datamaskiner har sin rot i matematikken, samtidig som «algoritmen» spiller en viktig rolle i utviklingen av datamaskinen. Senere er det også matematikere som utvikler og prøver å skape programmer som kan lede arbeidet med programmering inn i skolen. De viser til at den visuelle funksjonen Logo hadde, godt kunne kombineres med matematiske konsepter. Fra et historisk perspektiv er det en sammenheng mellom programmering og matematikken. Likevel er forskningen rundt det matematiske utbyttet av arbeidet med programmering, har vært splittet (Forsstøm & Kaufmann, 2018). Dette til tross for at koblingen mellom matematikken og datamaskinen historisk vært nært knyttet. Henvisningen til M87 viser at programmering i seg selv, som en del av skolens læreplan og som et verktøy i matematikken heller ikke er noe nytt diskusjonstema.

3.2 Programmering og Koding

Programmering handler i stor grad om å få datamaskiner eller digitale programmer, til å gjøre en jobb. For å få dette til skriver, og fører man inn algoritmer til det programmet og den datamaskinen man ønsker at skal gjennomføre et arbeid (Kidsakoder, 2020; Sanne et al., 2016; Sevik & m.fl., 2016). For at dette skal fungere må algoritmen som skrives være skrevet korrekt, og implementeres i riktig rekkefølge. Dette betyr blant annet at beskjedene den inneholder må bli gitt i riktig rekkefølge. Videre innebærer det at den som skriver algoritmene må kunne tenke fremover og kunne forutse konsekvensen av en gitt handling; «Hvis man gjør sånn, så skjer dette». Denne formen for tenkning kalles «algoritmisk tenkning» eller på engelsk «computational thinking» (Kidsakoder, 2020). Algoritmisk tenking blir definert nærmere i Kap. 3.5.1. Algoritmisk tenking.

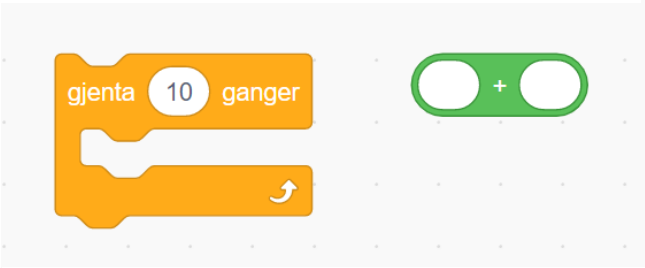
Figur3.1: Scratch - Eksempel blokkbasert programmeringsspråk (Scratch Foundation, 2019)

Det finnes flere ulike programmeringsspråk, der det skilles mellom tekstbaserte språk og mer grafiske språk som blokkbaserte språk. Eksempler på tekstprogrammeringsspråk kan være Javascript og Python, mens Scratch, Alice og Greenfoot er eksempler på blokkbaserte språk (Franklin et al., 2016; Kidsakoder, 2020; Maloney, Resnick, Rusk, Silverman & Eastmond, 2010). Tekstbaserte språk baserer seg på skriftlige koder, og settes sammen i bestemte rekkefølger for å få datamaskinen til å fungere.

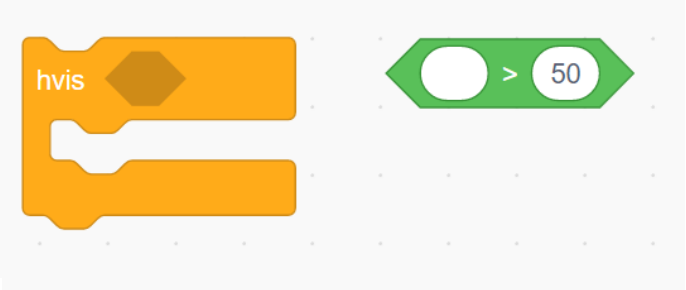
Blokkprogrammering fungerer på en slik måte at en i stedet for å skrive en hel instruksjon selv, kan plassere ferdig programmerte blokker med informasjon sammen. På denne måten kan de blokkbaserte språkene være en forenkling av de tekstbaserte språkene, og egner seg derfor godt blant de yngre elevene (Sanne et al., 2016). Blokkene er utformet slik at de

gjennom sin utforming veileder eleven. Dette gjør de ved å vise hvilke blokker som kan brukes sammen eller settes i kombinasjon. For eksempel har enkelte blokker «kroker», som i et puslespill, som viser hvilke blokker som passer inn (Se Figur 3.1). Eller at operatorene er formet slik at runde operatorer passer inn i runde «lommer» og sekskantede operatorer passer inn under sekskantede «lommer» (Figur 3.2 og Figur 3.3). På denne måten kan blokkprogrammering være en enkel måte for elever i komme i gang med programmering, uten at en behøver å ha særlige kunnskaper om hvordan syntaksen i programmeringsspråket skal være. Samtidig som en kan fokusere på prosessen i programmeringen ved å sette sammen koder.

Et program som skaper mulighet for å arbeide med både tekstprogrammering og blokkprogrammering, samt åpner for å se sammenhenger mellom disse er Micro:Bit. Dette er små firkantede datamaskiner. Micro:Bitten brukes som en interaktivt verktøy og består av alt fra lys- og lyd-sensorer til muligheten til å koble den opp til andre interaktive medier som for eksempel en Bit:Bot, roboter på hjul utformet som for eksempel en bil.



Figur 3.2: Scratch blokk Operator(Scratch Foundation, 2019)

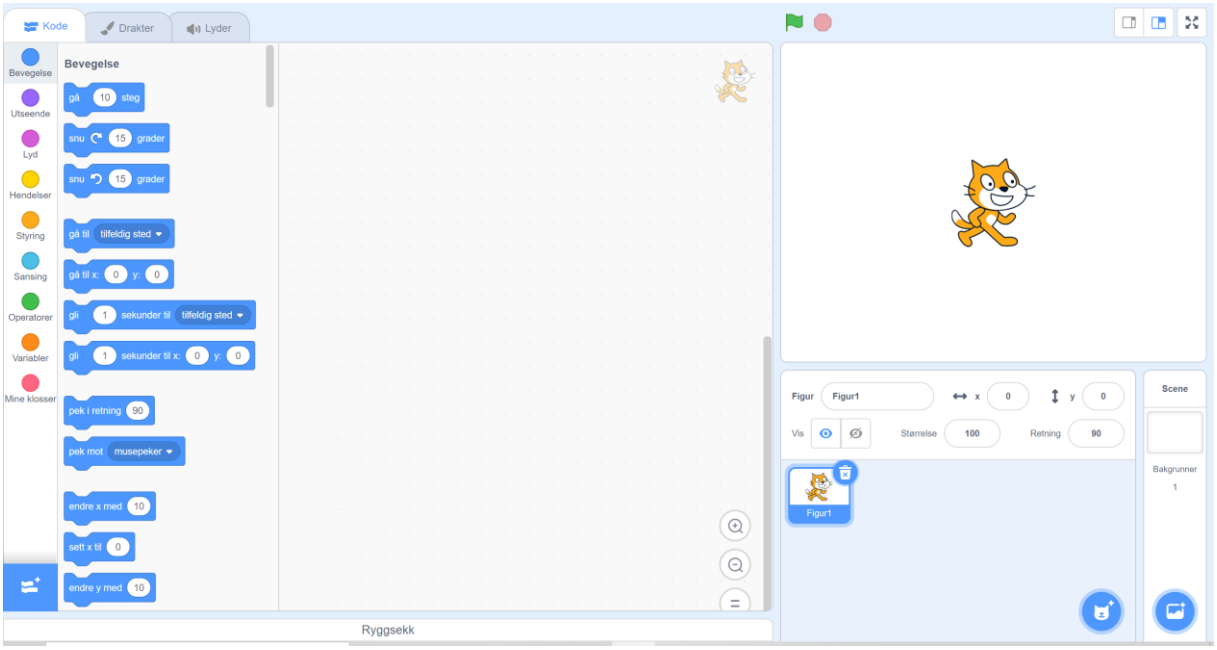


Figur 3.3: Scratch blokk Operator 2(Scratch Foundation, 2019)

Begrepene koding og programmering har ofte blitt brukt om hverandre. Likevel finner vi hovedsakelig igjen begrepet programmering i både læreplanen, NOUer, Rapporter og faglitteratur. Det er ikke noe tydelig definert skille mellom koding og programmering, men Kaufmann et al. (2018) viser til at slik begrepene brukes, kan koding knyttes til utviklingen av mindre og mindre komplekse programmer mens programmering kan knyttes til mer kompliserte, mer omfattende programmer. Nygård (2018) sier på sin side at programmering er «en prosess som innebærer problemanalyse, abstraksjon og planlegging av hvordan koden skal struktureres for å få datamaskinen til å gjøre det vi ønsker.» (s. 8). Koding er derimot prosessen med å «skrive programkoden og utvikle programmer ved hjelp av et programmeringsspråk» (Nygård, 2018, s. 8). InterviewBit (2021) legger i tillegg vekt på kompleksiteten ved det å programmere og uttrykker at selve programmerings handlingen resulterer i et fungerende og komplekst system eller et program. Dette krever blant annet logisk tenkning, analysering og design ferdigheter. Det kan dermed

virke som at programmering handler om selve prosessen og det å få et program som fungerer til slutt, mens koding handler om de konkrete kodene for å kunne sette i gang selve prosessen. I denne oppgaven vil jeg ta i bruk begrepet programmering, da dette er begrepet som går igjen i offentlige dokumenter og styringsformer.

Enkeltheten i det som kalles koding, kan dermed se ut til å være en passende beskrivelse av den aktiviteten elevene gjør når de arbeidet med blokkprogrammering. På samme måte som i beskrivelsen av koding handler dette i stor grad om å sette sammen koder for deretter å få noe ut av disse kodene. En del av prosessen, altså selve programmeringen, kan se ut til å kunne forsvinne i prosessen med å utvikle et programmeringsspråk som er forenklet i den grad blokkprogrammeringen har blitt. I den forstand at en ønsker å unngå tekstprogrammeringens mer komplekse syntaks. Franklin et al. (2016) ser i sin forskning på overføringen av kunnskap om algoritmisk tenking mellom blokkprogrammering i Scratch og tekstprogrammering i C og Java. Artikkelen viser til noen viktige forskjeller mellom Scratch som blockprogrammeringsverktøy og tekstprogrammer. For eksempel viser det seg at variasjonen av blokker og mangelen på et konkret utgangspunkt for å starte programmene, gjør det vanskeligere å identifisere hvor en skal putte kodene og hvilke blokker en skal bruke. Dette skal være enklere i et tekstprogram. Overføringen av kunnskap fra blokkprogrammering til etter hvert å bruke tekstprogrammering kan dermed bli vanskelig, da dette ikke er tilfellet i tekstprogrammering. Ved at den komplekse syntaksen ved tekstprogrammene tas bort, samtidig som utgangspunktene for kodene endres. Til tross for eksplisitt undervisnings av overføringsverdien mellom Scratch og C, hadde elevene vansker for å se sammenhengen mellom prinsippene innen blokkprogrammering og tekstprogrammering (Franklin et al., 2016).



Figur 3.4: Scratch programmeringsfelt (Scratch Foundation, 2019)

3.3 Programmering og matematikk

I kap. 1.1 Offentlige føringer, nevnes rapporten til Ludvigsen-utvalget (NOU 2015:8). Rapporten i seg selv nevner ikke programmering, men den vektlegger fire kompetanseområder som bør vektlegges i skolen (NOU 2015:8). Disse fire er; 1) «fagspesifikk kompetanse», 2) «kompetanse i å lære», 3) «kompetanse i å kommunisere, samhandle og delta» og 4) «kompetanse i å utforske og skape» (NOU 2015:8, s. 22). Sevik og m.fl. (2016) viser hvordan disse fire kompetanseområdene kan utvikles ved å knytte programmering inn i arbeidet i skolen. Enten ved å implementere programmering i fagene som f.eks. matematikk eller ved tverrfaglig arbeid. Argumentet som brukes er at ved å arbeide med programmering i en matematikkfaglig sammenheng, kan man «(…) knytte matematiske begreper opp mot praktiske kontekster der elevene får en forståelse for kunnskap i en større sammenheng.» (Sevik & m.fl., 2016, s. 12). Det uttrykkes at det blant annet krever algoritmer, syntaks og logikk i et slikt arbeid. Henvisningen til algoritmer finner vi også igjen hos Sanne et al. (2016), som uttrykker at implementeringen av programmering i undervisningen, blir algoritmebegrepet stadig viktigere. Dette henger sammen med at arbeidet med datamaskinen åpner for å i større grad utnytte den «kreative, matematiske aktiviteten som ligger i det å utvikle algoritmer» (Sanne et al., 2016, s. 89). I tilknytning til kompetanseområdet om metakognisjon; Å lære seg å lære, viser Sevik og m.fl. (2016) til paralleller mellom dette og algoritmisk tenking. Denne sammenhengen handler om å gi elever verktøy for å kunne ta tak i

oppgaver og problemer. Programmeringen gir en oversiktlig mulighet for å arbeide med problemløsing.

Forskningen er splittet rundt det matematiske læringsutbyttet av arbeidet med programmering. Kilhamn et al. (2021) trekker frem fire argumenter som brukes av lærere i Sverige for å legitimere implementering av programmering i faget matematikk. I artikkelen er målet å kartlegge og å forstå lærernes kunnskap om hvorfor programmering er en implementert del av matematikkfaget. Argumentene som kommer til uttrykk er 1) utvikling av algoritmisk tenking (computational thinking), 2) økt engasjement, 3) å lære matematikk og 4) at det helt enket kan være «a powerful tool» (Kilhamn et al., 2021, s. 171). Med dette viser Kilhamn et al. (2021) til tre viktige matematiske komponenter. Den første er algoritmisk tenking, som har en nær sammenheng med problemløsingsstrategier. Den andre er å lære matematikk, uten at det blir utdypet ytterligere hvilken type matematikk de mener en kan lære av å arbeide med programmering. Den siste handler om å bruke programmering som verktøy for å arbeide med matematiske komponenter, og handler altså å om å kunne bruke programmering som metode og et verktøy for å arbeide med matematikk. Kilhamn et al. (2021) viser videre at lærerne uttrykte at programmering skulle fungere som et pedagogisk verktøy. Programmering kan enten være et verktøy for å lære matematikk eller så kan programmering ses på som matematikk i seg selv (Kilhamn et al., 2021). Liknende funn finner vi i artikkelen til Misfeldt et al. (2019) der ønsket blant lærerne var at elevene skulle lære matematikk ved bruk av programmering og at lærerne mente at programmering på sikt kan fungere som en metode for innlæring av matematiske konsepter. Videre ble det uttrykt at programmering kunne brukes som metode for å arbeid med problemløsende aktiviteter og for arbeid med algoritmisk tenking. Det kan ut fra Kilhamn et al. (2021) og Misfeldt et al. (2019) forskning, virke som at programmering åpner for et potensiale for å arbeide med matematikk, men denne forskningen er ikke entydig når det gjelder implementeringen av programmering.

Til tross for at lærerne er positive, er både Kilhamn et al. (2021); Misfeldt et al. (2019) forskning noe kritisk til implementeringen av programmering i matematikkundervisningen. Funnene deres viser at til tross for at lærerne mente det kunne fungere som et pedagogisk verktøy og var positive til å arbeide med programmering i matematikk, så lærerne lite sammenheng mellom de to komponentene (Misfeldt et al., 2019). Det er mulig at ideen om å kunne bruke programmering for å arbeide med matematikk er et tiltalende punkt, som en kan se for seg vil være et motiverende supplement i elevenes matematikkundervisning. Likevel ser

det ut til at utfordringen er at de ikke klarer å se hvilke matematikkfaglige komponenter en potensielt kan arbeide med eller hvordan en skal fremheve matematikken i arbeidet.

Slik situasjonen fremstår er det et ønske om at programmering skal kunne brukes som metode, men at det foreløpig kan se ut til at bruken av programmering er rettet mot det å lære seg å programmere. Det finner i tillegg heller lite forskning på området som trekker frem det faktiske utbyttet av bruken av programmering, ei heller forskning som vektlegger årsak-virkning-sammenhengen. Dette samsvarer med det Kilhamn et al. (2021) uttrykker når de skriver at "Little is yet known about what is taught and why, in the revival of programming in schools taking place in the last decade" (s. 170). Dette kan komme av at hverken lærerne eller elevene er trygge nok på programmering til å kunne bruke det slik det er intendert. Derfor blir et viktig argument, som potensielt også kan glede for implementeringen i Norge, at programmeringen vil få en annen rolle i undervisningskontekst etter hvert som programmeringsmetoden har satt seg bedre blant lærerne (Kilhamn et al., 2021). Dette støttes av Bueie (2019) som uttrykker behovet for gode programmeringskunnskaper for at en skal kunne bruke programmering i matematikkfaget.

Et av de mest fremtredende argumentene som går igjen er argumentet om økt motivasjon for å arbeide med faget matematikk (Calao, Moreno-Leòn, Correa & Robles, 2015; Forsstøm & Kaufmann, 2018; Kaufmann et al., 2018; Kilhamn et al., 2021). Argumentet tar utgangspunkt i at programmering gir en ny vinking til fagstoffet som igjen gjør at elevene blir mer motiverte for å arbeide med oppgaver i undervisningen. I tillegg er tanken at elevene opplever innholdet som nyttig, ved at det matematikkfaglige innholdet får en kontekst. Det vises til både bedre prestasjoner innen matematikk og større grad av motivasjon for å arbeide med matematikk (Forsstøm & Kaufmann, 2018). Det skal likevel påpekes at det ikke nødvendigvis er slik at elevene lærer matematikk direkte gjennom programmering, men ideen er at økt motivasjon vil medføre økt engasjement for faget, og dermed også økt læringsutbytte i matematikk (Kilhamn et al., 2021). Dette betyr at til tross for at noen mener at det faglige læringsutbyttet i seg selv ikke er veldig høyt, vil motivasjonen og engasjementet det medfører for faget føre til at det matematiske læringsutbyttet likevel øker på bakgrunn av motivasjonen.

Selv om motivasjon er et viktig argument for å implementere programmering i matematikkundervisningen, finnes det også flere matematikkfaglige argumenter for å implementere programmering i matematikkfaget. Dette gjør at spørsmålet som Kaufmann et al. (2018) stiller «(…) om man skal studere matematikk ved hjelp av programmering, eller om man skal studere programmering med hjelp av matematikk» (s. 81) blir relevant vurdere når

en velger å implementere programmering i faget matematikk. Det handler ikke om hvorvidt matematikk er sentralt i arbeidet med programmering, men heller hvilken rolle programmeringen bør ha i arbeidet med matematikk.

Scratch er, som nevnt over, et av programmeringsspråkene som ofte tas i bruk i grunnskolen. Ifølge Taylor, Harlow og Forret (2010) vil ikke programmet Scratch i seg selv bidrar til læring innen et spesifikt matematisk område, men programmet skaper muligheter for at elever kan bruke matematikken de har lært eller kan fra før på utallige ulike måter. På den måten kan forståelsen for det matematikken utfolde seg på ulike vis avhengig av hvordan elevene velger å besvare en oppgave. Det kan dermed virke som at det er enighet i at matematikk kan være sentralt i arbeidet med programmering. Samtidig som det ikke er helt enighet om programmeringens rolle. Det kan i Taylor et al. (2010) tilfelle virke som at tanken er at Scratch funger som et program for å utforske eller finne nye måter å bruke allerede kjent matematisk kunnskap, men at programmet er mindre egnet for å lære nye konsepter i matematikk. På en annen side viser Calao et al. (2015) til en signifikant økning av matematikkunnskap blant deltakere i forsøksgruppen sin, der arbeidet foregikk i Scratch. Calder (2010) og Lambi`c (2010) gjennomførte på hver sin side prosjekter som resulterte i at flere elever fikk en forståelse for hvordan matematikk kunne brukes. Det å forstå hvordan man kan bruke matematikk, kan tolkes som en utvikling av matematisk kunnskap. Calder (2010) uttrykker videre at de aller fleste oppgavene elevene arbeidet med i Scratch involverer en eller annen form for utvikling av matematiske konsepter.

Scherer, Siddiq og Viveros (2019) forsket på hvordan arbeidet med programmering og ferdigheter som kreves for å klare dette, kan overføres til andre situasjoner. Her konkluderte de med at ferdigheter som kreativ tenking, matematiske ferdigheter, matakognisjon og logisk tenking ble styrket på generelt grunnlag av å arbeide med programmering. Kildene nevnt over, gir inntrykk av at det bør være et matematisk læringspotensial i arbeidet med programmering. Likevel er kildene mindre spesifikke i sine konklusjoner. For eksempel skriver Scherer et al. (2019) lite om hvilke konkrete matematiske ferdigheter som ble styrket.

Bueie (2019) skriver i sin bok *programmering for matematikklærere* at «I forbindelse med fagfornyelsen får matematikkfaget en sentral rolle i opplæring i programmering. Dette er naturlig ettersom programmering henger svært tett sammen med matematikk faglig og på mange måter har sitt utspring fra matematikkfaget.» (s. 23). Kaufmann et al. (2018) knytter på sin side programmering til informatikkfaget og poengterer at informatikken har rot i matematikken. På denne måten skaper de begge en sammenheng mellom matematikken som

fag og programmering. Kilhamn et al. (2021) viser til koblingen mellom programmering og matematikk og sier at programmering er en matematisk aktivitet i seg selv.

3.3.1 Fremgangsmetoder for implementering i matematikk

For at programmering skal kunne finne sin plass i matematikkfaget, må dette implementeres på en måte som gagner både matematikken og programmeringen. Det er flere som har sett på ulike måter dette kan gjøres. Misfeldt og Ejsing-Duun (2015) viser i sin artikkel *Learning mathematics through programming: An instrumental approach to potentials and pitfalls* til en modell for ulike tilnærminger for implementering av programmering i matematikk. Den første tilnærmingen tar utgangspunkt i eleven som produserer kunnskap og tar utgangspunkt i Seymour Paperts LOGO og «Turtle-programming», samt ideen om konstruktivisme. Den andre tar utgangspunkt i å støtte abstrakt tenking og i radikal konstruktivisme der en mener at all abstrakt læring har et konkret utgangspunkt. Denne tar også utgangspunkt i det Richard R. Skemp kalte for relasjonell forståelse, som handler om å forstå sammenhengene og relasjonene i matematikken. (Skemp, 1976). Den tredje og siste tilnærmingen tar utgangspunkt i å utvikle algoritmisk tenking (algorithmic thinking). Der eleven får tilgang til en digital verden med konstruksjon av figurer gjennom bruken av matematikk og tar utgangspunkt i å beskrive elevers ferdigheter i arbeidet med algoritmer. Algoritmer kan her forstås som systematisk beskrivelse av problemløsing og konstruksjon av strategier. Det handler i den tredje tilnærmingen altså om en årsak-virkning-effekt. Til tross for at disse tilnærmingene ikke alltid har hatt den forutsette effekten en skulle ønske, finner vi likevel muligheter for implementering av programmering som et middel i matematikkfaget, med et læringsutbytte som kan knyttes til nettopp matematikk.

Clements og Merdith (1992) viser til ulike måter Logo programmering kan brukes i matematikk, og trekker frem to ulike fremgangsmetoder. Den første metoden handlet om å bruke LOGO som et medium for å arbeide med tradisjonell matematikk. Den andre handlet om å endre tradisjonelle aktiviteter til å bli mer «higher-level thinking» (Clements & Merdith, 1992, s. 5). Dette kan forstås som at tradisjonelle aktiviteter må omformuleres til problemløsende aktiviteter, og kan igjen knyttes til algoritmisk tenking. Som handler om nettopp hvordan en skal ta tak i et problem og hvordan en skal gå frem for å løse denne. Måtene en på 90-tallet kunne arbeide med LOGO-programmering på er fortsatt svært relevant for programmeringsarbeidet som skjer i skolen i dag, dette fordi mye av programmeringsarbeidet bygger på programmeringsverktøyet Seymour Papert utviklet på 1960-tallet. Det er derfor nærliggende å tenke at arbeidet med programmering i dag kan ta to

ulike utgangspunkt i matematikkfaget. Enten som metode for å arbeide med tradisjonell matematikk i ny drakt eller som et problemløsingsverktøy der elevene kan arbeide med oppgaver som krever «Higher-level thinking».

Så langt kan det virke som at det finnes et generelt matematisk læringspotensial ved å arbeide med programmering. Det nevnes et matematisk utbytte og konsepter i litteraturen, som forskningen legger vekt på at kommer til uttrykk i arbeidet elevene gjør når de programmerer. Kaufmann et al. (2018) uttrykker at programmeringen bør være et supplement i den ordinære undervisningen, og dermed heller komme i tillegg til de metodene vi bruker i matematikkundervisningen i dag. På denne måten vektlegger Kaufmann et al. (2018) at programmering implementert i matematikkfaget ikke skal gjøre om på matematikkfaget og hvordan vi arbeider med dette, men også at programmering er ment som en metode for å arbeide med konsepter elevene potensielt er noe kjent med fra før. Dette er viktig å holde i mente når en videre skal se på det mer konkrete matematiske læringspotensialet i arbeidet med programmering. Men før dette bør det avklares hva som menes med matematisk kunnskap.

3.4 Matematisk kompetanse

For å definere hva som regnes som et matematisk læringspotensial, blir det viktig å kunne definere hva som regnes som matematisk kunnskap. Jeg har i denne oppgaven valgt å ta i bruk betegnelsen kompetanse fremfor kunnskap. Kunnskaper handler i stor grad om både å lære seg, men også å kunne ta i bruk teori og begreper, samt å se sammenhenger, mens ferdigheter handler om handlingene som behøves for å kunne løse en oppgave eller et problem (Kunnskapsdepartementet, 2017). I LK20 blir det forklart at «Kompetanse er å kunne tilegne seg og anvende kunnskaper og ferdigheter til å mestre utfordringer og løse oppgaver i kjente og ukjente sammenhenger og situasjoner. Kompetanse innebærer forståelse og evne til refleksjon og kritisk tenkning» (Kunnskapsdepartementet, 2017, s. 10). Dermed dekker begrepet kompetanse både ferdigheter og kunnskaper som kan sees i sammenheng med matematikkfaget. Matematikk i seg selv er et vidt begrep som innebærer mange ulike former for ferdigheter og kunnskap. Med dette til grunn vil begrepet kompetanse best dekke definisjonen for det som kan regnes som et matematisk læringspotensial.

Jeg har valgt å ta utgangspunkt i Niss et al. (2002) sitt rammeverk for matematisk kompetanse. Denne består av ulike kategorier for kompetanser i matematikk. Disse kompetansene når bredt, men er likevel spesifisert og definert. Overordnet er kompetansene

delt inn i to hovedkategorier; «Å kunne spørre og svare i og med matematikk» (Niss et al., 2002, s. 44) og «Å kunne håndtere matematikkens språk og redskaper» (Niss et al., 2002, s. 44). Det er en sammenheng mellom de ulike komponentene på tvers av disse to gruppene, og dermed også en sammenheng mellom de to hovedgruppene. Inn under den førstnevnte finner vi «Tankegangskompetanse», «Problemløsingskompetanse», «Modelleringskompetanse» og «Resoneringskompetanse». Inn under den andre kategorien finner vi «Representasjonskompetanse», «Symbol- og formalismekompetanse», «Kommunikasjonskompetanse» og «hjelpemiddelkompetanse». Disse kompetansene går fra det konkrete, som innebærer håndtering av de matematiske symbolene og bruk av den formelle matematikken, til det mer abstrakte som logisk tenking innen problemløsing og tankegangskompetanse. Videre handler det om å kunne bruke og formidle tanker, ideer og begrunnelser. Disse blir forklart i korte trekk i avsnittene under. Tilslutt vil jeg trekke paralleller til læreplanen for matematikk i LK20.

3.4.1 Tankegangskompetanse

Tankegangskompetanse handler om spørsmål som er relevante for matematikken. Det handler om å kunne stille de riktige spørsmålene og vite hvilke spørsmål som kan stilles, men også hvilke svar en kan forvente å få på denne type spørsmål. I tillegg til dette handler det om å kjenne til og kunne bruke matematiske begreper og forstå omfanget av de ulike begrepene. En må også forstå hva som ligger i å kunne generalisere løsninger samt skille mellom utsagn og påstander (Niss et al., 2002).

3.4.2 Problemløsingskompetanse

Problemløsingskompetanse handler om å kunne ta tak i et matematisk problem. En skal kunne formulere problemet, avgrense det og å kunne utforme en strategi for å løse problemet. Her er det viktig å påpeke at Niss et al. (2002) ser på begrepet «problem» som relativt, men at han her snakker om problemer som oppgaver som ikke kan løses ved bruk av rutinerte regneoperasjoner alene i seg selv. Ut over dette vil et matematisk problem avgjøres eller defineres som et problem avhengig av hvem som står ovenfor problemet (Niss et al., 2002).

3.4.3 Modelleringskompetanse

Å ha kompetanse om modeller brukt innen matematikk handler om mer enn kun å kunne anvende ulike modeller og å skape eller bygge modeller selv. Det handler i tillegg om å kunne analysere og vurdere egenskaper ved ulike modeller. Dette betyr at når en har bygd en modell skal en også kunne vurdere holdbarheten av denne for å visualisere et matematisk fenomen.

Det handler om å ha et kritisk blikk på selve modellen, det den skal fremvise og hvordan modellen skal tas i bruk (Niss et al., 2002).

3.4.4 Resonneringskompetanse

Resonneringskompetanse handler om både å følge og å vurdere matematiske resonnement. Dette gjøres ved å følge andres argumenter og bevis, vurdere disse og å legge frem egne argumenter og bevis. En må forstå hva et bevis er, hvordan dette kan legges frem og vurdere gyldigheten av dette (Niss et al., 2002).

3.4.5 Representasjonskompetanse

Representasjonskompetanse handler om at man har kjennskap til, kan bruke og forstår hvordan ulike matematiske komponenter, problemer, situasjoner eller likende kan representeres. Dette kan være representasjoner i form av blant annet helkonkreter, symboler, geometriske eller grafiske fremstillinger og liknende. I tillegg til dette innebærer denne kompetansen at en kan se sammenhenger mellom ulike representasjonsformer og kjenner til styrker og svakheter til de ulike. En må derfor kunne velge den best egnede representasjonen i en gitt situasjon, samt kunne gjøre om fra en representasjonsform til en annen(Niss et al., 2002).

3.4.6 Symbol- og formalkompetanse

Symbol- og formalkompetanse handler om å både å kunne avkode symboler og formler i matematikk og å kunne oversette det til et naturlig språk. En skal både kunne bruke det selv og kunne behandle og gjøre om på formler og symboler. Det handler om å bruke og kjenne til den formelle matematikkens spilleregler og bruk av symboler(Niss et al., 2002).

3.4.7 Kommunikasjonskompetanse

Kommunikasjonskompetanse handler om kommunikasjonen i samspill med andre. Det handler videre om å kunne sette seg inn i og tolke andres utsagn, i tillegg til å kunne uttrykke seg selv på ulike måter og på ulike vanskelighetsnivåer. Kommunikasjon handler ikke kun om den muntlige fremtoningen, men også skriftlig og visuell fremtoning (Niss et al., 2002).

3.4.8 Hjelpemiddelkompetanse

Hjelpemiddelkompetanse handler om å kjenne til og kunne bruke ulike hjelpemidler og verktøy knyttet til matematikk. Det handler også om å kunne kjenne til styrker og svakheter ved de ulike verktøyene (Niss et al., 2002).

3.4.9 Matematisk kompetanse og LK20

Niss et al. (2002) påpeker at flere av kunnskapsområdene flyter over i hverandre. For eksempel er representasjon-, kommunikasjon- og symbol- og formalkompetanse nært relatert, men vektlegger ulike elementer av kompetansen. Dette gjelder videre også for Tankegangs-, resonnement- og problemløsingskompetanse, der spørsmål, fremover tenkning og løsningsforslag ligger som elementer som er relevante for alle de tre kompetanseområdene. Modelleringskompetanse og representasjonskompetanse er så nært relatert at de kan være vanskelige å holde adskilt. Begge kan bidra til å kunne behandle et problem. Kompetanseområdene tar sikte på både den undersøkende delen av matematikken, men også den produktive delen av matematikken (Niss et al., 2002). Den produktive delen av matematikken kan se ut til å komme til uttrykk gjennom gjennomføring av de ulike kompetansene, mens den undersøkende delen handler om å kunne analysere, forstå og være kritisk(Niss et al., 2002).

I LK20 finner vi flere paralleller til disse kompetansene. Vi ser at blant kjerneelementene kan vi finne igjen både problemløsing, modellering, resonnering, representasjoner og kommunikasjon. I tillegg til disse finner vi også abstraksjon og generalisering. Selv om disse ikke konkret nevnes i Niss et al. (2002) sine kompetanseområder kan vi likevel knytte dem opp til flere av områdene. Abstraksjon handler i stor grad om å bevege seg fra det konkrete til det mer abstrakte og formelle i matematikken. Både språket og uttrykksformene, strategier og tanker. Denne kan indirekte kobles til det Niss kaller for tankegangskompetanse. Også her handler det om å stille de riktige spørsmålene, og ha kjennskap til hvilke svar en kan forvente.

Videre finner vi i LK20 det de kaller for «Matematisk kunnskapsområder». Dette er områder som tar for seg ulike og spesifikke matematiske konsepter som for eksempel tallforståelse, algebra, funksjoner, statistikk, sannsynlighet og geometri (Kunnskapsdepartementet, 2019). Til dette knyttes også en begrepsforståelse og en bruksforståelse av konseptene. Det heter at «Kunnskapsområdene danner grunnlaget som elevene trenger for å utvikle matematisk forståelse ved å utforske sammenhenger innenfor og mellom kunnskapsområdene.» (Kunnskapsdepartementet, 2019, s. 3-4). Disse kunnskapsområdene kan arbeides med gjennom arbeidet med de overstående kunnskapsområdene. På den måten kan en nesten se på de første kompetanseområdene som overordnet, og noe uavhengig av de spesifikke områdene nevnt i det siste avsnittet. De spesifikke kunnskapsområdene, samt de overordnede kommer videre til uttrykk gjennom LK20 kompetansemål i læreplanen, både direkte i form av at de

nevnes spesifikt, og indirekte gjennom valg av verb som «utforske», «forklare», «beskrive» og «diskutere», «løse», «lage» og «utvikle» (Kunnskapsdepartementet, 2019, s. 7-11).

Disse 8 kompetanseområdene til Niss et al. (2002) og kjerneelementene i LK20 ligger til grunn for hva som i denne oppgaven kan regnes som matematisk læringsutbytte eller læringspotensial. Videre viser sammenlikningen med Læreplanen LK20 at disse kunnskapsområdene er svært relevante for fagfornyelsen og arbeidet med denne. Når jeg nå videre i oppgaven analyserer hvilket læringspotensial litteraturen trekker frem, velger jeg å se litteraturen i lys av kunnskapsområdene beskrevet over.

3.5 Matematisk Læringspotensial

I dette delkapittelet vil jeg se på hvilket matematisk læringspotensial som ligger i arbeidet med programmering. Jeg har valgt å bruke begrepet «læringspotensial», fordi dette best beskriver muligheten for å ta i bruk programmering i en matematikk sammenheng. Begrepet åpner for at det er en mulighet for at læring kan skje, men også at det ikke nødvendigvis finner sted.

Først vil jeg ta for meg de konsepter som kan regnes som overordnede konsepter. Deretter vil jeg tar for meg mer spesifikke konsepter. Jeg har valgt å skille disse konseptene, men det er viktig å presisere at ingen av disse konseptene fungerer for seg selv. De henger sammen på ulike vis, og det vil i enkelte tilfeller krysse hverandres avsnitt. På samme måte som Niss et al. (2002) sine 8 kompetanser.

3.5.1 Overordnede konsepter

I denne delen vil jeg legge frem de konseptene som kan regnes som overordnende konsepter. Konseptene regnes som overordnet fordi de de kan knyttes til en overordnet del av matematikkfaget, og kan finnes igjen under kjerneelementer i læreplanen samt at de kan regnes som mer omfattede i sin definisjon (Kunnskapsdepartementet, 2019).

3.5.1.1 Algoritmisk tenking

En del av arbeidet med programmering i skolen skal ha som mål å arbeide med utviklingen av den algoritmiske tenkingen. Dermed blir begrepet algoritmisk tenking eller «computational thinking» viktig for å kunne sette ord på et tankesett som skal brukes i arbeidet med programmering og matematikk. I engelskspråklig litteratur har en tatt i bruk begrepet «computational thinking», dette begrepet har vi på norsk oversatt til «algoritmisk tenking».

Begrepet algoritmisk tenking er et stort, omfattende begrep som ved første introduksjon høres kort og ryddig ut, uten at en skal la seg lure av dette.

Kort forklart kan algoritmisk tenking sees på som en strukturert måte å løse et problem, samtidig som det innebærer det å kunne representere løsninger og å kunne vurdere løsningsstrategier. Dette gjøres ved å dele et problem opp i mindre deler for deretter å strukturert arbeide seg gjennom hvert delproblem frem til det overordnede problemet er løst. Denne måten å tenke på baserer seg på måten vi tenker når vi går frem for å løse en algoritme. I stedet for å starte rett på algoritmen som en helhet, deles den opp og vi regner delkomponenter av algoritmen før vi sitter igjen med svaret. Likevel kan ikke algoritmisk tenking kun defineres ved å introdusere det som en strategi for løsning og oppdeling av et problem. Det er et samlebegrep for flere ferdigheter knyttet til problemløsing. Dette skal jeg utdype videre.

Algoritmisk tenking skal ikke forveksles med «algorithmic thinking» selv om dette er nærliggende. «Algorithmic thinking» er en underkategori eller en ferdighet innenfor det overordnede «Computational thinking» (Angeli & Giannakos, 2020; Clarke-Midura, Silvis, Shumway, Lee & Kozlowski, 2021). «Algorithmic thinking» handler om den delen av «computational thinking» som omhandler det å kunne løse problemer som en trinnvis prosess (McVeigh-Murphy, 2019).

Bueie (2019) legger vekt på at vi ikke kan se på programmering i matematikkfaget uten å også inkludere algoritmisk tenking og uttrykker at «I debatten rundt programmering i matematikkfaget har det kommet tydelig frem at programmering i matematikkfaget skal sees i sammenheng med algoritmisk tenkning.» (s. 19). Likevel er det også viktig å se at det er et skille mellom programmering og algoritmisk tenkning. Først og fremst ved å være klar over at algoritmisk tenking ikke er det samme som programmering (Wing, 2006), men at ferdigheter innen programmering vil være en av fordelene ved å kunne tenke algoritmisk (Shute, Sun & Asbell-Clarke, 2017). Dette kan forklares ved at tankesettet vi kaller algoritmisk tenking kommer til sin rett når en arbeider med programmeringsoppgaver. En får altså bruk for ferdighetene innen algoritmisk tenking når en arbeider med programmering. Resultater av en undersøkelse gjennomført av Shute et al. (2017) viste at ferdigheter som arbeidsminne, identifisering av problemer og å strukturere løsninger er tegn på gode programmeringsferdigheter, men også ferdigheter vi finner igjen innenfor algoritmisk tenking. De konkluderte dermed med at bade algoritmisk tenking, programmeringsferdigheter og problemløsing henger tett sammen (Shute et al., 2017). Det er derfor ikke til å stikke under en

stol at en i arbeidet med programmering og matematikk, også må nevne og inkludere algoritmisk tenking.

Begrepet algoritmisk tenking er på ingen måte noe nytt begrep, i likhet med introduksjonen av programmering og «turtle-programming» var Seymour Papert også tidlig ute med å introdusere begrepet «computational thinking». Innholdet i dette ble allerede på 1980-tallet fremmet av Seymour Papert som en grunnleggende ferdighet som bør ligge til grunn for utviklingen av den problemløsende tenkingen (Papert, 1980; Sanne et al., 2016). I senere tid har Wing (2006) fremmet algoritmisk tenking som «a fundamental skill for everyone…» (s. 33). Videre argumenterer hun for at «computational thinking is using heuristic reasoning to discover a solution» (Wing, 2006, s. 34). Wing (2006)skaper dermed en kobling mellom matematikkfagets heuristiske tankesett og algoritmisk tenking. Algoritmisk tenking handler i stor grad om å omformulere utfordrende problemstillinger til problemstillinger med en kjent struktur med kjente løsningsstrategier, som dermed gjør at vi vet hvordan problemet kan løses. Dette gjøres ved å både bruke abstraksjon, bryte ned problemet, samt velge passende representasjonsformer eller å modellere relevante aspekter ved problemet. På denne måten bidrar den algoritmiske tenkingen med at en kan løse komplekse problemstillinger samt bruke, modifisere og påvirke komplekse systemer uten å nødvendigvis forstå alle detaljene i problemstillingen eller systemet (Wing, 2006).

Allerede i de innledende avsnittene kan en få inntrykk av at algoritmisk tenking er et vidt begrep. Kort oppsummert er det et problemløsende tankesett, men i realiteten er det veldig mye mer komplekst og vidt begrep. I sitt forskningsprosjekt, trekker Weintrop et al. (2015) frem ti ferdigheter innen algoritmisk tenking. Disse ti er; evnen til å håndtere åpne oppgaver, utholdenhet i arbeidet med utfordrende oppgaver, selvtillit, evnen til å kunne representere ideer på en meningsfull måte, dele store problemer inn i mindre mer oversiktlige delproblemer, utvikle abstraksjon, omformulere problemstillinger slik at de likner kjente problemer, kjenne til styrker og svakheter ved en representasjons form, utvikle algoritmiske løsninger og kjenne igjen feil i algoritmer (Weintrop et al., 2015). Haraldsrud et al. (2020) viser til algoritmisk tenking som en sammensatt prosess i en programmeringssammenheng. Der problemet deles opp i mindre deler og deretter søkes mulige løsningsforslag før en går i gang med selve programmeringen. Det innebærer å se på problemløsing gjennom sentrale strategier som for eksempel «systematisering, analyse, gruppering, abstraksjon og evaluering» (Haraldsrud et al., 2020, s. 189). Bueie (2019) på sin side knytter begrepet til matematikkens algoritmer, og beskriver det som en «ordnet liste med instrukser som er laget for å finne et

svar på et bestemt problem» (Bueie, 2019, s. 28). Nygård (2018) viser til algoritmisk tenkning som en systematisk måte å ta tak i et problem. Det handler om å finne og gjenkjenne mønstre for deretter å generalisere disse gjennom abstraksjon, for deretter å videre effektivt kunne bruke disse til å løse problemet.

Shute et al. (2017) mener det er mangel på anerkjente definisjoner, og prøver å definere hva algoritmisk tenking egentlig er. De definerer «computational thinking» som en grunnleggende ferdighet en trenger for å effektiv kunne løse problemer (Shute et al., 2017). Disse kan løses både med og uten bruken av digitale hjelpemidler. Videre ser de også på likhetene mellom matematisk tenking og algoritmisk tenkning, og finner flere likhetstrekk, blant annet problemløsing, modellering, analyse av data, statistikk og sannsynlighet (Shute et al., 2017). Disse konseptene finner vi også igjen hos Wing (2006) som sier at algoritmisk tenking innebærer «solving problems, designing systems and understanding human behavior» (s. 33). Sevik og m.fl. (2016) definerer algoritmisk tenkning som en «problemløsningsprosess som innebærer å tenke som en informatiker når man skal løse et problem» (s. 13).

Det som går igjen hos flere er at algoritmisk tenking innebærer å dele opp et større problem i mindre delproblemer, kunne kjenne til flere løsningsforslag og representasjonsformer samt å kunne arbeide med ulike nivåer av abstraksjon. Algoritmisk tenkning blir derfor en måte å tenke på for å kunne løse et problem eller en oppgave. Dette gjør man ved å bryte et større problem ned i mindre, mer oversiktlige problemer. Dette er en definisjon som Utdanningsdirektoratet (2019a) støtter seg på

> *«Å tenke algoritmisk er å vurdere hvilke steg som skal til for å løse et problem, og å kunne bruke sin teknologiske kompetanse for å få en datamaskin til å løse (deler av) problemet. I dette ligger også en forståelse av hva slags problemer/oppgaver som kan løses med teknologi og hva som bør overlates til mennesker.» (Utdanningsdirektoratet, 2019a, avsn. 1).*

Weintrop et al. (2015) har skapt et rammeverk som kan brukes for å definere hva computational thinking er. Målet med rammeverket er å skape et klassifiseringssystem for algoritmisk tekning som fokuserer på hvordan algoritmisk tenking kan implementeres i realfag som matematikk. Rammeverket til Weintrop et al. (2015) deles inn i 22 underkategorier som igjen er delt inn i fire hovedkategorier. Hovedkategoriene "data practices", "modeling and simulation practices", "computational problem solving practices" og "system thinking practices" kan beskrives ved å se på underkategoriene (Weintrop et al.,

2015, s. 135). Ved å se på «Data Practices» kan vi finne innsamling, manipulering, analysering og visualisering av innsamlet data. Videre finner vi det å bruke modeller for å forstå, prøve ut løsninger og designe og konstruere modeller under «Modeling and Simulation Practices». Den største variasjonen i innhold finner vi under «computational Problem Solving Practices». Denne innebærer blant annet å forberede problemer for algoritmiske løsninger, programmering, valg av effektive verktøy, ta i bruk ulike løsningsstrategier for et problem, feilsøking og «Debugging» (Weintrop et al., 2015, s. 140). «Systems thinking practices» innebærer på sin side å kunne arbeide med komplekse systemer som en helhet, forstå sammenhenger innad i et system, tenke på systemet på ulike nivåer, formidle informasjon om et system og definere systemer og håndtere kompleksitet.

Som nevnt over er ikke algoritmisk tenking synonymt med programmering, og det krever ikke nødvendigvis digitale hjelpemidler å kunne arbeide med algoritmisk tenking. Dette er et tankesett, en måte å løse problemer. Shute et al. (2017) forskningen viser at elever som prøver programmering med penn-og-papir viser signifikant bedre forståelse for algoritmisk tenking, enn de som lærer gjennom ulike digitale programmer. Det argumenteres derfor for arbeid med algoritmisk tenking uten digitale midler som PC. Dermed vektlegges igjen at Algoritmisk tenking handler mer om den problemløsende tankegangen enn digitale programmerings ferdigheter. Algoritmisk tenking som problemløsings tankegang og digitale programmerings ferdigheter kan dermed også fungere uavhengig av hverandre i den forstand at algoritmisk tenking kan være et problemløsende tankesett som kan brukes for seg selv, men en behøver algoritmisk tenking for å kunne programmere.

Når vi ser på algoritmisk tenking og ferdighetene dette innebærer i lys av læreplanen, kan vi finne paralleller til læreplanens kjerneelementer (Kunnskapsdepartementet, 2019). Blant disse finner vi «Utforsking og problemløsing», «Modellering», «Resonnering», «Representasjon» og «abstraksjon». Alle disse punktene blir benyttet for å forklare hva begrepet algoritmisk tenking innebærer. Kjerneelementene blir regnet som det «viktigste faglige innholdet elevene skal lære» (Utdanningsdirektoratet, 2019b, s. 1). Likhetstrekkene mellom kjerneelementene og ferdighetene som knyttes opp mot begrepet algoritmisk tenking, kan vise til omfanget, men også viktigheten av å ta høyde for og å implementere algoritmisk tenking som en del av undervisningen. Læreplanen og kompetansemålene ser ut til å legge vekt på flere av Weintrop et al. (2015) kategorier, men med noe ulik vekt. På mellomtrinnet kan se ut til å legges størst vekt på «data practices» og «Computational problem solving practices». Disse legger vekt på ferdigheter som «Assessing different approaches/solutions to a problem», «Creating

computational Abstractions» «trouble shooting and debugging", "manipulating Data" and "analysing data"(Weintrop et al., 2015).

Det kan se ut til at arbeidet med algoritmisk tenking kommer til sin rett gjennom arbeidet med kjerneelementene, og dermed også matematisk kompetanse. Der utforsking og problemløsing står øverst. Gjennom begreper som «problemløsing», «modellering», «resonnering», «representasjon» og «abstraksjon» kan en se likhetstrekk mellom det Niss et al. (2002) regner som matematisk kompetanse og innholdet i begrepet algoritmisk tenking. Likevel viser det seg at det å skape en sammenheng mellom algoritmisk tenking og den videre læreplanen i gjeldende fag kan se ut til å være en utfordring for lærere, da lærere generelt har lite kjennskap til algoritmisk tenking (Shute et al., 2017). I tillegg vises det til at elever som arbeider med spilldesign og programmering ikke nødvendigvis overfører kunnskapen de innehar om algoritmisk tenkning til andre områder som f.eks. matematikkfaget (Nygård, 2018; Shute et al., 2017). I likhet med andre aspekter ved matematikken er det mulig at dette henger sammen med bevisstgjøring av sammenhenger mellom konsepter. Det vil si at dersom lærere til liten grad har kjennskap til algoritmisk tenking vil det å kunne se sammenhengen til matematikkfaget selv også bli en utfordring. Dersom dette ikke gjøres eksplisitt for elever, er det stor sannsynlighet for at denne overføringen av kunnskaper som det algoritmiske tankesettet ikke nødvendigvis overføres til matematikken.

I matematikk skal elevene være problemløsere, og programmering som verktøy kan gi elevene muligheten til å arbeide med virkelighetsnære problemstillinger som åpner for ulike strategier og verktøy. Elevene kan eksperimentere og prøve og feile ved at de får umiddelbare tilbakemeldinger. Samtidig som elevene kan finne sin egen vei inn i matematikken i sitt eget tempo. Det gjør de ved at de kan løse problemer og oppgaver gjennom ulike tilnærminger. Det siste argumentet knytter programmering til dybdelæring i faget matematikk. Dybdelæring beskrives som «(…) å anvende kunnskaper og ferdigheter på ulike måter, slik at elevene over tid kan mestre ulike typer faglige utfordringer individuelt og i samspill med andre.» (Kunnskapsdepartementet, 2017, s. 11). Mange problemer kan løses både med og uten programmering, og å kunne mestre begge strategiene innebærer en dypere forståelse av matematikken (Utdanningsdirektoratet, 2021d).

Weintrop et al. (2015) viser til relevansen til arbeidslivet og at digitale midler som f.eks. programmering bidrar til et realistisk syn på arbeidslivet og de arbeidsoppgavene de kan komme til å stå ovenfor. I tillegg støttes tanken om dybdelæring dersom bruken er nøye gjennomtenkt og planlagt, samtidig som det motsatte også gjelder; matematikken skaper en

meningsfull kontekst for, og problemer som passer godt til bruken av algoritmisk tenkning (Weintrop et al., 2015). Det er likevel slik at for at implementeringen skal ha nytte som et dybdelæringsverktøy, blir det viktig at elevene selv utvikler og bruker programmene. Det er da det vil oppstå læring (Utdanningsdirektoratet, 2021c).

3.5.1.2 Problemløsing

Problemløsing er et konsept som trekkes frem i læreplanen, samt som en av de 8 matematiske kompetansene (Kunnskapsdepartementet, 2019; Niss et al., 2002). Denne har i flere tilfeller en direkte kobling til algoritmisk tenking, samt til programmering. Det heter under de digitale ferdighetene i læreplanen at «Digitale ferdigheter i matematikk innebærer å kunne bruke (…) programmering til å utforske og løse matematiske problemer.» (Utdanningsdirektoratet, 2019c, s. 5). Her med en direkte kobling mellom matematikkens problemløsing og programmering som verktøy. Kilhamn et al. (2021) viser til at elevene i arbeidet med programmering lærer å bryte ned problemstillinger i mindre problemer, og på denne måten gjøre problemene mer oversiktlige. Dette er i tråd med ideen om algoritmisk tenkning og programmering åpner for nye måter å jobbe med ulike oppgaver på; «Programmering gjør store oppgaver mindre og umulige oppgaver mulig» (Haraldsrud et al., 2020, s. 162).

Å arbeide med programmering kan påvirke hvordan elevene arbeider kognitivt. Dette bekrefter Calder (2010) og Kaufmann et al. (2018) ved å vise til at det kan føre til en utvikling av elevenes logiske tankegang og evnen til problemløsing. Arbeidet med programmering og digitale programmer gjør at det blir enklere å prøve seg frem uten at det får noen særlige konsekvenser (Haraldsrud et al., 2020). Dermed vil ikke redselen for å gjøre feil være et hinder for utforsking av ulike løsningsstrategier.

I tillegg finner vi den utforskende, åpne struktureringen igjen i oppgavedesignet for oppgavene en arbeider med i arbeidet med programmering. Taylor et al. (2010) viser til hvordan oppgavene elevene fikk i Scratch var designet slik at «Both children were able to take advantage of the “low” (easy to get started) and high Ceiling” (Potential for complex, sophisticated outcomes) attributes embed Scratch” (Taylor et al., 2010, s. 567). Grover og Pea (2013) bruker en lignende forklaring når de utdyper den type oppgaver som legger til rette for arbeid med algoritmisk tenking. Dette kan igjen knyttes til det vi i Norge kaller for LIST-oppgaver. LIST-oppgaver er oppgaver med lav inngangsterskel og stor takhøyde. Dette innebærer oppgaver som er forholdsvis lett tilgjengelige for de fleste elever, men som i tillegg kan utvides slik at den kan tilpasses elevenes behov i den forstand at elever får mulighet til å

gjøre besvarelsen sin så kompleks de selv ønsker eller får til (Matematikksenteret, 2017). LIST-oppgaver er ofte åpne oppgaver, og passer fint inn under kategorien problemløsende oppgaver. Dette passer godt inn i Seymour Paperts opprinnelige idè for hvordan han ønsket at arbeidet med programmering skulle foregå og hvilke muligheter ulike programmeringsspråk skulle ha (Resnick et al., 2009)

Resnick og Rosenbaum (2013) tar i bruk begrepet «Tinkerability» som direkte oversatt handler om ferdigheten å tenke. De definerer «Tinkering» som «(…) a valid and valuable style of working, characterized by playful, exploratory, interactive style of engaging with a problem or project" (Resnick & Rosenbaum, 2013, s. 164). Ved å "tinker" vil en person prøve nye ideer, tilpasse og justere ideer og løsninger, og eksperimentere med muligheter. Dette i en syklus av gjentakelser for å til slutt finne den beste løsningen. Denne formen for oppgaveløsing eller tankesett har en bottom-up tanke gang. Der en tar utgangspunkt i det materialet en har for hånd. Dette kan for eksempel være LEGO-klosser, et spill eller et programmeringsspråk. Etter hvert som en «leker seg» med programmet eller klossene og blir kjent med disse gjennom utforsking, vil det oppstå et mål med bruken av programmene.

Wing (2006) knyttet, som nevnt i kapittelet over, algoritmisk tenking til heuristikk. Schoenfeld (1985) beskriver dette som et problemløsings tankesett, som består av strategier og teknikker for å arbeide med ukjente, ikke-standardiserte problemløsingsoppgaver. Strategier som kan tas i bruk for å arbeide med denne type oppgaver er å tegne figurer, utforske relaterte problemer, Reformulere oppgaver og arbeide baklengs, samt teste og verifisere (Schoenfeld, 1985). Heuristisk problemløsing har flere likhetstrekk med algoritmisk tenking, og arbeidet med programmering som en problemløsende aktivitet (Wing, 2006). Deriblant bryte ned problemet til mindre problemer, teste og verifisere løsningsstrategier, skape tegninger gjennom programmene (modellere) samt bruke erfaringer fra arbeid med tidligere oppgaver eller relaterte oppgaver for å løse problemet. Feurzeig et al. (2011) viser til at erfaringer ved bruk av Logo kan bidra til læring av heuristiske aspekter ved matematikken.

LIST-oppgavenes åpne struktur, muligheten for «thinkering» i arbeidet med ulike oppgaver i et medium som programmering og sammenhengen mellom matematikkens heuristiske tenking og måten vi kan arbeide med programmering. Kan virke nært relatert til problemløsingskompetanse og tankegangskompetanse men også modelleringskompetanse, og representasjonskompetanse. Problemløsing, sammen med Algoritmisk tenking, fremstår dermed som en viktig overordnet komponent når en snakker om læringspotensialet i arbeidet med programmering.

3.5.1.3 Resonnering

Matematisk resonnering handler om å forstå matematiske tankerekker, og å følge og vurdere disse (Svorkmo & Valbekmo, 2020; Utdanningsdirektoratet, 2020a). I likhet med argumentasjon kan disse basere seg på matematiske bevis, men også bevis basert på mønstre, figurer og tegninger (Svorkmo & Valbekmo, 2020). Calao et al. (2015), Calder (2010), Ke (2013) og Taylor et al. (2010) trekker frem økt erfaring og ferdigheter innen resonnering gjennom arbeidet med oppgaver på et programmeringsspråk.

Taylor et al. (2010) viser også til et eksempel der en av de såkalt lavt presterende elevene viste evne til å kunne resonnere seg frem til mangler ved egne koder samt ser verdien i medelevenes forslag. Elever, bade faglig sterke og elever med matematikkvansker, har en tendens til å engasjere seg i utviklingsorienterte problemløsingsoppgaver gjennom ulike former for resonnering og resonnement (Ke, 2013). Forsstøm og Kaufmann (2018) viser til at utbyttet for lavt presterende elever både kan være en ny rolle i klasserommet/ gruppearbeidet og at de kan utvikle sofistikerte matematiske ideer.

Arbeidet med programmering kan øke elevers prestasjoner i matematikk i form av problemløsing, øving eller utholdenhets, modellering, resonnerings ferdigheter (Calao et al., 2015). Likevel er forskningen til Calao et al. (2015) noe sprikende. Til tross for at det viser til økte resonneringsferdigheter, viser de på en annen at modellerings og resonnerings ferdighetene utviklet seg bedre i tradisjonell undervisning uten bruken av programmering. I dette tilfellet kan det dermed se ut til at læringspotensialet kan være større i tradisjonell undervisning.

3.5.1.4 Modeller og Representasjoner

Logo ble i sin tid utviklet som et rammeverk for arbeid med matematiske konsepter med utgangspunkt i Piagets konstruktivistiske visjon (Feurzeig et al., 2011). Tanken var at læring skulle være en aktiv prosess av konstruksjon av kunnskap. Feurzeig et al. (2011) trekker frem fem argumenter for hvorfor programmering bør brukes i en matematisk sammenheng. To av disse argumentene ser ut til å konkret kunne knyttes opp mot modelleringskompetanse. Det første av de to handler om at programmering skaper gode muligheter for eksperimentering av matematiske ideer. Eleven kan observere, forstå og modifisere programmene. Dette fremmer mulighet for å eksperimentere med for eksempel problemløsingsstrategier i matematikk. Det andre argumentet handler om at programmerings oppgaver skaper gode muligheter for å kombinere «formal knowledge» og heuristisk kunnskap og skaper på den måten en god

kontekst for å arbeide med begge formene for matematisk kunnskap. Det siste argumentet handler om at programmering åpner for tilegnelsen av matematiske ideer. Gjennom å observere og modifisere i det første argumentet bruker en ferdigheter knyttet til kompetansen. Resnick et al. (2009) forteller at programmering ikke bare vil fungere som et problemløsingsverktøy og et middel for modellering og representasjon av ideer, men også hvordan dette påvirker muligheten for metakognisjon. «And since programming involves the creation of external representations of your problem-solving processes, programming provides you with opportunities to reflect on your own thinking even to think about thinking itself" (Resnick et al., 2009).

Datamaskinen vil ifølge Feurzeig et al. (2011) først og fremst fungere som et «mathematical laboratory» (s. 498). Der elevene kan finne, uttrykke og teste prosedyrer for å løse matematiske problemer. Arbeidet vil videre bidra til at elevene lærer seg å utvikle en plan for arbeidet med problemløsende oppgaver. Arbeidet med å løse et problem kan ses på som en konstruksjonsprosess. I denne prosessen kan elevene lære å formulere matematiske problemer som programmer og dermed også bruke feilsøking som en del av løsningsmetoden. Clements og Merdith (1992) fremhever at flere av elevene som brukte Logo ikke kun brukte dette som et matematisk verktøy, men også som et verktøy for tenking.

Haraldsrud et al. (2020) vektlegger modelleringsperspektivet. Dette perspektivet på bruken av programmering åpner for utforsking av sammenhenger uten at en behøver å ta hensyn til de begrensingene matematikken setter. Modellene skal fungere som en forenkling av et virkelig fenomen (Haraldsrud et al., 2020). Elevene kan lage modeller og dermed visualisere de matematiske fenomenene. I tillegg til at de kan bruke matematikkens uttrykk og symboler for å modellere virkelighetsnære problemstillinger (Ke, 2013). Dette vil også hjelpe elevene med å kunne generalisere løsningsstrategier. Programmering via spilldesign, har ifølge Ke (2013), fungert på denne måten. Ved å fungere som en modell åpen for manipulering har spilldesign også åpnet for muligheten til å utvikle matematisk tenking (Ke, 2013).

Elever får gjennom programmeringen muligheten til å representere ideer og kognitive prosesser. Samtidig gir de visuelle representasjonene direkte tilbakemelding, og sørger for at elevene han følge med på sin egen tankeprosess. De kan teste hva som fungerer og hva som ikke fungerer. På denne måten kan elever både følge med på sin egen tankeprosess, men også lære å kontrollere den (Clements, 1984). Det er videre, som Haraldsrud et al. (2020) påpeker, viktig å være bevist på at det ikke nødvendigvis handler om å gjøre noe nytt, men «å illustrere fagstoffet på flere måter» (s. 162).

3.5.2 Spesifikke konsepter

I denne delen vil jeg trekke frem de matematikkspesifikke konseptene som av litteraturen trekkes frem. Konseptene regnes som matematikkspesifikke, da de hovedsakelig brukes i en matematisk kontekst og kan knyttes til den mer «tradisjonelle» matematikkundervisningen.

3.5.2.1 Algoritmer og symboler

Bueie (2019) bruker eksempler fra de gamle grekerne og egypterne når han viser til at store deler av den første matematikken var algoritmisk, og bruker dette som et argument for å knytte matematikk, algoritmer og programmering sammen. Disse matematiske algoritmene viser likhetstrekk med hvordan en datamaskin regner i det binære tallsystemet. På denne måten skaper Bueie (2019) en sammenheng mellom programmeringens algoritmer og algoritmene i matematikk. Videre kan en si at matematisk kunnskap blir like relevant i utviklingen av dataprogrammer eller spill. Dette fordi hvert eneste program er basert på matematiske algoritmer (Lambi`c, 2010). Programmering innebærer å analysere, forstå og generelt løse problemer som resulterer i bruk av en algoritme (Norstein & Haara, 2018). I arbeidet må en «verifisere kravene til algoritmen, og undersøke dens korrekthet. Gjennomføre (ofte referert til som koding) algoritmen i et bestemt programmeringsspråk» (Norstein & Haara, 2018, s. 79-80).

I likhet med Bueie (2019), knytter Haraldsrud et al. (2020) matematikk og programmering sammen gjennom sammenlikning av tallene. De trekker frem de ulike tallene vi trenger for å få et program til å fungere slik vi ønsker. Blant disse finner vi heltallene som innebærer null, de naturlige tallene og negative tall. Disse er for datamaskinen entydige og tellbare. Videre finner vi desimaltallene som får en noe annen fremtoning i programmeringen og kalles for flyttall. Tallenes standardform blir et viktig hjelpemiddel for å holde styr på alle nuller. Bruken av tall på denne måten kan vitne om at elevene har behov for en grunnleggende tallforståelse for å kunne jobbe med programmering. Dette blir mest relevant i arbeidet med tekstprogrammering, men også i arbeidet med blokkprogrammering kan det bli behov for å arbeide med tallene.

De aritmetiske operasjonene og regnerekkefølgen er like i matematikk og i programmering. Regneoperatorene og rekkefølgen for kommandoene eller kodene er like relevante for at kodene skal fungere (Bueie, 2019; Haraldsrud et al., 2020). Likevel sier dette lite om hvor mye elevene faktisk arbeider med tallforståelsen og regneferdighetene mens de programmerer. Ke (2013) Viser til hvordan elevene i arbeidet med å designe oppgaver stadig

måtte reflektere over bruken av algoritmer, tallene og symbolene for at programmet skulle gjøre slik de ønsket.

I prosjektet til Stigberg og Stigberg (2020) la en av lærerne vekt på algoritmer og strukturer som fellesnevnere for både matematikk og programmering. Læreren konkluderte dermed med at dersom elevene øvde på programmering, ville de forstå matematiske strukturer bedre (Stigberg & Stigberg, 2020). I tillegg åpner det for numeriske metoder der det arbeides med tall, desimaltall og variabler, Aritmetikk og bruk av symboler, samt løkker og funksjoner (Haraldsrud et al., 2020). Haraldsrud et al. (2020) bruker begrepet numerisk matematikk for å beskrive den delen av matematikken som handler om å utforme algoritmer for å få matematiske løsninger med færrest mulig feil. Da utformingen av algoritmer kommer til sin rett i arbeidet med matematiske løsninger og i arbeidet med utforming av koder innen programmering, kan det, virke som at det er potensiale for å arbeide med den numeriske matematikken i arbeide med programmering.

3.5.2.2 Algebra

Bueie (2019) uttrykker at for å lage programmer, kan algebraiske uttrykk være gode utgangspunkt. Dette samsvarer med Ke (2013) som uttrykker at «…participants had to comprehend and apply the arithmetic and algebraic expressions embedded in programming» (s. 35). Elevene kunne på denne måten oppdage sammenhengen mellom symboler og representasjoner av matematiske konsepter, og skape en forståelse. Symbolene som brukes i algebraiske uttrykk og symbolene som brukes i programmering har likhetstrekk. Ved å introdusere programmering som metode i matematikk kan elever utvikle større innsikt i det å sammenlikne algoritmer og prosedyrer enn det de kan i tradisjonell matematikkundervisning (Calao et al., 2015).

Det finnes likheter mellom algoritmene som er nevnt over og algebra som vi arbeider med i tradisjonell matematikk undervisning. Smith (2003) fremhever at algebra involverer regler for manipulering av symboler og at disse derfor er nært relaterte. Det er dette vi gjør i programmering også. Vi arbeider med symbolene, og manipulerer og utformer koder med disse slik at datamaskinen gjør det vi ønsker. Videre kan operatorer klassifiseres avhengig av type og antall og hva slags type objekter en bruker som verdier (Feurzeig et al., 2011). Algebra handler om å generalisere mønster, om å se funksjoner, relasjoner og variasjon.

3.5.2.3 Mønster og mønstergjenkjenning

I arbeidet med programmering handler det i stor grad om å skape algoritmer som på en enkel måte gjentar mønstre. For eksempel kan en i Scratch eller i Python tegne opp geometriske figurer. Det kan handle om at en ønsker å tegne opp flere ulike geometriske figurer i et gjentakende mønster. I stedet for å skrive inn koder for hver enkelt figur flere ganger, kan en velge å heller gjenta en koderekke flere ganger. For å kunne dette må en definere hva en ser på som mønsterrekken, samt bestemme hvor ofte dette mønsteret skal gjentas. Haraldsrud et al. (2020) bekrefter mønstergjenkjenningens rolle, og sier at «Sentralt i algoritmisk tenkning ligger mønstergjenkjenning. Det vil si å gjenkjenne strukturer og likheter mellom ulike prosesser og rutiner.» (Haraldsrud et al., 2020, s. 189). Disse mønstrene kan brukes til å konstruere algoritmer. Mønster handler om hvordan noe kan utvides eller gjentas og dermed endres (Smith, 2003).

3.5.2.4 Tallforståelse

Når en arbeider med tall, blir forståelsen av disse viktig. Det blir relevant med en forståelse av tallenes verdier eller tallenes innhold. Altså en kjennskap til hva tallene står for og betyr. Benton, Saunders, Kalas, Hoyles og Noss (2018) utformet i forbindelse med Scratch Math prosjektet et undervisningsopplegg med fokus på plassverdisystemet. Opplegget tok for seg overgangen fra enere til tiere og hva som skjer med sifrene hvis en for eksempel adderer 1 til 9. Her ble det lagt fokus på overgangen fra ener til tier. Som et slutt resultat skulle elevene utforme en stoppeklokke. På denne måten arbeidet elevene med tier overganger, samt tid der de måtte huske at klokken går over til 1 minutt etter 60 sekunder. I dette prosjektet viste det seg at elevene som fikk god veiledning fra læreren klarte å utvikle både en fungerende stoppeklokke, og i tillegg tok i bruk kunnskap om plassverdisystemet for å få dette til. I klasser der læreren i mindre grad støttet elevens arbeid og elevene i større grad måtte oppdage løsninger selv, var det elever som klarte å utvikle stoppeklokker, uten å bruke kunnskaper om plassverdisystemet. Benton et al. (2018) viser i prosjektet et det er gode muligheter for å arbeide med tallforståelse i arbeidet med programmering, men at læringsutbyttet av dette i stor grad er avhengig av hvordan læreren skaper sammenhengen mellom programmerings arbeidet og det matematikkfaglige innholdet (Benton et al., 2018).

3.5.2.5 Variabler

Når en snakker om algebra og algoritmer blir det også naturlig å trekke inn variabler. Disse er i likhet med de andre overnevnte like sentrale i arbeidet med programmering. Variablenes rolle er å angi eller representere ulike og i noe grad varierende verdier som kan påvirke det

endelige produktet avhengig av variabelens verdi. Variabler i både matematikk og programmering har flere like egenskaper. Innen programmering blir variablenes oppgave å ta vare på informasjon som senere kan brukes i datamaskinen (Bueie, 2019). Smith (2003) definerer variabler som et middel for å uttrykke gjentatte handlinger og operasjoner. Det å kunne uttrykke en gjentakende operasjon kan være nyttig i utformingen av koder. Ke (2013) viser til at spilldesign gjennom programmering skaper en meningsfull kontekst for arbeid med matematiske konsepter, deriblant variabler, samtidig som det skaper en umiddelbar visuell tilbakemelding til elevene som arbeider med konseptene.

3.5.2.6 Funksjoner

I tillegg til å arbeide med ulike variabler, blir det relevant å trekke frem funksjoner og funksjonsuttrykk. Calder (2010) viser at elevene i prosjektet de arbeidet med, så en sammenheng mellom «input» og handlingen som skjedde på skjermen. Altså at de så en sammenheng mellom dataene eller verdiene som ble lagt inn i koden og resultatet fremtonet i en handling på skjermen. På denne måten jobbet elevene med relasjonell tenking mellom «input» og handling i programmeringsarbeidet. Sinclair og Petterson (2018) så på sin side at elevene i arbeidet med utforming av modeller av industrielle maskiner oppdaget hvordan hastigheten på et hjul kunne variere avhengig av størrelsen på hjulet. Elevene tok altså i bruk to variabler. En for størrelsen og en variabel for hastigheten og oppdaget at disse påvirket hverandre. Dette er i tråd med definisjonen Smith (2003) bruker for å definere en funksjon. Han sier at en funksjon er "a relationship between two existing sets, whereas the second gives the image of an input-output process in which the function is the process producing the output values" (Smith, 2003, s. 141). Den ene variabelen avhenger av den andre i en funksjon. Dette kan tolkes som at gjennom arbeidet med variabler i programmeringsarbeidet, med input verdier og påfølgende resultat, kan programmering brukes i arbeidet med begynnende funksjoner. Det viser også hvordan programmering kan brukes for å tegne opp gjeldende funksjoner og grafer (Bueie, 2019; Haraldsrud et al., 2020).

3.5.2.7 Koordinatsystem

I de fleste blokkprogrammeringsspråkene, arbeider man i et koordinatsystem. Dette koordinatsystemet forteller oss hvor i feltet et element plasseres, enten det er en gjenstand som skal forflyttes, et tegneverktøy eller en karakter i utformingen av et spill. Til tross for at Ke (2013) lar elevene arbeide med spilldesign, er utgangspunktet for arbeidet det samme; programmering. Flere av elevene arbeidet med koordinatsystemer med x- og y-akser i utviklingen av spillene sine (Ke, 2013). Dette bidro til å skape en bedre forståelse av

koordinatsystemet. Calder (2010) og Taylor et al. (2010) observerte i sine prosjekter at elevene arbeidet i et koordinatsystem. I arbeidet fikk elevene behov for å orientere seg, og å finne ut av hvordan de skulle tegne figurer ved hjelp av koordinatsystemet (Taylor et al., 2010). Til hvilken grad de fikk behov for koordinatsystemet, var noe avhenging av blokkene elevene valgte å bruke for å løse oppgaven de fikk. I et koordinatsystemet blir elever også introdusert for positive og negative tall (Haraldsrud et al., 2020; Ke, 2013; Taylor et al., 2010). Blant annet ved at origo ligger sentralt i skjermbildet, og derfor kan områdene til venstre og nedenfor origo angi negative verdier, og koordinatene kan få negative fortegn. Mens koordinatene til høyre og over origo kan angis ved positive tall.

3.5.2.8 Geometri

Selv om både algoritmer, funksjoner, positive og negative tall og koordinatsystemer ser ut til å være matematiske konsepter som kan jobbes med i forbindelse med programmering, er det likevel et matematisk konsept som fremtrer sterkere enn andre. Særlig fremtredende er geometri (Forsstøm & Kaufmann, 2018). For eksempel blir det ofte arbeidet med sirkelgeometri i arbeidet med å programmere roboter og i Scratch arbeides det ofte med, rektangler, trekanter, sirkler og vinkler. I tillegg fremmet alle de spurte lærerne i prosjektet til Stigberg og Stigberg (2020) geometri som et matematisk område der programmering kunne bidra til bedre forståelse. Benton et al. (2017) utviklet et undervisningsopplegg som tok utgangspunkt i 360 graders rotasjon. Prosjektet var en del av et større prosjekt, med fokus på flere ulike matematiske konsepter. I tillegg til 360 graders konseptet arbeidet de også med geometri i sin helhet. Mønster og gjentaking av mønster blir utgangspunkt for oppgavene elevene skulle arbeide med. Videre får vi eksempler på situasjoner der elevene ønsker å få en figur til å rotere. Elevene får da arbeidet med tallene som angir størrelsen på rotasjonen samt hvor mye figuren i virkeligheten fysisk roterer. Sinclair og Petterson (2018) oppdaget at elevene i arbeidet med utforming av modeller blant annet tok i bruk geometriske objekter som segmenter av geometriske figurer og sirkler. Dette gjaldt også i prosjektet til Lambi`c (2010) der de i tillegg jobbet med geometriske figurer med angitte størrelser som høyde og diagonal (Lambi`c, 2010).

I arbeidet med programmering og det å skrive koder vil en ifølge Feurzeig et al. (2011) også arbeide med matematiske ideer som f.eks. at en kan operere/manipulere ulike objekter. Gjennom programmering kan man «tegne» ulike todimensjonale figurer. Som f.eks. rektangler, trekanter og kvadrater i tillegg til rette linjer og vinkler. Clements og Merdith (1992) viser til van Hieles modell for geometrisk tenking og de ulike stadiene i utviklingen av

denne. Disse stadiene går fra det visuelle, altså å kunne gjenkjenne og sette navn på geometriske figurer på det laveste stadiet, til å på det høyeste nivået, kunne forstå teoretiske bevis og hvorfor det blir viktig å kunne bevise noe (Hinna, Rinvold & Gustavsen, 2012). Clements og Merdith (1992) tar altså utgangspunkt i van Hiele-modellen og viser til hvordan Logo-programmering kan bidra til å utvikle elevenes progresjon fra et nivå til neste, for på den måten å utvikle elevenes oppfatning av geometriske figurer.

Det finnes flere eksempler der en tar utgangspunkt i konsepter innenfor geometri. Haraldsrud et al. (2020) har i sin bok *Programmering i skolen* utformet flere eksempler det blokkprogrammeringsspråket Scratch brukes. I programmet tar han utgangspunkt i tegneverktøyet «penn», og uttrykker at «Særlig geometri er et felt det er lett å utforske med Scratch» (Haraldsrud et al., 2020, s. 96). Gjennom utformingen av ulike geometriske figurer, både enkeltvis og i sammensetning, arbeider elever med kjennetegn av ulike figurer som firkant, femkant og sirkler. I tillegg til 360 graders rotasjon og utvikling av mønstre basert på rotasjon av enkeltfigurer. Elevene må da arbeide med å finne ut av hvor mange ganger den gitte figuren må rotere for hver gang for få det ønskede resultatet. I tillegg til å arbeide med geometri, arbeider elevene med mønster, mønster gjenkjenning, og repetisjon av mønstre, som er i tråd med Smith (2003) idè om mønstre og begynnende algebra.

3.5.2.9 Måling og måleenheter

I de ulike programmeringsverktøyene ligger det til tider et behov for å angi avstand eller lengdemål. Dette blir som oftest angitt i måleenheten dots per inch. Behovet for å bruke dette kan oppstå ved forflytting av gjenstander, bevegelse eller ved for eksempel tegning av geometriske figurer og angitte lengder. I tillegg ligger det en mulighet for å tegne og angi grader i vinkler og for rotasjoner. I prosjektet til Lambi`c (2010) fikk elever opplæring i bruken av programmet *C++ Builder.* De fikk i oppgave å utvikle en kalkulator for utregning av arealet og omkretsen av ulike geometriske figurer. For å få til dette la elevene inn variabler for figurenes sidemål som lengdemål for høyden og bredden, men også variabel for diagonalen. Elevene fikk innsikt i hvordan matematikken kunne brukes i en praktisk sammenheng (Lambi`c, 2010). Prosjektet ble gjennomført med ungdom i alderen 13-19år, men paralleller til bruken av liknende måleenheter for geometriske figurers sidelengder kan trekkes til elever også på mellomtrinnet. Calder (2010) utviklet et prosjekt der elever på 6.trinn skulle utvikle et spill med matematisk innhold for yngre elever. I arbeidet med å utforme spillet i Scratch, måtte flere av gruppene justere tiden for gjentatte handlinger. For eksempel ønsket noen av elevene at en figur skulle vente på skjermen i noen sekunder før den

«fløy vekk». Elevene arbeidet på denne måten med tidsintervaller og måling av tid. Calder (2010) uttrykker at oppgavedesignet la til rette fo utforsking av både vinkler angitt i grader og måling av tid og lengde. Elevene i prosjektet til Benton et al. (2018) gjorde det samme da de fikk i oppgave å utvikle en stoppeklokke. Elevene tok da både hensyn til tid, omgjøring fra sekunder til minutter, i tillegg til tieroverganger.

I både litteraturen og i ferdige undervisningsopplegg ser det ut til at ulike måleenheter brukes for utforming og for å besvare praktiske oppgaver. Dette kan knyttes til kontekster som samsvarer med de kontekstene en møter på senere i den reelle verden der elever kan få bruk for disse ferdighetene.

3.5.2.10 Data, Statistikk og sannsynlighet

Wing (2006) uttrykker i sin artikkel at «Statistical learning is being used for problems on a scale, in terms of both data size and dimensions, unimaginable only a few years ago" (Wing, 2006, s. 34). Hun knytter dermed statistikk, problemløsing og datahåndtering sammen. Haraldsrud et al. (2020) trekker frem at programmering er godt egnet til å arbeide med data. Data som kan hentes fra utallige ulike tekstfiler som Excel, notatbøker eller andre databaser. Ved bruk av tekstspråket Python viser han hvordan man kan arbeide med å lage visuelle fremstillinger at informasjonen en kan innhente fra de ulike tekstfilene. Bueie (2019) viser gjennom tekstprogrammering hvordan en kan bruke ulike løkker for å arbeide med store datamengder og orientere seg i disse. Disse eksemplene tar utgangspunkt i et langt høyere ferdighetsnivå enn det en kan forvente av elever på mellomtrinnet. Likevel viser dette at det er et potensialet for å arbeide med statistikk. En kan i tillegg arbeide med og regne ut både sentralmål som gjennomsnitt og standardavvik i en datamengde ved hjelp av programmering i Python (Bueie, 2019; Haraldsrud et al., 2020). Dette krever igjen at elevene har kjennskap til både begrepene, algoritmene og kodene som skal brukes for å få dette til. Videre finnes det muligheter for å knytte arbeidet med statistikk til sannsynlighetsregning. En kan lære elevene å programmere prosjekter med tilfeldige utfall for deretter å bruke utfallene til å regne ut gjennomsnitt. Tilfeldige utfall kan for eksempel hentes ut fra et eksisterende datasett og baseres på tilfeldig trekning av heltall. Et annet eksempel som også kan brukes på mellomtrinnet er datainnsamling og bruk av Stigespillet for arbeid med innsamling av statistikk fra en klasse og utforske både gjennomsnitt av antall kast og sannsynlighet for å havne på en stige ved bruk av Python (Haraldsrud et al., 2020).

3.5.3 Matematisk læringspotensial og matematisk kompetanse

Matematikk er mye mer enn det å regne, og det kan se ut til at programmering åpner for å arbeide med den numeriske matematikken med muligheter for regning, men også den delen av matematikken som krever kognitiv nytenking. Programmering sørger dermed for at elevene får nye måter å bruke den matematiske kunnskapen de har tilegnet seg. Avsnittene over viser at det finnes muligheter for at det kan brukes kunnskaper om både algoritmer, tallforståelse, symboler, variabler, funksjoner, geometri, koordinatsystemer, måling og måleenheter. Den viser også til at elevene får bruk for mer overordnede ferdigheter som algoritmisk tenking, og at dette også bidrar til å lære elevene strategier for å ta tak i problemløsingsoppgaver. Dette blant annet gjennom prøving og feiling og ved at elevene får muligheten til å bruke programmeringen for å modellere handlinger som tidligere ikke har vært mulig. Det kan se ut til at det å implementere programmering i matematikkfaget kan åpne nye dører for arbeidet med faget.

I lys av det vi definerer som matematisk kompetanse kan vi finne igjen flere av Niss et al. (2002) kompetanser i arbeidet med programmering (Kap. 3.4). Alle de spesifikke konseptene åpner for muligheten til å ta i bruk symbol- og formalismekompetansen elevene innehar, og dermed også muligheten for å utvikle denne kompetansen videre. Dette er noe avhengig av abstraksjonsnivået i tillegg til at det kanskje kan være avhengig av om en velger å ta i bruk blokkprogrammering eller tekstprogrammering, der syntaksen er betydelig viktigere enn i blokkprogrammering. Elevene må ta i bruk matematikkens symboler på korrekt vis for at kodene skal fungere slik en ønsker, i tillegg blir det formelle satt når elever for eksempel skal definere figurer og utvikle kalkulatorer for å finne omkrets og areal. Ved å bli kjent med programmering i seg selv åpner man opp for å kunne bruke programmering som både et modelleringsverktøy og et hjelpemiddel i arbeidet med ulike matematiske konsepter både spesifikke og generelle eller overordnede. På denne måten åpnes det også for muligheten til å arbeide med utviklingen av hjelpemiddelkompetansen.

De overordnede matematiske konseptene åpner for arbeid med kompetanseområdene Tankegangskompetanse, Problemløsingskompetanse, Modelleringskompetanse, Resonneringskompetanse, Representasjonskompetanse og kommunikasjonskompetanse. Der tankegangskompetanse kan ses som en måte å sette i gang og rydde opp i tankene sine. I tillegg til at gode spørsmål kan bidra til å øke nysgjerrigheten og kanskje potensielt fremme løsninger og løsningsstrategier.

Det kan konstateres at det finnes mer enn et matematisk læringspotensial ved å arbeide med programmering i faget matematikk. Likevel er det lite forskning som forteller noe om hva elevene faktisk sitter igjen med etter undervisningen, men at elevene bruker disse konseptene åpner likevel for at det er et potensiale for å arbeide med matematikk gjennom programmeringsoppgaver.

3.6 Lærerens rolle

I dette delkapittelet vil jeg se på hvilken rolle læreren får i arbeidet med programmering, samt hva som kan gjøres for at implementeringen skal skje på best mulig vis. Jeg vil også se på hvordan kunnskap overføres fra matematikken til programmeringsarbeidet, her vil jeg benytte teori fra Salomon og Perkins (1989). Til slutt vil jeg legge frem Rammeverket 5E-er, som er et rammeverk utformet og tatt i bruk i forbindelse med Scratch Math prosjektet til Benton et al. (2016) og Benton et al. (2017).

3.6.1 En sentral rolle

For at programmering skal kunne bidra til læring av matematikk, må den grunnleggende matematiske kunnskapen ligge til grunn (Benton et al., 2017). Det vil være vanskelig å kreve av elever at de skal mestre å bruke programmering for å løse en oppgave når elevene ikke forstår matematikken. Videre legges det frem at matematikken vil forsvinne i bakgrunnen dersom elevene får arbeide med programmering uten nøye veiledning fra læreren. Lærerens rolle i å skape sammenhengen blir sentral. Benton et al. (2017) påpeker vil det å gjøre det matematiske innholdet eksplisitt og systematisk knytte det til elevenes forkunnskaper i og om matematikk være kritisk for at læring skal finne sted. Pea og Kurland (1984) viser på en annen side til at komplekse kognitive endringer sannsynligvis ikke skjer ved kun å eksplisitt undervisning, men elevene må også engasjeres i oppgavene som skal gjøres slik at de får arbeide med konseptene som skal læres. Det avhenger videre også av valgene læreren tar i forkant av underveiningen og underveis i denne. Blant annet gjennomvalget av arbeidsmetoder og valg av oppgaver og utformingen av oppgavene (Bueie, 2019; Wing, 2006). Best læringseffekt oppstår når læreren medierer elevene i deres arbeid med problemløsende aktiviteter (Clements & Merdith, 1992). Lærerens rolle blir dermed å lede arbeidet og å besvare elevenes spørsmål som skulle dukke opp underveis (Lambi`c, 2010).

For at læreren skal kunne mediere læring, skape sammenhenger og støtte transformering av læring fra matematikken til programmeringsmediet, må læreren være komfortabel i sin undervisning ved bruk av programmering (Weintrop et al., 2015). Pea og Kurland (1984) ser

på læring av programmering, og konkluderer med at «(…) is not taught by computers or by programming languages but by teachers (…)» (s. 149). De påpeker at det også er andre elementer som spiller inn, men lærerens kunnskap knyttet til programmering og kunnskap om hvilke oppgaver som passer elevenes ferdighetsnivå kan dermed også ha innvirkning på hva som læres bort.

Læreren blir dermed et viktig element i arbeidet med programmering. Det kan virke som at Elevene i større grad skal utforske, feilsøke og finne løsninger selv. Samtidig som lærerens rolle i større grad handler om å skape sammenhenger mellom elevenes eksisterende kunnskap, kunnskap de tar i bruk når de programmerer og den matematiske kunnskapen som kommer til nytte i det praktiske programmeringsarbeidet. Dette kan læreren gjøre gjennom valg av de gode arbeidsmetodene som skaper muligheter for utforsking, og bruken av oppgaver som bygger oppunder sammenhengen mellom programmeringsaktiviteten og det matematikkfaglige innholdet.

3.6.2 Strategier for implementering

3.6.2.1 Overføring av kunnskap

I Kap. 3.5 Matematisk læringspotensial, kan vi se at det er muligheter for læring av matematiske konsepter i arbeidet med programmering. Likevel er det ikke en selvfølgelighet at den matematiske læringen finner sted under dette arbeidet.

Kunnskapen elevene innehar om matematikk må kunne overføres til en ny situasjon. En situasjon der de må bruke denne kunnskapen for å kunne løse oppgaver i et programmeringsspråk. Salomon og Perkins (1989) omtaler denne overføringen som «transfer». Det å overføre kunnskap og ferdigheter fra et område til et annet, kan skje på to måter. Den første måten handler om den spontane, automatiske overføringen av godt innøvde ferdigheter og kunnskaper som ikke krever særlig refleksjon eller utfordringer kognitivt, og kalles «low-road of transfer» (Salomon & Perkins, 1989, s. 118). Den andre formen for overføring av kunnskap innebærer eksplisitt bevisstgjøring og abstraksjon fra et område til et annet samtidig som en skaper en kobling mellom de to områdene, og kalles «High-road transfer» (Salomon & Perkins, 1989, s. 118).

«High-road transfer» kan igjen deles inn i to ulike former; «forward-reaching transfer» og «backward-reaching transfer» (Salomon & Perkins, 1989, s. 119). Der førstnevnte i stor grad handler om å lære seg en kunnskap eller en ferdighet så godt at denne blir en integrert del av personen. Ferdigheten læres som et generelt prinsipp og kan ved en senere anledning spontant

fremstå som en god løsning i en annen situasjon. Sistnevnte handler i større grad om behovet for å finne en løsning på et problem, og man søker derfor bevisst i sitt repertoar av løsninger og erfaringer som kan fungere også i denne situasjonen. Overføring av kunnskap kan dermed skje på mange ulike måter, dersom forholdene legger til rette for det. Dersom det ikke skjer noen overføring av kunnskap eller ferdigheter vil dette være fordi forholdene ikke legger til rette for at overføringen av kunnskaper eller ferdigheter kan overføres. For «Low-road transfer» handler det i stor grad om å automatisere gjennom variasjon av situasjoner og kontekster. For «High-road» transfer er abstraksjon og generealisering viktige nøkkelord. Det er likevel viktig å være bevist på at en både kan overføre kunnskap «low-road» og «high-road» samtidig og at den ene ikke utelukker den andre. Disse to mekanismene for overføring av kunnskap kan gjerne arbeide sammen for å øke overføringsevnen(Salomon & Perkins, 1989).

Når det gjelder programmering, vektlegges det at kravene for å kunne overføre kunnskap ved «low-road of transfer» ikke er til stede, men dersom en bevisst arbeider med, og gjør kunnskapen og overføringen av kunnskapen eksplisitt i den nye situasjonen, altså innen programmering, vil mulighetene for overføring av kunnskapen være betydelig større. Eksemplene viser at dersom undervisningen møter kravene for “high-road transfer» finner overføringen av kunnskap som regel sted (Salomon & Perkins, 1989). Dette vil si at selv om det kan være en sammenheng mellom matematikk og programmering, og det kan være relevant å arbeide med programmering i matematikk, vil ikke overføring av kunnskap mellom disse områdene finne sted dersom ikke forholdene for «low-» eller «high-road of transfer» ligger til grunn. Salomon og Perkins (1989) anbefaler å eksplisitt undervise for overføring av kunnskap mellom områder. Et eksempel kan være å oppfordre elevene til å sammenlikne ved å se på likheter og forskjeller. Overføring av kunnskap skjer sjeldent av seg selv, men ved oppfordring og tilrettelegging kan vi bidra til at det skjer. «If only we teach it, we are most likely to get it.» (Salomon & Perkins, 1989, s. 137).

3.6.2.2 Bentons 5Eer

Benton et al. (2016); Benton et al. (2017) ser på implementeringen av programmering for arbeid med matematikk gjennom programmet Scratch. I den forbindelse har de videreutviklet diSessa og Cobbs (2004) rammeverk for handling. Dette nye rammeverket tar for seg pedagogiske strategier for å implementere programmering som et verktøy i matematikk, rammeverket fungerer veiledende (Benton et al., 2016, s. 5). I rammeverket finner vi fem komponenter, 5E-er. Dette er 5 handlinger læreren bør legge til rette for i undervisningen.

Disse fem er «Explore», «Envisage», «explain», «Exchange» og «BridgE» (Benton et al., 2016; Benton et al., 2017).

3.6.2.2.1 Explore

Explore handler om å skape muligheter for at elevene i sitt arbeid kan utforske ideer, arbeide med problemløsende aktiviteter, prøve nye ting selv og bruke prøve og feile metoden for på den å oppdage nye ideer igjen (Benton et al., 2016; Benton et al., 2017). Benton et al. (2016) legger vekt på at denne delen av rammeverket handler om at elevene skal ta kontroll over sin egen læring og utvikling og selv oppdage hvorfor noe fungerer eller ikke fungerer.

3.6.2.2.2 Envisage

Envisage handler om at elevene må kunne reflektere over og prøve å forutse resultater av en handling eller en kode før de prøver ut kodene de bruker. På denne måten blir utforskingen mer målrettet i form av hensiktsmessig feilsøking. Det handler om å konstant reflektere rundt kodene en utformer samt revurdere det en har gjort (Benton et al., 2016; Benton et al., 2017).

3.6.2.2.3 Explain

I *explain* legger Benton et al. (2016) vekt på at et viktig aspekt ved læring handler om å kunne begrunne de valgene en gjør samt å kunne sette ord på det en selv har lært. På denne måten kan en dele ideer. En del av å få til dette er å stille spørsmål som oppfordrer til refleksjon, fremfor å kun uttrykke svaret. Klassediskusjoner blir fremmet som en viktig ressurs for utvikling av læring (Benton et al., 2017).

3.6.2.2.4 Exchange

Exchange handler om å skape muligheter for erfaringsutveksling, dele ideer og dermed kunne utvikle nye ideer på bakgrunn av andres innspill (Benton et al., 2016; Benton et al., 2017). På denne måten kan klassekamerater også fungere som hjelpere (Benton et al., 2017).

3.6.2.2.5 BridgE

Benton et al. (2016) og Benton et al. (2017) femte strategi, *Bridge*, trekkes frem som den viktigste strategien i rammeverket for implementering av programmering i faget matematikk. Det er denne som skaper grunnlaget for implementeringen. «Bridge» handler om å knytte sammen de to konseptene, en brobygging mellom matematikk og programmering. Dette må skje eksplisitt ved lærerens hjelp (Benton et al., 2016; Benton et al., 2017). Det oppfordres til at lærerne er beviste på de mulighetene som finnes for læring av matematikk, og at det skapes eksplisitte koblinger mellom programmeringsarbeidet elevene gjør og det matematikkfaglige

innholdet disse kan by på (Benton et al., 2016). Det å gjøre det matematikkfaglige innholdet eksplisitt, skal bidra til at elevene, på sikt, ser verdien av arbeidet de gjør, også fra et matematikkfaglig perspektiv.

3.6.2.2.5.1 Hugging and bridging

Benton et al. (2018) viser i forbindelse med Bridge, til teknikker læreren kan ta i bruk for å skape sammenhengen mellom programmeringsarbeidet og det matematiske innholdet. Her skilles det mellom «Hugging Techniques» og «Bridging Techniques» (Benton et al., 2018, s. 3). «Hugging» defineres som teknikker for å gjøre undervisningen i det nye domenet, her programmering, så likt det originale/ tradisjonelle arbeidet som mulig. Mens «bridging» på sin side handler om å promotere abstraksjon og skape sammenhenger ved for eksempel å arbeide med problemløsings strategier. Det anbefales å bruke en kombinasjon av disse. Typiske teknikker for «bridning» er å forutse sammenhenger, generalisere konsepter, bruke analogier, arbeide med parallelle problemer, og reflektere på et metakognitivt nivå (Benton et al., 2018). Typiske teknikker for «hugging» er å uttrykke forventninger ved å konkret uttrykke hva elevene skal lære, tilpasse aktiviteten, simulere ved for eksempel rolle spill, modellere og problembasert læring (Benton et al., 2018; Jones, Jones & Vermette, 2009).

I artikkelen til Franklin et al. (2016) ser de på hvordan hugging og bridging brukes i forbindelse med overgangen mellom blokkprogrammering og tekstprogrammering. Det vises til at «bridging should occur to present more general , highlevel principles including the separate, special nature of initialization" (s. 222). Mens hugging "could occur when students encounter traditional languages to relate what they learned in Scratch to what they see in a new context" (s. 222). Selv om det her ikke handler om hugging og bridging mellom programmering og matematikk, kan det likevel trekkes noen likhetstrekk. Blant annet kan bridging handle om å overføre generelle ideer fra matematikken til arbeidet med programmering, og hugging kan handle om å bruke det matematiske språket for å sette ord på det som skjer under arbeidet med oppgaver på et programmeringsspråk.

4 Metode

I dette delkapittelet vil jeg gå nærmere inn på valg av metode, utvalget, hvordan datainnhentingen har foregått, samt hvordan jeg har valgt å foreta analysen av den innsamlede dataen. I tillegg vil jeg legge frem det som kan påvirke oppgavens validitet og reliabilitet. Til slutt vil jeg se på hvilke etiske hensyn jeg har tatt i arbeidet.

4.1 Metoder for innhenting av data

I arbeidet med dette prosjektet benyttet jeg en kombinasjon av flere former for kvalitativ datainnhenting. Hovedkilden for datainnhenting har vært intervjuer av lærere og observasjon av undervisning, i tillegg jeg har også foretatt intervjuer av elever. Jeg ønsket en dypere innsikt i temaet programmering i matematikkundervisningen, derfor ville en kvalitativ metode kunne gi tilgang på dette. Observasjonene åpner for muligheten til å se det som foregår i klasserommet, mens intervjuer åpner for muligheten til å stille åpne spørsmål som oppfordrer til mer reflekterte svar. I tillegg gir det muligheten til å stille oppfølgingsspørsmål ved behov.

4.1.1 Observasjoner

En av metodene jeg benyttet for datainnhenting, var observasjon. Målet med observasjonene var å observere mest mulig av det matematikkfaglige arbeidet både blant elevene og i kommunikasjonen mellom læreren og elevene. Videre ønsket jeg å observere èn til to undervisningsøkter hos hver av lærerinformantene. Under observasjonene valgte jeg å ta i bruk en åpen, ikke-deltakende observatørrolle i form av en tilstedeværende observatør (Fangen, 2010). Dette innebærer at jeg som observatør ikke deltok i samhandlingen, men både elever og lærere var klar over hva jeg ønsket å observere (Fangen, 2010).

For å strukturere observasjonene, tok jeg i bruk et observasjonsskjema (Se vedlegg: 9.6 Observasjonsskjema). Observasjonsskjemaet var utformet etter strukturen vi kan finne i en undervisningssekvens. Med rader for innledning, hoveddel og en avslutning av timen. I tillegg har jeg delt skjemaet opp i kolonner for observasjoner rundt læreren og kolonner for observasjoner rundt elevene. Jeg registrerte hvilke strategier som ble tatt i bruk, samt hvordan det matematikkfaglige kom til uttrykk i kommunikasjonen mellom elevene og mellom læreren og elevene. For å få best mulig oversikt over hendelsene i klasserommet valgte jeg å plassere meg i et hjørne bakerst i klasserommet ved felles gjennomgang. Mens elevene arbeidet valgte jeg å bevege meg rundt i klasserommet, samtidig som jeg holdt meg i bakgrunnen.

4.1.2 Intervjuer

I tillegg til å observere, ble det gjennomført semistrukturerte intervjuer av både lærerne som underviste i de observerte undervisningssekvensene og tre til fire elever i tre av klassene. Et intervju innebærer en samtale mellom meg som forsker og læreren eller eleven som informant, der det utveksles synspunkter om et angitt tema (Kvale & Brinkmann, 2019). Et semistrukturert intervju har en veiledende intervjuguide, der rekkefølgen på spørsmålene kan variere, samt at det er mulighet for å i større grad legge til oppfølgingsspørsmål (Bryman, 2016; Christoffersen & Johannessen, 2012). Intervjuene er planlagt og gjennomført etter Kvale og Brinkmann (2019) 7 stadier for intervjuundersøkelse. I første omgang tok jeg utgangspunkt i å formulere problemstillingen og planlegge hvordan informasjonen fra informantene skulle innhentes. Intervjuene ble gjennomført med en veiledende intervjuguide (Se vedlegg: 9.2 Intervjuguide Lærer og 9.5 Intervjuguide Elev).

4.1.2.1 Intervju av lærere

I intervjuene med lærerne var spørsmålene utformet med bakgrunn i den observerte undervisningssekvensen, lærernes oppfatning av begrepet algoritmisk tenking og deres tanker om programmering i matematikk. Hovedformålet var å finne ut av hva lærerne anså som det matematiske læringsutbyttet av deres undervisningstime og hvordan lærerne tenkte at matematikken kunne kobles til arbeidet med programmering. Nøkkelspørsmålene i intervjuet ble derfor rettet mot dette. Det var videre utformet åpne spørsmål, som skulle åpne for refleksjon fra lærerne. Da intervjuguiden var ledende, og spørsmålene tok utgangspunkt i den undervisningen jeg hadde observert, var det også naturlig med intuitive oppfølgingsspørsmål basert på informasjonen lærerne ga. Spørsmål med tilknyttning til læreplanen er inkludert for å se om lærerne så noen sammenheng eller mulighet får å nå disse ved bruk av programmering. I tillegg til å selv ha utformet de fleste av spørsmålene har jeg også tatt i bruk tre av spørsmålene fra artikkelen til Kilhamn et al. (2021) (Se vedlegg 9.2). Da disse passet godt til den problemstillingen jeg var ute etter å finne svar på. For å bedre huske hva intervjuobjektene fortalte ble det tatt lydopptak av alle intervjuene.

4.1.2.2 Intervju av elever

Intervjuene av elevene ble gjennomført med mål om å finne ut av hvilket matematikkfaglig utbytte elevene selv opplevde at de satt igjen med etter undervisningen, samt om det samsvarte med lærerens intensjoner. Intervjuer på denne måten kan være viktige kilder for å samle informasjon om observasjoner der observasjonsobjektene deltar (Kvale & Brinkmann, 2019). Da det var barn som deltok i intervjuet, er noen av spørsmålene også utformet for å få

i gang en samtale. Derfor har spørsmål som «Hva er den morsomste oppgaven du har gjort med programmering?» blitt brukt. Spørsmålene om hvilke triks elevene har for å klare vanskelige oppgaver baserte seg på et ønske om å finne ut av om elevene tok i bruk strategier basert på algoritmisk tenking og problemløsing.

4.2 Utvalg

Utvalget består av et strategisk utvalg med en form for kriteriebasert utvelgelse (Christoffersen & Johannessen, 2012). Datainnsamling skulle foregå gjennom blant annet observasjoner, derfor ble det viktig at informantene kunne rekrutteres innenfor akseptable reiseavstander. Utvalget ble derfor begrenset ved å holde søket etter informanter til Østlandet. Rekrutteringen foregikk ved kontakt via e-post og oppfølging via telefon. I e-posten som ble sendt ut etterspurte jeg lærere som tok i bruk programmering i matematikkundervisningen. E-postene ble blant annet sendt på bakgrunn av at skolene hadde deltatt eller skulle delta i kodetimen til *Lær Kidsa Koding (Lær Kidsa Koding, 2021)*. I tillegg er utvalget basert på tips og råd om relevante lærere fra deltakende informanter eller kolleger, en form for «snowball-sampling» (Bryman, 2016; Christoffersen & Johannessen, 2012).

Til dette master-prosjektet var jeg på utkikk etter informanter som arbeidet som lærere i matematikk på mellomtrinnet, og som tok i bruk programmering i sin undervisning. Dette kunne være lærere som hadde mye erfaring i bruken av programmering og hadde formell opplæring, samt lærere uten formell opplæring og med mindre erfaring på området. Blant elevene, ønsket jeg kun å intervjue elever som hadde deltatt i den observerte undervisningssekvensen. Her ønsket jeg 2-3 elever fra hver av de observerte klassene. Utvalget var videre basert på frivillig deltakelse og læreren i de observerte klassene avgjorde utvalget. Det skulle være en kombinasjon av gutter og jenter.

Det skulle vise seg å bli en utfordring å få tak i informanter til studien. Grunnet COVID-19-pandemien, uttrykte flere at det ikke var nok kapasitet til å kunne bidra til studien. Andre uttrykte at dette ikke var noe de brukte så ofte eller at programmering hovedsakelig ble brukt i en tverrfaglig sammenheng. Opprinnelig tok prosjektet sikte på grunnskolens mellomtrinn, altså 5.-7.trinn, men jeg valgte å utvide søket fra mellomtrinnet til også å se etter informanter som underviste på 4.trinn. Utvalget består både av lærere som har tatt i bruk programmering tidligere og som har en erfaring med arbeidet med dette temaet, samt lærere med begrenset erfaring.

Jeg kom i kontakt med totalt 5 lærere som hadde mulighet til å bidra med informasjon. Lærerne arbeidet på ulike skoler på Østlandet og hadde ulike grad av opplæring. To av lærerne hadde opplæring fra vitensenteret og to av lærerne hadde tatt er årsstudium i programmering for grunnskolen. Den siste læreren hadde ingen formell opplæring annet enn intern opplæring på skolen og i kommunen. Skolene disse arbeidet ved varierte i størrelse og det samme gjorde klassene. Jeg fikk intervjue 10 elever fra 3 av skolene.

Alle informantene ble anonymisert i transkripsjonsprosessen, og både elever og lærere har fått fiktive koder. Lærerne har fått kodenavn L1, L2, L3, L4 og L5. Klassene der observasjonene foregikk er kodet på samme måte, og har blitt kalt S1, S2, S3 og S4. Elevene har fått kodene E1.1, E1.2, E1.3, E3.1, E3.2, E3.3, E4.1, E4.2, E4.3 og E4.4. Det ble foretatt observasjoner i klassene S1-S4, der det var mellom 10 og 22 elever i hver av klassene. I noen av klassene var det noe høyere fravær en vanlig grunnet utbrudd av COVID-19. Lengden på undervisningsøkten varierte. Hos S4 var det en avgrenset undervisningstime, mens det hos S2 var satt omtrent en hel dag for arbeidet de skulle gjøre. Hos S1 og S3 ble det i utgangspunktet planlagt rundt èn undervisningstime, men undervisningsopplegget tok lengre tid enn planlagt og gikk derfor utover flere timer. I tillegg til observasjoner ble det foretatt intervjuer av elever i klassene S1, S3 og S4. Grunnet pandemiens høye smitte tall ble det siste intervjuet med L5, foretatt via Zoom. Det ble derfor heller ikke gjennomført observasjoner av L5 sin undervisning.

Tabell 4.1 Viser oversikt over utvalget som har deltatt i studien. Informasjon innhentet fra elever og lærere.

Trinn	Skole	Bruk av programmering	Lærer	Elever
5.trinn	S1	Flere ganger de siste 1-2 årene. Omtrent 4 ganger dette skoleåret.	L1	E1.1
				E1.2
				E1.3
6.trinn	S2	Første gang.	L2	-
4.trinn	S3	Noen ganger tidligere – 3-4 ganger dette skoleåret.	L3	E3.1
				E3.2
				E3.3
7.trinn	S4	Første gang dette skoleåret. Førstegang i matematikk	L4	E4.1
				E4.2
				E4.3
				E4.4
Timelærer underviser programmering for hele grunnskolen.	-	Ukentlig i forskjellige klasser på forskjellige trinn.	L5	-

Tabell 4.2 Viser lærernes arbeidserfaring i antall år, samt studiepoeng i matematikk og opplæring i programmering

Lærer	År undervist i matematikk	Studiepoeng i matematikk	Opplæring i programmering
L1	4	60	Kurs via Vitensenteret
L2	*Ikke tilgjengelig*	*Ikke tilgjengelig*	Videreutdanning programmering for grunnskolen.
L3	*Ikke tilgjengelig*	*Ikke tilgjengelig*	Kurs i kommunal regi
L4	7	60	Kurs via Vitensenteret
L5	15	30	Videreutdanning programmering for grunnskolen.

4.3 Datainnsamling

Informasjonen fra informantene ble samlet inn ved at jeg på skolene S1, S2, S3 og S4 i forkant kom i kontakt med representative lærere. Her fikk alle lærere tilsendt informasjon via et samtykkeskjema (Se vedlegg: 9.1 Samtykkeskjema Lærer) og et informasjonsskriv som ble sendt ut til foresatte (Se vedlegg: 9.3 Informasjonsskriv observasjon). L5 fikk tilsendt samme samtykkeskjema. I tillegg ble det sendt ut et samtykkeskjema, som lærerne delte ut til elevene som ønsket å delta i intervjuene (Se vedlegg: 9.4 Samtykkeskjema elev).

Ved ankomst på skolene var elevene og læreren informert om hva jeg skulle og at jeg kom til å observere undervisningen deres. I hver av de observerte klassene startet jeg først med en observasjon. Data ble notert ned i observasjonsskjemaet og ble senere transkribert over i et digitalt skjema. I etterkant av observasjonene fikk jeg intervjue elever ved skole S1, S3 og S4. Disse foregikk individuelt på et grupperom. Hver av disse varte i omtrent 10 minutter. Helt til slutt gjennomførte jeg intervjuene med lærerne. Disse foregikk individuelt og på et avgrenset rom. Tidsspennet på disse intervjuene var noe mer variert og varte fra omtrent 20 til omtrent 60 minutter, alt ettersom hvor utdypende lærerne svarte.

4.3.1 Transkripsjon

Alle intervjuer og observasjoner ble transkribert til skriftlige notater. Observasjonene er transkribert etter rekkefølgen på hendelsene og sortert etter hvem som var involvert. Læreren og elevene ble anonymisert gjennom koder som «gr.» For gruppe og «E» for elev eller «G»/«J» for gutt eller jente. Da jeg transkriberte intervjuene valgte jeg å transkribere ord for ord slik de ble sagt under intervjuet. Underveis i intervjuene valgte jeg å utelate «vente»- eller «nøle»-ord i de situasjonene der disse ikke utgjorde noen forskjell på meningen av innholdet. Navn på personer og steder i området informanten holder til i har også blitt anonymisert, for eksempel ved bruk av betegnelsen «lærer». I noen av lydopptakene var det deler av innholdet som var utydelig eller vanskelig å oppfatte. Jeg tok i bruk to enheter der jeg brukte *Nettskjema diktafon app* (Universitetet i Oslo, 2021). I tillegg testet jeg begge enhetene flere ganger i forkant av intervjuene. Likevel var det noe støy fra opptaksenhetene. Disse delene har blitt utelatt fra transkripsjonen og er markert med koden «(utydelig)». Det meningsbærende innholdet er likevel ivaretatt. Totalt satt jeg igjen med 99 sider transkribert datamateriale. Der 39 sider var lærerintervjuer, 33 sider elevintervjuer og 27 sider fra observasjonene.

4.4 Analyse

I analysen av det innsamlede datamaterialet har jeg valgt å ta i bruk en analyseform som baserer seg på en datastyrt analyse med meningsfortetning eller analytisk reduksjon av innholdet i det innsamlede datamaterialet (Christoffersen & Johannessen, 2012; Malterud, 2011). Datamaterialet består av informasjon fra flere informanter og flere observasjoner. Ingen av disse ble foretatt over tid. Det ble derfor foretatt en tverrgående analyse på tvers av kildene (Malterud, 2011). Meningsfortetningen tar utgangspunkt i Malteruds (2011) fire steg for systematisk tekstkondensering av datamaterialer. Hun kaller det for analytisk reduksjon som i stor grad handler om å redusere datamaterialet ved å luke bort det som ikke er relevant og fremheve det som er relevant gjennom dekontekstualisering og gjennom rekontekstualisering av innholdet. Altså gjennom å dele opp innholdet i mindre meningsbærende enheter, for deretter å brukene svarene fra dette og sørge for at de stemmer overens med det opprinnelige innholdet. De fire stegene er henholdsvis 1) Helhetsinntrykk, 2) Meningsbærende enheter, 3) Kondensering og 4) Sammenfatning(Malterud, 2011). Til tross for at analysen i stor grad er datastyrt, vil det teoretiske grunnlaget for oppgaven også legge føringer for hvilke kategorier analysen fremhever underveis.

I det første steget av analysen har jeg gjort meg opp et inntrykk av det helhetlige datamaterialet og hvilken informasjon denne gir. Denne delen ga noen overordnede, temaer

som kunne gi en viss retning til innholdet i datamaterialet. I neste steg har jeg tatt for meg hvert enkelt datamateriale og trukket ut hovedtrekk fra disse. I både første og andre steg har jeg sett etter mønstre i informasjonen datamaterialet legger frem. Organiseringen av dette ble gjort ved først å skille ut relevant informasjon. Hovedtrekkene ble gjort om til foreløpige analysetemaer, for deretter å kode dette mer spesifikt underveis i analysearbeidet. Kodingen har som hovedformål å sette merkelapp på utsnitt av tekstene basert på informasjonen de gir, og er et hjelpemiddel for å organisere informasjonen (Malterud, 2011). Med utgangspunkt i kodeprosessen handler det tredje punktet i analysen om å kondensere meningsinnholdet i de kodede datamaterialene. Kodene blir kategorisert i overordnede grupper og om nødvendig undergrupper. Kodingen ble gjort ved hjelp av fargekoder. På denne måten vil de opprinnelige kategoriene for kodene utvikles videre og meningsinnholdet fortettes ytterligere. I den siste delen av analysen sammenfattet jeg innholdet fra kondenseringen og kodingen. Samt at jeg så på disse i lys av problemstillingen og i lys av det opprinnelige førsteinntrykket av analysearbeidet.

4.5 Validitet og reliabilitet

Reliabilitet handler om «Forskningsresultatenes konsistens og troverdighet» (Kvale & Brinkmann, 2019, s. 276). Reliabiliteten av resultatene handler om hvorvidt noe kan reproduseres med samme resultat ved et senere tilfelle. En kan stille seg spørsmålet; Vil en annen forsker komme frem til de samme resultatene? Oppgaven tar for seg kvalitativ data innhentet blant et begrenset antall lærere og elever. Informasjonen disse gir er subjektive oppfatninger av de spørsmålene jeg har stilt, samt at antallet informanter ikke er stort nok for å kunne generalisere eventuelle funn. Funnene kan fremmes for det gitte utvalget. På samme måte som at intervjuer er basert på subjektive oppfatninger, kan heller ikke observasjoner sies å være fullstendig objektive. Det kan være et sprik mellom lærerens intensjoner og observatørens observasjoner. Av den grunn valgte jeg å ta i bruk flere former for kvalitativ datainnhenting. Observasjoner vil til enhver tid være farget av observatørens subjektive oppfatning av en situasjon, og intervjuene kan dermed bidra som støtte for å se observasjonene mer objektivt. Det er under intervjuene og i behandlingen av informasjonen i etterkant viktig å være bevisst på at mange av tankene og meningene i et intervju er subjektive (Kvale & Brinkmann, 2019). Likevel kan variasjonen av informanter og kombinasjonen av de ulike formene for kvalitativ datainnhenting bidra til å skape et mer nyansert bilde av informasjonen som kommer til uttrykk.

Analysen av datamaterialet, kan ha vært noe påvirket av den teoretiske bakgrunnen for oppgaven. Etter gjennomføringen av den datastyrte analysen bemerket jeg meg at mange av de overordnede kategoriene hadde store likhetstrekk med den litterære bakgrunnen for oppgaven. Jeg gjennomgikk derfor kategoriene fra analysen, med utgangspunkt i hovedlinjene ved litteraturen som er brukt i forbindelse med oppgaven. Denne gangen kom jeg frem til stort sett de samme overordnede kategoriene. Det vil der for ikke være helt korrekt å si at analysen ene og alene er databasert. Dette kan forklares på flere måter. Den første grunnen til at disse kan sammenfalle, er at jeg i planleggingsfasen av intervjuene tok utgangspunkt i en del av litteraturen som er benyttet i denne oppgaven. Det vil derfor være naturlig at kategoriene overlapper hverandre. En annen grunnen kan være at jeg har vært noe påvirket av den teoretiske bakgrunnen for oppgaven, og at jeg ubevisst har søkt etter teorier i datamaterialet.

Validitet handler om «hvorvidt en metode er egnet til undersøke det den skal undersøke.» (Kvale & Brinkmann, 2019, s. 276). Under analysen er det en fare for at en, som Malterud (2011) uttrykker, utsetter dataen for reduksjonisme. Dette fordi informasjonen som i oppgaven fremtrer fra data legges frem løsrevet fra den opprinnelige konteksten. Det er derfor særlig viktig å være bevist på å være lojal ovenfor det informanten faktisk har uttrykt og de meningene vedkommende har hatt (Malterud, 2011). Dette kan gjøres ved at en i analyse arbeidet stadig vender tilbake til de opprinnelige transkripsjonene, og sørger for at informasjon ikke går tapt eller endres på veien til den reduserte versjonen av innholdet. I følge Malterud (2011) finnes det flere mulige tolkninger i arbeidet med analyse av kvalitative data. Teorien som ligger til grunn for temaet det forskes på, kan dermed bidra i analysen som støtte til å vurdere tolkningene. Kvale og Brinkmann (2019) uttrykker at det å sørge for et valid forskingsresultat baserer seg på en kontinuerlig validering av innholdet gjennom hele prosessen. På denne måten ønsker jeg å ivareta tekstens validitet og reliabiliteten i fremleggelsen av den innsamlede dataen.

Basert på forespørselen min etter lærere som brukte programmering i matematikkundervisningen, responderte en av lærerne, L4 på denne. L4 planla deretter en time for bruk av programmering i en matematikktime, men det kom etter hvert frem at dette ble gjort basert på forespørselen jeg hadde sendt, og ikke noe læreren hadde gjort tidligere. Det så dermed ut til at forespørselen hadde påvirket hvordan læreren hadde planlagt undervisningen. Dette kan være med på å påvirke validiteten av oppgaven, da resultatet basert på undervisningen til L4, og dermed også S4 kan ha blitt påvirket av hva jeg etterspurte.

4.6 Etiske hensyn og bemerkninger

Datamaterialet i denne oppgaven ble innhentet via lydopptak i tillegg til at noe av informasjonen ble hentet fra barn. Prosjektet er derfor meldt inn til NSD (Se vedlegg: 9.7 Prosjektbeskrivelse NSD), og godkjent av disse før dataene ble hentet inn. Lærerne som deltok, fikk informasjonsskriv og samtykkeskjema tilsendt rundt en uke før observasjoner og intervjuer skulle finne sted. I forkant av observasjonen ble både elever, foresatte og lærer informert om hva prosjektet går ut på og alle fikk mulighet til å kunne trekke deltakelsen sin uten at dette skulle gå ut over elevenes undervisning. Disse elevene ville da fortsatt delta i undervisningen, men observasjoner der disse elevene deltok ble ikke tatt med i notatene for observasjonene og dermed heller ikke i analysen. Over står det nevnt at det ble foretatt lydopptak av de fleste intervjuene. Elevene E3.1, E3.2 og E3.3 virket usikre på bruken av lydopptak og jeg valgte derfor ikke å ta lydopptak av disse intervjuene. Det var en etisk vurdering der jeg ikke ønsket å gjøre noe elevene var ukomfortable med. Disse intervjuene ble derfor transkribert fortløpende mens intervjuet foregikk. Jeg vil derfor heller ikke ta i bruk sitater fra disse, men heller en sammenfatning av informasjonen de har gitt. Som vist i *Tabell 4.1* , har det ikke blitt foretatt intervjuer av elevene i S2. Det ble kort frist fra første kontakt med L2 til observasjon av undervisningen, derfor valgte jeg av etiske årsaker å ikke foreta intervjuer av elever i denne klassen. Det ville blitt for kort svarfrist til at elevene og dermed foresatte kunne ta et informert valg om deltakelse i intervjuet. Alle informanter er, som nevnt, anonymisert og har fått tildelt koder fremfor navn. Jeg har av den grunn også valgt å ikke skille mellom skjønn i oppgaven, da jeg ikke ser på dette som informasjon som er relevant for å kunne besvare problemstillingen. Alle informanter blir derfor omtalt som «hen».

5 Resultater og Analyse

I denne delen vil jeg legge frem funnene fra analysen av datamaterialet. Jeg har valgt å legge frem analysen etter hovedkategoriene fra analysearbeidet. Disse danner de overordnede kapitlene for analysen, og er henholdsvis Formålet med undervisningen, Metoder og strategier, Lærerens rolle, Programmeringens rolle og Matematikk. Data fra både lærer- og elevintervjuer, samt observasjoner, har blitt tatt i bruk.

5.1 Formålet med undervisningen

I denne delen av analysen vil jeg trekke frem hva som ble formidlet som formålet med undervisningen. Begrepet «formål» viser i denne analysen til begrunnelsen for bruken av programmering, både generelt i undervisningen, men også i den observerte undervisningstimen. Formålet kan vise til hvorfor lærerne tar i bruk undervisningsopplegget de har valg, hvilke faglige koblinger lærerne har gjort, samt at det kan gi innblikk i hva lærerne forventer at elevene skal sitte igjen med.

Formålet med undervisningen kom både til uttrykk under observasjonene der lærerne blant annet formidlet et formål til elevene og i intervjuene. Det lærerne fortalte i intervjuene skilte seg fra formålet som ble formidlet til elevene. I tillegg kom det fram at det var betydelig mer som lå til grunn for bruken av programmering i undervisningen enn hovedformålet tilsa.

I denne delen vil jeg derfor først legge frem formålet som ble formidlet til elevene. Deretter formålet lærerne selv uttrykte, og hvilke kompetansemål de hadde knyttet til arbeidet. Til slutt vil jeg trekke frem det jeg tolket som underliggende formål for undervisningen. Disse ble ikke konkret uttrykt som formål, men ble trukket frem gjentatte ganger i intervjuene.

5.1.1 Formålet formidlet til elevene

Formålet for undervisningen ble formidlet til elevene på flere måter. Gjennom hva som konkret ble sagt til elevene og hva som stod skrevet på tavlen. Hos to av de observerte klassene, S1 og S3, var det skrevet «*koding*» tavlen. S2 hadde ikke skrevet noen plan for dagen, men det ble fortalt at hele dagen var satt av til «*koding*». S4 hadde skrevet «*matte*» på tavlen. Dermed var det èn av de observerte klassene, S4, som hadde en matematikktime og tre, S1, S2 og S3, hadde «*koding*». I alle klassene ble målet for timen lagt frem på en måte som fortalte noe om hva elevene skulle «lage» eller «gjøre» (Se tabell 5.1).

Tabell 5.1 Formål formidlet til elevene

	S1	S2	S3	S4
På tavlen	Koding	-	Koding	Matte
Program	Micro:Bit	Micro:Bit og Bit:bot	Scratch	Scratch
Kode	Blokkprogrammering	Blokkprogrammering	Blokkprogrammering	Blokkprogrammering
Oppgave	Lage et trafikklys ved hjelp av lysdioder og en Micro:Bit.	Programmere en Bit:bot til å følge en bane.	Lage spillet *Ørkenløp* i Scratch, et spill der målet var å komme først til mål (Code Club UK, Undated)	Lage en prosentkalkulator.
Fokus	Forklare nøye	-	Følge en oppskrift	Regne med prosent

Det var i tillegg til dette også noen mindre fokusområder som ble formidlet til elevene i undervisningen. S1 hadde et fokus på at de skulle forklare nøye. S3 fokuserte på at elevene skulle klare å følge en oppskrift. S4 hadde et fokus på å kunne regne med prosent.

5.1.2 Hovedformål uttrykt av lærere i intervju

Som nevnt over uttrykte lærerne et noe annet formål ved undervisningen i intervjuet enn det som kom til uttrykk under observasjonene. Ett av de mest fremtredende formålene med undervisningen var samarbeid og samarbeidslæring. L2, L3 og L5 trakk dette fram som et overordnet formål. Dette formålet lå til grunn for arbeidsmetodene som ble tatt i bruk i undervisningen. Dette kommer jeg tilbake til senere i analysen.

> *L3: (…)Altså målet, i utgangspunktet det ene er jo at vi har jobbet veldig mye med samarbeid. (…) Så det å få jobbe to og to sammen (…), har jo vært en utfordring i seg selv. Det første målet egentlig er mest den sosiale biten av det. Å jobbe 2 og 2 sammen, å klare å samarbeide. Det har de egentlig blitt veldig gode på, synes jeg. Som vi også kan bruke videre til andre ting. (…)*

I tillegg til at samarbeid var særlig fremtredende, oppga L1, L2, L4 og L5 andre hovedformål. To av lærerne, L1 og L4, hadde dybdelæring som formål. I L1s tilfelle skulle elevene arbeide videre med det de hadde lært tidligere om bruken av Bit:boter. Senere i intervjuet, i forbindelse med kompetansemålet, ble det uttrykt at dette formålet var å bruke løkker og ta dette «*litt videre*» (L1). L4 uttrykte at målet var at elevene skulle få større forståelse for regning med prosent. L2 hadde som hovedformål at elevene skulle bli kjent med Bit:bot, og å prøve og feile.

> *L4: (…) å øke elevenes forståelse rundt prosentregning. Nå har vi jobbet mye med brøk, desimaltall og prosent. Så sett litt på sammenhengen mellom de, også skal de jo klare å regne med disse uttrykksformene også. (…)*

I tillegg til samarbeidslæring, trakk L5 fram algoritmisk tenking som et hovedformål. Underveis i intervjuet knyttet hen algoritmisk tenking opp mot «logikk», «mønster» og «dekomposisjon», samt at hen trakk paralleller til arbeidsmetoden som ble brukt gjennom å fikle, skape og å «debugge (…) feil i matteoppgaver».

> *L5: Da er det litt med baktanke på den dere 21st Century skills (…). Med den godeste plakaten med algoritmisk tankegang som du ser (viser plakaten på veggen). (…) Som er hovedbiten og gjerne kombinert med samarbeid og samarbeidslæring. Ut ifra de målene som står i planen. Så jeg prøver å få til litt sånn at det skal lande litt på flere tuer. Det med samarbeidet setter jeg veldig høyt.*

Av informantene oppga altså L4 og L5 et matematisk formål med undervisningen, henholdsvis algoritmisk tenkning og prosentregning. L1 hadde et programmeringsrettet formål, og L2 og L3 hadde enten et sosialt formål.

Tabell 5.2 Hovedformål uttrykt av lærere

	L1	L2	L3	L4	L5
Hovedformål	Dybdelæring: Bruke tidligere læret kunnskap – Løkker.	Bli kjent, Prøve og feile, Samarbeid	Samarbeid	Dybdelæring; Prosentregning	Samarbeid og Algoritmisk tenking

5.1.3 Formål knyttet til kompetansemål

Læreplanens kompetansemål la grunnlaget for undervisningen hos flere av lærerne, da jeg spurte hvilke kompetansemål de hadde tatt utgangspunkt i. L1, L4 og L5 knyttet arbeidet opp mot kompetansemål knyttet til matematikkfaget, men det var kun L4 som knyttet arbeidet opp mot et kompetansemål knyttet til et matematikkfaglig konsept i tradisjonell form. L4 hadde tatt utgangspunkt i et kompetansemål som var knyttet opp mot bruken av prosent, og tok da i bruk programmering som metode for å nå dette målet.

L1 og L5 hadde knyttet arbeidet opp mot kompetansemålene som omhandlet konsepter knyttet til programmeringsarbeidet. L5 snakket mer overordnet, mens L1 rettet det spesifikt til undervisningsøkten, der målet var å både kjenne til, men også å kunne ta i bruk løkker.

> *L1: Vi brukte en gjenta alltid den har de brukt før, men det er en løkke, og de må lære vite at en løkke alltid da går om og om igjen (…).*

5.1.4 Underliggende formål

Funnene i analysen har foreløpig gitt størst uttrykk for tilknytning til matematikken gjennom kompetansemålene lærerne hadde lagt til grunn. Underveis i intervjuene kom det likevel kom fram at undervisningen hadde flere underliggende formål. Disse formålene ble ikke spesifikt nevnt som hovedformål, men ble gjentatte ganger trukket frem av lærerne.

L1 og L2 trakk frem at elevene skulle utvikle seg til å bli «kreative skapere» (L1). Dette kommer blant annet frem i samtalen med L1 og L2, da vi snakket om lærerens rolle.

> *L1: De (elevene) skal jo være de kreative skaperne etter hvert da. Det tar en stund før dem kommer dit også. Men først så er det, det blir jo flaskepåfylling i starten, så etter hvert får dem bli egne skapere.*

I: Og slippe dem løs?

L1: Slippe dem løs. Når dem har det dem trenger. Så er det noen som nærmer seg, men de er ikke helt. Helt der enda. (…) Det er ikke mye koding som de gjør på egenhånd da, da er det mer å sette sammen puslebiter.

Både L1 og L2 sier «etter hvert» og viste til at det først var behov for «flaskepåfylling» (L1). «flaskepåfylling» kan forstås som opplæring i de grunnleggende ferdighetene og kunnskapene innen programmering. Altså kan det se ut til at det er et formål å lære elevene strategier for å tenke og løse problemer selvstendig, men at det på veien dit blir nødvendig å lære seg noen grunnleggende programmeringsferdigheter. Parallelt med dette forteller L1 at hen utfordrer elevene med oppgaver som krever utforsking i form av utvikling av spill og oppgaver med åpne rammer.

I delkapittelet *Formål uttrykt av lærere i intervju*, ser vi at L5 sier at formålet er å lære seg algoritmisk tenking. Til tross for at de andre lærerne ikke trekker dette frem som første punkt på listen sin, kommer algoritmisk tenking likevel til uttrykk hos de aller fleste utover i intervjuet. L4 uttrykker for eksempel at algoritmisk tenking er en av grunnene til at programmering har en viktig plass i matematikkfaget.

L4: (…) Og det er noe med å lære seg den algoritmiske tankegangen, at sånn, sånn, sånn. For de er ikke de vant til å tenke på den måten. I tillegg til den problemløsingserfaringen de får, og utforske muligheten. Så jeg synes egentlig at det er fint, for jeg tror det blir veldig fin mulighet for ungene til å bli mer selvstendige og være mer kreative og utforskende. (…)

L3 trekker frem «algoritme» som et viktig matematisk konsept, og uttrykker at det alltid kommer inn i arbeidet med programmering. Hen legger vekt på å følge instruksjoner og forstå systemer. Videre trekker både L3 og L5 frem at elevene får øvd seg på utholdenhet i arbeidet med programmering. Utholdenhet knyttes opp til det å ikke gi opp når noe ikke blir riktig, men prøve igjen. Dette kan trekkes videre inn i matematikkfaget. Både algoritme og utholdenhet kan tolkes som henvisning til algoritmisk tenking.

L3: (…)Tenker at det er å følge, og skjønne systemet da. Det er veldig viktig. Det å følge instruksjoner og følge og skjønne systemet. Og det å gå tilbake å lete og det å kunne stå lenge i en oppgave. For unger har, i hvertfall min opplevelse er, at unger har mindre tålmodighet nå enn lenger tilbake i tid liksom. Det å stå lenge i en

oppgave. For det blir jo ikke ferdig på én, to, tre. Det tar tid. Det er kanskje den største utfordringen, og det tenker jeg vi også kan bruke i matematikken. For det å få en problemoppgave over tid. At man øver på det da.

Motivasjon er en fremtredende faktor hos flere av intervjuene. Arbeidet med programmering skulle fungere som et «*motiverende avbrekk*» (L1) og ble omtalt som «*lystbetont*» (L2) og «*moro*» og «*veldig gøy*» (L3). L1 uttrykte også i intervjuet at oppleggene skulle fenge elevene, og at opplegg som ikke fanget elevenes interesse, gjerne ikke ble brukt igjen.

Tabell 5.3 Underliggende formål oppfattet fra intervju

	L1	**L2**	**L3**	**L4**	**L5**
Underliggende formål	Elevene skal bli kreative skapere. Motivasjon	Elevene skal bli kreative skapere. Motivasjon	Algoritme, Følge instruksjoner, Forstå systemet, Utholdenhet Motivasjon	Algoritmisk tenking	Utholdenhet

5.1.5 Elevenes oppfatninger

Alle de intervjuende elevene, i de tre klassene S1, S3 og S4, kunne i intervjuet gjenfortelle det de hadde arbeidet med i undervisningen. De kunne fortelle noe om hva de gjorde, og hvordan de gjennomførte oppgavene. I tillegg kunne de fortelle at de likte å arbeide med programmering. Elevene hadde i stor grad plukket opp det som av lærerne ble formidlet som formålet. Elevene nevnte hverken samarbeid eller algoritmisk tenking som et læringsmål, men enkelte kunne fortelle at de hadde samarbeidet og at de trivdes best med samarbeid i programmeringskontekst. E4.3 viste til motivasjon da eleven uttrykte at *«(…) men læreren vil liksom ikke at vi skal kjede oss liksom, for vi sitter jo veldig mye på rompa liksom. Så jeg tror hen ville at vi skulle gjøre noe liksom litt gøy.*».

5.2 Metoder og strategier

Underveis i observasjonene og i intervjuene, ble det gitt uttrykk for ulike undervisningsmetoder og strategier for å løse oppgavene som lærerne og elevene tok i bruk. Metoder ble bestemt av lærerne og strategier elevene tok i bruk legges i denne oppgaven inn

under samme kapittel. Metodene kan ha innvirkning på hvilke strategier som blir tatt i bruk. Både metodene lærerne legger til rette for og strategiene elvene tar i bruk kan si noe om hvorvidt det legges til rette for å arbeide med matematisk kompetanse. Det kan også si noe om hva lærerne gjør for å knytte programmeringsarbeidet til det matematikkfaglige innholdet.

Hos S1, S2, S3 og L5 tas det i bruk samarbeid. Dette gjenspeiler det L2, L3 og L5 i Kap. 5.1.2 Hovedformål uttrykt av lærere i intervju formidlet som hovedformål. Hos S1, S2, S3, S4 og L5 ble det i tillegg tatt i bruk klassesamtaler. I denne delen vil jeg først trekke frem de arbeidsmetodene lærerne la til rette for. Deretter vil jeg trekke frem de strategiene som ble tatt i bruk. Dette kapittelet tar i stor grad utgangspunkt i data hentet fra observasjonene, med innspill fra elevintervjuer og lærerintervjuer.

Tabell 5.4 Viser oversikt over hvilke metoder som ble tatt i bruk under observasjonene og hvilke metoder lærerne nevner under intervjuene.

Metode	S1	S2	S3	S4	L1	L2	L3	L4	L5
Samarbeid	X	X	X		X	X	X	X	X
Klassesamtale	X	X	X	X	X		X		X

Tabell 5.5 Viser oversikt over hvilke strategier som ble tatt i bruk under observasjonene og hvilke strategier lærerne nevner under intervjuene

Strategi	S1	S2	S3	S4	L1	L2	L3	L4	L5
Prøve og feile	X	X	X			X			X
Oppskrift/ følge instruksjoner			X		X		X		
Feilsøke	X	X	X		X		X		X
Tenke fremover	X	X							
Sammenlikne				X					
Forklare	X	X	X	X	X	X	X	X	X

5.2.1 Lærernes tilrettelegging; Samarbeid og klassesamtaler

I de observerte undervisningssekvensene ble det lagt opp til samarbeid mellom elevene, men i noe varierende grad. Hos S4 la læreren opp til at elevene, ved èn anledning, skulle snakke sammen i par to og to. Ut over timen arbeidet elevene individuelt. S2 og S3 satte elevene

sammen i par med beskjed om å samarbeide. Hos S1 ble elevene satt i grupper på fem. Det ble dermed lagt til rette for samarbeid i tre av klassene, henholdsvis S1, S2 og S3. I tillegg ga både L5 og L4 uttrykk for at dette var noe de pleide å legge til rette for.

> *L4: (…) Når jeg har programmert tidligere, så har de fått jobbe to og to. Eller tre og tre. Smågrupper eller par. Fordi det også er utfordrende å finne problemene alene. Og du får jo ikke den praten, sånn den dynamikken i det. Så videre kommer jeg til å bruke mye mer det(…).*

I tillegg til å legge til rette for samarbeid, ga L5 også uttrykk for at hen brukte elevene til å hjelpe medelever, ved å bruke det hen kalte for *«Kan litt mer»-elever* (KLM-elever).

> *L5: Da har vi hatt litt KLM-elever, sånn kan litt mer elever, som har fått lov til å hjelpe til(…) vi driver litt med den samarbeidslæringa(…).*

Elevene selv uttrykker også at de setter pris på samarbeidet, og det ser ut til at alle elevene, bortsett fra E1.2, ser ut til å foretrekke muligheten til å arbeide sammen med en medelev. Elevene uttrykker at de liker å ha muligheten til å utveksle idéer.

> *E1.1: (…) Men når man har med andre som utbytte ideer også kan man har tenkt på at det hadde vært kult eller den tenker på at det hadde vært kult som du aldri har tenkt at du kan gjøre.*

5.2.1.1 Samarbeid leder til erfaringsdeling

Det at elevene ble satt sammen i par, så ut til å bidra til at elevene utvekslet ideer og erfaringer. De delte erfaringer innad i gruppene, men også på tvers av grupper.

Et eksempel på dette er å finne hos S2 der en av elevene oppdager at banen er formet som et rektangel. Dette førte igjen til at deres valg av løkken «*gjenta 4 ganger*» ikke kunne fungere for å få bit:boten til å bevege seg etter banens form. Eleven formidlet oppdagelsen videre til samarbeidspartneren før de sammen endret koden.

5.2.1.2 Klassesamtalene oppfordrer til å forklare

I alle de observerte undervisningssekvensene foregår det både innledningsvis og avslutningsvis felles klassesamtaler. I klassesamtalene rettes oppmerksomheten blant annet mot de faglige konseptene som kommer til uttrykk. Dette kommer tydeligst frem hos S1, S2 og S4, men jeg finner også eksempler hos S3. Lærerne uttrykker også selv at klassesamtalen

brukes for å gjøre innholdet eksplisitt. L3 legger også vekt på den matematiske samtalen, særlig i etterkant av arbeidet for at programmering skal kunne tas i bruk som metode.

> *L1: Jeg må spørre dem rett og slett. Her har jeg 2000 millisekunder. Hva betyr det? Hvor lenge er det? Og da får jeg som regel svar, og ofte er det riktig og. Så vi har jobbet med det før, og de har jo mikser i omgjøringer av alle slag, så da. (…)*

Elevene ble oppfordret til å forklare hva begreper og tall betydde. For eksempel dukket begrepene «millisekunder», «sekunder» (S1 og S2) og «prosent» (S2 og S4) opp. 360, 180 og 90 graders rotasjon ble også tatt i bruk og forklart (S2). Elevene tok i bruk både bevegelser og ord for å forklare begrepene.

5.2.1.3 Klassesamtalene legger til rette for sammenlikning

Gjennom klassesamtalene ser det også ut til at det skapes en sammenlikning mellom konsepter i matematikk og kodene elevene lager. Dette kommer særlig til uttrykk hos S4. Der læreren underveis i klassesamtalen stadig oppfordrer til å se likhetene mellom koden på tavlen og algoritmen elevene har brukt tidligere for å regne med prosent. L4 brukte tid på å se på likheten mellom symbolene som ble brukt, regnerekkefølgen og tallverdiene. Elevene ble først oppfordret til selv å se på likhetene ved å stille spørsmålet «*Hva er det i denne koden som gjør at vi kan finne rabatten?*» (S4). Deretter sammenliknet læreren de to algoritmene sammen med elevene, trinn for trinn.

I dette delkapittelet viser analysen at metodene samarbeid og klassesamtale åpner for at elevene kan dele erfaringer, forklarer begreper og åpner for sammenlikning mellom matematikk og koden som tas i bruk.

5.2.2 Strategiene elevene tok i bruk

Strategiene elevene tok i bruk kan fortelle noe om hvorvidt elevene arbeider med matematiske kompetanser eller ikke, i tillegg kan de fortelle oss noe om hvordan de arbeider med matematisk kompetanse. Denne delen tar derfor for seg de strategiene som kom til uttrykk i undervisningen, samt hvilke strategier elevene selv uttrykte at de tok i bruk.

5.2.2.1 Utforske ved prøving og feiling

En av strategiene som gikk igjen hos S1, S2 og S3, var at elevene «prøvde og feilet». De testet ideene sine gjennom koden og fikk tilbakemelding på om det fungerte gjennom hvordan koden reagerte. Det var det stadig refleksjoner om hva som fungerte og hva de bør endre på. L1 oppfordret underveis i arbeidet elevene til å «prøve og feile» dersom de ba om hjelp, i

tillegg uttrykte elevene selv at «*Nå skal jeg prøve og feile*» underveis i arbeidet. E1.2 viste til denne strategien da hen fortalte hvordan hen hadde arbeidet med å løse trafikklys-oppgaven.

> *E1.2: Ja, jeg prøver og så feiler, også prøver jeg igjen på en annen måte også får jeg det til.*
>
> *I: (…) Måtte du det i dag?*
>
> *E1.2: Ja, det måtte jeg.*
>
> *I: Når måtte du det?*
>
> *E1.2: Jeg måtte det når jeg skulle ta sånn rekkefølge. Så måtte jeg endre litt fordi jeg tok at gult, også kom grønt og så rødt.*

L2 uttrykte under observasjonen at det å «prøve og feile» er nært knyttet til matematikken, men at det er noe en ikke øver så veldig mye på. Dette var en henvisning til arbeidet elevene holdt på med, og hvordan de tok tak i oppgaven.

5.2.2.2 Følge instruksjoner

Hos S3 tok elevene utgangspunkt i en oppskrift. Oppskriften ble brukt undervis i arbeidet både for å vite hva de skulle gjøre, men også for å feilsøke dersom koden ikke fungerte. Elevene gikk tilbake i oppskriften og gikk trinnvis gjennom denne samtidig som de sammenliknet med koden de hadde satt sammen. Likevel var det ikke alle elevene som så ut til å ta i bruk denne strategien. Et eksempel er en elev som ønsket å slette alt og starte på nytt da det viste seg at koden ikke var riktig. Da dette ikke var et alternativ for samarbeidspartnere, gikk eleven bort fra arbeidet.

S3 var den eneste klassen som tok i bruk en «fysisk oppskrift» på papir. Likevel kan det se ut til at alle de fire observerte klassene la til rette for bruken av en form for instruksjon/oppskrift da alle de fire lærerne hadde satt opp en kode på tavlen som elevene skulle følge. Helt eller delvis. Trinn for trinn stod hver enkelt blokk oppført på tavlen. Forskjell lå i til hvilken grad elevene skulle endre den oppførte koden.

5.2.2.3 Feilsøke

L3 trekker i intervjuet paralleller mellom det å feilsøke i programmeringsarbeidet og det å feilsøke når en regner matematikkoppgaver. Hen trekker frem likhetene mellom arbeidet elevene gjør i timen, der de stadig vender tilbake til oppskriften, og feilsøking i arbeidet med tradisjonelle regneoppgaver.

L3: (…) Jeg synes det går veldig mye på det å forstå systemene på en måte. Å forstå hva man egentlig driver på med da. Men også det som jeg sa i stad. Det å stå lenge nok i et problem og det å gå tilbake etterpå og finne feilene. Gå å feilsøke. Det synes jeg programmering er et veldig nyttig verktøy til. Som de kan dra med seg inn i matematikken etterpå. For det er, det hender jo at ting blir feil, når de regner. Og at de kan gå tilbake og finne ut av hvor du har gjort det hen. Det er noe man får øvd mye på i programmering.

5.2.2.4 Tenke fremover/ forutse

En annen strategi som ble brukt hos S2 og S3 var at elevene underveis i arbeidet prøvde å forutse hva som kunne skje. Hos S2 ble det for eksempel observert en gruppe som snakket sammen i flere minutter før de begynte å endre på den opprinnelige koden. I samtalen diskuterte de tiden det ville ta å få Bit:boten til å kjøre 1 meter frem. Disse elevene klarte å få bit:boten til å kjøre 1 meter frem på første forsøk. Andre grupper valgte å teste koden først, deretter å endre kodens sekunder trinn for trinn til det ble riktig lengde. En annen måte å forutse løsninger på, var å visualisere retninger ved hjelp av håndbevegelser.

I samtalen om L1 sine tanker om implementering av programmering i matematikk, trakk hen frem behovet for kunne tenke fremover, og at dette er noe elevene har et behov for å øve på. Gjennom å la elevene bevege seg etter gitte mønstre, kunne elevene øve på det å følge en angitt rekkefølge.

L1: (…)å tenke rekkefølge framover. Det sliter mange med. (..) Vi har et rutenett med dette her i gangen, og der legger jeg ut og små brikker med gaver man skal innom. Steiner man skal gå rundt og så mål og start, og så får du en brikke av meg til å legge og piler som går så og så mange. og jeg klarer helt fint i hodet mitt å tenke ok, nå står jeg der også står jeg der. OK hvis vi snur sånn så står jeg sånn, men elever må ofte faktisk gjøre det for at de skal få skjønne hvor dem er hen i rekkefølgen sin. Og det tenker jeg. Det er en øvelse som mange av oss voksne hvertfall de som har holdt på med matte en stund klarer ganske fint, men som du får inn. Ganske mye øvelse på en fin måte da.

I dette delkapittelet viser analysen at elevene tok i bruk strategier som «prøve og feile», følge oppskrifter, feilsøke og å forutse løsninger.

5.3 Lærerens rolle

Hvilken rolle læreren får i arbeidet med programmering kan påvirke hvordan kommunikasjonen mellom læreren og elevene blir, og hva som blir formidlet. I tillegg kan dette også fortelle noe om hvordan programmeringen brukes. Denne delen er i stor grad basert på lærernes intervjuer. Jeg vil først se på hvilken rolle lærerne gir uttrykk for at de har, deretter vil jeg se på hva de legger til grunn for valg av oppgaver og metode.

5.3.1.1 Læreren som veileder

L1 fortalte at hens rolle var å gi «ideer til videre utvikling». L2 uttrykte at rollen bestod av å «*observere og gi tilbakemelding*». L3 og L4 ga uttrykk for at deres rolle var å «veilede». I tillegg uttrykte L3 at de skulle «hjelpe» og «stille kritiske spørsmål». L5 uttrykte at hen skulle «*hjelpe*» og «*sørge for at alle har kommet i gang*». Lærerne skal altså veilede og føre elevene videre i utviklingen sin ved hjelp av blant annet spørsmål. På denne måten ønsker de å bidra til at elevene skal utvikle seg til selvstendige tenkere.

> *L4: Veilede og støtte, tenker jeg er hovedrollen. Jeg ønsker jo at de skal kunne finne sine problemer selv etter hvert at de skal klare den biten. Så tenker jeg det blir jo når de blir mer erfarne (ler). Først da lærer de hvordan det funger også kunne veilede de mer da. (…)*
>
> *L4: (…) Men jeg tenker en veiledende funksjon sånn fremover. Eller det er det jeg ønsker da. For at de skal lære seg, ja den algoritmiske tenkningen og diskusjonen i det og at det hører med.*

L3 formidler at det er viktig å inneha nok kunnskaper knyttet til programmering, og uttrykte at *«læreren må ha en viss kompetanse i det*» for at programmering skal kunne brukes som metode i matematikk. I tillegg gir både L2 og L4 uttrykk for et behov for erfaring for å kunne spesifisere læringsutbyttet ved bruken av programmering. Dette kan muligens knyttes til tryggheten lærerne opplever i undervisningen. L4 uttrykte at enkelte undervisningsopplegg ikke hadde fungert grunnet manglende trygghet. Tryggheten påvirket, ifølge L4, til hvilken grad hen kunne veilede elevene.

> *L4: Det har nok vært jeg som har vært veldig jeg har ikke, jeg har nok ikke vært trygg nok selv.(…). Til å på en måte veilede de godt nok underveis og på en måte sette de i gang på en god måte. (…) Det er nok litt den tryggheten man får selv når man gjør det noen ganger. Kunne hjelpe de litt underveis. Det er ikke alle som er like gode problemløsere.*

5.3.1.2 Et samarbeid mellom lærer og elev

L5 legger vekt på samarbeidet mellom elevene og læreren og viser til en endring i de tradisjonelle lærer-elev rollene. Der det det nå handler mer om et samarbeid mellom læreren og elevene. Likevel er det fortsatt læreren som «er trygg og god på det» når hen viser til det matematikkfaglige innholdet.

> *L5: (…) Noen ganger er det elevene som hjelper læreren, andre ganger er det motsatt, men det er et godt og morsomt samarbeid synes jeg.*

> *L5: (…). Elevene ser at det faglige her, sant i matematikken det er det læreren som er trygg og god på det, så det, hva skal jeg si, alt det utstyret og de tinga som vi har i matematikk da er jo med på å gjøre matematikken vår litt mer livlig, litt mer samarbeidsvillig enn tradisjonell matte før det her ble en del av innholdet. Håper jeg. (…)*

L2 viste til et eksempel der elevene hadde oppdaget muligheter i programmeringsarbeidet som hen selv ikke var kjent med. Hendelsen tar utgangspunkt i at elevene tok i bruk en løkke, med gjenta 4 ganger. Hendelsen viser hvordan elevene fant andre løsninger enn læreren.

> *L2: Så så vi en av de hadde jo lagt, eller funnet den gjenta fire ganger. Den hadde ikke jeg tenkt på. Det hadde jeg ikke sett før, direkte. (…) Det er jo sånn typisk da at noen tar i bruk ting man selv ikke har sett eller tenkt.*

5.3.1.3 Planlegging av undervisningen

L3 og L4 ga uttrykk for at elevene bør ha vært gjennom noe av det matematiske innholdet før de kan arbeide med det i en programmerings kontekst. Hva elevene hadde vært gjennom tidligere la føringer for hvilke oppgaver lærerne kunne ta i bruk i undervisningen.

> *L3: (…) at jeg har prøvd det selv og at det har noe for seg for det vi skal drive med. En del av det som går på matematikk, går jo på sirkler og sånt. Og vi har ikke jobbet noe med det, så det er liksom litt, okay de kan jeg bare ta bort. (…) Det er jo litt det målene, at vi skal ha vært gjennom det vi driver på med da.*

L3 uttrykker senere i intervjuet at valg av oppgaver er med på å avgjøre og legge til rette for at elever skal kunne bruke programmering som metode. Hen uttrykker behovet for «*de gode oppgavene*». L2 uttrykker at programmeringsarbeidet kan knyttes til matematikkfaget, men at dette kommer an på hvordan en veler å legge opp arbeidet som lærer. «*Kommer an på hva du legger programmeringen til da. Du kan jo legge det veldig likt det du tenker som matte» (L2).*

Ved spørsmål om L4 ville brukt programmering mer knyttet til matematikkfaget uttrykker hen, at hen ikke ønsker å bruke det bare for å bruke det, men at hen ønsker at programmeringsarbeidet skal bidra til dybdeforståelse. Dette kan knyttes til valg av oppgaver.

> *L4: Jeg har lyst til å få det til mer, så jeg har veldig lyst til å få lære meg gode strategier til å kunne bruke det i matematikkundervisningen fornuftig. Ikke bare sånn for å gjøre det liksom. For det er jeg ikke interessert i. Men å kunne bruke det sånn at det faktisk gir elevene en dypere forståelse.*

L3 uttrykker videre i intervjuet at det også er lærerens rolle å trekke inn matematikken, og at en del av å gjøre dette er å trekke det frem gjennom «*den matematiske samtalen*».

> *L3: (…) men det er å prøve å dra inn der matematiske man kan da. Og kanskje ha den matematiske samtalen i etterkant. Hva var det vi gjorde? Hva lærte vi? Hvorfor ble det sånn? (…)*

Tre av lærerne, L1, L3 og L4, trekker frem verdien av det å teste undervisningsopplegget selv før det introduseres for elevene. På denne måten kan de forsøke å forutse hvilke utfordringer elevene kan møte på, og hvilke problemstillinger de kommer til å stå ovenfor i arbeidet.

Lærerne uttrykker altså at de skal fungere som veiledere, hjelpe elevene i gang og støtte disse i arbeidet de gjør. Videre uttrykkes det at det forutsetter at lærerne har programmeringskompetansen til grunn og er trygge på det de skal veilede elevene i. Samtidig vises det til en rolleendring fra den tradisjonelle matematikken der det nå i større grad skal være et samarbeid mellom læreren og eleven. Videre uttrykker lærerne at en viktig del av deres rolle blir å være godt forberedt ved at de har prøvd ut undervisningsopplegget de skal bruke, samt at de har i oppgave å velge «de gode oppgavene» som legger til rette for bruk av matematikk.

5.4 Programmeringens rolle

Dette delkapittelet forteller noe om hva lærerne bruker programmeringen til, og kan også si noe om hvordan de knytter programmering og fag sammen. På denne måten tar dette kapittelet for seg hvilken rolle lærerne tildeler programmeringen. I likhet med de andre analysekapitlene er også denne delen delt inn etter kategorier som kom til uttrykk i datamaterialet. Den første kategorien tar for seg programmeringens visuelle funksjon. Den andre kategorien viser til lærernes bruksområde for programmeringsarbeidet. Den siste kategorien har sin tilknytning til Kap. 5.1.4 Underliggende formål, og viser til motivasjon i

tilknytning til programmeringsarbeidet. Funnene i denne delen av analysen er basert på funn fra intervjuene. Noen deler er basert på konkrete svar på hva lærerne anser at er programmeringens rolle. Andre funn er i større grad basert på innspill underveis i samtalene.

1.1.1 Alternativt og visuelt medium som åpner for dybdelæring

To av lærerne, L1 og L4, uttrykker at programmeringen er et «*alternativt medie*» (L1) eller et «*multimodalt verktøy*» (L4) som brukes i tillegg til andre medier for å arbeide med læring.

> *L1: Det er et alternativt medie, som jeg kommer til å bruke en del i tillegg til I-pad og i tillegg til mattebok og i tillegg til alt vi finner på.*

L1, L4 og L5 formidler at programmeringen brukes som et dybdelæringsverktøy, som igjen baserer seg på programmeringens visuelle funksjon. L4 trekker også frem at elevene arbeider med matematikken på nye måter som gir nye perspektiver på matematikken tatt i bruk. L5 forteller at programmeringen visualiserer «*det matematiske innholdet*», mens L1 uttrykker at programmeringsarbeidet kan gi ulike vinklinger på det faglige innholdet.

> *L4: (…) Jeg ser jo mange muligheter for å kunne bruke det i matematikk. (…) Så jeg tenker jo at det er en kjempefin mulighet til å lære, altså dybdeforståelsen av at de skal jobbe med det og forstå flere perspektiver på det, og ikke bare sitte å regne i matteboken alltid. Men det er jo det som lærer å lære å bruke det fornuftig da.*

1.1.2 Brobygger mellom fag

Alle lærerne gir uttrykk for at de hovedsakelig benytter programmeringsarbeidet til tverrfaglig arbeid, der oppgavene eller aktivitetene ikke nødvendigvis kun havner innenfor et bestemt fagområde. L1, L4 og L5 gir samtidig uttrykk for at denne tverrfagligheten kan fungere som en brobygger mellom fagene.

> *L5: (…)Det er det en får selv, det er litt som i Finland da, jeg tror vi sliter litt med sånne tette skiller mellom fagene i skolen. Det skulle vært myket opp litt der. Vi må nok litt mer på tvers enn vi har tradisjonelt gjort. Tror jeg. Og det tror jeg kanskje vi kan få til med litt programmering.*
>
> *I: Så det er litt sånn brobygging mellom fagene også da?*
>
> *L5: Ja. Også er det jo det å se sammenhenger også er jeg, det er veldig viktig at den skolen vi har den er ikke der for at du skal være der, men det er for at den skal forme*

deg for at du skal ut i arbeidslivet senere. Det er det som er min drivkraft, fordi det er elevene. Det er for at dem skal rustes for et til et liv senere og være kompetente der.

1.1.3 Motivasjon som vei inn i faget

I Kap. 5.1.4 Underliggende formål viser analysen at motivasjon er et underliggende formål for bruken av programmering. L1 uttrykker også at hen tar i bruk programmering som et «*motiverende avbrekk*» og viser til at motivasjonen kan sørge for at programmeringen på sikt kan få en større rolle. Både L1 og L5 uttrykker at programmeringens motivasjonsaspekt kan skape et matematikkfaglig grunnlag.

> *L1: (…) Når du får et produkt ut av det så er det jo få et produkt ut der, så er det jo. Da er det ikke bare gjøre matte for mattens skyld, noe som mye av matten er. Vi gjør en oppgave også, hva skal vi med den. Mens når man har programmering på den måten her, så er det, så har du et mål med det du gjør. Som blir til noe, ikke bare en lærer skal stille et spørsmål også blir det ferdig.*

L2 fremhever at elevene får «ting» til, og at de skal utforske og finne ut av «ting». Samtidig knytter hen dette opp mot at det er «lystbetont» og «moro», og konkluderer med at «*Det er jo lek og forskning*».

5.5 Matematikk

Underveis i de observerte undervisningssekvensene ble det, som nevnt, i Kap. 5.2 Metoder og strategier trukket frem noen matematiske konsepter. Konseptene ble trukket frem i klassesamtalene og i samarbeidssituasjonene. I dette kapittelet vil jeg se på de matematiske konseptene som ble observert, samt de matematiske konseptene som lærerne nevner i intervjuene. I tillegg vil jeg se på hva elevene opplevde at de hadde arbeidet med av matematiske konsepter. Tabell 5.6 viser en oversikt over matematiske konsepter som, med varierende grad, kom til uttrykk. Tabellen viser hvilke konsepter som ble observert hos klassene S1, S2, S3 og S4, samt hva lærerne L1, L2, L3, L4 og L5 omtalte.

Tabell 5.6 Matematiske konsepter som kommer til uttrykk gjennom observasjoner og intervju av lærere

Matematiske konsepter	S1	S2	S3	S4	L1	L2	L3	L4	L5
Måling Tid, størrelse/ lengde	X	X	X		X	X	X		X
Tall og regning			X	X	X		X	X	X
Koordinatsystem			X				X		X
Geometri	X				X		X		X
Sannsynlighet					X		X		X
Statistikk					X				X
Utforsking og problemløsing	X	X			X	X	X	X	
Algoritmisk tenking	X	X	X				X	X	X
Representasjoner/modellere	X				X		X	X	X

5.5.1 Matematiske konsepter formidlet i intervju med lærerne

Lærerne formidlet i intervjuene hva de anså som læringsutbyttet ved den observerte undervisningssekvensen, hva de anså som det matematiske læringsutbyttet generelt ved arbeidet med programmering, og hvilke matematikkfaglige kompetansemål de mente en kunne nå ved hjelp av programmering. De neste delkapitlene tar utgangspunkt i disse punktene og baserer seg på hva lærerne selv sa.

5.5.1.1 Matematikk med utgangspunkt i kompetansemål

L1, L3, L4 og L5 så muligheter knyttet til å bruke programmering som metode for å arbeide med matematikk. Lærerne fikk utdelt kompetansemål fra læreplanen som tilhørte det trinnet de underviste. I tillegg ble de spurt spesifikt om det matematiske innholdet i kompetansemålene omtalt i Kap. 1 Innledning.

L1, L3, og L5 så muligheter for å bruke programmering for å nå kompetansemål i matematikkfaget. Med kompetansemålene foran seg ramset de opp hvilke matematiske konsepter de så muligheter for å nå ved å arbeide med programmering, samt at enkelte fortalte hvordan de så for seg å arbeide med disse konseptene.

L1 så på kompetansemålene for 5.trinn og uttrykker at en både kan arbeide med sannsynlighet, brøk, prosentregning, formulering og løsning av et problem, likninger samt statistikk ved å bruke programmering. I tillegg refererer L1 til bruken av ulike

representasjonsformer, og det at elevene kan bruke Micro:biten til å representere både brøk og sannsynlighet ved hjelp av denne. Videre knyttet hen programmeringskonseptene variabler, vilkår og løkker til matematikken. L1 forklarer en sammenheng mellom variabler og algebra, og uttrykker at bokstavene blir et uttrykk for en verdi. Hen trekker frem vilkår, og knytter dette til «krokodilletegn», eller det vi i matematikken kaller for «ordningsrelasjon» og forklarer at elevene på den måten får arbeide med å sette verdier opp mot hverandre. L1 trekker også frem at løkker brukes i matematikken når noe skal gjøres gjentatte ganger. Hen ga videre også uttrykk for hvordan hen ville arbeide for å nå målet knyttet til det å «*utforske data og datasett*». Der hen vill brukt micro:bit til a registrere data som støy, lys eller temperatur, og overført disse dataene til et ferdig utfylt Excel-ark, med mulighet for å se grafen for dataene.

L3 så på kompetansemålene for 4.trinn, og trekker frem konsepter som de fire regneartene, strategier for divisjon og arbeid med tallmønstre. Videre trekker hen frem utforsking og sammenlikning av to- og tredimensjonale figurer og den overordnede kategorien geometri, som hen selv uttrykker at hen har brukt i undervisningen tidligere. I tillegg trekker hen frem ikke-standardiserte måleenheter.

I forbindelse med kompetansemålene som omhandlet programmering kunne L3 knytte disse opp mot matematiske konsepter som gjentakelser, geometriske figurer og mønstre. Videre knytter hen «*tabeller og datasett*» til opplegg hen hadde brukt tidligere, der de matematiske konseptene var rettet mot gangetabellen, der denne ble regnet som tabell.

L5 uttrykker at matematikken *«veier tyngst i programmering*». Hen tar utgangspunkt i de fire kompetansemålene som spesifikt er knyttet opp mot programmering, og forteller at algoritmisk tenking og regning med divisjon har flere fellestrekk. Et av disse er «*oppskriften*». Samtidig forteller hen at programmeringen bidrar til å gjøre matematikken visuell, som hen igjen forbinder med statistikk og sannsynlighet samt geometri.

> *L5: (…) Du er nødt til å på en måte ha algoritme, en ordning etter hverandre for å kunne løse det delestykket. Og da vise dem på en måte en algoritme i form av blokkprogrammering, og si det at her kunne jeg egentlig byttet ut den teksten her med når du deler (…) Og da vil jeg jo si at vi ufarliggjør og visualiserer litt av det matematiske innholdet på en annen måte. Som vi kanskje ikke har hatt muligheten til før. Også er det jo det her med, ja, data sett da, for så er det jo veldig mange som har fått følgende oppgave i matematikk at du skal ta med deg en terning også skal du sitte*

å kaste den terningen 100 ganger også skal du notere det i den boken din. Ved hjelp av den microbitten kunne lage, du skal velge noe random mellom 1 og 6 her også ved bruk av tabellen kan du gjenta dette 100 ganger, så har du løst terningkastet (…)Det er jo på en måte å forstå da, den verden vi er basert på i dag da. Der det ligger mye data og algoritmer bak (…). Og det er med sirkulering, visualisering, det å kunne lage noe visuelt da. Å se, å sitte, og dra ut et kvadrat i den thinkercaden, det å dra den ut i et rektangel, Det å kunne putte en pyramide inni rektanglet eller kvadratet, eller kuben. Og se liksom hva er greia med tanke på areal og omkrets og sånne ting. (…) Jeg tror det er klokt, ja. (ler).

5.5.1.2 Matematisk læringsutbytte ved undervisningen

I denne delen vil jeg se på det lærerne forteller når de blir spurt om hva de anser som det matematiske læringsutbyttet ved arbeidet med programmering, både generelt og i tilknytning til den observerte undervisningen. Først vil jeg legge frem hva lærerne anså at det matematiske læringsutbyttet ved den observerte undervisningstimen, deretter hva de anså som det matematiske læringsutbyttet ved bruken av programmering generelt.

L1 nevner «*omgjøring med tid*» og navn på geometriske figurer som det matematiske læringsutbyttet ved undervisningen. L2 uttrykker at elevene arbeidet med desimaltall og navn på plassverdier, prosent samt innblikk i millisekund-begrepet. L3 innleder med å si «*Jeg vil ikke kanskje påstå at det her*..», før hen stopper og trekker frem koordinatsystemer som er det eneste «*som er sånn kjempe matematikk*». I tillegg til at hen trekker frem «algoritmer» og at disse alltid kommer med. L4 uttrykker at hen tror elevene sitter igjen med innsikt i prosentregning, og en kobling mellom dette og «deling og ganging». Hen forteller at koden elevene tok i bruk var stilt opp «veldig matematisk».

5.5.1.3 Matematisk læringsutbytte ved programmering

Tre av lærerne, L2, L3 og L4, trekker frem de de anser som det matematiske læringsutbyttet ved arbeidet med programmering generelt. L2 knytter læringsutbyttet til forskning, og «*Det å bruke teorier og finne ut av ting. Få ting til å skje*». L3 forteller at utbyttet i stor grad handler om «*å følge instruksjoner og følge og skjønne systemet*». I tillegg trekker hen frem feilsøking i form av «å gå tilbake og lete», samt utholdenhet i arbeidet med oppgaver.

L4 forteller at hen kan se for seg *«at det kan være mye god læring i det»* da det finnes mange muligheter. Likevel uttrykte L4 at hen ikke hadde så mye erfaring og derfor ikke kjente til alle

mulighetene. L4 nevner likevel multiplikasjon, «*strategier*» og «*automatisering av regnearter*».

> *L4: Jeg har ikke så mye erfaring med det, så jeg kan ikke si sånn alt for mye, men jeg ser jo for meg at du kan bruke programmering for å lære en del. Du kan jo lage ulike gangespill, ikke sant, det er jo så mange muligheter. Men jeg kjenner ikke til alle. Jeg skal ikke utdype det alt for mye (ler). Men jeg ser for meg at det kan være mye god læring i det da. Og at du kan bruke det fornuftig til å lære deg ulike strategier eller automatisere regnearter, ja.*

L2 uttrykker som eneste en skepsis i forbindelse med læringsutbyttet ved bruk av programmering. Hen uttrykker en usikkerhet rundt det faktiske matematiske læringsutbyttet kontra tidsbruken.

> *L2: Har ikke brukt det så veldig mye(…). Men ellers så tror jeg nok eget nivå må opp en del for å se det med utbyttet. Men at det er lystbetont og at det er mye du får igjen for det. Ja. Absolutt. Men i forhold til mye annet som ligger i matte, så går det mye i hvor mye du skal bruke koding i forhold til hvor mye av det du skal lære. Der tror jeg kanskje det er for lite til å virkelig sette dette godt sammen.*

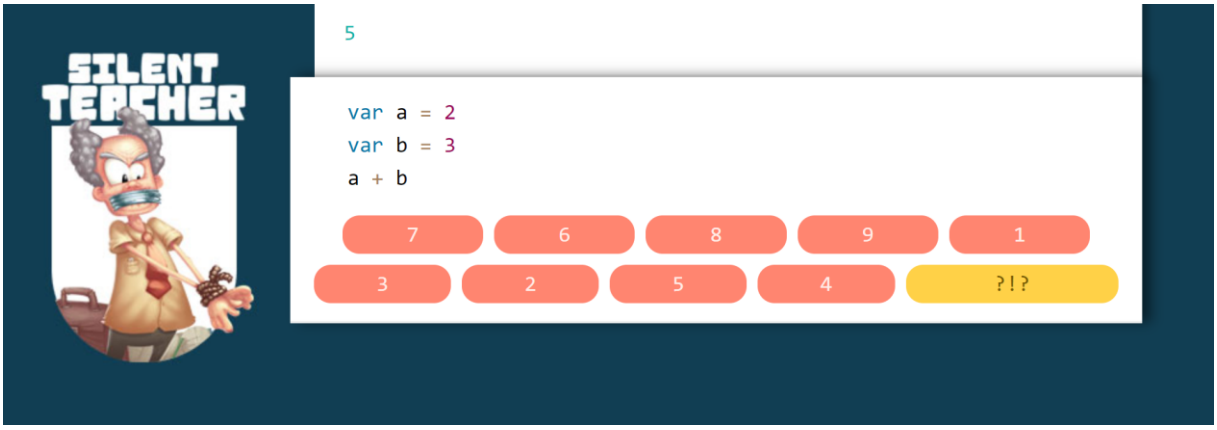
5.5.2 Matematiske konsepter uttrykt i intervju

5.5.2.1 Undervisningsopplegg

Lærerne blir spurt om de har noen undervisningsopplegg som de opplever har fungert spesielt godt. L1, L3, L4 og L5 forteller da om undervisningsopplegg som kan knyttes til matematiske konsepter som algebra og variabler, multiplikasjon og geometri. I dette delkapittelet vil jeg trekke frem hvilke aktiviteter lærerne trakk frem.

L1 trakk frem en oppgave som la vekt på en kombinasjon av koder brukt i programmering og algebraiske konsepter. Oppgaven het «Silent Teacher» (Toxicode, 2017). Utgangspunktet er at elevene blir involvert i oppgaver der læreren skal være taus. Nye oppgaver dukker opp etter hvert som elevene svarer og blir stadig vanskeligere. L1 uttrykker at opplegget åpner for en sammenheng mellom å lese koden og matematikk. Hen utdyper videre at elevene blir utfordret i å finne ut verdien av ulike variabler, og at lærernes taushet bidrar til at elevene reflekterer over hvorfor gitt svar er riktig eller feil. Videre skaper L1 en kobling mellom variabler og funksjoner, og hvordan disse brukes i kodene elevene får opp på tavlen. Ifølge L1 vil aktiviteten også vise at en variabel kan endre verdi, avhengig av oppgave. På denne måten

ser aktiviteten ut til å knytte sammen koding til bruken av matematiske symboler, variabler og funksjoner.



Figur 5.1 Viser et skjermbilde av en av oppgavene i spillet Silent Teacher, hentet fra: Toxicode (2017)

L3 forteller om bruk av spill, laget hinderløyper og programmert bitboter til å kjøre gjennom disse, samt bruken av kart for å arbeide i et koordinatsystem og bevege seg til de ulike rutene. L3 forteller at hen tidligere, med eldre elever har «kodet solsystemet» som et større prosjekt. Dette prosjektet knyttet L3 både til matematikk og Naturfag.

L4 trekker frem bruken av Bit:boter der disse programmers til å kjøre rundt, og til å registrere data som lyd og lys ved hjelp av Micro:biter. L5 kunne fortelle om undervisningsopplegg der elevene programmerte Micro:bit til å spille stein-saks-papir i en kombinasjon med en aktivitet som stafett. Videre trekker hen frem «gangespill i Micro:bit» og arbeidet med Geometri i Scratch og Thinkercad.

5.5.2.2 Geometri

L3 og L5 trekker flere ganger i intervjuet frem arbeidet med geometriske figurer i arbeidet med programmering. Begge trekker dette frem som et matematisk konsept de har brukt i undervisningen. L3 knytter det opp mot tegning av figurene, vinkler og retning. L5 knytter arbeidet med bruken av programmering til det å skape modeller, fysisk og virtuelt av geometriske figurer. Det å ha en modell av dette enkelt tilgjengelig kan ifølge L5 bidra med verdifull læring. Samtidig trekker L5 frem muligheten for å bruke programmeringens visualiserende funksjon til å utforske areal og omkrets. L5 trekker frem geometriske figurer i en tredimensjonal form, mens L3 ikke trekker frem dimensjonen. Både tegningen og modellen kan fremstå som ulike representasjonsformer av de geometriske figurene.

> *L5: (...)så kan man jo diskutere om det henger helt på programmeringa eller ikke, men vi har da, skolene har da egne 3D-printere, så benytter vi oss av et program som*

heter printercad, (…) Og senest i går så er de innom der og lager litt figurer og der er det fantastisk med geometri for eksempel til matematikkundervisningen. Det å kunne få skrevet ut en liten pyramide eller en kube eller hva det måtte være og ha i pennalet ditt, det tror jeg har en veldig god verdi i forhold til læringen.

L5: (…) og det med sirkulering, visualisering, det å kunne lage noe visuelt da. Å se, å sitte, og dra rektangelet eller kvadratet, eller kuben, Og se liksom hva er greia med tanke på areal og omkrets (…)

5.5.2.3 Måling

Underveis i intervjuet med L5 viste hen frem elevarbeid. Et av disse eksemplene var et elevarbeid gjennomført av en elev som ble regnet som en faglig lavt presterende elev. På skjermen kunne L5 vise en detaljert modell av et soverom laget i Thinkcad (Autodesk, 2022). I arbeidet med dette hadde eleven arbeidet med både målestokk og brukt måleenheter i en reel kontekst.

Ved spørsmål om læreren trodde eleven selv var bevisst på det matematikkfaglige innholdet i dette arbeidet uttrykker L5 hvordan elevene til tider bør lures inn i matematikken, og at de gjennom leken i programmeringsverktøyet arbeider med matematiske konsepter, uten at de nødvendigvis er klar over det selv. Samtidig uttrykker L5 at elevene i stor grad gi uttrykk for å se de store linjene, der de kan identifisere eller kjenne igjen fagene i arbeidet de gjør.

L5: akkurat på sånn måling og sånn, vil jeg nok si det, også er det nok mye her som, det er jo generelt. Noen av dem ser jo det som lek og moro, og forstår ikke matematikken bak. Noen ganger er det greit å bli lurt inn i matematikken. (…) Så de er på en måte bevisst om ikke målene er klare så de store linjene i fagene da.

5.5.2.4 Begreper

I forbindelse med bruken av begreper knyttet til arbeidet med programmering, kan det se ut til at L5 legger særlig vekt på bruken og forklaringer av begreper som er spesifikke for programmering som løkke, vilkår, variabel og funksjon.

L5: Så er det begrepslista, den må jeg jo ta opp hver gang og gjenta så ofte jeg kan i de timene det på en måte er(…).

I: (…) Bruker de (elevene) da begrepene løkke, vilkår, variabel, funksjon?

L5: Ja. det er nok å være i en algoritmisk tankegang, algoritme. Det har jeg kjørt ganske hardt på. Det er som å koke suppe eller nudler eller noe sånn, (…) Variabler begynner vi litt over i de spill-greiene med tid og poeng også videre, da innser de jo hva variabler er. Funksjoner er nok kanskje den vi har på en måte vært minst borti så langt(…).

I: Det er nok kanskje også den som det er vanskeligst å få grep om, vil jeg tro. Eller?

L5: Ja. Jeg liker jo å tro at det er et miniprogram inni et program, er jo egentlig en funksjon. At du lager en sånn liten. Et lite program inni der som gjør det enkelt for deg å bruke det flere ganger(…).

Det kan her virke som at L5 til stadighet tar frem begrepene som er relevante for arbeidet med programmering, altså begreper som er spesifikke for programmeringen. Samtidig som at hen knytter dette opp mot algoritmisk tenking.

L1 uttrykker i motsetning til L5 at hen generelt bruker begrepene lite og legger mindre vekt på dette. Hen uttrykker at det viktigste ikke nødvendigvis er at elevene kan begrepene i starten, men at hen tar i bruk noen av begrepene etter hvert som elevene blir eldre. Da vil de også i større grad få behov for å bygge ting selv, og sette ord på hva de gjør.

5.5.3 Matematiske konsepter i undervisning

I dette delkapittelet tar jeg for meg hvilke matematiske konsepter som ble observert i undervisningen. Hos S1, S2, S3 og S4 kom det matematiske konsepter til uttrykk i undervisningen. Blant disse konsepter finner vi Måling, Koordinatsystemer, Prosentregning og Begreper til uttrykk.

1.1.3.1 Måling

L3 legger opp til at elevene skal se på antall skritt hver av karakterene tar pr. tastetrykk. Der er det i utgangspunktet en én til fire korrespondanse. Der hver tast tilsvarer fire skritt.

L3: Hvor mange skritt går den av gangen?

E: fire!

L3: Hva skjer hvis man endrer den?

(elevene får rope ut forslag og læreren endrer tallet til 10. Figuren beveger seg med større avstand for hvert tastetrykk. Elevene foreslår tusen og titusen).

L3: Hva hvis jeg tar 100? (Læreren endrer til 100)

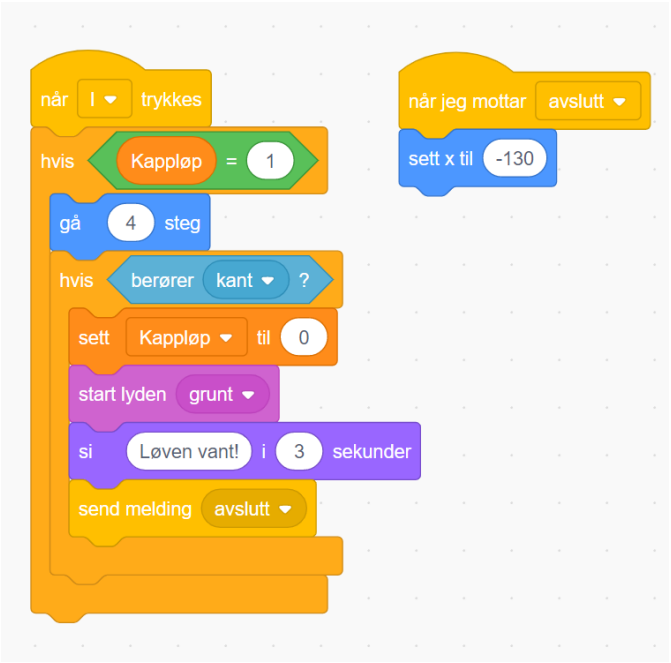
L3: Da gikk den mye fortere!

I dette tilfellet ser det ut til at antall skritt påvirker hastigheten, da antall skritt per tastetrykk øker vil antall tastetrykk minke. Den viser også at jo større avstand figurene går per skritt desto raskere vil figuren bevege seg.

I tillegg til hastighet, satt elevene hos S3 å arbeidet med størrelsen på karakterene i spillet. De testet ulike tallverdier, og så hvordan dette påvirket størrelsen på figuren. L3 viser hvor de kan gjøre endringer, men ikke hvilket tall de skal bruke. Elevene tester først et lavt tall og papegøyen forsvinner fra skjermen. Deretter tester de et betydelig større tall og papegøyen dekker hele skjermen. Elevene leker seg med tallene og størrelsesforholdene.

1.1.3.2 koordinatsystem

I undervisningssekvensen hos S3 kunne jeg observere bruken av koordinatsystemer. Dette var også et punkt som ble tatt opp gjennom felles klassesamtale. Først da hen eksplisitt trekker frem x-verdien og at denne angir karakterens plass. L3 uttrykker «*Det som er likt er X-en. Den sier noe om hvor de står*». Hen går på dette tidspunktet ikke nærmere inn på koordinatsystemene. Deretter da hen helt til slutt endrer x-verdien og viser at elevene kan flytte karakteren ved å gjøre dette. Elevene deltok i liten grad i samtalen angående x-verdiene.



Figur 5.2 Rekonstruksjon av koden for karakteren løven hos S3. (Scratch Foundation, 2019)

Figur 5.3 Rekonstruksjon av koden for karakteren papegøyen hos S3 (Scratch Foundation, 2019)

1.1.3.3 Matematikk med utgangspunkt i begreper

S1, S2, S3 og S4 arbeidet med begreper. Hos S1 og S2 observeres det klassesamtaler der det legges fokus på benevnelsen «ms» som står for millisekunder. Elevene må forklare hva millisekunder betyr. I begge observasjonene utgjør denne delen av samtalen kun en mindre del av den totale undervisningstiden. Likevel blir millisekunder sentralt i oppgavene elevene skal gjøre. I tillegg uttrykte begge at dette var begreper de hadde arbeidet med tidligere, samt at de hadde arbeidet med omgjøring mellom måleenheter tidligere. Det kan dermed virke som at undervisningen bygger på det elevene kan fra før av, men at det nå brukes i en annen kontekst. S2 og S4 arbeidet med prosent begrepet.

Hos S3 ble det observert flere situasjoner der matematiske begreper ble brukt og korrigert. En av hendelsene viser en gruppe som ønsker å endre størrelsen på karakteren. Elevene diskuterer hvordan de skal endre størrelsen og en av elevene forteller at «*Da må vi forøke den*». Læreren korrigerer eleven ved å si «*Forminske*» før hen går videre. Her kan det virke som at elevene forveksler begrepene «øke» og «minke». Et annet eksempel er forståelsen av «*å telle ned*». Elevene lurer på hvorfor oppskriften spør om dette. Læreren trykker på knappen og spør elevene «*Teller den ned?*» samtidig som hen teller ned mens sifrene kommer opp på skjermen «*Tre, to, en, løp!*». Elevene responderer med et «nei». Dette til tross for at tallene på skjermen teller ned. Læreren gjentar spørsmålet og tester koden igjen. Denne gangen med mer vekt på nedtellingen. Elevene svarer nå «*ja*». I begge tilfellene kan det se ut til at elevene har vansker med å forstå innholdet av begrepene som brukes. Lærerens valg av veiledning er basert på å gjenta ordet, samt fremheve handlingen.

I tillegg til de overstående begrepene er det et annet begrep som går igjen hos S3 og S4; «variabel». Begrepet brukes aktivt i begge klassene, og begge lærerne bruker ordet når de hjelper elevene. Ordet ble ikke forklart på noe tidspunkt. I intervjuene med elevene ser det derimot ikke ut til at elevene var kjent med dem. Da de på spørsmål «*Vet du hva en variabel er?*» svarer «*Nei*». Det er i tillegg kun en av elevene, E4.2, som gir uttrykk for at hen har hørt ordet tidligere, via læreren, men hen kan ikke fortelle hva dette er.

5.6 Matematiske konsepter uttrykt av elevene

Ved intervju av E1.1, E1.2, E1.3 og E1.4 gir alle elevene uttrykk for at en kan bruke programmering i arbeidet med matematikk. Disse elevene knytter det til tall og tallregning generelt. E1.1 uttrykte at det var stort behov for bruken av tall, og å kunne arbeide med disse, samt halvering av tallverdiene. E1.2 knytter det spesifikt opp mot divisjon eller multiplikasjon. E1.3 uttrykte at hen hadde arbeidet med tid i form av sekunder og at de måtte telle og addere. E1.4 sa hen hadde arbeidet med «matte», men kunne ikke spesifiser dette ytterligere.

> *E1.1: Ja, Absolutt. For det er mye sånn hvis den er det, så må halvparten være det. Og da må vi få det tallet til å stemme med det tallet.*

> *E1.2: Fordi at hvis man kommer til en matteoppgave som skal løses. Må man skrive løsningen da, og at man klarte å løse det. Da er det fint å liksom. Så for eksempel hvis det var tre bokser og med 20 i hver. Også var det noe som skal skrives opp under hver boks 3 minus 7 eller noe sånt.*

Tabell 5.7 viser hvilke konsepter elevene uttrykker at de har arbeidet med, samt hvilke konsepter de knytter opp mot programmering generelt

Elever fra S1	**E1.1**	**E1.2**	**E1.3**	**E1.4**
Konsepter uttrykt	Halvering og tall	Tid Divisjon og multiplikasjon	Tid; Sekunder. Telle og addere	Matte – ikke spesifisert
Elever fra S3	**E3.1**	**E3.2**	**E3.3**	
Konsepter uttrykt	Lese teksten. Tekstoppgaver. Ikke gi opp.	Følge instruksjoner og lese nøye.	Lese nøye.	
Elever fra S4	**E4.1**	**E4.2**	**E4.3**	**E.4.4**
Konsepter uttrykt	Reproduksjon av lærerens kode	Reproduksjon av lærerens kode	Prosent	Reproduksjon av lærerens kode

Til tross for at alle fire elevene fra S1 ser ut til å mene at en kan bruke programmering til å arbeide med matematikk, kan hverken E1.3 og E1.4 begrunne hvorfor de mener dette. E1.3 svarer «vet ikke» nå hen blir spurt om hvordan man kan bruke programmering i matematikk.

Hen har før dette svart «Ja» til at programmering kan arbeides med i matematikk. E1.4 knytter det til «tall og sånn», men spesifiserer ikke ytterligere når en får bruk for dette.

> *E1.4: ja, jeg tror det.*
>
> *I: Kan du prøve å forklare hvorfor?*
>
> *E1.4: Det er fordi man må bruke litt matte i det fordi det er jo tall og sånn.*

I forbindelse med den observerte undervisningssekvensen, knytter E1.2 og E.1.3 arbeidet direkte til «tid».

> *E1.3: For eksempel hvor mange sekunder. Da måtte du telle og plusse sammen (…) vi pleier å bruke ganske mye matte når vi driver med programmering.*

E3.1, E3.2 og E3.3 knytter i liten grad matematikk til arbeidet de har gjort. Samtlige av disse elevene svarer «nei» på spørsmål om de har brukt matematikk i oppgaven. E3.2 forteller at de kun måtte gjøre det som stod i oppskriften. E3.1 kan på spørsmål om man kan arbeide med programmering i matematikk knytte lesing av oppgaven opp mot tekstoppgaver i matematikk. Enkelte av elevene kan også fortelle hvilke arbeidsmetoder de tar i bruk. E3.1 kan blant annet fortelle at man ikke skal gi opp, og fortsette selv om man ikke får det til og lese oppgaven nøye. E3.2 og E3.3 forteller at man må lese nøye. I tillegg forteller E3.2 videre at en skal følge instruksjonene i tillegg til at man se dobbeltsjekke oppskriften dersom man er usikker og for å sjekke at det blir riktig. Disse arbeidsmetodene kan ha hentydning til algoritmisk tenking.

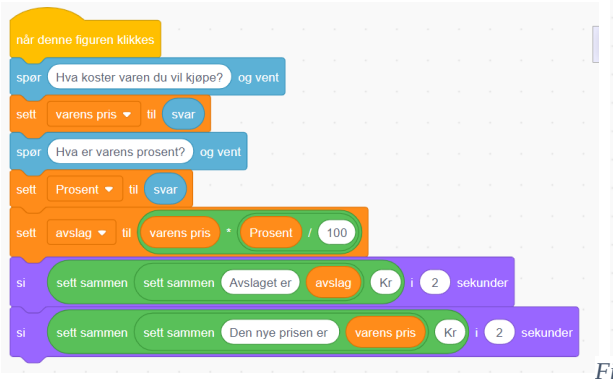
Under intervjuene med E4.1, E4.2, E4.3 og E4.4 kunne alle fire elevene fortelle hva de har arbeidet med, og hvordan en finner avslaget på en vare. Ved spørsmål om de fikk behov for å bruke «matte» under arbeidet med programmering var det én av elevene, E4.3, som uttrykte at de hadde arbeidet med prosent.

E4.3: Ja. Det var jo på en måte prosentregning, også måtte man jo skrive inn det der med ganging og deling. Finne det også måtte man skrive inn de riktige tallene og sånn.

Figur5.4 Rekonstruksjon av koden brukt hos S4(Scratch Foundation, 2019)

Til tross for at den siste delen av koden ble flyttet til side, hadde likevel E4.3 valgt å prøve seg på den siste koden, som beregnet sluttprisen. Eleven kunne forklare hvordan hen brukte informasjonen tidligere i koden for å finne ut av hvordan den siste koden fungerte.

E4.3: Å liksom følge instruksjoner riktig og liksom hvis man skal prøve mer sånn som jeg gjorde så liksom se på hva den forrige regningen gjorde også se på en måte om man kan ish gjøre det samme. Da blir det litt lettere.



Figur 5.5 Rekonstruksjon av koden brukt i undervisning S4 før den siste kodeblokken ble tatt bort. (Scratch Foundation, 2019)

Figur 5.6 Rekonstruksjon av koden som beregnet sluttprisen. (Scratch Foundation, 2019)

De tre andre elevene, E4.1, E4.2 og E4.3, uttrykte derimot at de ikke hadde fått bruk for noe matematikk i arbeidet, og knyttet programmeringsarbeidet opp mot reproduksjon av lærerens kode.

E4.1: (stille) egentlig ikke. Eller. Nå så vi på læreren sin da, men hvis vi ikke hadde sett på læreren sin så måtte vi sikkert ha brukt litt matte. Men siden vi så det, så måtte jeg ikke bruke det.

6 Drøfting

I denne delen vil jeg se på analysen i lys av teorien lagt frem i Kap. 3 Teoretisk bakgrunn. Overskriftene gjenspeiler funn fra analysen. Denne drøftingen består av fem delkapitler, som igjen består av flere underkapitler. Først trekker jeg frem hvordan undervisningens tilknytning til matematikfaglige konsepter kommer til uttrykk i det lærerne uttrykker som formålet og hensikten ved undervisningen. Deretter vil jeg se på hvilke pedagogiske strategier som har blitt tatt i bruk, og hvordan disse kan bidra til å arbeide med matematisk kompetanse. Den tredje delen handler om de matematiske konseptene som kommer til uttrykk, og jeg vil se på hva lærerne forteller og hva de faktisk gjør i undervisningen. Etter dette vil jeg se på overføringen av kunnskap, og til hvilken grad den matematikkfaglige kunnskapen har blitt tatt i bruk i arbeidet med programmering. Den siste delen tar for seg lærerens rolle.

6.1 Formålet for undervisningen

I dette delkapittelet vil jeg se på hvordan formålet gjengitt i Kap. 5.1 Formålet med undervisningen kan knyttes til de formelle begrunnelsene og bakgrunnene for implementering av programmering i matematikk. Jeg vil i denne delen blant annet se på hvordan kompetansemålene, som lærerne knytter til undervisningen, skaper tilhørighet i faget matematikk. I tillegg vil jeg se på hvordan lærernes formål om samarbeidslæring kan knyttes til kompetanseområder trukket frem av Ludvigsen-utvalget(NOU 2015:8). Til slutt vil jeg se på hvordan det underliggende formålet om motivasjon kan fungere som et matematikkfaglig formål.

6.1.1 Generelle og matematikkspesifikke kompetanseområder

I korte trekk kan lærernes formål oppsummeres som henholdsvis; Samarbeid og samarbeidslæring, Algoritmisk tenking, dybdelæring og motivasjon (Se Kap. 5.1). Formålene er å finne på ulike nivåer, der èn ting er hva elevene får vite. Dette er gjerne det elevene klarer å gjenfortelle at de har arbeidet med. En annen ting er det bakenforliggende formålet som er mer komplekst sammensatt av både ferdigheter og kunnskaper. Det er i disse bakenforliggende formålene at vi hovedsakelig kan finne igjen ferdigheter som kan knyttes opp mot utforsking, problemløsing og algoritmisk tenking. Denne delen kan det se ut til at elevene selv uttrykker lite kjennskap til.

Kilhamn et al. (2021) viser til fire argumenter lærere i deres forskning bruker for å ta i bruk programmering i forbindelse med matematikkfaget (Se Kap. 3.3). Disse er utvikling av algoritmisk tenking, økt engasjement, å lære matematikk og at programmering er et nyttig

verktøy (Kilhamn et al., 2021). De tre matematiske komponentene nevnt her samsvarer med de matematiske komponentene vi finner igjen i analysen; Algoritmisk tenking, matematiske konsepter og programmering som et verktøy. Videre viser Ludvigsen-utvalget til fire kompetanseområder som kan utvikles gjennom arbeidet med programmering (NOU 2015:8; Sevik & m.fl., 2016) . Kompetanseområdene er henholdsvis 1) «fagspesifikk kompetanse», 2) «kompetanse i å lære», 3) «kompetanse i å kommunisere, samhandle og delta» og 4) «kompetanse i å utforske og skape» (NOU 2015:8, s. 22). Disse kompetanseområdene strekker seg fra det fagspesifikke til det sosiale og spenner på denne måten vidt. I likhet med de overstående argumentene til Kilhamn et al. (2021) og kompetanseområdene til (NOU 2015:8), kan vi se at lærerne viser til formål som spenner vidt. Fra det matematikkspesifikke, som prosentregning, til de metakognitive aspektene som kommer til uttrykk gjennom arbeid mot algoritmisk tenking, samt de sosiale aspektene som samarbeid og motivasjon.

De fagspesifikke områdene som oppgis av lærerne, er hovedsakelig knyttet til algoritmisk tenkning. L5 trakk det spesifikt fram, mens L3 og L4 trakk det fram mer bakenforliggende. L4 trakk frem prosentregning, som de hadde den aktuelle timen. Prosentregning åpner for arbeidet med «algoritmen» for regning med prosent, men også symboler og tallverdier.

Algoritmisk tenking åpner for en fagspesifikk, men også en metakognitiv tilnærming. Den fagspesifikke tilnærmingen, kan som nevnt, knyttes til blant annet algoritmebegrepet, og kompetanse i å løse problemer. Som omtalt i Kap. 3.5.1.1 Algoritmisk tenking, handler algoritmisk tenking om å benytte seg av systematiske strategier for å håndtere problemer (Bueie, 2019; Haraldsrud et al., 2020; Nygård, 2018; Weintrop et al., 2015; Wing, 2006). Gjennom denne prosessen vil elevene i arbeidet etter hvert også bli bevisste på hvilke strategier de tar i bruk og hvordan de skal starte tankeprosessen i arbeidet. Dette samsvarer med det Sevik og m.fl. (2016) uttrykker om sammenhengen mellom algoritmisk tenking og det metakognitive kompetanseområdet, der den algoritmiske tenkingen bidrar som et verktøy for løsning av ulike oppgaver. I Kap. 3.5.1.1 Algoritmisk tenking vises det til ferdigheter som kan knyttes til algoritmisk tenkning. Disse finner vi også igjen når L3 og L5 trekker frem ferdigheter som utholdenhet, å følge instruksjoner, samt begrepet «algoritme» som igjen knyttes til algoritmisk tenking (Bueie, 2019; Haraldsrud et al., 2020; Weintrop et al., 2015; Wing, 2006). «Algoritme»-begrepets likhetstrekk til den algoritmiske tenkingen og hvordan vi ønsker å arbeide i matematikken kan finnes i det å dele opp problemer, og arbeide med disse trinn for trinn. Dermed blir algoritmen og den algoritmiske tenkingen brukt i lys av hverandre.

De fagspesifikke formålene som lærerne oppgir, samsvarer i stor grad med kompetanseområdene 1) Fagspesifikk kompetanse og 2) kompetanse i å lære (NOU 2015:8). De matematikkfaglige formålene gjøres ikke nødvendigvis eksplisitte for elevene, men ligger i bakgrunnen ved at lærerne vet at dette er noe som det arbeides med. Det matematikkfaglige innholdet ser likevel ut til å komme i skyggen av de andre formålene, og kan virke å komme noe tilfeldig frem i undervisningen og i samtalen under intervjuet. Den matematikkfaglige koblingen skapes først og fremst hos L4 og L5, med hovedformål som algoritmisk tenking og prosentregning, men den viktigste koblingen skjer gjennom formålet uttrykt ved valg av kompetansemål.

De kompetansemålene som lærerne knytter til undervisningen, er det som i stor grad skaper tilknytningen til matematikkfaget hos tre av lærerne. L1, L4 og L5 viser til kompetansemål som knyttes til programmering i matematikkfaget. Valget av kompetansemål, legger videre føringer for fagtilhørighet, og er på den måten fagbestemmende (Kunnskapsdepartementet, 2019). Disse kompetansemålene er utformet slik at de skaper en tilknytning til matematikkfaglige konsepter. Løkker, nevnt av L1, er i utgangspunktet et programmeringsspesifikt konsept, likevel har det blitt lagt inn under faget matematikk. På denne måten skaper læreplanverket en føring for hvilken fagtilhørighet dette konseptet skal ha. Dermed kan også konseptet løkker sies å ha en matematikkfaglig tilhørighet. Det som formidles til elevene ser derimot ut til å ha et konstruktivistisk preg, der de skal lage eller gjøre noe (Feurzeig et al., 2011), og kan knyttes opp mot kompetanseområde 4) kompetanse i å utforske og skape.

I likhet med kompetanseområdene, ser lærerne ut til å ilegge arbeidet med programmering en samfunnsnyttig rolle ved å trekke frem at ferdighetene elevene bruker i dette arbeidet er ferdigheter de vil får bruk for i flere sammenhenger, uavhengig av fag. Kunnskaper og ferdigheter som en får bruk for i flere sammenhenger, passer godt inn i den tanken L5 uttrykte om mindre skarpe skiller mellom fagene. Ideen om en mer tverrfaglig arbeidsinndeling, åpner også for å arbeide med ferdigheter som kreves fra et samfunnsperspektiv. Her kommer samarbeid og samarbeidslæring inn som hovedformål (L2, L3 og L5). I likhet med lærernes andre formål kan denne også knyttes til et kompetanseområde, henholdsvis 3) kompetanse i å kommunisere, samhandle og delta, og kan begrunnes med at samarbeid krever kommunikasjon.

6.1.2 Motivasjon som veien inn

Tre av lærerne viser flere ganger i løpet av intervjuet til at arbeidet med programmering skal være til motivasjon for elevene. Motivasjonen knyttet til programmeringsarbeidet kan sørge for at elevene kan arbeide med matematiske komponenter i en ny setting, som igjen kan gi elevene flere «perspektiver» (L1, L4 og L5) på faget. Dermed kan det på flere måter se ut til henvisningen til at arbeidet med programmering skal være motiverende for elevene, blir en vei inn til arbeidet med matematikken (Forsstøm & Kaufmann, 2018; Kilhamn et al., 2021). Dette svarer til det Forsstøm og Kaufmann (2018) og Kilhamn et al. (2021) uttrykker om at arbeidet med programmeringsaktiviteten i seg selv ikke nødvendigvis behøver å tilsvare et matematisk læringsutbytte, men engasjement for arbeidet sørger for at elevene engasjerer seg i oppgavene, og på den måten arbeider med matematikken. L5 forteller om at elevene kan lures inn i matematikken. Eksemplet L5 kom med viste hvordan elever som vanligvis ble regnet som lavt presterende i matematikk ofte kunne klare ganske komplekse oppgaver. Eksemplet på dette er eleven som tegnet soverommet sitt i riktig målestokk vitner om kanskje kompleks bruk av matematiske konsepter i arbeidet med programmering.

Ved at elevene jobber med oppgaver som engasjerer, kan de arbeide med matematiske konsepter. På en annen side kan det se ut til at dette igjen fører til at det matematikkfaglige ikke blir tydeliggjort for elevene, og derfor forblir skjult. Dette kan igjen støttes opp av intervjuene med elevene, da et fåtall av elevene opplevde at de hadde arbeidet med matematikk. Dette viser samtidig til at bevisstgjøring er en nøkkelfaktor. Denne nøkkelfaktoren kan knyttes til det Benton et al. (2017) og Salomon og Perkins (1989) påpeker at innholdet må gjøres eksplisitt for elevene. Lærerne forteller selv også at klassesamtalen bør brukes til å trekke frem det matematikkfaglige innholdet. Spørsmålet er om de gjør dette i stor nok grad. Dette vil jeg komme tilbake til senere i kapittelet.

Det store spriket mellom hva som konkret blir formidlet til elevene og hva lærerne uttrykker i intervjuet kan, basert på et motivasjonsperspektiv, fremstå som et strategisk valg. Dette da de fleste elever virket å ha positive assossiasjoner knyttet til programmeringsarbeidet. Det er ikke nødvendigvis tilfellet for matematikk (L1 og L5). Lærernes valg av å legge frem hva elevene skal lage eller gjøre, og kalle faget «koding» fremfor «matematikk», kan bidra til at elevene som har mindre positive asosiasjoner til matematikkfaget, ikke forbinder det med det de selv opplever som negativt. Elevene kan da heller oppleve det som en positiv aktivitet. De kan dermed arbeide med matematiske komponenter gjennom et motivasjonspreget medium uten de negative assosiasjonene. Dette samsvarer med hvordan L1 og L4 forteller at de tar i bruk

programmering som et visuelt medium på lik linje som andre elementer i matematikken. På en annen side, kan det bidra til at elevene ikke ser sammenhegnen mellom aktiviteten de gjør og det matematikkfaglige innholdet læreren trekker frem. Det matematikkfaglige innholdet blir gjort usynlig for elevene. Ifølge Salomon og Perkins (1989) har elevene har et behov for at den matematikkfaglige kunnskapen blir gjort eksplisitt for at det skal kunne skje en overføring av kunnskap.

6.1.3 Fagspesifikke kompetanser og motivasjon

I dette delkapittelet ser vi hvordan lærernes formål samsvarer med de kompetanseområdene som ludvigsen-utvalget trekker frem. Det kommer også frem at det finnes fagspesifikke kompetanseområder i lærernes formål ved undervisningen, som kan knyttes til matematikkfaget. Likevel kommer disse i skyggen av andre kompetanseområder. I tillegg kan det, til tross for at det fremstår som et underliggende formål, se ut til at motivasjonsaspektet ved programmeringsarbeidet fungerer som en viktig brikke i det å la elevene arbeide med matematiske konsepter. Ved at elevene finner arbeidet engasjerende, kan det legges opp til at elevene tar i bruk matematiske konspeter i arbeidet, uten at elevene opplever det som «typisk matematikk». På denne måten kan motivasjonen fungere som en vei inn til det matematiske innholdet. Ulempen ved dette er at matematikkfaglige innholdet blir usynlig for elevene.

6.2 Pedagogiske strategier som brobygger

I Kap. 5.2 Metoder og strategier vises det til at lærerne legger til rette for undervisningsmetoder som samarbeid og klassesamtaler. Disse fører i tillegg til at det legges til rette for at elevene kan ta i bruk ulike strategier for å løse oppgavene. Strategiene elevene bruker, ser ut til å ha fellestrekk med rammeverket 5E-er, som er omtalt i Kap. 3.6.1.1 Bentons 5E-er, og består av pedagogiske strategier for å arbeide med programmering. Parallellene mellom disse, vil jeg gå nærmere inn på i avsnittene under.

Benton et al. (2016) og Benton et al. (2017) påpeker at særlig den siste strategien «Bridge» er strategien som skal bidra til å skape en eksplisitt kobling til det matematikkfaglige innholdet i arbeidet, og fungerer derfor som brobygger mellom programmeringsarbeidet og det matematikkfaglige innholdet. I tillegg til dette kan det se ut til at samtlige av de andre strategiene også kan bidra til at elevene arbeider med matematiske kompetanser. Strategiene kan fremstå som kompetanser elevene skal lære seg i arbeidet med programmering. Handlinger som utforsking, forutse, forklare og utveksle ideer, gjør at strategiene også fremstår som kompetanser. De kan være handlinger elevene bør lære seg å bruke og som

lærerne skal legge til rette for at elevene kan ta i bruk. Fire av de fem E-ene fremstår derfor først og fremst som programmeringsspesifikke kompetanser, dette fordi det legges opp til at de pedagogiske strategiene skal brukes i en programmeringskontekts.

I Kap. 3 Teoretisk bakgrunn, får en inntrykk av at programmering har sin naturlige tilknytning til matematikkfaget, både historisk, men også i matematikkens rolle i programmeringen (Bueie, 2019; Kaufmann et al., 2018; Papert, 1980). Jeg vil derfor se på den mulige koblingen mellom de pedagogiske strategiene til Benton et al. (2016) og Benton et al. (2017), og Niss et al. (2002) sine matematiske kompetanseområder. Ved å tilrettelegge for bruk av undervisningsmetodene og strategiene nevnt i Kap. 5.2 Metoder og strategier, ser det ut til at elevene arbeider innenfor de matematiske kompetanseområdene. På denne måten ser det ut til at de pedagogiske strategiene fungerer som brobyggere mellom programmeringsarbeidet og matematikken. Der arbeid med matematiske kompetanser ser ut til å være konsekvensen av strategiene som blir tatt i bruk. Dette kan igjen åpne for at elevene kan utvikle seg innenfor disse kompetanseområdene. I avsnittene under vil jeg legge frem hvordan de matematiske kompetansene kommer til uttrykk, og hvordan de pedagogiske strategiene som har blitt tatt i bruk kan fungere som en brobygger inn mot faget matematikk

6.2.1 BridgE

Koblingen til matematikken, blir først og fremst gjort gjennom den femte pedagogiske strategien *Bridge* (Benton et al., 2016). Strategien handler om at det eksplisitt, ved lærerens hjelp, skal skapes en sammenheng mellom komponentene matematikk og programmering. *Bridge* fungerer som en brobygger mellom programmeringsarbeidet og de matematiske konseptene. Særlig de spesifikke konseptene. I klassesamtalene blir elevene oppfordret til å forklare matematiske begreper, i tillegg til at lærerne trekker frem eksempler på deler av oppgavene som tar for seg matematiske konsepter. Dermed trekker lærerne eksplisitt frem de matematiske konseptene i arbeidet gjennom klasseromssamtalene. Dette skjer i noe ulikegrad.

Lærerne tar i bruk hugging- og bridging-teknikker (Se Kap. 3.6.2.2.5.1) for å trekke frem likhetstrekkene mellom matematikken og programmeringsarbeidet (Benton et al., 2018; Franklin et al., 2016). Hugging-teknikker er teknikker for å gjøre arbeidet med programmering så likt det tradisjonelle arbeidet med matematikk som mulig. Bridging-teknikker er i større grad teknikker som lar elevene ta i bruk matematiske ideer i en programmeringssetting, og har en mer problemløsende tilnærming. Ved å gjøre arbeidet med koden så lik algoritmen for prosentregning som mulig benytter L4 seg av

sammenlikningsmulighetene. Sammenlikningene kan knyttes til hugging-teknikker, der det i arbeidet trekkes fram bruken av symboler og tallverdier, samt rekkefølgen i den opprinnelige algoritmen. Bridging-teknikkene ser ut til å komme til uttrykk i klassene der undervisningen i større grad er av problemløsende karakter, som for eksempel hos S1 og S2. Der de arbeider med matematiske konsepter som tid gjennom utforskende aktiviteter. På denne måten skapes det en sammenheng gjennom problemløsing, der det tas i bruk generelle ideer fra matematikken som for eksempel feilsøking, og algoritmisk tankegang til arbeidet med programmering (Benton et al., 2017; Benton et al., 2018; Weintrop et al., 2015; Wing, 2006). I begge tilfeller, særlig hos S4, men også hos S2, blir den formelle matematikken trukket frem i klasseromssamtalene og elevene må, ved bruk av eget språk sette ord på formelle symboler og begreper. Det ser dermed ut til at det i begge tilfeller skapes en mulighet for at elevene kan arbeide med symbol- og formalkompetanse (Niss et al., 2002). Uavhengig av om det har blitt tatt i bruk hugging- eller bridging-teknikker.

6.2.2 Explore

Benton et al. (2016) første strategi, *Explore*, kommer fram i strategiene elevene bruker. *Explore* handler om at elevene skal få arbeide med problemløsende aktiviteter som åpner for utforsking av ideer og bruke prøving og feiling som strategi for å finne løsninger. Oppgavene brukt hos S1 og S2 har en åpen utforming, som ser ut til å bidra til utforsking. Elevene må selv finne gode løsninger på oppgavene de møter på. De bruker strategier som prøving og feiling, samt feilsøking for å rette opp i koder som ikke fungerer. Elevene ser da ut til å arbeide strategisk mot en løsning på oppgaven de skal gjøre. Elevene får prøve seg frem og se hvordan kodene reagerer, dermed får elevene videre muligheten til å ta kontroll over sin egen læring (Benton et al., 2016). Å arbeide med en oppgave på denne måten, samsvarer i store trekk med det Resnick og Rosenbaum (2013) definerer som «*tinkering*», samt strategiene som tas i bruk i den heuristiske matematikken (Schoenfeld, 1985). Disse likhetstrekkene, skaper en sammenheng mellom *Explore*, og det vi definerer som matematisk problemløsing. Selv om oppgavene hos S1 og S2 er åpne, er det likevel formulert og avgrenset et problem elevene skal løse. Problemet kan ikke løses ved hjelp av enkle regneoperasjoner, men krever en helhetlig forståelse av problemet de skal løse, som blant annet innebærer en forståelse av tid, rekkefølge, hastighet og retning. Dette samsvarer med strategier som kan knyttes opp mot matematisk problemløsingskompetanse (Niss et al., 2002).

6.2.3 Envisage

Den tredje strategien som kommer til uttrykk blant strategiene som blir brukt, er den Benton et al. (2016) kaller for *Envisage*. Denne strategien handler om å skape muligheter for at elevene må reflektere over valg de tar underveis, samt i denne refleksjonen forsøke å forutse konsekvensene av handlingene de gjør. Denne måten å arbeide på skal bidra til at utforskingen skal bli mer målrettet. Elevene ser i stor grad ut til å prøve å forutse konsekvensen av en gitt handling i koden de bruker. Dette bidrar til at de utvikler en effektiv «prøve og feile»-strategi. Elevene må reflektere over koden og forutse hva deres endringer kan resultere i. I denne refleksjonsprosessen blir det stilt spørsmål, i tillegg til at det oppstår muligheter for å prøve å forutse svaret på egne spørsmål som dukker opp underveis. Elevene vender stadig tilbake til koden for å løse problemet og tester hvordan endringene påvirker helheten. Ved å gjøre dette, får elevene mulighet til å følge med på sin egen tankeprosess (Clements, 1984). Elevene ser dermed ut til å ta i bruk matematisk tankegangskompetanse (Niss et al., 2002). Måten elevene hos S2 diskuterte i forkant av utprøvingen av koden, så ut til å bidra til at elevene ved å tenke fremover på resultatet, klarte å komme frem til riktig løsning, og de så dermed ut til å ligge i forkant av feilene. Det kunne også se ut til at elevene som forsøkte å tenke fremover, og prøvde å forutse konsekvensen av den gitte koden, hadde behov for færre utprøvinger av koden enn de elevene som kun baserte seg på resultatet av koden de brukte. Observasjonene av undervisningen, kan vise at elevene ved å forestille seg neste trekk, raskere klarte å finne en løsning på oppgaven.

6.2.4 Explain og Exchange

Benton et al. (2016) tredje strategi er *Explain*. Strategien handler om at en skal legge til rette for at elevene må begrunne valg, samt sette ord på det de har lært. I samarbeidet mellom elevene og i klasseromssamtalene ledet av læreren, blir elevene oppfordret til å forklare hverandre matematiske begreper og konsepter som rotasjon, prosent og tid, samt at de må forklare hverandre hvordan de kan løse de gitte oppgavene. I kombinasjonen av klasseromssamtaler og samarbeid, kan det se ut til at elevene ved hjelp av lærerne kommuniserer på ulike ferdighetsnivåer. Der klassesamtalene kan bidra med å spesifisere og bruke de rette matematiske begrepene, mens samarbeidet mellom elevene foregår på et mer uformelt nivå. Denne prosessen med kommunikasjon kan videre også knyttes opp mot *Exchange* (Benton et al., 2016). Exchange handler om at det skal legges til rette for at elevene kan utveksle ideer og erfaringer med hverandre. Elevene må forklare og begrunne valg de tar i samarbeidssituasjonene, samt at formidlingen av dette må skje på en måte som gjør at det er

forståelig for andre. På denne måten kan det se ut til at de pedagogiske strategiene *Explain* og *Excahnge* leder elevene til å arbeide med matematisk kommunikasjonskompetanse (Niss et al., 2002). I klasseromssamtalene bidrar elevene ved å sette ord på faglige begreper og forklare hverandre faglige konsepter. Elevene deler ideer og erfaringer med hverandre som leder dem videre i arbeidet de gjør. Gjennom erfaringsutvekslingen, vil elevene måtte ta stilling til de argumentene medeleven kommer med og vurdere disse. Dermed kan det se ut til at elevene også arbeider med resonneringskompetanse (Niss et al., 2002).

6.3 Matematikk uttrykt og brukt

Lærerne uttrykker en stor variasjon og bredde i det matematikkfaglige læringsutbyttet ved programmering, og det er at det enighet om at programmering åpner for mange muligheter for å arbeide med matematikk. Både under observasjonene og underveis i intervjuene blir det trukket frem flere matematiske konsepter (Se tabell 5.6). I dette delkapittelet vil jeg se på hva lærerne sier, og hva som observeres som matematikkfaglige områder som kan knyttes opp mot programmering og arbeidet med dette i undervisningen. Først vil jeg se på hva lærerne bruker programmering til.

6.3.1 Et visuelt medium som åpner for dybdelæring

Feurzeig et al. (2011) omtaler programmering som et «mathematical laboratory», og viser sammen med Clements og Merdith (1992) og Haraldsrud et al. (2020) til programmeringens utforskende og undersøkende muligheter. Programmeringen kan dermed bidra som verktøy for elevenes tenking. Tre av lærerne, L1, L4 og L5, tildeler programmeringen en visualiserende og representativ rolle, og det blir omtalt som et alternativt medium som kan brukes for å arbeide tverrfaglig. Å ta i bruk programmeringsarbeidet på tvers av fag, kan bidra til at en holder dørene inn til alle kompetanseområder åpne. På denne måten settes det ingen begrensinger knyttet til fag. Likevel kan det å knytte arbeidet til spesifikke fag bidra til at det blir tydeligere for elevene hva de arbeider med, og hva de skal lære. I tillegg kan dette også bidra til å gjøre det matematikkfaglige potensialet i oppgaver tydeligere for læreren, da læreren snevrer blikket inn mot det fagspesifikke.

L1, L4 og L5 fremhever at programmering kan bidra til å representere matematikken på flere måter, i tillegg til at det kan virke som at matematikken gjennom bruken av programmering ikles en ny drakt. Der det i stor grad tidligere har vært en samling av tall og symboler fremstår det nå en visuell representasjon. Det visuelle kan bidra til å gjøre matematikken mindre skremmende, slik den for enkelte muligens kan være (L5).

Det kan se ut til at lærerne viser til programmeringens komplekse rolle i form av både en brobygger, men også en sammensmelting av kompetanser. Kompetansene er ikke knyttet til et enkelt fag, men flere. Dette samsvarer med kompetanseområdene i (NOU 2015:8) nevnt innledningsvis i dette drøftingskapittelet. Ved å bruke programmeringens visuelle muligheter kan elevene i arbeidet med blant annet geometri konstruere sin egen kunnskap i tråd med Piagets konstruktivistiske visjon (Feurzeig et al., 2011). Programmeringens mulighet for visuell fremtoning, kan skape muligheter for elevene for å følge med på sin egne tanker (Resnick et al., 2009). Det kan se ut til at programmeringen settes inn i en kontekst der de bruker matematikk på nye måter for å løse mer praktiske oppgaver. L1 og L5 uttrykker at programmeringen kan bidra til at bruken av matematikk kan få et formål, og gir dermed et bilde av hvordan matematisk kunnskap kan komme til nytte. Dette kan tolkes å være noe i motsetning til den tradisjonelle matematikken, med mengdetrening og gjentakende oppgaver. Dermed kan programmeringen ved sine visuelle muligheter og sine ulike representasjoner av matematiske konsepter bidra til dybdelæring.

6.3.2 Matematiske konsepter nevnt viser til mulighetene lærerne ser

Lærerne viser til flere matematiske konsepter som kan knyttes opp mot arbeidet med programmering, i tillegg viser tre av lærerne, L1, L3 og L5, på forespørsel til hvilke kompetansemål som kan nås ved bruk av programmering. Der særlig to av lærerne L1 og L3, konkluderer med at stor sett alle de matematikkfaglige kompetansemålene tilhørende deres representative trinn, kan nås ved bruk av programmering. Dette kan vise til en stor bredde i læringsmulighetene lærerne ser, og det kan se ut til at de ser på muligheten for å ta i bruk programmering som metode. Misfeldt et al. (2019), trekker frem at lærere i deres undersøkelse hadde vansker med å se sammenhengen mellom programmering og matematikk. Det at noen av lærerne i denne undersøkelsen knytter samtlige kompetansemål fra deres representative trinn til bruken av programmering, kan vise at flere av lærerne i denne undersøkelse kan se sammenhengene. I tillegg kunne blant annet S1 også skape spesifikke koblinger mellom programmeringsspesifikke konsepter og matematiske konsepter (Se . Kap. 5.5). Videre blir det også uttrykt at elevene skal ha kjennskap til de matematikkfaglige konseptene fra før, før de arbeider med det gjennom programmering. Spekteret av matematiske konsepter, som lærerne trekker frem, spenner fra de spesifikke konseptene til de overordnede konseptene i likhet med de matematiske konseptene i Kap. 3.5 Matematisk læringspotensial. De spesifikke konseptene blir konkret nevnt, men de overordnede

konseptene ser i større grad ut til å vises til gjennom handlinger og ferdigheter de mener elevene skal ta i bruk.

6.3.2.1 Tall og tallregning; algoritmer, symboler og regnerekkefølge

Lærerne viser til koblinger mellom aritmetikk, symboler og programmeringsarbeidet. Både de aritmetiske operasjonene, regnerekkefølgen og symbolene brukt i arbeidet blir uttrykt som en del av læringsutbyttet ved arbeidet med programmering (Bueie, 2019; Haraldsrud et al., 2020; Norstein & Haara, 2018). Samtidig så vi henvisninger til regneoperasjonene tilknyttet de firegneartene. Calao et al. (2015) viser til hvordan programmeringsarbeidet i forbindelse med matematikk kan skape muligheter for at elevene utvikler en innsikt i likhetene mellom algoritmene. Dette finner vi igjen blant lærerne som selv skaper koblinger mellom algoritmer i matematikk og algoritmene brukt i programmeringskontekstene. Feilsøking blir også trukket frem i tilknytning til algoritmene brukt i matematikken der feilsøking som ferdighet blir trukket frem som nyttig også i utregning.

Henvisninger til feilsøking, trinnvis oppstilling og trinnvis løsning av problemer, kan alle knyttes til algoritmisk tenking og problemløsing (McVeigh-Murphy, 2019; Nygård, 2018; Weintrop et al., 2015). I tillegg skapes det gjennom det lærerne sier en sammenheng mellom det lærerne knytter til algoritmisk tenking og aritmetikken. Dette gjør de ved å forteller om koblingen mellom det å ta for seg en oppgave trinn for trinn og en trinnvis utregning med blant annet de fire regneartene. Videre trekker lærerne frem utholdenhet som et viktig konsept i arbeidet ikke bare med programmering, men også i matematikk. Det handler om å ikke gi opp, også dette er en ferdighet som kan knyttes til algoritmisk tenking (Weintrop et al., 2015).

6.3.2.2 Målestokk og koordinatsystemer

Forsstøm og Kaufmann (2018) uttrykker hvordan elevene kunne utvikle sofistikerte matematiske ideer i arbeidet med programmeringsoppgaver gjennom resonnering sammen med andre elver. Eksemplet, nevnt av L5, ved bruk av målestokk er et eksempel på ganske sofistikert bruk av matematiske konsepter. I tillegg skiller dette eksemplet seg fra det Forsstøm og Kaufmann (2018) sier ved at eleven arbeidet alene. Dette er også det eneste eksemplet på bruk av målestokk. Kap. 3.5.2 trekker frem måleenheter, men det finnes ingen henvisning til arbeidet med målestokk. Både L1 og L2 vider til måleenheten tid, som vi kan finne igjen i forskningen til både Benton et al. (2018) og Calder (2010)Men det finnes også eksempler på omgjøring og arbeid med måleenhetene millisekunder og sekunder i analysen, samt henvisninger til arbeid med tid (Benton et al., 2018; Calder, 2010).

Ke (2013), Calder (2010) og Taylor et al. (2010) åpner for at det i de fleste undervisningsopplegg i arbeidet med programmering finnes muligheter for å arbeide i et koordinatsystem. Det er hovedsakelig en av lærerne som trekker frem dette i intervjuet som en del av det elevene kan arbeide med i undervisningsopplegget som blir tatt i bruk. L3 trakk dette frem som en del av læringsutbyttet ved undervisningsopplegget. Til tross for at vi i Kap. 3.5.2 Spesifikke konsepter, får innblikk i mulighetene for å arbeide med koordinatsystemer, blir dette likevel vektlagt i liten grad av lærerne.

6.3.2.3 Geometri

Særlig fremtredende i litteraturen er bruken av programmering for å arbeidet med geometri (Forsstøm & Kaufmann, 2018). Clements og Merdith (1992) fremhever muligheten for at arbeidet med programmering kan bidra til utvikling innenfor van Heiles stadier for geometrisk tenking. Ingen av lærerne tok i bruk geometri i den observerte undervisning. Likevel kunne flere vise til at de hadde brukt dette tidligere, og de vektlegger den visuelle funksjonen programmeringen har. Der mulighetene for å justere, og leke seg med de ulike formene blir trukket frem. Geometrien blir trukket frem både i todimensjonal form, men også tredimensjonal form. På denne måten viser lærerne til hvordan man i likhet med det Feurzeig et al. (2011) forteller kan manipulere de geometriske objektene i et digitalt format og dermed utforske mulighetene og kjennetegnene ved figurer. Den visuelle funksjonen kan videre skape muligheter for å arbeide med geometri på et metakognitivt nivå, da en stadig kan følge med på endringene som gjøres eller som settes i spill (Resnick et al., 2009).

6.3.3 Mulighetene lærerne ser

Oppsummert knytter lærerne matematikkspesifikke konsepter som tall og tallregning, måling, geometri og statistikk og sannsynlighet til læringsutbyttet som ligger i programmeringsarbeidet. Dette i tillegg til de overordnede konseptene som algoritmisk tenking og problemløsing. Det kan virke som at lærerens egen erfaring og muligens kompetanse setter begrensing for hvilke muligheter de ser ved bruken av programmering.

6.3.4 Matematiske konsepter observert viser muligheter som finnes i undervisningen

Lærerne nevner flere matematiske konsepter. Noen av disse kunne observeres i undervisningen, men ikke alle. Dette er naturlig da jeg kun observerte enkelte undervisningsopplegg. Likevel ble det også observert matematiske konsepter i undervisningen som lærerne ikke nevnte, hverken i tilknytning til selve opplegget eller som et generelt utbytte av undervisningen. Da enkelte matematiske konseptener ikke så ut til å bli vektlagt, ser det ut

til at det oppstod muligheter som ikke ble utnyttet. De matematiske konseptene som ble observert kan si noe om hvilke matematikkfaglig potensiale som lå i oppgavene. Til hvilken grad dette potensialet ble utnyttet er en annen sak, og vil bli diskutert i neste kapittel.

6.3.4.1 Måling

Oppgavene der elevene tar i bruk måling av tid i form av millisekunder og sekunder kan knyttes til praktiske kontekster der tiden knyttes til avstand hos S2. Hos S1 knyttes den opp mot den reelle funksjonen til et trafikklys som må lyse i riktig rekkefølge, men også til riktig tid. Oppgavene åpnet derfor for muligheten til å arbeide med matematikken i en praktisk sammenheng (Calao et al., 2015). Der oppgavene i likhet med prosjektet til Calder (2010) oppfordrer til utforsking av tid. Det blir også observert matematiske konsepter som justering av størrelser, hastighet og retning. Selv om en fikk inntrykk av at L3 åpnet opp for en felles utforsking av hastigheten til karakterene i spillet og hva som påvirker denne, kan det likevel oppfattes som forbigående. Forbigående fordi elevene i mindre grad ble involvert i utforskingen, og fordi elevene i selve arbeidet kun fulgte oppskriften og de verdiene som lå i denne, Det ble i liten grad gjort mer eksplisitt.

6.3.4.2 Matematikkfaglige begreper

I tillegg til arbeid med matematikken i praksis, oppstår det flere muligheter for å arbeide med forståelsen av matematikkfaglige begreper. Eksempler på dette er «forstørre», «forminske», «telle ned» og «variabler» (L3 og L4). De matematiske begrepene knyttes da opp mot praktiske kontekster (Sevik & m.fl., 2016). Begrepet «variabler» går igjen hos flere, og blir brukt aktivt av lærerne. Likevel brukes det ikke av elevene, og de ser heller ikke ut til å knytte noen forståelse til dette begrepet. Dette gjelder også de andre observasjonene, der det kan se ut til at bruken av begreper ikke blir vektlagt i noen særlig grad. Begrepene brukes, men det arbeides i liten grad med betydningen eller innholdet i begrepene. På en annen side ser det ut til at elevene arbeider veldig praktisk med begrepene millisekunder, sekunder og prosent, da disse legger føringer for hvordan de skal løse oppgavene de arbeider med.

6.3.4.3 Algoritmisk tenking og problemløsing

Weintrop et al. (2015) trekker frem ti ferdigheter som kan knyttes til algoritmisk tenking. Gjennom lærernes uttalelser og elevenes arbeid ser flere av disse ferdighetene til å komme til uttrykk. Hos både S1 og S2 kunne vi observere at elevene fikk deler av en kode som de skulle utvikle videre. Målet var satt, men veien frem til målet var åpent. Elevene måtte i den forstand kunne håndtere åpne oppgaver (Weintrop et al., 2015). Ved å samarbeide måtte elevene

formidle ideer, og på denne måten finne en måte å representere disse ideene til samarbeidspartneren. Gjennom kodenes trinnvise oppbygging, ble løsningene representert på en algoritmeliknende måte, der en trinn for trinn kunne se hva de hadde satt sammen og hvordan koden fungerte (Bueie, 2019). Elevene ble videre oppfordret til å søke etter feilene sine selv ved at de måtte bruke prøve og feile metoden. Dette kan på sikt hjelpe elevene med å finne og kjenne igjen feil i sine digitale algoritmer. På denne måten så elevene ut til å evaluere sine egne koder og løsninger (Haraldsrud et al., 2020). Elevene selv kunne også peke på disse ferdighetene i sin utdyping av strategiene de tok i bruk i arbeidet (se Kap.5.6). Disse ferdighetene kan igjen knyttes til ferdigheter knyttet til algoritmisk tenking. Ingen av elevene så ut til å være drevne algoritmiske tenkere, men noen hadde mer erfaring enn andre. Likevel så alle elevene ut til å være i et utviklingsstadium der en stadig arbeidet med konsepter og ferdigheter i tilknytning til algoritmisk tenking.

Strategier for problemløsing ser videre også ut til å ligge i arbeidet elevene gjør. I Kap. 6.2 Pedagogiske strategier som brobygger, blir strategien *Explore*, trukket frem. Denne knyttes opp mot blant annet «tinkerability» og heuristisk matematikk (Resnick & Rosenbaum, 2013; Schoenfeld, 1985). De overordnede konseptene kommer dermed til uttrykk gjennom det elevene gjør. Det samme gjelder erfaring med resonnering, der elevene gjennom samarbeidet og klassesamtalene blir oppfordret til å resonnere sammen med en samarbeidspartner. Dette samsvarer med det Calao et al. (2015), Calder (2010); Ke (2013); Taylor et al. (2010) forteller om økt erfaring og også økte resonneringsferdigheter.

6.3.5 Matematiske konsepter nevnt og observert, men ikke utnyttet

Lærerne trekker frem mange muligheter for å arbeide med programmering i en matematikkfaglig kontekst. Disse nevnes i intervjuene. Særlig nevner de mange matematikkfaglige kompetansemål, der programmering kan brukes for å nå disse. Likevel virket det også som at utgangspunktet for bruken av programmering tar lite utgangspunkt i de matematikkspesifikke kompetansemålene. Dette til tross for at kompetansemålene er fagbestemmende for matematikk, ser de likevel ut til å være de mer programmeringsspesifikke konsepter knyttet til de kompetansemålene som velges. Observasjonene av undervisningen viser at til tross for at det ikke blir nevnt som hovedformål ved undervisningen, arbeider elevene med flere ulike matematiske konsepter. Dette kan være alt fra begreper og ulike måleenheter til overordnede konsepter som algoritmisk tenking og problemløsing. Til tross for at elevene arbeider med mange konsepter, fremstår flere av disse forbigående. Det vil si elevene arbeider med dem, men de vektlegges ikke ytterligere. Dette

kan vise at selv om lærere uttrykker en sammenheng, betyr ikke det nødvendigvis at de benytter seg av sammenhengen. Det kan ser ut til at det er et sprik mellom det lærerne forteller og mulighetene de uttrykker, og det de faktisk bruker programmeringen til. I tillegg vil det at en lærer legger opp til et undervisningsopplegg som skal benytte seg av sammenhengen mellom arbeidet med programmering og matematikken, ikke nødvendigvis sørge for at potensialet blir nådd. Videre ser det ut til at det å bruke programmering tar tid. Både det å sette av tid i undervisningen og det å lære elevene de grunnleggende programmeringsferdighetene som det kan virke som elevene bør lære seg for å kunne ta det i bruk. L1 uttrykte, som tidligere nevnt, behovet for «f*laskepåfylling*» i starten. Dette kan også være grunnen til at fokuset på det som foregår i undervisningen i mindre grad legges på det matematikkfaglige, og i større grad ferdigheter tilknyttet programmeringsarbeidet.

6.4 Overføring av kunnskap og bevisstgjøring

Kap. 6.2 Pedagogiske strategier som brobygger viser at elevene ved å ta i bruk pedagogiske strategier, kan arbeide med matematisk kompetanse. Det betyr ikke at de har tilegnet seg disse kompetansene og kan sies å ha matematisk kompetanse, men at de er i en prosess der det legges til rette for at disse kan utvikles videre. I dette delkapittelet vil jeg se på overføring av kunnskap mellom arbeidet med programmering og det matematiske konseptene som kommer til uttrykk. Hva som faktisk overføres til elevene er vanskeligere å si noe om, men vi kan se på hva elevene arbeider med og hva elevene opplever at de har arbeidet med.

S1 har som formål at elevene skal bruke tidligere lærte ferdigheter i en ny kontekst. I arbeidet ser elevene også ut til å gjøre dette. De tar i bruk det de har lært tidligere om løkker for å løse oppgaven de har fått. I tillegg bruker de kunnskapen de har lært om begrepet «millisekunder». Dette kan samsvare med det det Salomon og Perkins (1989) kaller for «high-road of transfer» i form av «backward reaching transfer» (Salomon & Perkins, 1989). Da elevene fortsatt ikke ser ut til å ha integrert programmeringsferdighetene ser de ut til å tenke tilbake på de erfaringene de har fra tidligere. Liknende kan vi finne igjen hos S2, der elevene bruker programmering for første gang, og i større grad ser ut til å vende tilbake til de erfaringene de har med avstand, tid og retninger.

Selv om det kan se ut til at det i enkelte tilfeller skjer en vellykket overføring av kunnskap, kan det i andre tilfeller se ut til at noe av overføringen av den matematikkfaglige kunnskapen går tapt underveis i arbeidet. Elevene blir gjort bevisste på bruken av matematikken i programmeringsarbeidet, men opplever ikke at de bruker konseptene i arbeidet. Elevene hos

S4, viser til reproduksjon av lærerens kode, dette kan vise til at de selv ikke opplever arbeidet som intendert. En av elevene (E4.1) uttrykker også selv at reproduksjon er grunnen til at de selv ikke har arbeidet med matematikk, dermed ser det heller ikke ut til å foregår noe form for overføring av kunnskap.

Bakgrunnen for hva som har skjedd, kan muligens belyses ved å se på hvordan E4.3 arbeidet med oppgaven. Der E4.3 uttrykker at hen fikk behov for å ta i bruk kunnskaper brukt tidligere i koden, og dermed også kunnskapen hen innehar om regning med prosent. Til tross for at eleven ikke behøver å søke langt tilbake, ser hen likevel tilbake på det hen har gjort tidligere og bruker denne kunnskapen for å klare den siste oppgaven. Dette er forankret i algoritmen brukt for å beregne avslaget for en vare. For denne eleven kan det se ut til at overføringen av kunnskap fra matematikken til programmeringsarbeidet hadde funnet sted som en form for «backward reaching transefer» (Salomon & Perkins, 1989). Det kan videre virke som at matematikken eleven kunne fra før ble brukt i en ny setting, med endring av kontekst og situasjon for bruk av den kjente algoritmen. Dette i motsetning til de andre elvene, der det kunne virke som at det ikke foregikk noen form for overføring av kunnskap mellom de to områdene. Forskjellen mellom det E4.3 har gjort og resten av elevene, er utformingen av oppgavene. Der E4.3 utvikler koden videre og på denne måten får et behov for å tenke selvstendig og søke egne løsninger på utfordringene hen møter på. Dette kan tyde på at elever som i større grad har arbeidet med problemløsende oppgaver, også i større grad opplevde at de arbeidet med matematiske konsepter. Særlig når disse ble gjort eksplisitte. Intervjuene med elevene hos S3, samt observasjonene hos S2 kan tyde på likende funn. Dette samsvarer med det Pea og Kurland (1984) forteller om at sannsynligheten for læring er størst dersom elevene selv engasjerer seg i oppgaver som involverer de nye konseptene.

I andre tilfeller kan det se ut til at læreren forsøker å skape en sammenheng mellom de matematiske konseptene som kommer til uttrykk i oppgavene, uten at elevene oppfatter denne sammenhengen. Et eksempel på dette finner vi hos S3 i klassesamtalen om x-verdiene i oppgaven. Læreren trekker frem x-verdiene, og at disse forteller hvor figuren er plassert. I intervjuet, trekker L3 frem koordinatsystemer som et mulig læringsutbytte. Likevel ga elevene ikke uttrykk for å ha arbeidet med matematikk. Dermed kan det se ut til at lærerens forsøk på å gjøre koordinatsystemer eksplisitt for elevene ikke var tydelig nok. En av grunnene til dette kan være at læreren snakker om «x-verdier», men ikke nevner begrepet «koordinater» eller «koordinatsystem». Det kan dermed virke som at elevene ikke har knyttet «x-verdi»-begrepet opp mot de forkunnskapene de har om koordinatsystemer. I tillegg var x-verdiene allerede satt

i oppgaven, slik at elevene ikke selv behøvde å ta stilling til hvilke verdier som bør legges inn. På denne måten knyttes ikke x-verdiene eksplisitt opp mot koordinatsystemene. Dette kan vitne om at undervisningsopplegget ikke i stor nok grad legger til rette for koblingen mellom arbeidet elevene gjør og konseptet elevene kjenner til fra før. Salomon og Perkins (1989) viser til at mulighetene for overføring av kunnskap er betydelig større dersom kunnskapen elevene skal sitte igjen med gjøres eksplisitt i den nye situasjonen, som her altså er arbeidet med programmering. Dermed kan det se ut til at denne kunnskapen ikke har blitt gjort eksplisitt nok. I tillegg uttrykker Benton et al. (2017) at det å skape sammenhengen mellom elevenes forkunnskaper og det matematikkfaglige i programmeringsarbeidet vil være lærerens oppgave. Undervisningsopplegget hos S3 ser heller ikke ut til å være visuelt ledende, da hverken koordinatene eller rutenett er synlig i skjermen annet enn ved endring av x-verdier. På denne måten mister også undervisningsopplegget programmeringens visuelle styrke (Haraldsrud et al., 2020) og dermed også muligheten for å følge med på egen tenking rundt bruken av koordinater (Clements, 1984).

På en annen side ser elevene hos S3 ut til å ta i bruk strategier. Strategienes tilknytning til de matematiske kompetansene har blitt begrunnet i Kap. 6.2 Pedagogiske strategier som brobygger. Noen av disse strategiene som for eksempel å følge instruksjonen eller å feilsøke ved hjelp av instruksjonen ser ut til å bli i ferd med å integreres i elevene, der de bevist og ubevisst tar i bruk disse. På denne måten kan det se ut til at strategiene for elevene kan knyttes til «low-road of transfer» (Salomon & Perkins, 1989). Der det etter mye øvelse ikke krever veldig mye kognitivt av elevene å ta i bruk disse strategiene, som har sin tilknytning til blant annet algoritmisk tenking.

Ved å legge til rette for både bevisstgjøring av hvilke matematikkfaglige konsepter elevene arbeider med, og i tillegg legge til rette for at elevene får bruk for konseptene i arbeidet med oppgaven. Ser elevene i større grad ut til å oppleve at de har arbeidet med matematikk og det ser i større grad ut til at det foregår «transfer» av matematisk kunnskap til arbeidet med matematikk i en programmerings kontekst. Dette samsvarer med de opplevelsene elevene hos S1 hadde, at de hadde arbeidet med ulike former for matematikk. Likevel, kan de overordnede konseptene uttrykt som ferdigheter, se ut til å overføres til elevene og over tid muligens bli en integrert del av elevenes repertoar, uten at disse blir spesifisert mer enn at elevene blir oppfordret til å arbeide på den gitte måten.

6.5 Lærerens valg danner grunnlaget for potensialet

Lærernes rolle blir svært viktig i arbeidet med å legge til rette for læring (Benton et al., 2016; Benton et al., 2017; Salomon & Perkins, 1989). Det er lærerne som skal gjøre det faglige innholdet eksplisitt. Som vi ser i analysen kan det videre se ut til at det er lærerne som gjennom valg av metoder, valg av oppgaver og utformingen av oppgavene bestemmer hva læringspotensialet til arbeidet med programmering skal være. I tillegg kan det se ut til at det er lærerne som påvirker til hvilken grad læringspotensialet blir utnyttet og til hvilken grad dette omgjøres til et læringsutbytte. Lærerne uttrykker selv at mulighetene for læring av matematiske konsepter er mange og knytter dette ikke kun til overordnede matematiske konsepter som problemløsing og algoritmisk tenking, men også spesifikke konsepter som tallregning, geometri, koordinatsystemer, måling, statistikk og sannsynlighet. Dermed er det opp til læreren hvor vidt matematikken eksplisitt trekkes frem og til hvilken grad elevene gjøres beviste på de matematikkfaglige koblingene.

Taylor et al. (2010) mener at Scratch åpner for utforsking av allerede kjent matematisk kunnskap, men er mindre egnet for innlæring av nye konsepter. Dette kommer også til uttrykk gjennom lærernes formulering av hvordan de velger oppgavene. Elevene må ha kjennskap til det matematikkfaglige innholdet fra før. Lærerne forteller at de ved å forberede seg godt, teste undervisningsoppleggene selv og reflektere over hvilke utfordringer en kan møte på, forbereder seg til å undervisning i programmering.

Det ingen av lærerne trekker frem er hvordan de i planleggingsarbeidet tar hensyn til det matematikkfaglige utbyttet. Forberedelse av undervisningen vil i like stor grad kreve en refleksjon over hva innholdet og potensialet kan være. I følge Calder (2010) bør det heller ikke være vanskelig å finne de matematiske konseptene i arbeidet med Scratch da de aller fleste oppgavene i stor grad involverer en eller annen form for matematiske konsepter. Gjennom bevisst søken etter passende oppgaver, eller konsepter i oppgavene, vil forberedelsene videre kunne bidra til å gjøre de matematiske konseptene eksplisitte i undervisningen. I tillegg til at undervisningsoppleggene bør sørge for at oppgaven åpner for å bruke de matematiske konseptene i problemløsende settinger. Settingene bør igjen åpne for selvstendig tenking og resonnering sammen med andre.

I likhet med det Clements og Merdith (1992) uttrykker, finnes det flere veier inn til det å bruke programmering for å arbeide med matematikk. Vi har sett eksempler der programmeringsarbeidet har åpnet for likheter mellom programmeringen og tradisjonelt

arbeid med matematiske konsepter, men vi har også sett at programmeringen har åpnet for en mer problemløsende vei inn til de matematiske konseptene. Det som likevel ser ut til å være felles for arbeidet med matematiske konsepter ved bruk av programmering, er tydeligheten i hva de matematiske konseptene i arbeidet er. For å kunne se disse, blir dermed tryggheten i bruken av programmering videre sentralt, og det blir et behov for «subject matter knowledge» (Stigberg & Stigberg, 2020). Dette kommer til uttrykk i at de lærerne som hadde mest erfaring i bruken av programmering også i størst grad, under intervjuene, klarte å koble matematikkfaglige konsepter til arbeidet med programmering. I tillegg til at de formidlet disse som mulige læringsutbytter ved arbeidet med programmering.

6.6 Drøftings konklusjon.

I dette drøftingskapittelet kan vi se at lærerne ser ut til å se mange muligheter for å ta i bruk programmering i en matematikkfaglig kontekst. Det at programmeringen i all hovedsak brukes tverrfaglig og for motivasjonens skyld, ser i liten grad ut til å være et hinder for den matematikkfaglige tilknytningen. Likevel må en ta høyde for at begrepet matematikk i denne oppgaven har en vid forståelse og favner bredt, akkurat som Niss et al. (2002) sin definisjon av matematisk kompetanse. Gjennom metodene og strategiene som blir tatt i bruk ser elevene ut til å arbeide med kompetanser som både kan knyttes til programmeringsarbeidet, men også matematiske kompetanser. Det er kanskje her den viktigste koblingen til matematikken skjer, i tillegg til valg av kompetansemål. Til tross for at lærerne nevner og trekker frem flere konkrete matematiske konsepter, ser de likevel ikke ut til å få en betydelig plass i selve undervisningen. Dette kan vise til at det matematikkfaglige potensialet som ligger både i de observerte oppgavene, men også i lærernes generelle formening om det matematikkfaglige læringsutbyttet, ikke blir utnyttet til det fulle. Dermed kan det se ut til at lærerne legger til rette for at bruken av matematikken i programmeringsarbeidet i stor grad handler om å «studere programmering ved hjelp av matematikk» fremfor at matematikken ligger som hovedformål (Kaufmann et al., 2018, s. 81). Elevene bruker altså matematiske kompetanser og konsepter i arbeidet med å lære seg å programmere, men programmeringsarbeidet spiller hovedrollen i undervisningen.

7 Konklusjon

I de neste avsnittene vil jeg i lys Kap. 6 Drøfting, forsøke å besvare problemstillingene lagt fram i Kap. 2 Problemstilling. Det finnes ingen «enkle» svar på noen av problemstillingene. Dette er fordi oppgaven favner bredt, og både matematikk og programmering har vide definisjoner. I tillegg grunner resultatet i et begrenset utvalg, med et begrenset antall informanter. Derfor vil resultatene for denne undersøkelsen være gjeldende for det utvalget, og kan ikke generaliseres ytterligere. Likevel kan det gi et bilde av hvordan, og til hvilken grad det matematiske læringsutbyttet knyttes opp mot programmeringsarbeidet på mellomtrinnet. Jeg vil først ta for meg delproblemstillingene, deretter den overordnede problemstillingen.

1) Hva er det matematiske læringspotensialet i arbeidet med programmering?

Den første problemstillingen er av teoretisk karakter, og har et forholdsvis bredt spenn. I Kap. 3 Teoretisk bakgrunn, finner vi matematisk læringspotensial både i form av overordnede matematiske konsepter, samt matematikkspesifikke konsepter. Til tross for noe sprikende funn, kan dette vise til at arbeidet med programmering har et omfattende potensiale for læring og arbeid med matematikk. Øverst troner muligheten for å arbeide med algoritmisk tenking og problemløsing. Deretter kommer muligheten for å gjøre matematikken mer visuell og gi programmeringen en modellerings- og representasjonsrolle. Arbeidet med programmering kan videre bidra til erfaringer og oppfordring til resonnering. Til slutt kommer de matematikkspesifikke konseptene, som det er mange av, og som det må legges spesifikt til rette for. Her finner vi bruken av tall og regning med tall på ulike vis, bruk av og forståelse for algoritmer, bruk av og gjenkjennelsen av mønstre, variabler, algebra og funksjoner. Videre er særlig det visuelle aspektet nyttig i arbeid med geometri og koordinatsystemer, måling og bruken av måleenheter. Det matematikkfaglige læringspotensialet er stort og mangfoldig. Det kan virke som at en kan knytte arbeidet til det matematiske konseptet en finner hensiktsmessig.

2) Hva anser lærerne at det matematiske læringsutbyttet er?

Den andre problemstillingen, er knyttet til hva lærerne selv ser av muligheter. Lærerne i studien knytter læringsutbyttet i stor grad til overordnede konsepter som løsning av problemer og algoritmisk tenking. I tillegg knyttes det i stor grad til programmeringens mulighet til å skape flere perspektiver på det matematikkfaglige innholdet, spesielt i geometri. Lærerne trekker videre frem matematikkens bruk av algoritmer, regning og strategier med tall, samt

automatisering av disse som læringsutbyttet. Tabell 7.1 sammenligner de potensielle læringsutbyttene som lærerne nevner med de konseptene som ble gjort rede for i Kap. 3.5 Matematisk læringspotensial. Det kan videre se ut til at det er lærerens erfaringer som setter begrensninger for hva de anser som læringsutbytte. Både erfaringen deres i form av opplæring innen programmering, men også erfaring i bruken av ulike programmeringsverktøy i undervisningen.

Tabell 7.1 Tabellen viser matematisk læringspotensial uttrykt i teorien versus matematisk læringsutbytte uttrykt av lærerne

Matematiske konsepter	**Matematikkfaglig læringspotensialet uttrykt i teori**	**Matematikkfaglig læringsutbytte uttrykt av lærere**
Overordnede konsepter		
Algoritmisk tenking	X	X
Problemløsing	X	X
Modellering/representasjoner	X	X
Resonnering	X	
Spesifikke konsepter		
Algoritmer og symboler	X	X
Algebra	X	X
Mønster og mønstergjenkjenning	X	X
Tallforståelse	X	
variabler	X	X
Funksjoner	X	X
Koordinatsystem	X	X
Geometri	X	X
Måling og måleenheter	X	X
Data, statistikk og sannsynlighet	X	X

3) Hva gjør lærerne for å implementere programmeringen på matematikkfagets premisser?

Funnene til de to første problemstillingene, er også relevante for den tredje. Til tross for at lærerne trekker frem et mangfold av matematiske konsepter som potensielt læringsutbytte ved bruk av programmering, ser det ut til at det vektlegges i liten grad i undervisningen. Formålene og det som formidles til elevene har i liten grad et matematikkfaglig fokus, men handler om å lære seg å programmere med et konstruktivistisk fokus. Derimot viser arbeidsmetodene og strategiene at arbeidet likevel bidrar til at elevene tar i bruk matematiske kompetanser som problemløsingskompetanse, kommunikasjon og resonneringskompetanse, symbol og formalkompetanse, tankegangskompetanse. I tillegg til disse matematiske kompetansene er det valg av kompetansemål som først og fremst skaper den matematikkfaglige koblingen.

Lærerne i denne studien trekker også frem matematikkfaglige konsepter, men dette skjer mer forbigående, og blir i liten grad plukket opp av elevene. Dette til tross for at lærerne ser mange muligheter. Èn av de fem lærerne tok spesifikt i bruk programmering som metode i en matematikktime. Dette kan ikke generaliseres, da dette var første gangen læreren gjorde dette. Det som derimot er verdt å merke seg, er at motivasjonen for programmeringsarbeidet brukes som en vei inntil det matematiske innholdet der elevene ubevist arbeider med og tar i bruk matematiske konsepter og kompetanser.

Til hvilken grad knytter lærere det matematiske læringsutbytte til arbeidet med programmering på mellomtrinnet?

Som vi ser i Kap. 3.5 Matematisk læringspotensial, finnes det mange muligheter for å arbeide med matematikkfaglige konsepter. Videre gir lærerne uttrykk for å se mange muligheter, men disse mulighetene ser i mindre grad ut til å bli utnyttet. De spesifikke konseptene blir trukket frem i undervisningen, men likevel forbigående. Dette er heller ikke så rart da hovedformålet hovedsakelig ikke fremstår som matematikkrettet, men rettet mot samarbeid. Ved å legge til rette for samarbeid, oppstår det, ved at programmeringsarbeidet legges i en matematikkfaglig problemløsingssetting, muligheter for å arbeide med matematiske kompetanser. Disse matematiske kompetansene kan igjen knyttes opp mot matematikk fra et problemløsende perspektiv, i form av strategier som gir hentydning til algoritmisk tenking. Dermed kan en si at tilknytning til det matematiske læringsutbyttet først og frem kommer til sin rett gjennom bruken av strategier.

I lærernes planlegging kan det også se ut til at matematikken kommer i andre rekke, da selve planleggingen i mindre grad tar stilling til de mer spesifikke matematiske konseptene som

kommer til uttrykk. Lærerne i denne undersøkelsen forbereder seg på det som angår programmeringsarbeidet, og de konseptene som kan skape utfordringer for elevene på dette området, men i mindre grad det matematiske.

Det er i planleggingsarbeidet at en må starte for å øke graden av matematikkfaglig arbeid. Planlegging som leder til innsikt i programmeringsarbeidet, men som også handler om å finne det matematikkfaglige potensialet som ligger i hver oppgave. I tillegg kan drøftingen vise hentydning til at for at overføringen av de matematikkfaglige kunnskapene skal skje til arbeidet med programmering, må ikke bare lærerne være bevisste allerede i planleggingsfasen. De må bygge på de forkunnskapene elevene har i tilknytning til det matematikkfaglige innholdet, samt åpne for at elevene får arbeide med de matematiske konseptene i en utforskende setting, der de selv får behov for å ta stilling til og ta i bruk de matematiske, kunnskapene, ferdighetene og kompetansene. Det kan likevel vise at det finnes muligheter for å spesifikt planlegge for arbeid med matematikk i arbeidet med programmering, men at lærerne må ta valget og planlegge undervisningen fra et matematikkfaglig ståsted. Det er som Salomon og Perkins (1989) påpeker; «If only we teach it, we are most likely to get it.» (Salomon & Perkins, 1989, s. 137).

8 Bruk i skolen og veien videre

Denne oppgaven viser gjennom litteraturen og de innsamlede dataene at det finnes mange matematiske konsepter som kan knyttes opp mot bruken av programmering. Både gjennom innholdet i oppgavene elevene skal gjøre, men også gjennom arbeidsmetodene og strategiene som kan benyttes i arbeidet. Gjennom «annerledes» matematikk kan både motivasjon, samarbeid og engasjement bidra til mange læringsmuligheter. Forskningen her viser at disse mulighetene finnes, men også at det krever at læreren er bevisst disse mulighetene og har verktøy for å kunne skape denne sammenhengen. Det kan se ut til at det er et behov for en kombinasjon av sammenlikning med den tradisjonelle matematikken, samt en løsrivelse fra dette der matematikken i større grad brukes problemløsende. Uten denne bevisstheten vil mulighetene for å dra programmeringsarbeidet inn på matematikkfagets premisser gå tapt, dette gjelder også den matematikkfaglige overføringsverdien. Dersom en i større grad trekker disse parallellene vil en potensielt få et læringsverktøy inn i skolen som åpner for å arbeide med matematikk på den moderneelevs premisser.

Dette krever arbeid fra lærernes side. Det er ikke kun å hoppe inni et ferdig undervisningsopplegg, som det finnes mange av i forbindelse med programmering. Læreren

må velge oppgaver med omhu, være bevist de konseptene som kommer til uttrykk, og benytte muligheten til å knytte arbeidet opp mot elevenes forkunnskaper. Dette betyr muligens at lærere må gjøre endringer i eksisterende undervisningsopplegg, og tenke nytt. For at dette skal kunne skje trenger man et større nettverk og flere lærere å spille på lag med.

Det er fortsatt behov for mye forskning på området, og i likhet med Bentons arbeid (Benton et al., 2016), vil det fremover være relevant å se på hva som kan gjøres for å legge til rette for implementering på matematikkfagets premisser og hva lærere gjør for å skape sammenhengen mellom programmeringsarbeidet og det matematikkfaglige innholdet. Mye av forskningen er i dag rettet mot ungdomsskolen og videregående opplæring, men hvis programmering virkelig skal få sin plass i opplæringen, bør det også forskes på læringspotensialet hos de yngre elevene. I tillegg bør en gi programmering en større plass i undervisningen, både i enkelt fag og på tvers av fagene. På denne måten vil en kunne støtte opp om innlæringen av de programmeringsspesifikke ferdighetene, som det tar tid å lære. Ved å tidlig begynne med dette, vil elevene på et tidligere tidspunkt kunne ta i bruk programmeringen som metode for å arbeide med matematikk, enn det de gjør i dag. Dersom vi ikke gjør det, vil vi i større grad fortsette å se på programmering som noe som «kommer i tillegg», fremfor et verktøy vi kan bruke. Dette vil muligens gjelde uansett fag.

Det er ingen tydelige funn på den faktiske overføringen av kunnskap mellom arbeidet med programmering og matematikk, hverken når det kommer til spesifikke konsepter eller overordnede konsepter. Det er i tillegg heller lite norsk forsking på bruken av programmering på mellomtrinnet, og hva programmeringsarbeidet faktisk brukes til. Forskning videre bør se nærmere på dette. Andre områder det er relevant å få mer kunnskap om er bruken av blokkprogrammering som en innledning til programmering. Dataene viser at elevene hovedsakelig arbeider med blokkprogrammering, til tross for at dette kanskje i større grad bør kalles koding enn programmering. I tillegg blir det etter hvert likevel en forventning om at elevene skal gå over til tekstprogrammering. Hvordan foregår den overføringen? Og hvor mye av det de har lært i arbeidet med tekstprogrammering tar elevene med seg videre i arbeidet med tekstprogrammering? Til tross for at jeg i denne oppgaven finner hentydninger til mange muligheter for å ta i bruk programmering for å arbeide med matematikk, er det likevel fortsatt et stort behov for videre forskning. Det er som en av lærerne i denne studien sier; vi må «bruke det fornuftig» (L4).

Kilder

Angeli, C. & Giannakos, M. (2020). Computational thinking education: Issues and challenges. *Computers in human behavior, 105*. Hentet fra https://doi.org/10.1016/j.chb.2019.106185

Autodesk. (2022). Thinkercad. Hentet 11.04 2022 fra https://www.tinkercad.com/

Benton, L., Hoyles, C., Kalas, I. & Noss, R. (2016). Building mathematical knowledge with programming: insights from the ScratchMaths project.

Benton, L., Hoyles, C., Kalas, I. & Noss, R. (2017). Bridging Primary Programming and mathematics: Some Findings of design research in England. *Digit Exp Math Educ, 3*, 115-138.

Benton, L., Saunders, P., Kalas, I., Hoyles, C. & Noss, R. (2018). Designing for learning mathematics through programming: a case study of pupils engagig with place value. *International Journal of Child-Computer Interaction, 16*, 68-76. Hentet fra https://discovery.ucl.ac.uk/id/eprint/10041424/

Berggren, S. & Jom, P. (2019). Fagartikkel: Lærerne er positive til programmering - men mangler kunnskap. Hentet 09.10 2021 fra https://www.utdanningsnytt.no/fagartikkel/fagartikkel-laererne-er-positive-til-programmering---men-mangler-kunnskap/220753

Bryman, A. (2016). *Social Research Methods* (5th edition. utg.)Oxford Univerity Press.

Bråting, K. & Kilhamn, C. (2021). The Integration of Programming in Swedish School Mathematics: Investigating Elementary Mathematics Textbooks. *Scandinavian journal of educational research, 1-16*.

Bueie, H. (2019). *Programmering for matematikklærere*. Oslo: Universitetsforlaget.

Calao, L. A., Moreno-Leòn, J., Correa, H. E. & Robles, G. (2015). *Developing Mathematical Thinking with Scratch. An Experiment with 6th Grade Students.* Innlegg presentert ved European Conference on Technology Enhanced Learning. EC-TEL 2015: Design for Teaching and Learning in a Networked World. , Toledo, Spain.

Calder, N. (2010). Using Scratch: An Intergrated Problem-Solving Approach to Mathematical Thinking. *Australian Primary Mathematics Classroom, 15*(4), 9-14.

Christoffersen, L. & Johannessen, A. (2012). *Forskningsmetode for Lærerutdanningene*. Oslo: Abstrakt forlag.

Clarke-Midura, J., Silvis, D., Shumway, J. F., Lee, V. R. & Kozlowski, J. S. (2021). Developing a kindergarten computational thinking assessment using evidence-centered design: the case of algorithmic thinking. *Computer Science Education, 31*(2), 117-140. Hentet fra https://www-tandfonline-com.ezproxy.oslomet.no/doi/epub/10.1080/08993408.2021.1877988?needAccess=true

Clements, D. H. (1984). Effevts of comuputer programming on ypung children's cognition. *Journal of educational psychology*.

Clements, D. H. & Merdith, J. S. (1992). Research on Logo: Effects and Efficacy *Logo Foundation*.

Code Club UK. (Undated). Ørkenløp (Anne-Marit Gravem, Overs.). Hentet fra https://oppgaver.kidsakoder.no/scratch/orkenlop/orkenlop

Copeland, B. J. (2020). The modern History of computing. I E. N. Zalta (Red.), *The Stanford Encyclopedia of Philosophy* (Winter 2020. utg.). Metaphysics Research Lab, Stanford Univerity. Hentet 08.05.2022 fra https://plato.stanford.edu/entries/computing-history/

European Schoolnet. (2015). *Computing our future. Computer programming and coding. Priorities, School curricula and initiatives across Europe.* Brüssel. Hentet fra http://www.eun.org/documents/411753/817341/Computing+our+future_final_2015.pdf/d3780a64-1081-4488-8549-6033200e3c03

Evans, M. K. (2017). The Seeds That Seymour Sowed. Hentet 28.02 2022 fra https://www.media.mit.edu/posts/the-seeds-that-seymour-sowed/

Fangen, K. (2010). *Deltakende observasjon* (2. utg.). Bergen: Vigmostad & Bjørke.

Feurzeig, W., Papert, S. A. & with a preface by Bob Lawler. (2011). Programming-languages as a conceptual framework for teaching

mathematics. *Interactive Learning Environments, 19*(5), 487-501. Hentet fra https://doi.org/10.1080/10494820903520040
Forsstøm, S. E. & Kaufmann, O. T. (2018). A literature review exploring the use of programming in mathematics education. *International Journal of Learning, Teaching and Educational Research, 17, No. 12*, 18-32.
Foundation, L. (2015). Logo History. Hentet 19.02.22 2022 fra https://el.media.mit.edu/logo-foundation/what_is_logo/history.html
Franklin, D., Hill, C., Dwyer, H. A., Hansen, A. K., Iveland, A. & Harlow, D. B. (2016). Initialization in Scratch: Seeking Knowledge Transfer. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, 217-222. Hentet fra https://dl.acm.org/doi/pdf/10.1145/2839509.2844569
Grover, S. & Pea, R. (2013). Computatiopnal Thinking in K-12: A Review of the state of the field. *Educational Reseacher, 42*(1), 38-43.
Haraldsrud, A. D., Sveinsson, H. A. & Løvold, H. H. (2020). *Programmering i skolen*. Oslo: Universitetsforlaget.
Hinna, K. R. C., Rinvold, R. A. & Gustavsen, T. S. (2012). *QED 1-7. Matematikk for lærerutdanningen*. Kristiansand Høyskoleforlaget.
Horsten, L. (2022). Philosophy of Mathematics. I E. N. Zalta (Red.), *The Stanford Encyclopedia of Philosophy* (Spring 2022. utg.). Metaphysics Research Lab, Stanford University. Hentet 08.05.2022 fra https://plato.stanford.edu/entries/philosophy-mathematics/#ComPro
InterviewBit. (2021). Difference Between Coding and Programming. Hentet 10.01 2022 fra https://www.interviewbit.com/blog/difference-between-coding-and-programming/
Jones, J. L., Jones, K. A. & Vermette, P. J. (2009). Teaching mathematics understandings for transfer. *Teaching Mathematics and Its Applications: International Journal of the IMA, 28*(3), 145-149. Hentet fra https://doi-org.ezproxy.oslomet.no/10.1093/teamat/hrp008
Kaufmann, O. T., Stenseth, B. & Holone, H. (2018). Programmering i matematikkundervisningen. I Anne Norstein & F. O. Haara (Red.), *Matematikk undervisning i en digital verden* Oslo: Cappelen Damm Akademisk.
Ke, F. (2013). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. . *Computers & Education,* (73), 26-39.
Kidsakoder. (2020). Koding i skolen. Hentet 10. mai 2021 fra https://www.kidsakoder.no/skole/#toggle-id-15
Kilhamn, C., Bråting, K. & Rolandsson, L. (2021). Teachers' arguments for including programming in mathematics education
Kunnskapsdepartementet. (2017). *Overordnet del – verdier og prinsipper for grunnopplæringen.* Fastsatt som forskrift ved kongelig resolusjon. Læreplanverket for Kunnskapsløftet 2020. Hentet fra https://www.udir.no/lk20/overordnet-del/prinsipper-for-laring-utvikling-og-danning/kompetanse-i-fagene/?lang=nob
Kunnskapsdepartementet. (2019). *Læreplan i matematikk 1.-10. trinn* (MAT01-05). Fastsatt som forskrift. Hentet fra https://data.udir.no/kl06/v201906/laereplaner-lk20/MAT01-05.pdf?lang=nob
Kvale, S. & Brinkmann, S. (2019). *Det kvalitative forskningsintervju* (3. utg.). Oslo: Gyldendal Norsk Forlag
Lambi`c, D. (2010). Presenting practical application of mathematics by the use of programming software with easily available visual components. *Teaching mathematics and its applications,* (30), 10-18.
Linden, M. v. d. (2015). *"Å bygge er å tenke" - en oppgave om bruk av programmering i matematikkundervisningen* (Bacheloroppgave). Høgskolen i Oslo og Akershus.
Lær Kidsa Koding. (2021). Kodetimen. Hentet fra https://www.kidsakoder.no/kodetimen/

Maloney, J., Resnick, M., Rusk, N., Silverman, B. & Eastmond, E. (2010). The Scratch Programming Language and Environment. *ACM Transactions on Computing Education, 10*(4). Hentet fra https://web.media.mit.edu/~jmaloney/papers/ScratchLangAndEnvironment.pdf

Malterud, K. (2011). *Kvalitative Metoder i medisinsk forskning. En innføring.* (3. utg. utg., bd.). Oslo: Universitetsforlaget.

Matematikksenteret. (2017). Hva er MatteLIST og hvordan kan du bruke nettsidene? Hentet 10. mai 2021 fra https://www.mattelist.no/artikkel/om-mattelist

McVeigh-Murphy, A. (2019). Computational Thinking, Algorithmic Thinking, & Design Thinking Defined. Hentet 07.12 2021 fra https://equip.learning.com/computational-thinking-algorithmic-thinking-design-thinking

Meld.St. 28 (2015-2016). *Fag - Fagfordypning - Forståelse. En fornyelse av Kunnskapsløftet.* . Regjeringen. Hentet fra https://www.regjeringen.no/contentassets/e8e1f41732ca4a64b003fca213ae663b/no/pdfs/stm201520160028000dddpdfs.pdf

Micro:Bit. (Undated). About. Our History. . Hentet 28.02 2022 fra https://microbit.org/about/

Misfeldt, M. & Ejsing-Duun, S. (2015, 2015-02-04). Learning mathematics through programming: An instrumental approach to potentials and pitfalls. *CERME 9 - Ninth Congress of the European Society for Research in Mathematics Education* (s. 2524-2530). Hentet fra https://hal.archives-ouvertes.fr/hal-01289367

Misfeldt, M., Szabo, A. & Helenius, O. (2019). Surveying teachers' conception of programming as a mathematics topic following the implementation of a new mathematics curriculum. *HAL archives-ouvertes.fr*.

Moreau, H. N. (2021, 30.12 2021). Skal lære elevene koding, men forstår det ikke selv. *NRK.no*. Hentet fra https://www.nrk.no/innlandet/laerere-trenger-hjelp-til-a-knekke-koden-pa-koding-1.15781343?fbclid=IwAR2qJVTs2p81UMocIuDPvQDAchBD3OLCHeCIpPa-HYwaXM4dK88duqo9HL0

Niss, M., Jensen, T. H., Andersen, T. B., Andersen, R. W., Christoffersen, T., Damgaard, S., … Nissen, K. (2002). Kompetencer og matematiklæring. Ideer og inspiration til udvikling af matematikundervisning i Danmark. *Utdannelsesstyrelsens temahæfteserie,* (18). Hentet fra file:///C:/Users/Mischa/Downloads/Kompetencer%20og%20matematikl%C3%A6ring.pdf

Norstein, A. & Haara, F. O. (2018). *Matematikkundervisning i en digital verden* Oslo: Cappelen Damm Akademisk

NOU 2014:7. (2014). *Elevenes læring i fremtidens skole. Et kunnskapsgrunnlag.* . Hentet fra https://www.regjeringen.no/contentassets/e22a715fa374474581a8c58288edc161/no/pdfs/nou201420140007000dddpdfs.pdf

NOU 2015:8. (2015). *Fremtidens skole - Fornyelse av fag og kompetanser*. Kunnskapsdepartementet. Hentet fra https://www.regjeringen.no/contentassets/da148fec8c4a4ab88daa8b677a700292/no/pdfs/nou201520150008000dddpdfs.pdf

NOU 2015:13. (2015). *Digital sårbarhet - sikkert samfunn. Beskytte enkeltmennesker og samfunn i en digitalisert verden*. Justis- og bedreskapsdepartementet. Hentet fra https://www.regjeringen.no/contentassets/fe88e9ea8a354bd1b63bc0022469f644/no/pdfs/nou201520150013000dddpdfs.pdf

Nygård, K. (2018). *Programmering i skolen. Hvordan komme i gang?* Oslo: Pedlex.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. United States of America Basic books Hentet fra https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxha2hsYWdoZWF8Z3g6NzgyOWYxNWNjMjE5ZjVh

Pea, R. D. & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology, 2*(2). Hentet fra https://www.sciencedirect.com/science/article/pii/0732118X84900187?ref=pdf_download&fr=RR-2

Resnick, M., Maloney, J., Monory-Hernàndez, A., Rusk, N., Estmond, E., Brennan, K., … Kafai, Y. (2009). Scratch Programming for all. *Communications of the ACM, 52*(11). Hentet fra https://web.media.mit.edu/~mres/papers/Scratch-CACM-final.pdf

Resnick, M. & Rosenbaum, E. (2013). Chapter 10. Designing for Tinkerability. I M. Honey & D. E. Kanter (Red.), *Design, Make, Play. Growing the Next Generation of STEM Innovators.* New York: Routledge.

Salomon, G. & Perkins, D. N. (1989). Rocky roads to transfer: Rethinking mechanism of a neglected phenomenon. *Educational psychologist, 24*(2), 113-142. Hentet fra https://web-p-ebscohost-com.ezproxy.oslomet.no/ehost/pdfviewer/pdfviewer?vid=1&sid=a36b486f-3265-4d18-a75e-3a8247df1b02%40redis

Sanne, A., Berge, O., Bungum, B., Jørgensen, E. C., Kluge, A., Kristensen, T. E., … Voll, L. O. (2016). *Teknologi og programmering for alle. En faggjennomgang med forslag til endringer i grunnopplæringen- august 2016*.

Scherer, R., Siddiq, F. & Viveros, B. S. (2019). The Cognitive Benefits of Learning Computer PRogramming: A MEta-Analysis of Transfer Effects. *Journal of educational psychology, 111*(5), 764-792.

Schoenfeld, A. H. (1985). *Mathematical problem solving*. Orlando, Florida: Academic Press, inc.

Scratch Foundation. (2019). Scratch. I: https://scratch.mit.edu. Hentet fra https://scratch.mit.edu/projects/editor/?tutorial=getStarted

Sevik, K. & m.fl. (2016). *Programmering i skolen - Notat fra Senter for IKT i utdanningen* Senter for IKT i utdanningen Hentet fra https://www.udir.no/globalassets/filer/programmering_i_skolen.pdf

Shute, V. J., Sun, C. & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*.

Sinclair, N. & Petterson, M. (2018). The dynamic Geometrisation of Computer Programming. *Mathematical thinking and learning, 20*, 54-74.

Skemp, R. R. (1976). Relational Understanding and Instrumental Understanding. *Mathematics Teaching, 077*(December), 60. Hentet fra https://www-atm-org-uk.ezproxy.oslomet.no/write/MediaUploads/Journals/MT077/Relational_Understanding_and_Instrumental_Understanding_%E2%80%93_Richard_R._Skemp.pdf

Skott, J., Jess, K. & Hansen, H. C. (2013). *Matematikk for lærerstuderende. Delta Fagdidaktikk* (5. . utg.). Danmark: Forlaget Samfundslitteratur.

Skott, J., Skott, C. K., Jess, K. & Hansen, H. C. (2018). *Matematik for lærerstuderende: Delta 2.0 Fagdidaktikk, 1.-10. klasse*. Danmark: Samfundslitteratur

Smith, E. (2003). Statis and Change: Integrating Patterns, Functions, and Algebra Throughout the K-12 Curriculum. I *A Research companion to principles and standards for school mathematics* (s. 136-150). Reston, VA: National Council of Teachers of Mathematics.

Stigberg, H. & Stigberg, S. (2020). Teaching programming and mathematics in practice: A case study form a swedish primary school. *Policy futures in education, 18(4)* 483-496.

Super:Bit. (Undated). Hva er Super:Bit? Hentet 11.05 2022 fra https://www.superbit.no/hva-er-superbit/

Svorkmo, A.-G. & Valbekmo, I. (2020, 12.05.20). Rike oppgaver, resonnering og argumentasjon. Hentet 26.12 2021 fra https://www.matematikksenteret.no/blogg/rike-oppgaver-resonnering-og-argumentasjon

Taylor, M., Harlow, A. & Forret, M. (2010). Using a Computer Programming Environment and an Interactive Whiteboard to investigate some Mathematical Thinking. *Procedia Social and BEhavioral Sciences, 8*, 561-570.

Toxicode. (2017). Silent Teacher. Hentet 29.03 2022 fra https://silentteacher.toxicode.fr/

Universitetet i Oslo. (2021). Nettskjema-Diktafon. I(bd. Google Play). Hentet fra https://play.google.com/store/apps/details?id=no.uio.mobileapps.diktafon&hl=no

Utdanningsdirektoratet. (2019a). Algoritmisk tenkning. Hentet 31.08.21 2021 fra https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/

Utdanningsdirektoratet. (2019b). Hva er kjerneelementer? Hentet fra https://www.udir.no/laring-og-trivsel/lareplanverket/stotte/hva-er-kjerneelementer/

Utdanningsdirektoratet. (2020a). *Kjerneelement*. Oslo. Hentet fra https://www.udir.no/lk20/mat01-05/om-faget/kjerneelementer

Utdanningsdirektoratet. (2020b). PRogrammering og algoritmisk tenkingin; Fellesmodul 1-10; Del 1: Programmering og språk; Film: Programmering i skolen. Hentet 31.08 2021 fra https://bibsys.instructure.com/courses/387/pages/nb-film-programmering-i-skolen%7Cse-filbma-programmeren-skuvllas%7C?module_item_id=25400&lang=nb

Utdanningsdirektoratet. (2021a). Kompetansepakke; Programmering og Algoritmisk tenking; Matematikk 1-7; Del2: Algoritmisk tenking i matematikk; Aktivitet 1: Kjerneelementer og algoritmisk tenking. . Hentet 10.05 2022 fra https://bibsys.instructure.com/courses/387/pages/nb-aktivitet-1-kjerneelementer-og-algoritmisk-tenkning%7Cse-aktivitehta-1-guovddaselemeanttat-ja-algoritmmalas-jurddaseapmi%7C?module_item_id=27850&lang=nb

Utdanningsdirektoratet. (2021b). Kompetansepakke; Programmering og algoritmisk tenknign; Matematikk 1-7; Del 2: Algoritmisk tenkning i matematikk; Oversikt: Algoritmisk tenkining i matematikk. Hentet 31.08 2021 fra https://bibsys.instructure.com/courses/387/pages/nb-oversikt-algoritmisk-tenking-i-matematikk%7Cse-bajilgovva-algoritmmalas-jurddaseapmi-matematihkas%7C?module_item_id=27848&lang=nb

Utdanningsdirektoratet. (2021c). Programmering og algoritmisk tenkning; Matematikk 1-7; Del 4: Programmering i matematikk 5-7; Introduksjon til programmering på mellomtrinnet. Hentet 31.08 2021 fra https://bibsys.instructure.com/courses/387/pages/nb-introduksjon-til-programmering-pa-mellomtrinnet%7Cse-oahpasmuvvan-programmeremii-gaskadasis%7C?module_item_id=27870&lang=nb

Utdanningsdirektoratet. (2021d). Programmering og algoritmisk tenkning; Matematikk 1-7; Velkommen til matematikk 1-7. Hentet 31.08 2021 fra https://bibsys.instructure.com/courses/387/pages/nb-velkommen-til-matematikk-1-7%7Cse-buresboahtin-1-7-matematihkkamodulii%7C?module_item_id=27833&lang=nb

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L. & Wielensky, U. (2015). Defining Computational Thinking for Mathematics and Sience Classrooms. *J Sci Educ Technol, 25*, 127-147.

Wing, J. M. (2006). Computational Thinking *Comunications of the ACM, 49* (March).

9 Vedlegg

9.1 Samtykkeskjema lærer

Vil du delta i forskningsprosjektet
Programmering i matematikkundervisningen

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å å finne ut at hvordan lærere implementerer programmering i matematikkundervisningen og til hvilken grad de knytter det matematikkfaglige innholdet til arbeidet med programmering i faget. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål

Dette prosjektet er en masteroppgave der formålet er å se på hvordan programmering implementeres i matematikkundervisningen. Problemstillingen som er utgangspunktet for prosjektet er:

- Til hvilken grad knytter lærere det matematiske læringsutbyttet til arbeidet med programmering i matematikkundervisningen på mellomtrinnet?

Prosjektet tar utgangspunkt i analyse av litteratur om programmering, observasjon av undervisning av programmering i matematikkundervisningen, samt intervju av lærere og elever om blant annet det matematiske utbyttet av undervisningen.

Hvem er ansvarlig for forskningsprosjektet?

Henrik Forssell, Førsteamanuensis ved Oslomet er ansvarlig for prosjektet. Prosjektet gjennomføres av mastergradsstudent Mischa van der Linden.

Hvorfor får du spørsmål om å delta?

Prosjektet tar for seg matematikkundervisning på mellomtrinnet. I den forbindelse søker jeg lærere som i sitt daglige arbeid står for matematikkundervisning for grunnskolens 5., 6. eller 7.trinn. I utvalget søkes lærere med og uten opplæring i bruken av programmering. Totalt søkes 6-8 lærere som har mulighet til å stille sin undervisning åpen for observasjon og deretter

kan stille til intervju om innholdet av undervisningsøkten og programmering i matematikk undervisningen. Omtrent 3 lærere med opplæring eller formel kompetanse innen programmering og 3 lærere uten formell opplæring eller kompetanse. For å begrense søket rettes prosjektet mot skoler og lærere på Østlandet.

Hva innebærer det for deg å delta?

I dette prosjektet tar jeg i bruk flere metoder. Jeg ønsker å kombinere observasjon og intervju. Dersom du velger å delta i prosjektet, vil du observeres av en ikke deltakende observatør som vil notere seg observasjoner som er relevante for problemstillingen. Omfanget vil være èn til to observasjoner av fullstendige undervisningssekvenser. Både lærerens og elevenes arbeid vil observeres. Opplysningene vil i første omgang registreres som et manuelt notat, og deretter transkriberes elektronisk.

Observasjonsdeltakerne vil anonymiseres med koder allerede i det første notatet. Etter observasjonene vil du bli intervjuet. I den forbindelse vil spørsmålene blant annet omhandle planleggingen av undervisningen og valg undervis i undervisningssekvensen. I tillegg vil spørsmålene omhandle programmeringens og matematikkens rolle i undervisningssekvensen. Jeg vil ta lydopptak av intervjuene, og sitater fra disse kan brukes i masteroppgaven. Lydopptaket vil transkriberes og anonymiseres elektronisk.

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du når som helst trekke samtykket tilbake uten å oppgi noen grunn. Alle dine personopplysninger vil da bli slettet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg.

Ditt personvern – hvordan vi oppbevarer og bruker dine opplysninger

Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler opplysningene konfidensielt og i samsvar med personvernregelverket.

- Veileder for prosjektet og student og vil ha tilgang til opplysningene.
- Navnet og kontaktopplysningene dine vil erstattes med en kode som lagres på egen navneliste adskilt fra øvrige data. Datamaterialet vil lagres i en kryptert mappe. Lydopptak lagres sikkert i «Nettskjema».

- Deltakerne vil i liten grad kunne gjenkjennes i publikasjonen. Det vil publiseres observasjoner av hendelser/ samtaler samt til hvilken grad deltakerne har fått opplæring.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?

Opplysningene anonymiseres og lydopptak slettes når prosjektet avsluttes og oppgaven er godkjent, noe som etter planen vil være senest høsten 2023.

Hva gir oss rett til å behandle personopplysninger om deg? Vi behandler opplysninger om deg basert på ditt samtykke.

På oppdrag fra Oslomet har NSD – Norsk senter for forskningsdata AS vurdert at behandlingen av personopplysninger i dette prosjektet er i samsvar med personvernregelverket.

Dine rettigheter

Så lenge du kan identifiseres i datamaterialet, har du rett til:

- innsyn i hvilke opplysninger vi behandler om deg, og å få utlevert en kopi av opplysningene
- å få rettet opplysninger om deg som er feil eller misvisende
- å få slettet personopplysninger om deg
- å sende klage til Datatilsynet om behandlingen av dine personopplysninger

Hvis du har spørsmål til studien, eller ønsker å vite mer om eller benytte deg av dine rettigheter, ta kontakt med:

- Oslomet ved Henrik Forssell (veileder), Henrik.Forssell@oslomet.no og Mischa van der Linden (student), s188417@oslomet.no.
- Vårt personvernombud: Ingrid S. Jacobsen, tlf. 67235534, mail; personvernombud@oslomet.no, ved Oslomet.

Hvis du har spørsmål knyttet til NSD sin vurdering av prosjektet, kan du ta kontakt med:

- NSD – Norsk senter for forskningsdata AS på epost (personverntjenester@nsd.no) eller på telefon: 55 58 21 17.

Med vennlig hilsen

Henrik Forssell og Mischa van der Linden

(Forsker/veileder) (Student)

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet Programmering i matematikkundervisningen og har fått anledning til å stille spørsmål. Jeg samtykker til:

- □ å delta i observasjon.
- □ å delta i intervju.

Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet

---

(Signert av prosjektdeltaker, dato)

9.2 Intervjuguide Lærer

Intervjuguide Masteroppgave
Intervju av lærere

Dette er et arbeidsverktøy for oppgaven. Intervjuguiden er ledende for et semistrukturert intervju. Planen er at intervjuet skal foregå som en samtale mellom informanten og den som intervjuer. Ved behov vil relevante oppfølgingsspørsmål legges til.

1. Det er nå bestemt at programmering skal implementeres i flere ulike fag, deriblant matematikk.
    a. Hvilke erfaringer har du med å bruke programmering i matematikkundervisning?
        i. Brukes det da som metode eller som et mål i seg selv?
    b. Har du fått noe opplæring?
        i. Hva?
    c. Hvordan har skolen/ kommunen lagt til rette for at dette kan brukes i undervisningen?
    d. Hva er dine tanker rundt matematikkundervisningen slik den er lagt opp til i LK20? Kjerneelementene og kompetansemålene.

2. Har du hørt om begrepet computational thinking, eller algoritmisk tenking?
    a. Hva legger du i begrepet computational thinking/algoritmisk tenking?
    b. Hvilken relevans har dette for matematikkfaget?
    c. Hva er dine tanker om bruken av dette i matematikkundervisningen? Finnes det noen kobling? Hva er, etter din mening, sammenhengen?

3. Jeg har nå observert en av dine undervisningsøkter her har dere arbeidet med programmering;
    a. Hva var formålet med undervisningen? Hvilke kompetansemål lå til grunn?
    b. Hvordan forberedte du deg til den observerte undervisningssekvensen?
    c. Hvor finner du inspirasjon og ideer? ((Kilhamn et al., 2021)
    d. Hvilke arbeidsmetoder tok du i bruk? Hvorfor?
    e. Hva er din (lærerens rolle) i arbeidet?
    f. Hva er elevenes rolle?
    g. Hvilket (matematikkfaglig) læringsutbytte bør elevene sitte igjen med etter denne timen?
    h. Hva gjør du for å skape denne sammenhengen?
    i. Hvilke forberedelser gjør du ved planlegging av undervisning der programmering er et av hovedkomponentene?
    j. Hvis du skal jobbe med brøk, sannsynlighet eller statistikk på mellomtrinnet, kunne du da brukt programmering som metode for dette arbeidet? Hvordan?
    k. Kan du gi eksempler på gode programmerings aktiviteter du har brukt? (Kilhamn et al., 2021)

4. Hvis du ser på disse (eget ark) kompetansemålene som er satt for 5., 6. og 7.trinn.
    a. På hvilken måte mener du at disse får innpass i matematikkfaget?
    b. Hvordan vil du planlegge undervisningen med disse målene i fokus?
    c. Hvilke hjelpemidler tar du i bruk?
    d. Er det noen matematiske konsepter som bør knyttes til arbeidet med å nå disse målene?
    e. Er det noen konsepter ved programmering som er viktige/ relevante å trekke frem i matematikkfaget? (Kilhamn et al., 2021)
    f. Finnes det andre kompetansemål i matematikkfaget som en kan oppnå gjennom arbeid med programmering? Hvordan ?

5. Til hvilken grad, tenker du, er elevene bevisste på koblingen mellom programmering og matematikk?
    a. Til hvilken grad tenker du at elevene mestrer å kunne ta i bruk programmering som metode for å løse matematiske problemstillinger etter 7.trinn? Nå? Hva med i fremtiden?
    b. Hva tenker du må ligge til grunn for at programmering skal kunne fungere som metode for å arbeide med matematiske konsepter?

9.3 Informasjonsskriv observasjoner

Vil du delta i forskningsprosjektet
Programmering i matematikkundervisningen

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å finne ut at hvordan lærere implementerer programmering i matematikkundervisningen og til hvilken grad de knytter det matematikkfaglige innholdet til arbeidet med programmering i faget. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål
Dette Prosjektet er en masteroppgave der formålet er å se på hvordan programmering implementeres i matematikkundervisningen. Problemstillingen som er utgangspunktet for prosjektet er:

- Til hvilken grad knytter lærere det matematiske læringsutbyttet til arbeidet med programmering i matematikkundervisningen på mellomtrinnet?

Prosjektet tar utgangspunkt i analyse av litteratur om programmering, observasjon av undervisning av programmering i matematikkundervisningen, samt intervju av lærere og elever om blant annet det matematiske utbyttet av undervisningen.

Hvem er ansvarlig for forskningsprosjektet?
Henrik Forssell, Førsteamanuensis ved Oslomet, er ansvarlig for prosjektet. Prosjektet gjennomføres av mastergradsstudent Mischa van der Linden.

Hvorfor får du spørsmål om å delta?
Prosjektet tar for seg matematikkundervisning på mellomtrinnet. I den forbindelse ønsker vi å observere en matematikktime der programmering brukes i en undervisningstime.

Hva innebærer det for deg å delta?
I dette prosjektet tar jeg blant annet i bruk observasjon som metode for innhenting av informasjon.

Ved å delta i prosjektet vil du observeres i en undervisningstime der du arbeider med programmering. Jeg vil ta observasjonsnotater av de observasjonene som er relevante for problemstillingen. Dette innebærer at personer som observeres vil anonymiseres, og hverken elever eller lærere skal kunne identifiseres/gjenkjennes gjennom notatene.

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Du når som helst trekke deg fra deltakelsen uten å oppgi noen grunn. Alle dine opplysninger vil da bli slettet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg. Dette vil heller ikke gå ut over elevens opplæring.

Du kan trekke deg ved å gi beskjed til din kontaktlærer, skolens ledelse eller sende prosjektleder eller student en mail. Se kontaktinformasjon under.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?

Opplysningene anonymiseres og lydopptak slettes når prosjektet avsluttes og oppgaven er godkjent, noe som etter planen vil være senest høsten 2023.

Hvis du har spørsmål til studien, eller ønsker å vite mer om eller benytte deg av dine rettigheter, ta kontakt med:

- Oslomet ved Henrik Forssell (veileder), Henrik.Forssell@oslomet.no
- Mischa van der Linden (student), s188417@oslomet.no.

Med vennlig hilsen

Henrik Forssell og Mischa van der Linden

(Forsker/veileder) (Student)

9.4 Samtykkeskjema elev

Vil du delta i forskningsprosjektet
Programmering i matematikkundervisningen

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å finne ut at hvordan lærere implementerer programmering i matematikkundervisningen og til hvilken grad de knytter det matematikkfaglige innholdet til arbeidet med programmering i faget. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål
Dette prosjektet er en masteroppgave der formålet er å se på hvordan programmering implementeres i matematikkundervisningen. Problemstillingen som er utgangspunktet for prosjektet er:

- Til hvilken grad knytter lærere det matematiske læringsutbyttet til arbeidet med programmering i matematikkundervisningen på mellomtrinnet?

Prosjektet tar utgangspunkt i analyse av litteratur om programmering, observasjon av undervisning av programmering i matematikkundervisningen, samt intervju av lærere og elever om blant annet det matematiske utbyttet av undervisningen.

Hvem er ansvarlig for forskningsprosjektet?
Henrik Forssell, Førsteamanuensis ved Oslomet, er ansvarlig for prosjektet. Prosjektet gjennomføres av mastergradsstudent Mischa van der Linden.

Hvorfor får du spørsmål om å delta?
Prosjektet tar for seg matematikkundervisning på mellomtrinnet. I den forbindelse søker vi elever på mellomtrinnet som vi har deltatt i en observert undervisningstime der programmering brukes i en undervisningstime. Vi søker gutter og jenter som kan snakke om erfaringer fra den observerte undervisningssekvensen. For å begrense søket rettes prosjektet mot skoler på Østlandet.

Hva innebærer det for deg å delta?

I dette prosjektet tar jeg blant annet i bruk intervju som metode for innhenting av informasjon. Dersom du velger å delta i prosjektet vil du intervjues i etterkant av en undervisningstime der det har blitt arbeidet med programmering. I den forbindelse vil spørsmålene blant annet handle om det dere har

arbeidet med i timen. Jeg vil ta lydopptak av intervjuene, og sitater fra disse kan brukes i masteroppgaven. Lydopptaket vil transkriberes og anonymiseres elektronisk.

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du når som helst trekke samtykket tilbake uten å oppgi noen grunn. Alle dine personopplysninger vil da bli slettet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg.

Ditt personvern – hvordan vi oppbevarer og bruker dine opplysninger

Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler opplysningene konfidensielt og i samsvar med personvernregelverket.

- Veileder for prosjektet og student og vil ha tilgang til opplysningene.
- Navnet og kontaktopplysningene dine vil erstattes med en kode som lagres på egen navneliste adskilt fra øvrige data. Datamaterialet vil lagres i en kryptert mappe.
- Deltakerne vil i liten grad kunne gjenkjennes i publikasjonen. Det vil publiseres observasjoner av hendelser/ samtaler samt til hvilken grad deltakerne har fått opplæring.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?

Opplysningene anonymiseres og lydopptak slettes når prosjektet avsluttes og oppgaven er godkjent, noe som etter planen vil være senest høsten 2023.

Hva gir oss rett til å behandle personopplysninger om deg? Vi behandler opplysninger om deg basert på ditt samtykke.

På oppdrag fra Oslomet har NSD – Norsk senter for forskningsdata AS vurdert at behandlingen av personopplysninger i dette prosjektet er i samsvar med personvernregelverket.

Dine rettigheter

Så lenge du kan identifiseres i datamaterialet, har du rett til:

- innsyn i hvilke opplysninger vi behandler om deg, og å få utlevert en kopi av opplysningene
- å få rettet opplysninger om deg som er feil eller misvisende
- å få slettet personopplysninger om deg
- å sende klage til Datatilsynet om behandlingen av dine personopplysninger

Hvis du har spørsmål til studien, eller ønsker å vite mer om eller benytte deg av dine rettigheter, ta kontakt med:

- Oslomet ved Henrik Forssell (veileder), Henrik.Forssell@oslomet.no og Mischa van der Linden (student), s188417@oslomet.no.
- Vårt personvernombud: Ingrid S. Jacobsen, tlf. 67235534, mail; personvernombud@oslomet.no, ved Oslomet.

Hvis du har spørsmål knyttet til NSD sin vurdering av prosjektet, kan du ta kontakt med:

- NSD – Norsk senter for forskningsdata AS på epost (personverntjenester@nsd.no) eller på telefon: 55 58 21 17.

Med vennlig hilsen

Henrik Forssell og Mischa van der Linden

(Forsker/veileder) (Student)

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet Programmering i matematikkundervisningen og har fått anledning til å stille spørsmål. Jeg samtykker til:

- □ å delta i intervju.

Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet

---

(Signert av prosjektdeltaker, dato)

9.5 Intervjuguide Elev

Intervjuguide Masteroppgave
Intervju av elever

Dette er et arbeidsverktøy for oppgaven. Intervjuguiden er ledende for et semistrukturert intervju. Planen er at intervjuet skal foregå som en samtale mellom informanten og den som intervjuer. Ved behov vil relevante oppfølgingsspørsmål legges til.

1. Nå hadde dere en matematikktime. Hva var målet med denne timen? Hva lærte dere?
2. Hva tenker du om oppgaven(-e) dere fikk? Likte du dem? Hvorfor/hvorfor ikke?
3. Måtte du bruke noe matematikk for å løse oppgaven (-e)? I så fall hva?
4. Har dere jobbet med programmering tidligere? Liker du å jobbe med programmering? Hvorfor/hvorfor ikke?
5. Hva er en algoritme? Er dette noe dere har brukt i dag?
6. Hva tenker du er viktig å tenke på når du jobber med programmering/ å lage koder?
7. Hvordan liker du best å jobbe med programmerings oppgaver? Alene? Sammen med andre?
8. Hvilke triks har du for å klare vanskelige oppgaver? hvordan tenker du? Hvilke lure måter å tenke på har du for å løse en vanskelig oppgave?
9. Hvordan pleier læreren deres å hjelpe dere med oppgavene? Hva gjør hen?
10. Hvorfor tror du at dere jobbet med programmering i en matematikktime?
11. Hva kan du bruke kunnskap om programmering til?

9.6 Observasjonsskjema

Observasjonsskjema Programmering i matematikk

Rammer for Undervisningen		Notat
Dato:		
Skole:		
Tid:		
Trinn:		
Antall elever:		
Antall lærere:		
Plassering av elever:		
Plassering av observatør:		

Tid	Hva	Lærer	Elev
	Oppstart		
	Hoveddel		
	Avslutning		

NSD NORSK SENTER FOR FORSKNINGSDATA

Meldeskjema

Referansenummer

420981

Hvilke personopplysninger skal du behandle?

- Navn (også ved signatur/samtykke)
- E-postadresse, IP-adresse eller annen nettidentifikator
- Lydopptak av personer

Prosjektinformasjon

Prosjekttittel

Programmering i matematikkundervisningen

Prosjektbeskrivelse

Prosjektet er en masteroppgave knyttet til studiet Master i skolerettet utdanningsvitenskap med fordypning i matematikk ved Oslomet. Fokuset for oppgaven er programmering i matematikkundervisningen. Innhenting av data og informasjon vil skje ved kvalitative undersøkelser som observasjon av undervisning og intervju av lærere og elever.

Begrunn behovet for å behandle personopplysningene

Det ønskes lydopptak av intervjuer. Dette for å bedre fange opp det som blir sagt av informanter, samt å mer presist kunne gjengi informasjonen disse gir.
E-post adresser behøves for å komme i kontakt med informantene. Disse vil ikke brukes i selve oppgaven.
Grunnet pandemien og ønske om minst mulig nærkontakter, er det behov for å kunne gjennomføre intervjuer over digital plattform som zoom i tillegg til muligheten for lydopptak med Nettskjema diktafon.

Ekstern finansiering

Type prosjekt

Studentprosjekt, masterstudium

Kontaktinformasjon, student

Mischa van der Linden, s188417@oslomet.no, tlf: 45066293

Behandlingsansvar

Behandlingsansvarlig institusjon

OsloMet – storbyuniversitetet / Fakultet for lærerutdanning og internasjonale studier / Institutt for grunnskole- og faglærerutdanning

Prosjektansvarlig (vitenskapelig ansatt/veileder eller stipendiat)

Henrik Forssell , Henrik.Forssell@oslomet.no, tlf: +4797001598

Skal behandlingsansvaret deles med andre institusjoner (felles behandlingsansvarlige)?

Nei

Utvalg 1

Beskriv utvalget

Lærere som underviser i matematikk på mellomtrinnet. Utvalget skal bestå av lærere med og uten formell opplæring i bruken av programmering.

Rekruttering eller trekking av utvalget

For å rekruttere informanter til prosjektet, sender jeg ut en mail til lærere som tar etterutdanning i programmering ved Oslomet. I tillegg plukker jeg tilfeldige skoler fra oversikten til Kidsakoder.no over skoler som har deltatt i kodetimen tidligere år og sender disse en mail med forespørsel om deltakelse.

Alder

23 - 67

Inngår det voksne (18 år +) i utvalget som ikke kan samtykke selv?

Nei

Personopplysninger for utvalg 1

- Navn (også ved signatur/samtykke)
- E-postadresse, IP-adresse eller annen nettidentifikator
- Lydopptak av personer

Hvordan samler du inn data fra utvalg 1?

Personlig intervju

Grunnlag for å behandle alminnelige kategorier av personopplysninger

Samtykke (art. 6 nr. 1 bokstav a)

Ikke-deltakende observasjon

Grunnlag for å behandle alminnelige kategorier av personopplysninger

Samtykke (art. 6 nr. 1 bokstav a)

Informasjon for utvalg 1

Informerer du utvalget om behandlingen av opplysningene? Ja

Hvordan?

Skriftlig informasjon (papir eller elektronisk)

Utvalg 2

Beskriv utvalget

Elever som har deltatt i observert undervisning med programmering i matematikkundervisning. Gutter og jenter.

Rekruttering eller trekking av utvalget

Læreren velger ut to til tre elever der foresatte har samtykket til intervju. Elevene skal ha deltatt i en observert undervisningstime der det arbeides med programmering i en matematikktime.
Under observasjonene skal det ikke registrerer personopplysninger, foreldrene skal likevel informeres og få mulighet til å reservere barnet fra deltakelse uten at dette skal gå ut over "normal" undervisning. Med mindre skolen har innvendinger mot dette. I et slikt tilfelle kan jeg ikke registrere observasjoner som involverer denne eleven. Informasjonen til foresatte vil gå ut via skolen.

Alder

10 - 13

Inngår det voksne (18 år +) i utvalget som ikke kan samtykke selv?

Nei

Personopplysninger for utvalg 2

- Navn (også ved signatur/samtykke) Lydopptak av personer

Hvordan samler du inn data fra utvalg 2?

Personlig intervju

Grunnlag for å behandle alminnelige kategorier av personopplysninger

Samtykke (art. 6 nr. 1 bokstav a)

Hvem samtykker for barn under 16 år?

Foreldre/foresatte

Ikke-deltakende observasjon

Grunnlag for å behandle alminnelige kategorier av personopplysninger

Samtykke (art. 6 nr. 1 bokstav a)

Hvem samtykker for barn under 16 år?

Foreldre/foresatte

Informasjon for utvalg 2

Informerer du utvalget om behandlingen av opplysningene? Ja

Hvordan?

Skriftlig informasjon (papir eller elektronisk)

Tredjepersoner

Skal du behandle personopplysninger om tredjepersoner?

Nei

Dokumentas jon

Hvordan dokumenteres samtykkene?

- Manuelt (papir)
- Elektronisk (e-post, e-skjema, digital signatur)

Hvordan kan samtykket trekkes tilbake?

For lærere kan dette gjøres ved å muntlig gi student eller veileder beskjed om at samtykket trekkes tilbake, eller ved å skriftlig sende en elektronisk beskjed på e-post. For elevene kan dette gis muntlig under intervjuene til studenten, muntlig til læreren eller skriftlig elektronisk på e-post til Lærer, student eller veileder.

Hvordan kan de registrerte få innsyn, rettet eller slettet opplysninger om seg selv?

De registrerte kan få innsyn ved å be om å få lese oppgaven før den leveres. De kan også be om å få se transkripsjoner av lydopptak.

Totalt antall registrerte i prosjektet

1-99

Tillatelser

Skal du innhente følgende godkjenninger eller tillatelser for prosjektet?

Behandli

Hvor behandles

- Mobile enheter tilhørende
- Ekstern tjeneste eller nettverk
- Maskinvare tilhørende

Hvem behandler/har tilgang til

- Prosjektansv
- Student
- Databehan

Hvilken databehandler har tilgang til

Nettskjema, OneDrive, Zoom

Tilgjengeliggjøres opplysningene utenfor EU/EØS til en tredjestat eller internasjonal organisasjon?

Nei

Sikkerhet

Oppbevares personopplysningene atskilt fra øvrige data (koblingsnøkkel)?

Ja

Hvilke tekniske og fysiske tiltak sikrer personopplysningene?

- Opplysningene anonymiseres fortløpende
- Opplysningene krypteres under lagring
- Adgangsbegrensning

Varighet

Prosjektperiode

16.08.2021 - 31.10.2023

Skal data med personopplysninger oppbevares utover prosjektperioden?

Nei, data vil bli oppbevart uten personopplysninger (anonymisering)

Hvilke anonymiseringstiltak vil bli foretatt?

- Personidentifiserbare opplysninger fjernes, omskrives eller grovkategoriseres Lyd- eller bildeopptak slettes

Vil de registrerte kunne identifiseres (direkte eller indirekte) i oppgave/avhandling/øvrige publikasjoner fra prosjektet?

Nei

Tilleggsopplysninger