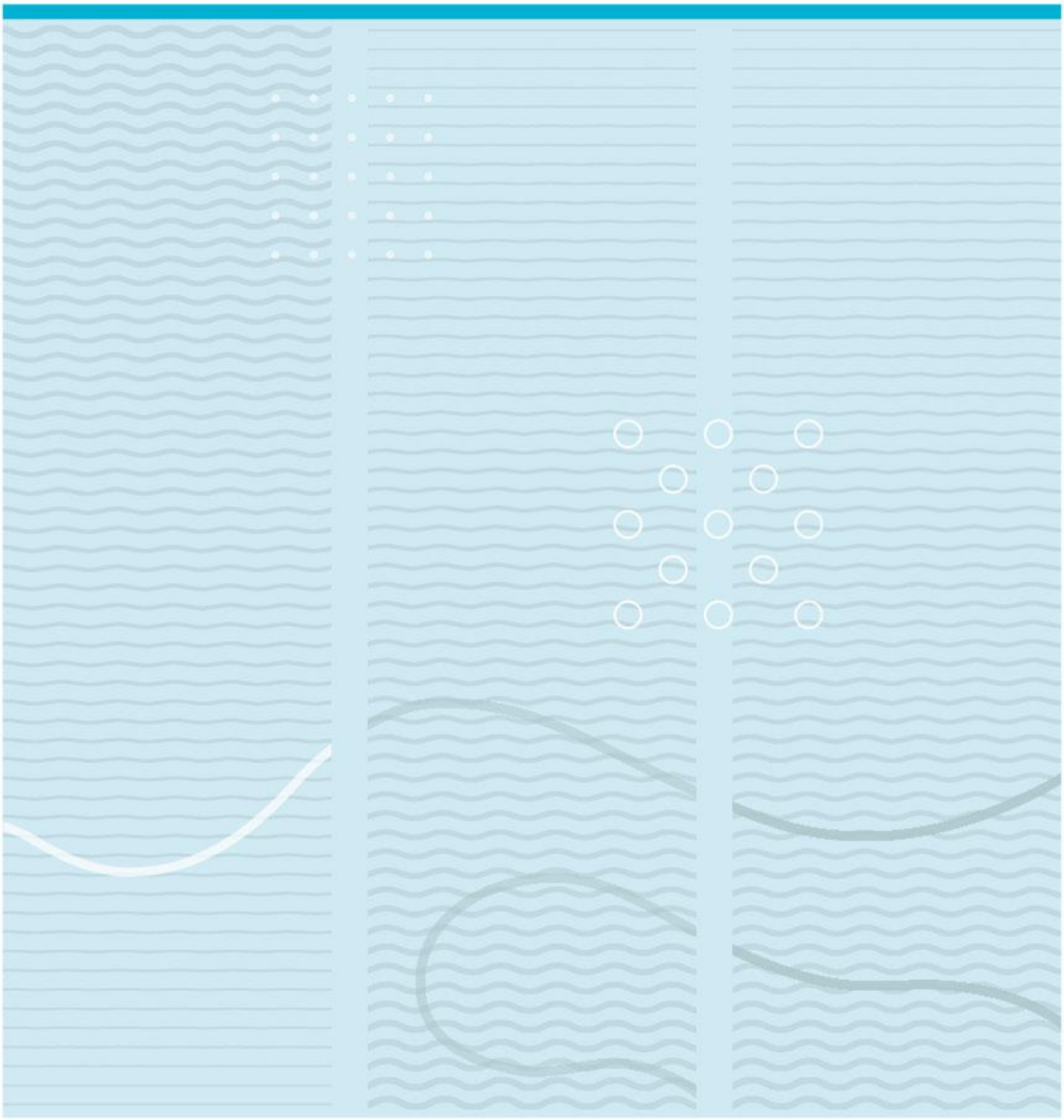
Universitetet i Sørøst-Norge
Fakultet for humaniora, idretts- og utdanningsvitenskap
–
Mastergradsavhandling
Studieprogram: MGLU1
Vår 2022

Siri Dyrhovd Leirvik

Programmering i begynneropplæringen

En kvalitativ casestudie av kjennetegn ved programmeringsundervisning



Universitetet i Sørøst-Norge
Fakultet for humaniora, idretts- og utdanningsvitenskap
Institutt for pedagogikk
Postboks 235
3603 Kongsberg

http://www.usn.no

Denne avhandlingen representerer 45 studiepoeng

Sammendrag

Vi er i en tid med økt digitalisering av samfunnet og flere lærere viser sin bekymring rundt deres egne kunnskaper og ferdigheter innenfor digital kompetanse og programmering. Programmering og algoritmisk tenkning har kommet inn i den norske læreplanen og selv om programmering ikke er et nytt fenomen, er det relativt nytt i den norske skolen. Det fører til at undringer, bekymringer og debatter oppstår. Dette masterprosjektet har som hensikt å bidra til forskningsfeltet med kjennetegn på programmeringsundervisning i begynneropplæringen.

Problemstillingen er formulert slik: Hva kjennetegner undervisning i programmering i begynneropplæringen?

Masteroppgaven består av en casestudie hvor det har vært gjennomført åtte observasjonsøkter med tilhørende intervjuer, i tillegg til ett selvstendig intervju om programmeringsundervisning i begynneropplæringen generelt sett. Studien har kun hatt én informant: en lærer med faglig kunnskap og lang praktisk erfaring med programmering i begynneropplæringen. Hovedfunnene i prosjektet viser at programmeringsundervisningen i begynneropplæringen har fokus på analog programmering, samarbeid og lærerrollen som veileder. I tillegg trekkes analoge og digitale programmeringsverktøy frem som svært gjeldene i programmering og som et hjelpemiddel i fag. Gjennom den analoge programmeringen i samarbeid med medelever og ved hjelp av kjente og hverdagslige situasjoner og verktøy, er hensikten at elevene skal lære seg «den grunnleggende forståelsen» for programmering. Det innebærer en forståelse for hva som skjer når vi programmerer. Algoritmisk tenkning trekkes frem som et sentralt aspekt i programmeringen i begynneropplæringen. Det innebærer ulike konsepter og tilnærminger som kan benyttes ved problemløsning. Lærerrollen som veileder er gjeldene i programmering, da læreren bør gi elevene rom til å benytte problemløsningsstrategier for å løse programmeringsoppgaver. Rammer og «verktøy» er hensiktsmessig, men også å gi elevene mulighet til å utforske og undersøke selv, basert på kunnskaper de har tilegnet seg i undervisningen og i samspill med medelever. Oppsummert kan det sies at programmeringsundervisning i begynneropplæringen kjennetegnes ved samarbeid, lærerrollen som veileder og bruk av analoge og digitale programmeringsverktøy gjennom kjente og hverdagslige situasjoner, som elevene har et nært forhold til.

Innholdsfortegnelse

Forord

Etter fem år på grunnskolelærerutdanningen og ett år med masterskriving, kjennes det litt surrealistisk å skrive dette forordet. Dette siste året på universitetet har vært intenst, spennende, frustrerende og lærerikt, men nå er jeg endelig klar for læreryrket! Jeg er takknemlig for muligheten jeg har fått til å fordype meg i temaet og tillære meg kunnskaper som er svært relevante for min lærerpraksis og samfunnet vårt. Interessen for programmering har blitt sterkere gjennom arbeidet med prosjektet og det blir spennende å anvende det som nyutdannet grunnskolelærer.

Jeg ønsker å vise min takknemlighet til de fantastiske foreleserne ved Universitetet i Sørøst-Norge og alle medstudentene mine. Spesielt ønsker jeg å takke veilederen min Siv Svendsen for all støtte og veiledning! Jeg er svært takknemlig for alt du har bidratt med, for å gjøre veien litt enklere, i løpet av en til tider svært krevende prosess. En stor takk til veiledergruppen for gode diskusjoner, tilbakemeldinger og tips. Jeg vil også takke studievenninnen min Kristina for gode samtaler, verdifulle innspill og hjelp!

Til slutt, en stor takk til familien min som har stått ved meg, heiet på meg og bidratt med korrektur og tips! Takk for all omtanke, støtte og oppmuntring. Det setter jeg stor pris på!

Drammen, juni 2022
Siri Dyrhovd Leirvik

1 Innledning

1.1 Bakgrunn for valg av tema

Da jeg startet på grunnskolelærerutdanningen i 2017, hadde jeg et stort ønske om å hjelpe barn med å oppnå mestring og motivasjon for læring, gjennom varierte arbeidsmetoder og tilpasset opplæring. Ved å velge begynneropplæring som masterfag, økte min interesse ytterligere for bruk av digitale verktøy som arbeidsmetode blant de yngste elevene på skolen. Til å begynne med, tenkte jeg at programmering bare var en arbeidsmetode, som kun innebar digitale verktøy, som ulike apper og roboter. Etter hvert erfarte jeg at programmering handlet om så mye mer. Ettersom nysgjerrigheten rundt bruk av programmering i begynneropplæringen økte, ønsket jeg å undersøke nærmere hvordan det tas i bruk i undervisningen.

Flere lærere har fått kjennskap til programmering som et fenomen og ønsker å ta det i bruk i undervisningen i begynneropplæringen. Dette ønsker også jeg som kommende lærer å lære mer om, da min kunnskap om dette temaet ikke er tilfredsstillende, selv etter flere år på grunnskolelærerutdanningen. Gjennom samtale med ulike lærere på forskjellige skoler, fant jeg ut at samtlige lærere kjenner på manglende kunnskaper og erfaring med hvordan programmering kan tas i bruk i praksis. Etter mer kjennskap til viktigheten av programmering og digitale ferdigheter i skolen, lærerens bekymringer og erkjennelsen av manglende kunnskap om feltet gjennom min egen utdanning, bestemte jeg meg for å undersøke dette videre.

1.2 Programmeringens historie

Programmering i skolen er ikke en ny idé, selv om det kanskje kan føles slik i forbindelse med økt fokus på programmering gjennom fagfornyelsen. Seymour Papert utviklet allerede på 1960-tallet programmeringsspråket LOGO, med tanke på at det skulle tas i bruk i utdanningen (Sevik et al., 2016, s. 8). Hensikten med å lære programmering var å øke elevenes evne til problemløsning. Sevik et al. (2016) forteller om gjennomføring av ulike forsøk med programmering i skolen på 1980-tallet, men at det ikke hadde like stor innvirkning som forventet. I dag er situasjonen annerledes og tilgjengeligheten for programmeringsverktøy og -læring er lettere (Sevik et al., 2016, s. 8), samt at programmering er en del av den nye norske læreplanen fra 2020.

1.2.1 Programmering i skolen

Dagens samfunn er preget av en økende digitalisering (Mannila, 2020, s. 11), noe som gir behov for kunnskap og forståelse om digital kompetanse (Hovda & Liukas, 2020, s. 5). Ettersom samfunnet blir påvirket av digital teknologi i større grad, er det viktig for den demokratiske utviklingen at befolkningen stiller kritiske spørsmål og påvirker ytterligere utviklingen (Hovda & Liukas, 2020, s. 5).

Regjeringen la i 2017 frem sin digitaliseringstrategi for grunnopplæringen 2017-2021 – *Framtid, fornyelse og digitalisering* (2017a). Der legges det frem at digitale ferdigheter blant annet innebærer å kunne bruke digitale medier, verktøy og ressurser på en hensiktsmessig og forsvarlig måte, for å kunne løse praktiske oppgaver, kommunisere og skape digitale produkter (Kunnskapsdepartementet, 2017a, s. 18). Det skal vurderes i fagfornyelsen hvordan algoritmisk tenkemåte, teknologi og programmering skal inngå i bestemte læreplaner for fag, deriblant naturfag og matematikk (Kunnskapsdepartementet, 2017a, s. 18). Videre trekkes det frem at norske lærere selv rapporterer om manglende tilrettelegging for utvikling av egen IKT-kompetanse og at det heller ikke blir ivaretatt i lærerens grunnutdanning. Likevel poengteres det at lærerkompetanse er en forutsetning for at IKT-tiltak på læring i fag skal ha god effekt (Kunnskapsdepartementet, 2017a, s. 9). *Rammeverk for lærerens profesjonsfaglige digitale kompetanse (PfDK),* tar for seg hva lærerens profesjonsfaglige kompetanse innebærer og hvordan kvaliteten på undervisningen kan øke (Kelentric, Helland & Arstorp, 2017). Det poengteres blant annet, at en sentral del i denne sammenhengen er lærerens rolle. For at lærere skal være i stand til å utvikle slik kompetanse hos elevene, må lærerne utvikle sin egen profesjonsfaglige digitale kompetanse (Kelentric, Helland & Arstorp, 2017, s. 4).

I forbindelse med fagfornyelsen og regjeringens digitaliseringstrategi, oppsto det en debatt om programmering i skolen. Bekymringen rundt undervisning i programmering av lærere med lite kunnskap innenfor feltet, er et av aspektene som er sentralt i debatten (Solberg, 2018; Steenbuch, 2016; Østerås & Andersen, 2020; Johansen, 2020; Moreau, 2021; Sommerfeldt, 2018). Forskjellen på det *å «lære seg programmering og det å lære seg å gi god undervisning i programmering*» (Østerås & Andersen, 2020), er et aspekt som trekkes frem i debatten (Østerås & Andersen, 2020). I tillegg debatteres det om programmering bør bli et eget fag, eller om det er nok å trekke det inn i eksisterende fag i skolen, som i matematikkfaget (Solberg, 2018; Steenbuch, 2016; Østerås & Andersen,

2020). I en debatt arrangert av Lær Kidsa Koding, settes søkelyset på ulike problemstillinger som bremser regjeringens ambisjon om innføringen av undervisning i programmering i grunnskolen (Sommerfeldt, 2018). Grunnet læreres manglede kunnskap, og for enkelte, interesse for programmering, ytres det bekymring for at programmering kan bli ekskludert fra undervisningen, selv om programmering og algoritmisk tenkning er inkludert i matematikkfaget (Sommerfeldt, 2018). I tillegg trekkes redselen for at håndskriften skal vike for skriving på digitale enheter frem i debatten, og at elevene tilbringer mye tid både på skolen og hjemme foran en skjerm. Det argumenteres for at håndskriften ikke vil forsvinne fra grunnskoleopplæringen, men at det foretrekkes en kombinasjon med skriving for hånd og på digitale enheter (Sommerfeldt, 2018). I debatten fremheves forskjellen mellom å bruke digitale verktøy til et læringsformål og til å se på ulike videoer på YouTube. Videre nevnes skoler med manglende ressurser og relevant utstyr til programmering (Sommerfeldt, 2018). Til slutt konkluderes det med bakgrunn i kravene til samfunnet, som er livslang læring, hjelper det ikke nødvendigvis å sende lærere på kurs og anta at problemet med manglende kunnskap er løst. Det er viktig å opprettholde søkelyset på programmering i skolen og sørge for at både lærere og elever tillærer seg kunnskaper og ferdigheter, som gjør de rustet til å være en del av samfunnets digitalisering (Sommerfeldt, 2018).

1.2.2 Læreplanarbeid

I overordnet del – verdier og prinsipper for grunnopplæringen, under grunnleggende ferdigheter, trekkes det frem at ferdighetene; lesing, regning, skriving, muntlige ferdigheter og digitale ferdigheter, er viktige for elevenes utvikling og deltakelse i utdanning, arbeid og samfunnsliv. I tillegg til at de er en del av den faglige kompetansen og redskaper som er nødvendige for læring og faglig forståelse (Kunnskapsdepartementet, 2017b, s. 11). Utviklingen av disse ferdighetene har betydning gjennom hele opplæringsløpet, og starter fra den første lese- og skriveopplæringen (Kunnskapsdepartementet, 2017b, s. 11) i begynneropplæringen.

I den nye læreplanen (2020), trekkes det frem seks kjerneelementer innenfor matematikkfaget (Kunnskapsdepartementet, 2019). Under det første kjerneelementet, *utforsking og problemløysing*, finner vi begrepet *algoritmisk tenkning*, som forklares som viktig i prosessen med å utvikle fremgangsmåter og strategier for å løse ulike problemer. I tillegg innebærer det å bryte ned et problem i delproblemer, som videre kan løses systematisk, samt vurdere om delproblemet kan løses best med eller uten

digitale verktøy (Kunnskapsdepartementet, 2019). I et notat fra Senter for IKT i utdanningen, forklares algoritmisk tenkning som «*en problemløsningsprosess som innebærer å tenke som en informatiker når man skal løse et problem*» (Sevik et al., 2016, s. 13). I tillegg står det at ens teknologiske kompetanse bør tas i bruk til å få en datamaskin til å løse problemet (Sevik et al., 2016, s .14). Det å undervise i programmering handler ikke bare om at elever skal lære seg å forstå datamaskiner eller få bedre kompetanse i bruk av digitale verktøy. Det handler også om å gi elevene gode verktøy for å kunne løse problemer og ulike oppgaver. Algoritmisk tenkning kan også benyttes som metode i ulike fag og sammenhenger (Sevik et al., 2016, s. 13).

I og med at kompetansemålene for småtrinnet ikke nevner datamaskiner eller programmering spesifikt, mener Lær Kidsa Koding (u. å.) at flere mål kan ses på som *programmeringsmål*. I forbindelse med programmering i kompetansemålene for småtrinnet, er algoritmisk tenkning hovedfokuset. Lær Kidsa Koding (u. å.) understreker at mye av programmeringen på småtrinnet kan løses ved hjelp av aktiviteter og tradisjonelle leker med fokus på algoritmisk tenkning. I tillegg er det en rekke programmeringsbegreper som enkelt kan trekkes inn i undervisningen i sammenheng med aktivitetene og lekene, i ulike kontekster. Fagene som trekkes frem i tråd med programmering og kompetansemålene er i hovedsak matematikk, men også musikk i forbindelse med algoritmisk tenkning og bruk av digitale verktøy (Lær Kidsa Koding, u. å.).

1.3 Problemstilling

En del av forskningen om programmering i undervisningen i skolen, er tilknyttet andre deler av verden. Dermed er det behov for forskning på programmering i norske skoler, spesielt i forbindelse med den nye læreplanen fra 2020. Basert på lesing av tidligere forskning, fagfornyelsen og med læreres kommentarer i minnet, endte jeg opp med følgende problemstilling:

Problemstilling:

Hva kjennetegner undervisning i programmering i begynneropplæringen?

Forskningsspørsmål:

1. Hva kjennetegner programmeringsundervisning som fremmer aspekter innenfor fagfornyelsen og samfunnets digitalisering?

2. Hva mener læreren er viktig i planleggingen, gjennomføring og vurdering av programmeringsundervisningen på 1. til 4. trinn?

Ved å stille disse spørsmålene, kan jeg undersøke hvordan programmering er en del av undervisningen og hva læreren anser som viktig i programmeringsundervisningen i begynneropplæringen. I tillegg vil jeg få innsikt i hvordan læreren har tillært seg kunnskaper og ferdigheter i programmering, og anvender dette i undervisningen, både før og etter fagfornyelsen.

1.4 Begrepsavklaring

I dag er ordet «koding» mer utbredt i skolen når det er snakk om at elevene skal lære seg å programmere. Det gjenspeiles også i bevegelser som Kodeklubben i Norge og Lær Kidsa Koding. Likevel har tradisjonelt sett ordet programmering vært det mest foretrukne begrepet, for å lage instruksjoner til digitale enheter om å utføre en oppgave. Det kalles også *å skrive programkoder*. I tillegg innebærer ordet programmering prosessen for å komme fram til denne programkoden, altså problemløsning, å identifisere et problem, finne mulige løsninger på problemet, skrive en kode som kan forstås av en datamaskin, feilsøke og hele tiden forbedre koden (Sevik et al., 2016, s. 9). I og med at denne studien omfatter mer enn det å skrive programkoder, vil begrepet programmering benyttes i denne masteroppgaven.

Denne studien er en master i begynneropplæring, og det er dermed hensiktsmessig å definere hva som legges i begrepet og hvordan begynneropplæring legger føringer for prosjektet. Begynneropplæring er et begrep som sjeldent blir definert i faglitteratur og styringsdokumenter (Hoff-Jenssen et al., 2020, s. 143). Likevel kan det sies at begynneropplæringen omhandler overgangen fra barnehage til skole og de fire første årene på skolen (Palm et al., 2020, s. 13). I min studie er de fire første skoleårene sentrale, og det tas opp både faglige og sosiale aspekter som er grunnleggende innenfor begynneropplæringen (Hoff-Jenssen et al., 2020, s. 143) gjennom metodene som tas i bruk. I kapittel 2.6.1 Begynneropplæring, vil begrepet trekkes frem ytterligere i forbindelse med sentrale rammeverk for programmering i barneskolen.

2 Teori

2.1 Tidligere forskning

2.1.1 Analog programmering

Forskere trekker frem analog programmering i tilknytning til å lære algoritmisk tenkning (Aranda & Ferguson, 2018; Nouri et al., 2019; Berg, 2021). Et aspekt fra algoritmisk tenkning som samtlige forskere nevner er feilsøking, som anses som sentralt i analog programmering (Nouri et al., 2019; Chibas et al., 2018; Heikkilä & Mannila, 2018). Det poengteres at elevene lærer mye av å gjøre feil og rette de opp igjen (Nouri et al., 2019). Nouri et al. (2019) fremhever viktigheten av å forstå hva programmering er før digital programmering innføres (Nouri et al., 2019). Flere forskere viser til sammenhengen mellom analog og digital programmering, der en variasjon mellom dem er å foretrekke (Berg, 2021; Chibas et al., 2018; Heikkilä & Mannila, 2018; Otterborn et al., 2020). Berg (2021) trekker frem variasjonen mellom analog og digital programmering som betydningsfull for elevenes motivasjon, samt at det er en fin måte å tilpasse til hver enkelt elev (Berg, 2021). Chibas et al. (2018) understreker hvordan en kombinasjon gjør at elevene får ulike tilnærminger til å arbeide med kunnskapen de tilegner seg. Likevel poengteres det at gjennom analog programmering, får læreren et bedre overblikk over hva elevene synes er vanskelig og kan være en utfordring når de går over til digital programmering (Chibas et al., 2018). Ifølge Berg (2021) kan motivasjonen for programmeringen ses i sammenheng med at de analoge programmeringsaktivitetene er hverdagslige og gjør at overgangen til digital programmering forenkles (Berg, 2021).

Nouri et al. (2019) nevner at elevene lærer og utvikler forståelse for ulike programmeringsbegreper gjennom analog programmering. De er blant annet sekvens, gjentakelser (loop), trinnvise instruksjoner og betingelser (Nouri et al., 2019). Spesielt trinnvise instruksjoner angis i forskning i forbindelse med ulike aktiviteter innenfor analog programmering (Heikkilä & Mannila, 2018; Berg, 2021; Chibas et al., 2018; Nouri et al., 2019). Heikkilä & Mannila (2018) forteller blant annet om en mer hverdagslig og gjenkjennbar analog programmeringsaktivitet, som innebærer trinnvise instruksjoner ved påkledning (Heikkilä & Mannila, 2018). Berg (2021) viser til en oppgave hvor elevene skulle bygge et tårn av centicuber ved hjelp av trinnvise instruksjoner (Berg, 2021). Heikkilä & Mannila (2018) løfter frem hvordan

programmering blant de yngste elevene kan ses på som lekbasert læring, hvor de kan ta i bruk fantasien sin samtidig som problemløsningsferdigheter utvikles (Heikkilä & Mannila, 2018). I tillegg presiserer Nouri et al. (2019) at programmering gir elevene mulighet til å engasjere seg i kreativ problemløsning på en leken måte (Nouri et al., 2019). Aranda & Ferguson (2018) anbefaler analog programmering som en metode i rollelek. Videre fremheves boka «Hello Ruby», eller «Hei Ruby» på norsk, som gode hjelpemidler for unge elever til å lære programmering på en leken og fantasifull måte (Aranda & Ferguson, 2018).

Det gjøres et poeng av analog programmering som hensiktsmessig å begynne med, da denne formen for programmering tar for seg den grunnleggende forståelsen for hvorfor vi programmerer og hvordan programmering foregår. Det påpekes som en fordel, ved start av digital programmering, å inneha kunnskaper om analog programmering (Nouri et al., 2019). Likevel er en kombinasjon av analog og digital programmering å foretrekke (Berg, 2021; Chibas et al., 2018; Heikkilä & Mannila, 2018; Otterborn et al., 2020). I tillegg trekkes bruk av algoritmisk tenkning frem som et sentralt aspekt innenfor den analoge programmeringen (Aranda & Ferguson, 2018; Nouri et al., 2019; Berg, 2021). I det følgende fremlegges synspunkter fra forskning om bruk av programmeringsverktøy i analog og digital programmering.

2.1.2 Programmeringsverktøy

Ulike programmeringsverktøy, både for analog og digital programmering, vises til i forskning. I forbindelse med analog programmering, er bruken av artefakter fremtredende i ulike forskningsartikler (Heikkilä & Mannila, 2018; Chibas et al., 2018; Aranda & Ferguson, 2018; Berg, 2021). Det kan være konkrete artefakter som LEGO-klosser, centikuber, lekefigurer og kort med bilder, eller å skrive eller tegne (Heikkilä & Mannila, 2018; Chibas et al., 2018; Aranda & Ferguson, 2018; Berg, 2021). Heikkilä & Mannila (2018) forteller hvordan programmering ofte kan ses på som en abstrakt aktivitet, mens mer fysiske artefakter som for eksempel roboter gjør at aktiviteten blir mer konkret. Artefaktene i programmering brukes til å illustrere eller støtte læringen og/eller undervisningen (Heikkilä & Mannila, 2018). Aranda & Ferguson (2018) viser til hvordan elevene i deres studie tok i bruk både fysiske og virtuelle artefakter, i tillegg til sine egne hender og kropper for å gestikulere. På den måten følte de læringen ble mer konkret og forståelig (Aranda & Ferguson, 2018).

I tillegg til at fysiske artefakter benyttes analogt i programmeringen, forteller forskere om at artefakter også blir kombinert med digitale artefakter som roboter og ulike apper på iPad (Otterborn et al., 2020; Palmér, 2017). Programmering med blant annet Blue-Bot, ble ifølge Otterborn et al. (2020) ofte kombinert med andre objekter. Objektene kan videre benyttes eller lages om til hindre for roboten. Dette kalles for fysisk utvidet digital programmering. I tillegg kan roboter som Blue-Bot både programmeres direkte på roboten og via Blue-Bot-appen (Otterborn et al., 2020). Palmér (2017) poengterer at bruk av visuell programmering med yngre elever er hensiktsmessig, da det fører til økt fokus på logikken og strukturen istedenfor læringen av programmeringsspråk. Visuell programmering innebærer programmering av roboter, som gjør feilsøking og undersøking mer håndterlig for elevene (Palmér, 2017). Ifølge Palmér (2017) kan programmerbare roboter som Bee-Bot ses på som «manipulativer» eller verktøy. «Manipulativer» er objekter som er laget for å representere abstrakte matematiske ideer (Palmér, 2017). Otterborn et al. (2020) tar også i bruk begrepet «manipulativ» i forbindelse med roboter som Blue-Bot, men spesifiserer det mer med begrepet «håndgripelig» foran (Otterborn et al., 2020).

Programmeringsverktøy kan ifølge forskning innebære både artefakter som LEGO, lekefigurer og kort (Heikkilä & Mannila, 2018; Chibas et al., 2018; Aranda & Ferguson, 2018; Berg, 2021), men også roboter (Otterborn et al., 2020; Palmér, 2017). Uavhengig av artefaktene, formidles det at bruk av fysiske visuelle programmeringsverktøy er hensiktsmessig å ta i bruk blant de yngste elevene. Dette begrunnes ved å ha positiv innvirkning på deres læring innenfor programmering (Heikkilä & Mannila, 2018). Videre vil forskningens blikk på bruk av samarbeid i programmeringsundervisningen formidles.

2.1.3 Samarbeid

Som flere forskere poengterer, er samarbeid eller programmering i par en sentral del av programmeringsundervisningen (Zhong et al, 2016; Otterborn et al, 2019; Kjällander et al, 2021; Iskrenovic-Momcilovic, 2019). Likevel diskuteres det ulike måter å lage elevpar på og hva som er mest hensiktsmessig for programmeringsundervisningen (Zhong et al., 2016; Iskrenovic-Momcilovic, 2019). Elever som undervises med *pair programming (PP)*, som innebærer at to personer arbeider ved siden av hverandre med én datamaskin, hevdes at gjør det bedre innenfor algoritmisk tenkning enn ved individuell programmering (Zhong et al., 2016). Det trekkes frem flere faktorer som

påvirker effektiviteten av PP. Noen av de er: *oppgavekompleksitet, partnerens ferdigheter og erfaringer, partnernes læringsstiler og partnernes personlighet og temperament*. I tillegg ses PP på som en god måte å forminske kjønnsskiller i programmeringsundervisning. Elevene sosialiserer med hverandre og *«få[r] en god venn i læring»* (Zhong et al., 2016). Jevnlig kommunikasjon er dermed en viktig del av PP (Zhong et al, 2016). Iskrenovic-Momcilovic (2019) poengterer at programmering i par gir bedre resultater for begynnere i forhold til individuell programmering. I tillegg trekkes det frem at programmering i par gir bedre motivasjon for å lære ny kunnskap, raskere hukommelse av nye konsepter og mer effektiv læring av programmeringsspråk (Iskrenovic-Momcilovic, 2019).

For å oppsummere, er samarbeid en stor del av programmeringsundervisningen (Zhong et al., 2016; Otterborn et al., 2019; Kjällander et al., 2021; Iskrenovic-Momcilovic, 2019). Å samarbeide med medelever kan blant annet gi bedre motivasjon og resultater (Iskrenovic-Momcilovic, 2019), samtidig som det er en viktig del av utviklingen av gode kommunikasjonsferdigheter (Zhong et al, 2016). I neste delkapittel vil forskeres vektlegging av læreres tanker om programmeringsundervisning i skolen synliggjøres.

2.1.4 Lærere om programmeringsundervisning

I og med problemstillingen og forskningsspørsmålene i denne studien trekker frem bruken av programmering fra et lærerperspektiv, er det hensiktsmessig å se på forskningens oppfatning av læreres tanker og meninger om og rundt bruken av programmering i undervisningen. I flere forskningsartikler, kommer læreres usikkerhet frem rundt programmering generelt og hvordan lærere skal undervise i programmering (Otterborn et al, 2019; Kjällander et al, 2021; Stigberg & Stigberg, 2020). Blant annet i forbindelse med å anvende programmering i undervisningen (Stigberg & Stigberg, 2020; Otterborn et al., 2020; Kjällander et al., 2021). Utgangspunktet tas i forbindelse med den svenske læreplanen, hvor programmering er et formål i matematikkundervisningen (Stigberg & Stigberg, 2020; Skolverket, 2019). Grunnen til at den svenske læreplanen nevnes, er fordi mye av forskningen finner sted der. Stigberg & Stigberg (2020) konkluderer med lærernes indikasjon for mangel på tilstrekkelig kunnskap om programmering, samt et behov for at lærere og elever må forstå hvorfor og hvordan programmering er en del av matematikkopplæringen og en matematisk kontekst (Stigberg & Stigberg, 2020). Kjällander et al. (2021) trekker også frem usikkerheten til lærere rundt digital kompetanse. Til tross for deres usikkerhet skal de

lære elevene å bli digitalt kompetente. I tillegg formidles viktigheten av læreres forventninger til elevene, motivasjon og muligheten for å lære (Kjällander et al, 2021). Tillit til barns egen evne understrekes som viktig i forkant av integreringen av programmeringsaktiviteter (Otterborn et al, 2019).

Et annet aspekt som kommer tydelig frem i flere forskningsartikler fra lærernes perspektiv, er å ytre positivitet rundt å prøve og feile (feilsøke), samt at læreren lar elevene feilsøke før hen trer inn i diskusjonen (Heikkilä & Mannila, 2018; Kjällander et al., 2021; Chibas et al., 2018; Nouri et al., 2019). Kjällander et al. (2021) viser til programmering som en læringsaktivitet hvor elevene skal prøve, feile og prøve på nytt frem til de lærer. Dette skal også skje i samspill med medelever og læreren. Ved å gjennomgå elevenes løsninger, vil det bli tydeligere hva som er gjort feil, hvordan det kan endres og prøves ut neste gang (Kjällander et al, 2021; Heikkilä & Mannila, 2018). Heikkilä & Mannila (2018) trekker frem at personen som initierer til feilsøking varierer, men at det som oftest er læreren som initierer gjennom å poengtere et problem. Videre vil elevene bli klar over problemet og forsøke å løse det. Likevel ville det vært interessant å be læreren om å ikke tre inn i programmeringen «for tidlig», men heller ser an og la elevene forsøke å finne problemene selv (Heikkilä & Mannila, 2018).

Det kommer tydelig frem i forsking at lærere føler på usikkerhet og mangel på kunnskaper om og rundt programmering (Otterborn et al, 2019; Kjällander et al, 2021; Stigberg & Stigberg, 2020). Det anses som nødvendig at lærerne har visse forutsetninger til elevenes læring (Kjällander et al, 2021) og at lærerne har tillit til elevenes egne evner før programmering tas i bruk i undervisningen (Otterborn et al, 2019). Feilsøking i samspill med andre oppmuntres (Kjällander et al, 2021; Heikkilä & Mannila, 2018) og elevene må få mulighet til å undre og prøve seg frem før læreren trer inn i diskusjonen (Heikkilä & Mannila, 2018; Kjällander et al; Chibas et al., 2018; Nouri et al., 2019).

2.1.5 Implikasjoner

Stort sett all forskningen som er presentert over, er hentet fra internasjonale studier. Antakeligvis med ulike rammefaktorer som skiller seg fra den norske skolen. Fra en rapport gjort i 2016, vises det at programmering og algoritmisk tenkning er en hovedprioritet i flere europeiske land (Bocconi et al., 2018, s. 1), men i de nordiske landene varierer graden av integrering eller planleggende integrering i læreplaner og

nasjonale planer (Bocconi et al., 2018). Som nevnt over har programmering og algoritmisk tenkning fått en plass i den nye læreplanen, men i begynneropplæringen tas kun algoritmisk tenkning med som begrep i kompetansemålene. Tidligere forskning som er presentert, foregår i hovedsak i barnehagen (Heikkilä & Mannila, 2018; Otterborn et al., 2020; Palmér, 2017) eller på mellomtrinnet (Zhong et al., 2016; Stigberg & Stigberg, 2020; Iskrenovic-Momcilovic, 2019; Nouri et al., 2019), som indikerer manglende forskning på programmering i begynneropplæringen.

Tidligere forskning trekker frem viktigheten av programmering knyttet til elevenes hverdagsliv og algoritmisk tenkning, samt at samarbeid og lærerens veilederrolle er sentralt. Dermed er det hensiktsmessig å undersøke om og hvordan dette anvendes i klasserommet i praksis, og ha en tilnærming der interaksjoner og programmeringsundervisningens strukturer er i fokus. I tillegg til å høre lærerens begrunnelser. Ved å ta i bruk en sosiokulturell tilnærming, er dette mulig.

2.2 Sosiokulturell læringsteori

I et sosiokulturelt perspektiv legges det vekt på at kunnskap konstrueres i en kontekst gjennom samhandling (Dysthe, 2001b, s. 42). Samarbeid og interaksjon blir i denne teorien sett på som grunnleggende for læring generelt og ikke bare som et positivt element i læringsmiljøet. For å lære, er det sentralt å delta i sosiale praksiser der læring skjer (Dysthe, 2001b, s. 42). Ifølge Vygotsky (1978), begynner barns læring allerede før de begynner på skolen og den læringen de tilegner seg der, bygger på en tidligere historie (Vygotsky, 1978, s. 84). Læring skjer over alt og til alle tider (Dysthe, 2001a, s. 11). Det å lære og utvikle seg i vår tid, handler i stor grad om å utnytte kognitive ressurser som er knyttet til bruken av kulturelle redskaper, som stadig fornyes. Datamaskinen trekkes frem som et eksempel på et slikt kulturelt redskap (Dysthe & Igland, 2001, s. 77).

Læring trekker med seg utvikling, og uten læring ville en rekke utviklingsprosesser ikke vært mulige (Dysthe & Igland, 2001, s. 78). Som sagt over, mener Vygotsky (1978) at læring skjer før barnet begynner på skolen, og da har det med seg en lang læringshistorie (Vygotsky, 1978, s. 84). For å forklare dette forholdet, forteller Dysthe & Igland (2021) at Vygotsky tar utgangspunkt i måling av den mentale utviklingen til barnet (Dysthe & Igland, 2001, s. 78). Måten dette kan gjøres på, forklares i det følgende.

2.2.1 Den proksimale utviklingssonen

I *Mind in Society* (1978) trekker Vygotsky frem et eksempel om to barn på ti år. Begge barna mestrer å løse problemer opp til et nivå som tilsvarer utviklingsnivået åtte år, men ikke mer enn det. Når barna får hjelp med problemløsning, viser det seg at begge klarer å løse problemer opp til et høyere utviklingsnivå enn utgangspunktet. Forskjellen mellom det opprinnelige utviklingsnivået og utviklingsnivået med hjelp, er det Vygotsky kaller for *the zone of proximal development* (Vygotsky, 1978, s. 86), eller den proksimale utviklingssonen på norsk. Vygotsky (1978) definerer den proksimale utviklingssonen som:

> « *[...] The distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance or in collaboration with more capable peers*» (Vygotsky, 1978, s. 86).

Den proksimale utviklingssonen utgjør området mellom det barnet klarer på egenhånd og det barnet klarer med hjelp (Vygotsky, 1978, s. 87). Det som er den proksimale utviklingssonen i dag, vil ifølge Vygotsky (1978) bli det virkelige utviklingsnivået i morgen, og det barnet kan gjøre med hjelp i dag, vil hen klare alene i morgen (Vygotsky, 1978, s. 87).

Lock & Strong (2014) poengterer at læreren må klare å tilpasse undervisningen til elevenes ferdighetsnivå, for å kunne opprette en proksimal utviklingssone. Ny læring vil med andre ord ikke skje dersom nivået er for høyt eller for lavt. I tillegg, vil det å jobbe med andre innenfor sin proksimale utviklingssone føre til etablering av mål, samt videreutvikling av ressurser og verktøy for å nå dem (Lock & Strong, 2014, s. 153).

2.2.1.1 Scaffolding

Scaffolding, eller stillasbygging, er en videreføring av Vygotskys teori om den proksimale utviklingssonen (Belland, 2017, s. 28). Ifølge Wood et al. (1976), er diskusjoner om tilegnelse av ferdigheter eller problemløsning ofte basert på en antakelse om at eleven er alene og uten hjelp (s. 90). Dette blir ofte behandlet som et eksempel på modellering. Likevel innebærer innblandingen fra en veileder mye mer enn det, i form av en slags stillasprosess. Dermed er det mulig for en elev å løse et problem, utføre en oppgave eller oppnå et mål, som uten hjelp ville vært utenfor elevens evner (Wood et al., 1976, s. 90). Stillaset består i hovedsak av at veilederen «kontrollerer» de

elementene i oppgaven som i utgangspunktet er utenfor elevens kapasitet, slik at eleven kan konsentrere seg om de elementene som er innenfor sitt kompetanseområde (Wood et al., 1976, s. 90).

Belland (2017) trekker frem viktigheten av støtte gjennom stillaser i arbeid med problemløsning (Belland, 2017, s. 4). Det forklares som interaktiv støtte, med utgangspunkt i det eleven allerede vet, slik at eleven får ferdigheter i oppgaver som normalt er utenfor dens evner. Det må nødvendigvis ikke bare være læreren som gir stillasstøtte, da det også kan være jevnaldrende, som medelever, eller dataverktøy (Belland, 2017, s. 25). Det Belland (2017) kaller *peer scaffolding*, innebærer gitt stillasstøtte fra jevnaldrende elever, eller eldre elever (Belland, 2017, s. 25). Det poengteres at *peer scaffolding* krever et rammeverk som styrer stillaset og som kan veilede elevene med strategier, som bør tas i bruk under veiledningen. Studier viser at *peer scaffolding* påvirker positive kognitive utfall. Likevel er ikke *peer scaffolding* tilstrekkelig som eneste kilde til stillasstøtte, da elevene ikke har tilsvarende pedagogisk kompetanse og kunnskaper innenfor vurdering og tilpasning som læreren (Belland, 2017, s. 26).

2.2.2 Medierende artefakter

Innad den sosiokulturelle teorien er mediering et sentralt begrep (Dysthe & Igland, 2001, s. 77; Säljö, 2001, s. 83). Dysthe (2001b) forteller at et av de nye begrepene Vygotsky har innført i pedagogisk tenkning, er *mediering*, også kalt formidling. Mediering blir bruk om alle typer hjelp eller støtte i læringsprosessen uavhengig om det er av personer eller redskaper, som i vid forstand kalles for *artefakter* (Dysthe, 2001b, s. 46). Säljö (2001) viser til artefakter som redskaper vi benytter når vi blant annet skal løse problemer og bearbeide informasjon (Säljö, 2001, s. 31). I det daglige møter vi og bruker ulike typer fysiske redskaper til å opprettholde aktiviteter og løse problemer, som for eksempel datamaskiner, telefoner og kopieringsmaskiner (Säljö, 2001, s. 76). I programmering tas ulike fysiske redskaper i bruk som både kan anses som hverdagslige redskaper (Heikkilä & Mannila, 2018; Chibas et al., 2018; Aranda & Ferguson, 2018; Berg, 2021) og mer digitale redskaper (Otterborn et al., 2020; Palmér, 2017). Begge typer redskaper kan anvendes for å løse problemer.

Dysthe forteller at «*redskapar medierer læring på mange ulike* vis» (Dysthe, 2001b, s. 46). I den sosiokulturelle læringsteorien er samspillet mellom redskapet og den lærende

sentralt. Det gjelder blant annet hva introduksjonen av nye redskaper gjør med læringskulturen (Dysthe, 2001b, s. 47). Eksempler på artefakter som penn, kalkulator og skrivebok tenker naturligvis ikke på egenhånd. Tenkningen foregår hverken i apparatet eller i brukerens hode. Vi fungerer derimot i samspill med artefaktene (Säljö, 2001, s. 78). Gjennom disse artefaktene kan vi løse problemer og beherske sosiale praksiser på måter som vi ellers ikke ville ha kunnet. Det som beskrives her, er det Säljö (2001) kaller for et av de mest særpregende trekkene innenfor den sosiokulturelle utviklingen (s. 78). Likevel er det en risiko for å gjøre artefaktene abstrakte og virkelighetsfjerne, dersom evnen til å integrere de i forståelsen av utvikling og læring mangler (Säljö, 2001, s. 78).

Det viktigste medierende læringsredskapet mennesket har, er språket (Dysthe, 2001a; Säljö, 2001). Det er fordi læring skjer gjennom å lytte, snakke, skrive og lese (Dysthe, 2001a, s. 12). I og med språket og kommunikasjon er sentralt i et sosiokulturelt perspektiv, er det viktig å studere den språklige kommunikasjonen i læringssituasjoner (Dysthe, 2001a, s. 12). Språkets viktige rolle i den sosiokulturelle læringsteorien, trekkes ytterligere frem i delkapittel 2.2.3.1 Kommunikasjon.

2.2.3 Samarbeid

Det legges stor vekt på at læring skjer i samspill med andre innenfor den sosiokulturelle teorien. Dysthe (2001a) poengterer at læring og samspill er knyttet sammen og er innebygd i måten vi organiserer læring på, i institusjoner som skoler. I skolen blir barn og unge organisert i klasser, grupper og fellesskap. Det vil dermed være forskjeller på hvordan samspillet mellom deltakerne er (Dysthe, 2001a, s. 11). Derfor er det nødvendig å stille spørsmål ved om samspillet i fellesskapene faktisk fungerer som arbeidsfellesskap og læringsfellesskap. I tillegg til om det foregår samspill som er læringsfremmende og hva som eventuelt er læringshemmende (Dysthe, 2001a, s. 11). Som forskning viser, er samarbeid en svært sentral del av programmeringen (Zhong et al, 2016; Otterborn et al, 2019; Kjällander et al, 2021; Iskrenovic-Momcilovic, 2019) og ulike metoder for inndeling av elevpar er omdiskutert (Zhong et al, 2016; Iskrenovic-Momcilovic, 2019). Likevel trekkes samarbeid frem som blant annet en viktig del av å utvikle gode kommunikasjonsferdigheter (Zhong et al, 2016). I det følgende vil kommunikasjon og språk innenfor læring i et sosiokulturelt perspektiv fremheves.

2.2.3.1 Kommunikasjon

I et sosiokulturelt perspektiv på menneskelig læring og utvikling er kommunikative prosesser svært sentrale. Det er gjennom kommunikasjon at individer blir delaktig i ferdigheter og kunnskaper (Säljö, 2001, s. 38). Å lære av erfaringer gjennom å tolke hendelser i begrepslige termer kan sammenlignes. Eksempler på begreper som hjelper oss med å se likheter og forskjeller mellom objekter, samt lære oss å forholde oss til de i ulike situasjoner er: farge, form og vekt (Säljö, 2001, s. 35). Vi har en unik evne til å dele erfaringer med hverandre gjennom språket. I samspill med medmennesker låner og utveksler vi stadig informasjon, ferdigheter og kunnskaper (Säljö, 2001, s. 35). Et annet aspekt ved språket er at vi kan «oversette» språklige fenomener, termer og begreper til fysisk handling, og motsatt (Säljö, 2001, s. 38). Gjennom å kommunisere om hvordan ulike handlinger kan utføres, kan vi skape og overføre innsikter og praktiske ferdigheter, noe som kan ses på som en kraftfull ressurs. Det å for eksempel kunne lese, tegne og skrive innebærer som oftest en beherskelse av kommunikative praksiser, som også inneholder en form for fysisk virksomhet (Säljö, 2001, s. 38).

2.3 Learning by doing

Shanck et al. (2009) sier: «*There is only one effective way to teach students how to do anything, and that is to let them do it*» (Shanck et al., 2009, s. 164). Programmering er en aktivitet hvor vi lærer gjennom å gjøre (Mannila & Nordén, 2020, s. 71). John Deweys syn på kunnskap innebærer, ifølge Dysthe (2001b), at kunnskap blir skapt gjennom aktivitet (s. 33) og gjennom å samhandle med andre (Dysthe, 2001b, s. 43). Han blir med det ofte knyttet til uttrykket «learning by doing», noe som i dag har spilt en viktig rolle i deltakeraktive læringsformer i skolen (Vaage, 2001, s. 130). Relasjonen mellom kunnskap og handling samt hvilken etisk verdi aktiviteten har, er det som er det primære for Dewey (Vaage, 2001, s. 130).

2.4 Akkomodasjon og assimilasjon

Piagets beskrivelse av prosessene akkomodasjon og assimilasjon har ifølge Säljö (2001), likheter med et sosiokulturelt perspektiv (Säljö, 2001, s. 67). Vi er konstant i samspill med omverdenen og blir regulert av prosessene assimilasjon og akkomodasjon samtidig. Det er en grunnleggende tanke i Piagets syn på utvikling (Säljö, 2001, s. 61). Assimilasjon innebærer å ta inn og registre informasjon om hvordan verden rundt oss er

organisert og fungerer. Det oppstår dermed ingen overraskelser, da verden bekrefter antakelsene våre og opptrer slik vi forventer. Akkomodasjon innebærer en grunnleggende forandring i måten vi ser virkeligheten på (Säljö, 2001, s. 61). Säljö (2001) viser til et eksempel av et barn som ser at en ballong fylt med gass svever oppover istedenfor å falle ned (s. 61). Da oppstår det en ubalanse mellom barnets forestillingsverden og erfaring av virkeligheten. Den siste må da endres til at ikke alle objekter faller ned, for at en balanse igjen kan oppstå (Säljö, 2001, s. 61). Denne endringen kalles læring. De to prosessene fungerer samtidig og må være i balanse med hverandre (Sjøberg, 1996, s. 3).

2.5 Learning by discovery

Learning by discovery innebærer å lære gjennom å oppdage og er innenfor Dewey sin eksperimentelle fremgangsmåte (Vaage, 2001, s. 145). Denne undersøkelsesmodellen inneholder fem trinn:

1. *utgangspunktet er ein følt vanske, eit problem eller ein konflikt*
2. *problemet må så lokaliserast og definerast*
3. *koma med ulike løysingsforslag*
4. *resonnera omkring følgjene, konsekvensane av dei ulike løysingsforslaga*
5. *evaluering og vidare observasjon og eksperiment som fører til aksept eller forkasting av valde løysingar* (Vaage, 2001, s. 146).

I en pedagogisk sammenheng går denne metoden under problemløsningsmetoden. Vaage (2001) legger til at de ulike trinnene ofte kan gå over i hverandre i praksis, men at det ikke er en rekkefølge som må følges slavisk (Vaage, 2001, s. 146). I problemløsningsoppgaver er det nødvendig å stoppe opp, reflektere og tenke over hva som er neste steg (Vaage, 2001, s. 145). Algoritmisk tenkning er en problemløsingsstrategi (Sevik et al., 2016, s. 13) og som sagt tidligere, er algoritmisk tenkning en faktor for å lære seg å programmere (Aranda & Ferguson, 2018; Nouri et al., 2019; Berg, 2021).

2.6 Rammeverk for programmering i skolen

I NOU-meldingen *Fremtidens skole – Fornyelse av fag og kompetanser* (2015), også kalt Ludvigsenutvalget, gis det et kunnskapsgrunnlag og forslag om valg som vi som et

samfunn bør ta om kompetanser for fremtiden, samt fornyelse av fag. I tillegg skal det legge til rette for å skape gode liv for Norges borgere og et samfunn som er produktivt, samt bidra i en global verden. Det er fire kompetanseområder som ligger til grunn for fornyelsen av skolens innhold. De er: *fagspesifikk kompetanse, kompetanse i å lære, kompetanse i å utforske og skape* og *kompetanse i å kommunisere, samhandle og delta*. Disse kompetansene skal elevene utvikle gjennom arbeid med fagene (NOU 2015:8).

Ludvigsenutvalget (2015) viser til:

> *«Skolen bør […] bidra til at elevene lærer å utforske og skape. Det er viktig for at elevene skal kunne bidra i arbeid og samfunn og være med på å utforske og finne løsninger på nye utfordringer»* (NOU 2015:8),

samt at:

> *«[…] digital kompetanse [ses på] som en sentral del av fagområdene i skolen. Digital kompetanse er i dag en forutsetning for å kunne delta i ulike former for læring […] og for å delta aktivt i arbeids- og samfunnsliv»* (NOU 2015:8).

Sitatene er i tråd med det som nevnes i Overordnet del under «Skaperglede, engasjement og utforskertrang» i Læreplanverket for Kunnskapsløftet (2020): *«Kreative og skapende evner bidrar til å berike samfunnet. Elever som lærer om og gjennom skapende virksomhet, utvikler evnen til å uttrykke seg på ulike måter, og til å løse problemer og stille nye spørsmål»* (Kunnskapsdepartementet, 2017c). Akerbæk & Karlsen (2019) trekker frem at undervisning av programmering i skolen ikke trenger en annen begrunnelse enn at det gir mulighet til å utforske noe ukjent, å kunne uttrykke seg ved hjelp av en datamaskin som et medium, samt at det er en morsom, verdifull og utfordrende aktivitet i seg selv (Akerbæk & Karlsen, 2019, s. 158). Hovda & Liukas (2020) forteller at barn har en naturlig interesse og nysgjerrighet for verden rundt seg. Som voksne skylder vi dem å være i stand til å kunne svare på ulike spørsmål de måtte ha, samt tilby de riktig kunnskap om samfunnet de vokser opp i (Hovda & Liukas, 2020, s. 5).

2.6.1 Begynneropplæring

Ifølge Palm et al. (2020) handler begynneropplæringen i hovedsak om overgangen fra barnehage til skole og de fire første skoleårene (Palm et al., 2020, s. 13). Begrepet

begynneropplæring er nært knyttet til det som skjer i klasserommet de første skoleårene, men begrepet blir ikke definert i faglitteratur eller styringsdokumenter (Hoff-Jenssen et al., 2020, s. 143). I Hoff-Jenssen et al. (2020) sin studie, trekkes det frem to forståelser av begynneropplæringen: en *smal* og en *vid*. Samtidig vises det til en mer *helhetlig* forståelse som innebærer både den smale og den vide forståelsen. Den helhetlige forståelsen tar for seg både de faglige og sosiale aspektene innenfor opplæringen på skolen, som er med på å bidra til et godt klassemiljø (Hoff-Jenssen et al., 2020, s. 150). Det er fokus på at elevene skal lære om ulike fag og de grunnleggende ferdighetene, samt mer faguavhengige ferdigheter som å lære seg å gå på skolen (Hoff-Jenssen et al., 2020, s. 149). De grunnleggende ferdighetene er viktige for elevenes utvikling og deltakelse i samfunnslivet, arbeid og egen utdanning (Kunnskapsdepartementet, 2017b, s. 11). I det følgende vil jeg ta for meg to aspekter som er sentrale innenfor begynneropplæringen; lek og algoritmisk tenkning, og deres plass innenfor matematikkfaget i den nye læreplanen.

2.6.1.1 Lek i begynneropplæringen

Vatne (2006) viser til læreres bruk av lek som et pedagogisk virkemiddel innenfor matematikken i de tre første årene på barneskolen (Vatne, 2006, s. 70). I kompetansemålene etter 2. trinn, står det: *«[elevene skal kunne] lage og følgje reglar og trinnvise instruksjonar i leik og spel»*. I tillegg står det i kompetansemålene etter 4. trinn at: *«[elevene skal kunne] utforske og beskrive strukturar og mønster i leik og spel»*. Læreren skal, innenfor matematikken i begynneropplæringen, legge til rette for at: *«elevane får utforske matematikk gjennom å bevege seg, leike, undre seg og bruke sansane»*, samt skal *«elevane […] få høve til å prøve og feile […] [og] utvikle kompetansen sin i utforsking og problemløysing […] og kompetansen sin i kommunikasjon med matematiske omgrep»* (Kunnskapsdepartementet, 2019). I forsking trekkes trinnvise instruksjoner frem i forbindelse med programmeringsundervisning (Heikkilä & Mannila, 2018; Berg, 2021; Chibas et al., 2018; Nouri et al., 2019) samt at ordene «leken» og «fantasifull» benyttes i forbindelse med programmering (Aranda & Ferguson, 2018). Ifølge Palm et al. (2020) er barnet lekende og lærende, samt at barnet lærer gjennom lek (Palm et al., 2020, s. 21). Lek kan ses på som et virkemiddel i skrive-, lese-, matematikk- og engelskopplæringen og som et grunnlag for å skape et fellesskap i klassen (Vatne, 2006, s. 55), men for at leken skal fungere må elevene kunne samhandle med hverandre (Vatne, 2006, s. 64).

2.7 Algoritmisk tenkning

Begrepet *algoritmer* er et kjent begrep fra matematikken (Haraldsrud et al., 2020, s. 183) og finnes også overalt i hverdagen vår. Det er viktig å ha et bevisst forhold til hvordan ulike algoritmer virker og hvordan man kan lage og vurdere algoritmer. I hverdagen finnes algoritmer blant annet i matoppskrifter, strikkeoppskrifter og i Netflix sitt filmutvalg (Haraldsrud et al., 2020, s. 184).

A*lgoritmisk tenkning* er en metode som er svært sentral i programmering og som både kan øves på med og uten datamaskin (Haraldsrud et al., 2020, s. 189). Algoritmisk tenkning ble gjort kjent av Jeannette M. Wing (Lye & Koh, 2014) og er en problemløsningsprosess (Wing, 2006). Wing (2006) trekker frem viktigheten av å lære ulike strategier og ferdigheter rundt bruken av datamaskiner (Wing, 2006) som feilsøking, abstraksjon og problemløsning (Lye & Koh, 2014). Algoritmisk tenkning forklares av Liukas (2018) med at mennesket tenker på et problem på samme måte som en datamaskin kan løse det (s. 112). Det er kun mennesker som kan tenke algoritmisk. Det vil si å tenke i algoritmer, tenke logisk, evnen til mønstergjenkjenning, bryte ned problemer og sortere bort detaljer som er uvesentlige (Liukas, 2018, s. 112). Mønstergjenkjenning går ut på gjenkjenning av strukturer og likheter mellom ulike rutiner og prosesser. Når mønstrene i et problem er analysert, kan likheter i disse mønstrene benyttes til å konstruere algoritmer. Eksempler på dette er gjennom spill, lek eller matematiske analyser og eksperimenter (Haraldsrud et al., 2020, s. 189).

Algoritmisk tenkning er i tråd med flere aspekter innenfor det som kalles *21st Century Skills*, eller kompetanser og ferdigheter for det 21. århundre (Sevik et al., 2016, s. 10). Eksempler på 21st Century Skills er kreativitet, problemløsning og kritisk tenkning (Lye & Koh, 2014; Nouri et al., 2019; Binkley et al., 2012), i tillegg til metakognisjon, samarbeid, kommunikasjon, medborgerskap og digital kompetanse (Sevik et al., 2016, s. 10). Skolen vektlegger programmering som en viktig kompetanse for å lære, arbeide og leve i dagens samfunn. Disse kompetansene refereres ofte til 21st Century Skills (Sevik et al., 2016, s. 10).

Når elever skal utvikle sin algoritmiske tankegang innebærer det å tilnærme seg problemer på en systematisk måte. I tillegg til å foreslå løsninger som datamaskiner kan brukes til for å løse deler av problemet. Med andre ord: *«tenke på hvilke steg som skal til for å løse et problem, og bruke sin teknologiske kompetanse for å få datamaskinen til*

å løse problemet» (Sevik et al, 2016, s. 14). Mannila (2020) viser til algoritmisk tenkning som et sett med ferdigheter, arbeidsmetoder og konsepter som bidrar til å løse problemer, slik at vi kan la datamaskinen hjelpe oss mest mulig (s. 40). Det finnes likevel ikke en universell modell, men ulike områder har ulike behov (Mannila, 2020, s. 40). Begrepet blir ifølge Mannila (2020) sett på som et paraplybegrep for holdninger og evner som kan trenes opp gjennom å programmere (Mannila, 2020, s. 79).

Uenigheten er stor rundt begrepets innhold og hvilke strategier som skal benyttes for å vurdere utviklingen av unge menneskers algoritmiske tenkning (Brennan & Resnick, 2012; Lye & Koh, 2014), noe som er utfordrende når algoritmisk tenkning skal integreres i utdanningen (Nouri et al., 2019, s. 4). I det følgende vil jeg ta for meg ulike definisjoner og forklaringer på algoritmisk tenkning, samt trekke frem definisjonen som tas forbehold om i denne studien.

2.7.1 Definisjoner på algoritmisk tenkning

I 2010 la Cuny, Snyder, og Wing (2010) frem en oppdatert definisjon på algoritmisk tenkning. Definisjonen trekker frem visjonen om at alle kan ha nytte av å lære prinsippene, begrepene og tilnærmingene rundt informatikk (Nouri et al., 2019, s. 3). Mannila (2020) beskriver to modeller for algoritmisk tenkning (s. 79). De er hentet fra Barefoot Computing og fra forskerne ved MIT, som også står bak programmeringsverktøyet Scratch. Barefoot Computing sin modell tar for seg seks konsepter og fem tilnærminger innenfor algoritmisk tenkning. Konseptene er *logisk tenkning, algoritmer, nedbrytning av problemer i deler, mønstergjenkjenning, abstraksjon og evaluering*. Tilnærmingene er *utforskning, skape, feilsøking, utholdenhet og samarbeid*. Dette er et lignende rammeverk for algoritmisk tenkning som forskerne ved MIT har utviklet (Mannila, 2020, s. 79). De fem tilnærmingene i Barefoot Computing sin modell, samsvarer godt med Ludvigsenutvalgets (2015) vurdering av viktig kompetanse for norske elever i fremtiden (Sevik et al., 2016, s. 14).

I likhet med Barefoot Computing sin modell som bygger på forståelsen for et sett med sentrale programmeringskonstruksjoner, har forskerne ved MIT fastslått at algoritmisk tenkning også handler om arbeidsmetoder (Mannila, 2020, s. 79-80). I tillegg har de gjennom arbeid med Scratch, i samarbeid med barn i ulike aldre, også lagt merke til at barn samtidig utvikler en forståelse av seg selv, sitt forhold til andre samt den digitale verden rundt seg (Mannila, 2020, s. 80). Dette passer ikke inn under konseptene eller

arbeidsmetodene i modellen. Dermed valgte de å legge til en annen dimensjon ved algoritmisk tenkning, i form av tre perspektiver: *å uttrykke seg* - innse at programmering er verktøy for å skape, *å samarbeide* - det er en styrke å samarbeide med og for andre, og *å stille spørsmål* - kunne stille spørsmål om den digitale verden (Mannila, 2020, s. 80). Selv om det ikke finnes en klar definisjon på algoritmisk tenkning, viser de to modellene bredden av ferdigheter og arbeidsmetoder som kan inkluderes. Ved å blant annet lage pedagogiske aktiviteter rundt programmering kan det samtidig arbeides med konseptene, arbeidsmetodene og perspektivene som inngår i modellene til Barefoot Computing og MIT (Mannila, 2020, s. 85).

Til tross for at det ikke finnes én direkte definisjon på algoritmisk tenkning vil jeg i forskningsprosjektet mitt støtte meg til Barefoot Computing sin modell (Mannila, 2020, s, 79). Algoritmisk tenkning handler også om å uttrykke seg, samarbeide og stille spørsmål, slik som forskerne ved MIT hevder (Mannila, 2020, s. 80). Ved å ta hensyn til dette håper jeg å kunne formidle at programmering omhandler mer enn å forstå datamaskiner og økt kompetanse innenfor digitale verktøy. Det handler også om å kunne gi elevene tilgang til gode verktøy for problemløsning (Sevik et al., 2016, s. 13).

2.8 Programmering

Seymour Papert (1993) sier: «*I believe that certain uses of very powerful computational technology and computational ideas can provide children with new possibilities for learning, thinking, and growing emotionally as well as cognitively*» (Papert, 1993, s. 17-18). I utdanningssituasjoner hvor barn møter datamaskiner, brukes datamaskinene som et verktøy til å gi nivåbaserte oppgaver, tilbakemeldinger og informasjon. I LOGO-miljøet (Papert, 1993, s. 19) er forholdet reversert. Da er det barnet, allerede i barnehagealder, som har kontrollen. Med andre ord kan det sies at det er barnet som programmerer datamaskinen. Gjennom denne læringsprosessen, å lære datamaskinen å «tenke», utforsker barnet hvordan de selv tenker (Papert, 1993, s. 19).

I dagens skole har flere tatt i bruk elev-iPad'er som et verktøy i opplæringen, men som Haraldsrud et al. (2020) poengterer, blir ikke barn automatisk digitalt kompetente ved å få en iPad. *«Det er hva vi gjør med det digitale mediet, som har noe å si»* (Haraldsrud et al., 2020, s. 15). Nå som samfunnet gjennomdigitaliseres må det vurderes om de digitale verktøyene spiller på lag med samfunnets verdier. Likestilling, som innebærer at alle skal bli ivaretatt og ha like muligheter, er en viktig del av fellesskolen. Haraldsrud et al.

(2020) mener digitaliseringen utfordrer dette (Haraldsrud et al., 2020, s. 15). Likevel poengteres det at vi bidrar til å opprettholde samfunnets store fellesskap ved å sørge for at alle har tilgang på det grunnleggende innenfor all digital teknologi (Haraldsrud et al., 2020, s. 16). Dette er i tråd med debatten som tas opp i innledningen i forbindelse med programmering i læreplanen.

2.8.1 Programmering i klasserommet

Som nevnt er programmering ikke et nytt fenomen, men relativt nytt i den norske skolen og for lærerne som skal undervise det. Sitatet under trekker frem viktige synspunkter i en begynnerfase med programmering i grunnskolen:

> «*Så länge ingen årskull har gått igenom grundskolan med programmering som en del av utbildningen från första början, kan vi inte veta vad som fungerar bäst. [...] Till en början startar dock största delen av både lärare och elever från noll. Detta innebär att vi måste börja med grunderna för alla som inte tidigare haft någon undervisning i programmering, oberoende av om det handlar om en grupp nyfikna förstaklassare eller tonåringar i åttonde klass*» (Mannila, 2020, s. 126).

Mannila (2020) trekker frem eksempler på blant annet aktiviteter og samtaletemaer som kan være nyttig når «det grunnleggende» skal begynnes med. Det innebærer, fra og med, analog programmering, altså programmerings aktiviteter uten bruk av datamaskiner, til programmering med ulike apper som code.org og mer fysiske programmeringsaktiviteter med bruk av roboter (Mannila, 2020, s. 126).

«*Programmering byr både på mange muligheter og mange utfordringer i klasserommet*» (Haraldsrud et al., 2020, s. 161). Programmering kan ifølge Haraldsrud et al. (2020), brukes på tvers av fag. Dette er igjen med på å bidra til dypere forståelse av problemløsningsstrategier og ulike arbeidsmåter (Haraldsrud et al., 2020, s. 14). Når programmering innføres som en naturlig del av pensum i flere fag kan elever blant annet få flere sjanser til å prøve og feile, og flere tilnærminger til fagstoffet (Haraldsrud et al., 2020, s. 161). På lik linje med andre fag, som for eksempel norsk og matematikk, er det noen elever som ikke kommer til å like programmering. Likevel må læreren arbeide for at alle elever kan føle på mestring og glede (Haraldsrud et al., 2020, s. 163). Dersom programmering er nytt både for elevene og læreren vil det ofte være mange spørsmål rundt programmeringsundervisningen. Dermed poengterer Haraldsrud et al.

(2020) at det er viktig å lære elevene å feilsøke tidlig. Det finnes ofte flere løsninger på et problem og ved å lære elevene å bruke hverandre og utforske sammen, kan elevene komme langt i programmeringsprosessen. Selv profesjonelle programmerere gjør feil og hvis løsningen på problemet ikke finnes med en gang, er det ingen skam å prøve igjen (Haraldsrud et al., 2020, s. 165).

3 Metode

Basert på studiens problemstilling og valg av teori, er det hensiktsmessig at metodevalget reflekterer hvordan læreren underviser programmering i begynneropplæringen og begrunnelsene som ligger bak. I metodekapittelet skal jeg beskrive hvordan forskningsprosessen min har vært og begrunne valgene jeg har tatt. Innledningsvis vil jeg kort plassere meg i et forskningsparadigme, før jeg tar for meg prosjektets forskningsdesign. Kapittelet består også av redegjørelser for forskningsmetoder og forskningsprosessen. Det innebærer de benyttede metodene i dette prosjektet: en metodetriangulering bestående av deltakende observasjon og semistrukturert intervju. I tillegg til en beskrivelse av hvordan forskningsprosessen foregikk. Til slutt vil jeg ta for meg analyseprosessen og diskutere studiens forskningsetikk.

3.1 Forskningsparadigmer

3.1.1 Kvalitativ forskningsmetode

I et forskningsprosjekt kan en kvalitativ eller en kvantitativ forskningsmetode velges mellom. Krumsvik (2014) trekker frem *isfjellmetaforen* i forbindelse med forskjellen på kvalitativ og kvantitativ forskningsdesign. Her forklares den kvalitative tilnærmingen som delen av isfjellet som er under vann og skal blant annet illustrere dybdeforståelsen, det kontekstnære og -avhengige (Krumsvik, 2014, s. 46), samt nærheten til feltet og informantene. Den kvantitative tilnærmingen illustreres dermed av selve isfjellet over vannflaten og er mest opptatt av årsaksforklaringer av grunnlaget som allerede er til stede. I kvalitativ forskning ligger et vitenskapsteoretisk rammeverk til grunn for å spesifisere forskningsspørsmål. Deretter blir disse gransket gjennom de valgte metodene, samlet inn, analysert og skrevet ut (Krumsvik, 2014, s. 47). Ved å velge en kvalitativ forskningsmetode i en studie, åpner det for å undersøke egenskaper ved enkelte fenomener (Kvarv, 2020, s. 156). Forskningsprosjektet mitt har en klar kvalitativ vinkling, da jeg ønsker å gå i dybden og undersøke hvordan programmering tas i bruk i undervisningen i begynneropplæringen.

Kvalitativ forskning kalles også for *eksplorerende metoder* (Høgheim, 2020, s. 129) og viser til utforskning, samt har som mål å skape teori om det som forskes på (Høgheim, 2020, s. 27). En kvalitativ tilnærming blir ofte forbundet med forskning som innebærer

inkludering mellom forsker og forskningsdeltakere, som for eksempel ved intervju og deltakende observasjon. Når metoder som disse tas i bruk, vil en forståelse av sosiale fenomener utvikles, med bakgrunn i de koblingene vi etablerer gjennom forskningsprosjektet. Denne forståelsen for sosiale fenomener er et viktig mål innenfor kvalitative tilnærminger (Thagaard, 2018, s. 11). Ved å ta i bruk en kvalitativ tilnærming, vil jeg ha mulighet til å se på ett fenomen i sin naturlige kontekst, samtidig som jeg kan skape en tilknytning til forskningsdeltakeren gjennom metodene som er valgt.

3.2 Forskningsdesign

3.2.1 Etnografisk studie

Ifølge Høgheim (2020) handler *etnografi* om å beskrive ulike kulturer i vårt eget samfunn. Spesielt innenfor pedagogisk forskning finnes det noen fellestrekk med etnografi (s. 149). Det handler altså om å være til stede og studere andre mennesker i deres situasjon eller felt (Høgheim, 2020, s. 149). Etnografi betyr bokstaveligtalt «*writing about people*» (Johnson & Christensen, 2012, s. 48). Forskeren ser på atferdsmønstre, språk og samhandling i gruppene, og bakgrunnen eller meningen bak (Høgheim, 2020, s. 149). Ifølge Gobo (2011) har etnografiens originalt vært basert på observasjon. Selv om blant annet intervju i hovedsak er basert på lytting og å stille spørsmål (Gobo, 2011, s. 16), er det også essensielt innenfor etnografien å lytte til samtalene som foregår under forskningen, samt stille spørsmål. Likevel blir intervju og lignende sett på som et slags hjelpemiddel, da det som skiller etnografien fra andre metoder, er en mer aktiv rolle knyttet til observasjon (Gobo, 2011, s. 17). Den vanligste metoden i en etnografisk studie er observasjon med feltnotater, men i tillegg kan andre metoder som for eksempel intervju tas i bruk. Forskeren er med andre ord ikke låst til å bare bruke observasjon som metode (Høgheim, 2020, s. 150). Ved for eksempel å ta i bruk intervju med deltakerne som observeres, kan forskeren komme mer i dybden av observasjonene. Det poengteres at det er mye relevant informasjon som ikke kan observeres. Dermed er det en styrke å intervjue disse deltakerne (Høgheim, 2020, s. 150). Et eksempel som trekkes frem innenfor pedagogisk etnografisk forskning er å se på hva mennesker gjør, de strategiene som tas i bruk og meningen som ligger bak handlingene. Hvis det trekkes inn i en skolesammenheng kan eksempelvis læreres praksis i klasserommet undersøkes. I tillegg kan lærernes holdninger og oppfatninger til

elevene eller til læringen som finner sted også studeres (Woods, 1986). Min studie har dermed et klart etnografisk synspunkt, da jeg som forsker går inn i et klasserom og undersøker lærerens praksis i programmeringsundervisningen. I tillegg fanges lærerens begrunnelser opp i etterkant av undervisningen. Fokuset er å studere lærerens handlinger og strategier som tas i bruk, samt meningen bak (Høgheim, 2020, s. 150). Dette gjøres gjennom et sosiokulturelt synspunkt, basert på at kunnskap konstrueres i samspill med andre gjennom praktiske aktiviteter (Skaalvik & Skaalvik, 2018, s. 78). Dette er også i tråd med forskningsparadigmet jeg har plassert meg i, en kvalitativ tilnærming.

3.2.2 Casestudie

Casestudier er empiriske undersøkelser hvor forskeren undersøker et fenomen i dybden og i sin naturlige kontekst (Postholm, 2005, s. 17 og 51; Yin, 2014, s. 18). Slike studier kjennetegnes ved å ta for seg mye informasjon av få informanter (Thagaard, 2018, s. 51), ved bruk av flere metoder (Gobo, 2011, s. 16; Johnson & Christensen, 2012, s. 49). Casestudier kan være beskrivende, men også beskrivende og tolkende, eller på samme tid være beskrivende, tolkende og vurderende (Postholm, 2005, s. 51). Det er spesielt casestudier som er beskrivende og tolkende som er relevant for min studie. Det er fordi de inneholder en beskrivende del, altså er nyttige dersom det er mangel på teori på feltet, samt har de som hensikt å illustrere, støtte, utvikle og utfordre eksisterende teori. Ved å samle inn mye data vil det være mulig å beskrive, tolke og teoretisere de studerte fenomenene i forhold til problemstillingen (Postholm, 2005, s. 51). Studien min omhandler et fenomen som har vært forsket en del på tidligere, men er relativt nytt i den norske skolen. I tillegg er det viktig at informantens meninger, beskrivelser og utførelser løftes frem og kan diskuteres sammen med eksisterende teorier og besvare problemstillingen. På den måten vil både informantens perspektiver og mine vitenskapelige synspunkter trekkes frem. Grunnen til at en beskrivende, tolkende og vurderende tilnærming ikke er like relevant for min studie, er at det vurderende aspektet innebærer at forskerens synspunkter og vurderinger, med utgangspunkt i teorier, er mer i fokus enn informantens. Noe som er «på grensen» til et kvalitativt studie (Postholm, 2005, s. 51), da informantens perspektiver skal løftes frem og ikke falle i bakgrunnen av forskerens vurderinger og betraktninger (Postholm, 2005, s. 52). Dette vil ikke være tilfellet for min studie, da essensen i studien er informantens utførelser og begrunnelser for disse i undervisning med programmering. Basert på dette kvalifiserer studien min

som en *indre casestudie* (Postholm, 2005, s. 52), da hensikten er å løfte frem en lærers praksis i programmeringsundervisning. En sosiokulturell tolkning av det som utforskes, innenfor en casestudie, er ønskelig og blir kalt for en etnografisk casestudie (Postholm, 2005, s. 53). Min studie er dermed en etnografisk casestudie, da jeg studerer en lærer i sin naturlige kontekst, klasserommet. Jeg tar i bruk metoder som støtter det etnografiske forskningsdesignet og som vil bidra til å undersøke fenomenet i dybden.

3.3 Forskningsmetoder

Hovedpoenget med en metode defineres av Postholm & Jacobsen (2018) som en forskningsprosess, der forskeren tvinges til å tenke gjennom valgene som må tas og hvilke konsekvenser valgene vil få (Postholm & Jacobsen, 2018, s. 25). I det følgende vil jeg ta for meg metodene som er benyttet til dette prosjektet, og begrunnelser for metodevalget.

3.3.1 Triangulering

Ifølge Postholm (2005) er triangulering en prosedyre som både brukes i etnografiske studier og i casestudier (Postholm, 2005, s. 138). Denne prosedyren er for å sikre kvaliteten på en forskningsstudie (Postholm, 2005, s. 84) ved å blant annet ta i bruk flere ulike datainnsamlingsstrategier (Postholm, 2005, s. 134). Det begrunnes ved at dersom flere ulike kilder kan bekrefte og understøtte hverandre er det med på å styrke studien (Postholm, 2005, s. 132). Studien er som nevnt tidligere en etnografisk casestudie, hvor jeg tar i bruk to ulike datainnsamlingsmetoder: deltakende observasjon og semistrukturert intervju. I og med at det holdes intervjuer etter hver observasjonsøkt, vil denne prosessen bekrefte og understøtte hverandre. Spørsmålene som stilles i intervjuer er delvis basert på observasjonene som gjøres i undervisningsøkten. På den måten vil jeg innhente data på to forskjellige måter som kan styrke hverandre og studien som en helhet.

3.3.2 Deltakende observasjon

Ifølge Gobo (2011) omfatter etnografisk metodikk to forskningsstrategier. De er ikke-deltakende observasjon og deltakende observasjon (Gobo, 2011, s. 23). I min studie tar jeg i bruk deltakende observasjon i åtte ulike undervisningsøkter. Deltakende observasjon kjennetegnes ved at forskeren oppholder seg i felten og er aktivt deltakende sammen med forskningsobjektene (Thagaard, 2018, s. 70). Postholm (2005) poengterer

at deltakende observasjon i en kvalitativ studie er tjenlig dersom formålet med forskningen er å forstå hva som skjer på et bestemt sted, til et bestemt tidspunkt (Postholm, 2005, s. 27). Gjennom deltakende observasjon kan jeg observere hva læreren sier og gjør i en undervisningssituasjon, samt hvordan læreren interagerer med og veileder elevene under ulike programmeringsaktiviteter. Dermed har jeg grunnlaget for det som gjøres i praksis, som senere gjennom intervju vil begrunnes ytterligere (Thagaard, 2018, s. 70). Begrepet *feltroller* tas i bruk for de ulike måter forskeren kan etablere seg i felten. En variasjon i bruken av deltakelse i aktiviteter og distanse anbefales, for å reflektere over inntrykkene (Thagaard, 2018, s. 70). Årsaken bak valget av deltakende observasjon er i stor grad at jeg har mulighet til å interagere med elevene og observere læreren i en veiledningssituasjon med elevene. I tillegg har jeg mulighet til å bidra dersom det er behov. Samtidig er rollen min som observatør å distansere meg for å kunne notere ned observasjoner og eventuelle spørsmål som dukker opp. På den måten vil jeg ikke ha en passiv rolle i klasserommet, men i tillegg ha mulighet til å samtale med elevene og læreren underveis. Dette er en forutsetning jeg har tenkt nøye over, da jeg har en relasjon til elevene og læreren fra før av. I kapittel 3.7.2.1 Forskerrollen tar jeg for meg min rolle som forsker med en relasjon til forskningsfeltet og informanten.

3.3.3 Semistrukturert intervju

Semistrukturerte intervjuer kan ses på som kvalitative intervjuer, da det er denne typen intervju som oftest tas i bruk innenfor en kvalitativ studie (Thagaard, 2018, s. 91; Gleiss & Sæther, 2021, s. 80). Semistrukturerte intervjuer er ofte basert på en delvis strukturert intervjuguide hvor temaene er fastlagt på forhånd. Rekkefølgen på spørsmålene bestemmes underveis og forskeren kan tilpasse seg intervjuobjektets beskrivelser og tanker. Selv om strukturen på intervjuet er fleksibel, er det viktig å sørge for at problemstillingen blir belyst (Thagaard, 2018, s. 91). Som oftest vil oppfølgingsspørsmål stilles for å konkretisere og utdype momenter som anses som interessante i intervjuet (Gleiss & Sæther, 2021, s. 80). Semistrukturerte intervjuer har som formål å innhente kvalitativ kunnskap som er i tråd med det forskningsspørsmålene etterspør. Når forskeren hører på forskningsobjektets beskrivelser, bør de generelle spørsmålene følges opp med konkrete spørsmål for å holde fast ved «tråder» fra intervjuobjektet (Krumsvik, 2014, s. 125). I min studie tas semistrukturerte intervjuer i bruk med formål om å innhente begrunnelser for elementer fra observasjonene. Selv om

en intervjuguide er laget og baseres på observasjonsskjemaet, skrives det ned spørsmål underveis i undervisningsøkten som kan stilles for å utdype mer om enkelte aspekter. I og med at intervjuene baseres på observasjonsøkten, vil alle spørsmål og oppfølgingsspørsmål ha sammenheng med økten og felles observasjoner. Dermed vil det være enklere for meg som forsker og intervjuer, å følge opp «tråder» fra lærerens svar. I tillegg er det en fordel å ha noe kunnskap om programmering før intervjuene finner sted. Det ble gjort lydopptak av alle intervjuene i forskningsperioden ved å benytte appen «Nettskjema-Diktafon». Bruk av lydopptak og intervjuprosessen generelt, vil trekkes frem ytterligere i kapittel 3.5.1 Gjennomføring.

3.4 Utvalg

Det viktigste kriteriet rundt valg av informant til min studie, var å velge en lærer på småtrinnet som aktivt bruker programmering i undervisningen til elevene. Gjerne en lærer som også arbeider på en skole med et mål og verdier om digital kompetanse og utforskning. Etter å ha undersøkt hva de ulike barneskolene i mitt område arbeidet for og deres verdier, tok jeg kontakt med én av skolene. Skolen skriver at de tar i bruk digitale verktøy aktivt i undervisningen og at de har teknologisk utstyr som blant annet brukes til programmering. Jeg kom i kontakt med en kontaktlærer på 4. trinn som blant annet har videreutdanning i programmering. Læreren har også fra elevene gikk i 1. klasse tatt i bruk programmering i undervisningen.

Ved å velge én informant som både har teoretisk og praktisk grunnlag innenfor bruk av programmering i undervisningen i begynneropplæringen, støttes valget av casestudie som metode for denne studien. Studien innebærer å forske på hvordan lærere kan anvende programmering i undervisningen. Av den grunn vil det være hensiktsmessig med en informant som har tidligere og nåværende erfaringer og kunnskaper om hvordan dette kan gjøres i praksis. Gjennom observasjon får jeg undersøkt hvordan læreren tar i bruk programmering i undervisningen, altså i sin naturlige kontekst, i tillegg til at jeg gjennom intervjuene får en dypere forståelse for begrunnelsene som ligger bak handlingene.

3.5 Forskningsprosessen

Som jeg var inne på innledningsvis benytter jeg en kvalitativ tilnærming, som også kalles for en eksplorerende metode. Forskningsprosessen startet med at jeg innhentet

informasjon, som videre ble analysert for å finne mønstre i datamaterialet mitt. Til slutt brukte jeg informasjonen til å trekke slutninger om hvordan et fenomen er. Denne måten å forske på kalles en *induktiv* måte, hvor mer spesifikke data startes med før de løftes opp til mer generelle slutninger (Høgheim, 2020, s. 28). Ved å velge en eksplorerende metode, kan jeg «fange opp» fenomener som kanskje ikke er representert i teorier (Høgheim, 2020, s. 28). En *deduktiv* måte er det motsatte av en *induktiv* måte, hvor forskeren har utarbeidet ulike variabler som ikke endres i løpet av forskningen. Postholm (2005) forteller at gjennom interaksjonen mellom en *induktiv* og *deduktiv* tilnærming, utvikler forskeren sin forståelse for forskningen som en helhet og forskningsinformantenes meninger (Postholm, 2005, s. 36).

En kvalitativ undersøkelse blir ikke sett på som en stegvis og lineær prosess, selv om en kvalitativ metode innebærer ulike stadier. Det er altså ikke en bestemt rekkefølge som må følges når det arbeides kvalitativt, og forskeren går gjerne frem og tilbake mellom de ulike delene i forskningsprosessen (Maxwell, 2013, s. 3; Kvarv, 2020, s. 157; Postholm, 2005, s. 36). Maxwell (2013) kaller derfor kvalitativ forskning for en *«do-it-yourself»*-prosess (Maxwell, 2013, s. 3). I og med planen for en kvalitativ forskningsstudie ikke er helt fastlagt på forhånd, bør forskeren ha et åpent sinn. Når det tas i bruk en kvalitativ forskningsprosess, fokuserer forskeren på forskningsdeltakernes perspektiv og det er dette fokuset som avgjør om forskerens antakelser bekreftes eller ikke (Postholm, 2005, s. 36).

3.5.1 Gjennomføring

Før det ble skrevet mal for observasjonsskjema og intervjuguide, gikk jeg igjennom teori om sosiokulturell læringsteori og tidligere forskning om programmering i undervisningen. Ut ifra det, lagde jeg noen punkter som jeg syntes virket spennende og interessante, som i tillegg var i tråd med problemstillingen min. Jeg noterte ned aspekter som var ønskelig å undersøke under observasjon, og i oppfølgingsintervjuene fra observasjonsøktene. Med disse notatene og punktene utformet jeg et observasjonsskjema (vedlegg 1) og en intervjuguide for samtalen etter observasjonsøkten (vedlegg 2). Da alle observasjonsøktene og intervjuene var gjennomført, gikk jeg igjennom materialet og spørsmål jeg hadde stilt, for å finne ut hva jeg ønsket å vite mer om. Jeg utformet dermed en ny intervjuguide (vedlegg 3), som innebar mer generelle spørsmål om bruk av programmering i undervisningen fra elevene gikk i 1. klasse og frem til i dag. I tillegg til spørsmål som var rettet mot

lærerrollen og hvordan det kan utformes og planlegges undervisningsøkter med programmering i begynneropplæringen.

Gjennom forskningsprosessen er det gjennomført totalt åtte observasjonsøkter med intervju i etterkant av hver observasjonsøkt. I tillegg hadde jeg et lengre intervju som var mer generelt der spørsmålene tok for seg lærerrollen og bruk av programmering i undervisningen fra 1. til 4. klasse. Under observasjonsøktene noterte jeg både i notatbok og i observasjonsskjemaet. I tillegg skrev jeg ned spørsmål om aspekter jeg ønsket å vite mer om av det som ble sagt eller gjort av læreren i løpet av øktene. Etter hver observasjonsøkt hadde jeg et intervju med læreren. I intervjuet stilte jeg spørsmål rundt det jeg hadde observert og ønsket å vite mer om. I og med at intervjumetoden er semistrukturert, forberedte jeg noen spørsmål i forkant som samsvarte med observasjonsskjemaet. I tillegg stilte jeg spørsmål som jeg hadde notert under observasjonsøkten og oppfølgingsspørsmål dersom læreren nevnte elementer som jeg ønsket å få utdypet mer om. Det ble utført lydopptak av intervjuene, noe læreren hadde samtykket til på forhånd. Intervjuene foregikk i klasserommet, etter endt skoledag eller i pauser hvor elevene ikke var til stede. Lydopptakene ble deretter automatisk opplastet til UiO Nettskjema, som videre ble benyttet i transkriberingsprosessen.

Intervjuet som var basert på mer generelle spørsmål om lærerens tanker rundt programmering i undervisningen, var lengre enn de forrige. Jeg ønsket å gå i dybden og få lærerens begrunnelser for bruk av programmering i begynneropplæringen. I tillegg til hvordan lærere på småtrinnene kan ta i bruk programmering i elevenes undervisning, selv om erfaringen med programmering muligens ikke er optimal.

Etter hver forskningsøkt ble observasjonsskjemaet og intervjuguiden lagt i en egen mappe, sammen med en oversikt over når forskningsøktene fant sted. Denne mappen ble trygt plassert i et skap med lås, for å sikre at dokumentene ble holdt adskilt og trygt oppbevart.

3.5.1.1 Undervisningsøkter

Gjennom datainnsamlingsperioden observerte jeg flere økter som hadde fokus på ulike aspekter av programmering, samt oppgaver som bidro til å forsterke dette. I det følgende vil jeg presentere kort hva de ulike materiellene er, og hva undervisningsøktene gikk ut på. De tre første punktene går inn under analog

programmering, mens de tre siste inngår i digital programmering og var en del av stasjonsundervisning.

«Hei Ruby»

«Hei Ruby» er et undervisningsmateriell som introduserer den grunnleggende forståelsen i algoritmisk tankegang. Gjennom denne boken lærer barn hvordan store problemer kan brytes ned til mindre, legger stegvise planer, leter etter mønstre og tenker utenfor boksen. Boka er delt opp i kapitler, hvor hvert kapittel er en kort fortelling fra Ruby sin verden. Der kan barna oppleve spenning og bruke fantasien sin. I tillegg er det ni små leksjoner i algoritmisk tankegang. Bakerst i boken er det ulike oppgaver som hører til hvert kapittel. Forfatteren forteller at disse oppgavene er basert på skaperglede. Det er opp til hver enkelt som benytter boka om boka leses som en helhet eller kun fokuserer på ett kapittel om gangen, og de tilsvarende oppgavene (Liukas, 2018, s. 3).

«Hei Ruby»-boka ble tatt i bruk i to av øktene. Den første økten tok utgangspunkt i kapittel 1: Møt Ruby. Læreren leste kapittelet høyt for elevene og deretter gjorde de en av oppgavene i grupper:

> «*Bestemte lille Ruby. Merka du at Ruby var litt egenrådig når hun skulle rydde rommet sitt. Det har hun lært seg av datamaskinen sin. Den trenger nøye beskrivelser av alle detaljer man må gjøre, også de som er selvsagte for oss. Hva ville du sagt at Ruby skulle gjort i følgende situasjoner. Skriv ned steg for steg hvordan man: Spiser frokost […]»* (Liukas, 2018, s. 73).

Elevene fikk i oppgave å lage en «oppskrift» som innebar trinnvise instruksjoner for hvordan Ruby spiser frokost. Elevene var delt opp i grupper og skrev ned instruksjonene på et ark. Da instruksjonene var skrevet ned, skulle de «spilles ut» i klasserommet. En elev leste opp instruksjonene og en annen elev skulle utføre de i praksis. Denne økten hadde som mål å øve på å lage trinnvise instruksjoner, feilsøke, samarbeide og forsøke å «tenke» som en datamaskin.

En annen undervisningsøkt tok for seg en annen oppgave i boka. Oppgaven ble gjort i plenum i starten av økten sammen med læreren. Oppgaven var:

> «*Danse, danse, danse! Ta på deg danseskoa – nå skal vi bevege oss! Ruby og vennene hennes liker å danse. Alle har sine favorittbevegelser. Prøv å gjøre det samme som dem! Hvor mange ganger klarer du å gjøre dansen deres*

> *(snøleoparden ville kalt det en loop) Er det mulig å danse slik til favorittsangen din?*» (Liukas, 2018, s. 90).

I boka er det tegnet opp instruksjoner som læreren tegnet på tavla. Instruksjonene inneholdt ulike dansebevegelser inndelt i tre sekvenser og informasjon om hvor mange ganger hver sekvens skulle gjennomføres. En sekvens inneholder ulike dansesteg og flere sekvenser utgjør et program. Elevene øvde først på hver sekvens med riktig antall gjentakelser før de prøvde med musikk. Deretter fikk elevene i oppgave å lage egne dansebevegelser i et tilsvarende skjema som ble bruk i plenum. Oppgaveteksten var:

> «*Min dansebevegelse: Tegn din egen dansebevegelse. Finn på navn på dine egne dansesteg og sett dem i den rekkefølgen du vil (bruk gjerne rutene om du vil). Ikke la dansebevegelsen bli for lang, slik at du kan gjenta den flere ganger. Tenk også ut når den skal begynne og når den skal slutte*» (Liukas, 2018, s. 91).

I tillegg til å lage dansebevegelser, fikk elevene mulighet til å bestemme eventuelle betingelser for sitt danseprogram. Mot slutten av økten viste alle elevene frem sin dans med musikk. For mer informasjon om «Hei Ruby»-bøkene, gå inn på https://www.heiruby.no/ (Hei Ruby, u. å.).

LEGO

I en av undervisningsøktene tok læreren i bruk LEGO-klosser som konkreter i en oppgave. Elevene ble delt i grupper og fikk utdelt likt antall LEGO-klosser. Likevel var ikke alle gruppenes LEGO-klosser like i form, farge eller størrelse. Elevene fikk i oppgave å bygge en figur med klossene de hadde fått utdelt. Deretter skulle de skrive ned instruksjoner på et ark om hvordan de bygde figuren sin. LEGO-klossene og arket med instruksjoner ble gitt til en annen gruppe som så skulle forsøke å bygge samme figur ut fra de nedskrevne instruksjonene. Læreren hadde ikke lagt føringer for bruk av programmeringsspråk, så elevene kunne skrive ned eller tegne instruksjonene slik de selv ønsket.

Puslespill

I en annen undervisningsøkt, fordelte læreren elevene i grupper på tre og fire. Hver gruppe fikk utdelt hvert sitt puslespill med tolv brikker. En av elevene skulle ha bind for øynene, mens de andre skulle gi tydelige instruksjoner på hvordan eleven skulle pusle puslespillet. De andre elevene fikk ikke lov til å røre puslespillbrikkene og måtte

dermed gi tydelige, detaljerte instruksjoner til eleven som puslet. Elevene byttet på slik at alle fikk puslet.

Blue-Bot

Blue-Bot ble tatt i bruk i to økter med stasjonsundervisning. Jeg vil kun ta for meg to av stasjonsoppgavene i den ene økten, da de trekkes frem i resultatkapittelet og vil drøftes senere. Den første økten besto av fire oppgaver med Blue-Bot på gjennomsiktige matter med kort inni. Elevene ble delt inn i grupper og fordel på de fire stasjonene. På den ene stasjonen var det fokus på en- og tosifrede regnestykker med svar opp til tretti. Elevene trakk et kort med et regnestykke på. Deretter skulle de programmere Blue-Bot, ved hjelp av pilene på roboten, til ruten med riktig svar. Elevene på gruppa rullerte slik at alle fikk prøve.

På en annen stasjon skulle elevene trekke kort med ord på, for så å programmere Blue-Bot med pilene til et ord som rimte på ordet på kortet. Læreren forteller at elevene har arbeidet mye med rimord tidligere og selv om rimord ikke er en del av kompetansemålene for elevenes årstrinn, er det en fin repetisjonsoppgave. I tillegg får elevene øvd på å programmere Blue-Bot.

Som ved alle teknologiske «duppeditter», oppsto det problemer med Bluetooth-tilgangen i løpet av økten. Det førte til at elevene ikke fikk like stort utbytte av stasjonene, da store deler av tiden gikk til tilkobling av roboten. Likevel kan det ses på som lærdom, da ulike tekniske problemer lett kan oppstå i ulike situasjoner. Læreren vekslet dermed mellom veiledning med selve programmeringsaktiviteten og veiledning med tilkobling. Læreren forteller at det er en forutsetning som må tas hensyn til ved planlegging av programmeringsøkter. For mer informasjon om Blue-Bot, gå til https://www.terrapinlogo.com/products/robots/blue/blue-bot-family.html (Terrapin Logo, u. å.).

Sphero BOLT

I den andre økten med stasjonsundervisning var en av oppgavene at elevene skulle bli kjent med Sphero BOLT-roboten. BOLT er en robot-ball som både kan styres via en app, med en slags kontrollfunksjon, eller ved hjelp av blokkprogrammering. Denne roboten hadde elevene ikke benyttet tidligere. Læreren fortalte at fokuset dermed var på at elevene skulle bli kjent med hvordan roboten fungerte og forsøke å kjøre rundt med den. Etter hvert fikk elevene en «utfordring», som læreren kalte det. De skulle bruke

blokkprogrammeringsverktøyet i appen og programmere BOLT til å lage et kvadrat på gulvet. Her måtte elevene ta i bruk ulike blokker og kunnskaper i matematikk for å løse oppgaven. I og med dette var første gang elevene prøvde BOLT, mente læreren det var hensiktsmessig å gi elevene frihet til å prøve seg frem og undersøke hvordan BOLT virker, men også veilede dem underveis. Les mer om Sphero BOLT på https://sphero.com/products/sphero-bolt (Sphero, u. å.).

Osmo Coding

I den andre stasjonsøkten, sammen med BOLT, innebar en annen stasjon å ta i bruk Osmo Coding. Dette er en god måte å introdusere programmering for barn. Osmo Coding er en spillbasert app hvor figuren Awbie trenger hjelp til å samle ulike ting som jordbær og trestammer. For å gjøre dette tas kodebrikkene i bruk ved å plassere de foran iPad´en. Ved hjelp av et speil som plasseres over kameraet foran på iPad´en, blir kommandoen brikkene utgjør lest av og fører til at Awbie utfører bevegelsene. Her kan elevene øve på å gi instruksjoner og ta i bruk ulike programmeringsbegreper ved hjelp av brikkene. Elevene arbeidet to og to på én iPad og samarbeidet for å gi riktige kommandoer til Awbie. For mer informasjon og instruksjoner rundt Osmo Coding, gå inn på https://www.playosmo.com/en/shopping/kits/coding/ (Osmo, u. å.).

3.6 Analyseprosessen

Hensikten med en kvalitativ analyseprosess er å sortere datamaterialet for å gjøre materialet mer forståelig. Analyseprosessen starter i det forskningen i feltet og intervjuene starter, og materialet samles inn (Postholm & Jacobsen, 2018, 2. 139). Som nevnt tidligere har forskningsprosessen og analyseprosessen bestått av en kombinasjon av en induktiv og deduktiv måte. Ifølge Gleiss & Sæther (2021), kalles det gjerne for en *abduktiv* analysemåte. Det defineres ved at datamaterialet stegvis analyseres og identifiserer temaer på tvers av intervjuer og observasjoner (Gleiss & Sæther, 2021, s. 171). I casestudiers analyse skal meningen i dataene finnes og utvikle en forståelse av det studerte fenomenet (Postholm & Jacobsen, 2018, s. 157). I analysen av casestudier letes det etter mønstre som av og til er kjent på forhånd eller dukker opp i løpet av analysen (Postholm & Jacobsen, 2018, s. 158). I det følgende vil jeg ta for meg prosessene transkribering og skriving av feltnotater, i tillegg til hvordan jeg har gått frem i analysen.

3.6.1 Transkribering

Gjennom transkripsjon blir intervjuet, samtalen som utvikles gjennom to personer, abstrahert og gjort om til en skriftlig form. Det sies at transkripsjon er oversettelser fra talespråk til skriftspråk (Kvale & Brinkmann, 2010, s. 186). Ved å transkribere samtalen fra intervjuet til en skriftlig form blir det strukturert på en måte som gjør det bedre egnet for å analyseres (Kvale & Brinkmann, 2010, s. 188). Transkribering av intervjuene ble gjennomført etter hver forskningsøkt. Intervjuenes lydopptak ble spilt av, fra UiO Nettskjema, og nedskrevet i et eget dokument. Lærerens navn ble anonymisert under transkriberingen for å opprettholde lærerens personvern. Ifølge Kvale & Brinkmann (2010) er transkribering av intervjuer en anstrengende og kjedelig prosess (s. 189), men at når datamaterialet senere skal analyseres er det verdt strevet. Når lydopptaket transkriberes til tekst, er det viktig at det som skrives er nøyaktig det samme som uttales i intervjuet (Kvale & Brinkmann, 2010, s. 189). Alle gjentakelser, «eh»-er og lignende tas med for å skape en direkte oversettelse av intervjuet. Kvale & Brinkmann (2010) trekker frem viktigheten av å lytte til lydopptaket flere ganger, for å være sikker på at den transkriberte teksten stemmer med selve intervjuet (s. 193). Det poengteres at kvaliteten på opptaket eller feiltakelser kan gjøre at de samme ordene kan ha ulik betydning basert på måten de skrives ned (Kvale & Brinkmann, 2010, s. 193). Da jeg transkriberte de ulike intervjuene, hendte det at jeg flere ganger måtte lytte til opptaket på nytt for å få fatt på det som ble sagt. Kvaliteten på lydopptaket og små bakgrunnslyder skapte til tider vanskeligheter med å transkribere korrekt første gangen.

Etter hvert som intervjuene ble transkribert, skrev jeg ned stikkord og elementer som gikk igjen i intervjuene, som senere var utgangspunktet for kodingen og kategoriseringen. Dette fant jeg svært nyttig, da jeg til slutt hadde mye omfattende datamateriale. Jeg vil trekke frem dette ytterligere i kapittel 3.6.2 Analyse av datamaterialet.

3.6.2 Feltnotater

Når det observeres i et forskningsprosjekt, er det vanlig å skrive feltnotater. Høgheim (2020) forteller at det finnes to muligheter når det gjelder feltnotater (s. 137). Feltnotater kan enten registreres i løpet av eller i etterkant av observasjonsøkten. Det poengteres at det kan være ulemper ved begge. Det er mulig å gå glipp av sentrale ting som skjer under skrivingen i felten, men samtidig kan detaljer glemmes dersom feltnotatene

skrives etter observasjonsøkten. Dermed er det å foretrekke å både notere i felten og i etterkant av observasjonen (Høgheim, 2020, s. 137). Som nevnt over skrev jeg ned notater både i observasjonsskjemaet og i en skrivebok under selve observasjonsøkten. Jeg noterte L foran direkte sitater som læreren sa i klasserommet eller direkte til elever, og i egne avsnitt for mine observasjoner av hendelser og lærerens atferd. Dermed ble det mer strukturert og oversiktlig da jeg i etterkant skulle skrive feltnotat etter observasjonsøkten.

Etter hver observasjonsøkt skrev jeg ut notatene mine fra skriveboken og observasjonsskjemaet. Jeg noterte øverst hvilket trinn økten foregikk i, antall elever og øktens varighet, i tillegg til et kort sammendrag av hva økten innebar. Det vil si hva oppgaven(e) var, målet for økten og eventuelt andre relevante notater som kom frem i økten. På den måten følte jeg selv at struktureringen av hvert feltnotat ble tydeligere, da jeg etter hvert skulle begynne med analysen av materialet. I selve feltnotatet skrev jeg en mer sammenhengende tekst ut ifra notatene mine. Jeg markerte tydelig hva som var mine direkte observasjoner og hva læreren hadde sagt til elevene i løpet av økten, i likhet med strukturen i feltnotatet fra felten. På den måten var det også enklere å kunne se sammenhengen mellom observasjonene og lærerens begrunnelser i intervjuet etter økten.

3.6.3 Analyse av datamaterialet

Etter at datainnsamlingsperioden var ferdig og alt materialet var transkribert og omskrevet til feltnotater, begynte jeg med analysering av materialet. Som jeg var inne på tidligere hadde jeg skrevet ned noen stikkord som gikk igjen i datamaterialet. I tillegg leste jeg igjennom alt av materialet og noterte i et skjema det jeg tenkte var relevant i forbindelse med problemstillingen min. Notatene besto av kategorier som var mer overordnet fra datamaterialet mitt, med underpunkter som passet innenfor kategorien. Med andre ord kan det sies at jeg så etter *temaer* i datamaterialet for å få en bedre oversikt over sammenhengen mellom de ulike funnene. En slik prosess kalles for en *tematisk innholdsanalyse*, hvor det analyseres på en systematisk måte som beskriver innholdet (Anker, 2021, s. 40). Målet med en slik analyse er å gruppere disse underpunktene i mer generelle kategorier (*temaer*), som sammen kan besvare problemstillingen (Johannessen, Rafoss & Rasmussen, 2021, s. 279). Johannessen, Rafoss & Rasmussen (2021) presenterer fire faser for tematisk analyse, som er en oversikt over hvordan jeg gikk frem i min analyse:

1. *Forberedelse (der du skaffer til veie og får oversikt over data)*
2. *Koding (der du fremhever og setter ord på viktige poenger i data)*
3. *Kategorisering (der du kategoriserer de kodede dataene dine i mer generelle temaer)*
4. *Rapportering (der du rapporterer temaene og deres innhold)* (s. 282).

Det understrekes at rekkefølgen ikke må følges slavisk, men at det kan varieres ved å gå frem og tilbake mellom fasene (Johannessen, Rafoss & Rasmussen, 2021, s. 283). Etter de tre første punktene hadde jeg laget en tabell som tok for seg fem kategorier, med fem til åtte koder under hver kategori (vedlegg 4). Etter hvert ble noen kategorier og koder luket vekk og noe omgjort til funn. Analysen består av funn som jeg observerte i samtlige forskningsøkter og som læreren også begrunnet i intervjuene. Funnene er dermed hentet fra hele datamaterialet som en helhet. Forskningsspørsmålene til studien er basert på metodene som benyttes: observasjon og intervju. Dermed er det forskningsspørsmålene som danner resultatene.

Da skrivingen av restultatkapittelet begynte, hadde jeg mye transkribert materiale som skulle tas i bruk som sitater. Gjentakelser, småord og «eh»-er ble fjernet og erstattet med klammeparenteser. Det var fordi informantens beskrivelser og meninger skulle komme frem som representativt og for å få en mer helhetlig tekst. I tillegg er enkelte ord i setninger stokket om for å få en bedre flyt i sitatet og teksten som en helhet. Dette ble ikke gjort for å endre på informantens begrunnelser. Tvert imot for å få frem informantens budskap og kunnskaper ytterligere på en representativ måte. Når analysen skrives frem på en troverdig måte, handler det i stor grad om å vise til det som gjøres. Dette kalles for transparens eller gjennomsiktighet. Ved at en leser kan følge analysene og selv synes de fremstår som rimelige tolkninger, er analysen transparens (Anker, 2021, s. 88). Dette har jeg forsøkt etter beste evne å mestre.

3.7 Feilkilder ved forskningsprosjektet

Innhenting av forskningsdata foregikk over en periode som var preget av flere utsettelser grunnet sykdom. Til tross for disse utsettelsene, har datainnsamlingen vært innholdsrik og jeg har innhentet relevant data som kan bidra til å svare på problemstillingen. I og med at samfunnet vårt i dag er preget av en pandemi, har det også hatt innvirkning på programmeringsundervisningen til elevene. Dette er i hovedsak grunnet at lærerne ikke kunne ta i bruk ulike programmeringsverktøy på grunn av fare

for smitte. I tillegg var elevene jevnlig delt opp i kohorter og mindre grupper. Grunnet dette, valgte læreren å ha undervisningsøkter som var planlagt ut ifra et mer nybegynnerstadium, med en antakelse om manglende kunnskaper i programmering hos elevene. Likevel er det realistisk å anta at enkeltelever sitter inne med programmeringskunnskaper til tross for en lengre pause mellom undervisningsøktene. Læreren forteller at undervisningen bærer preg av en mer «grunnleggende» start. Fokuset var i hovedsak på den analoge programmeringen og innlæring av begreper, programmeringsspråk og essensen med trinnvise instruksjoner.

Gjennom arbeid med masteroppgaven er det tydelig at jeg har mye bekreftende funn. En stor årsak til det, mener jeg, er at programmering ikke er nytt for læreren og elevene, til tross for en lengre pause. I tillegg er klassen preget av et relativt godt klassemiljø, samt viser stor motivasjon for programmering. Det er ikke gitt at det er tilfellet for alle klasser. Til tross for dette var utgangspunktet for valget av informant, en lærer med mye erfaring i bruk av programmering i begynneropplæringen. Dermed er det også å forvente at elevene og læreren er engasjerte i forbindelse med programmeringsundervisning. Klassen og deres utgangspunkt kan gis et stempel som «prakteksempel» i en slik forskningssituasjon. Det kan dermed ikke garanteres at andre klasser og skoler kan tilegne seg samme nivå av kunnskap og ferdigheter med det første. Et annet aspekt som kan ha bidratt til bekreftende funn, er at skolen har tilgang på mye analogt og digitalt utstyr som kan tas i bruk i programmering. Det forutsetter også at læreren er kjent med utstyret som benyttes i undervisningen. I tillegg tas det i bruk utstyr, i enkelte av timene i forskningsperioden, som elevene allerede har et forhold til.

I min studie var klassens og lærerens egenskaper innenfor programmeringsundervisning å forvente, da utvalget og problemstillingen forutsatte en informant og et felt med tilstrekkelig kunnskap og ferdigheter innenfor programmering i begynneropplæringen. Selv om dette kan ses på som en svakhet for min studie, velger jeg å være kritisk til resultatene mine. Likevel ønsker jeg å presentere funn i tråd med annen forskning og enkelte aspekter som kan generaliseres til andre felt og lærere. Som en forlengelse av dette vil jeg i det følgende ta for meg en vurdering av andre elementer i prosjektet mitt og dets kvalitet i forbindelse med pålitelighet og gyldighet.

3.8 Vurdering av prosjektets kvalitet

En viktig del av et forskningsprosjekt er å vurdere og reflektere over kvaliteten på ens eget forskningsarbeid. For å gjøre dette er det vanlig å ta utgangspunkt i pålitelighet (reliabilitet) og gyldighet (validitet) (Gleiss & Sæther, 2021, s. 201). I det følgende vil jeg ta for meg disse to aspektene i et sosialkonstruktivistisk syn, da dette synet er i tråd med mitt kvalitative forskningsprosjekt.

3.8.1 Pålitelighet

Pålitelighet, eller *reliabilitet*, handler i stor grad om kvaliteten på forskningsprosessen og hvorvidt undersøkelsen er til å stole på. Med et utgangspunkt i en sosialkonstruktivistisk tradisjon vil forskningen ha spor av forskerens subjektivitet, som gjør det vanskelig å holde seg helt objektiv i forskningen. Det vektlegges dermed å være balansert og å kunne inkludere alle relevante perspektiver i samspill med forskerens tolkninger av datamaterialet. Gleiss & Sæther (2021) poengterer at forskeren gjennom triangulering, vil få flere perspektiver som senere kan diskuteres opp mot hverandre (s. 203). Som nevnt tidligere, består studien av en triangulering av intervju og observasjon. Ved å ta i bruk begge metodene, som baseres på hverandre, vil jeg ha to forskjellige perspektiver som inngår i samme økt. Dette er med på å styrke prosjektets pålitelighet, da metodene styrker hverandre.

Hvorvidt jeg har påvirket forskningsprosessen, er i forbindelse med utvalg av informant til min casestudie. Informanten er, som nevnt, valgt grunnet mye kunnskap og ferdigheter innenfor programmering, både teoretisk og praktisk sett. Det at informanten sitter inne med mye kunnskaper er positivt for studien, da informanten både har grunnlag gjennom flere år med programmeringsundervisning i begynneropplæringen.

Gjennom forskningsprosessen har jeg etter beste evne forsøkt å skrive så transparent som mulig, da jeg ønsker at andre selv kan være med på å vurdere de valgene som har blitt tatt (Gleiss & Sæther, 2021, s. 204). Gleiss & Sæther (2021) trekker frem viktigheten rundt å være balansert og inkludere alle perspektiver som er relevante i analysen (Gleiss & Sæther, 2021, s. 203). I analyseprosessen har jeg tatt utgangspunkt i aspekter som går igjen i store deler av datamaterialet. Disse aspektene er hentet fra observasjonene, som videre blir ytterligere forklart gjennom informantens begrunnelser fra intervjuene. På den måten inkluderes alle relevante perspektiver, for å kunne besvare

problemstillingen med to ulike synspunkter. Dette er igjen med på sikre pålitelighetenen i studien.

3.8.2 Gyldighet

Gyldighet, eller *validitet*, brukes til å si noe om kvaliteten på datamaterialet og forskerens fortolkninger og konklusjoner (Gleiss & Sæther, 2021, s. 201). Med andre ord kan det sies at gyldighet handler om i hvilken grad de ulike delene av forskningsdesignet henger sammen (Gleiss & Sæther, 2021, s. 204). Ved å ha en kvalitativt casestudie vil jeg ha mulighet til å se hvordan én lærer, med stor kompetanse i feltet, tar i bruk programmering i undervisningen i begynneropplæringen. Jeg har da mulighet til å komme mer i dybden i hva denne læreren gjør og begrunner valgene sine med, i løpet av en lengre periode. Det ville ellers ikke vært mulig dersom jeg hadde hatt flere informanter. Som nevnt tidligere, har jeg tatt i bruk triangulering av flere metoder. Ved å kombinere ulike metoder kan det sikres at informasjonen som kommer fram er mest mulig riktig, noe som er med på å styrke gyldigheten i prosjektet (Gleiss & Sæther, 2021, s. 205). I tillegg ser jeg en tydelig sammenheng mellom mine egne funn og funn fra tidligere forskning. Dette er også med på å styrke gyldigheten, som Gleiss & Sæther (2021) trekker frem (Gleiss & Sæther, 2021, s. 205).

Deltakervalidering er sentralt i en sosialkonstruktivistisk tradisjon (Gleiss & Sæther, 2021, s. 205-206). En stor del av min studie er lærerens utførelser og begrunnelser, da jeg gjennom en kvalitativt casestudie undersøker lærerens bruk av programmering i undervisningen i begynneropplæringen. Basert på lærerens perspektiver, tolker jeg det opp mot tidligere forskning og relevant teori. Dette gjøres ikke for å bekrefte at det læreren gjør og sier er «korrekt», men jeg ønsker å undersøke hva som gjøres i praksis. I tillegg om det sammen med teori kan bidra til en økning av mine og forskningsfeltets kunnskaper om programmering i begynneropplæringen i norsk skole.

For å styrke gyldigheten vil jeg kort trekke frem hva slags kunnskaper og begrensninger de ulike metodene har for mitt prosjekt. Ved å velge deltakende observasjon hadde jeg mulighet til å ha interaksjoner med elevene i løpet av økten uten at det ville ha noen påvirkning på forskningsdataene. I tillegg til å både komme nært på læreren og distansere meg i undervisningen. Deltakende observasjon har som hensikt å få en mer direkte tilgang til lærerens handlinger. Til tross for dette går min forståelse gjennom det som observeres. På den måten vil jeg ikke ha direkte tilgang til en observert virkelighet

(Gleiss & Sæther, 2021, s. 206). Ved å ta i bruk semistrukturerte intervjuer har jeg mulighet til å stille spørsmål basert på allerede planlagte elementer, men også få lærerens begrunnelser og tanker rundt det som ble observert undervisningsøkten. I og med at intervjuet i seg selv ikke kan si noe om hva som faktisk ble gjort i klasserommet (Gleiss & Sæther, 2021, s. 206), blir det styrket av å kombineres med observasjon. I kapittel 3.8.3 Generalisering vil jeg ta for meg casestudie-metoden ytterligere. I det følgende vil jeg ta for meg min forskerrolle og hvordan relasjoner mellom forsker og informant kan påvirke prosjektet og datainnsamlingen. Dette har mye å si for gyldigheten av prosjektet som en helhet.

3.8.2.1 Forskerrollen

Relasjonen mellom forsker og informant kan påvirke kunnskapen som utvikles i intervjuet (Gleiss & Sæther, 2021, s. 87). Dermed bør forskeren reflektere over relasjonen til informanten og hvordan de oppfatter hverandre. I tillegg trekker Gleiss & Sæther (2021) frem viktigheten av å tenke over rollen som vil være mest hensiktsmessig, ut ifra det som ønskes å få ut av en observasjonssituasjon (Gleiss & Sæther, 2021, s. 107). I etterkant av utvalget til masterprosjektet, ble jeg ansatt som lærervikar på skolen og har vært i klassen opptil flere ganger i uken i en lengre periode før forskningen fant sted. Ved å være i klassen jevnlig har elevene og læreren blitt godt kjent med meg, og jeg dem. Dermed er de vant til å ha meg til stede i klasserommet.

Det er selvfølgelig både fordeler og ulemper ved å forske i en klasse og på en lærer jeg har en relasjon til. Gleiss & Sæther (2021) trekker frem begrepene «insider» og «outsider», som handler om forholdet eller relasjonen mellom forskeren og den som studeres, både i en intervju- og observasjonssituasjon. For en forsker som er «på innsiden», er forskeren med andre ord en del av feltet som studeres (Gleiss & Sæther, 2021, s. 88). En fordel ved å være insiderforsker, er at forskeren har kunnskaper og erfaringer som kan bidra til å gjøre det lettere å få tilgang til informanten, samt stille gode spørsmål. Som en ulempe trekkes det frem at viktig informasjon kan unngås eller overses, da det kan tas for gitt (Gleiss & Sæther, 2021, s. 89). For å hindre dette har jeg valgt å ha en rolle som forsker hvor jeg holder meg i bakhold under store deler av undervisningen. På den måten får jeg et overblikk over klasserommet og læreren. I tillegg påvirkes elevene i liten grad av meg som observatør. Til tross for dette, har jeg valgt deltakende observasjon for å ha mulighet til å tre inn i undervisningssituasjonen og observere læreren i interaksjon med elevene under aktivitetene. Det er sannsynlighet

for at elevene tar kontakt med meg i løpet av undervisningsøkten, da de er vant til at jeg er til stede i klasserommet. Ved å veksle mellom en distansert rolle mens jeg noterer og observerer på avstand, og å ha mulighet til å ta del av elevenes og lærerens interaksjoner med hverandre, får jeg observert undervisningen uten at det har stor påvirkning på elevenes læring og datainnsamlingen min. Gjennom å ha en relasjon til læreren gjør det at intervjusituasjonen blir mer som en samtale. Det virker mer avslappet enn om det hadde vært en informant jeg ikke hadde kjennskap til, noe jeg antar gjelder for begge parter. Ved at jeg tok i bruk observasjonsrollen som forklart over, ga det meg mulighet til både å stille spørsmål basert på overordnede observasjoner, men også spørsmål angående situasjoner hvor læreren interagerte med elevene. Det er blant annet disse situasjonene som har vært utgangspunktet i flere funn hvor lærerrollen trekkes frem.

Som nevnt tidligere, foregår det en pandemi samtidig som arbeidet med masteroppgaven finner sted. Dermed har løsningen med å forske på en lærer i en klasse jeg har en relasjon til, og er jevnlig sammen med i klasserommet, vært en god løsning for mitt forskningsprosjekt. Ved å ha mulighet til å forske i løpet av en arbeidsdag, har jeg kunnet gjennomføre samtlige forskningsøkter med kort tid mellom hver økt, noe jeg ved en annen omstendighet antageligvis ikke ville hatt mulighet til under en pandemi.

3.8.3 Generalisering

Generalisering har vært diskutert i forbindelse med casestudier (Kvale & Brinkmann, 2010, s. 265; Flyvbjerg, 2006). Begrepet *analytisk generalisering* handler om i hvilken grad funn fra et studie kan brukes som en slags oppskrift for hva som kan skje i andre studier (Kvale & Brinkmann, 2010, s. 266). Ifølge Kvale & Brinkmann (2010) hviler analytisk generalisering på rikholdige kontekstuelle beskrivelser samt at det inkluderer forskerens argumentasjoner. Det avgjøres med andre ord ut ifra hvor godt forskeren forklarer situasjoner, som for eksempel intervju- og observasjonsprosessen, samt at det er høy kvalitet med tanke på blant annet validering av forskningen (Kvale & Brinkmann, 2010, s. 269). Ifølge Gleiss & Sæther (2021) innebærer analytisk generalisering eksempelvis at noen kategorier utarbeides, som kan ha relevans for andre (Gleiss & Sæther, 2021, s. 207) skoler eller klasserom. Postholm (2005) trekker frem sin egen studie, som jeg kan relatere til mitt eget masterprosjekt. Gjennom min kvalitative casestudie kan leseren overføre enkelte aspekter til eget klasserom. På den måten tilpasses handlingene i min studie til informantens klasseromssituasjon. Som Postholm (2005) poengterer trenger nødvendigvis ikke aktiviteter som fungerer i ett

klasserom å fungere i et annet klasserom. På det grunnlaget kan ikke studien min direkte overføres eller generaliseres (Postholm, 2005, s. 38). Likevel er det muligheter for å hente inspirasjon fra aspekter fra min studie og undersøke det ytterligere, noe jeg vil trekke frem ytterligere i 5.4.2 Videre forskning.

3.9 Forskningsetikk

Som forsker er det mitt ansvar å sørge for at jeg arbeider etter de gjeldende etiske retningslinjene (Høgheim, 2020, s. 80). NESH (2021) poengterer at retningslinjene bør ligge til grunn fra planlegging og gjennomføring til levering og formidling (NESH, 2021, s. 8). Informanten har skrevet under på et samtykkeskjema som inneholder informasjon om prosjektet og dets formål. I tillegg har informanten blitt informert om retten til når som helst å kunne trekke samtykket, få innsyn i opplysninger og stille spørsmål. Informanten ble også opplyst om at alle personopplysninger vil bli anonymisert og at det dermed ikke vil være mulig å bli gjenkjent i prosjektet. Det opplyses også om at det vil bli tatt lydopptak av intervjuene og hvordan disse blir behandlet i etterkant. Informasjonsskrivet og samtykkeskjemaet (vedlegg 5) er gjennomgått av NSD, i likhet med et utkast av intervjuguiden (vedlegg 6), og vurdert prosjektet som godkjent ut ifra lovverk om personopplysninger.

Jeg ønsker å poengtere at selv om jeg observerer økter i klasserommet med elevene til stede, vil ikke detaljer om elevene involveres i prosjektet. Det er fordi problemstillingen og formålet med mitt masterprosjekt er å undersøke lærerens bruk av programmering i undervisning. Dermed har det ikke vært nødvendig å innhente samtykke fra elevenes foresatte. I tillegg har både jeg og informanten taushetsplikt og dermed nevnes ingen av elevenes navn eller andre personopplysninger, hverken i mine notater fra observasjonene eller i intervjuene med informanten.

Basert på det jeg har presentert i dette kapittelet, har jeg gjort mitt beste til å forholde meg til de forskningsetiske retningslinjene og vært så redelig som jeg kan.

4 Resultater

Som sagt i 3.6 Analyseprosessen baseres resultatkapittelet på forskningsspørsmålene:

> **F1:** Hva kjennetegner programmeringsundervisning som fremmer aspekter innenfor fagfornyelsen og samfunnets digitalisering?
>
> **F2:** Hva mener læreren er viktig i planleggingen, gjennomføring og vurdering av programmeringsundervisningen på 1. til 4. trinn?

Forskningsspørsmål 1 er mest gjeldende i observasjon, mens forskningsspørsmål 2 er mest fremtredende i intervju. Til sammen danner de resultatene.

4.1 Analog programmering

Det første hovedfunnet jeg ønsker å trekke frem er at analog programmering brukes i undervisningen i samtlige økter. Hovedfunnet deles opp i underkategorier, som innebærer lærerens begrunnelser for bruk av analog programmering i undervisningen samt observasjoner fra øktene.

4.1.1 Programmeringsbegreper og -språk

Læreren knytter programmeringsbegreper og -språk til ulike aktiviteter i undervisningen. Målet for øktene har i stor grad vært å lære ulike programmeringsbegreper som gjenta (loop), sekvens og program, i tillegg til trinnvise instruksjoner. Læreren forklarer at det er disse begrepene som har vært mest hensiktsmessig å lære om i forhold til utstyret elevene senere vil bruke, når de går over til digital programmering:

> «*Det å gjenta ting, en sekvens og hva et program er, er jo de viktigste å ha kontroll på, og å kunne skille de fra hverandre. […] Det er viktig at de har med seg de begrepene og forståelsen inn i det digitale*».

Læreren poengterer likevel at begrepene som velges å trekke inn i undervisningen, avhenger av hvor i programmeringen elevene er, og hva som er neste steg. Læreren nevner også at begrepene kan tas i bruk i analog programmering i tilknytning til mer hverdagslige situasjoner, samt å kunne dra paralleller til for eksempel matematikken.

Blant annet kan begrepet «gjenta» tas i bruk istedenfor «doble», under matlaging, da disse begrepene innebærer det samme.

I en av programmeringsøktene var det fokus på å lage og følge trinnvise instruksjoner, samt de nevnte begrepene, gjennom undervisningsmateriellet «Hei Ruby». I starten av økten forteller læreren dette og utdyper ytterligere hva oppgaven går ut på. Læreren skriver opp begrepene på tavla og klassen har en samtale om hva de ulike begrepene innebærer. Etter det skriver læreren opp en sekvens med dansetrinn i en trinnvis rekkefølge, samt hvor mange ganger den sekvensen skal gjentas. Læreren legger til to sekvenser til, med antall gjentakelser, og til slutt hvor mange ganger hele programmet (alle sekvensene til sammen) skal gjentas. Elevene øver på dansen først uten musikk, etter gitte betingelser for takt som læreren har informert om, før de danser med musikk. Etter det er det elevenes tur til å lage egne dansebevegelser i grupper, velge antall gjentakelser og eventuelle betingelser. Læreren forteller i intervjuet om sammenhengen mellom begrepene og dans:

> «*Det der å prøve å følge et lagd mønster først, men så å lage et sjøl etterpå, og så si at; Alle delene dere har lagd, kan jo være små program i seg sjøl, og vi kan også sette dem sammen sånn at, hver er en sekvens i et stort program, og se den sammenhengen da*».

Læreren sier videre at det er lettere å forstå og huske begrepene når det kan relateres til noe:

> «[Å] *relatere det til den dansen da, de bevegelsene der; Det var en sekvens og hvis jeg gjorde det flere ganger så hadde jeg gjentatt den. De sekvensene vi lagde, satte [vi] sammen til ett program. Da blir det litt lettere å forstå da*».

Læreren viser til at det finnes forskjellige typer programmeringsspråk og gjennom årene elevene har holdt på med programmering, har de fått kjennskap til samtlige. I løpet av de analoge programmeringsøktene under datainnsamlingen, har ikke elevene fått beskjed om hvilket programmeringsspråk de skal bruke. Dermed tar elevene i bruk ulike programmeringsspråk til samme oppgave. Læreren forteller i intervjuet:

> «*Jeg sa jo at jeg ikke ville at de skulle låse seg til ett programmeringsspråk, men at dere skal få lov å velge. Jeg kunne jo ha sagt at dere skal bruke piler eller*

dere skal tegne, men da er det en diskusjon de imellom; hvordan er det jeg forklarer det best?».

Læreren sier at elevene gjennom code.org og Kodetimen, hadde fått kjennskap til ett programmeringsspråk ganske tidlig. Da de skulle begynne med analog programmering ved hjelp av Blue-Bot, var programmeringsspråket likt, men med noen unntakelser. Læreren forteller om en observasjon av elevene da de jobbet med Blue-Bot for første gang:

«*Vi fikk disse Blue-Botene og så begynte de å tenke akkurat som de tenkte på code.org. […] En gruppe sitter og tenker; Dette har vi kontroll på. Så kikker de på roboten, og så sier de; Hva er det som skjer? Den gjør ikke som vi sier! Da oppdaget de at det var forskjell på å trykke høyre pil på Blue-Boten og det å gjøre det på code.org. Så selv om kommandoen var det samme, så gjorde ikke robotene det samme. […] De ligner på hverandre, men det er ikke helt likt*».

I forbindelse med eksempelet over, forteller læreren at elevene måtte samarbeide og finne ut av problemet sammen. Læreren veiledet elevene og de utforsket hvordan Blue-Bot fungerte, og hva de skulle trykke på for å komme seg videre i programmeringen.

4.1.2 Prøve og feile (feilsøking)

Som en del av analog programmering, har læreren stort fokus på elementet feilsøking, innenfor algoritmisk tenkning. En observasjon som ofte gikk igjen i øktene, var at læreren trakk frem feilsøking som viktig for elevenes læring. Læreren sier i intervjuet:

«*La de feile. Det er så viktig å tenke, at det er ikke noe farlig som skjer. Hva er det verste som kan skje da? At vi må prøve en gang til*».

I et av intervjuene poengterer læreren at elevene er nødt til å feile hvis de skal lære noe i programmering. Hvis de ikke feiler, vil de aldri helt skjønne hva som må rettes opp for å kunne lære av feilene sine. Et eksempel som trekkes frem i intervjuet, som jeg også observerte i økten, var en samtale mellom to elever under arbeidet med en programmeringsoppgave. Læreren forteller:

«*Det jo litt uenigheter og så måtte de prøve også fant de ut; Ja, men det var jo du som hadde rett! Det er jo veldig gøy, og det ligger ikke noe surhet i det, men*

det er bare sånn; Da skjønte jeg også det du egentlig prøvde å si. At det var sånn det ble, men jeg måtte liksom se det før jeg kunne tro helt på det».

Læreren uttrykker glede når hen ser og hører at elevene reflekterer rundt deres egen problemløsning og feilsøking:

«*Da blir jeg glad. Da føler man at man har fått til noe*».

Læreren poengterer at det er hensiktsmessig å bruke begrepet *feilsøking* konsekvent, når det snakkes om å prøve og feile:

«*[…] Det er viktig at man bruker det ordet også. At det er det som skjer når vi skriver analogt. Det er en feil som skjer i programmeringa vår, som gjør at vi må finne ut hvor feilen er og hva kan man gjøre annerledes for at vi skal få det til. […] Det å prøve å bruke begrepene som allikevel er i programmeringa er lurt, allerede fra starten, sånn at de er kjent med de begrepene*».

Ved å oppmuntre til bruk av feilsøking i analog programmering, mener læreren begrepet vil være gjenkjennbart for elevene når de går over til digital programmering. Det gjelder både å anvende begrepet i undervisningen og feilsøke i arbeid med programmeringsoppgaver.

4.1.3 Programmeringsverktøy

I programmeringsundervisningen legger læreren opp til bruk av ulike verktøy som bidrar til den generelle forståelsen for programmering. De verktøyene var LEGO-klosser, puslespill og «Hei Ruby». Det disse verktøyene har til felles er at de er analoge, de forbindes ofte med lek og spill og enten er eller tar i bruk kjente, hverdagslige objekter som passer til elevenes alderstrinn. Ved å ta i bruk verktøy som LEGO og puslespill, som elevene allerede har et forhold til, blir fokuset mer på forståelsen og målene som ligger til grunn for den analoge programmeringen, og ikke verktøyet i seg selv. I et av intervjuene sier læreren:

«*Det [er] mye mindre forstyrrende elementer, slik at du får mer fokus på den [analoge] tenkninga når du jobber*».

Læreren tror dette har mye å si for forståelsen og at det er svært overførbart til det digitale.

Inntrykket mitt som observatør i disse øktene, er at det er fint å se ulike måter å lære om trinnvise instruksjoner på, ved hjelp av verktøy som er tilgjengelig i klasserommet. I tillegg kan lek prege undervisningen, fordi elevene tar i bruk verktøy og metoder som opprinnelig er ment for lek og spill. Dette viser læreren også til:

> «*Det er viktig å […] leke litt. Bruk puslespill, eller bruk elementer som du kjenner til. LEGO og tegne*.»

Læreren mener det er en fordel å ta i bruk verktøy som er gjenkjennbare for elevene, da de ses på som trygge ting.

For å begynne med programmering er det ikke nødvendigvis viktig med tilgang på «Hei-Ruby»-boka eller mye utstyr. Læreren sier at paralleller i stedet kan trekkes til hverdagslivet. Læreren kan for eksempel be elevene om hjelp på kjøkkenet, som gjør at de må jobbe ut ifra en oppskrift. Da må de finne ut hvordan de skal gå frem, hva som skal blandes og lignende. Læreren sier i intervjuet:

> «[…] D*u kan relatere det til noe ungene skjønner. Når du da får inn mer komplekse ting, så [kan du si]; Husker dere da vi var på kjøkken og jobba med det? Hva gjorde vi da? [At man] greier å se koblingene inni der da, ved å ha forståelsen og det dagligdagse i bunn*».

Som læreren sier kan elevene hjelpes til å forstå for eksempel begreper eller arbeidsmetoder, ved å relatere det til matlagingen, da elevene innehar en grunnleggende forståelse.

4.1.4 Den grunnleggende forståelsen for programmering i skolen

Læreren trekker frem viktigheten av at elevene forstår hvordan programmering foregår og hva som skjer når de programmerer. Læreren forteller i intervjuene at de har brukt en god del tid på analog programmering. Grunnen er at elevene skal forstå grunnstammen til programmeringen, altså å forstå hva som egentlig gjørs når de programmerer, før de går over til å programmere digitalt. Selv om elevene har gått over til digital programmering, poengteres det at den analoge programmeringen ikke må gis slipp på:

> «[Det er viktig] a*t du ikke gir slipp på'n. Hvis du skal introdusere nye ting, så kan du ha en økt analogt, og så gjøre mye av det samme digitalt etterpå. At man får med seg forståelsen hele veien*».

Det kommer tydelig frem at læreren ønsker at elevene skal kunne se sammenhengen mellom det som gjøres i analog programmering og i digital programmering. Dette ligger til grunn for elevenes programmeringsundervisning.

Øktene med LEGO, puslespill og «Hei Ruby» har til felles å programmere «en datamaskin» til å utføre en oppgave, men at datamaskinen i stedet er elevene. Læreren mener det er en fordel å jobbe grundig med analog programmering først for å få den digitale forståelsen:

> «*Jeg synes det er viktig å ha med forståelsen. Jeg tror kanskje du får mindre forstyrrelser på forståelsen når du jobber uten data enn med data, og så er det lettere å gå da fra å ha jobba med det analogt, og vet hva det er også kan du heller si; Forrige gang så jobba vi med det, og nå skal vi gjøre det bare med maskiner. Så du får dratt med sammenhengen*».

Med «Hei Ruby» får elevene et inntrykk av hvordan en datamaskin er og forbinder datamaskinen med en sta jente som trenger tydelige og detaljerte instruksjoner. Et eksempel fra mine observasjoner, handler om når elevene skal «spille ut» sine instruksjoner under frokostoppgaven med Ruby. Elevene har skrevet ned at Ruby åpner kjøleskapet, tar ut melken og setter den på bordet. I det instruksjonene utføres i praksis, oppdager elevene at de har glemt å skrive ned at Ruby også må lukke igjen kjøleskapsdøren. Læreren viser til dette i intervjuet:

> «E*n hverdagslig ting, som alle har en erfaring med, er å spise frokost og hva som må til, men vi gjør jo veldig mye ting på automatikk. [Noe] som er en selvfølgelighet for oss, [men] som ikke en datamaskin ville ha skjønt. Så det å skjønne den sammenhengen der, synes jeg er grunnleggende*».

I eksempelet og sitatet over, kommer det frem at læreren synes det er grunnleggende at elevene forstår hvordan en datamaskin «tenker». Samtidig at når elevene «spiller ut» instruksjonene sine i praksis, blir det veldig tydelig for dem hva som må endres for at en «datamaskin» skal gjøre det som forventes.

Læreren trekker frem viktigheten av å bruke mye tid på forståelsen i programmeringen først:

> «*[Vi] bruker mye tid på å forståelsen bak og ikke bare programmering og programmeringsdelen, men at man etablerer en felles forståelse for hva er det vi gjør da, og hva er det vi skal lære nå*».

Hvis elevene har den grunnleggende forståelsen for programmeringen, vil det bli lettere å bygge videre på programmeringen med for eksempel andre programmeringsbegreper og -språk. Læreren poengterer at det er viktig å begynne litt rolig med det analoge. Da får elevene lov til å ha det gøy og teste ut mange av de samme tingene som senere kan gjenkjennes ved programmering av roboter. Dette kan også bygges videre på.

4.2 Programmering som en arbeidsmetode i fag

Det andre hovedfunnet er bruken av programmering som en arbeidsmetode i fag. Læreren forteller at stort sett alle fag kan trekkes inn i programmeringen og at programmering kan ses på som en arbeidsmetode. I undervisningen og i intervjuene, eksemplifiserer læreren hvordan programmering kan trekkes inn i forskjellige fag, og motsatt. Funnet deles opp i fagkategorier og består av observasjoner fra programmeringsundervisningen og lærerens begrunnelser.

Matematikk

I økten hvor elevene skulle lage trinnvise instruksjoner til deres egenproduserte LEGO-figur, måtte de ta i bruk kunnskaper om hvordan LEGO bygges, ha en viss kreativ tankegang og kunne skrive ned en stegvis prosess, fra start til slutt. Instruksjonene burde blant annet innebære hvilken farge og form LEGO-klossene hadde, samt antall «knotter». Samtidig som elevene fikk leke med LEGO under byggingen av figurene, lærte de om målet innenfor matematikk om trinnvise instruksjoner. Dette nevnte læreren under økten og i intervjuet:

> «J*eg [sa] også i starten, at målet for timen var at vi skal jobbe med instruksjoner, og at det er et mål i matematikken. Det går nok litt forbi, men noen fanger jo opp det og hvilket fag som er i fokus nå*».

Tilknytningen til matematikkfaget forteller læreren også om etter økten med puslespill. Etter hvert som elevene hadde prøvd seg frem og undersøkt ulike metoder, fant de ut hvilke instruksjoner som var mest effektive. Læreren forteller om dette i intervjuet:

> «*Det å gi litt mer beskrivelser og begynne å; Ja, men, den hjørnebrikka har jo to flate sider. Begynte å se litt flere egenskaper som kunne skille brikkene litt fra hverandre og sånn. Det er veldig spennende å se progresjonen i det*».

I tillegg forteller læreren om enkeltelevers bruk av elementer fra matematikkfaget i forbindelse med pusleoppgaven. Elevene benyttet kunnskaper om rotasjon, vinkler og grader i instruksjonene. Læreren uttrykker positivitet rundt at gruppen fant en felles forståelse og bruker matematiske begreper som alle gruppemedlemmene er kjent med. Dette er noe som kan bygges videre på og endre fokusområde til, neste gang elevene skal ha samme programmeringsoppgave. Læreren sier det for eksempel kan settes betingelser for bruk av matematiske begreper i instruksjonene. På stasjonen med BOLT tok elevene i bruk de samme matematiske kunnskapene om rotasjon, vinkler og grader, da de skulle programmere BOLT til å «lage» et kvadrat gjennom blokkprogrammering. Læreren viser til et eksempel hvor elevene hadde lagt inn «spinn» som kommando og skulle prøve ut kommandoene på seg selv først. Elevene kom frem til at spinn betydde å snu 360 grader fort i ubestemt tid. Det ga de en pekepinn på hva som måtte gjøres for å fikse opp i kommandoen.

Da elevene skulle gå over til digital programmering på «koderommet», hadde læreren lagt opp til stasjonsundervisning med blant annet Blue-Bot på matter. I Blue-Bot-oppgaven med regnestykker, poengterer læreren at oppgaven ikke gir stor utfordring for elevene matematisk sett, men at målet var å få oppleve mestring med regnestykkene og programmeringen. På stasjonen med Osmo Coding øvde elevene på å programmere ved hjelp av blokkprogrammering og å se likheter mellom ulike kommandoer og begreper. Et eksempel er at en gjentablokk og 2x (ganger)-blokker er det samme. Læreren forteller at når elevene kommer lenger i «spillet», blir de utfordret til å ta i bruk slike blokker for å gi riktig kommando til Osmo.

I flere av intervjuene trekker læreren frem algoritmisk tenkning som en viktig del av programmeringsundervisningen:

> «D*e tankeprosessene rundt den [algoritmiske tenkeren] har ligget til grunn for at jeg kan forsvare det, uten at det nødvendigvis da finnes noe i læreplanen, og at en kan tenke da at dette er nyttig for flere fag*».

I tillegg poengterer læreren sammenhengen mellom algoritmisk tenkning og problemløsning i matematikk. Når programmeringsøktene planlegges, forteller læreren at det er fornuftig at det forankres i noe. Læreren trekker derfor frem fokuset på den algoritmiske tenkeren i programmeringsøktene, i tillegg til et fagmål.

Norsk

I et intervju viser læreren til et eksempel på hvordan sammenhengen mellom programmering og norskfaget kan ses:

> «N*år de [jobber med] leseforståelse, leser de og så svarer de på spørsmål som er trinnvise i forhold til en tekst*».

Læreren sier at selv enkeltelever har kommentert at de ser sammenhengen mellom programmering og leseforståelsesoppgavene. At elevene begynner å se koblingen mellom programmering og fag, er noe læreren påpeker som veldig gøy.

Selv om rimord ikke er et mål for 4. trinn, forteller læreren om flere elever som følte på usikkerhet i forbindelse med rimordstasjonen med Blue-bot. Likevel poengterer læreren at det er en fin måte å repetere på. Samtidig som det ikke blir «flaut» å gjøre feil i en slik situasjon, hvor fokuset er på programmeringen. Læreren begrunner det med at «feilene» blir mer synlig dersom det hadde vært en norskøkt med fokus på rimord.

I forbindelse med blant annet historiefortelling, forteller læreren om bruk av Bloxels. Læreren viser til at i Bloxels kan elevene lage figurer og rekvisitter samt bygge flere scener inn i sitt eget spill, som en slags fortelling. Eksempelvis nevner læreren at elevene kan lage Bukkene bruse inn i en spillversjon.

Mat og helse

Læreren trekker frem et eksempel på hvordan programmering kan trekkes inn i matlaging med fokus på matematikk:

> «V*i skal prøve å lage vaffelrøre neste uke og da [tenkte jeg først], skal vi bare lage vaffelrøre, også skal de doble og sånt kanskje, men så tenkte jeg; Hvorfor kan vi ikke ta programmering da? De kan instruere hverandre. For eksempel at; Du henter tre egg, også gjenta det. Da dobler du uten at du trenger å regne ut doblingen*».

Ifølge læreren er det svært relevant å trekke paralleller mellom programmering og andre fag. I tillegg ses det på som en fordel å kunne følge en oppskrift, noe som er svært hverdagsnært og noe elevene har et forhold til.

Kroppsøving og musikk

Under programmering av dans med Ruby var bevegelser som hopp, klapp, spinn og tramp eksempler på dansetrinn som sekvensene innebar. Enkelte bevegelser hadde også ulikt takt og tempo, noe som kan trekkes inn i musikkfaget. I tillegg forteller læreren at en av elevenes dans skal benyttes i kroppsøvingstimen:

> «V*i kunne like godt brukt det i musikk, fordi vi dansa til takt. En gruppe har jo kommet på at vi skal gjenta det her helt til sangen er ferdig, og det er betingelsen vår, og da tenker jeg at da kan vi jo bruke det som en oppvarming i gymtimen. Altså du kan jo putte alle type fag inn*».

På den måten trekkes programmering inn i musikkfaget og kroppsøvingsfaget samtidig som elevene øver på å følge instruksjoner. I tillegg blir oppvarmingen i gymtimen mer elevstyrt, noe læreren mener er positivt.

4.3 Samarbeid

Det tredje hovedfunnet jeg ønsker å trekke frem, er bruken av samarbeid i alle observasjonsøktene i datainnsamlingsperioden. Elevene arbeidet aldri alene med programmeringsoppgaver. De arbeidet alltid enten i par eller i grupper. Under vil jeg trekke frem observasjoner og lærerens begrunnelser for hvorfor elevene alltid arbeidet sammen i programmeringsøktene.

4.3.1 Par eller grupper

I samtlige økter starter læreren med å fortelle elevene om viktigheten av samarbeid og at det er et mål for økten. Samarbeid nevnes også i intervjuene:

> «*Jeg bruker mye samarbeid, enten om det er to og to eller i gruppe. At de på en måte drar nytte av hverandres kunnskap [og] at de ikke blir sittende alene med problemene. [At de] forklarer hva de tenker og sånt. Samarbeidslæring*».

Ifølge læreren varieres det om gruppene og parene er tilfeldig valgt eller ikke. I klasserommet sitter elevene i grupper som rulleres ukentlig. Læreren poengterer

viktigheten av at alle skal kunne samarbeide med alle, uavhengig av kjønn og faglige eller sosiale ferdigheter. I intervjuet begrunner læreren hvorfor elevene alltid arbeider sammen:

> «J*eg [synes] ofte at det er lurt å være to når man programmerer. Når du sier et problem høyt og har en å begynne en setning [med], så kan den andre tenke videre. [Da] er det lettere å løse et problem enn om du sitter bare i din egen tanke. Også når du jobber digitalt, sikrer [man] større læring enn om du sitter alene og trykker, fordi du får en diskusjon kontra at kanskje noen bare sitter og prøver og feiler hele veien, også er det er ikke sikkert de helt vet hva de driver med. Men at [læreren] heller går rundt og hører på diskusjonen og ser alltid hva som foregår i hodet på dem*».

Læreren poengterer flere ganger at «t*o tenker bedre enn en*». I tillegg varieres det hvem som arbeider sammen:

> «J*eg [prøver] å fordele de som jeg vet kan litt, sammen med de som jeg vet trenger litt drahjelp. At de kan lære litt av hverandre*».

Likevel forteller læreren at det er hensiktsmessig å se an programmeringsaktiviteten elevene skal utføre. Noen ganger passer det best å være to og andre ganger er det en fordel å være flere. Ved å velge grupper eller par mer tilfeldig, mener læreren at hen kan bli positivt overrasket over hvem som samarbeider godt sammen:

> «N*oen ganger så oppdager du [gruppesammensetninger] som går mye bedre enn det du hadde trodd, og kanskje ikke hadde turt å satse på selv. Så det er, kanskje for alle lærere egentlig, å tørre å prøve å være litt mer tilfeldig noen ganger. Vi liker jo å ha veldig styringa*».

Læreren trekker også frem samarbeid som en viktig del av elevenes programmeringsferdigheter:

> «J*eg tenker at hvis du skal være god til å programmere, så er du god til å dra med andre i gruppearbeid, du er god til å lytte til andres meninger*».

I tillegg viser læreren til at fordelen ved å samarbeide i programmering, er at elevene har noen å sparre med og kunne være sammen om det.

4.3.2 Klassemiljø

I flere av intervjuene forteller læreren at elevene er veldig trygge på hverandre og at de har et godt klassemiljø. Grunnen for dette mener læreren i stor grad kan være fordi de begynte med programmering allerede i 1. klasse:

> «*Jeg [ser] jo det at da vi begynte med det i første klasse, [har det hatt] ganske positivt effekt på det å være trygg på å komme med ideer, det å være trygg på at, okei, det ble kanskje ikke sånn vi har tenkt, men vi har nå prøvd da kan vi prøve noe annet og se om det fungerer bedre. At det er rom for det da. Så det skaper en høyere takhøyde i klasserommet. Jeg tror det har vært med på å bidra til den tryggheten som er her, selv om det er mange ulike utfordringer, så er de jo trygge på hverandre*».

Læreren forteller at inkludering av elever med ulike utfordringer, som kanskje ikke er i stand til å for eksempel fysisk trykke på knappene til en Blue-Bot, kan sikres i større grad ved å bruke Blue-Bot appen som et supplement. Dermed kan elever med utfordringer programmere roboten på lik linje med andre elevene, men via appen.

Læreren viser også til at klassen er god til å oppmuntre hverandre når noen gjør feil:

> «[De] *lar hverandre slippe til og selvfølgelig er det jo unntak, men sånn generelt sett så synes jeg de er veldig flinke, både tålmodige med hverandre og prøver å gi hverandre litt sånn småhint og hjelpe hverandre på veien. Det synes jeg er gøy å se. De vil jo at de andre skal lykkes og noen får jo noen veldig gode hjelpere og [spør]; Hva hvis du tenker sånn? Det er så godt å se da, at de sier ikke; Ja, men det er sånn. [De] prøver å hjelpe dem frem til det. Det er veldig fint*».

Ifølge læreren er elevene trygge på hverandre, noe som ses på som en stor fordel når de jobber med programmering. I tillegg har elevene blitt trygge gjennom programmeringen. Likevel trekkes det frem at det antakeligvis er forskjell på å begynne med programmering med fokus på samarbeid allerede fra 1. klasse av og det å begynne senere, men at det selvfølgelig er forskjeller fra klasse til klasse. I tillegg observerte jeg at elevene hadde stort engasjement for programmering og ønsket å arbeide sammen for å oppnå et felles mål.

4.3.3 Lærerrollen

I og med at elevene arbeider med programmeringsoppgaver i par og har mulighet til å diskutere med hverandre, er lærerens rolle å være veileder. Det gjelder uavhengig om elevene arbeider analogt eller digitalt:

> «L*ærerrollen i programmering er [å være] veileder. Det er ikke sånn at du skal lære dem alt, men du skal gi dem noen verktøy, også må du [kanskje] inn å [se]; Okei, her trengs det mer verktøy og gi litt hint her og litt hint der, også få de på tankegangen og få satt i gang. Ikke gi svarene nødvendigvis, men [spørre]; Hva tenker du om det?*».

Læreren forteller at lærerrollen i programmering i hovedsak er å hjelpe elevene til å finne ut svaret på det de eventuelt lurer på, uten å direkte gi de fasiten:

> «*Prøve å veilede, stille noen spørsmål som gjør at de må reflektere over hva er det vi driver med nå skal vi hva trenger vi?*».

Dette gjelder også når nye oppgaver eller programmeringsverktøy skal introduseres:

> «*Jeg [tror det] er viktig som lærerrolle i det her, at [man gir] de rommene til å prøve. Du skal ikke være sånn; Vær så god, bare start. Det må være noen rammer. Det skal være betingelser som du setter, eller noe informasjon som man må ha og så må ikke alle gå samme veien. [Si]; Her er du inne på noe spennende. Hvor skal vi? Hvorfor valgte dere å gjøre sånn på den? Det å få dem til å reflektere hvorfor de tar de valgene de tar*».

Som lærer i programmering, kan elevene veiledes til å finne ut hvordan de kan komme seg videre og hva som trengs for å kunne gjøre det. Noe læreren forteller at kan sammenfattes med å prøve og feile:

> «*Få de til å reflektere og tenke når de står fast. Hvordan kan vi komme oss videre? Hva er det vi trenger [og] hva mangler vi for å kunne gå videre? Det er veiledningen videre, hvis de står fast og la de få lov å prøve litt. De må få litt informasjon til å komme i gang, men at de skal få lov å skjønne at her trenger [vi] mer for å få [det]til, og så søke den hjelpen*».

Læreren kan også alltid oppmuntre elevene til å finne flere løsninger på problemet:

«*Også har vi jobba litt med [det at]; Vi fikk det til med en gang, men hvordan […] kunne [dere] gjort det annerledes? Også få en diskusjon på det. Så må det ikke være så mye annerledes oppgave for de som «catcher» det med en gang, men da kan du få i gang tankevirksomheten på hvordan kunne dere [ellers] ha gjort det? […] Så får man litt diskusjon[er]. Det er to løsninger til samme svar, men da kommer det an på hva skulle du gjøre? Skulle du finne raskeste vei eller skulle du finne den treigeste veien eller var det betingelser for hvilken vei det skulle? Hvor mange løsningsforslag kan du finne? Du [kan] på en måte variere det såpass inn i en gruppe at det blir veldig tilpasset ut ifra hvor de enkelte er*».

Samme oppgave kan tilpasses ut ifra hvor eleven er i prosessen. Læreren nevner i intervjuet at hvis en gruppe elever har vært raske med å løse en oppgave, kan de utfordres litt samtidig som essensen i oppgaven beholdes.

5 Drøfting

Før prosjektets startpunkt ønsket jeg å lære mer om og undersøke hvordan programmering tas i bruk i undervisningen til de yngste elvene på barneskolen. Temaet har fasinert meg i lengre tid og er i tillegg svært relevant for dagens og fremtidens samfunn. Dette trekkes inn i problemstillingen min:

Problemstilling:

Hva kjennetegner undervisning i programmering i begynneropplæring?

Ved å ta i bruk en kvalitativt casestudie har jeg hatt mulighet til å undersøke i dybden hvordan en lærer utfører og begrunner sine handlinger, i feltets kjerne, klasserommet. Gjennom deltakende observasjon og semistrukturerte intervjuer har jeg tillært meg kunnskaper om hvordan en lærer, med både faglig grunnlag og praktiske erfaringer, anvender programmering i undervisningen. I resultatkapittelet har jeg trukket frem elementer som er viktige i prosessen med å undersøke hvordan problemstillingen kan besvares, gjennom forskningsspørsmålene og metodene som er tatt i bruk. I det følgende vil jeg ta for meg de mest sentrale funnene fra resultatkapittelet og trekke paralleller til tidligere forskning og teori som er presentert. Disse elementene er gjort om til mer overordnede temaer, som drøftingen tar utgangspunkt i:

- Samarbeid
- Programmeringsverktøy
- Lærerrollen

De overordnede temaene som vil drøftes er i tråd med problemstillingen, i den forstand at de kan ses på som kjennetegn ved programmeringsundervisningen på 1. til 4. trinn. Gjennom metodene som er benyttet i studien, er det disse tre temaene som er sentrale i datamaterialet og som samsvarer med tidligere forskning på feltet. Dermed er det relevant å drøfte dette videre for å kunne svare på problemstillingen.

5.1 Samarbeid

Samarbeid eller programmering i par anses som en sentral del av programmeringsundervisningen (Zhong et al., 2016; Otterborn et al., 2019; Kjällander et al., 2021; Iskrenovic-Momcilovic, 2019) Dette er noe som også er svært fremtredende i datamaterialet da elevene alltid arbeider sammen i programmeringsundervisningen.

Læreren trekker jevnlig frem at «*to tenker bedre enn en*» og mener at samarbeid er en viktig del av elevenes programmeringsferdigheter. Å være god til å programmere definerer læreren med å være gode til å ta med andre i gruppearbeid, i tillegg til å lytte til andres meninger og ideer. Det reflekteres rundt at å programmere sammen med andre, kan gi bedre motivasjon for å lære ny kunnskap og mer effektiv læring innenfor programmering (Iskrenovic-Momcilovic, 2019). I tillegg trekkes det frem at å samarbeide i programmeringsaktiviteter bidrar til at elevene får en god læringspartner, samt at de øver på å sosialisere med hverandre (Zhong et al., 2016). I datamaterialet er det synlig at elevene ønsker å samarbeide i programmeringsundervisningen, da de har noen å diskutere og reflektere med, i tillegg har de et ønske om at alle skal oppleve mestring i arbeidet.

Ifølge den sosiokulturelle læringsteorien blir kunnskap konstruert i en kontekst gjennom samhandling med andre (Dysthe, 2001b, s. 42). Dette kan sammenlignes med programmeringsundervisningen, da elevene lærer å løse ulike problemløsningsoppgaver ved å samarbeide. På den måten får de også tillært seg kunnskaper og ferdigheter som kan tas med i andre fag eller hverdagslige situasjoner, noe også læreren uttrykker. Samarbeid, samhandling og interaksjoner blir innenfor et sosiokulturelt perspektiv sett på som grunnleggende for læring og læringsmiljøet (Dysthe, 2001b, s. 42). Funn fra datamaterialet viser at læreren ønsker at elevene skal samarbeide for å både styrke læringsmiljøet og fremme læring i programmering og problemløsning. Ifølge læreren har en stor del av elevenes trygghet innad i klassen, skyltes at de begynte med programmering i 1. klasse. Læreren understreker at elevene fra en tidlig alder har lært hvordan et godt klassemiljø kan opprettholdes. Det kan gjøres ved å fremme positivitet rundt det å tørre å komme med ideer, prøve og feile og arbeide sammen mot et felles mål gjennom programmeringsundervisningen. Imidlertid er ikke alle klasser like og det kan være elever som bidrar til at problemer oppstår for klassemiljøets utvikling. Klassen som ble forsket på i denne studien kan, som nevnt tidligere, ses på som et «prakteksempel». I intervjuene forteller læreren at elevene er trygge på hverandre og har et godt klassemiljø, noe som er utmerket. Poenget er at alle klasser nødvendigvis ikke har det som utgangspunkt, noe som kan ha konsekvenser for samarbeidet mellom elevene. Igjen kan det påvirke programmeringsaktivitetenes hensikt og elevenes læring. Dysthe (2001a) forteller om viktigheten rundt å stille spørsmål ved fungeringen av samspillet i fellesskapet og om samspillet kan ses på som læringsfremmende eller læringshemmende (Dysthe, 2001a, s. 11). I dette prosjektet vil jeg si at fellesskapet er

læringsfremmende, da læreren trekker frem en klar sammenheng mellom klassemiljø og programmeringsundervisningen i positiv forstand. I tillegg observerte jeg at elevene heiet på hverandre og ønsket at medelevene skulle lykkes i arbeidet. Det er imidlertid ikke slik at klasser med mindre godt klassemiljø, ikke kan yte godt av programmeringsundervisning og -aktiviteter eller at klassemiljø ikke kan arbeides med. Tvert imot, programmering kan for eksempel tas i bruk for å arbeide med klassemiljøet og samarbeidet, og muligens oppnå gode resultater deretter.

5.1.1 Gruppesammensetninger

De mest hensiktsmessige måtene å inndele elevpar på i programmeringsundervisningen diskuteres i forskning (Zhong et al., 2016; Iskrenovic-Momcilovic, 2019). Funn fra datamaterialet viser at elevene alltid arbeidet i par eller grupper i programmeringsundervisningen og at en viktig forutsetning for læreren var at alle kunne samarbeide med hverandre. Læreren forteller at gruppene ukentlig ble rullert og enten tilfeldig fordelt eller fordelt etter hvem som «passer» sammen. Likevel poengteres det at hvis læreren selv tør å velge par og grupper mer tilfeldig, kan hen bli positivt overrasket over hvem som samarbeider godt sammen. En faktor i læring av algoritmisk tenkning og en måte å forminske kjønnsskiller på er *pair-programming*. I pair-programming arbeider to og to elever på én felles digital enhet (Zhong et al., 2016). I datamaterialet ses tydeligst bruk av pair-programming under aktiviteten med Osmo Coding, hvor elevene arbeidet to og to sammen på én iPad. Elevene måtte samarbeide for å løse oppgavene ved hjelp av kodebrikkene foran seg, i tillegg til å lytte til hverandres ideer. På den måten kommuniserer elevene jevnlig med hverandre, noe som er et sentralt aspekt i pair-programming (Zhong et al., 2016).

Når gruppene eller parene blir fordelt basert på samarbeidsevne eller kunnskapsnivå, er en faktor at elevene kan hjelpe hverandre med ulike ting og lære av hverandre. Funn fra datamaterialet viser anvendelse av *peer scaffolding* (Belland, 2017, s. 25). Eksempelvis forteller læreren at elevene kan opptre som drahjelp for hverandre. Likevel er det læreren som står for mest av veiledningen, selv om elevene lærer mye av å hjelpe hverandre og tenke sammen. Slik som læreren forteller, kan elevene bygge på hverandres tanker. På den måten vil det være enklere å løse problemer sammen, enn om de hadde sittet alene med sin egen tanke og muligens ikke kommet videre. Ved stasjonsundervisningen med digitale programmeringsverktøy, var én av stasjonene ny for elevene. Dermed var det elevene på denne stasjonen som trengte mest hjelp for å

kunne få et utbytte av stasjonen. De andre stasjonene ble derfor mer selvgående, og elevene fikk i mindre grad veiledning dersom de sto fast. Elevene som arbeidet i par var delt opp på en måte som la føring for at de kunne veilede hverandre. Til tross for at elevene kan veilede hverandre på enkelte aspekter, er lærerens kunnskaper innenfor feltet, til en viss grad, uerstattelig. Dermed kan ikke peer scaffolding være den eneste formen for veiledning (Belland, 2017, s. 26) i løpet av en programmeringsøkt.

5.1.2 Feilsøking

I forskning trekkes feilsøking frem som en sentral del av analog programmering (Nouri et al., 2019; Chibas et al., 2018; Heikkilä & Mannila, 2018), da elever lærer mye av å gjøre feil og rette de opp igjen (Nouri et al., 2019). Dette er noe læreren også nevner og har mye fokus på i programmeringsundervisningen. Læreren poengterer at det er viktig at elevene prøver seg frem og at det ikke er farlig å feile, da det verste som kan skje er at de må prøve igjen. I tillegg forteller læreren at dersom elevene ikke feiler, vil de heller aldri skjønne hva som må rettes opp for å kunne lære av feilene sine. Dette er en sentral del av programmeringen. Haraldsrud et al. (2020) informerer om viktigheten av å lære elevene å feilsøke tidlig, da det ofte finnes flere løsninger på et problem (s. 165). Ved å lære elevene å utforske sammen, understrekes det at de kan komme langt i programmeringsprosessen. I tillegg trekker Haraldsrud et al. (2020) frem, i likhet med læreren i datamaterialet mitt, at det er hensiktsmessig å tenke over at alle gjør feil og hvis elevene ikke finner løsningen med en gang, er det ingen skam å prøve igjen (Haraldsrud et al., 2020, s. 165). Elevene viser at de er nysgjerrige, ønsker å løse oppgaven og samarbeider godt for å komme frem til en mulig løsning, noe læreren setter høyt. I tillegg vises det i observasjonene at elevene hverken gir opp eller blir irriterte dersom det ønskede resultatet ikke oppnås med en gang. Tvert imot, blir de mer motiverte til å løse oppgaven sammen. Dette støttes av teori i forbindelse med programmering i par (Iskrenovic-Momcilovic, 2019).

I datamaterialet mitt ser jeg tydelig at feilsøking er en svært sentral del av både den analoge og den digitale programmeringsundervisningen. Det legges også opp til at elevene må prøve seg frem for å finne svar på oppgavene. Viktigheten av å utforske og finne løsninger på nye utfordringer i forbindelse med å kunne bidra i arbeid og samfunn, er noe skolen skal tilrettelegge for (NOU 2015:8). Læreren oppmuntrer også elevene til å finne nye, enklere og raskere metoder å løse problemet på, slik at de øver på å se muligheter til å forbedre svaret eller løsningen. Spesielt i oppgavene med puslespill og

BOLT er det tydelig at feilsøking er en sentral del. I puslespill-aktiviteten måtte elevene stadig undersøke nye måter å formulere seg på for å gi tydelige og detaljerte instruksjoner. Det viste seg å være utfordrende til å begynne med, men etter hvert som elevene kom mer inn i det og prøvde seg frem på ulike vis, mestret de aller fleste å fullføre puslespillet sammen. Her trekkes også kommunikasjon inn som en viktig faktor, da elevene måtte øve på å formulere seg på forståelig måte til eleven som puslet. Ifølge Säljö (2001) kan erfaringer sammenlignes og læres av, gjennom å tolke hendelser og dele erfaringer med hverandre gjennom språket (s. 35). Han poengterer at det er i samspill med andre at vi stadig utveksler og låner kunnskaper og ferdigheter (Säljö, 2001, s. 35). Da elevene prøvde ut ulike måter å kommunisere instruksjonene på, lærte de hva som fungerte og ikke. I tillegg spilte de på hverandres kommunikative strategier. Ved å gjøre dette og ta i bruk begreper og termer som elevene hadde en felles forståelse for, i tillegg til turtaking, mestret elevene i større grad å gi tydelige instruksjoner. Dermed klarte de til slutt å fullføre puslespillet. Det tyder dermed på at gjennom samspill og samhandling med andre i praktiske aktiviteter, kan elevene oppnå et felles mål. Gjennom LEGO- og puslespillaktiviteten, måtte elevene øve på å se likheter og ulikheter mellom objektene for å kunne gi riktig instruksjon til hverandre. Dette forteller Säljö (2001) om som et eksempel på hvordan vi kan lære gjennom å tolke hendelser i begrepslige termer. Han trekker frem et eksempel på dette ved å se på farge, form og vekt (Säljö, 2001, s. 35), noe som også ble tatt hensyn til i aktivitetene.

Under stasjonsaktiviteten med BOLT var målet i starten å undersøke hvordan roboten fungerer ved hjelp av kontrollerfunksjonen på iPad'en. Etter hvert ga læreren elevene en utfordring. I og med dette var en helt ny robot for elevene, i tillegg til blokkprogrammering som også var delvis nytt for dem, ble oppgaven basert på feilsøking. At læreren ikke trer inn i elevenes arbeid for tidlig, anses som viktig i arbeid med problemløsning og programmering (Heikkilä & Mannila, 2018). Dette kommer tydelig frem i datamaterialet, da læreren stadig oppmuntret elevene til å prøve seg frem og utforske først. Dermed fikk elevene mulighet til å fikle, undersøke og ta i bruk deres naturlige nysgjerrighet og kreativitet, samt tidligere kunnskaper fra matematikken for å forsøke å løse oppgaven. Dette er i tråd med Dewey sin teori om *learning by discovery*, hvor elevene lærer gjennom å oppdage. I tillegg poengteres det at programmering er en aktivitet som læres gjennom å gjøre (Mannila & Nordén, 2020, s. 71). I en problemløsningsoppgave kreves det at elevene stopper opp, reflekterer og tenker over hva de skal gjøre videre (Vaage, 2001, s. 145). Da kan problemløsningsmetoden, eller

undersøkelsesmodellen til Dewey, benyttes (Vaage, 2001, s. 146). Videre kan algoritmisk tenkning trekkes inn, da programmering også handler om å gi tilgang på gode verktøy for problemløsning (Sevik et al., 2016, s. 13). I og med algoritmisk tenkning innebærer flere elementer som også inngår i programmeringsundervisningen, er det noe som kommer tydelig frem gjennom hele datamaterialet mitt. Blant annet forteller læreren at den algoritmiske tenkeren ligger til grunn for programmeringsundervisningen og de ulike oppgavene som tas i bruk, da det er med på å forsvare bruken av programmering opp mot læreplanen.

5.2 Programmeringsverktøy

5.2.1 Analoge programmeringsverktøy

Artefakter, eller programmeringsverktøy tas i bruk for enten å støtte eller illustrere læring og undervisning (Heikkilä & Mannila, 2018). I tillegg er artefakter et sentralt begrep innenfor den sosiokulturelle læringsteorien (Dysthe & Igland, 2001, s. 77; Säljö, 2001, s. 83). I denne oppgaven presenteres funn der artefakter ble tatt i bruk i ulike programmeringsoppgaver. For eksempel benyttet læreren puslespill slik at elevene i samhandling la brikker trinn for trinn. I denne prosessen valgte læreren å benytte den sosiokulturelle måten å lære på, der elevene arbeidet sammen med bruk av artefakter ((Dysthe & Igland, 2001, s. 77; Säljö, 2001, s. 83). Videre valgte læreren å benytte artefakter som et redskap for å støtte undervisningen. På den måten fulgte læreren en undervisningsform som forskning viser til et godt læringsutbytte for elevene (Heikkilä & Mannila, 2018; Chibas et al., 2018; Aranda & Ferguson, 2018; Berg, 2021).

Lek og læring ligger til grunn for motiverende læring for småtrinnene (Vatne, 2006; Palm et al., 2020). Kompetansemål i matematikk etter 2. og 4. årstrinn, tar utgangspunkt i at elevene skal lære seg trinnvise instruksjoner i lek og spill (Kunnskapsdepartementet, 2019), noe oppgavene i datamaterialet kan relateres til. For eksempel forbindes LEGO med lek og er noe de aller fleste barn har et forhold til. Ved at læreren tok i bruk et kjent verktøy som LEGO, ble elevenes fokus ifølge læreren, på den analoge grunnleggende tenkningen i programmeringsoppgaven. Fordi elevene fikk lære programmering gjennom kjente artefakter som oppfattes som lek, tok elevene i bruk fantasien sin og var kreative samtidig som de utviklet problemløsningsferdigheter (Heikkilä & Mannila, 2018; Nouri et al., 2019).

Programmering i skolen trenger ifølge Akerbæk & Karlsen (2019) ingen annen begrunnelse enn at det gir mulighet for å utforske, uttrykke seg gjennom et digitalt medium og at det er en morsom, verdifull og utfordrende aktivitet i seg selv (Akerbæk & Karlsen, 2019, s. 158). I datamaterialet kommer dette tydelig frem som viktig for læreren, da oppgavene preges av utforskning og kreativitet. Læreren mener det er viktig at elevene får tid til å leke, samtidig som oppgavene og undervisningen kan relateres til hverdagslivet. For eksempel i frokostoppgaven med Ruby måtte elevene uttrykke seg ved hjelp av trinnvise instruksjoner og spille de i ut klasserommet, som en slags rollelek. Ut ifra elevenes oppfatning, vises det at elevene selv syntes oppgaven var morsom og på et vis utfordrende, uten at det var demotiverende. Ved å glemme å skrive at kjøleskapsdøra må lukkes igjen, ble det en situasjon som elevene lo av og kunne endre på til neste gang. Dette er noe som fremmes i forskning og i teori, hvor fokus på feilsøking og en innstilling om at å feile ikke er negativt, men en forutsetning for læring (Heikkilä & Mannila, 2018; Kjällander et al., 2021; Chibas et al., 2018; Nouri et al., 2019; Haraldsrud et al., 2020). I tillegg fikk elevene et mer visuelt blikk på det de hadde skrevet og programmerte hverandre som «roboter» eller datamaskiner basert på instruksjonene de sammen hadde skrevet. Gjennom arbeidet med denne oppgaven la læreren opp til å lære forståelse for hvordan programmering foregår i et perspektiv som elevene anser som hverdagslig og kjent. Ut ifra funn og tidligere forskning, anser jeg dette som hensikten med analog programmering. Elevene får et innblikk i at programmering ikke bare omfatter roboter og digital kompetanse, men en forståelse for hvordan en datamaskin «tenker» og hvordan problemløsning- og programmeringsferdigheter kan benyttes i hverdagslige situasjoner.

5.2.2 Digitale programmeringsverktøy

I datamaterialet mitt arbeider elevene en del med analog programmering før de går over til digital programmering. Ifølge læreren er det for at elevene skal lære om ulike programmeringsspråk og -begreper, hvordan en datamaskin «tenker» og øve seg på å løse ulike problemer på forskjellige måter, i samarbeid med andre. Det argumenteres for at benyttelse av visuell programmering med de yngre elevene er hensiktsmessig, da det fører til mer fokus på logikken og strukturen. I tillegg er det mer håndterlig for elevene (Palmér, 2017). All den kunnskapen de tillærer seg gjennom analog programmering kan de benytte seg av når de begynner med digital programmering. Eksempelvis tar elevene i bruk kunnskaper om ulike programmeringsspråk da de arbeidet med

programmeringsaktiviteten Osmo Coding. Da måtte de undersøke ulike måter å blant annet gjenta en sekvens på med stadig færre brikker. Dette er også i tråd med algoritmisk tenkning og elementet mønstergjenkjenning, hvor strukturer og likheter mellom ulike prosesser skal gjenkjennes og benyttes videre (Haraldsrud et al., 2020, s.189).

Da elevene tok i bruk Blue-Bot, lagde de sekvenser og etter hvert programmer for å løse ulike oppgaver. Enten om det er via Blue-Bot appen eller direkte på roboten, bruker de kunnskaper de har tillært seg gjennom den analoge programmeringen. Programmering kan ifølge Heikkilä & Mannila (2018) ses på som en abstrakt aktivitet, men at aktiviteten blir mer konkret når for eksempel roboter anvendes som mer fysiske artefakter (Heikkilä & Mannila, 2018). Flere forskere viser til at fysiske og digitale artefakter kan kombineres med mer analoge artefakter eller gestikulere ved hjelp av hendene og kroppene (Aranda & Ferguson, 2018; Otterborn et al., 2020; Palmér, 2017). Blant annet trekkes Blue-Bot frem i kombinasjon med andre objekter eller artefakter i programmeringsundervisning (Otterborn et al. 2020). Et eksempel på bruk av både digitale og analoge verktøy i datamaterialet mitt, er da Blue-Bot ble kombinert med kort med ord eller regnestykker. Elevene måtte forholde seg til kortene, som er selve problemløsningsoppgaven, og roboten som skulle programmeres til å finne løsningen på oppgaven. Selv om Blue-Bot var mest i fokus i disse oppgavene, måtte elevene ta i bruk algoritmisk tenkning og overføre det til et digitalt media. I tillegg ble tidligere kunnskaper om hvordan roboten fungerte og trinnvise instruksjoner anvendt for å løse oppgaven som sto på kortet. Med andre ord kan det sies at elevene tok i bruk kunnskaper de har tilegnet seg gjennom analog programmering inn i den digitale programmeringen, slik som forskning påpeker som hensiktsmessig (Nouri et al., 2019).

Læreren forteller om første gang elevene skulle programmere Blue-Bot etter mye bruk av code.org og Kodetimen i programmeringsundervisningen. I og med at programmeringsspråket ser likt ut, men at enkelte elementer fungerer på forskjellig måte, skapte det undring hos elevene. Å trykke høyre pil betydde ulike ting. Det gjorde at elevene måtte tenke annerledes da de for første gang skulle prøve å programmere Blue-Bot. Dette kan sammenlignes med Piagets syn på utvikling gjennom balansen mellom prosessene assimilasjon og akkomodasjon (Säljö, 2001, s. 61). Assimilasjon var tilfellet da elevene lærte seg programmeringsspråket i code.org og Kodetimen. Da registrerte de informasjon om hvordan programmeringsspråket fungerte. Senere da

Blue-Bot skulle prøves, tok de i bruk den informasjonen de hadde tillært seg fra før av. Elevene opplevde da at roboten ikke gjorde det som var forventet. Dermed oppsto det en ubalanse mellom elevenes forestillingsverden og deres tidligere erfaringer. Den tidligere erfaringen måtte da endres til at høyre pil kan bety å utføre forskjellige kommandoer. Denne forandringen kalles akkomodasjon (Säljö, 2001, s. 61). Ved å endre erfaringen, er prosessene assimilasjon og akkomodasjon i balanse. Endringen er dermed det vi kaller læring (Sjøberg, 1996, s. 3).

I utvalget mitt var en viktig komponent at skolen aktivt arbeidet for og med programmering i undervisningen. Dermed var det å anta at skolen også hadde tilgang på en del ressurser og programmeringsmateriell. Det er imidlertid ikke tilfellet for alle skoler, noe som også tas i opp i debatten om programmering i skolen (Sommerfeldt, 2018). Læreren forteller at det nødvendigvis ikke trengs mye utstyr for å starte med programmering i skolen. Analoge konkreter som lekrelaterte elementer eller dagligdagse «problemer» kan tas i bruk i tillegg til algoritmisk tenkning. Likevel er det hensiktsmessig å ha noe digitalt verktøy tilgjengelig, for å senere kunne overføre det analoge til det digitale. Hvis skolens ansatte har tilgang på ressurser i form av digitale programmeringsverktøy, bør de også ha kunnskaper om hvordan disse kan tas i bruk. På lik linje som at elever ikke blir digitalt kompetente av å få en iPad (Haraldsrud et al., 2020, s. 15), kan ikke elevene utnytte programmeringsmulighetene ved å få en robot foran seg. Selvfølgelig er det mye som kan læres gjennom algoritmisk tenkning, uten mye informasjon om roboten. Ønsket er likevel å ha et mer faglig utbytte ved bruk av roboten. Da er det å foretrekke å ha kunnskaper om hvordan verktøyet kan brukes til å oppnå nettopp det. Ved å sørge for at alle har tilgang på det grunnleggende innenfor digital teknologi, bidras det til å opprettholde fellesskapet i samfunnet (Haraldsrud et al., 2020, s. 16). Dette handler i stor grad om likestilling, noe digitaliseringen utfordrer (Haraldsrud et al., 2020, s. 15). I tillegg er det hvordan verktøyene anvendes, som har noe å si for læringen elevene skal tilegne seg. Säljö (2001) trekker frem at artefaktene kan risikeres å gjøres abstrakte og virkelighetsfjerne dersom integreringen av artefaktene i forståelsen av utvikling og læring ikke mestres (Säljö, 2001, s. 78.) Uten tilstrekkelig opplæring i hvordan lærere kan utnytte programmeringsverktøy til det fulleste, vil det muligens være vanskelig å trekke paralleller til det faglige. Bruk av digitale programmeringsverktøy kan anvendes som et hjelpemiddel og/eller en alternativ arbeidsmetode som kan benyttes i flere fag, uavhengig av kompetansemålene fra læreplanen. Dette vil jeg ta for meg ytterligere i det følgende.

5.2.3 Fagrelatert programmering

Programmering i fag bidrar til en dypere forståelse for problemløsningsstrategier og arbeidsmetoder (Haraldsrud et al., 2020, s. 14). I datamaterialet benyttes programmering som en arbeidsmetode og både analoge og digitale programmeringsverktøy kan benyttes som et hjelpemiddel i ulike fag. I stor grad gjelder algoritmisk tenkning i flere fag, selv om det ofte forbindes med matematikkfaget. I og med algoritmisk tenkning er i tråd med 21st Century Skills, trekkes blant annet programmering inn som nødvendige kompetanser for blant annet å kunne leve i dagen samfunn (Sevik et al., 2016, s. 10). Det samme gjelder for de grunnleggende ferdighetene (Kunnskapsdepartementet, 2017b, s. 11). I begynneropplæringen skal elevene lære om både de faglige og sosiale aspektene innenfor skoleopplæringen (Hoff-Jenssen et al., 2020, s. 150). Funn fra datamaterialet viser at programmeringsundervisningen tar for seg både faglige og sosiale aspekter. Eksempelvis benyttes algoritmisk tenkning i store deler av oppgavene som både er et element i matematikken og som er gjeldende i problemløsning. Skolen skal legge tilrettelegge for å skape et samfunn og gode liv for Norges befolkning (NOU 2015:8). Dette samsvarer med 21st Century Skills (Lye & Koh, 2014; Nouri et al., 2019; Binkley et al., 2012; Sevik et al., 2016, s. 10), som innebærer egenskaper elevene bør benytte i arbeid med fag og generelt i hverdagen.

I og med at samfunnet vårt går igjennom en stor digitalisering blir det stadig mer aktuelt å introdusere flere digitale arbeidsmetoder, som programmering, tidligere i skoleopplæringen. På den måten lærer elevene fra en tidlig alder flere viktige ferdigheter som kan være nyttige å trekke inn i fag og i hverdagslige situasjoner. I tillegg er det en fin måte å tilpasse undervisningen på og benytte mer varierte arbeidsmetoder i undervisningen. I dagens skole har flere tatt i bruk elev-iPad´er som et verktøy i opplæringen (Haraldsrud et al., 2020, s. 15). Dermed vil det være hensiktsmessig å utnytte disse verktøyene til det fulleste og lære elevene hvordan de kan ta det i bruk i undervisningen. Hvordan verktøyene benyttes har mye å si for elevenes læring (Haraldsrud et al., 2020, s. 15). I datamaterialet vises det til at bruk av programmeringsverktøy i fag er en fin måte å variere og tilpasse undervisningen på, i tillegg til at elevene får brukt kreativiteten sin. Eksempelvis fortelles det om bruk av Bloxels inn i norskfaget. Istedenfor at elevene for eksempel dikter og skriver fortellinger skriftlig, enten på ark eller iPad, kan for eksempel Bloxels benyttes. Som læreren viser til er det mulig å lage figurer, rekvisitter, bygge ulike scener og lage

fortellinger i Bloxels. Det understreker både mål innenfor norskfaget og målet innenfor matematikk, med å utforske og lage strukturer og mønster i spill. I tillegg lærer elevene å anvende et digitalt verktøy og bruker kreativiteten sin på en annen måte. Å være kreativ og å kunne uttrykke seg på ulike måter, er evner som bidrar til å berike samfunnet vårt (Kunnskapsdepartementet, 2017c). Samtidig er digital kompetanse en sentral del av fagene i skolen, samt en forutsetning for å kunne delta i læring (NOU 2015:8).

Læreren kobler ofte programmering til hverdagslige situasjoner som for eksempel å lage mat og følge oppskrifter på kjøkkenet. Haraldsrud et al. (2020) viser til at algoritmer anvendes på flere steder i hverdagen vår, hvor blant annet matoppskrifter trekkes frem som eksempel (Haraldsrud et al., 2020, s. 184). Selv om faget mat og helse muligens ikke er like utbredt i begynneropplæringen, er det likevel viktige ferdigheter og kunnskaper å ta med seg, fra matlaging og inn i matlaging, for yngre elever. Som læreren poengterer kan matematikk og regning trekkes inn i matlaging, men at programmering like enkelt kan anvendes. Ved å se sammenhengen mellom å doble og å gjenta, kan elevene få et mer visuelt og praktisk inntrykk av hvordan «gjenta» brukes i programmering. Dersom elevene har lært seg å doble i matematikken, vet de ofte at det er det samme som å enten multiplisere med to eller å addere tallene sammen. Eller som læreren eksemplifiserer, å hente først tre egg og deretter hente tre egg til. Da dobles antall egg, men handlingen gjentas også. Imidlertid kan benyttelse av begrepet «gjenta» i stedet for begrepet «doble» i et matematisk perspektiv, gå på bekostning av elevenes forståelse for grunnleggende matematiske begreper. Om det er gunstig å lære å «gjenta» i stedet for å «doble» eller å trenge å regne, kan diskuteres. Det viktigste er at elevene får en fullstendig begrepsforståelse for ulike matematiske begreper gjennom undervisningen.

I tillegg til å øve på slike begreper og deres betydning i matlaging, lærer elevene også å følge trinnvise instruksjoner i en oppskrift. Trinnvise instruksjoner er som sagt, både et kompetansemål i matematikken i begynneropplæringen (Kunnskapsdepartementet, 2019), og et vesentlig aspekt innenfor programmering. Eksempelet med matlaging er en fin representasjon av algoritmisk tenkning og programmering i ulike situasjoner og fag, hvor det nødvendigvis ikke tenkes at programmering kan trekkes inn. Det er ifølge Liukas (2018) kun mennesker som kan tenke algoritmisk (Liukas, 2018, s. 112) og når elevene skal utvikle sin algoritmiske tankegang, skal de tilnærme seg problemer på en

systematisk måte. I tillegg til å bruke sin teknologiske kompetanse til å overføre det til et digitalt media som skal løse problemet (Sevik et al, 2016, s. 14). Dermed kan den algoritmiske tenkningen trenes opp gjennom å programmere (Mannila, 2020, s. 79).

I skolen kan det være elever som misliker enkelte fag som norsk og matematikk. Dette kan også være tilfellet for programmering. Det er ikke gitt at programmering motiverer alle elever og at alle har interesse for å lære å programmere (Haraldsrud et al., 2020, s. 163). I forskningsprosjektet mitt observerte jeg at elevene virket motiverte for og gledet seg til programmeringsøkten, da de så det oppført på dagsplanen på tavla. Når elever viser motivasjon i et fag er det ofte et fag eleven liker og, ikke minst, mestrer. For de elevene som ikke er like begeistret for programmering, kan det antas at tilfellet er annerledes. Eleven har muligens ikke motivasjon for eller viser engasjement for programmering, fordi programmering kanskje ikke er noe eleven føler mestring i. Dette er en reell situasjon som også er svært relevant for andre fag og temaer. Dermed er lærerrollen svært viktig for å kunne tilpasse til disse elevenes proksimale utviklingssone og muligens også interesser, samt veilede dem med et mål om å oppnå mestring.

5.3 Lærerrollen

Det at elevene i datamaterialet mitt arbeider sammen i programmeringsundervisningen, gjør at elevene også samhandler med hverandre for å undersøke og finne ut av det de lurer på. Dermed blir læreren mer en veileder som ikke direkte gir elevene svar på det de lurer på, men hjelper de i gang med tankeprosesser. Stillasbygging blir som oftest forbundet med at læreren veileder elevene (Belland, 2017, s. 25). Ifølge Wood et al. (1976) er diskusjoner om problemløsning eller tilegnelse av ferdigheter ofte basert på en antakelse om at eleven er alene og uten hjelp (Wood et al., 1976, s. 90). I og med at elevene arbeider sammen, innebærer veilederrollen mer enn selve modellæringen. Veilederens rolle er å bidra med de elementene som er utenfor elevens kapasitet, slik at eleven kan konsentrere seg om de elementene som er innenfor sitt kompetanseområde (Wood et al., 1976, s. 90). I problemløsningsoppgaver, som programmeringsaktiviteter ofte er, er støtte gjennom stillaser viktig (Belland, 2017, s. 4). I programmering handler veilederrollen om at læreren ikke skal lære elevene alt, men gi de noen «verktøy» de kan jobbe ut fra. I tillegg er det viktig å komme tilbake og se hvordan de tar i bruk disse verktøyene og vurdere om de trenger mer verktøy eller ikke. Læreren understreker at elevene ikke skal gis svarene direkte, men at læreren bør samhandle med dem og få de i dialog. Da kan de enten alene eller sammen med gruppen, komme frem til en mulig

løsning på problemet. På den måten må elevene selv reflektere over hva oppgaven går ut på, hva de trenger og hvordan skal de ta det i bruk.

I forskning uttrykkes viktigheten av læreres forventinger til elevenes motivasjon og muligheter for å lære. I tillegg bør læreren ha tillit til elevenes evner i forkant av integreringen av programmering i undervisningen (Kjällander et al, 2021; Otterborn et al, 2019). Selv om læreren uttrykker at veilederrollen i stor grad handler om å hjelpe elevene med å komme i gang, veilede dem der de står fast og motivere de til videre arbeid og problemløsning, er en stor del av lærerrollen å oppmuntre til samarbeid og felles feilsøking. I forskning opplyses det om at læreren ikke bør tre inn i programmeringen «for tidlig», men la elevene undersøke og prøve seg frem (Heikkilä & Mannila, 2018). Dette er noe læreren også nevner. Det må gis rom til å prøve, men sette noen rammer, betingelser eller gi elevene noe informasjon de kan jobbe ut fra. Dette er spesielt viktig når nye oppgaver eller programmeringsverktøy skal introduseres.

Forskere fremhever feilsøking som et viktig aspekt i programmering. Hvordan læreren initierer og oppmuntrer elevene til dette, har mye å si for elevenes læring (Kjällander et al, 2021; Heikkilä & Mannila, 2018). Kjällander et al. (2021) forteller at feilsøking skal skje i samspill mellom elever, men også mellom elever og læreren (Kjällander et al, 2021). Blant annet ved å gjennomgå elevenes løsninger, vil det komme tydeligere frem hva som er gjort feil, hvordan det kan endres og prøves ut neste gang (Kjällander et al, 2021; Heikkilä & Mannila, 2018). I løpet av flere av programmeringsaktivitetene fra datamaterialet mitt, ser jeg at feilsøking finner sted både blant elevene og med læreren involvert. Som læreren sier, er det flere løsninger til samme svar og det å gjøre feil er ikke nødvendigvis feil i ulike situasjoner. For eksempel kan en gruppe få i oppgave å finne den raskeste veien til svaret og en annen gruppe skal finne treigeste veien. Begge løsningene vil være riktige, men de har ulike måter å komme frem til det samme svaret på. Ifølge læreren skaper dette diskusjoner, noe som ofte fremmes i undervisningen. Ved å diskutere en oppgave, løsning eller lignende, får elevene hørt hverandres perspektiver og synspunkter. Da kan de argumentere og reflektere rundt de ulike løsningsstrategiene.

Språket er det viktigste medierende læringsredskapet vi har (Dysthe, 2001a; Säljö, 2001) og sammen med kommunikasjon, er de sentrale i et sosiokulturelt perspektiv. Læring skjer gjennom å lytte, snakke, skrive og lese (Dysthe, 2001a, s. 12), og det er gjennom kommunikasjon at vi blir delaktige i ferdigheter og kunnskaper (Säljö, 2001, s.

38). Vi har en unik evne til å dele erfaringer med hverandre gjennom språket (Säljö, 2001, s. 35). Dermed er en viktig forutsetning for læring i programmering at elevene kommuniserer, reflekterer og undersøker ulike løsninger, valg av programmeringsspråk og lignende. Ved at læreren initierer til dette i plenum og oppfordrer elevene til å ta i bruk aspekter innenfor algoritmisk tenkning som feilsøking, undring, utforskning og samarbeid (Nouri et al., 2019, s. 9-11) i arbeid med oppgaver, vil elevene ta med seg nyttige ferdigheter og kunnskaper. Videre kan de benyttes i andre fag, i hverdagslivet og senere i arbeidslivet.

5.3.1 Programmeringsundervisning

Programmering læres gjennom å gjøre (Mannila & Nordén, 2020, s.71) og utforskning ses på som et relevant aspekt i læringen (Shanck et al., 2009, s. 164). Det samme gjør John Deweys syn på kunnskap, som ofte uttrykkes ved *learning by doing* (Vaage, 2001, s. 130). Aktiviteten med BOLT kan være et godt eksempel på dette. Elevene fikk nok informasjon til å sette i gang, men gjennom å utforske, fikle og feilsøke lærte de hvordan BOLT fungerte. I tillegg lærte de hvordan de kunne ta i bruk disse kunnskapene i oppgaveløsning, som ble sett på som en utfordring. Gjennom en programmeringsprosess lærer elevene også hvordan de selv tenker (Papert, 1993, s. 19). Dette har en sammenheng med det forskerne ved MIT mener om at barn utvikler en forståelse av seg selv, til andre og den digitale verden gjennom programmering av datamaskiner (Mannila, 2020, s. 80). I datamaterialet vises dette blant annet under LEGO-aktiviteten. Elevene fikk ikke beskjed om hva de kunne eller ikke kunne lage, hva slags programmeringsspråk som burde tas i bruk eller at det er hensiktsmessig å teste ut instruksjonene før de delte det med andre. De fleste elevene begynte ganske raskt med å lage en figur, som kan sees på som lekaspektet ved oppgaven. Elevene hadde dermed frihet til å ta i bruk kreativiteten og fantasien sin. Da de så skulle gå over til å skrive ned instruksjoner, var det flere som tok utgangspunkt i eget perspektiv, med forkunnskaper om hvordan enderesultatet skulle bli. Dermed er det tydelig, i likhet med frokost-oppgaven fra «Hei Ruby», at det som går på automatikk eller tas som en selvfølge at elevene vet og gjør, ikke tas med i instruksjonene. Ved å benytte disse kunnskapene, lærte elevene hva som måtte til for å rette opp feilene. Elevene lærte hvordan de selv har tenkt gjennom utforskning og undersøkelse av sine egne instruksjoner og kunne gjøre endringer deretter.

Lock & Strong (2014) mener at læreren må tilpasse undervisningen til elevenes ferdighetsnivå, for å opprettholde den proksimale utviklingssonen. Dersom nivået er for høyt eller for lavt, vil læring ikke skje (Lock & Strong, 2014, s. 153). Eksempelvis kan problemløsningsoppgaver i programmeringen tilpasses ut ifra elevene ståsted. Ved å tilpasse ulike mål for videre utforskning og problemløsning, som for eksempel ved å undersøke ulike løsningsstrategier og løsningsforslag, kan ulike grupper få forskjellige innfallsvinkler og strategier å arbeide ut fra, som videre kan diskuteres i plenum. På den måten arbeider alle elevene med samme hovedproblemløsningsoppgave, men har fått mer individuelle vinklinger og tilpasninger ut ifra hvor de er i utviklingen. Da blir det både tilpasset til det gruppen kan klare sammen og det blir ikke like synlig hvem som er hvor i utviklingen, da at alle arbeider med samme oppgave. I tillegg kan det tilpasses gjennom bruk av ulike programmeringsverktøy. Ifølge læreren er benyttelse av Blue-Bot roboten i tillegg til Blue-Bot appen en fin måte å inkludere alle elever på. For de elevene som fysisk ikke er i stand til å programmere direkte på roboten, kan de programmere den samme roboten via et annet media. Da er de en del av fellesskapet og arbeider med samme robot og oppgave, bare ved hjelp av et annet digitalt hjelpemiddel.

Både i tidligere forskning og i datamaterialet mitt, vises det at en kombinasjon mellom analog og digital programmering er å foretrekke når elevene skal begynne å programmere (Berg, 2021; Chibas et al., 2018; Heikkilä & Mannila, 2018; Otterborn et al., 2020). Likevel er det hensiktsmessig å ha fokus på hva programmering er før elevene går over til digital programmering (Nouri et al., 2019; Berg, 2021). Læreren mener det er viktig at elevene forstår hva som skjer når de programmerer og hvordan programmering foregår. Samtidig poengterer læreren at når elevene først går over til digital programmering, må den analoge programmeringen ikke gis slipp på for å sikre at forståelsen forblir. Ved å trekke paralleller til elevenes hverdagsliv, kan programmeringen ses på som mer relevant i elevenes øyne, da de har noe konkret å koble det opp mot. Eksempelvis da elevene så sammenhengen mellom programmering og leseforståelse, eller å se sammenhengen mellom «spinn»-kommandoen i blokkprogrammering med BOLT og «spinn» som et dansetrinn i danseoppgaven med Ruby.

Flere forskere trekker frem læreres usikkerhet og manglende kunnskaper rundt programmering og hvordan det skal anvendes i undervisningen (Otterborn et al, 2019; Kjällander et al, 2021; Stigberg & Stigberg, 2020). Som Mannila (2020) utdyper, er det

vanskelig å vite hva som fungerer best når ingen årskull har gjennomført grunnskolen med programmering som en del av utdanningen (s. 126). Dermed begynner lærere og elever fra null. Da er det hensiktsmessig å begynne med det grunnleggende (Mannila, 2020, s. 126). Gjennom datamaterialet mitt er det tydelig at læreren tar i bruk analog programmering sammen med konkreter og hverdagslige aktiviteter, som er kjente for elevene. Videre kobles det opp mot gjenkjennelige elementer i programmeringen, algoritmisk tenkning og kompetansemål i matematikk. Ved å gjøre dette, gir læreren elevene mulighet til å lære seg å programmere og forstå hva programmering innebærer. I tillegg til å sikre den grunnleggende forståelsen i programmering. I undervisningen er samarbeid en nøkkelfaktor. Det fremmer klassemiljø, læringsmiljø, inkludering og metakognisjon. Læreren poengterer at tilgangen på mye utstyr nødvendigvis ikke er viktig for å begynne med programmering, men at for eksempel LEGO, puslespill, eller rett og slett å følge en oppskrift på kjøkkenet kan benyttes i starten. Med det har elevene mulighet til å leke, være kreative, lære viktige faglige aspekter på en annen måte og samtidig lære å programmere. Disse kunnskapene, sammen med digital forståelse og ferdigheter, er med på å fremme elevenes læring og vårt fremtidige samfunn.

5.4 Begrensninger og videre forskningsstudier

5.4.1 Svakheter og begrensninger

Som nevnt i kapittel 3.7 Feilkilder, består denne studien av mye bekreftende funn. Dette er i stor grad grunnet utvalget, da læreren jeg forsket på både har faglig grunnlag og praktiske erfaringer med bruk av programmering jevnt over flere år med samme klasse. Dermed er det nærmest forventet at læreren tar i bruk programmering på en måte som blant annet fremmer faglighet, problemløsningsstrategier og samarbeidslæring i undervisningen. Elevene har fått programmeringsundervisning av en lærer med stor kompetanse, samtidig som skolen har mye ressurser i forbindelse med programmering. Selv om elevene har hatt en lengre pause fra programmering under pandemien, har de bakgrunnskunnskaper som anvendes i undervisningen. I og med at utvalget mitt forbeholder en skole og en lærer med kompetanse, ressurser og utdanning innenfor programmering, kan det ses på som en svakhet i forbindelse med overførbarhet til andre felt. I tillegg er studien min en casestudie, hvor jeg kun har én informant. Dermed kan jeg ikke si at undervisningsoppleggene og aktivitetene som benyttes i mitt forskningsprosjekt, vil fungere i like stor grad for andre. Til tross for at en casestudie gir

meg mulighet til å undersøke programmeringsundervisningen i dybden og tilegne meg kunnskaper fra en erfaren lærer, kan det ses på som en svakhet i forskningsfeltet.

5.4.2 Videre forskning

Med bakgrunn i det som ble sagt over, er det behov for mer forskning på programmeringsundervisning i begynneropplæringen. Gjerne med flere informanter, skoler og eventuelle andre som kan styrke forskningsfeltet med kunnskaper. Spesielt i den norske skolen i dag, med programmering som en del av den nye læreplanen, er det hensiktsmessig at alle lærere får mulighet til å tillære seg kunnskaper og ferdigheter innenfor programmering. I tillegg er det formålstjenlig at lærere får innblikk i hvordan det kan undervises i praksis. Gjennom videre forskning mener jeg det vil være aktuelt å undersøke nærmere bruk av analog programmering i begynneropplæringen, da jeg gjennom denne studien har ervervet kunnskaper om dets store innvirkning på programmeringsundervisningen. I tillegg vil det bidra til å sette lys på det faktum at programmering er mye mer enn bare datamaskiner, roboter og iPad'er, noe jeg antar flere ønsker å opplyses om. Ved å informere om dette, håper jeg, og tror, at flere lærere vil se på programmering som mindre «skummelt» og tenke mer hverdagslig, praksisnært og samfunnsrettet, ved å trekke programmering inn i elevenes faglige undervisning. Det vil også være interessant å undersøke elevenes tanker rundt programmering i undervisningen. I datamaterialet vises det at elevene hadde stor glede av programmering, noe som er svært positivt. Ved å undersøke om dette er tilfelle for flere elever, kan opplegg og undervisning planlegges og tilrettelegges deretter om endringer bør forekomme.

6 Konklusjon

I starten av arbeidet med masteroppgaven, var ønsket mitt å lære mer om hvordan programmering blir tatt i bruk i undervisningen i begynneropplæringen. Det var både for min egen del, som fremtidig lærer, og for å bidra til forskningsfeltet med kunnskaper om programmeringsundervisning for de yngste elevene på barneskolen. Ut ifra mine egne ønsker, læreres bekymringer, viten om en økende digitalisering av samfunnet og fagfornyelsen, formulerte jeg følgende problemstilling:

Hva kjennetegner undervisning i programmering i begynneropplæringen?

Gjennom masteroppgaven har relevant forskning og teori blitt trukket frem, og drøftet opp mot funn fra datamaterialet. I tillegg har jeg benyttet to forskningsmetoder: semistrukturerte intervjuer og deltagende observasjon, som sammen med forskningsspørsmålene gjør det mulig å besvare problemstillingen. Studien inneholder en del funn som bekreftes av teori og tidligere forskning. Dette skinner igjennom i drøftingen og når jeg nå skal vise til de drøftede temaene basert på hovedfunnene og trekke slutninger opp mot problemstillingen.

Analog programmering kommer frem som svært fremtredende innenfor programmering i begynneropplæringen. Læreren tar i bruk analog programmering i store deler av forskningsperioden og benytter verktøy i form av kjente, lekbaserte artefakter som LEGO og puslespill. I tillegg belyses viktigheten av det å relatere programmeringen til mer hverdagslige og aldersnære oppgaver, som å lage mat, kle på seg, spise frokost og lage en egen dans. Videre kan dette kombineres med ulike fag og trekke inn faglig og sosial læring, som er sentralt i begynneropplæringen (Hoff-Jenssen et al., 2020, s. 150). Programmering læres gjennom å gjøre (Mannila & Nordén, 2020, s.71) og læreren tar i bruk aktiviteter i undervisningen som krever at elevene både bruker ulike programmeringsverktøy og hverandre, for å løse oppgavene. Læreren poengterer likevel at mye utstyr ikke nødvendigvis er gjeldende for å begynne med programmering i begynneropplæringen. Ved å ta i bruk verktøy som elevene har kjennskap til, blir fokuset på programmeringen og målet for undervisningen. I tillegg til at lek trekkes inn som et sentralt aspekt. I den sosiokulturelle læringsteorien er samhandling og interaksjoner med andre sentralt (Dysthe, 2001b, s. 42). Elevene programmerer aldri alene og benytter seg av hverandres kunnskaper og ferdigheter til å arbeide mot et felles mål. Spesielt i den analoge programmeringen er samarbeid fremtredende og læreren

oppmuntrer elevene jevnlig til å kommunisere, utforske og feilsøke med hverandre gjennom problemløsning. Læreren tar i bruk samarbeid for at elevene kan opptre som drahjelp for hverandre. Belland (2017) kaller det for *peer scaffolding* (Belland, 2017, s. 25). Ved å benytte seg av det, kan elevene bygge videre på hverandres tanker og kunne løse problemene lettere, enn om de hadde vært alene om det.

Et annet aspekt som kommer tydelig frem innenfor den analoge programmeringen, er læring om og bruk av ulike programmeringsspråk og -begreper. Læreren veksler mellom fri bruk av programmeringsspråk og bruk av et gitt programmeringsspråk i undervisningen. I tillegg tok læreren i bruk de ulike programmeringsbegrepene i undervisningen for at elevene skulle få et forhold til dem fra start. Ved å se sammenhengen mellom programmeringsbegreper i hverdagslige situasjoner, oppgaver og matematiske begreper, kan elevene benytte tidligere kunnskaper og erfaringer til å forstå hva programmeringsbegreper innebærer. Når elevene lærer disse begrepene gjennom analog programmering, vil de ha kjennskap til begrepene når de går over til digital programmering. Læreren viser til «grunnleggende forståelse» om det som er viktig å tillære seg gjennom analog programmering. «Den grunnleggende forståelsen» forbindes med nyttige kunnskaper om hva programmering handler om og innebærer, samt hva som egentlig skjer når vi programmerer. I samspill med aktiviteter som fremmer lek og algoritmisk tenkning, lærer elevene programmering gjennom å benytte seg av kreativiteten sin, samarbeid med medelever og strategier for problemløsning og logisk tenkning. Disse ferdighetene og egenskapene, med flere, er forutsetninger for digitaliseringen og dagens skole og samfunn. Lek er sentralt i begynneropplæringen og i kompetansemål for matematikkfaget (Kunnskapsdepartementet, 2019). Algoritmisk tenkning forbindes også ofte med matematikkfaget, men kan inkluderes i fag og i hverdagslivet, hvor problemløsning foregår (Haraldsrud et al., 2020, s. 183). Å tenke algoritmisk er det kun mennesker som kan (Liukas, 2018, s. 112) og ved å programmere trenes den algoritmiske tankegangen opp (Mannila, 2020, s. 79). Et aspekt innenfor algoritmisk tenkning, som benyttes mye av i undervisningen, er feilsøking. Læreren oppmuntrer elevene til å prøve og feile og poengterer stadig at det verste som kan skje er at de må prøve igjen. Det er også mange ulike måter å komme frem til riktig svar på, noe som kan være en motivasjonsfaktor for enkelte. Læreren mener det er fremtredende å legge opp til feilsøking og oppmuntre elevene til det allerede fra start. I tillegg sier læreren at begrepet bør brukes konsekvent, i likhet med andre programmeringsbegreper. Når elevene feilsøker og programmerer generelt, er samarbeid en nøkkelfaktor. Læreren

mener gruppesammensetningene bør varieres for at alle kan øve på å samarbeide med hverandre uavhengig av ulike faktorer. Læreren trekker jevnlig frem at «*to hoder tenker bedre enn ett*», noe som legger føringer for at elevene ofte kan ha behov for å diskutere, undersøke, reflektere og feilsøke med medelever. På den måten kan elevene hjelpe hverandre og, ikke minst, lære av hverandre. Andre elementer i algoritmisk tenkning er selvfølgelig også sentralt, men i datamaterialet kommer feilsøking spesielt frem som viktig.

Trinnvise instruksjoner «spiller hovedrollen» i programmeringsoppgavene og -undervisningen, noe som også er en del av læreplanen innenfor matematikk i begynneropplæringen (Kunnskapsdepartementet, 2019). Dermed er det flere måter å trekke inn programmering på i ulike fag, sammen med kompetansemålet i matematikk om trinnvise instruksjoner. Når det lages instruksjoner som er trinnvise, lages det en slags oppskrift som skal følges for å komme frem til et ønsket resultat. Akkurat som i programmering. I tillegg kan både analoge og digitale programmeringsverktøy tas i bruk som en arbeidsmetode i ulike fag. På den måten tilpasses undervisningen til elevene og undervisningen blir mer variert. I den sosiokulturelle læringsteorien fungerer vi i et samspill med artefaktene. Gjennom artefaktene kan vi løse problemer, samt beherske sosiale praksiser (Säljö, 2001, s. 78). Dermed er det ikke artefaktene i seg selv som er avgjørende for å løse problemet, men det er hvordan vi benytter det i samspill med våre egne tanker og ideer, som bidrar til å komme frem til en løsning. Programmeringsverktøyene kan derfor anses som arbeidsmetoder eller hjelpemidler i programmering, eller i fag generelt.

Et siste aspekt som er sentralt for å besvare problemstillingen min, er lærerrollen som veileder. Det innebærer å gi elevene nok informasjon og «verktøy» til å komme i gang, men ikke direkte gi de fremgangsmåten for løsningen eller si hva løsningen er. Ved å veilede elevene, er det også hensiktsmessig å vurdere om elevene eventuelt trenger flere «verktøy» eller ikke. Videre bør elevene oppmuntres til å stille spørsmål ved og reflektere over arbeidet de allerede har gjort, for å finne ut hva neste steg er. Til slutt ønsker jeg å poengtere at rammer er nødvendige å sette i enhver programmeringsøkt, men at mye av hensikten med programmeringsoppgaver er problemløsning og samarbeid. Dermed er det elevene som skal ta i bruk den algoritmiske tenkeren, ved blant annet å feilsøke og utforske sammen, for å komme frem til en løsning.

Denne studien viser at programmeringsundervisning i begynneropplæringen kjennetegnes ved både faglige og sosiale aspekter, med fokus på analog programmering og samarbeid gjennom arbeid med algoritmisk tenkning og hverdagsnære situasjoner og verktøy. I tillegg vises det at lærerrollen er svært vesentlig for at elevene skal lære å programmere i samhandling med hverandre. Sammen med disse kjennetegnene skal undervisningen gjøre elevene rustet for å ta del i samfunnets digitalisering. Videre bør det arbeides for at lærere får mulighet til å tillære seg kunnskaper om programmeringsundervisning og at skoler får tilgang til de programmeringsressursene som behøves. Gjennom masteroppgaven har jeg tillært meg mye kunnskaper som har vært svært gjeldende for oppgaven, men også for min lærerpraksis som fremtidig lærer i begynneropplæringen. Ønsket mitt om mer kunnskap i programmering og bruk av digitale og analoge verktøy i undervisningen, har blitt oppfylt. I tillegg verdsetter jeg prosessen med masteroppgaven, da jeg har fått et dypere innblikk i hvordan jeg selv kan trekke inn programmering i undervisningen. Min motivasjon for programmering i undervisningen i begynneropplæringen har økt og jeg gleder meg til å ta i bruk kunnskapene og lærdommen jeg har tillært meg med mine fremtidige elever og kollegaer.

Litteraturliste

Akerbæk, M. & Karlsen, J. (2019). Koding som skapende og utforskende aktivitet. I K. H. Karlsen & G. B. Bjørnstad (Red.), *Skaperglede, engasjement og utforskertrang: nye perspektiver på estetiske og tverrfaglige undervisningsmetoder som redskap I pedagogisk virksomhet* (s. 143-159). Universitetsforlaget.

Anker, T. (2021). *Analyse i praksis: En håndbok for masterstudenter*. Cappelen Damm Akademisk.

Aranda, G. & Ferguson, J. P. (2018). *Unplugged Programming: The future of teaching computational thinking?* Pedagogika, 68(3), 279-292. https://doi.org/10.14712/23362189.2018.859

Belland, B. R. (2017). *Instructional Scaffolding in STEM Education: Strategies and Efficacy Evidence*. Springer.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M. & Rumble, M. (2012). Defining Twenty-First Century Skills. Springer.

Bocconi, S., Chioccariello, A. & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. Report prepared for the Nordic@BETT2018 Steering Group. https://www.itd.cnr.it/doc/CompuThinkNordic.pdf

Dysthe, O. & Igland, M.-A. (2001). Vygotskij og sosiokulturell teori. I O. Dysthe (Red.), *Dialog, samspel og læring* (s. 73-90). Abstrakt Forlag.

Dysthe, O. (2001a). Om sammenhengen mellom dialog, samspel og læring. I O. Dysthe (Red.), *Dialog, samspel og læring* (s. 9-30). Abstrakt Forlag.

Dysthe, O. (2001b). Sosiokulturelle teoriperspektiv på kunnskap og læring. I O. Dysthe (Red.), *Dialog, samspel og læring* (s. 33-72). Abstrakt Forlag.

Flyvbjerg, B. (2006). Five Misunderstandings About Case-Study Research. Sage.

Gleiss, M. S. & Sæther, E. (2021). *Forskningsmetode for lærerstudenter: Å utvikle ny kunnskap i forskning og praksis*. Cappelen Damm Akademisk.

Gobo, G. (2011). Ethnography. I D. Silverman (Red.), *Qualitative research: issues of theory, method and practice* (3. utg., s. 15-34). Sage.

Haraldsrud, A. D., Sveinsson, H. A. & Løvold, H. H. (2020). *Programmering i skolen.* Universitetsforlaget.

Hei Ruby. (u. å). *Hei Ruby: Bli med Ruby på eventyr inn i datamaskinens magiske verden.*Hei Ruby. https://www.heiruby.no/

Hoff-Jenssen, R., Bjerke, M. O. & Afdal, H. W. (2020). Begynneropplæring – et kjent, men uklart begrep: En analyse av læreres perspektiver. *Nordisk tidsskrift for pedagogikk og kritikk, 6*, 143-157. https://doi.org/10.23865/ntpk.v6.2030

Hovda, K. & Liukas, L. (2020). *Algoritmisk tenkning med Ruby: Inspirasjon og veiledning til arbeid med algoritmisk tenkning i skole og barnehage*. InfoVest forlag.

Høgheim, S. (2020). *Masteroppgaven i GLU*. Fagbokforlaget.

Johannessen, L. E. F., Rafoss, T. W. & Rasmussen, E. B. (2021). *Hvordan bruke teori?: Nyttige verktøy i kvalitativ analyse*. Universitetsforlaget.

Johansen, A.-K. (2020, 11. juli). *Programmering vil bli en utfordring for lærere.* Forskning.no. https://forskning.no/barn-og-ungdom-hogskolen-i-ostfold-matematikk/programmering-vil-bli-en-utfordring-for-laerere/1711838

Johnson B. & Christensen, L. (2012). *Educational research: Quantitative, Qualitative, and Mixed Approaches* (4. utg.). Sage.

Kelentric, M., Helland, K. & Arstorp, A.-T. (2017). Rammeverk for lærerens profesjonsfaglige digitale kompetanse. *Senter for IKT i utdanningen*. https://www.udir.no/contentassets/081d3aef2e4747b096387aba163691e4/pfdk-rammeverk-2018.pdf

Kjällander, S., Mannila, L., Åkerfeldt, A. & Heintz, F. (2021). Elementary Students' First Approach to Computational Thinking and Programming. Education Sciences.

Krumsvik, R. J. (2014). *Forskningsdesign og kvalitativ metode – ei innføring*. Fagbokforlaget.

Kunnskapsdepartementet. (2017a, 25. August). *Framtid, fornyelse og digitalisering: Digitaliseringstrategi for grunnopplæringen 2017-2021*. https://www.regjeringen.no/contentassets/dc02a65c18a7464db394766247e5f5fc/kd_framtid_fornyelse_digitalisering_nett.pdf

Kunnskapsdepartementet. (2017b). *Overordnet del – grunnleggende ferdigheter.* Fastsatt som forskrift ved kongelig resolusjon. Læreplanverket for Kunnskapsløftet 2020. https://www.udir.no/lk20/overordnet-del/prinsipper-for-laring-utvikling-og-danning/grunnleggende-ferdigheter/?TilknyttedeKompetansemaal=true

Kunnskapsdepartementet. (2017c). *Overordnet del – skaperglede, engasjement og utforskertrang.* Fastsatt som forskrift ved kongelig resolusjon. Læreplanverket for Kunnskapsløftet 2020. https://www.udir.no/lk20/overordnet-del/opplaringensverdigrunnlag/1.4-skaperglede-engasjement-og-utforskertrang/?lang=nob

Kunnskapsdepartementet. (2019). *Læreplan i matematikk (MAT01-05).* Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020. https://data.udir.no/kl06/v201906/laereplaner-lk20/MAT01-05.pdf?lang=nno

Kvale, S. & Brinkmann, S. (2010). *Det kvalitative forskningsintervju* (2. utg.). Gyldendal Akademisk

Kvarv, S. (2021). *Vitenskapsteori – tradisjoner, posisjoner og diskusjoner* (2. utg.). Novus Forlag.

Liukas, L. (2018). *Hei Ruby: Oppdag koding*. InfoVest Forlag AS.

Lock, A. & Strong, T. (2014). *Sosial konstruksjonisme: teorier og tradisjoner*. Fagbokforlaget.

Lye, S. Y. & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? Elsevier Ltd.

Lær Kidsa Koding. (u.å.). Koding i skolen: *Oppgavehefte i tråd med LK20*. Lær Kidsa Koding. https://docs.google.com/presentation/d/1vV3OY0w3um3I0IFJMLkmKRoNDp6u3-D31PeKIZtuyxc/edit?usp=sharing

Mannila, L. & Nordén, L.-Å. (2020). *Att undervisa textbaserad programmering i skolan*. Studentlitteratur.

Mannila, L. (2020). *Att undervisa i programmering i skolan: Varför, vad och hur?* Studentlitteratur.

Moreau, H. N. (2021, 31. desember). Skal lære elevene koding, men forstår det ikke selv. *NRK, Innlandet*. https://www.nrk.no/innlandet/laerere-trenger-hjelp-til-a-knekke-koden-pa-koding-1.15781343

NESH. (2021). *Forskningsetiske retningslinjer for samfunnsvitenskap og humaniora* (5. utg.). https://www.forskningsetikk.no/globalassets/dokumenter/4-publikasjoner-som-pdf/forskningsetiske-retningslinjer-for-samfunnsvitenskap-og-humaniora.pdf

NOU 2015: 8. (2015). *Fremtidens skole – Fornyelse av fag og kompetanser.* https://www.regjeringen.no/no/dokumenter/nou-2015-8/id2417001/?ch=2

Nouri, J., Zhang, L., Mannila, L. & Norén, E. (2019). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. Education Inquiry.

Osmo. (u. å.). *Coding Starter Kit*. Osmo. https://www.playosmo.com/en/shopping/kits/coding/

Otterborn, A., Schönborn, K. J. & Hultén, M. (2019). Investigating Preschool Educators' Implementation of Computer Programming in Their Teaching Practice. Springer.

Palm, K., Becher, A. A. & Michaelsen, E. (2020). Den viktige begynneropplæringen: Aktuelle fagområder og kritiske perspektiver. I K. Palm & E. Michaelsen (Red.), *Den viktige begynneropplæringen: En forskningsbasert tilnærming* (2. utg., s. 13-31). Universitetsforlaget.

Papert, S. (1993). *Mindstorms: Children, Computers, and Powerful Ideas* (2. utg.). Basic Books.

Postholm, M. B. & Jacobsen, D. I. (2018). *Forskningsmetode for masterstudenter i lærerutdanningen*. Cappelen Damm Akademisk.

Postholm, M. B. (2005). *Kvalitativ metode: En innføring med fokus på fenomenologi, etnografi og kasusstudier*. Universitetsforlaget.

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

Shanck, R. C., Berman, T. R. & Macpherson, K. A. (2009). Learning by doing. I C. M. Reigeluth (Red.), *Instructional-design theories and models: A New Paradigm of Instructional Theory* (2. utg., s. 161-181). Routhledge. https://books.google.no/books?hl=no&lr=&id=OWavJCNfhcsC&oi=fnd&pg=PT173&dq=learning+by+doing&ots=29OwMP7X2i&sig=f4HkRAmY8dm0PSnkjaJlexmxioQ&redir_esc=y#v=onepage&q&f=false

Silverman, D. (2013). *Doing qualitative research* (4. utg.). Sage.

Sjøberg, S. (1996). Forstått og misforstått? – Brukt og misbrukt? *Nordisk Pedagogikk* nr 2/1998, s. 1-19. https://docplayer.me/21432529-Jean-piaget-9-aug-1896-17-sept-1980-forstatt-og-misforstatt-brukt-og-misbrukt.html

Skolverket. (2019). *Matematik: Ämnets syfte*. Skolverket. https://www.skolverket.se/getFile?file=4206

Skaalvik, E.M & Skaalvik, S. (2018). *Skolen som læringsarena: Selvoppfatning, motivasjon og læring* (3. utg.). Universitetsforlaget.

Solberg, T. T. (2018, 8. januar). NHO bommer om koding i skolen. *Aftenposten, debatt*. https://www.aftenposten.no/meninger/debatt/i/VR1z2l/nho-bommer-om-koding-i-skolen-torstein-tvedt-solberg

Sommerfeldt, S. (2018, 10. juli). *Lær Kidsa Koding lærer byråkrater koding og inviterer til debatt på Arendalsuka*. Lær Kidsa Koding.

https://www.kidsakoder.no/2018/07/10/laer-kidsa-koding-laerer-byrakrater-koding-inviterer-debatt-pa-arendalsuka/

Sphero. (u. å.). *Sphero BOLT Coding Robot*. Sphero. https://sphero.com/products/sphero-bolt

Steenbuch, B. (2016, 10. oktober). Oslo kommune skal lære niåringer å programmere. *Aftenposten, Oslo*. https://www.aftenposten.no/oslo/i/dBV2X/oslo-kommune-skal-laere-niaaringer-aa-programmere

Stigberg, H. & Stigberg, S. (2020). Teaching Programming and Mathematics in Practice: A Case Study from a Swedish Primary School. Sage.

Säljö, R. (2001). Læring i praksis: et sosiokulturelt perspektiv. Cappelen Akademisk Forlag.

Terrapin Logo. (u. å.). *Blue-Bot Family*. Terrapin Logo: Tools for thinking. https://www.terrapinlogo.com/products/robots/blue/blue-bot-family.html

Thagaard, T. (2018). *Systematikk og innlevelse: En innføring i kvalitative metoder* (5. utg.). Fagbokforlaget.

Vatne, B. (2006). Leik. I P. Haug (Red.), *Begynnaropplæring og tilpassa undervisning – kva skjer i klasserommet?* (s. 55-84). Caspar Forlag A/S.

Vygotsky, L. S., Cole, M., John-Steiner, V., Scribner, S. & Souberman, E. (1978). Mind in Society: The Development of Higher Psychological Processes. Harvard University Press.

Vaage, S. (2001). Perspektivtaking, rekonstruksjon av erfaring og kreative læreprosessar: George Herbart Mead og John Dewey om læring. I O. Dysthe (Red.), *Dialog, samspel og læring* (s. 129-150). Abstrakt Forlag.

Wing, J. M. (2006). Computational thinking. Communications of the ACM.

Wood, D., Bruner, J. S. & Ross, G. (1976). THE ROLE OF TUTORING IN PROBLEMSOLVING*. *Journal of Child Psychology and Psychiatry*, 17(2), 89 100. John Wiley & Sons, Ltd.

https://acamh.onlinelibrary.wiley.com/doi/epdf/10.1111/j.1469-7610.1976.tb00381.x

Woods, P. (1986). *Inside Schools. Ethnography in educational research*. Routhledge & Kegan Paul.

Yin, R. K. (2014). *Case Study Research: Design and Methods* (4. utg.). Sage.

Østerås, R. E. & Andersen, K. G. (2020, 13. oktober). Programmering er viktig i dagens skole. *Midtnorsk debatt*. https://www.midtnorskdebatt.no/meninger/ordetfritt/2020/10/31/Programmering-er-viktig-i-dagens-skole-22896702.ece

Vedlegg

Liste over vedlegg:

1. Observasjonsskjema
2. Intervjuguide etter observasjonsøkt
3. Intervjuguide: programmering i begynneropplæringen
4. Førsteutkast av kategorier og koder
5. Informasjonsskriv og samtykkeskjema
6. Intervjuguide NSD

Vedlegg 1 – Observasjonsskjema

<table>
<tr><td>Trinn:</td><td>Elever:</td><td>Jenter:</td><td>Gutter:</td></tr>
<tr><td colspan="2">Andre voksne:</td><td colspan="2">Økt/varighet:</td></tr>
<tr><td colspan="2">Forslag til observasjoner</td><td colspan="2">Kommentarer</td></tr>
<tr><td colspan="2">Lærerens introduksjon til oppgaven(e)?
Hva slags oppgave(r) skal elevene gjøre? (problemløsning, koding, e.l.)
Hva slags programmeringsverktøy?
Hvor lenge skal elevene arbeide med denne oppgaven? (Stasjoner? En oppgave for hele økten? Flere oppgaver?)
Utfordringer (proksimale utviklingssone?)
Eventuelle forventninger (for samarbeid, individuelt arbeid og lydnivå)
Åpne oppgaver?</td><td colspan="2"></td></tr>
<tr><td colspan="2">Samarbeid (grupper/par)
Hvordan legges programmeringsoppgavene opp når elevene skal samarbeide?</td><td colspan="2"></td></tr>
</table>

▪ Er dette en oppgave elevene er vant med fra før av? ▪ Hvordan introduseres oppgaven? ▪ Hvordan blir elevene satt sammen i grupper? - nivådelt - hvor mange – samme kjønn? ▪ Gutt/gutt, jente/jente, jente/gutt?	
• **Selvstendig arbeid** ▪ Hva slags oppgaver skal elevene arbeide med? ▪ Tar elevene i bruk programmeringsobjekter? ▪ Hva oppfordrer læreren elevene til å gjøre når de trenger hjelp?	
• **Lærerens interaksjoner under programmeringsarbeidet** ▪ Når? ▪ Hvor ofte? ▪ Hvem initierer?	
• **Oppsummering av økten** ▪ Tar læreren opp eventuelle misforståelser/feil/rettelser? ▪ Trekkes det frem positivitet rundt prøve/feile/rette?	
• **Andre observasjoner**	

Vedlegg 2 – Intervjuguide etter observasjonsøkt

Dette intervjuet, eller jeg kaller det en slags samtale, handler om programmeringsøkten som ble gjennomført i dag. Spørsmålene er i stor grad hentet fra mine observasjoner og er aspekter jeg ønsker å vite litt mer om. Denne samtalen vil være en slags hjelp for meg, slik at jeg forstår dine valg og hensikten bak dem, og kan få et mer helhetlig bilde på programmeringsøkten som ble utført.

Materialet skal brukes i masteroppgaven min og du vil være anonym. Det vil bli tatt lydopptak av samtalen og opptaket vil bli slettet ved prosjektslutt.

Først noen få «faktaspørsmål».

- Hva slags utdanning har du?
- Hvor lenge har du brukt programmering i undervisningen din?
- Hva tenker du elevene synes om bruk av programmering i undervisningen?

1. Hva er bakgrunnen for valg av oppgave som ble brukt i denne økten?
    a. (hva slags type oppgave? Problemløsning, koding, m.m.)
    b. Hva er årsaken til bruken av (fysiske visuelle verktøy (robot), iPad, analogt, e.l.) som programmeringsmåte/verktøy for akkurat denne oppgaven?
        i. Hva var målet for denne økten?
    c. Har elevene arbeidet med lignende oppgave og programmeringsverktøy tidligere? Kan du fortelle litt om hvordan det ble tatt imot av elevene første gangen de skulle ta det i bruk?

        i. Kan du fortelle litt om hvordan du introduserte oppgaven og programmeringsverktøyet i dagens økt? Var det noe spesielt du tenkte var viktig å formidle til elevene før de satte i gang?

2. (ble det gjennomgått eventuelle feil eller misforståelser på slutten av økten? hvorfor/hvorfor ikke?)
    a. Hva er grunnen for at du hadde en gjennomgang av oppgavene på slutten av økten?
3. (lærerens interaksjon med elevene under arbeidet – lar læreren eleven spørre medelever, tenke selv, prøve å få svar på egenhånd før eleven får hjelp? Hva er hensikten bak det? når er «for tidlig» å hjelpe? Er det forskjell på om elevene har arbeidet med en lignende oppgave før og om oppgaven/metoden er ny for elevene, hvorfor?)
    a. Hva var grunnen til at du ba elevene spørre hverandre om hjelp før de spurte deg?
    b. Hva var grunnen til at du ba eleven prøve selv før han/hun spurte deg om hjelp?
    c. Hva var grunnen til at du tok opp … (et problem med oppgaven, en mulig løsning, noe flere lurte på, e.l.) i plenum under økten?
4. Hvordan avgjør du det faglige nivået på undervisningsøktene med programmering?
    a. Tar du utgangspunkt i elevenes faglige ståsted i «normal undervisning», eller er det andre aspekter du må tenke på når du skal planlegge nivået på en programmeringsøkt?
5. Når du planlegger undervisningsøkten, planlegger du ut ifra læreplan, lærebøker, el. eller ut ifra hva du tenker elevene vil like/ha nytte av?
    a. (Utdype)
6. Hva er grunnen til at du valgte at elevene skulle arbeide (sammen/i par/individuelt) i denne økten?
    a. (par/samarbeid) Er det en grunn for at elevene ble satt sammen slik de gjorde i denne økten, i så fall hvilken? (samme kjønn, forskjellig kjønn?)
    b. (individuelt) Kan du utdype mer om dette?
    c. Måten de arbeidet denne økten, er det noe som varierer fra økt til økt, eller oppgave til oppgave, og er det eventuelt en spesiell grunn for det?

7. Er det noe mer du har lyst til å utdype eller fortelle om fra denne programmeringsøkten?

Vedlegg 3 – Intervjuguide: Programmering i begynneropplæringen

Ut ifra det jeg har hørt fra andre lærere da, som har lyst til å ta i bruk programmering i undervisningen på småtrinnet, så er det mange som lurer på hvor de skal begynne.

1. Hvordan gikk du fram da du skulle innføre programmering for elevene dine for første gang?
    a. Hvor gamle var elevene da? Hadde du/har du eventuelt noen tanker om når man kan ta i bruk programmering første gang?
    b. Husker du hva slags oppgaver eller type programmering du tenkte var hensiktsmessig å starte med?
        i. Hadde elevene noen forutsetninger fra før av, og hvordan kan man eventuelt spille videre på det når kanskje de fleste ikke har vært borti programmering før?
    c. Har du noen tanker rundt grunnen til at man i skolen ofte kaller det for «koding» og ikke «programmering»?
        i. Kan du utype mer?
2. Hva er dine tanker om analog programmering og hva det innebærer?
    a. Hvor tenker du grensa går mellom analog og digital programmering?
        i. Er det for eksempel slik at programmeringen kalles digital dersom man tar i bruk en form for digitalt hjelpemiddel inn i programmeringen? For eksempel Blue-Bot på matter?
3. Når det gjelder valg av programmeringsverktøy for øktene, er det noen du har brukt mer enn andre?
    a. Kan du utdype hvorfor?
    b. Hvordan vil du anbefale å introdusere et nytt digitalt programmeringsverktøy for elevene?
    c. Ville du tatt i bruk programmering på iPad i en programmeringsøkt, hvordan ville du i så fall ha lagt opp økten?

        i. Hva slags apper/programmer er tilgjengelig og tilpasset programmering for de yngste elevene, og hva er dine tanker rundt disse?
        ii. Hvordan ville du ha organisert elevene i arbeidet med programmering på iPad? Par, grupper, individuelt, i stasjoner?
4. Hva er dine tanker rundt gjennomgang av eventuelle misforståelser og feil som kan ha oppstått under økten? Eller en gjennomgang av økta som en helhet?
    a. Hvordan kan det gjøres?
    b. Hva får du som lærer ut av disse gjennomgangene?
    c. Hva håper du elevene får ut av gjennomgangene?
5. Hvordan tenker du at man kan vurdere elevenes programmeringsferdigheter?
6. Kompetansemålet som handler om at elevene skal kunne trinnvise instruksjoner i lek og spill. Hva legger du i sammenhengen mellom programmering, og lek og spill?
    a. Lek er jo et veldig omdiskutert emne i skolen, både med tanke på at mange ønsker det skal bli mer lek i begynneropplæringen og at det skal være lekbasert læring. Ser du på programmering som lekbasert læring?
        i. Kan du i så fall utype hvorfor?
    b. Etter 4. trinn så skal de kunne utforske og beskrive mønster og strukturer i lek og spill. Vil du si at programmering er noe som kan bidra til å oppnå det kompetansemålet for elevene? På hvilke måter?
    c. Hva er dine tanker rundt algoritmisk tenkning i forbindelse med programmering?
7. Vi har jo vært en del inne på det med samarbeid og godt klassemiljø, og jeg merker godt at denne klassen er veldig trygge på hverandre. Kan du fortelle litt om hvordan du la til rette for at det var viktig, og jobbet med det i begynneropplæringen og i programmeringsundervisningen?
    a. Har det en sammenheng med programmeringen, eller er det et resultat av andre aspekter?
    b. Har du noen tanker rundt det om programmering kan bidra til å skape gode relasjoner innad i klassen?
    c. Ser du noen fordeler ved det å jobbe sammen i par eller grupper kontra individuelt, når de programmerer? Kan du utdype mer om hvorfor?

8. Hva slags programmeringsbegreper har det vært fokus på at elevene skal lære seg fra 1. til 4. klasse?
    a. Er det en spesiell grunn for at det er akkurat de som de har lært/skal lære?
9. Det finnes mange ulike programmeringsspråk. Er det ett eller noen du tenker er best å forholde seg til når man skal begynne med programmering med de yngste, og kanskje for lærere som er ukjent med programmering?
    a. Hvordan ser du progresjonen i bruken av programmeringsspråk eller programmering som en helhet etter hvert som elevene har drevet med programmering i en stund?
10. I skolesammenheng så snakker man ofte om at elevene må få nok utfordringer, men samtidig kjenne på mestring, også litt rundt den proksimale utviklingssonen (det de klarer med hjelp i dag klarer de alene i morgen). Har du noen tanker rundt det i programmeringssammenheng?
    a. Er det noe du tenker på når du planlegger øktene, og i så fall, i hvilke sammenheng trekkes det inn?
11. Hvordan vil du si lærerrollen er i en programmeringsøkt?
    a. På hvilken måte forholder du deg til elevene når de arbeider med ulike programmeringsoppgaver?
    b. Hvordan er din rolle når du introduserer programmering i undervisningen, eller en programmeringsoppgave for elevene?
        i. Er det noe du har fokus på å fortelle eller lar være å fortelle?
12. Til slutt: Har du et tips eller noe du ønsker å formidle til de lærerne som ønsker å implementere programmering inn i undervisningen, men som ikke føler de har nok kunnskaper, erfaringer eller forutsetninger til å gjøre det?

Andre spørsmål:

Vedlegg 4 – Førsteutkast av kategorier og koder

Kategori:	Analog programmering	Digital programmering	Relasjoner, samarbeid	Kommunikasjon, sosiale ferdigheter	VFL
Koder:	Grunnleggende	Fysiske, visuelle programmeringsverktøy	Godt klassemiljø	Samarbeid (forutsetni	Oppsummering/samtale etter aktiviteter/økten

	kunnskaper Overførbarhet til digital programmering Forståelse Begreper Lek Tverrfaglighet Algoritmisk tenkning Feilsøking	Blokkprogrammering Lek og spill (læreplanmål) Forståelse og kunnskap fra analog programmering Tverrfaglighet Digitale ferdigheter Algoritmisk tenkning	(forutsetning og resultat) Trygge på hverandre Ønsker at alle mestrer Ler med hverandre ved prøving og feiling Relasjoner bygges gjennom samarbeid med forskjellige elever fra klassen Læreren er veileder	ng og resultat) Spille på hverandres styrker Lytte Styrker klassemiljø og relasjoner Gi tydelige instruksjoner Turtaking	Snakke med elevene Bygge videre på hverandres innspill En pekepinn på elevenes progresjon Vurdering av egen læring og innsats

Vedlegg 5 – Informasjonsskriv og samtykkeskjema

Vil du delta i forskningsprosjektet

«Implementering av programmering i undervisningen i begynneropplæringen

- En kvalitativ studie av suksesskriterier og utfordringer med programmering på småtrinnet»?

Dette er en forespørsel til deg om å delta i et forskningsprosjekt hvor formålet er å undersøke hvordan lærere kan implementere programmering i undervisningen i

begynneropplæringen. I dette skrivet gir jeg deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål

Dette forskningsprosjektet er en masteroppgave som har som formål å undersøke hvordan lærere kan implementere programmering i undervisningen i begynneropplæringen. Dette gjøres gjennom observasjon og intervju av lærer på småtrinnet. Problemstillingen for dette forskningsprosjektet er; «Hvordan kan lærere implementere programmering i undervisningen i begynneropplæringen?».

Hvem er ansvarlig for forskningsprosjektet?

Masterstudent: Siri Dyrhovd Leirvik, tlf.: 46 46 68 10
Veileder: Siv Svendsen, tlf. 90 10 43 86
Universitetet i Sørøst-Norge er ansvarlig for prosjektet.

Hvorfor får du spørsmål om å delta?

Du får spørsmål om dette med bakgrunn i kunnskap og erfaringer med programmering i begynneropplæringen.

Hva innebærer det for deg å delta?

Hvis du velger å delta i prosjektet, innebærer det at du blir observert i undervisningen og intervjuet i etterkant, med utgangspunkt i observasjonene som ble gjort. Det vil dermed foregå korte intervjuer etter hver undervisningsøkt, samt et lengre intervju med fokus på undervisning med programmering generelt på småtrinnene. Det vil bli tatt lydopptak og notater gjennom intervjuet, og enkelte opplysninger om deg vil oppgis. Disse opplysningene omhandler stilling, yrke og utdanning.

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

Jeg vil bare bruke opplysningene om deg til formålene jeg har fortalt om i dette skrivet. Jeg behandler opplysningene konfidensielt og i samsvar med personvernregelverket. Det er jeg og min veileder som vil ha tilgang til opplysningene som oppgis. Navnet ditt

og kontaktopplysninger vil anonymiseres med en gang. Du vil ikke kunne gjenkjennes i prosjektet. Databehandleren jeg tar i bruk for å samle inn, bearbeide og lagre data er UiO Nettskjema, som videre overføres til TSD.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?
Opplysningene anonymiseres når prosjektet avsluttes/oppgaven er godkjent, noe som etter planen er i juni 2022. Ved prosjektslutt vil alle personopplysninger og lydopptak slettes.

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

På oppdrag fra Universitetet i Sørøst-Norge 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:

- Universitetet i Sørøst-Norge ved Siv Svendsen, Universitetslektor.
    - Kan kontaktes på epost siv.svendsen@usn.no eller på telefon: 35 02 63 29.
- Vårt personvernombud: Paal Are Solberg
    - Kan kontaktes på epost paal.a.solberg@usn.no eller på telefon: 35 57 50 53

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
Siri Dyrhovd Leirvik

Prosjektansvarlig

(Forsker)

Siri Dyrhovd Leirvik

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet *Implementering av programmering i undervisningen i begynneropplæringen*, og har fått anledning til å stille spørsmål. Jeg samtykker til:

- ☐ å delta i observasjon
- ☐ å delta i intervju
- ☐ at Siri Dyrhovd Leirvik kan gi opplysninger om meg til prosjektet

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

---

(Signert av prosjektdeltaker, dato)

Vedlegg 6 – Intervjuguide NSD

1. Hva slags utdanning har du?
2. Hvor lenge har du jobbet som lærer?
3. Hvilke fag underviser du i?

4. Hva tenker du om å implementere programmering og koding i undervisningen i begynneropplæringen?
5. Implementerer du programmering og koding inn i undervisningen din? Hvis ja, på hvilke måter?

    a. Hvor lenge har du bruk programmering og koding i undervisningen din?
        i. Implementerte du det annerledes før, enn du gjør nå?
6. Hva legges det til rette for når du planlegger undervisning med programmering og koding?
7. Tar du i bruk fysiske og visuelle programmeringsobjekter, og eventuelt hvilke?
    a. Er det noen programmeringsobjekter du har prøvd, som du ikke tar i bruk i undervisningen i dag? Kan du spesisere hvilke og hvorfor?
8. Hvordan organiserer du elevene i undervisning med bruk av programmering og koding?
    a. Dersom elevene skal samarbeide, hvordan deles de opp i grupper eller par?
    b. Dersom du deler elevene opp i grupper eller par, jobber de med de samme oppgavene, eller arbeider de med forskjellige oppgaver?
        i. Har du noen eksempler på konkrete oppgaver elevene arbeider med i grupper?
        ii. Har du da en lærerstyrt aktivitet, og i så fall hva kan denne aktiviteten innebære?
    c. Dersom elevene arbeider alene, tar de i bruk programmeringsobjekter, eller arbeider de med programmering og koding på andre måter?
9. Når du implementerer programmering og koding i undervisningen, er det tilknyttet et spesifikt fag eller brukes det tverrfaglig?
10. Vil du anta at elevene vet hva programmering og koding er og dens hensikt? I så fall, hva er grunnen til det?
11. Når begynte du med programmering og koding i undervisningen?
12. Har du noen kriterier du følger når du planlegger en programmering og kodings økt? Eventuelt hvilke?
13. Vil du si det er noen utfordringer ved programmering og koding i undervisningen i begynneropplæringen, og kan du spesisere de?
14. Hvilke fordeler ser du med programmering og koding i begynneropplæringen?
15. Hva er dine tanker rund programmering og koding blant de yngste elevene?