Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for lærer- og tolkeutdanning
Institutt for grunnskolelærerutdanning 5-10
og bachelor i tegnspråk og tolking

Hedda Louise Lang-Ree

"Vi må tenke og ikke bare tegne"

En kvalitativ studie om bruk av programmering som verktøy i arbeid med matematikk

Masteroppgave i matematikkdidaktikk 5-10
Veileder: Trygve Solstad
Trondheim, juni 2016

NTNU
Kunnskap for en bedre verden

Hedda Louise Lang-Ree

"Vi må tenke og ikke bare tegne"

En kvalitativ studie om bruk av programmering som verktøy i arbeid med matematikk

Masteroppgave i matematikkdidaktikk 5-10
Veileder: Trygve Solstad
Trondheim, mai 2016

Norges teknisk-naturvitenskapelige universitet
Fakultet for lærer- og tolkeutdanning
Institutt for grunnskolelærerutdanning 5-10
og bachelor i tegnspråk og tolking

NTNU
Kunnskap for en bedre verden

Forord

En masteroppgave er noe helt spesielt, en tid med både latter og tårer, og oppturer og nedturer. Mange ganger har jeg sagt til meg selv at med motgang kommer medgang, og krysset fingrene for at det stemmer. Med mitt ønske om å skrive en annerledes masteroppgave hoppet jeg ut i havet, begynte og svømme rundt og har siden prøvd å komme meg til land. Jeg klarte det tilslutt og har nå en ferdigskrevet masteroppgave i hånden.

Takk til veilederen min Trygve Solstad for innspill, råd og tips når jeg har stått fast og skrivesperren har inntatt meg. Takk til læreren som ga meg muligheten til gjennomføre undersøkelsene mine i klassen sin, og til alle de herlige elevene som tok dette prosjektet i mot på strak arm!

En takk er på sin plass til Tjerand for all inspirasjon, støtte og gode innspill på veien mot den masteroppgaven jeg ønsket å skrive.

Sist med ikke minst, takk til den fantastiske familien jeg har, som alltid støtter meg og kommer med gode med råd.

Hedda Louise Lang-Ree,

Trondheim 18.mai 2016.

Innholdsfortegnelse

1.0 Innledning

I et samfunn hvor teknologien stadig mer preger hverdagen vår, kreves det nye kompetanser som elever i skolen må mestre. Elever som vokser opp i det 21.århundre må undervises annerledes enn de som vokste opp i det 20.århundre, og på bakgrunn av denne tanken har begrepet *21st Century Skills* blitt et uttrykk for hvilken kompetanse elever vil ha behov for i fremtiden. Uttrykket refererer til et bredt sett av kunnskaper, ferdigheter, arbeidsvaner og personlige egenskaper som vil være viktige for å lykkes i dagens og fremtidens samfunn (Abbot, 2014).

I Opplæringsloven står det: *«Opplæringa i skole og lærebedrift skal, i samarbeid og forståing med heimen, opne dører mot verda og framtida og gi elevane og lærlingane historisk og kulturell innsikt og forankring» (Opplæringslova, 1998, s. § 1-1).* I denne oppgaven ønsker jeg å se nærmere på hvordan bruk av teknologi i matematikkundervisningen kan åpne slike dører for elevene i norsk skole.

1.1 Programmering i skolen - en utfordring vi skal ta?

I juni 2015 presenterte Ludvigsenutvalget, et offentlig utvalg utnevnt av Kunnskapsdepartementet, sin utredning av *Fremtidens skole.* Målet med utredningen er å vurdere grunnskolens fag og kompetanser opp imot hva det fremtidige samfunns- og arbeidsliv vil kreve av elevene. Matematikkfaget trekkes spesielt frem i rapporten fordi utvalget mener matematikk inngår og berører de andre skolefagene, og fordi matematisk kompetanse vil være viktig i fremtiden (NOU 2015:8). Flere kompetansemål i læreplan for matematikk fellesfag kan nås med bruk av digitale verktøy i matematikkundervisningen, der programmering kan være et av disse. I Kunnskapsløftet er digitale ferdigheter definert som en av de fem grunnleggende ferdighetene i alle fag. I Norge har digital kompetanse i matematikk først og fremst omhandlet bruk av regneark og grafiske verktøy som Geogebra. Programmering er imidlertid en ferdighet som også passer inn under digital kompetanse og har mange bruksområder innen matematikk.

I flere andre land har innføringen av programmering i skolen kommet lenger enn i Norge, og er nå enten en del av, eller på vei inn i læreplanene. Ikke nødvendigvis kun som en del av et enkeltfag, men like mye som en måte å arbeide tverrfaglig på. 30.januar 2016 introduserte Barack Obama «Computer Science for All» med blant annet ordene: *«In the new economy,*

computer science isn´t an optional skill. It´s a basic skill.» (The White House, 2016). Dette er en plan for hvordan alle elever i USA skal få muligheten til å lære seg programmering i skolen. Samtidig kom Utdanningsdirektoratet med et tilbud, til kommuner og friskoleeiere i Norge, om å delta på et 3-årig pilotprosjekt med programmering som valgfag på ungdomstrinnet fra og med høsten 2016 (Utdanningsdirektoratet, 2016a). Det ble lansert i august 2015, og i april 2016 fikk 146 skoler plass i pilotprosjektet, og programmering er på vei inn i den norske skolen. I Europa har riktignok 15 land allerede innført programmering i skolen siden 2013 på enten nasjonalt, regionalt eller lokalt nivå. Fra og med høsten 2016 blir også Finland med når programmering blir en del av læreplanen helt fra første klasse (Euractiv, 2015).

«If we teach today's students as we taught yesterday's, we rob them of tomorrow», skrev filosofen John Dewey i 1944 (West, 2011). Dette sitatet er kanskje mer aktuelt enn noen gang.

1.2 Problemstilling og forskningsspørsmål

Et programmeringsspråk som er egnet til bruk i undervisning i skolen er Scratch. Det er laget som et pedagogisk verktøy, det krever liten eller ingen forkunnskaper av verken lærere eller elever, og er i tillegg visuelt engasjerende (Resnick et al., 2009). Flere studier indikerer at elever gjennom å bruke programmering i undervisningen, kan utvikle matematiske ferdigheter og forståelse for grunnleggende konsepter i programmering. Det er i de samme studiene kommet frem at Scratch har en positiv innvirkning på elevenes affektive sider. Elevene opplevde arbeidet med programmering i Scratch som motiverende, engasjerende, nyttig og som en positiv opplevelse (Brown et al., 2013; Fessakis, Gouli & Mavroudi, 2012; Sáez-López, Román-González & Vázquez-Cano, 2016; Wilson & Moffat, 2010). Kan programmering berike matematikkundervisningen i norsk skole?

Vi vet lite om hva som er det matematiske innholdet i programmeringsaktiviteter, og hvorvidt arbeid med programmering engasjerer norske elever til matematisk aktivitet. Det er viktig å få et innblikk i dette før man innfører det i skolen, og før man avgjør om det er noe som skal inn i matematikktimene eller andre fag. På bakgrunn av dette ønsker jeg med min oppgave å undersøke bruk av programmeringsspråket Scratch i arbeid med geometriske figurer i matematikkfaget, og se på hvordan arbeid med programmering i matematikken påvirker elevenes affektive sider. I denne oppgaven vil affektive sider referere til elevenes holdninger,

oppfatninger og følelser knyttet til arbeidet med programmering i matematikkundervisningen. Min problemstilling blir som følger:

«Hvilke tilnærminger til matematikk finnes det i arbeid med det blokkbaserte programmeringsspråket Scratch, og hvordan påvirkes elevenes affektive sider av arbeid med programmering?»

For å undersøke problemstillingen har jeg gjennomført en kvalitativ studie av arbeid med geometriske figurer i programmeringsspråket Scratch. Studien har blitt gjennomført i en syvendeklasse, og datamaterialet er innhentet hovedsakelig i form av lydopptak av elevsamtaler og intervju.

1.3 Oppbygging av oppgaven

Innledningsvis presenterer jeg teori som er relevant for oppgaven, og som kan være med på å belyse problemstillingen. Det sosiokulturelle perspektivet på læring, teori som omhandler hva matematikk kan være, og hva det vil si *å gjøre matematikk*, vil bli presentert. Kompetanser i fremtidens skole, *21st Century Skills* og *Computational Thinking* er begreper som vil være relevante å beskrive. I kapittelet gir jeg også en innføring i historien til programmering i skolen, og en beskrivelse av programmeringsspråket Scratch som jeg har valgt å bruke i mine undersøkelser. I analysekapittelet blir deler av datamaterialet i form av utdrag og små utsnitt presentert og videre analysert. Analysen og mine funn belyser jeg med den presenterte teorien i drøftingskapittelet, før det avslutningsvis kommer en konklusjon der trådene samles, videre forskning presenteres, og nye spørsmål stilles.

2.0 Teori

Innledningsvis beskriver jeg grunnleggende ferdigheter i matematikk fellesfag, samt en introduksjon av Ludvigsenutvalgets rapport *Fremtidens skole* fra 2015. Det er relevant å gjøre rede for *hva matematikk* er, og hva det *å gjøre matematikk* kan være. Videre vil det sosiokulturelle perspektivet på læring presenteres, der særlig Vygotsky og språket som medierende redskap vil trekkes frem. Ettersom oppgaven bygger på relasjoner mellom matematikk og programmering, presenterer jeg derfor utviklingen av programmering som innhold i skolen, samt en beskrivelse av programmeringsspråket Scratch og tidligere forskning gjennomført med dette språket.

2.1 Matematikk i dagens og fremtidens skole

2.1.1 Grunnleggende ferdigheter i matematikk

I læreplanen for matematikk fellesfag beskrives det hvordan de fem grunnleggende ferdighetene skal inngå i matematikkfaget. I muntlige ferdigheter i matematikk fellesfag står det: *«Det vil seie å vere med i samtalar, kommunisere idear og drøfte matematiske problem, løysingar og strategiar med andre*» (Utdanningsdirektoratet, 2013, s. 4). Elevene skal gå fra å bruke et enkelt matematisk språk til å bli mer presise i uttrykksmåten og bruken av matematiske begrep. Å skrive i matematikk innebærer å bruke matematiske symboler til å løse problemer og presentere løsninger. Elevene skal gå fra å bruke en enkel uttrykksform til å ta i bruk et mer formelt symbolspråk. Det omhandler også å beskrive og forklare hvordan man har tenkt, og å kunne få frem oppdagelser og ideer. «*Digitale ferdigheter i matematikk inneber å bruke digitale verktøy til læring gjennom spel, utforsking, visualisering og presentasjon. Det handlar òg om å kjenne til, bruke og vurdere digitale verktøy til berekningar, problemløysing, simulering og modellering.»* (Utdanningsdirektoratet, 2013, s. 5). Å regne i matematikk innebærer å kjenne igjen og beskrive matematiske situasjoner, og innebærer også at elevene gradvis skal ta i bruk ulike hjelpemidler i beregning, modellering og kommunikasjon. *«Å kunne rekne i matematikk inneber å bruke symbolspråk, matematiske omgrep, framgangsmåtar og varierte strategiar til problemløysing og utforsking [..]»* (Utdanningsdirektoratet, 2013, s. 5). Å kunne lese i matematikk innebærer å kunne sortere informasjon, analysere og vurdere form og innhold. Elevene skal gå fra tekster med enkelt matematisk symbolspråk til å finne mening i og reflektere over mer komplekse fagtekster med bruk av mer avansert symbolspråk og begreper (Utdanningsdirektoratet, 2013).

2.1.2 Fremtidens skole og matematikkens plass

I juni 2015 kom Ludvigsenutvalget, et offentlig utvalg utnevnt av Kunnskapsdepartementet, med rapporten *Fremtidens skole*. Målet med rapporten var å vurdere hvilke kompetanser elevene i norsk skole vil trenge i det fremtidige samfunns- og arbeidsliv, og med bakgrunn i dette vurdere hvordan fagenes innhold bør fornyes slik at elevene skal kunne utvikle disse kompetansene (NOU 2015:8). I rapporten trekkes det særlig frem fire kompetanser som vil være viktige for at elevene skal lykkes i dagens og fremtidens samfunn. De fire kompetansene er: fagspesifikk kompetanse, kompetanse i å lære, kompetanse i å kommunisere, samhandle og delta, og kompetanse i å utforske og skape, se Figur 1.

Figur 1: Kompetanser i fremtidens skole

Fagspesifikk kompetanse innebærer at elevene utvikler kunnskap om sentrale metoder og tenkemåte, begreper og prinsipper i alle fag. Kompetanse i å utforske og skape inkluderer kritisk tenking og problemløsning, sammen med kreativitet og innovasjon. *«Kreativitet forstås som å være nysgjerrig, utholdende, fantasifull i problemløsning, alene og ikke minst i samarbeid med andre»* (NOU 2015:8, s. 10). Kompetanse i å lære innebærer metakognisjon og selvregulert læring. Elevene skal kunne reflektere over hensikten med det de lærer, det de har lært og hvordan de lærer. I tillegg skal de lære å ta initiativ og arbeide målrettet for å lære som beskrives som selvregulert læring. I rapporten trekkes også dybdelæring inn som et viktig aspekt ved elevenes læring. Dybdelæring handler om å utvikle forståelse for et fagområde eller på tvers av fagområder, samtidig som de tilegner seg kunnskaper og ferdigheter, reflekterer over det de lærer og knytter det til det de tidligere har lært (NOU 2015:8).

Under fagspesifikk kompetanse trekkes *matematikk, naturfag og teknologi* inn som et av de sentrale fagområdene hvor det vil være behov for at elever utvikler kompetanse.

«Utvalget anbefaler at matematikk styrkes i skolen og synliggjøres bedre i fag der matematisk kompetanse er en viktig del av kompetansen, spesielt samfunnsfag og naturfag» (NOU 2015:8, s. 9-10). Matematikk er viet en spesiell plass i fremtidens skole, og det begrunnes blant annet med at matematikk griper inn i alle skolefagene. I samfunnet vil matematisk kompetanse være viktig for å sikre et konkurransekraftig næringsliv og innovasjon på en rekke områder, samtidig som elevene vil ha behov for matematikk både i utdanning, yrkesliv, og i livet generelt. I rapporten presenteres et eksempel på hvordan matematikkfaget kan fornyes. Matematisk kompetanse beskrives med fem komponenter: forståelse, beregning, anvendelse eller strategisk tankegang, resonnering og engasjement. De fem komponentene støtter opp under og er avhengige av hverandre. De er sammenflettet og må utvikles parallelt.

2.2 Et sosiokulturelt perspektiv på læring– mediering, artefakter og tenking

Å være sosial og kommuniserende er av menneskets natur. Sentralt i et sosiokulturelt perspektiv på læring og utvikling er menneskets samspill med språklige og fysiske redskaper. Dette samspillet og vår utvikling og benyttelse av redskaper er det som skiller oss fra andre arter. Disse artefaktene har ikke i seg selv noen kommunikativ funksjon, men i samspill med et tenkende individ kan redskapet bli en del av komplekse og intellektuelle praksiser, det medierer virkeligheten for mennesker i konkrete virksomheter. Dette samspillet er det som skiller det sosiokulturelle perspektivet fra andre teoretiske perspektiver (Säljö, 2001).

Språket ses på som den viktigste av alle medierende redskaper og artefakter. I menneskets kunnskapsbygging i form av å samle erfaringer og å kommunisere disse videre, er språket den viktigste bestanddelen. Gjennom språket vårt, i form av ord og utsagn, medieres verden for mennesket slik at den fremstår som meningsfull for oss. På en og samme tid er språket et individuelt, et interaktivt og et kollektivt sosiokulturelt redskap for mennesket. Dette fører til at språket kan være med på å binde sammen kultur, interaksjon og den individuelle tenkingen til mennesket. Tenking i et sosiokulturelt perspektiv trenger ikke være noe som finner sted kun i individet, det kan like gjerne være en kollektiv prosess som innebærer både samspill men like mye det å lytte til hverandre (Säljö, 2001).

2.3 Hva er matematikk?

I følge Alrø og Skovsmose (2004) lever vi i et samfunn der matematikk og matematisk forståelse har blitt en integrert del av hverdagen vår. Matematikk finnes som et felt for undersøkelse og forskning, som en måte for resonnering, som en kilde og ressurs for teknologi, som tenking i hverdagen, og som et skolefag. Felles for disse formene er at de alle er i sentrum av den sosiale utviklingen til mennesket. Boaler (2008) beskriver matematikk som en menneskelig aktivitet, et sosialt fenomen, et sett av metoder for å kunne belyse verden og som en del av kulturen vår. Matematikk er også en måte å uttrykke forhold og ideer på, både numerisk, grafisk, symbolsk, verbalt og billedlig (Boaler, 2008).

2.3.1 Å gjøre matematikk – Doing Mathematics

Van de Walle, Bay-Williams og Karp (2014) beskriver at det *å gjøre matematikk* handler om å finne passende strategier for å løse ulike problemer, å bruke disse tilnærmingene, se om de leder til noen løsninger, og til slutt for å sjekke om løsningene gir mening. *«Doing mathematics in classrooms should closely model the act of doing mathematics in the real world"* (Van de Walle, Bay-Williams & Karp, 2014, s. 13). Matematikk handler om å finne og utforske regelmessige mønstre og logiske rekkefølger for så å gi det mening. Elever trenger å lære seg grunnleggende kunnskap og ferdigheter, og påpeker at elevene må kunne mer enn bare å pugge og huske for å være i stand til og forberedt på matematikken som vil kreves i det 21.århundret (Van de Walle, Bay-Williams & Karp, 2014).

Van de Walle, Bay-Williams og Karp (2014) beskriver en rekke verb som viser til handlinger der elevene engasjerer seg i å gjøre matematikk, handlinger som gir mulighet til tenking på et høyere nivå og som omfatter det å gi mening og finne ut av ting.

«[…]the following verbs engage students in doing mathematics:

Compare	*explain*	*predict*
Conjecture	*explore*	*represent*
Construct	*formulate*	*solve*
Describe	*investigate*	*use*

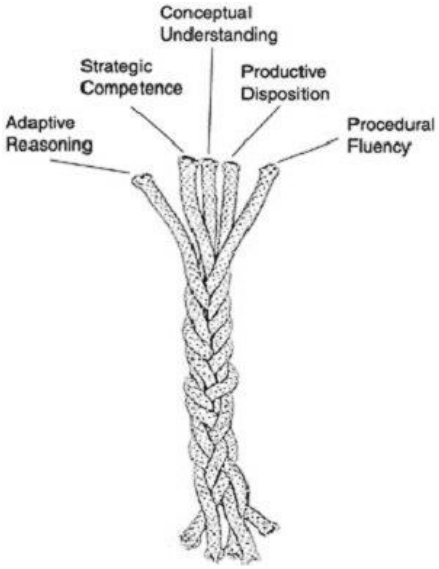
Develop justify verify» (Van de Walle, Bay-Williams & Karp, 2014, s. 14)

I et klasserom der læreren skaper forventninger og etablerer praksiser som oppmuntrer elevene til å ta sjanser, gjette seg frem, resonnere og dele, kan dette bidra til at elevene ser mening i matematikken de lærer. Viktige faktorer i læringen av matematikk er også utholdenhet, innsats og konsentrasjon. I delingen av ideer i klasserommet er alle ideer like viktige, og alle har noe de kan tilføye. Det å lytte til de ulike ideene som kommer frem kan bidra til at elever klarer å strategisk velge gode strategier eller den som passer best. Sammen skal elevene overveie ideene og evaluere om de kommer til å fungere i problemet de skal løse. Det blir påpekt at man skal se på «feil» som muligheter for læring, og prøve å forstå hvorfor ideen ikke ville fungere i problemet de skulle løse. Å gjøre matematikk handler om å reflektere over denne prosessen, og å fange opp og justere feil som elevene oppdager på veien mot mål. Elever ser etter og diskuterer sammenhenger. Elever bør se sammenhenger mellom ulike strategier for å løse et bestemt problem, og også se tilknytning til andre matematiske konsepter og til reelle kontekster og situasjoner. Når elever ser etter og diskuterer sammenhenger ser de matematikken som verdifull og viktig, istedenfor en isolert samling av fakta (Van de Walle, Bay-Williams & Karp, 2014).

Det er viktig å poengtere at det er elevene som skal stå for tenkingen, pratingen og matematikken, noe som krever utholdenhet og innsats. Det er viktig at alle rundt, alle medelever, forstår at det er dette som fører til læring i matematikken. Å gjøre matematiske forhold eksplisitte og engasjere elevene i «Productive Struggle» er det som leder til «Conceptual Understanding». Læreren skal legge til rette for at elevene ser koblinger i oppgavene de gjør. Fokuset skal være på elevenes søken etter grunnleggende kunnskap, testing av ideer, oppdagelse av koblinger og sammenhenger, og utredelse av antakelser. I «Productive Struggle» er begge ordene like viktige. Elevene må ha redskapene og kunnskapen for å kunne løse et problem som gis dersom det skal være produktivt, det må ikke være for enkelt eller utenfor rekkevidde. Dersom elevene vet at det er forventet at de vil streve litt som en del av prosessen av å gjøre matematikk, vil de omfavne strevet og føle suksess når de finner en løsning (Van de Walle, Bay-Williams & Karp, 2014).

2.3.2 Matematikk som en flette av ferdigheter

Mathematical Proficiency, oversatt til matematiske ferdigheter, beskrives av National Research Council (2001) som de aspektene ved matematikk som elever har behov for å kunne for å lykkes med matematikklæringen. Matematiske ferdigheter består av fem komponenter eller tråder, og danner til sammen en flette som symboliserer matematiske ferdigheter. De fem trådene er Conceptual Understanding, Procedural Fluency, Strategic Competence, Adaptive reasoning og Productive Disposition.



Figur 2: Mathematical Proficiency

Conceptual Understanding, eller begrepsmessig forståelse, omfatter matematiske begreper, operasjoner og sammenhenger. Procedural Fluency, oversatt til prosedyremessig kunnskap, er ferdigheter i å utføre prosedyrer fleksibelt, nøyaktig, effektivt og hensiktsmessig. Strategic Competence vil si å ha evne til å formulere, representere og løse matematiske problemer. Adaptive Reasoning innebærer kapasiteten til logisk tenking, refleksjoner, forklaringer og bevis. Productive Disposition er evnen til å se matematikk som fornuftig, nyttig og verdifull, i tillegg til en tro på at innsats lønner seg (National Research Council, 2001)

National Research Council (2001) påpeker at trådene ikke kan ses på som separate, men som en flette som til sammen utgjør matematiske ferdigheter. Det tar tid å utvikle denne fletten av ferdigheter, og den er ikke ferdig selv om to tråder er «fullført». For å utvikle denne fletten trenger elevene god tid til hvert enkelt matematiske tema som læres. Elevene må «gjøre matematikk» over en lengre periode, i form av å løse problemer, resonnere, utvikle forståelse, praktisere ferdigheter og å finne sammenhenger mellom tidligere og ny kunnskap.

2.3.3 Matematikk som utforsking

Inquiry, eller utforsking, vil si å bevege seg fra det sikre til det mer åpne og usikre. Utforskelseslandskap, eller *Landscape of Investigation*, viser til et åpent læringsmiljø, oppgaver erstattet av en kontekst som introduserer landskapet elevene skal bevege seg i. Elevene formulerer spørsmål, planlegger en rute for utforsking og blir en del av en

utforskende prosess preget av elever som spør «Hva om» og «Hvorfor er det sånn?». Aktiviteter kan knyttes til en semi-virkelighet som for eksempel dynamiske geometriprogrammer eller regneark (Alrø & Skovsmose, 2004). Boaler (2008) påpeker at elever må få oppleve matematikken som levende ved å få stille spørsmål og utvide problemer i nye retninger. Dette kan være med på å skape et eierskap og en glede ved matematikken.

Boaler løfter frem problemet med at mange forbinder det å være presis i språket med de såkalte «drill and kill» metodene. Hun mener at det å være presis ikke trenger å henge sammen med puggemetodene. Behovet for presisjon i bruken av begrep og notasjon trenger ikke utelukke åpen og kreativ utforsking i matematikken. Hun påpeker at matematikere gjennom å være presise i bruken av språk, symboler og diagram tillater dem å fritt utforske ideer. Boaler påpeker at om elever bare kan få lov til å arbeide litt som matematikere, ved å komme opp med problemer, gjette og gjøre antakelser, utforske og diskutere ideer med andre vil de få muligheten til å finne en glede ved matematikken og lære den på den mest produktive måten (Boaler, 2008).

2.3.4 Matematikk som forståelse

Skemp (1976) poengterer at dobbeltbetydningen til ordene forståelse og matematikk kan være roten til mange misforståelser og utfordringer i matematikkundervisningen. Det undervises i både relasjonell og instrumentell forståelse i skolen og i matematikkundervisningen. For noen er forståelse det å kunne bruke en algoritme og få rett svar, ikke å forstå hvorfor akkurat den algoritmen kan tas i bruk og hvorfor den gir korrekt svar. I undervisningen av relasjonell matematikk vil elevene få en forståelse for hvorfor man gjør som man gjør, som vil føles mer givende, ha en egenverdi og kan føre til at elevene frivillig ønsker å fortsette læringen. Instrumentell matematikk kan derimot preges av pugging av algoritmer og regler, som for en del fort kan oppleves som kjedelig. Med en gang en oppgave krever litt mer enn bare pugging, vil det for mange bli veldig vanskelig. På en annen side understreker Skemp (1976) at begge «typene» matematikkundervisning vil kunne ha sine fordeler, og kanskje også komplementere hverandre.

Tradisjonelt sett har matematikkfaget vært dominert av forventninger om at elevene skulle kjenne til og kunne bruke en rekke faglige begreper og ferdigheter. Med tiden har forventningene utviklet seg til også omfatte at elevene skal kunne undersøke, beskrive, forklare og forutsi sammenhenger og mønstre i matematikken. Fra å ha brukt mye tid og energi på produktsiden ved matematikken, har det blitt mer og mer fokus på fagets prosesser og det å skape matematikk. For elever som utelukkende får matematikk presentert i form av at det er trening av begreper og prosedyrer, vil dette kun avspeile resultater av matematisk aktivitet, ikke aktiviteten eller prosessen som førte dit (Skott, Jess & Hansen, 2008).

2.3.5 Papert, MathLand og Microworlds

Papert (1980) beskriver programmeringsspråket LOGO som en verden der elever kan utforske deler av matematikken på ved å tre inn i en «microworld». Ideen om å prate matematikk til en datamaskin kan ses på som læring av matematikk i et «MathLand». Papert tar opp to fundamentale ideer i boken «Mindstorms». Den ene omhandler at det er mulig å designe datamaskiner slik at det å lære og kommunisere med dem kan bli en naturlig prosess for elevene. Når denne kommunikasjonen oppstår vil barna lære matematikk som et levende språk. Han bruker det å lære seg fransk ved å bo i Frankrike som en sammenligning med den mer unaturlige prosessen med å lære fransk som andrespråk i et annet land. Den andre ideen er at det å lære å kommunisere med en datamaskin, kan endre måten annen læring skjer på ved at datamaskinen kan være en matematikk-pratende enhet. Det handler om å kommunisere med en datamaskin på en språk som både elevene og datamaskin kan forstå. Alle barn lærer å prate, og å lære språk er noe av det barn gjør aller best. På bakgrunn av dette mener Papert (1980) at det ikke er noen grunn for at elevene ikke skal lære å prate med en datamaskin. Han trekker også frem at når barn vokser opp i en teknologirik verden vil dette føre til at det som før har blitt sett på som for matematisk og formelt for yngre elever, læres enkelt.

I programmering er det å «debugge» en viktig del av prosessen når man skriver instruksjoner og kommandoer til en datamaskin. Å se etter feil, rette de opp, og på den måten finne en bedre løsning på problemet. Å bruke feil eller misoppfatninger til læring er viktig i programmering, og det påpekes at det er viktig å ikke kritisere elevene for feil, men å i stedet snu det til noe positivt vi kan lære av. Papert mener LOGO er et sted der elever ikke blir «dømt, et sted der det er lov til å ta feil (Papert, 1980).

2.4 Ferdigheter i det 21.århundret

2.4.1 Computational Thinking

Computational thinking handler ikke om å tenke som en datamaskin, men innebærer å bryte ned store komplekse problemer i mindre, mer håndterbare problemer som vi kan klare å løse. Denne metoden for problemløsning bygger på de grunnleggende konseptene som vi finner i programvareutvikling, men er også en problemløsningsprosess som kan brukes i mange andre sammenhenger og i ulike fagfelt (Wing, 2006). The International Society for Technology in Education (ISTE) og The Computer Science Teachers Association (CSTA) har sammen utviklet en definisjon for hva Computational Thinking er, og hvordan K-12, som kan sammenlignes med vår grunnskole, kan tilnærme seg denne problemløsningsprosessen i skolen. Computational thinking beskrives som en problemløsningsprosess hvor man gjennom logisk tenking organiserer og analyserer data, finner mulige løsninger, for så å generalisere og overføre denne prosessen til andre problemer (The International Society for Technology in Education & The Computer Science Teachers Association, 2011). The Barefoot Programme, et program som støtter lærere i grunnskolen med undervisning av IKT og som skal bidra til at elever blir «computational thinkers», har definert seks konsepter og fem tilnærminger til computational thinking. De seks konseptene er logikk, algoritmer, dekomposisjon, mønstre, abstraksjon og evaluering, og de fem tilnærmingene er eksperimenterende, skapende, feilsøkende, utholdende og samarbeidende (The Barefoot Project, 2014).

2.4.2 Twentyfirst Century Skills - ferdigheter for fremtiden

The Partnership har, sammen med innspill fra utdannere, utdanningseksperter og forretningsledere utviklet det som kalles «The Framework for 21st Century Learning». Dette er et rammeverk som definerer og illustrerer hvilke ferdigheter, kunnskaper, ekspertiser og støttesystemer elever vil trenge for å lykkes i jobb, generelt i livet og som medborgere og samfunnsborgere (Greenhill, 2009). Lærings- og innovasjonsferdigheter, informasjon-, media- og teknologiferdigheter, liv- og karriereferdigheter, og sosiale og flerkulturelle ferdigheter er de fire hovedkategoriene. Spesielt er det viktig å poengtere at det under lærings- og innovasjonsferdigheter nevnes kreativitet og innovasjon, kritisk tenking og problemløsning, og kommunikasjon og samarbeid (Greenhill, 2009).

For å lykkes i livet og i jobb i det 21.århundret kreves det nye ferdigheter av studentene. I møte med endringene som skjer i samfunnet er utdanningsinstitusjonene nødt til å endre på standarden og vurderingene som foregår i utdanning og skole. Med utgangspunkt i en analyse av tolv allerede eksisterende rammeverk for 21st century skills fra en rekke land i verden, har det blitt utarbeidet et forslag til hvordan man kan tenke når man skal vurdere de nye ferdighetene som kreves av elevene, og hvordan man skal tenke når ferdighetene skal vurderes. De ti ferdighetene er delt inn i fire kategorier: hvordan vi tenker, hvordan vi arbeider, verktøyene vi bruker når vi arbeider og hvordan vi lever i verden (Binkley et al., 2012).

Figur 3:Ferdigheter som elevene må mestre i det 21.århundre. Illustrert med bakgrunn i Binkley et al. (2012).

2.5 Programmering – et overblikk

Det å programmere vil si at man setter sammen en rekke instrukser som styrer f.eks. en datamaskin og bestemmer hvordan den skal reagere på tastetrykk, musebevegelser m.m. (Rossen, 2015). I min forskning vil det være programmeringsspråket Scratch det fokuseres på, et blokkbasert programmeringsspråk utarbeidet av Massachusetts Institute of Technology (MIT).

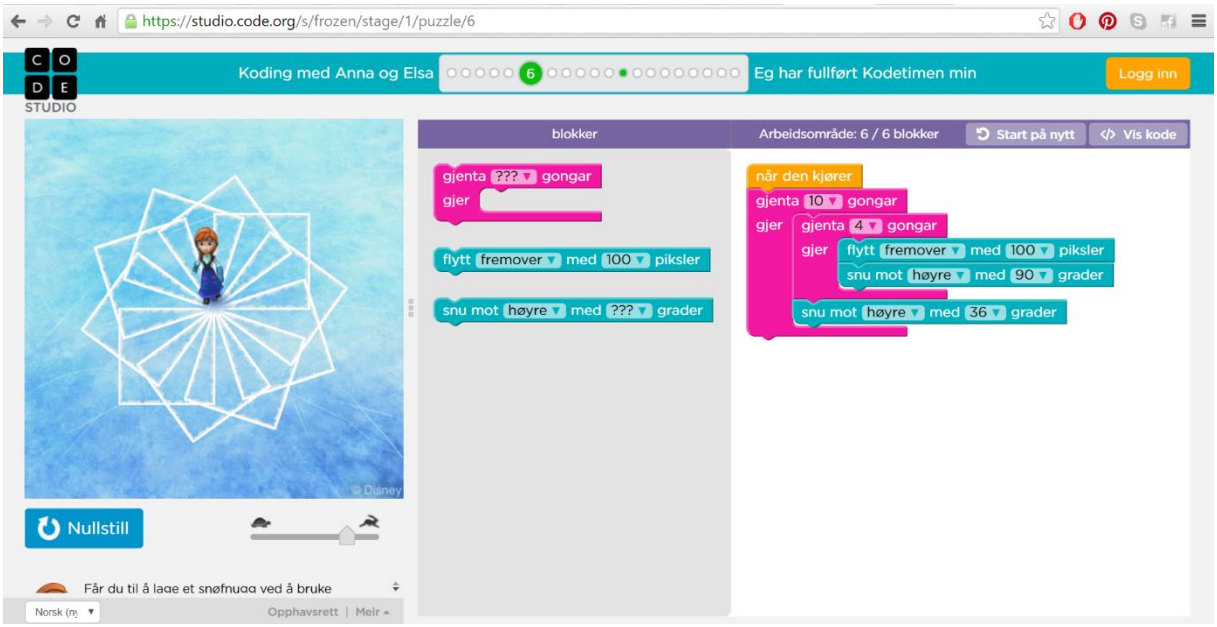
2.5.1 Programmering i skolen – et historisk tilbakeblikk

Programmering i skolen begynte på 1980-tallet med programmeringsspråkene LOGO, Basic og Pascal (Kafai & Burke, 2015). Alle de tre programmeringsspråkene bygget på et konstruktivistisk læringssyn, hvor programmeringen ga elevene umiddelbar respons på arbeidet de gjorde. Elevene hadde ofte en time eller to med programmering på en lab, men som et selvstendig fag uten tilknytning til de andre fagene de ble undervist i. På 1990-tallet ble det færre og færre skoler som underviste i programmering på verdensbasis. Dette, ifølge Kafai og Burke (2015), på bakgrunn av vanskeligheter med å integrere og knytte fagstoff til arbeidet, samt mangel på kvalifiserte lærere og instruktører. Spørsmålet rundt formålet med å lære programmering var også en faktor som var med på å skape denne trenden.

Med Internettet på midten av 1990-årene ble elevene lært opp til å søke på og i nettsider, finne informasjon på en rask måte osv. Men det var ingen opplæring i hvordan alt fungerte. De siste årene har programmering begynt å tvinge seg inn på skolen i form av at elevene bruker teknologi utenfor skolen. De er kreative, de samhandler og de bruker ulike plattformer, apper og spill. Derimot mangler de forståelse for hvorfor et musetrykk fører til en handling på skjermen. Vi har kommet til et punkt hvor dette er nødvendig å forstå for å holde tritt med samfunnets forandring og utvikling (Kafai & Burke, 2015).

2.5.2 Blokkbasert programmering – et sett blokker blir til et program

I blokkbasert programmering er instruksjonene representert ved klosser istedenfor tekst. Man programmerer ved å sette sammen klosser til blokker, der blokkene representerer programmet man har laget. I CodeStudio på code.org sine nettsider finnes det flere opplegg som baserer seg på blokkbasert programmering, blant annet et Frost-opplegg, se Figur 7. CodeStudio sitt programmeringsspråk er veldig likt Scratch, men er en mer forenklet og begrenset versjon og som derfor er en fin introduksjon til Scratch.

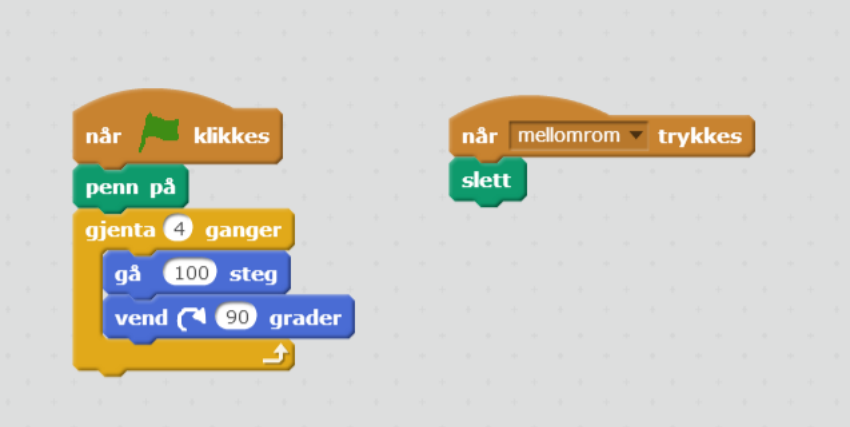


Figur 4: Frost-opplegg i CodeStudio. Skjermbilde hentet fra code.org/frozen.

Programmeringsspråket Scratch er utviklet av «Lifelong Kindergarten Group» ved Massachusetts Institute of Technology Media Lab, og er et blokkbasert programmeringsspråk. Et program laget i Scratch, som for eksempel en interaktiv fortelling, en animasjon, eller et spill, er representert med blokker bestående av klosser. En kloss er i Scratch en instruksjon som for eksempel «vend *x* grader» eller «gå *x* steg». En blokk er i Scratch det programmet man lager, og består av klosser som er satt sammen til det som blir en blokk, se Figur 5.

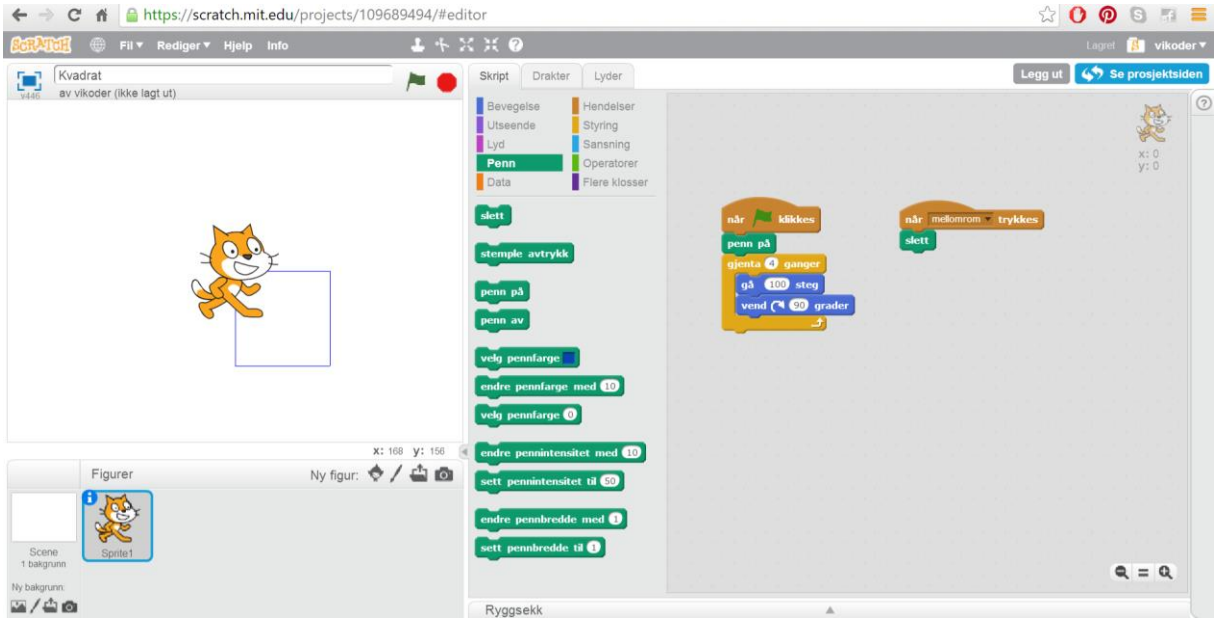
Scratch ble utviklet med mål om å lage et programmeringsspråk som skulle gjøre det enkelt for alle å programmere og dele egne interaktive fortellinger, spill, animasjoner og simuleringer. Målet var at verken alder, bakgrunn eller interesse skulle være en hindring (Resnick et al., 2009). Papert (1980) poengterte at et programmeringsspråk måtte være enkelt å ta i bruk, samtidig som det skulle gi muligheter for å skape komplekse prosjekter over tid. Med Scratch ønsket utviklerne å gjøre terskelen for å programmere enda lavere, samtidig som det fortsatt skulle støtte oppunder læringen av de grunnleggende konseptene i programmering. Dette for at det skal være enklere å lære seg programmering og for at man skal kjenne igjen kommandoer når man går videre til tekstbasert programmering. En kloss som i blokkbasert programmering heter «gå *x* steg» kan i for eksempel JavaScript skrives *moveForward(x);*. Med dette som mål etablerte de tre hovedprinsipper for designet til Scratch: det skulle være

mer utforskende, mer meningsfylt og mer sosialt enn tidligere programmeringsspråk (Resnick et al., 2009).



Figur 5: Klosser blir satt sammen til blokker – her tegnes et kvadrat.

Figur 6 viser hvordan Scratch ser ut i nettleseren. Skjermen er delt inn i fire områder: et bibliotek med alle klossene, en oversikt over bakgrunner og figurer, et område der du setter sammen klossene til blokker, og et felt der du kan se hva som skjer når du «kjører» programmet du har laget. Ulike figurer, bakgrunner og lyder kan hentes fra Scratch sin egen nettside, lastes opp fra andre nettsteder eller lages selv. Scratch har blitt en arena for kreativitet, læring, mestring og deling.



Figur 6: Scratch i nettleseren. Skjermbilde fra eget Scratch-program

I delkapittel 3.3.1 i metodekapittelet finnes det en detaljert beskrivelse av hvordan CodeStudio og Scratch ble tatt i bruk i arbeid med geometriske figurer.

2.6 Tidligere forskning på Scratch

Meerbaum-Salant, Armoni og Ben-Ari (2013) påpeker at Scratch kun er et verktøy og at det fortsatt vil kreve gode undervisningsmetoder og læringsmateriale for et maksimalt læringsutbytte. I studien fant de ut at elever kun lærte når de eksplisitt ble undervist om konseptene, samtidig som de lagde prosjekter der konseptene skulle bli brukt. Et annet funn var at det var viktig med tett oppfølging underveis i arbeidet, fordi en god del elever hadde lett for å bruke Scratch som et verktøy for å skape animasjoner og spill dersom de ble overlatt til seg selv, og av den grunn ikke lære de konseptene læreren hadde forestilt seg. Av affektive sider fant de at kurset hadde en positiv effekt på studentenes motivasjon og interesse for programmering. I intervjuer fortalte studentene at Scratch hadde endret deres oppfatning av programmering, og påvirket avgjørelsen om å fortsette med det på videregående skole (Meerbaum-Salant, Armoni & Ben-Ari, 2013).

Wilson og Moffat (2010) viser til en studie av bruk av Scratch på barnetrinnet, der elevene skulle evaluere øktene med tre ulike ansikter: fornøyd, nøytralt eller kjedelig. Det kom tydelig

frem at elevene likte disse øktene, da det var ingen som markerte på kjedelig og kun noen få hver økt som markerte på nøytralt ansikt. Læreren viste samme engasjement under intervju og hadde sett de positive sidene ved Scratch. Elevene lo mye, de ønsket å vise hverandre det de hadde laget og ønsket også at læreren skulle se. Det bør bemerkes at det var noen av de faglig svake elevene som gjorde det bedre enn forventet, imens noen av de faglig sterke elevene presterte under forventet. Hvorfor det ble slik vet ikke forskerne (Wilson & Moffat, 2010).

Med studien konkluderte Wilson og Moffat (2010) at de emosjonelle eller affekive sidene var like viktige som det å lære de grunnleggende konseptene i programmering. Affekt er viktig for læring og ikke bare som noe ekstra som følger med. Læring vil neppe utvikle seg uten motivasjon, men kan opprettholdes med positive affekter. Forskerne poengterer at det er vanskelig å vite hvor representativt dette engasjementet er, kanskje er det fordi Scratch var helt nytt. Samtidig var de tilstede i klassen over åtte uker, med en time per uke, noe som er en god del tid. Varierende opplegg og litt mer avansert opplegg for hver gang mener de at kan ha hatt positiv effekt.

Meerbaum-Salant, Armoni og Ben-Ari (2011) stiller spørsmål om læring med og i Scratch, etter en studie gjort i 2011, kan være potensielt skadelig for å lære programmering vellykket. Forskerne oppdaget at det kan oppstå vaner når elever bruker Scratch som kan være til hinder for videre læring. Funnene viste at elevene fant frem alle blokkene de trengte for å løse oppgaven, for så å finne ut hvordan de skulle sette de sammen til en kode. Den andre vanen de oppdaget var at elevene brøt ned problemene de skulle løse i så små deler at skriptene til slutt var så små at de ikke ga mening logisk sett. Dette førte til at elevene endte opp med en hel rekke skript, som er helt unødvendig for å løse oppgaven. Det ble vanskelig å lese og forstå hva elevene har gjort og hva de prøvde å formidle. Forskerne ble på bakgrunn av disse to funnene redde for at disse vanene blir med videre når mer avansert programmering introduseres og skal læres. Til tross vanene var forskerne likevel fornøyde med elevenes innsats og engasjement i programmering gjennom bruk av Scratch, og med de tekniske ferdighetene som de utviklet (Meerbaum-Salant, Armoni & Ben-Ari, 2011).

Malan og Leitner (2007) med en studie ved Harvard Summer School´s Computer Science S-1: Great Ideas in Computer Science, foreslår Scratch som en inngangsport til

programmeringsspråk som for eksempel Java. Scratch engasjerer studentene i det kritiske tidspunktet der alt ved programmering er nytt, og gjør dem kjent med mange av de grunnleggende konseptene ved programmering (Malan & Leitner, 2007).

Sáez-López, Román-González og Vázquez-Cano (2016) gjorde en studie på 107 elever på fem barneskoler i Spania. De fant at mange elever ofte synes programmering er vanskelig og forvirrende, noe som kan gjøre at de lett gir opp. De observerte fordeler med å bruke programmering i en undervisningskontekst ved siden av programmeringskunnskapen. Både studenter og forskere påpekte at å arbeide med visuell programmering gjennom prosjekter fører til lek, motivasjon, entusiasme og engasjement fra studentenes side. Det ble også gjort positive funn når det gjaldt en aktiv læring i form av kommunikasjon og deltakelse, fagbegreper, grunnleggende konsepter, følelsen av at det var nyttig og lærerikt, og morsomt (Sáez-López, Román-González & Vázquez-Cano, 2016).

I en studie gjort av Fessakis, Gouli og Mavroudi (2012) på førskoleelever, ble det konkludert med at elevene på få uker med Scratch utviklet problemløsningsstrategier, matematiske ferdigheter i form av forståelse for vinkler, grader, telling og orientering i «rutenett». I tillegg utviklet de ferdigheter i å kommunisere og samarbeide. Brown et al. (2013) gjennomførte en studie der formålet var å undersøke bruken av Scratch på ungdomskolen for å lære ferdigheter som trengs i problemløsning. Funn som ble gjort viste at studentene, til tross for at de ikke fikk bruke datamaskinen og programmere i Scratch i så mye tid av gangen, viste et tydelig ønske og engasjement til å delta i programmeringsøktene.

3.0 Metode

I løpet av undersøkelsene mine har jeg tatt en rekke metodiske valg som har påvirket datainnsamlingen og analysemetoden, og dermed fått konsekvenser for studien. I dette kapittelet presenterer og begrunner jeg disse valgene ved å beskrive datainnsamlingen, gjennomføringen og analysemetoden. Avslutningsvis reflekterer jeg over etiske overveielser og gyldigheten av resultatene.

3.1 Metodisk tilnærming

Med en hypotese om at programmering kan berike matematikkundervisningen, og et ønske om å undersøke dette ute i skolen, ble det naturlig å ta i bruk en kvalitativ tilnærming. For å undersøke problemstillingen min hadde jeg behov for et rikt og beskrivende datamateriale for å få frem elevenes tanker, meninger og opplevelser om arbeidet med matematikk i programmeringsøktene (Patton, 1987; Postholm, 2005; Tjora, 2012). Jeg fant jeg det mest hensiktsmessig å ta lydopptak av samtaler og enkelte intervju, i tillegg til feltnotater av observasjoner og egne refleksjonsnotater. Jeg valgte å gjennomføre undersøkelsene mine i én enkelt klasse i deres respektive klasserom, for så å se på hvordan elevene arbeidet sammen med matematikk i programmering. Studien og undersøkelsene ble formet av klasseromssituasjonen, og jeg valgte å ha en induktiv tilnærming til transkripsjonene av lydopptakene (Creswell, 1998; Postholm, 2005; Tjora, 2012).

3.2 Innsamling av datamateriale

Jeg har i min studie valgt å anvende meg av lydopptak og feltnotater for å samle inn datamateriale. Dette er lydopptak av elevsamtaler under arbeid med programmering, samt intervjuer av enkelte elevgrupper etter endt programmeringsøkt. Jeg har i tillegg lydopptak av en fellesøkt som prosjektet ble avsluttet med og feltnotater av observasjoner jeg har gjort meg underveis. Videre har lydopptakene blitt transkribert, og feltnotatene er blitt skrevet på en ryddig måte. I dette avsnittet beskriver jeg konteksten for studien og de ulike innsamlingsmetodene som har blitt brukt i undersøkelsene.

3.2.1 Utvalg og kontekst

Jeg har i min studie foretatt et kriterieutvalg som kjennetegnes med at undersøkelsene foregår i dybden av relativt få strategisk utvalgte enheter, og at det er satt visse kriterier til informantene som deltar. Informantene undersøkes uavhengig av en allerede eksisterende avgrensning, der formålet med studien er knyttet til deltakernes erfaringer og opplevelser. Utvalget mitt består av en skoleklasse med elever som uavhengig av min studie går i denne klassen på denne skolen. Valg av skole og klasse ble gjort på grunnlag av behovet for en lærer som var villig til å sette av en god del matematikktimer, og behovet for å ha nok datamaskiner tilgjengelig til hver økt. Utvelgelse på grunnlag av hensiktsmessighet er i kvalitativ datainnsamling det som kalles for strategisk utvelging (Johannessen & Tufte, 2002). På disse premissene tok jeg kontakt med en lærer som hadde interesse for bruk av digitale verktøy i undervisningen, og som derfor ga meg stor frihet når det gjaldt antall matematikktimer jeg hadde behov for. Jeg gjennomførte innsamlingen av datamateriale i en syvendeklasse på en skole i Sør-Trøndelag bestående av 26 elever, hvorav 14 jenter og 12 gutter.

Vi hadde ved alle timene to klasserom med SmartBoard, samt to grupperom og korridor tilgjengelig. Tilgangen på SmartBoard gjorde at det var enkelt å gjennomføre felles gjennomganger med alle elevene, og muligheten for å kunne spre elevene skapte gode forhold for lydopptak. Den første økten ga en liten pekepinn på hvordan oppleggene jeg hadde forestilt meg og planlagt fungerte, hvordan elevene samarbeidet og hvordan kvaliteten på lydopptakene ble. Jeg valgte bevisst å ikke ha noen pilot, da jeg ønsket at dette skulle være et helhetlig prosjekt fra start til slutt.

Inndelingen av elever i grupper ble gjort av elevene selv, med eneste kriterium at de skulle være to til tre elever per gruppe, og at disse gruppene skulle gjelde for hele prosjektet. Ved å la elevene velge grupper selv gjorde dette at jeg ikke fikk noen innvirkning på sammensetningen. Mer detaljerte beskrivelser av elevgruppene som er med i analysekapittelet finnes som vedlegg 3. Disse beskrivelsene er gjort på bakgrunn av mine egne oppfatninger av elevene i løpet av øktene, i tillegg til innspill fra elevenes lærer underveis i prosjektet.

3.2.2 Intervju

En grunnleggende form for menneskelig samspill er samtalen, og gjennom denne kan man få kjennskap til personers opplevelser, følelser, holdninger og den verdenen som de lever i. Forskningsintervjuet er en profesjonell samtale som bygger på dagliglivets samtaler. Kunnskapen konstrueres i samspillet eller interaksjonen med den som intervjuer og den som intervjues. Målet til det kvalitative forskningsintervjuet er å prøve å forstå verden ut fra intervjupersonens synspunkter, og på den måte knytte mening til opplevelser og å avdekke personens livsverden (Kvale & Brinkmann, 2009).

Da jeg skulle intervjue elevene hadde jeg på forhånd tenkt ut noen temaer jeg ønsket at de skulle komme inn på, men jeg var også åpen for at de kunne ta opp andre temaer. Dette er det Kvale og Brinkmann (2009) kaller for et semistrukturert intervju. Slike intervju har bakgrunn i en forhåndslaget intervjuguide der bestemte temaer er valgt ut, og hvor det også finnes forslag til spørsmål som kan stilles. Intervjuene av elevene foregikk i gruppene som de hadde arbeidet i. Hensikten med dette var å få elevene til å føle seg trygge under intervjuene ved å skape en naturlig kontekst rundt situasjonen.

3.2.3 Observasjon

Alle mennesker gjør observasjoner, men ikke alle har et bestemt fokus for sine observasjoner, slik en forsker har. En kvalitativ forsker ønsker å observere et fenomen i dets naturlig kontekst eller setting, og selv om forskeren har et fokus, har den ikke nødvendigvis forhåndsbestemte kategorier den ser etter (Postholm, 2005). Mine observasjoner foregikk i klasserommet der elevene til vanlig har undervisning, og ble skrevet som feltnotater. I kvalitativ forskning vil forskeren, til tross for tidligere kunnskap og antakelser, forsøke å være induktiv i observasjonen og dermed åpen for at andre temaer og fokus enn de som var tenkt ut på forhånd, kan dukke opp underveis. Dette vil føre til at det alltid vil være en kontinuerlig interaksjon mellom deduksjon og induksjon i slike studier (Postholm, 2005).

Observasjoner er ulike i den grad av hvor deltakende forskeren er, altså hvordan forskeren forholder seg til deltakerne underveis i undersøkelsene (Gold, 1958). I mine undersøkelser visste elevene hvorfor jeg var i klasserommet og hva formålet med oppleggene og prosjektet

var. Under observasjonen ga jeg i blant noen grupper litt hjelp, men foruten dette var jeg ikke deltakende i undersøkelsene som ble gjort. Gold (1958) kaller denne rollen for Participant-as-Observer eller som av Postholm (2005) er oversatt til observerende deltaker. Rollen kjennetegnes med at observatøren har innsamlingen av datamateriale som første prioritet, og at ønsket med en slik rolle er å utforske forskningsdeltakernes perspektiv uten å bli en fullstendig deltaker selv (Postholm, 2005). Med en slik rolle hadde jeg mulighet til å gå rundt i klasserommet og observere alle elevene, og på den måten samle inn datamateriale i tillegg til det som ble fanget opp gjennom lydopptakene som ble gjort. Tjora (2012) påpeker at på bakgrunn av at man befinner seg i en situasjon som utspiller seg der og da, kan man ikke alltid ha full kontroll på sin egen rolle. Gjennom at man er tilstede i situasjonen som utspiller seg kan det dukke opp uventede situasjoner man er nødt til å ta tak i, der ulik involvering og interaksjon kan oppstå. Dette er det Tjora (2012) kaller *interaktiv observasjon*, forskeren kan inngå i interaksjon som for eksempel samtale og interaksjon. Dette passer inn i min egen rolle i undersøkelsene, da jeg var nødt til å gripe inn da det oppstod problemer med programmeringen fordi dette var noe nytt for elevene.

3.3 Oppbygging av prosjektet

Min datainnsamling foregikk over seks dobbelttimer i løpet av litt under to måneder, på samme skole, i samme klasse og med samme elever. I alle øktene var det matematiske temaet geometriske figurer og deres egenskaper. Oppleggene var laget og planlagt på forhånd av undervisningsøktene, men det vil ofte skje endringer da en økt sjelden blir som man har sett for seg. Min prioritet gjennom hele prosjektet var å ha elevene i fokus, og å tilpasse oppleggene etter hvor langt de kom for hver gang. Dette førte til at mine opplegg og økter endret seg kontinuerlig, noe som var svært lærerikt.

Den første økten var en introduksjon av blokkbasert programmering med Frost-opplegget i CodeStudio, da programmering var nytt for elevene. De neste øktene arbeidet elevene med ulike oppgaver, hvor de hver gang jobbet i grupper på to eller tre. Hver økt ble avsluttet i plenum med noen tanker om hva de hadde gjort. Den siste økten var litt annerledes og bestod av en økt med felles programmering på SMARTBoard hvor alle elevene i klassen deltok. Underveis i alle øktene observerte jeg elevenes arbeid og tok notater. De seks øktene blir sett på som et helhetlig prosjekt, se Figur 8, og en mer utfyllende beskrivelse finnes i Vedlegg 1.

Prosjektet som helhet, hadde som overordnet mål at elevene skulle arbeide med geometri i Scratch, og på denne måten få en forståelse for betydningen av å kjenne til egenskapene til ulike geometriske figurer.

Figur 7: Oversikt over de ulike programmeringsøktene

Etter fire av øktene ble det foretatt intervju av enkelte elevgrupper for å få innsikt i deres tanker om det de hadde arbeidet med. Dette for å kunne komme inn på temaer jeg på forhånd hadde sett meg ut, samt å la elevene samtale fritt om sine opplevelser og erfaringer, da jeg finner dette som verdifull informasjon for å få enda bedre beskrivelser. I etterkant av alle øktene har jeg laget et undervisningsopplegg som senere kan tas i bruk, og finnes som Vedlegg 2.

3.3.1 Scratch – et programmeringsspråk med mange muligheter

I oppstarten av prosjektet med programmering som verktøy i arbeid med geometri, tok vi i bruk Frost-opplegget i CodeStudio, som man finner på code.org sine nettsider. Programmering i CodeStudio ligner på Scratch, men består av opplegg der elevene må løse en rekke spesifikke utfordringer for å komme videre til neste oppgave. I oppgavene skal elevene, ved hjelp av å sette sammen klosser med instruksjoner, få Anna og Elsa til å tegne snøkrystaller på skjermen. I arbeidet med å lage snøkrystaller ble elevene nødt til å arbeide med geometri i form av vinkler og lengder.

Etter denne introduksjonen hadde elevene fått testet hvordan klosser kan settes sammen til blokker, og hvordan dette kan knyttes til matematikk. Disse erfaringene kunne de ta med seg videre da vi i de etterfølgende programmeringsøktene tok i bruk Scratch for å arbeide med geometriske figurer og deres egenskaper.

Programmeringsspråket Scratch ble valgt på bakgrunn av at det er et programmeringsspråk som er relativt enkelt å ta i bruk når man ikke har arbeidet med programmering tidligere. Dette fordi det bygger på bruk av klosser og blokker, og ikke bare tekst. I Scratch kan elevene sette sammen klosser slik at Scratch-figuren tegner den geometriske figuren de ønsker. Dette programmeringsspråket er spesielt godt egnet til å arbeide med geometri fordi det er visuelt, og har et innebygd bibliotek med klosser for å behandle lengder, vinkler, retning, koordinater, rotasjon og matematiske operasjoner. Når elevene drar sammen klosser til blokker er de nødt til å ta i bruk kunnskap om egenskaper til ulike geometriske figurer for at de skulle bli tegnet riktig. Det var for eksempel ikke nok å bare vite hvordan et kvadrat ser ut, de måtte også vite at det har fire like lange sider med en 90 grader vinkel i hvert hjørne. Med Scratch kan elevene prøve og feile, oppdage sammenhenger mellom geometriske figurer og dermed bli mer bevisst egenskapene til for eksempel et kvadrat, en trekant og et rektangel.

3.4 Analysemetode

I denne studien er det naturlig å bruke en induktiv tilnærming i analysen av datamaterialet, som vil si at det er datamaterialet som danner grunnlaget for fenomenene som undersøkes. For å strukturere datamaterialet og gjøre det mer forståelig benyttet jeg meg av deskriptive analyser i form av koding og kategorisering, inspirert av *Grounded Theory* og *den konstante komparative analysemåten* som er beskrevet i Strauss og Corbin (1998). I tillegg gjorde jeg teoretiske analyser av enkelte utdrag av elevsamtaler og intervju for å forstå enkeltepisoder i datamaterialet i lys av eksisterende teori.

3.4.1 Grounded Theory og den konstante komparative analysemåten

Glaser og Strauss utviklet Grounded Theory som en metodisk tilnærming, med et ønske om å utvikle teori i møte med det innsamlede datamaterialet. Spesielt av interesse var hendelser, interaksjon og sosiale prosesser mellom mennesker. Innenfor denne tilnærmingen ble det senere utviklet en analysemåte kalt «the constant comparative method of analysis», på norsk oversatt til «den konstante komparative analysemåten». Denne analysemåten er en systematisk prosess bestående av tre faser, på norsk kalt åpen koding, aksial koding og selektiv koding (Strauss & Corbin, 1998). Postholm (2005) antyder at denne analysemåten også kan brukes innenfor andre kvalitative studier der koding og kategorisering blir viktig i analysearbeidet, og ikke bare innenfor den metodiske tilnærmingen Grounded Theory. Min

oppgave har ikke Grounded Theory som metodisk tilnærming, men som inspirasjon og støtte i analysen.

I *åpen koding* blir transkripsjonene som er gjort gjennomgått nøye og man leter etter fremtredende kategorier som informasjonen kan samles innunder. I prosessen blir datamaterialet delt inn i mindre deler ved at hver del får en kode. Disse kodene representerer ulike fenomen i datamaterialet. Etter hvert som antall koder øker, er forskeren nødt til å begynne å gruppere de kodene som kan dekkes av en og samme kategori. Målet med den åpne kodingen er å redusere datamaterialet til et mindre sett av kategorier som karakteriserer prosessene eller hendelsene som har inntruffet og som skal utforskes (Strauss & Corbin, 1998).

Under *den aksiale kodingen* skal kategoriene fra den åpne kodingen relateres til hverandre slik at forklaringene av fenomenene blir mer presise og fullstendige. For å spesifisere en kategori eller et fenomen må man lete etter trekk som var med på å utforme dem, det er disse trekkene som kalles for subkategorier. For å komme frem til disse kan man stille seg spørsmål om når, hvorfor og under hvilke forhold kategoriene dukket opp. På denne måten kan man finne relasjonen mellom kategoriene og deres respektive subkategorier (Strauss & Corbin, 1998).

Selektiv koding er fasen der forskeren prøver å finne kjernekategorien eller hovedkategorien som alle kategoriene fra de andre fasene kan relateres eller knyttes til. Denne kjernekategorien er det som representerer forskningens hovedtema (Strauss & Corbin, 1998). Creswell (1998) beskriver selektiv koding som den delen der det bygges en slags fortelling som forklarer hvordan kategoriene er knyttet sammen. Målet er å skape en helhet av alle kategoriene og et fåtalls ord som kan beskrive forskningen.

Postholm (2005) poengterer at den konstante komparative analysemåten ikke er statisk, men at fasene kan gli over i hverandre og at man i mange tilfeller allerede i første fase kan se sammenhenger mellom kategorier. Patton (1987) påpeker at en slik analysemåte krever at forskeren er kreativ og selvstendig, at enhver kvalitativ studie er unik og at analysen dermed også må få være unik.

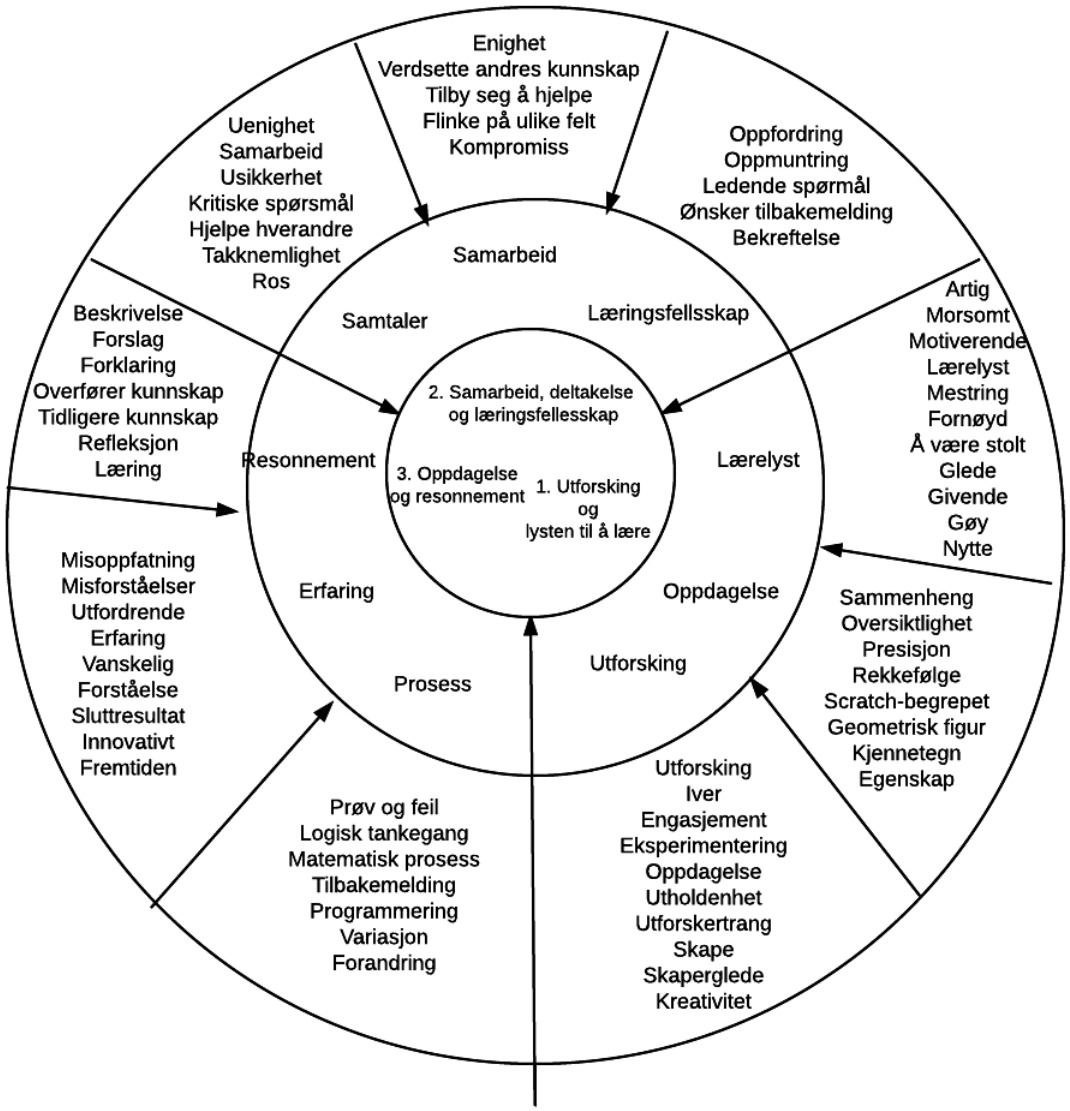
3.4.2 Koding og kategorisering

Jeg utviklet til dels mine egne navn på koder og kategorier ut ifra hva jeg fant i datamaterialet. Det er viktig å poengtere at til tross for at jeg har funnet egne koder og kategorier, har disse blitt påvirket av teori jeg har lest og hva jeg ønsket å finne i datamaterialet mitt. Kategoriene vil av den grunn ikke bare være selvlagde, de er også inspirert av kompetanser som trekkes frem i Fremtidens skole, ferdigheter som det fokuseres på i *21st Century Skills* og i *Computational Thinking*, og det som står sentralt i det sosiokulturelle perspektivet på læring.

Lydopptakene ble transkribert så nøyaktig som mulig for å få med alt som ble sagt, og deretter ordnet på en ryddig måte i regneark slik at jeg lett kunne finne frem i transkripsjonene. Jeg valgte først å lage en kolonne der jeg kommenterte hva som kom frem av hver elevs ytring og av observasjoner jeg hadde gjort. Disse kommentarene ble så gjennomgått og jeg begynte å sette koder på utsnitt og utdrag. I kodingen og kategoriseringen så jeg etter samarbeid, tegn på forståelse, ulike tilnærminger til matematikk, og affektive hendelser i form av motivasjon, engasjement, iver og glede. Det ble etter hvert som jeg kom meg igjennom datamaterialet flere og flere koder, på det meste 69 ulike koder. Betydningen av å kategorisere kodene for å få det mer oversiktlig kom tydelig frem. For å få koder som passet og beskrev datamaterialet mitt på en best mulig måte ble kodene gjennomgått og endret i flere omganger.

Etter første kategorisering av kodene sto jeg igjen med kategoriene: *samarbeid, samtaler, læringsfellesskap, lærelyst, oppdagelse, utforsking, prosess, erfaring* og *resonnement*. Etter å ha sett igjennom kodene flere ganger og flyttet de frem og tilbake mellom kategoriene jeg hadde satt opp, ettersom jeg fant fellestrekk, slo jeg sammen noen av kategoriene. Tilslutt satt jeg igjen med fire hovedkategorier: *utforsking og lysten til å lære, samarbeid, deltakelse og læringsfellesskap,*og *oppdagelse og resonnement.* I tillegg til disse tre hovedkategoriene vil det også presenteres datamateriale som belyser misoppfatninger jeg oppdaget, og hvordan elevene tok i bruk og ble mer bevisste på bruken av Scratch-begreper.

For å tydeliggjøre hvordan jeg har gått frem i kodingen og kategoriseringen av datamaterialet har jeg valgt å lage en figur som illustrerer dette. Den ytterste sirkelen inneholder kodene jeg hadde etter første gjennomgang av datamaterialet. I neste sirkel har kodene blitt samlet i kategorier som dekker flere koder, og i den innerste finner vi de tre hovedkategoriene.



Figur 8: Koding og kategorisering

3.5 Kvalitet på forskningen og etiske overveielser

I kvalitativ forskning er de tre kriteriene reliabilitet, validitet og generaliserbarhet indikatorer på forskningens pålitelighet og kvalitet (Tjora, 2012). For en kvalitativ forsker vil målet være å forsikre leseren om at fremstillingen som gis ikke er feilaktig, og å bevise at funnene som er gjort er troverdige og samsvarer med datamaterialet som er samlet inn (Nilssen, 2012). For å kunne belyse min problemstilling og komme med eventuelle svar er det viktig å begrunne hvordan disse kriteriene er blitt behandlet i forskningen.

3.5.1 Reliabilitet, validitet og generaliserbarhet

Reliabiliteten, eller påliteligheten, til forskningen vil kunne påvirkes av holdningene og engasjementet til forskeren om temaet det forskes på. Idealet er å være nøytral eller objektiv, men forskerens engasjement kan også ses på som en ressurs (Tjora, 2012). Dersom en forsker er ute i feltet en lengre periode har den mulighet til å identifisere de viktigste karakteristikaene til forskningsfeltet, noe som ytterligere kan bidra til å sikre kvaliteten (Nilssen, 2012). I min forskning vil mine interesser for programmering i matematikkundervisningen og mitt syn på dette kunne påvirke påliteligheten, og dette er derfor noe jeg må ha i bakhodet under analysen.

Validiteten til en forskning omhandler gyldigheten til forskningen, og om svarene vi finner faktisk er de svarene på spørsmålene vi stilte innledningsvis i forskningen. Gyldigheten til en forskning kan styrkes ved at den er forankret i relevant forskning (Tjora, 2012). En kvalitativ studie vil aldri kunne være helt objektiv, da man som forsker bringer med seg ulike former av forforståelser inn i prosessen. Tidligere erfaringer og kunnskap man har om emnet vil kunne påvirke hvordan man som forsker opptrer og hvilke valg man tar. Nilssen (2012) beskriver disse antakelsene man har som et kjennetegn ved kvalitative studier. En forsker påvirkes av sin teoretiske bakgrunn og egne antakelser, og tar disse med når han/hun møter sitt forskningsfelt. Gjennom denne bakgrunnen danner forskeren et filter som observasjonene oppleves igjennom (Postholm, 2005).

I en kvalitativ studie vil forskeren både påvirke og bli påvirket, da den er en del av det sosiale livet eller prosessen som det forskes på. Den må være bevisst sin refleksivitet og hvordan dens rolle påvirker forskningen (Leming, 2009). Leming (2009) nevner tre kategorier for

refleksivitet: forskerens tilstedeværelse som kan påvirke informantenes atferd, forholdet til forskningsdeltakerne som kan påvirke informasjonen forskeren får, og forforståelsen den møter feltet med som kan ha konsekvenser for hvilke og hvordan vi stiller spørsmål. Da jeg i min forskning har gjennomført deltakende observasjon, kan det tenkes at min rolle kan ha hatt innvirkning på elevenes atferd. Avhengig av hva elevene føler om min tilstedeværelse, vil de kunne holde igjen på informasjon og oppføre seg annerledes enn ellers. Dette til tross for at de vet hvorfor jeg er der og hva formålet er (Nilssen, 2012). Forskeren møter feltet med en egen referanseramme bestående av kunnskaper, forforståelse, erfaringer og opplevelser. Studien vil av den grunn være verdiladet og aldri helt objektiv (Postholm, 2005).

Kunnskap produsert i en kvalitativ studie vil være knyttet til det bestemte stedet og det bestemte tidspunktet da forskningen og undersøkelsene fant sted, kunnskapen er kontekstuell. Til tross for dette kan den allikevel være til nytte og være overførbar til andre (Postholm, 2005). Metodetriangulering kan føre til en naturalistisk generalisering ved at man med ulike typer materiale kan skape detaljerte og fyldige beskrivelser (Nilssen, 2012). Naturalistisk generalisering vil si at leseren kan kjenne seg igjen i situasjonen, og dermed kan erfaringer og funn fra studien være nyttig for egen situasjon (Postholm, 2005). Da mine undersøkelser skjer gruppevis i én enkelt klasse vil ikke denne kunne være representativ for alle andre klasser på samme trinn, men den kan ses på som overførbar til andre situasjoner ved at lærere med elever på samme trinn kan oppleve en naturalistisk generalisering.

3.5.2 Etiske overveielser

Det vil i enhver studie være etiske overveielser som må tas i betraktning. Forskeren balanserer på en smal egg der riktige valg skal tas, og hvor det hele tiden må tas stilling til om disse etiske overveielsene opprettholdes. I forkant av undersøkelsene mine ble metoden for datainnsamling godkjent av *Personvernombudet for forskning*, på bakgrunn av at datamaterialet kunne inneholde sensitive personopplysninger. Elevene ble opplyst om hvordan konfidensialiteten skulle bli ivaretatt, og det ble laget et samtykkeskjema til elevenes foresatte fordi elevene var under 18 år. Der ble forskningsprosjektet beskrevet, hva formålet var, hvordan innsamlingen skulle foregå og hva som ville skje med personopplysninger etter endt oppgave. Dette er det som (Kvale & Brinkmann, 2009) kaller for et informert samtykke. I skjemaet kunne foreldrene enten samtykke i at barna deres deltok, eller ikke samtykke. Stort

sett alle samtykket, og de få som ikke kom tilbake med skjemaet ble det ikke tatt lydopptak av. For å verne om elevenes personlige opplysninger, slik at de ikke skal kunne identifiseres, har jeg anonymisert studien ved å bruke andre karakteristika på elevene. Lydopptakene ble slettet etter at de var blitt transkribert, og transkripsjonene var kun tilgjengelige for meg som forsker.

4.0 Analyse

Målet med dette kapittelet er å gi innsikt i elevenes arbeid med programmering, og deres opplevelse av denne arbeidsmetoden for å kunne belyse problemstillingen min. Denne ble presentert tidligere i oppgaven, og er som følger: *«Hvilke tilnærminger til matematikk finnes det i arbeid med det blokkbaserte programmeringsspråket Scratch, og hvordan påvirkes elevenes affektive sider av arbeid med programmering?»*

Datamaterialet ble analysert gjennom koding og kategorisering med inspirasjon fra «den konstante komparative analysemåten» i Grounded Theory. Analysen førte til fire hovedkategorier: *utforsking og lysten til å lære, samarbeid, deltakelse og læringsfellesskap, og oppdagelse og resonnement.* I tillegg til disse fire hovedkategoriene vil det også presenteres datamateriale som belyser misoppfatninger som dukket opp underveis, og utdrag som viser elevenes bruk av Scratch-begreper for å være presise.

4.1 Utforsking og lysten til å lære

I datamaterialet mitt har jeg funnet tydelige tegn på at elevenes arbeid med programmering er preget av utforsking og en lyst til å lære. Dette ser vi gjennom at elevene utforsker og prøver mer enn hva som var forventet av dem og hva oppgavene krevde. Vi finner tegn til elevenes lyst til å lære gjennom elevenes positive holdning, engasjement og glede ved å skape. Kategorien er representert i alle elevgruppene, og eksemplene på samtaler er dermed ikke unike i den forstand at det finnes flere eksempler. Jeg mener likevel at de er unike på den måten at de er spesielt tydelige eksempler på elevenes utforsking og lyst til å lære.

Samtidig som denne kategorien er formet av empiri, er den også blitt påvirket av teori. Programmeringsaktiviteter i matematisk problemløsning krever at elevene har en evne til å utforske og skape, som trekkes frem i «Fremtidens skole» som viktige kompetanser elever vil trenge i fremtiden (NOU 2015:8). I læreplanen i matematikk fellesfag i Kunnskapsløftet står det: «*Opplæringa vekslar mellom utforskande, leikande, kreative og problemløysande aktivitetar og ferdigheitstrening»* (Utdanningsdirektoratet, 2013). Utforsking og lysten til å

lære som kategori er dermed relevant både for dagens læreplan og for fremtidens kompetanser.

4.1.1 Det å utforske

Under er et utdrag fra en situasjon der elevgruppe 2 arbeider med å tegne geometriske figurer i Scratch. De to elevene har sammen klart å skrive et program som tegner et kvadrat på skjermen, men ser seg ikke ferdige av den grunn. Utdraget viser at de fortsetter å utforske selv etter å ha erkjent overfor hverandre, med ytringer i linje 210 og 211, at de har fått til oppgaven. De ser ut til å ha et behov for å teste, utforske, forstå og å lære mer.

210. 2B :	Ja, og vi fikk det til
211. 2A:	Ja
212. 2B:	Vi prøver en gang til, jeg så det jo nesten ikke
213. 2A:	Prøv å ta enda lengre med “steps” da
214. 2B:	Ok
215. 2A:	200 kanskje
216. 2B:	Nei, 300
217. 2A:	Bare flytt den lengre opp (peker på Scratch-figuren) også trykk på flagget
218. 2B:	Kult!
219. 2A:	Ja vi kunne lage den større
220. 2B:	Vi kan lage mindre også (endrer på lengden, antall steps, Scratch-figuren skal gå)
221. 2A:	Også tar vi 250. Da blir det jo sånn, også tar vi 200
222. 2B:	Ja
223. 2A:	Også 150
224. 2B:	100

225. 2A: 100

226. 2B: 50! 50! 50!

227. 2A: Ja!

Videre kan vi se av utdraget at elevene prøver seg frem, de vil ikke fortsette på neste oppgave med en gang. Det kommer frem at de ønsker å se hva som skjer dersom de endrer noen av variablene de har brukt i koden. I dette tilfellet gjelder dette antall steg og antall ganger noe skal skje. De vil se om endringer kan gjøre kvadratet de har laget større og mindre. Dette ser vi i ytring 219 med: *«Ja vi kunne lage den større»* og 220 med: *«Vi kan lage mindre også»*. Mine observasjoner av elevene ga meg inntrykk av to ivrige og engasjerte elever som var dypt konsentrerte med oppgaven. Utforskingen i seg selv ser ut til å være motiverende nok til å skape en positiv holdning til arbeidet, noe som kommer tydelig frem når elev 4A og 4B forteller om sine tanker om dagens programmeringsøkt.

56: 4A: Vi finner ut hvordan de skal være også gjør datamaskinen det

57: 4B: Det er skikkelig artig å finne ut av ting

Ytringen til 4A får frem at elevene har en opplevelse av at det er de som finner ut av ting, for at det så er datamaskinen sin oppgave å fullføre. De to ytringene får frem at elevene synes det er *artig* å finne ut av ting. De oppdager og finner fremgangsmåten, og får deretter datamaskinen til å gjøre eller utforske det de har tenkt og funnet ut. I en samtale i elevgruppe 5 kan det se ut til at den ene eleven har utforsket «nok», og påpeker dette i ytring 3 med å si: «*Nå begynner jeg å bli lei…».* Samtidig ser det ut til at den andre eleven ikke er interessert i å gi seg ennå, og fortsetter til oppgaven er løst og ytrer: *«Klarte det, jeg klarte det»*. Det kommer av samtalen frem både en positiv og en negativ holdning til arbeidet med oppgaven. 5B opplever mestring gjennom å få til oppgaven, mens 5A nærmest ikke tror at 5B har fått det til. Dette kommer tydelig frem med 5A sine ytringer *«Tuller du?»* og *«Gjorde du?»*.

3: 5A:	Nå begynner jeg å bli lei…
4: 5B:	Jeg prøver jeg
5:	(Flytter blokker frem og tilbake)
6: 5B:	Endelig!
7: 5A:	Tuller du?
8:	(trykker kjør og det blir riktig)
9: 5B:	Klarte det, jeg klarte det!
10: 5A:	Gjorde du?
11: 5B:	Ja!

I dette delkapittelet har vi sett eksempler på datamateriale som tyder på at programmeringsøktene kan være en arena for utforsking, noe som er en viktig del av prosessorientert matematikk der fokuset ligger på selve prosessen frem mot et produkt.

4.1.2 Lysten til å lære

Nesten utelukkende alle elevene i studien så det som gøy og motiverende å arbeide med programmering i matematikktimene. Ytringene under er hentet fra et intervju med elevgruppe 1 der de sammenligner arbeid på datamaskin med tavleundervisning, og hvor det kommer frem at arbeid på datamaskin foretrekkes. Bruk av datamaskin i undervisningen kan for noen elever i seg selv være motiverende, men jeg har i mitt datamateriale funnet eksempler på at den matematiske prosessen i seg selv er motiverende.

21: Elev 1B:	En artig måte
22: Elev 1A:	Og det er mye mer motiverende å sitte og gjøre det der (peker på datamaskinen) siden det er gøy, og her (peker på tavla) er det ikke like gøy

Bruk av datamaskin kan i seg selv være en motivasjonsfaktor til elevenes lærelyst. Likevel kom det frem at flere elever ble faglig stimulert av utforskningen som programmering i det

nye koordinatsystemet i Scratch medfører. I dialogen til elevgruppe 5 påpekes det at programmering er vanskeligere, men at det samtidig er morsommere. De trekker frem både fordeler og ulemper med å lage et kvadrat når de arbeider med programmering. Dette kommer frem i ytringen til 5A: *«Jeg synes det er enklere å tegne i boka»*, og ytringen til 5B *«Da trenger vi bare blyant, ark og linjal. Men det er morsommere på pc da»*.

14: Hedda:	Hvordan er det å for eksempel lage et kvadrat i Frost da, enn å tegne et kvadrat i boka?
15: Elev 5A:	Det er på en måte likt, men på data blir det liksom litt annerledes
16: Hedda:	Men er det enklere eller vanskeligere da?
17: Elev 5A:	Jeg synes det er enklere å tegne i boka
18: Elev 5B:	Da trenger vi bare blyant, ark og linjal. Men det er morsommere på pc da
19: Hedda:	Så selv om det er vanskeligere, så er det morsommere?
20: Elev 5B:	Ja

Av samtalen kommer det frem at elevene liker utfordringen med å tegne de geometriske figurene i Scratch. Selv om de påpeker at det en vanskeligere, mener de fortsatt at det er morsommere enn å arbeide i matematikkboka. Selv om programmeringen kanskje krever mer, er dette en utfordring elevene tar.

Utfordringene med programmeringen ser ut til å ha positiv innvirkning på elevenes lærelyst, og flere fant arbeidet både givende og lærerikt. I dialogen mellom meg selv og en elev på vei ut av klasserommet ser vi et eksempel på en elev som ser ut til å se nytten i de erfaringene han/hun har gjort seg og det som har blitt lært i klasserommet.

156. Hedda:	Da håper jeg dere har lært litt om de geometriske figurene og deres egenskaper
157. Elev 1A:	Vi har ikke lært litt, vi har lært mye, Hedda
158. Hedda:	Det var hyggelig å høre
159. Elev 1A:	Vi hadde ikke lært alt dette hvis vi skulle gjort det i boka. Da hadde vi ikke visst hvor mange grader det er i en trekant, kvadrat og de andre figurene. I boka tegner vi bare, men PCen vet jo ikke hvordan den skal tegne figurene, den trenger hjelp av oss og da må vi vite hvordan vi skal gjøre det for å hjelpe den.

Eleven har oppdaget at det ikke bare dukker opp en geometrisk figur på datamaskinen av seg selv, det må litt innsats til. I ytring 159 poengteres det at man må hjelpe datamaskinen med å lage figurene, man må altså skape noe. Både opplevelsen av å aktivt skape noe og følelsen av å «hjelpe til» kan altså være kilder til motivasjon i arbeid med programmering.

I et refleksjonsnotat i etterkant av en økt kommer opplevelsen av å se elevene utforske og skape gjennom arbeid med programmering, sett fra min eller forskerens side, frem gjennom at jeg har skrevet:

> […] Når det glitrer i øynene til elever fordi de har laget en snøkrystall alene kan man ikke annet enn å komme med et stort smil. Det går ikke alltid på første forsøk, men denne klassen har skjønt at det er helt greit. Når de prøver for femte gang å endre på koden, vil jeg si at dette viser en viktig egenskap-utholdenhet. De gir ikke opp, men prøver videre. Når de etter flere forsøk kommer i mål, kommer det gledesrop ut i klasserommet! Akkurat som om det ble scoret et mål i en fotballkamp. *" Vi klarte det, kan du komme og se?"* Og selvfølgelig vil jeg se. Det er viktig å vise at de fem blokkene de har satt riktig sammen er noe de kan være stolte av. […] (Hedda 30.10.15).

Vi har sett i eksemplene som er trukket frem at programmeringsaktiviteter kan ha en positiv innvirkning på elevenes holdninger, engasjement og glede ved å skape. Elevene liker at programmeringsaktivitetene er litt utfordrende og dermed krever litt innsats. Programmeringen åpner opp for at elevene arbeider med matematisk problemløsning gjennom «Productive Struggle», oppgavene er innen rekkevidde av det de kan klare og når elevene får det til kan de få følelsen av suksess og mestring. Dataene viser at den positive holdningen og engasjementet til elevene ser ut til å påvirke motivasjonen til å arbeide. De som på arbeidet som meningsfylt, og ikke som noe «bortkastet».

4.2 Samarbeid, deltakelse og læringsfellesskap

I første økt spurte flere elever om de ikke bare kunne ha en datamaskin hver. Spørsmålet kan reflektere de matematiske arbeidsformene de er vant med, men etter denne ene gangen kom ikke dette spørsmålet opp igjen. Det kan tenkes at gleden over å kunne dele erfaringer og vise hverandre hva de hadde laget overgikk ønsket om det å ha en egen datamaskin. Samarbeid har hatt en stor rolle i prosjektet mitt da alt av arbeid skulle gjøres i par eller grupper, og der også én økt foregikk i klassen som helhet. Denne kategorien har på denne måten både bakgrunn i empiri, men også i teori i form av det sosiokulturelle perspektivet på læring. Samarbeid i form av samhandling og deltakelse trekkes frem som viktige ferdigheter og kompetanser i både *Fremtidens skole, 21st Century Skills* og *Computational Thinking*. Kategorien finner vi igjen i datamaterialet til alle elevgruppene. Dette kommer sannsynligvis på grunnlag av at det på forhånd var bestemt at elevene skulle arbeide i par eller grupper på tre. Elevgruppene som ble intervjuet fikk også spørsmål om betydningen av samarbeid i programmering, både fordeler og ulemper. Eksemplene som trekkes frem er derfor ikke unike, men de er spesielt tydelige.

Betydningen av samarbeid i programmering var det flere av elevene som hadde reflektert over da jeg stilte spørsmål ved akkurat dette. Intervjuet med elevgruppe 2 er et eksempel som viser elevenes oppfattelse av hva som er bra med samarbeid. Dette vises tydelig i ytring 51 og 52, der de kommenterer at både det å få til ulike ting og det å kunne fordele oppgaver, er noe som er bra med samarbeid.

48. H:	Hvordan er det å samarbeide når man jobber med sånne her oppgaver? Liker dere å samarbeide om sånne her oppgaver?
49. 2A:	Det er gøy å samarbeide ja
50. H:	Hva er det som er bra når man samarbeider da?
51. 2B:	For hvis jeg ikke får til noe, så får kanskje 2A til det. Og hvis 2A ikke får det til, så får kanskje jeg det til
52. 2A:	Det var sånn vi holdt på i stad, for jeg klarte ikke å få ting inni den boksen, men det klarte elev 2B. Da kunne 2B gjøre det. Så vi var faktisk ganske flinke til å fordele oppgaver

Forståelsen for betydningen av samarbeid i arbeidet, og at vi kan være flinke på ulike felt kan underbygges med en kommentar fra elev 2A ved en tidligere økt: *«Nei det trenger vi*

ikke….for se! eller nei, du tenkte sånn ja. Det er bra det». Eleven viser i ytringen forståelse for at medeleven tenkte på en annen måte og påpeker at den har rett den også. De har funnet to veier til samme mål, og eleven viser med ytringen at den verdsetter begge like mye.

Verdsettelse i form av å se verdien av hverandres kunnskap og ferdigheter ble påpekt av flere elever. De hadde reflektert over både sin egen og medelevers innvirkning på samarbeidet. Intervjuet med elevgruppe 4 viser elevenes tanker og meninger om betydningen av samarbeid i programmering. Elevenes oppfattelse av hva som er bra med samarbeid vises tydelig i ytring 32-36, der de poengterer det å dele og gi bort ideer, hjelpe hverandre og at det kan bli enklere. Videre i samtalen kan vi se at elevene også trekker frem ulemper med programmering som arbeidsmåte. Ytringen til elev 4C får frem potensielle utfordringer med samarbeid i programmering: *«…, men noen ganger kan det være greit å være alene siden at man ikke alltid har lyst til å gjøre det samme og har like ideer».*

32. H:	Hva betyr samarbeid i programmering?
33. 4B:	Det blir mye lettere
34. 4C:	For du kan gi bort ideer til hverandre
35. 4A:	Dele ideer
36. 4C:	Hjelpe hverandre
[…]	
40. 4C:	Noen ganger kan det være bra og sitte to og to, men noen ganger kan det være greit å være alene siden at man ikke alltid har lyst til å gjøre det samme og har like idéer
41. 4B:	For eksempel så ville 4A gjøre straffespark, imens jeg ville gjøre spøkelsesjakten

Stort sett hele klassen var enige i at samarbeid var til god hjelp under programmeringen, og at dette var positivt for løsningen av oppgavene. Likevel var det noen elever som også hadde reflektert over at det både er fordeler og ulemper med programmering, og et eksempel på dette finner vi i et utdrag av et intervju med elevgruppe 6 som er med på å underbygge elevenes mening om betydningen av samarbeid. Det er 6A som står for flesteparten av svarene, og 6B samtykker med blant annet å ytre: *«Sånne ting.»* der 6A har forklart at det oppstår en del utfordringer når man har hver sin pc. Denne elevgruppen ser også, som elevgruppe 4, både fordeler og ulemper med samarbeid i programmering. *«Da blir det jo ett arbeid da, og da kan det bli litt vanskelig å bytte på. Men det er veldig lett å samarbeide da»* viser en elev som har erfart både positive og mindre positive sider ved programmering, og ser

ikke ut til å være enten for eller imot samarbeid. Det å ha ulike ideer problematiseres med *«Ja, for da får alle gjort sine ideer. For hvis vi har en pc og begge to skal gi sine ideer så blir det litt krangling»*.

64. H:	Men hvordan er det å samarbeide sånn da, enn om man samarbeider om en oppgave i boka?
65. 6A:	Det kommer an på om vi har en eller to pcer da
66. H:	Hvordan er det å ha en pc og være to da?
67. 6A:	Da blir det jo ett arbeid da, og da kan det jo bli litt vanskelig å bytte på. Men det er veldig lett å samarbeide da
68. H:	Mhm
69. 6A:	Men når vi lagde labyrintspillet da, så lagde vi på hver vår pc, og da ble det litt sånn «du må vente på meg da», «Jeg trenger hjelp, jeg skjønte det ikke», «Å, jeg sletta alt, jeg må se hva du gjorde»
70. 6B:	Sånne ting. Jeg syns hvertfall det var artig når vi hadde hver vår pc
71. 6A:	Ja, for da får alle gjort sine ideer. For hvis vi har en pc og begge to skal gi sine ideer så blir det litt krangling
72. H:	Mhm
73. 6A:	«Nei, jeg skal ha rosa», «Nei, jeg vil ha blå» og sånne ting

I denne kategorien har jeg vist flere eksempler der det tydelig kommer frem at elevene er positive til å samarbeide, men at de også klarer å se både fordeler og ulemper. I arbeid med matematisk problemløsning skaper programmering en arena for samarbeid, samtale og deltakelse. Dette er et viktig aspekt i det sosiokulturelle perspektivet på læring, og som også trekkes frem som en viktig kompetanse for å lykkes i dagens og fremtiden samfunn.

4.3 Oppdagelse og resonnement

Datamaterialet mitt viser at Scratch egner seg godt til å gi muligheter for å arbeide med matematiske resonnement og oppdagelse. Det oppstod flere situasjoner der elevene underveis i arbeidet oppdaget at noe var feil eller at det var noe som kunne bli bedre i koden de hadde satt sammen. Dette førte til at elevene var nødt til å resonnere seg imellom for å finne ut hvordan de skulle løse oppgaven. Kategorien kommer frem av datamaterialet, men er også

påvirket av teori om viktigheten av dypere forståelse av matematikk, og som viktigheten av dybdelæring som trekkes frem i Fremtidens skole. Denne kategorien finnes i alle elevgruppene, men i ulik grad. Eksemplene ses derfor på som unike fordi de på en spesielt tydelig måte får frem de matematiske resonnementene og oppdagelsene.

4.3.1 Prøv og feil som oppdagelse

I arbeidet med matematikk i programmering tok elevene i bruk matematiske resonnement og oppdagelse. Et eksempel på dette finner vi i et utdrag fra et intervju med elevgruppe 4 som forklarer hvordan de har arbeidet med å lage en trekant. Det kan av utdraget se ut til at elevene har prøvd seg frem i sitt forsøk på å løse oppgaven som her var å tegne en trekant i Scratch. Dette kommer frem av blant annet ytring 232: *«Man kan prøve seg litt fram, enten blir det en trekant eller så blir det ikke en trekant»*. De forklarer at det enten blir rett eller galt, med henvisning til trekanten, og videre ser vi at de påpeker at de først fikk en sekskant da de egentlig skulle lage en trekant. Med ytringen: *«Og så bygde vi oss oppover og oppover og oppover helt til det ble en hel trekant»* kan det se ut til at elevene har sett en sammenheng mellom kanter og antall grader på veien mot å tegne en trekant. Av utdraget ser vi at elevene etter noen forsøk får frem en trekant og at den har *«sånn 60 grader»*. Jeg prøver med min ytring 243 å skape mulighet for resonnement hos elevene gjennom å rette fokus mot sekskanten og trekanten, for å se om elevene kan se en sammenheng.

230. 4A:	Og på trekanten tenkte jeg på 120 grader, og på gradskiva, og hvordan den er, og så fant jeg ut det
231. H:	Hvordan visste du at det skulle være 120 da?
232. 4A:	Man kan prøve seg litt fram, enten blir det en trekant eller så blir det ikke en trekant
233. 4B:	Ja, og da ser man om det er rett eller feil
234. 4D:	Først fikk vi en sekskant
235. H:	En sekskant ja? Hvordan fikk dere til det?
236. 4B:	Vi lagde det vi trodde det var
237. H:	Også ble det en sekskant?
238. 4B:	Eh, ja…

239. 4A: Og så bygde vi oss oppover og oppover og oppover helt til det ble en hel trekant

240. 4D: Sånn 60 grader

241. H: Hvorfor blir det 60 grader da?

242. 4B: Halvparten av 120

243. H: Og 120 måtte dere snu når dere lagde en?

244. 4A: Trekant, så da ble 60 grader i en sekskant

Det kan av ytringene se ut til at elevene underveis i kodingen av trekanten har oppdaget noen sammenhenger, og at denne «feilen» har ført til mer læring enn nederlag hos elevene. Denne prosessen med prøving og feiling fører til at elevene ser sammenhengen mellom en sekskant og en trekant, og at de må vende 120 og ikke 60. Denne «misforståelsen» kan snus om og bli en nyttig erfaring, en prosess som fører til kunnskap man kanskje ikke ville ha kommet inn på ellers. Denne måten å arbeide på krever en viss utholdenhet og innsats over en lengre tid, da de har gjort flere forsøk med antall grader Scratch-figuren skal vende. De har på veien funnet ut nye ting gjennom å oppdage feil. Dette resonnementet rundt sekskanten og trekanten kan knyttes til indre og ytre vinkler, og på den måten kan dette trekkes med inn i videre undervisning.

Matematiske resonnement sammen med oppdagelser underveis i arbeidet, ble viktig i elevenes arbeid med matematiske problemer i Scratch. For å underbygge dette vises et eksempel fra økten med felles programmering i klassen som helhet og spiller videre på kodingen av en trekant. Med utdraget ønsker jeg å vise at elevene sammen bruker matematiske resonnement når de sammen skal programmere en trekant. Jeg har et ønske om at elevene skal finne ut hvor stor hver vinkel er når alle vinklene er like store. Vi ser i ytringen: *«90….nei 180 delt på tre…60»* at elevene klarer å resonnere seg frem til hvor mange grader hver vinkel må være når de får oppgitt at det skal være 180 grader til sammen. Med ytringen: *«Det har liksom med hvordan den går…det blir liksom….»* ser det ut til at eleven har sett en sammenheng med antall grader Scratch-figuren vender og antall grader vinkelen blir. Dette kan være en begynnelse på en forståelse for indre og ytre vinkler i geometriske figurer.

200. H: Alle vinklene i en trekant...den pluss den pluss den. Har dere lært hvor mange grader det skal bli tilsammen?

201. Klassen: 360 grader kanskje?

202. H: Det er i en…

203. Klassen: Sirkel!

204. H: Og i en trekant skal det være. Kanskje dere ikke har lært det? 180 grader. Så hvis de er like store hvor mange grader må hver være da hvis det skal bli 180 til sammen?

205. Klassen: 90…...nei 180 delt på tre…..60

206. H: 60 ser dere, men hvorfor snur vi 120 da? Hvorfor snur vi 120 når vi sier at den er 60 her?

207. 6A: Det har liksom noe med hvordan den går...det blir liksom….

208. H: Du tenker helt riktig.

209. 6A: Ja…

210. H: Så hvis dere ser her (peker på en av vinklene i trekanten, den skal være 60, men som 6A sier så må han gå 120 for å snu seg han derfor vender vi 120. Men vinkelen er 60. (…)

I delkapittelet har vi sett eksempler som tydelig viser elevenes bruk av matematiske resonnement og oppdagelse i programmeringen av geometriske figurer. Dette er en form for metakognisjon som i Fremtidens skole trekkes frem som kompetansen «Å lære seg å lære». I programmering er «debugging» en viktig del av prosessen med å forstå programmet, og Papert (1980) påpekte at det var viktigere å fokusere på feilen enn å prøve å glemme den. Denne egenskapen er viktig i blant annet prosessorientert matematikk, hvor det er prosessen og ikke produktet som er det viktigste.

4.3.2 Programmering som arbeidsmetode

I intervju med elevene ble det av flere elever nevnt at det var annerledes å arbeide med geometriske figurer i Scratch enn i matematikkboka. Et eksempel på dette finner vi i et intervju med elevgruppe 1 som viser hvilke meninger de har om forskjellen på å arbeide med geometriske figurer i matematikkboka, og det å arbeide med det i Scratch. Det ser ut til at de har en opplevelse av at det kreves mer tenking når de arbeider i Scratch, og en elev poengterer dette med *«Tenke mye, og når man vet det er det veldig enkelt»*. Av utdraget kan man også lese at eleven ser på en trekant som noe man bare gjør på 1, 2, 3, og så er man ferdig. Det ser ut til å være noe man gjør helt uten videre.

140. 1A: I boka kan man gjøre det 1, 2, 3

141. 1C: På Scratch da må man…

142. 1A: Tenke

143. 1C: Tenke mye, og når man vet det er det veldig enkelt

I et intervju med en annen elevgruppe, der jeg også ønsker å få frem elevenes meninger om forskjeller i arbeidsmetode ser vi at det å tenke nevnes også her. *«Man må tenke istedenfor å bare tegne»*, det ser av utdraget ut til at denne eleven ser på det å lage en geometrisk figur som en handling uten en tanke, noe automatisk som man bare gjør. Med ytringen *«Mye mer, det her kan jeg liksom gjøre i blinde sånn her. Men her må jeg liksom vite hvordan jeg gjør det da»* ser det ut til at eleven har en opplevelse av at det krever med av dem når de skal kode enn når de gjør det i boka. Der gjør de det i blinde kommer det frem fra ytringen.

106. H: Men når man koder et kvadrat da, så er det kanskje bittelitt annerledes enn når man tegner i boka?

107. 1A : Ja, altså man må tenke

108. H: Ja

109. 1A: Man må tenke istendenfor å bare tegne

110. H: Må man tenke mer når man koder enn når man tegner?

111. 1A: Mye mer, det her kan jeg liksom gjøre i blinde sånn her. Men her må jeg liksom vite hvordan jeg gjør det da.

Elevenes opplevelse av at programmering er en annerledes arbeidsmetode kommer fram i et intervju med elevgruppe 3. Ved å ytre «Artig!» samtidig gir 3A og 3B inntrykk av at dette er noe de har likt å holde på med. Videre i samtalen begrunner de allikevel mer konkret hvorfor de er positive til denne typen arbeid, blant annet med: *«Mye bedre. For du lærer mer om en*

datamaskin, også forstår du mer hva du kan gjøre også lærer du hvordan man kan lage det på pc».

10. H: Men hvordan er det å arbeide med for eksempel geometri da? Med trekanter, firkanter… Hvordan er det å arbeide med det i Scratch istedenfor i boka der man konstruerer med passer og linjal?

11. 3A/3B: Artig!

12. 3A: Mye bedre. For du lærer mer om en datamaskin, også forstår du mer hva du kan gjøre også lærer du hvordan man kan lage det på pc

13. H: Ja, men er det enklere eller vanskeligere da?

14. 3B: Det er kanskje litt vanskeligere når du skal programmere sånne figurer til å gjøre det (henviser til Scratch-figuren)

15. H: Ja

16. 3A: Hvis vi for eksempel går inn på Paint da som et eksempel nå, så er det forskjellige verktøy sånn at vi kan lage linjer, og da blir det veldig enkelt å lage rette firkanter og sånt

I dette delkapittelet har jeg med eksempler vist hva elevene tenker om bruken av programmering som arbeidsmetode i matematikkundervisningen. Det kommer av eksemplene frem at elevene ser på det som litt vanskeligere og at de er nødt til å tenke mer over hva de gjør. Kategorien kan knyttes til prosessorientert matematikk, da elevene ser nytten av at en geometrisk figur er mer enn bare 1,2,3 eller noe man gjør i blinde.

4.4 Læring gjennom å rette opp i misoppfatninger hos elevene

I datamaterialet mitt fant jeg eksempler på at det i noen elevgrupper hadde skjedd misoppfatninger i arbeidet med programmering av geometriske figurer. Denne kategorien kom frem utelukkende fra datamaterialet, da dette var misoppfatninger jeg ikke observerte før jeg transkriberte lydopptakene. De to utdragene som er presentert i denne kategorien er unike i den forstand at det var her det tydelig kom frem misoppfatninger. Denne kategorien finner vi

ikke representert i alle elevgruppene, men det kan tenkes at det finnes flere, men at de ikke kommer like tydelig frem.

Underveis i arbeidet dukket det opp enkelte situasjoner der det hadde oppstått misoppfatninger hos elevene. Et eksempel på en situasjon der det har oppstått en misoppfatning i arbeidet med geometriske figurer finner vi i elevgruppe 4. Elevene nevner i ytring 61-64 egenskaper ved en sirkel man må ta hensyn til når man skal sette sammen blokker for å lage en kode for en sirkel. At en elev nevner *«Og hvor fort den skal gå.»* refererer sannsynligvis til opplevelsen av at figurene tegnes i «ulik hastighet» ettersom antall steg og antall ganger. Ved en trekant trenger man kun å gjenta lengde og vinkel tre ganger, i et kvadrat må dette skje fortere. Farten avhenger av lengde og repetisjoner. I utdraget ser vi at jeg som lærer prøver å få elevene til å fortelle hva de vet om sirkler, og baktanken var å få frem at det er 360 grader i en sirkel. I stedet ser vi at elev B ytrer: *«Man måtte gjenta 37 ganger»*, og videre at *«Ja, og det ble riktig»*. Elevene har brukt et tall som ikke er det første man assosierer med egenskapene til en sirkel.

60. H:	Hva måtte dere tenke på da dere skulle lage en sirkel?
61. 4B:	Hvor mange steg man måtte ta
62. 4D:	Og hvor fort den skulle gå
63. 4A:	Og hvor mange grader
64. 4B:	Og hvor mange ganger det skulle skje
65. H:	Hva vet dere om en sirkel da?
66. 4B:	Man måtte gjenta 37 ganger
67. H:	Så dere brukte 37?
68. 4B:	Ja, og det ble riktig

Denne elevgruppen forstår ikke selv at det har skjedd en misoppfatning, da de faktisk har fått rett svar i form av at de har tegnet en sirkel. Dette grunner i at elevene har latt «Scratch-figuren» bevege seg mer enn en runde, men dette kommer ikke frem av programmering foruten at Scratch ikke starter og stopper på samme sted. Dette kan ikke forventes av alle

elever, da dette var noe vi ikke hadde på forhånd pratet om. Sirkelen var helt klart den vanskeligste geometriske figuren for de aller fleste.

Det oppstod misoppfatninger også i fellesøkten, hvor vi da sammen fikk mulighet til å rette opp i dette. Et eksempel på en misoppfatning som kom frem i klassediskusjonen i fellesøkten på SMARTboard omhandler programmering av en trekant.

167. H:	Da kan vi begynne med trekant, hva vet vi om en trekant, 2A?
168. 2A:	Den har ikke 90 grader, den har for eksempel 45 grader
169. H:	For eksempel 45….elev 6A
170. 6A:	Det finnes en trekant som har 90 grader, men ikke alle 90 grader (tenker på alle vinklene i en trekant). Men én liksom.
171. H:	Det finnes trekanter med én vinkel med 90 grader ja, det er riktig
172. 6A:	Ja
173. H:	Men hvis den skal ha like store vinkler da, hvor mange grader må hver være da?

Av utdraget ser vi at det gjennom samtale avdekkes det som kan være en misoppfatning hos den ene eleven, eleven tror ikke at det finnes trekanter med 90 grader. Det kan hende at dette har en sammenheng med at det kun har vært fokus på regulære mangekanter, og at 90 grader av den grunn har blitt forbeholdt kvadrater og rektangler for noen av elevene. Av utdraget ser vi at en annen elev med ytringen: *«Det finnes en trekant som har 90 grader, men ikke alle 90 grader, men en liksom»* poengterer dette overfor de andre elevene med å påpeke at det finnes, men at ikke alle vinklene kan være 90 grader.

Delkapittelet viser tydelig to misoppfatninger som har skjedd i klasserommet, hvor den ene ikke rettes opp da det er et lydopptak av en elevgruppe som arbeider imens det andre er en fellesøkt der jeg sammen med elevene kunne rette opp misoppfatningen. Det kommer tydelig frem av eksemplene at dette er fallgruver som kan skje når man tar i bruk programmering i matematikkundervisningen. Samtidig kan det som vi ser i eksempelet fra fellesøkten brukes til diskusjon dersom man legger merke til det. Denne måten å møte misoppfattelser på er viktig

når elevene skal tilegne seg en dypere forståelse av matematikk, og der det å nå målet er et resultat av å løse problemer gjennom å reflektere over dem og kommunisere dem. Tilegnelsen av en dypere forståelse av matematikk er det Ludvigsenutvalget kaller for dybdelæring i fagene (NOU 2015:8).

4.5 Språkets rolle når klassen programmerer i fellesskap

Datamaterialet mitt viser at elevene i løpet av øktene med programmering ble mer bevisst på bruken av Scratch-begreper når de skulle forklare hverandre hvordan de skulle løse oppgavene. Det ser ut til at de så en verdi i å kunne være presise når de samtaler og dermed være sikre på at medelevene forstår hva de mener. Denne kategorien kom frem av datamaterialet, men jeg hadde selv en anelse om at dette kunne bli et resultat av læringen av programmeringsspråket. I fellesøkten der klassen programmerte geometriske figurer på SMARTboard finner vi tydelige eksempler på at elevene har tatt i bruk Scratch-begreper når de skal forklare og hjelpe hverandre med oppgavene. Elevene går fra å bruke «skritt» med ytringen: *«Gå hundre skritt»* til å bruke steg som er en begrep som brukes i programmeringsspråket med ytringen: *«Steg!»*. Det kan se ut til at elevene vil være mer presise og på den måten sørge for at det blir brukt riktige blokker når koden lages.

26. Klasse:	Gå hundre skritt
27. Hedda:	Ja (henvender seg til elev i klassen)
28. Elev:	Gå ti skritt
29. Hedda:	Ok, gå ti skritt, da tar vi det?
30. Klassen:	Steg!

Videre i samme fellesøkt ser vi at elevene tar i bruk Scratch-begrepene *bevegelse*, *gjentablokk* og *vend* når de diskuterer hvordan de skal løse oppgaven hvor de skal programmere en sirkel.

228. Hedda:	Hva vet vi om en sirkel da?
229. Klassen:	Den er 360 grader
230. Hedda:	360 grader ja
231. Klassen:	Bevegelse…

232. Hedda: Du sier gjentablokk

233. Elev: Ja

234. Hedda: Skal vi prøve det?

235. Klassen: Nei……jo

236. Hedda: En gjentablokk ja. Hva skal vi sette inni da?

237. Klassen: 1, det er jo en runde

238. Hedda: Ja en runde. Hva mer?

239. Elev: Vi trenger ikke noen gjentablokk da

240. Elev: Det er jo ikke noe vits i

241. Klassen: 360 grader

242. Hedda: Vende 360?

243. Klassen: Ja

244. Elev: Men da trenger vi jo ikke gjentablokken, hvis den skal gjenta én gang

245. Klassen: Jo vi trenger den

246. Elev: Ikke hvis det er én gang

247. Elev: Det er ikke noe poeng og ha 1 i gjentablokken

248. Hedda: Nå må dere hjelpe hverandre

249. Klassen: En 360-blokk og en bevegelsesblokk….

250. Elev: Også vend

Jeg observerte at disse uttrykkene på en måte kom ubevisst ved at elevene ikke anstrengte seg med å finne de, og på en annen måte at de var bevisste for å være sikre på at de snakket om det samme. I denne kategorien har jeg presentert eksempler som tydelig viser elevenes bruk av Scratch-begreper når de samtaler med hverandre. Gjentablokken er et viktig begrep i programmering, men det er også en viktig prosess i matematikken når man leter etter sammenhenger og mønstre.

4.6 Oppsummering av analysen og funn

I min analyse av elevsamtaler, elevintervju og observasjoner har jeg gjort flere funn som jeg ønsker å drøfte i neste kapittel. Kategoriene jeg har kommet frem til er i seg selv funn, men jeg har også gjort mer spesifikke funn innenfor kategoriene. Det er tydelige tegn til at elevene utforsker og har en glede i å skape når de programmerer. I utforskingen til elevene ser vi tegn til en utholdenhet i arbeid med oppgavene, et viktig funn som jeg skal komme tilbake til. Det kan også antas at enkelte elever opplever en mestringsfølelse og at denne formen for arbeid er motiverende. Samarbeid og deltakelse går som nevnt igjen i alle gruppene, og elevenes holdning til samarbeid skal senere drøftet og ses i lys av teori. Ut i fra utdragene som representeres ser oppdagelsene og refleksjonene elevene gjør seg ut til å bidra til forståelse og ulike tilnærminger til matematikk. Misoppfatningene som presenteres er viktige funn og interssante å undersøke nærmere. Elevenes utvikling fra å bruke hverdagslige uttrykk til å bli mer bevisst sitt bruk av Scratch-begreper er et interessant funn, som vil ses nærmere på.

5.0 Drøfting

I starten av oppgaven presenterte jeg følgende problemstilling: *«Hvilke tilnærminger til matematikk finnes det i arbeid med det blokkbaserte programmeringsspråket Scratch, og hvordan påvirkes elevenes affektive sider av arbeid med programmering?»*

I dette kapittelet skal jeg ved hjelp av teori som er presentert tidligere, drøfte hvordan mine analyser av eksemplene fra datamaterialet kan være med på å besvare problemstillingen. For å gjøre dette har jeg valgt å stille to spørsmål: «*Hvorfor skal elevene jobbe med programmering i matematikktimene?»* og *«Hvorfor er elevene så positive og engasjerte når de jobber med Scratch?»*.

5.1 Programmering som et verktøy for å arbeide med matematikk

Funnene fra programmeringsøktene viser at arbeidet har ført til matematisk resonnering og bruk av ulike problemløsningsstrategier, prosessorientert matematikk og utforsking hos elevene. Det kommer også frem at elevene i løpet av programmeringsøktene lærte seg en del av de grunnleggende konseptene ved programmering. Funnene i studien føyer seg dermed inn i rekken av andre studier som har vist at elever gjennom programmering kan utvikle matematiske ferdigheter og forståelse for grunnleggende konsepter i programmering (Brown et al., 2013; Fessakis, Gouli & Mavroudi, 2012; Iversen, 2015; Sáez-López, Román-González & Vázquez-Cano, 2016; Wilson & Moffat, 2010). Med disse funnene skal jeg drøfte de ulike tilnærmingene til matematikk som jeg har funnet i datamaterialet.

5.1.1 Matematisk problemløsning og bruk av ulike strategier

Datamaterialet viser flere eksempler der det tydelig kommer frem at elevene både har sett betydningen av rekkefølgen til klossene og funksjonen til gjentablokken. Forståelsen av gjentablokken kan knyttes til additiv og multiplikativ tenking. Fra å sette sammen fire «sett» av klosser for å lage ett kvadrat, har de sett effektiviteten av å i stedet sette én gjentablokk med tallet fire rundt ett enkelt «sett» av klosser. De fant ut at det var raskere, mer plassbesparende og enklere å bruke gjentablokken. Dette samsvarer med Wilson og Moffat (2010) og Sáez-López, Román-González og Vázquez-Cano (2016) som konkluderte med at elevene på bare noen uker med Scratch lærte seg de grunnleggende konseptene ved programmering som rekkefølge, gjentakelse og utvelgelse av riktige klosser.

Brown et al. (2013) konkluderer etter en studie at programmering i Scratch er en effektiv måte for å øve på matematisk resonnering, som man har behov for å kunne i problemløsning. Dette samsvarer med studien til Fessakis, Gouli og Mavroudi (2012) der det kom frem at at barna lærte problemløsningsstrategier og matematiske ferdigheter i form av telling, orientering i rutenett, forståelse av vinkler og sammenligning av tall. Elevene i studien oppdaget, på samme måte som elevene i studiene til Sáez-López, Román-González og Vázquez-Cano (2016); Wilson og Moffat (2010), at det var mer effektivt å ta i bruk en gjentablokk. På bakgrunn av at mine funn samsvarer med disse resultatene, mener jeg å kunne si at måten vi arbeidet med Scratch på også var en tilnærming til problemløsningsstrategier og matematisk resonnering. Elevene i studien min lærte problemløsningsstrategier i form av at de erfarte verdien av gjentablokken gjennom arbeidet med geometriske figurer, og på bakgrunn av dette endret de strategi for hvordan de skrev kode for å tegne ulike geometriske figurene.

Elevenes oppdagelse av verdien til gjentablokken er et funn på at det oppstår metakognisjon hos elevene, hvor de reflekterer over egen og medelevers læring og på den måten finner en bedre strategi for å løse oppgavene. Å reflektere over egen læring beskrives i *Fremtidens skole* som kompetanse i å lære, det kommer tydelig frem som viktig er en viktig ferdighet i *21st Century Skills* og *Computational Thinking*, og i fletten av matematiske ferdigheter finner vi denne egenskapen i tråden «Adaptive Reasoning».

5.1.2 Utforsking og utholdenhet

Gjennom arbeidet med programmering dukket det opp mange muligheter for utforsking, som flesteparten av elevene grep. De endret antall gjentakelser, byttet rekkefølge på klossene som blokkene var satt sammen av, og lagde flere blokker i samme program. Denne utforskingne var noe de brukte en god del tid på, og henger ammen med å ha utholdenhet i arbeidet. På en side kan en slik utholdenhet føre til at noen elever vil bli sittende med samme oppgave lenge uten å fortsette videre, på en annen side kan denne lange utforskingen skape forståelse som kanskje er viktigere enn antall oppgaver eller dagens mål. Det kommer frem i datamaterialet at den relativt lange tiden som ble brukt på utforsking førte til at elevene fant ulike strategier for å endre størrelse på kvadratet. Dette kunne de ta med seg videre når de skulle lage andre

geometriske figurer. Fordi datamaskinen kjapt gir tilbakemeldinger, ser elevene raskt om de har gjort rett. Van de Walle, Bay-Williams og Karp (2014) beskriver denne måte å tilnærme seg problemer på som det «å gjøre matematikk».

Da elevene arbeidet med å lage et kvadrat viste de en trang til å utforske gjennom å prøve og lage det en rekke ganger, og dermed utholdenhet i arbeidet. Å ha utholdenhet i arbeid med en vanskelig oppgave trekkes frem av The International Society for Technology in Education og The Computer Science Teachers Association (2011) i deres definisjon av Computational Thinking. Det kan tenkes at denne utholdenheten også kan komme på bakgrunn av endring i arbeidsmetode, og ikke nødvendigvis fordi elevene var spesielt glad i å arbeide med akkurat de oppgavene som ble gitt. Likevel var denne utholdenheten noe jeg opplevde i hver økt, og kan dermed anta at elevene sannsynligvis likte arbeidet de holdt på med. Denne utholdenheten elevene har med oppgavene kan knyttes til «Productive Struggle» (Van de Walle, Bay-Williams & Karp, 2014). Elevene må bruke tid for å klare og løse oppgavene, men de er innenfor rekkevidde av hva de kan klare, og i datamaterialet kommer det frem at elevene opplever en mestringsfølelse når de etter flere forsøk får det til.

I arbeidet med å tegne en regulær trekant, fikk elevgruppe 4 først frem en sekskant da de kjørte koden sin. Ved hjelp av prøving og feiling fant de sammenhengen mellom de to geometriske figurene og deres egenskaper i form av størrelsen på vinkler og antall kanter. Elevene tar i bruk det Wing (2006) beskriver som et kjennetegn ved Computational Thinking, de bryter problemet ned i mindre deler for å gjøre det mer overkommelig. Papert (1980) kaller denne metoden for «debugging», og ser på dette som en viktig del av programmeringen. Elevene kritiseres ikke for feilene de gjør, i stedet bruker de feilene på veien videre frem til riktig svar. Det kan tenkes at noen elever vil oppleve dette i mindre grad på bakgrunn av at de har et behov for å få riktig svar på første forsøk, eller fordi de er redde for å mislykkes, og dermed ikke ønsker å utforske i like stor grad som andre elever.

En studie om utforskende læring gjort av Meerbaum-Salant, Armoni og Ben-Ari (2011) trekker frem to uvaner de mener programmering i Scratch kan føre til og som senere kan påvirke den videre læringen av programmeringen når det blir tekstbasert. Det kan føre til

uvaner i form av å bare finne blokkene du trenger og sette de sammen eller uvaner i form av at man forenkler så mye at det tilslutt er vanskelig å forstå hva elevene har tenkt. Jeg har i mitt datamateriale ikke funnet tegn til at det har oppstått slike uvaner i klassen som arbeidet med Scratch, men det er ikke nødvendigvis slik at disse uvanene alltid vil oppstå. På en annen side skal man være bevisst på at Scratch er et verktøy og at det er behov for rammer rundt arbeidet. Dette samsvarer med flere studier som poengterer betydningen av lærerens rolle og gode undervisningsopplegg i arbeidet med Scratch (Fessakis, Gouli & Mavroudi, 2012; Kalelioglu & Gülbahar, 2014; Meerbaum-Salant, Armoni & Ben-Ari, 2013).

5.1.3 Matematikk som prosess og som produkt, instrumentell og relasjonell forståelse

Ved flere anledninger ytret elevene at det var vanskeligere å programmere geometriske figurer enn å tegne de. For noen av elevene kan det virke som at det de tidligere har sett på som en geometrisk figur er noe man tegner enten på 1, 2, 3, 4 eller i blinde. Disse oppfatningene kan knyttes til det Skemp (1976) beskriver som instrumentell forståelse. Elevene ser på det å tegne ulike geometriske figurer kun som noe de gjør, og har ikke nødvendigvis tenkt igjennom egenskapene til figurene når de utfører arbeidet. I arbeidet med Scratch var elevene nødt til å forklare hverandre og datamaskinen hvordan figurene skulle tegnes, det var ikke bare å ta en blyant og gjøre en bevegelse. Man kan anta at det i arbeidet med Scratch kan oppstå det Skemp (1976) ser på som den eneste formen for forståelse, den relasjonelle forståelsen. Elevene er nødt til å forstå hvilke egenskaper de geometriske figurene har for å kunne vite hvordan de skal programmere. Å programmere et kvadrat kan ses på som en prosess hvor en rekke klosser er satt sammen til blokker, og som et produkt når blokkene er laget, blitt til et program og kan brukes om igjen. Instrumentell og relasjonell forståelse kan ifølge Van de Walle, Bay-Williams og Karp (2014) knyttes til de to stråene «Conceptual understanding» og «Procedural Fluency» i fletten av matematiske ferdigheter. Denne dypere forståelsen for et fagspesifikt område omtales i rapporten til Ludvigsenutvalget som behovet for dybdelæring i elevenes læring (NOU 2015:8).

5.1.4 Sammenligning av arbeidsmetode

I analysekapittelet kommer det frem at elevene sammenligner arbeid i Scratch på datamaskin med arbeid i matematikkboka. Flesteparten mener det er vanskeligere og morsommere å jobbe

med Scratch, selv om enkelte mener det er vanskeligere i matematikkboka. Hvorfor synes elevene det er morsommere å arbeide med geometriske figurer i Scratch, selv om det er vanskeligere? Det kan tenkes at dette kan ha en sammenheng med at de får tilbakemelding på om de har gjort rett med en gang de kjører koden, dette ser vi i analysekapittelet der en elevgruppe forteller om hvordan koden enten tegner en trekant eller ikke. Ved å finne feil og endre disse, ved såkalt «debugging», kan de prøve på nytt og se om endringen har ført til at figuren blir tegnet riktig (Papert, 1980). Kan det tenkes at elevene har lengre utholdenhet med å prøve flere ganger i Scratch enn i matematikkboka? I programmering kan man enkelt slette det man har laget ved et klikk og prøve på nytt, imens det i matematikkboka er mer tidkrevende fordi det trengs linjal, blyant, passer og viskelær. De får også raskere tilbakemelding enn om de skulle ha tegnet på nytt i boka og spurt læreren om svaret.

Da elevene arbeidet med Frost på CodeStudio påpekte de at det kunne bli litt kjedelig fordi alle «tallene» stod der. Elevene hadde alternativer de kunne velge mellom, og de visste dermed at et av alternativene var riktig. Ved å ha tomme felt ville elevene i større grad blitt tvunget til å reflektere over egenskaper til geometriske figurer.

5.1.5 Matematikk som språk, deltakelse og samtale

Scratch kan ses på som det Papert (1980) beskriver som en microworld, en slags verden eller sted der man kan arbeide med en bestemt del av matematikken. Scratch ble et sted der elevene kunne arbeide med geometriske figurer og deres egenskaper. Selve klasserommet i sin helhet kan ses på som det Papert (1980) beskriver som et MathLand, hvor man legger til rette for matematisk kommunikasjon og mulighet for matematikklæring. Ved å hjelpe hverandre og forklare hvordan de skulle løse oppgavene ble elevene «tvunget» til å forklare og prate matematikk til hverandre og til datamaskinen. I datamaterialet ser vi tydelig at elevene tar i bruk Scratch-begreper når de samtaler med hverandre, og det ser ut til at dette skjer på bakgrunn av at de ønsker å være presise. De er nødt til å bruke begrepene i Scratch når de skal forklare medelever hvor de skal finne de ulike klossene, slik at de forstår hverandre tydelig.

Språket blir et viktig redskap i utforskingen og læringen til elevene, noe som er et sentralt begrep i sosiokulturell læringsteori. Elever lærer i samspill med andre og med språket som et

medierende redskap (Säljö, 2001). I min studie kan Scratch ses på som et medierende verktøy i læringen av geometri, gjennom arbeid med programmeringsspråket blir geometrien meningsfull for elevene.

At elevene prater matematikk i programmeringsøktene kommer frem av samtalen der elevene har resonnert seg frem til sammenhengen mellom en trekant og en sekskant, og i en kommentar der de samme elevene regner seg frem til hvor stor hver vinkel i en trekant er. Når elevene setter sammen blokker, må de vite antall grader og antall ganger de må vende. Papert (1980) påpeker at dersom barn lærer seg å «prate» med en datamaskin, vil de lære matematikk som et levende språk. I datamaterialet kommer det frem at elevene mener de må forklare datamaskinen hva den skal gjøre og «hjelpe» den til å forstå hva den skal gjøre.

Det sosiokulturelle perspektivet på læring legger vekt på samarbeid og deltakelse som en viktig faktor i all læring uavhengig fag og tema (Säljö, 2001). Flesteparten av elevene ytret gjennom samtaler og intervju at samarbeid var noe positivt, og det var en generelt sett positiv holdning til samarbeid i klasserommet. Elevene trakk frem både fordeler og ulemper med å være flere om en oppgave. Av ulemper med samarbeid som ble nevnt, var dette hovedsakelig at man kan være uenige i valg som må tas, og at det derfor kan være utfordrende å samarbeide. Positivt med samarbeid mente de var at de kunne dele ideer, inspirere og hjelpe hverandre. Likevel vil ikke alltid samarbeid være den beste arbeidsformen, men i dette arbeidet spilte det en positiv rolle. En studie gjort av Taylor, Harlow og Forret (2010) konkluderte med at Scratch gir muligheter for samarbeid i klasserommet, noe jeg med mine funn også mener å ha belegg for å si. I datamaterialet kommer det av elevenes utsagn frem at de har behov for å bli sett og anerkjent, samtidig som de ser verdien av medelevers kunnskap. Det oppstår et læringsfellesskap i klasserommet som elevene føler tilhørighet til (Wenger, 1998).

Det å samarbeide og delta poengteres som positivt både i Fremtidens skole, fremheves som en viktig del av Computational Thinking og som en viktig ferdighet i det 21.århundre (Abbot, 2014; NOU 2015:8; The International Society for Technology in Education & The Computer Science Teachers Association, 2011). Det skal allikevel poengteres at det alltid må være en

balanse i arbeidet som gjøres i klasserommet, alt kan ikke gjøres gjennom samarbeid. Alle har behov for å kunne jobbe individuelt i sitt eget tempo og med oppgaver på sitt faglige nivå.

5.1.6 Læring gjennom å rette opp i misoppfatninger hos elevene

Et av funnene i mitt datamateriale var at det underveis i arbeidet med programmeringen dukket opp enkelte misoppfatninger eller misforståelser i elevenes arbeid med geometriske figurer. Det er viktig å poengtere at jeg var i en posisjon der jeg kunne oppdage misoppfatninger på bakgrunn av at jeg hadde lydopptak av flere av elevgruppene. Dette ser vi blant annet i samtalen der elevgruppe 4 mener man må gjenta klossene med «gå *x* steg» og «vend *x* grader» 37 ganger for å programmere en sirkel. Tallet 37 ser det ut til at de har tatt med seg fra arbeidet med Frost-opplegget, der de skulle lage mange sirkler både oppå hverandre og etter hverandre. Å De har latt Scratch-figuren tegne flere sirkler oppå hverandre, og ikke fokusert på hvordan man tegner nøyaktig én sirkel. Å ta med seg tallet 37 i videre arbeid med sirkler i matematikkboka kan fort føre til misoppfatninger, da er det viktig å få tak i denne slik at man i stedet kan bruke det til å lære om egenskapene og kjennetegnene til en sirkel. Til tross for at elevgruppe 4 først fikk opp en sekskant, klarte de allikevel å få frem en trekant etter noen forsøk. Denne oppdagelsen er en mulighet til å lære elevene om indre og ytre vinkler, og sammenhengen mellom en trekant og en sekskant.

I økten med felles programmering på SMARTboard dukket det opp en misoppfatning da elevene skulle programmere en trekant. Jeg oppfattet det som at den ene eleven hadde et inntrykk av at vinkler på 90 grader ikke eksisterer i trekanter. På en side kan det tenkes at eleven har en misoppfatning når det gjelder egenskapene til en trekant, sett fra en annen side kan det hende at vårt arbeid med kun regulære mangekanter har påvirket eleven til å kun tenke på mangekanter som har like store vinkler. Med ytringen *«Den har ikke 90 grader, den har for eksempel 45 grader»* underbygger dette påstanden om at det har skjedd en misoppfatning, da 60 grader som er vinkelstørrelsen i en likesidet trekant, ikke nevnes. Likevel kan det tenkes at det ikke er intuitivt for eleven å nevne 60 grader, da vi kun arbeidet med regulære mangekanter i programmeringsøktene, og av den grunn ikke arbeidet med trekanter som hadde annet enn vinkler på 60 grader. Fordi denne situasjonen oppstod da vi programmerte på SMARTboard hadde vi mulighet til å bruke denne misoppfatningen til læring, som kommer frem av datamaterialet.

Mine undersøkelser avdekket misoppfatninger, men takket være lydopptak og felles programmering kunne vi rette opp i disse. Det er viktig å poengtere at dette var spesielt med mine undersøkelser, og at dette kan være noe man bør være observant på når man tar programmering inn i undervisningen. Ved å være bevisst på dette, kan man som lærer lage undervisningsopplegg hvor man har tenkt igjennom på forhånd hvordan misoppfatninger kan oppstå og hvordan man kan håntere disse. Jeg har erfart at en felles programmerinsøkt på SMARTboard som oppsummering er en mulig måte å oppdage misoppfatninger på, og bruke de til læring.

5.2 De affektive sidene ved arbeid med programmering

Hvorfor fenger Scratch slik det gjør, og hvorfor er elevene så positive og engasjerte? Funnene mine viser at elevene opplever arbeid med Scratch som både motiverende og engasjerende, de affektive sidene påvirkes i positiv retning. Med dette føyer min studie seg inn i rekken av flere andre studier hvor det har kommet frem at arbeid med visuell programmering i Scratch har positive innvirkninger på elevenes affektive sider, og at positive affektive sider er like viktig som programmeringskunnskapen i seg selv (Malan & Leitner, 2007; Meerbaum-Salant, Armoni & Ben-Ari, 2013; Sáez-López, Román-González & Vázquez-Cano, 2016; Wilson & Moffat, 2010). Hvorfor fenger Scratch slik det gjør, og hvorfor er elevene så positive og engasjerte?

Ytringen til 4B: *«Det er skikkelig artig å finne ut av ting»,* og 5B sin ytring: *«Klarte det, jeg klarte det!»* får tydelig frem en positiv holdning og et engasjement hos elevene i arbeidet med Scratch. De positive holdningene av programmeringen kan komme av at Scratch var nytt for alle i klassen, og at dette var en helt ny form for arbeidsmetode og en variasjon fra matematikkundervisningen de til vanlig har. Dette samsvarer med en studie gjort av Wilson og Moffat (2010) som poengterer at de ikke kan være sikre på hvor representativt det positive engasjementet til elevene er, og at de kan ikke utelukke at det kan være på bakgrunn av at Scratch var nytt for elevene i studien. På en annen side påpeker de at det positive engasjementet også kan komme av variasjonen, den økende vanskelighetsgraden og studiens varighet, noe som også stemmer overens med studien min. Til tross for at flesteparten av

elevene var positive, vil det alltid være vanskelig å nå absolutt alle elever, og sørge for at alle er motiverte til enhver tid. Disse tre faktorene kan være viktige å ha i bakhodet dersom man ønsker å ta i bruk programmering i undervisningen. Samtidig vil det alltid være vanskelig å treffe alle elever uansett hvilken arbeidsmetode som tas i bruk i klasserommet, da ulike metoder fenger ulike elever.

Wilson og Moffat (2010) påpeker viktigheten av motivasjon for å kunne lære, og at denne motivasjonen kan opprettholdes med positive affekter. Både elever og lærer i studien påpekte programmeringen som motiverende og at det var en måte å arbeide på som de likte. I studien til Sáez-López, Román-González og Vázquez-Cano (2016) viser de til positive funn for at visuell programmering fører til blant annet økt motivasjon, noe som vi kan finne tegn til i analysekapittelet der en elev trekker frem arbeid på datamaskin som mer motiverende enn arbeid i matematikkboka.

Etter flere programmeringsøkter med Scratch kan man anta at mange elever ble trygge på å bruke Scratch som verktøy for å arbeide med geometriske figurer og deres egenskaper på. Jeg opplevde at Scratch som en arbeidsmetode som traff elevene uavhengig faglig nivå i matematikk. I programmeringsøktene fikk elevene lære seg viktigheten av å kjenne til egenskapene til ulike geometriske figurer, fordi de opplevde at de ikke kunne programmere uten å kunne dette. Å arbeide med matematikk ved å ta i bruk digitale hjelpemidler som Scratch kan knyttes til de grunnleggende ferdighetene i læreplanen for matematikk fellesfag, der det poengteres at elever skal bruke digitale verktøy i løsingen av matematiske problem (Utdanningsdirektoratet, 2013). Likevel må man huske på at en ny arbeidsmetode for enkelte elever kan oppfattes som noe usikkert og ukjent. Dette kan være en grunn til at noen få av elevene var mer nøytrale til arbeidet med Scratch. Wilson og Moffat (2010) fikk elevene etter hver økt med Scratch til å rangere hva de følte om arbeidet, der det kun var et par elever som svarte med et nøytralt ansikt og resten med et fornøyd ansikt. Til tross for at jeg ikke gjennomførte en slik rangering, mener jeg at analysen min gir meg belegg for å ha samme inntrykk ved at flertallet viser tegn til å like arbeidsmetoden. Læreren i Wilson og Moffat (2010) sin studie påpekte at flere svake elever presterte bedre enn normalt, og at noen sterke elever presterte svakere enn flere av de andre elevene. Dette hadde ikke læreren noen god

forklaring på hvorfor skjedde. I mitt datamateriale i form av mine observasjoner av elevene, opplevde jeg i motsetning til Wilson og Moffat (2010) ingen tegn til dette. Av læreren til elevene fikk jeg underveis informasjon om hvordan elevene til vanligvis presterer, og det er på bakgrunn av dette jeg kan si at jeg ikke opplevde det samme som Wilson og Moffat (2010). En grunn til det antar jeg er fordi mitt åpne opplegg kunne følges av alle elevene, uansett faglig nivå. Jeg var tydelig på at det var mange veier til mål, at det ikke bare var en riktig måte å gjøre det på og at det ikke var om å gjøre og bli fortest ferdig. Med dette mener jeg å ha belegg for å si at programmering kan være en arbeidsmetode som kan treffe alle elever bare lærere legger gode rammer rundt undervisningen og har tilpassede undervisningsopplegg, slik studien til Meerbaum-Salant, Armoni og Ben-Ari (2013) påpeker.

6.0 Konklusjon

I denne oppgaven har jeg med relevant teori og innsamlet datamateriale, drøftet og belyst problemstillingen: *«Hvilke tilnærminger til matematikk finnes det i arbeid med det blokkbaserte programmeringsspråket Scratch, og hvordan påvirkes elevenes affektive sider av arbeid med programmering?»*. Med dette kapittelet presenteres avsluttende refleksjoner, samt en beskrivelse av videre forskning på feltet jeg har undersøkt.

6.1 Avsluttende refleksjoner

Vil det være fornuftig å innføre programmering i norsk skole? I min studie ønsket jeg å se på hvilke tilnærminger til matematikk som finnes i arbeid med programmering, og hvordan elevenes affektive sider påvirkes ved denne arbeidsmetoden. Helt fra begynnelsen av oppgaveskrivingen hadde jeg en hypotese om at programmering kan være med på å berike matematikkundervisningen. I datamaterialet mitt har jeg funnet eksempler som viser at elevene gjennom arbeid med programmering, har opplevd flere ulike tilnærminger til matematikk, blant annet når det gjelder problemløsningsstrategier og matematisk resonnering. På grunn av Scratch sine innebygde muligheter til å arbeide med vinkler og lengde, samt muligheter til å få ting til å skje gjentagende ganger, har dette vist seg å være et veldig nyttig verktøy i arbeidet med geometri. Dette ga elevene blant annet muligheter til å utforske generelle mønstre i geometrien, noe som ikke er like enkelt med penn og papir. Det er også flere eksempler i datamaterialet som viser at programmeringen har hatt positive innvirkninger på elevenes affektive sider. Det kommer tydelig frem et engasjement, en positiv holdning og en mestringsfølelse hos elevene. Min studie har vist at bruk av programmeringsspråket Scratch i arbeid med geometri i matematikkundervisningen, ga rom for at elevene fikk oppleve både en skaperglede og en utforskertrang i timene. Med disse funnene har jeg fått styrket mitt syn om at programmering med fordel kan innføres og brukes som metode og verktøy i matematikkundervisningen.

Med nye verktøy og undervisningsformer, og nye kompetanser og ferdigheter som vil kreves av elevene, kommer det også nye utfordringer som må tas hensyn til. Jeg fant i min studie funn på at misoppfatninger kan oppstå, og at det er viktig å være observant på dette for å kunne bruke det til læring. Jeg mener at min studie kan være med på å gi et innblikk i hvordan

programmering kan berike matematikkundervisningen, og hvordan vi kan møte fremtidens nye krav om ferdigheter og kompetanser elevene vil ha behov for.

Med studien har jeg fått et innblikk i hvordan jeg som fremtidig lærer kan ta i bruk programmering i matematikkundervisningen, og fått innblikk i studier som er gjort og som pågår innen dette feltet. Jeg ønsker med studien å inspirere studenter til å skrive videre innen dette emnet, slik at vi får mer forskning på bruk av programmering i norsk skole, og bedre innsikt i hvordan programmering kan berike undervisningen.

6.1 Pågående forskning og veien videre

I min studie har jeg kun undersøkt noen av de mange mulighetene som åpner seg når vi kombinerer matematikk og programmering. Videre forskning på programmering i norske klasserom vil være med på å definere hvilken plass programmering får i fremtidens skole og undervisning, både nasjonalt og internasjonalt. Rapporten til Ludvigsenutvalget om fremtidens skole og pilotprosjektet som starter opp med programmering som valgfag i ungdomskolen, er viktige bidrag til hvordan norsk skole kan bli og hvordna undervisning kan gjennomføres i fremtiden. Det pågår for tiden også forskning med fokus på matematikk og programmering i skolen ved Høgskolen i Bergen og ved Høgskolen i Østfold.

I *Fremtidens skole* trekkes det frem at elevene vil ha behov for dybdelæring og nye kompetanser for å lykkes i fremtidens samfunn. Flere frykter at noe av innholdet i dagens pensum må bort for å gi plass til programmering, men som denne masteroppgaven viser er det mulig å bruke programmering som en metode og et pedagogisk verktøy for å berike undervisningen, og gi elevene flere perspektiver ved matematikk. Det bør også nevnes at programmering har et bredere bruksområde enn kun som et verktøy i matematikk. Det vil være spesielt interessant å forske videre på bruken av programmering ved tverrfaglig arbeid i skolen, med og uten tilknytning til matematikk. Kunnskapsdepartementet har vedtatt å gjennomføre en revidering av alle fag i skolen, og det vil da bli tydeliggjort om de mener at programmering bør ha en tydeligere rolle i fremtidens skole eller ikke (Meld. St. nr. 28 (2015-2016)).

I august 2014 startet de tre forskerne Johan Lie, Inge Olav Hauge og Ketil Arne Bergesen ved seksjon for matematisk fagdidaktikk ved Høgskolen i Bergen opp et prosjekt med tittel: *«Læring av matematikk og algebra i programmering»* (Lie, Hauge & Bergesen, 2014), der programmeringsspråket Scratch brukes som pedagogisk verktøy. I prosjektbeskrivelsen står det blant annet: *«[…]hvordan elevers programmeringsaktiviteter kan bidra til økning av elevers matematikkforståelse. Prosjektet fokuserer på matematisk språk, læring, holdninger til faget og motivasjon for å lære matematikk »* (Lie, Hauge & Bergesen, 2014). Prosjektet pågår fremdeles, og det vil være spennende å følge med når det etter hvert vil bli publisert funn.

I juni 2015 startet lærerutdanningen og informatikkavdelingen ved Høgskolen i Østfold et felles forskningsprosjekt med tittel: *«Programmering som fag og pedagogisk verktøy i skolen»* (Høgskolen i Østfold, 2015). Med prosjektet ønsker de å se på programmering både som eget fag og som pedagogisk verktøy. I prosjektbeskrivelsen står det blant annet: «*Vi har stor tro på at programmering, der problemer brytes opp i enkeltdeler og deretter beskrives/visualiseres som en prosess som skal utføres, kan motivere for læring og gi bedre og dypere forståelse for matematikk, realfag og teknologi*.» (Høgskolen i Østfold, 2015, s. 1).

Høsten 2016 starter 146 norske skoler opp med programmering som valgfag på ungdomskolen, et pilotprosjekt som skal gjennomføres over tre skoleår. I læreplanen til faget står det blant annet at elevene skal: *«Utvikle og feilsøke programmer som løser definerte problemer, inkludert realfaglige problemstillinger og kontrollering eller simulering av fysiske objekter»* (Utdanningsdirektoratet, 2016b, s. 5). Elevene skal med faget lære seg programmering, men også kunne knytte det opp mot matematikk og realfag. Det er naturlig å forvente at erfaringer fra dette pilotprosjektet vil kunne gi ytterligere innspill til hvordan matematikk og programmering kan kombineres i fremtidens skole, også innenfor matematikkfagets rammer.

Høsten 2016 har Finland vedtatt å innføre programmering som en del av matematikkundervisningen, under hovedområdet «Matematisk tenkning og matematiske

metoder», på alle klassetrinn i grunnskolen (Sevik, 2015). I tiden som kommer vil det være viktig og interessant for norske forskere å følge med på denne innføringen av programmering, som inspirasjon til hvordan norsk skole og undervisning kan bli i fremtiden.

7.0 Litteraturliste

Abbot, S. (2014). 21st Century Skills. *The Glossary of Education Reform*. Hentet fra http://edglossary.org/21st-century-skills

Alrø, H. & Skovsmose, O. (2004). *Dialogue and Learning in Mathematics Education. Intention, Reflection, Critique*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M. & Rumble, M. (2012). Defining Twenty-First Century Skills. I P. Griffin, B. McGaw, & E. Care (Red.), *Assessment and Teaching of 21st Century Skills* (s. 17-68): Springer.

Boaler, J. (2008). *The Elephant in the Classroom. Helping Children Learn and Love Maths.* : Souvenir Press.

Brown, Q., Mongan, W., Kusic, D., Garbarine, E., Fromm, E. & Fontecchio, A. (2013). Computer Aided Instruction as a Vehicle for Problem Solving: Scratch Programming Environment in the Middle Years Classroom.

Creswell, J. W. (1998). *Qualitative inquiry and research design: Choosing among five traditions*. Thousand Oaks California, USA: SAGE Publications, Inc. .

Euractiv. (2015, 21.04.2016). EU Code Week 2015. Hentet fra http://en.euractiv.eu/wp-content/uploads/sites/2/special-report/euractiv_special_report_-_eu_code_week_2015-1.pdf

Fessakis, G., Gouli, E. & Mavroudi, E. (2012). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education, 63 (2012)*, 87-97.

Gold, R. L. (1958). Roles in sociological field observations. *Social Forces, 36*(3), 217-223.

Greenhill, V. (2009). P21 framework definitions document. *Retrieved December, 15*, 2010.

Høgskolen i Østfold. (2015). Programmering som fag og pedagogisk verktøy i skolen. Hentet fra http://www.ia.hiof.no/~janh/progped_prosjektbeskrivelse_sammendrag.pdf

Iversen, S. (2015). *Koding som digital grublis. En kvalitativ studie om hvordan elevenes læringsstrategier påvirkes gjennom programmering.* (Mastergradsavhandling. Universitetet i Tromsø - Norges Arktiske Universitet). Hentet fra http://munin.uit.no/bitstream/handle/10037/8089/thesis.pdf?sequence=2&isAllowed=y

Johannessen, A. & Tufte, P. A. (2002). *Introduksjon til samfunnsvitenskapelig metode*. Oslo: Abstrakt forlag AS.

Kafai, Y. B. & Burke, Q. (2015). Computer programming goes back to school. *Education Week*, 61-65.

Kalelioglu, F. & Gülbahar, Y. (2014). The effects of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13*(1), 33.

Kvale, S. & Brinkmann, S. (2009). *Det kvalitative forskningsintervju*. Oslo: Gyldendal Norsk Forlag AS

Leming, T. (2009). Hvor ble JEG av? I R. Jakhelln, T. Leming, & T. Tiller (Red.), *Emosjoner i forskning og læring* (1 ed., s. 35-50). Tromsø: Eureka Forlag.

Lie, J., Hauge, O. I. & Bergesen, K. A. (2014). Læring av matematikk og algebra i programmering. Hentet fra https://www.cristin.no/app/projects/show.jsf?id=456274

Malan, D. J. & Leitner, H. H. (2007). *Scratch for budding computer scientists.* Paper presented at the ACM SIGCSE Bulletin.

Meerbaum-Salant, O., Armoni, M. & Ben-Ari, M. (2011). *Habits of programming in scratch.* Paper presented at the Proceedings of the 16th annual joint conference on Innovation and technology in computer science education.

Meerbaum-Salant, O., Armoni, M. & Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23*(3), 239-264.

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

National Research Council. (2001). *Adding it up: Helping children learn mathematics* (J. Kilpatrick, J. Swafford, & B. Findell Red.). Washington, DC: National Academy Press.

Nilssen, V. L. (2012). *Analyse i kvalitative studier: den skrivende forskeren*. Oslo: Universitetsforlaget.

NOU 2015:8. *Fremtidens skole. Fornyelser av fag og kompetanser*. Oslo: Departementenes sikkerhets- og serviceorganisasjon, Informasjonsforvaltning Hentet fra https://www.regjeringen.no/no/dokumenter/nou-2015-8/id2417001/?ch=1&q=.

Opplæringslova. (1998). *Lov om grunnskolen og den vidaregåande opplæringa (opplæringslova).* Hentet fra https://lovdata.no/dokument/NL/lov/1998-07-17-61/KAPITTEL_1#KAPITTEL_1.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas* (2. utgave ed.). New York: Basic Books.

Patton, M. Q. (1987). *How to Use Qualitative Methods in Evaluation*. California, USA: SAGE Publications, Inc. .

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

Resnick, M., Maloney, J., Hernández, A. M., Rusk, N., Eastmond, E., Brennan, K., . . . Kafai, Y. (2009). Scratch: Programming for All. *Communications of the ACM, 52*(11), p. 60-67.

Rossen, E. (2015). programmering - IT. Hentet fra https://snl.no/programmering%2FIT

Sáez-López, J.-M., Román-González, M. & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using “Scratch” in five schools. *Computers & Education, 97*, 129-141.

Sevik, K. (2015). *Koding i skolen. Notat fra Senter for IKT i utdanningen.* . http://iktsenteret.no/sites/iktsenteret.no/files/attachments/koding_i_skolen_-_sikt-notat_nr_2_-_temakonferanse_2015.pdf

Skemp, R. R. (1976). Relational understanding and instrumental understanding. *Mathematics Teaching, 77*.

Skott, J., Jess, K. & Hansen, H. C. (2008). *DELTA. Fagdidaktik.* (1 ed.): Forlaget Samfundslitteratur.

Strauss, A. & Corbin, J. (1998). *Basics of Qualitative Research. Techniques and Procedures for Developing Grounded Theory*. (2 ed.). California: SAGE Publications, Inc.

Säljö, R. (2001). *Læring i praksis - Et sosiokulturelt perspektiv*. Oslo: Cappelen Akademisk Forlag.

Taylor, M., Harlow, A. & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia-Social and Behavioral Sciences, 8*, 561-570.

The Barefoot Project. (2014). Computational Thinking. Hentet fra http://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/computational-thinking/

The International Society for Technology in Education & The Computer Science Teachers Association. (2011). Operational Definition of Computational Thinking for K–12 Education. Hentet fra http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf?sfvrsn=2

The White House. (2016, 30.01.2016). Giving Every Student an Opportunity to Learn Through Computer Science For All. *Weekly Address.* Hentet fra https://www.whitehouse.gov/the-press-office/2016/01/30/weekly-address-giving-every-student-opportunity-learn-through-computer

Tjora, A. (2012). *Kvalitative forskningsmetoder i praksis*. Oslo: Gyldendal Norsk Forlag AS.

Utdanningsdirektoratet. (2013). Læreplan i matematikk fellesfag. Hentet fra http://data.udir.no/kl06/MAT1-04.pdf?lang=nno

Utdanningsdirektoratet. (2016a, 01.02.2016). Forsøk med programmering som valgfag. Hentet fra http://www.udir.no/Lareplaner/Forsok-og-pagaende-arbeid/forsok-med-programmering-som-valgfag/

Utdanningsdirektoratet. (2016b). Forsøkslæreplan i valgfag programmering. Hentet fra http://www.udir.no/globalassets/filer/lareplan/forsok/forsokslareplan-programmering-som-valgfag.pdf

Van de Walle, J. A., Bay-Williams, J. M. & Karp, K. S. (2014). *Elementary and Middle School Mathematics: Pearson New International Edition: Teaching Developmentally* (8 ed.): Pearson.

Wenger, E. (1998). *Communities of Practice. Learning, Meaning, and Identity.* New York: Cambridge University Press.

West, D. M. (2011). Using technology to personalize learning and assess students in real-time. *Washington, DC: Brookings Institution*.

Wilson, A. & Moffat, D. C. (2010). Evaluating Scratch to introduce younger schoolchildren to programming. *Proceedings of the 22nd Annual Psychology of Programming Interest Group (Universidad Carlos III de Madrid, Leganés, Spain*.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM, 49*(3), 33-35.

Vedlegg 1: En beskrivelse av de ulike programmeringsøktene

1. Introduksjon av CodeStudio

Introduserte elevene for CodeStudio på code.org, et nettsted med mange ulike oppgaver til ulike programmeringsspråk. På disse sidene gjøres oppgavene i selve nettleseren, i motsetning til de senere øktene der de var nødt til å lese oppgavene på én nettside, og så programmere i Scratch. Elevene gjennomførte «Frost» og «Angry Birds», der begge oppleggene bygger på blokkbasert programmering.

2. Introduksjon av Scratch

Introduksjon av det blokkbaserte programmeringsspråket Scratch gjennom filmklipp fra code.org og gjennom å ha Scratch på SMARTboard, slik at jeg kunne vise elevene hvordan de skulle komme i gang. Elevene arbeidet så i par med oppgaven «Felix og Herbert» fra Lær Kidsa Koding sine oppgavesider: kodeklubben.github.io.

3. Å bli enda bedre kjent med Scratch

Vi repeterte litt hva vi hadde gjort forrige gang jeg var i klassen. Elevene fortalte litt om sine opplevelser med Scratch. Elevene fortsatte å arbeide med Scratch for å bli kjent med flere av funksjonene og mulighetene programmet åpner for. Da vi hadde opprettet en brukerprofil på scratch.mit.edu, kunne de som ikke ble ferdig med forrige oppgave gjøre denne. Resten av elevene startet på «Astrokatt» fra oppgavesidene til Lære Kidsa Koding, og arbeidet med denne resten av økten.

4. Å skape geometriske figurer i Scratch

I denne økten skulle elevene tegne geometriske figurer som kvadrat, rektangel, trekant og sirkel i Scratch. Med bakgrunn i de tidligere øktene hadde elevene nå blitt kjent med mange funksjoner i programmet og hvordan man lager ulike programmer ved å sette sammen forskjellige klosser. Her måtte elevene finne frem tidligere kunnskap om geometriske figurer og deres egenskaper for å kunne tegne figurene.

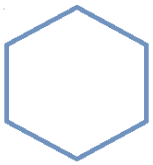
5. Vi lager eget spill

I denne timen fikk elevene fritt velge et spill fra Lær Kidsa Koding sine oppgavesider som de ønsket å programmere med bruk av Scratch.

6. Oppsummering og felles programmering på SMARTboard

Vi satte opp Scratch på SMARTboard og oppsummerte det elevene hadde lært om å tege geometriske figurer i Scratch ved å programmere de i fellesskap. Alle elevene flyttet seg frem til SMARTboarden og det var fritt frem til å hjelpe til. Samarbeid var i fokus og elevene måtte spille på hverandre og det de husket og hadde lært.

Vedlegg 2 – undervisningsopplegg

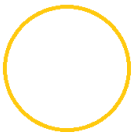


Geometriske figurer i Scratch

Dette undervisningsoppplegget er laget på bakgrunn av opplegget som ble gjennomført med elevene i min innsamling av datamateriale. Opplegget er tenkt til å vare i 2-3 skoletimer, litt ettersom hvor mye programmering elevene har holdt på med tidligere. Alle oppgavene er tenkt som samarbeidsoppgaver. Man trenger ikke nødvendigvis gjennomføre første programmeringsøkt dersom elevene allerede kjenner til programmering, men dette var en fin introduksjon for mine elever hvor programmering var noe helt nytt.

1.økt (enkelttime)

Tema: introduksjon av geometriske figurer og mønstre ved hjelp av Frost-opplegget i CodeStudio som finnes på nettsidene til code.org/frozen. De som ble ferdige med Frost kunne velge mellom *Angry Birds* eller *Kunstner*.



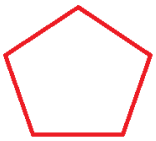
2.økt (1-2 timer)

Tema: Tegne geometriske figurer i Scratch, og enten gjenskape snøkrystaller fra Frost eller lage egne. Først hadde vi en liten introduksjon av nyttige klosser

- penn på/av
- lag et program slik at det som tegnes, slettes når en gitt tast trykkes
- velge en x- og y-koordinat som figuren flytter seg til når det grønne flagget klikkes

Oppgave til elevene (dette kan man ha fremme på SMARTboard så elevene ser oppgavene):

- velg en figur dere vil bruke
- sett sammen klosser til blokker slik at Scratch-figuren tegner et kvadrat, et rektangel, en trekant og en sirkel



- gjenskap snøkrystaller fra Frost eller lag deres egne
- utfordring: klarer dere å tegne en femkant og en sekskant?

3.time (enkelttime)

Tema: felles programmering av geometriske figurer.

Spørsmål til elevene:

- hvilke geometriske figurer klarte vi å tegne forrige time?

Oppgaver:

- elevene skal som klasse sammen programmere de ulike geometriske figurene i Scratch (bruk SMARTboard slik at elevene kan programmere sammen, resten av elevene flytter stolene sine frem til SMARTboarden slik at de opplever at de er sammen om dette)
- hvordan kan vi få til å tegne et kvadrat, et rektangel, en trekant og en sirkel?
- la elevene komme opp og programmere på SMARTboard (alene eller i par)
- dette er et samarbeid, dere må diskutere hva dere vet om figurene slik at dere sammen får det til. Snakk sammen og finn løsninger.
- hvis tid: klarer dere sammen å finne ut hvordan vi kan tegne en femkant og en sekskant?
- hjelp elevene til å opprettholde samarbeidet og diskusjonen dersom det trengs. Hvorfor velger dere den klossen? Hvorfor vender dere så mange grader? Hvorfor bruker dere gjentablokken?

Erfaringer

- den siste økten opplevde jeg som svært nyttig for elevene både faglig og når det kommer til samarbeid, deltakelse og kommunikasjon
- vi fikk snakket om begreper som vinkler, grader, kanter, indre og ytre vinkler

Vedlegg 3: Beskrivelse av elevgrupper

Elevgruppe 1

1A, 1B og 1C

Dette er en gruppe på tre elever der to av dem til vanlig ikke gjør noe mer enn det de er nødt til. Elevene har, i følge læreren, ikke så stor utholdenhet når det kommer til oppgaveløsing i matematikktimene til vanlig. Elevgruppen var åpne og positive til programmering som en ny måte å arbeide på i matematikktimene. Samtidig hadde de behov for at det var litt nytt i hver programmeringøkt slik at det ikke ble kjedelig.

Elevgruppe 2

2A og 2B

Disse elevene er til vanlig to flittige elever i matematikktimene. De er opptatt av at samarbeid er bra, de ønsker å utforske, de ønsker å gjøre det best mulig, og ting skal bli helt riktig. Denne elevgruppen er positiv til stort sett alt jeg innfører. De er åpne for nye ting, men samtidig ønsker de å gjøre seg ferdig med opplegget og ikke bare hoppe videre til neste med en gang. De tar utfordringer på strak arm og spiller på hverandre.

Elevgruppe 3

3A og 3B

Dette er to elever, der en er svak og en er sterk. Dette kom alikevel ikke særlig frem i datamaterialet, da koding var nytt for begge to. Dette er to elever som gjerne er muntlig aktive, som ofte har behov for bekreftelse. Eleven som er litt svak fikk brukt andre ferdigheter, eleven var på gruppa hvor de gikk fra sekskant til trekant. Han så sammenhengen, og fant svaret, men ikke på første forsøk.

Elevgruppe 4

4A, 4B, 4C og 4D

Dette er tre elever som er veldig pratsomme og faglig sterke, som liker å kunne arbeide i grupper, som ønsker å lære. De viser tegn til å være både positive og negative til samarbeid. De ser både fordeler og ulemper. Dette er tre sterke personligheter somhar satt seg sammen, og det kan tenkes at det av den grunn kunne ha oppstått konflikter.

Elevgruppe 5

5A og 5B

Dette er to jenter som fort kan gå lei av ensformige oppgaver. De ønsker å gjøre rett, og er kanskje av den grunn ikke like utforskende som de andre gruppene. De vil heller holde seg til det sikre enn det som er nytt. De liker det trygge, sikre.

Elevgruppe 6

6A og 6B

Dette er to sterke elever som til vanlig ofte er muntlig aktive, som føler de har kontroll. Som gjerne tar med seg ting de liker hjem og arbeider videre med det der. Det er to sterke personligheter som begge står på sitt, så at disse ser ulemper ved samarbeid er inget sjokk.