UiT
NORGES
ARKTISKE
UNIVERSITET

Institutt for lærerutdanning og pedagogikk

Programmering + matematikk = sant?

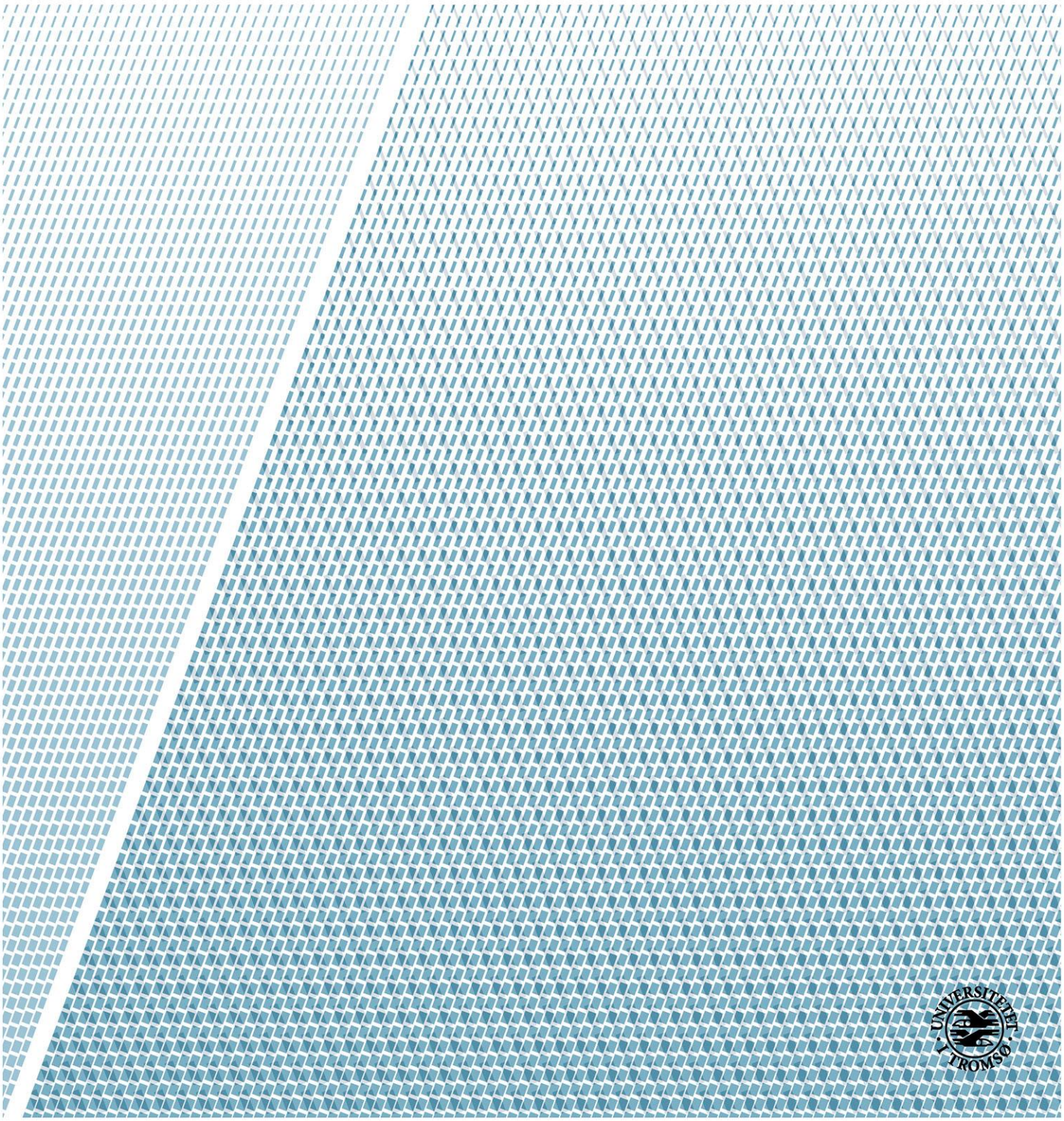
En casestudie om overføringsverdien mellom programmering valgfag og matematikkfaget

—

Andrej Verstad

Masteroppgave i lærerutdanning 5.-10. trinn, Mai 2017

LRU-3903 Matematikkdidaktikk



Sammendrag

Programmering + matematikk = sant? Er en casestudie som ser på overføringsverdien mellom programmering valgfag og matematikkfaget. Programmering valgfag innføres som en prøveordning i ungdomsskolen, der det hevdes flere tilknytninger til matematikkfaget.

Det teoretiske rammeverket for studien tar utgangspunkt i teorien rundt situert kognisjon, samt det Hierbert og Lefevre (1986) beskriver som konseptuell og prosedyrebasert kunnskap. Dette for å se hvordan forbindelser kan skapes mellom matematikkfaget og programmeringsfaget. Begrepet algoritme trekkes frem, både i matematikken og i programmeringsfaget, som er sentralt i studien. Ved at elevene programmerer kommer begrepet algoritmisk tankegang frem i oppgaven, der problemløsning gjør seg sentralt. I denne sammenheng har jeg valgt å trekke inn Polyas (1971) problemløsningsprosess, samt problemløsning og modellering fra Lesh og Zawojewski (2007). Prosjektet baserer seg på programmeringsspråket Scratch.

For å kunne svare på i hvilken grad det eksisterer en overføringsverdi, ble datainnsamlingen delt opp ut fra rammeverket intended, enacted og attained fra til Stein et al (2007). Jeg benyttet meg av observasjon og intervju som metode for datainnsamling. Studien ble gjennomført på en programmeringsklasse, der det ble trukket ut 12 informanter, over en varighet på fire måneder.

I datamaterialet kommer det frem en overføringsverdi til matematikkfaget, gjennom problemløsingsprosessen som tar sted i rundt det å lage programmeringskode. Denne prosessen kalles algoritmisk tankegang. På bakgrunn av denne prosessen gjør flere av elevene viktige forbindelser mellom matematikkfaget og programmeringsfaget, både med tanke på innhold or arbeidsmåter. Videre kommer det frem at samarbeid gjør seg gjeldene i valgfaget, noe flere av elevene fremhever verdien av.

Forord

Denne masteroppgaven er et produkt av de fem årene jeg har studert ved UiT Norges arktiske universitet. Jeg setter pris på å ha fått arbeide med et lengre og omfattende prosjekt som har tatt for seg et tema jeg har stor interesse for. Ved å begynne et halvt år før har jeg fått sjansen til å bli bedre kjent med programmeringsfaget, min rolle som forsker, og ikke minst meg selv, der jeg sitter igjen med mange gode erfaringer som jeg kan ta med meg videre inn i læreryrket.

Jeg vil rette en stor takk til begge lærere i programmeringsvalgfaget som ga meg muligheten til å observere og intervjue klassen og ikke minst elevene jeg fikk lov til å forske på. Prosjektet hadde ikke vært mulig å gjennomføre uten dere.

Jeg vil utrette en stor takk til mine veiledere Geir Olaf Pettersen og Ove Gunnar Drageset, som har i løpet av det siste året gitt meg alt jeg har trengt for å kunne gjennomføre prosjektet. Gode diskusjoner og meget gode tilbakemeldinger har hjulpet meg med å se oppgaven i flere perspektiv.

Videre vil jeg takke min mor og hennes samboer for støtten og hjelpen med korrekturlesing, og ikke minst min samboer som har vært der i både oppturer og nedturer i prosjektet. Jeg vil også takke venner, familie og medstudenter som har støttete meg under prosjektet.

Innholdsfortegnelse

1 Innledning

I en pressemelding ifra 2012 påpeker nåværende kunnskapsminister Torbjørn Røe Isaksen et realfagsproblem i den norske skolen bl.a. med bakgrunn i pisaresultatene fra 2012 (Regjeringen.no, PISA 2012: Svakere resultater i matematikk og naturfag, 2013). I den samme undersøkelsen kom det frem at det hadde vært en nedgang i resultatene siden 2009 (Udir.no, Fortsatt en vei å gå, 2013), samt at Norge ligger under gjennomsnittet på verdensbasis, noe TIMMS undersøkelsen også bekrefter (TIMSS, 2001). På bakgrunn av dette har Norge satt inn mange ressurser i realfagene, i et håp om å løfte resultatene (Regjeringen.no, Satsing på realfag, 2013). En rapport fra 2011 viser dessuten at nordmenn sliter med matteangst, og påpeker i tillegg at holdninger til faget står i veien for læring (Regjeringen.no, Fra matteskrekk til mattemestring, 2013).

Resultatene fra disse undersøkelsene, og de politiske kreftene bak disse, danner kontekst til innføringen av prøvevalgfaget programmering valgfag, og samtidig bakgrunnen for argumentene til fordel for valgfaget. I forbindelse med innføringen av prøveordningen på 146 skoler, argumenterer Isaksen blant annet for at programmering vil gjøre realfag interessant og aktuelt. (Regjeringen.no, Koding blir valgfag på 146 skoler, 2016). Andre politiske krefter i programmeringsmiljøet argumenterer for at valgfaget vil «gjøre matte gøy» (Astad, 2013) med begrunnelse i flere av undersøkelsene som er nevnt i forrige avsnitt.

I denne sammenheng vil jeg trekke inn min motivasjon som forsker, da jeg vil forske på i hvilken grad det finnes en overføringsverdi mellom programmering valgfag og matematikkfaget. I forhold til overføringsverdien refererer jeg til hvordan innholdet og arbeidsmåtene i programmeringsfaget kan gi en overføringsverdi til matematikkfaget. På bakgrunn av dette har jeg stilt forskningsspørsmålet:

> *På hvilken måte vil innhold og arbeidsmåter i programmering valgfag kunne gi en overføringsverdi til matematikkfaget?*

Forskningsspørsmålet åpner for muligheten at det ikke vil eksistere en overføringsverdi, fordi man tar for gitt at det vil eksistere matematikk i programmeringsfaget. Samtidig eksisterer det en mulighet for en betydelig overføringsverdi. For å kunne svare på forskningsspørsmålet må det eksistere en mulighet for en overføringsverdi i planene og føringene for programmeringsvalgfaget. Dernest må jeg kunne svare på i hvilken grad en overføringsverdi

vil kunne eksistere i klasserommet, der undervisningen tar sted. Deretter må jeg kunne se i hvilken grad elevene som deltar i programmeringsfaget sitter igjen med en overføringsverdi. På bakgrunn av dette har jeg delt forskningsspørsmålet inn i tre underspørsmål:

> *Hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil vi kunne finne i planer og føringer til programmeringsvalgfaget?*
>
> *Hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil vi kunne finne i utøvelsen av programmeringsvalgfaget?*
>
> *Hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil elevene påpeke på bakgrunn av å delta i programmeringsvalgfaget?*

For å svare på det øvrige forskningsspørsmålet, vil jeg dele oppgaven inn i: teori, metode, analyse og diskusjon, oppsummerende diskusjon av tiltenkt, utført og oppnådd innhold samt innledning og avslutning. I kapittel 2.0 vil det presenteres teori som underbygger oppgaven, med teoretiske rammeverk som vil brukes i senere kapitler. Deretter svarer jeg på de metodiske valgene som ble tatt for å svare på forskningsspørsmålet i kapittel 3.0, metode. I kapittel 4.0, analyse og diskusjon, presenterer jeg min analyse av datamaterialet i oppgaven, med en påfølgende diskusjon. Dette følges opp med en oppsummering av analysen, der jeg vil svare på de tre underspørsmålene som blir stilt for å svare på det primære forskningsspørsmålet. Videre arbeid presenteres i kapittel 6.0, etterfulgt av litteraturliste og vedlegg.

2 Teori

I dette kapittelet skal jeg presentere teori som vil være grunnlaget for oppgaven. Jeg vil trekke inn ulik teori som er viktig for å kunne svare på forskningsspørsmålet, samt rammeverk som har betydning for andre deler av oppgaven. Avslutningsvis en presentasjon av programmeringsspråket Scratch, som er sentralt i oppgaven.

2.1 Situert læring:

Situert læring vektlegger at mye av det som læres er spesifikt for den situasjonen det læres i (Andreson, Lynne & Herbert, 1996, s. 5). Dette gjør teorien rundt situert læring meget relevant for å se på overføringsverdien mellom matematikkfaget og programmering valgfag. Det gjøres oppmerksom på at situert læring er en teori som kan overføres til mange arenaer innenfor utdanning, men at teorien i høy grad har påvirket forskning innenfor matematikkdidaktikk (Ibid). Videre understrekes det at forskningen har fokusert på sammenhenger mellom typiske skolesituasjoner, og «real world situations» (Ibid) der man bruker matematikk, samt forskjellen mellom det som læres i klasserommet og det som læres utenfor klasserommet (Ibid).

Hvis kunnskapen skal være situert, vil det bety at den vil være bundet i konteksten den oppstår i, og ikke kan overføres til andre kontekster. For at kunnskap skal overføres fra konteksten, så trekkes det frem flere faktorer som gjeldende. Representasjoner eller i hvilken grad man anvender kunnskapen påpekes som store påvirkningsfaktorer for overføring fra konteksten (Andreson et al, 1996, s. 8). Mengden av overføring avhenger av hvor oppmerksomheten gis under læringen eller i overføringen (Ibid). Videre trekkes abstraksjon frem som et viktig fenomen i situert læring, der det spesifiseres at abstrakt instruksjon kombinert med konkrete eksempler kan være viktig for overføring til andre kontekster, særlig når læringen må være anvendt på en rekke nye og uforutsette oppgaver i fremtiden. (Andreson et al, 1996, s. 9). Det fremheves at man heller bør få elevene engasjert og motivert i kognitive prosesser som kan overføres, og ikke fokusere på hvilke kognitive prosesser et problem eller en oppgave utløser. I sammenheng med situert læring fremheves læring som et sosialt fenomen der begrepet communities of practice trekkes frem som gjeldende. Begrepet referer til læringsmiljø der mennesker av lik status arbeider sammen for å forbedre deres tilegnelse av kunnskap (Ibid). Det gjøres oppmerksom på at det kan eksistere en mulighet for at elevene vil gjøre forskjellige valg i forhold til organisering og gjennomføring av arbeidet,

som å dele opp arbeidet, der noen vil gjøre alt, mens andre elever bare vil gjøre en del. På denne måten kan elevene få forskjellig utbytte.

2.2 Tiltenkt, utført og oppnådd innhold

For å forske på overføringsverdien mellom matematikk og programmering valgfag, vil jeg trekke inn teorien om intended, enacted og attained curriculum. Curriculum, heretter kalt innhold, defineres som hva som skal læres, og kan tenkes på som innholdet til det som skal læres. (Stein, Remilliard & Smith, 2007, s. 321).

Det påpekes en større forskjell i innhold fra det som står i førende dokumenter, og det som kommer frem i klasserommet (Ibid). I denne forbindelse, deles innholdet inn i tre nivåer, der det er sentralt at det forekommer tolkende og interaktive prosesser i og imellom nivåene (Ibid). Nivåene deles inn i intended, enacted og attained, som heretter oversettes til tiltenkt, utført og oppnådd innhold. Tiltenkt innhold kan deles inn i to deler, og tar for seg det formelle, som førende styringsdokumenter, men også hvordan læreren tolker disse dokumentene og transformerer innholdet for å fungere best i klasserommet (Ibid). Det utførte nivå tar for seg hvordan innholdet forekommer i klasserommet og den øvrige undervisningen som tar sted. På dette nivået vil lærere og eleven i samspill ta frem innholdet og lage noe eget som er forskjellig fra det som forekommer i dokumenter, og det læreren plana i undervisningen. Helt sist har vi det oppnådde nivå, som tar for seg det eleven sitter igjen med (Ibid). Det påpekes at aktiviteten i klasserommet i det utførte nivå har størst påvirkning på hva elevene sitter igjen med; på det oppnådde nivå (Ibid).

2.3 Konseptuell og prosedyrebasert kunnskap

I en større diskusjon rundt hva som er kunnskap i matematikk, trekker James Hiebert og Patricia Lefevre (1986) frem begrepene konseptuell og prosedyrebasert kunnskap (Hiebert & Lefevre, 1986, s. 1). Konseptuell kunnskap defineres som rik på forbindelser, og kan ses på som et nett av kunnskap med forbindelser til hverandre. Kunnskapen kan dermed ikke være isolert, men må være forbundet med andre deler av kunnskap (Hiebert & Lefevre, 1986, s. 4). Videre kan det bare være konseptuell kunnskap om den som holder kunnskap innser forholdet til andre deler av kunnskap (Ibid). Dermed skjer utviklingen av konseptuell kunnskap ved å konstruere relasjoner mellom deler av informasjon (Ibid). Konseptuell kunnskap kan enten være på et primært nivå, der kunnskap forbindes på samme nivå av abstraktet, eller på et refleksivt nivå der forbindelsene er mindre knyttet til spesifikke kontekster.

Prosedyrebasert kunnskap defineres som todelt, der første del tar for seg det formelle språket, eller symbol og representasjonsystemet av matematikken. Andre del tar for seg regler, algoritmer eller prosedyrer som brukes for å løse matematiske oppgaver (Hiebert & Lefevre, 1986, s. 6). Andre del er instruksjoner som beskriver hvordan man skal løse en oppgave stegvis (Ibid). Det fremheves at prosedyrebasert kunnskap struktureres i et hierarki, der noen prosedyrer fungerer som underprosedyrer av andre (Ibid). To typer prosedyrer trekkes frem, på bakgrunn av objektene de opererer på. Det kan være skrevne symboler eller ikke-symbolske objekter. Det fremheves at symbolske objekter ofte dominerer utover skoleløpet i grunnskolen (Ibid). I forbindelse med ikke-symbolske objekter fremheves prosedyre som en problemløsningsstrategi, som opererer på konkrete objekter, mentale bilder eller visuelle diagram (Ibid). Noen prosedyrer manipulerer matematiske symboler, der andre opererer på konkrete objekter slik som mentale bilder eller visuelle diagrammer (Ibid).

Det påpekes at matematisk kunnskap i sin helhet inkluderer et forhold mellom konseptuell og prosedyrebasert kunnskap (Hiebert & Lefevre, 1986, s. 9). I dette forholdet påpekes det flere fordeler, der:

«*Linking conceptual knowledge with rules, algorithms, or procedures reduces the number of procedures that must be learned and increases the likelihood that an appropriate procedure will be recalled and used effectively*» (Hiebert & Lefevre, 1986, s. 14).

Når prosedyrer blir satt sammen med konseptuell kunnskap blir de lagret som en del av et nettverk av kunnskap, der disse forbindelsene gjør det mer sannsynlig at prosedyrer hentes fram når det trengs. I denne sammenheng kan det forbedre problemrepresentasjoner og simplifisere prosedurale krav (Ibid), der det presiseres at problemløsning blir løst ved at man bygger mentale representasjoner av problemene, og deretter tar hånd om representasjonene ved å velge ut den passende prosedyren. På denne måten kan konseptuell kunnskap gjøre et vanskelig problem lettere, som kan løses med en tilgjengelig prosedyre (Davis, 1984) i (Hiebert & Lefevre, 1986, s. 12). Representasjoner som er rike, fasiliterer en god fremgangsmåte, mens problemer som mangler konseptuell representasjon blir løst ved å velge ut memoriserte prosedyrer (Ibid). I denne sammenheng påpekes det at man kan bruke prosedyrer på flere problemer, hvis de er forstått og lært på en meningsfull måte (Hiebert & Lefevre, 1986, s. 13). Sagt på en annen måte, så tar konseptuell kunnskap bort prosedyren fra overflaten i konteksten der den ble lært, og oppmuntrer til å brukes på andre lignende situasjoner (Hiebert & Lefevre, 1986, s. 14).

Koblingen mellom prosedyrebasert og konseptuell kunnskap fremheves videre med at man kan få mer utøvende kontroll over prosedyrer, der man kan evaluere nytten eller unytten av prosedyrer, velge bort uakseptable prosedyrer, samt kunne bedømme utfallet av en prosedyre (Hiebert & Lefevre, 1986, s. 12). Prosedyrer kan også brukes til å løse konsepter, ved at problemer blir løst gjentakende, der den konseptuelle kunnskapen omgjøres til prosedural kunnskap (Hiebert & Lefevre, 1986, s. 15). Prosedyrer promoterer også konsepter, der det påpekes at elevenes konseptuelle utvikling kan bli motivert av deres prosedyrer (Hiebert & Lefevre, 1986, s. 16). Prosedural kunnskap gir et formelt språk, samt handlingssekvenser som kan forhøye nivået og anvendbarheten av konseptuell kunnskap (Ibid).

Det nevnes flere faktorer som går imot konstruksjon av forbindelser mellom kunnskap. Sentralt er mangler i kunnskapsbasen, der forbindelser ikke kan bli konstruert om ikke kunnskapen eksisterer. Mangler i konsepter eller prosedyrer kan være kilden til manglende eller ikke eksisterende forbindelser (Hiebert & Lefevre, 1986, s. 17). Videre påpekes en tendens der elever overser forbindelser mellom kunnskap, som ofte blir tatt for gitt av læreren som prøver å lære det bort (Hiebert & Lefevre, 1986, s. 18). Det påpekes også at kunnskap ofte deles opp i seksjoner, der det som læres i en spesiell kontekst blir knyttet til overflatekarakteristikker av konteksten, noe som ofte forhindrer det å se sammenheng mellom nylig konstruert kunnskap og tidligere konstruert kunnskap (Ibid). Denne kontekstbaserte kunnskapen ser ikke etter forbindelser som er utenfor den umiddelbare konteksten den ble tillært i. Kunnskap som er delt inn i seksjoner kan ofte forbli isolerte, selv om de er godt utviklet i de forskjellige seksjonene. I denne sammenheng påpekes det at man ofte motstår konseptuell forandring, og at det dermed lagres i andre avdelinger som ikke forstyrrer eksisterende konsepter (Ibid).

2.4 Overføringsverdi mot regning som grunnleggende ferdighet, eller mot matematikkfaget?

I forbindelse med spørsmål om i hvilken grad det vil være en overføringsverdi av matematikk i programmering valgfag vil det være viktig å skille mellom matematikkfaget og regning som grunnleggende ferdighet. Matematikkfaget i skolen skal medvirke til å utvikle den matematiske kompetanse den enkelte elev og samfunnet trenger, ved at elevene både arbeider teoretisk og praktisk (Utdanningsdirektoratet, 2013). Faget griper inn i viktige samfunnsområder, slik som teknologi og økonomi, der en solid kompetanse i matematikk er en forutsetning for utviklingen av samfunnet (Ibid). Den matematiske kompetansen innebærer

«... å bruke problemløysing og modellering til å analysere og omforme eit problem til matematisk form, løyse det og vurdere kor gyldig løysinga er» (Ibid).

Regning som grunnleggende ferdighet går på tvers av fag og innebærer å bruke matematikk på en rekke livsområder, der man bruker matematiske begreper, fakta, fremgangsmåter og verktøy, samt resonerer for å løse problemer og *«...for å beskrive, forklare og forutse hva som skjer»* (Utdanningsdirektoratet, 2013). Matematikksenteret.no påpeker at regning er *«å kunne anvende matematikk i ulike fag når det er relevant og på de ulike fagenes premisser»* (Matematikksenteret.no, 2014), noe som er avgjørende for læringen i de ulike fagene. Å regne består av fire ferdighetsområder: Gjenkjenne og beskrive, bruke og bearbeide, samt reflektere og vurdere, som er sentrale prosesser eleven må igjennom når man skal regne i de ulike fagene (Ibid). Kommunikasjon trekkes frem som det fjerde ferdighets området. Denne helhetlige prosessen kalles for modellering (Ibid). For å kunne utvikle regning som grunnleggende ferdighet, påpekes det at man bør bruke den i konkrete situasjoner, men også mer sammensatte situasjoner som er knyttet til ulike fagområder (Ibid).

Problemløsning kommer tydelig frem som en sentral faktor mellom begge begrepene. I matematikken presiseres det at den matematiske kompetansen innebærer problemløsning og modellering, som er sentralt for flere områder i samfunnet. Når matematikken anvendes i andre fag, på fagets premisser, vil det per definisjon beskrives som regning som grunnleggende ferdighet. Det vil være viktig å presisere at gjennom å benytte seg av regning som grunnleggende ferdighet, vil elevene få mer erfaring med problemløsning. Denne erfaringen, sammen med matematikken elevene bruker igjennom denne prosessen vil være sentral i overføringspotensialet mellom fag, og i vårt tilfelle mellom programmering valgfag og matematikkfaget. Jeg vil påpeke at matematikken vi vil finne i programmeringsfaget, på lik linje med matematikk i andre fag, vil per definisjon være regning som grunnleggende ferdighet. I denne sammenheng mener jeg det vil være viktig med en presisering mellom begrepene regning som grunnleggende ferdighet og matematikkfaget, med tanke på hvilken overføringsverdi begge begrepene tar for seg. Overføringsverdien for et fag vil være på fagets premisser når det gjelder regning som grunnleggende ferdighet. Når matematikken finner sted i faget, både i form av innhold og i form av arbeidsmåter ved for eksempel problemløsning, vil det kunne knyttes i høyere grad opp mot matematikkfaget. I denne sammenheng vil jeg trekke frem prosedyrebasert kunnskap og konseptuell kunnskap. Når elevene bruker symbol- og representasjonssystemet i matematikken, eller regler, algoritmer eller prosedyrer som man

bruker for å løse matematiske oppgaver, vil det i høyere grad knyttes opp mot regning som grunnleggende ferdighet. Dette fordi at overføringsverdien går mot faget. Når elevene benytter seg av matematikk i andre fag, og det skapes forbindelser til konseptuell kunnskap, så vil overføringsverdien i høyere grad være mot matematikkfaget, fremfor regning som grunnleggende ferdighet, siden det skapes en overføringsverdi i den konseptuelle kunnskapen.

2.5 Programmering og matematikk

Sammenhengen mellom programmering og matematikk kan sies å ha vært tilstede siden programmering ble oppfunnet av matematikeren Alan Touring (Ejsing-Duun & Misfeldt, 2015, s. 2524). Det har siden den gang kommet flere prosjekter inn for å forske på sammenhengen mellom matematikken og programmeringen. I nyere tid har programmering kommet inn i skoleverket i flere land i verden, med en påfølgende forskning rundt dette faget, spesielt i forbindelse med matematikk. I noen land, slik som Frankrike og Estland, ble programmering satt inn som en del av matematikkfaget (Ibid).

Det første seriøse forsøket på å koble matematikken med programmering i skolen kom da Symor Papert introduserte Microworld, et interaktivt univers elevene fikk tilgang til gjennom å bruke matematikk (Ejsing-Duun & Misfeldt, 2015, s. 2525). Microworld skulle inspirere til matematisk tenkning hos elevene gjennom små drypp av matematisk kunnskap som elevene kom i kontakt med når de utviklet prosjekter i Microworld. I denne sammenheng ble programmeringsspråket Logo utviklet, og det påpekes en bred entusiasme rundt både programmeringsspråket og Microworld (Ibid). Det fremheves at resultatene ikke svarte til forventningene av prosjektet, da det kom frem flere dårlige resultat i implementeringen av prosjektet. Flere av elevene overså de små dryppene av matematikk, noe som gjorde arbeidet i prosjektet ikke-matematisk (Ibid).

2.5.1 Tidligere forskning på programmering og matematikk:

Søken etter tidligere forskning på temaet programmering og matematikk, viste at det fantes få studier i Norge. Dermed måtte jeg gå til internasjonalt forskning. I denne sammenheng har jeg trukket ut fem studier jeg mener er relevante for oppgaven. Tre av artiklene tar for seg forskning på bruken av programmering for å lære matematikk, hvorimot to av artiklene fokuserer på hvilken rolle matematikken har i programmeringsfaget.

Første studie tar for seg det å lage spill ved å bruke programmeringsspråket Scratch, der det påpekes at hvis elevene får lage spill fremfor å spille spill, vil elevenes utbytte være større.

(Koptelov & Taube, 2015, s. 88). Det vil gi elevene en mulighet til å forstå og lære mange utfordrende matematiske konsepter, slik som rasjonelle tall, funksjoner og ferdigheter, som kreves av dagens arbeidskraft (Ibid). Motivasjon påpekes som en stor faktor i denne sammenheng, der det å arbeide med spill vil være hensiktsmessig for å lære seg matematikk. Flere tema trekkes frem i matematikken, slik som problemløsning, funksjoner, rasjonelle tall og proporsjonalitet. Disse kan oppfattes som irrelevante å lære seg, samt at elevene kan ha dårlige holdninger mot disse, som kan føre til at de ikke vil involvere seg i aktiviteter for å lære disse temaene. Ved bruk av kreativitet og tilgang til teknologi, kan elevene få muligheter til å lære seg disse temaene til tross for negative holdninger, som ved f.eks. å lage spill. (Koptelov & Taube, 2015, s. 90). Studien fant ut at elevenes motivasjon og interesse økte for matematikk ved å engasjere seg i utvikling av spill (Koptelov & Taube, 2015, s. 94).

Dette er noe den andre forskningsartikkelen påpeker, der det konkluderes med at ut fra observasjoner fra arbeid med visuell programmering gjennom prosjektarbeid, bidrar dette arbeidet til økt motivasjon, entusiasme og forpliktelse fra eleven. Forskningsstudien skulle evaluere hvordan det var å bruke programmeringsspråket Scratch som introduksjon til programmering, i en gruppe yngre elever i grunnskolen (Ibid). Ved at elevene lager sitt eget innhold som er relatert til læreplanen, så får man flere fordeler slik som: *«motivation, fun, commitment, and enthusiasm, showing improvements related to computational thinking and computational practices»* (Saez-Lopezs, Roman-Gonzalez & Vazquez-Cano, 2016, s. 129). Det trekkes også frem at elevene samarbeidet bra i prosjektet, samt fikk utviklet god kompetanse i kommunikasjon (Koptelov, 2015, s. 95).

Tredje studie påpeker at arbeid med Scratch-prosjekter fører til at elevene får økt problemløsningskompetanse. Hovedfunnet var at elevene som deltok i det Scratch-baserte prosjektet fikk høyere resultater enn elevene som ikke var med, på bakgrunn av at alle elevene gjennomførte en prøve i matematikk i etterkant av prosjektet (Brown et al, 2008, s. 12). En stor majoritet av elevene viste stor interesse, fortrolighet og tilgang til det å bruke Scratch, som viser til at Scratch som et verktøy, vil være en effektiv måte å praktisere matematisk argumentasjon som er nyttig for problemløsning (Brown et al, 2008, s. 10). I studien påpekes det en stor sammenheng mellom matematikken og dagliglivet, der elever med høy måloppnåelse i matematikk i høyere grad kan gjøre en kobling mellom matematikk og hverdagsaktiviteter. Dette fremheves som gunstig for at elevene skal kunne utvikle seg i

matematikkavhengige felt, sammen med at elevene burde mestre ferdigheter tidlig i skolegangen (T.N.A Sciences, 2002) i (Brown et al, 2008, s. 2).

I fjerde studie fokuseres det på førsteårselever på høyskolenivå og deres utfordringer med å lære seg grunnleggende programmering. Det påpekes flere utfordringer ved å lære elever programmering når man ikke har noe grunnlag fra før av (Pacheco et al, 2008, s. 1). Det ble gjennomført tre delstudier i løpet av et undervisningssår, hvor man undersøkte en større elevgruppe der alle hadde strøket i programmeringsfaget. Studiene tok for seg mulige relasjoner mellom manglende matematiske problemløsningsferdigheter og de manglende ferdighetene i programmering. Det poengteres en sammenheng mellom dårlige problemløsningsferdigheter, med en tydelig vektlegging på det som var nært matematikk, og dårlige ferdigheter i programmering (Ibid). Videre trekkes det frem at de fleste studentene som strøk i programmeringsfaget ikke kunne grunnleggende konsepter i matematikk, som igjen bør ses i sammenheng med deres dårlige prestasjoner i problemløsning og ferdigheter i programmering (Pacheco et al, 2008, s. 6). Det ble også påpekt en positiv korrelasjon mellom karakteren i calculus og resultatene i programmering (Ibid).

I femte studie fokuseres det på elevers mangler i matematisk kunnskap, og hvordan dette påvirker deres kapasitet i programmering. Her trekkes det frem at en majoritet av elevene ikke kunne grunnleggende konsepter i matematikken, som reflekteres i deres problemløsningsferdigheter og derfor i deres lave programmeringsferdigheter. Det trekkes videre frem en tydelig sammenheng mellom lave programmeringsferdigheter og lave ferdigheter i matematisk kompetanse, der sistnevnte ofte forårsaket førstnevnte (Gomes, Birgotte. Carmo & Mendes 2006, s. 5).

2.5.2 Matematiske tema i programmering:

På bakgrunn av tidligere forskning, samt grunnlaget for programmering, vil jeg trekke frem flere matematiske tema som er gjeldene i programmering. I flere programmeringsspråk vil man forholde seg til enten et todimensjonalt eller tredimensjonalt rom, der man vil bevege seg ved bruk av koordinater, grader og enheter i koordinatsystemet. Videre er begreper slikt som funksjoner og variabler sentrale i programmering. En variabel i matematikken kjennetegnes som det å utrykke likninger og uttrykk i mønster og generaliseringer (Van de Walle, Bay-Williams, Lovin & Karp, 2006, s. 232). Variabler kan brukes som mengder som varierer eller som ukjente verdier, der det ofte påpekes at elever forbinder variabler med sistnevnte og at

det kan være et vanskelig begrep å forstå (Ibid). Funksjoner forstås som situasjoner som samvirker, og trekkes også frem som et vanskelig begrep (Van de Walle, 2006, s. 242). Kontekster og det å bruke geometriske mønster påpekes som en viktig del av det å introdusere funksjoner. Å tenke på funksjoner i en input/output måte, fremheves som en god måte å vise meningen bak en funksjon (Ibid). Input/output funksjoner kan beskrives som en fysisk funksjonsmaskin der man setter inn en verdi og en annen verdi kommer ut, og der elevene må finne ut funksjonen av maskinen (Van de Walle et al, 2006, s. 245).

Generaliseringsaktiviteter trekkes frem som sentral for å forme uttrykk og likninger som er objekter av algebra (Kieran, 2007, s. 713). Likninger som innebærer en ukjent som representerer en ukjent, uttrykk av generalitet som oppstår fra geometriske mønster og numeriske sekvenser og uttrykk av reglene som styrer numeriske forhold påpekes som gode eksempler på generaliseringsaktiviteter i algebra (Ibid). Denne aktiviteten trekkes videre til å innebære arbeid med variabler, ukjente og likhet.

2.6 Problemløsning og modellering

Innenfor problemløsning i matematikk, vil jeg trekke frem George Polya (1971) som foreslår en firestegsprosess for å løse problemer. Innenfor disse fire stegene, understrekes det at man når som helst kan hoppe over stegene. Det å bare prøve ut en løsning, der man hopper over stegene, kan føre til mange feil og en lengre og dårligere problemløsningsprosess (Polya, 1971, s. 6). De fire stegene er å: Forstå problemet, lage en plan, utføre planen og se tilbake (Ibid). Det poengteres at man i første steg både må ha forståelse for problemet, men også vise en interesse for problemet. Eleven må forstå alle delene, hva som er konteksten og hva som er ukjent (Polya, 1971, s. 7). Man har en plan når man vet hvilke kalkulasjoner eller fremgangsmåter man må gjøre for å løse det som er ukjent. Veien mot å finne en løsning kan være lang og ofte fylt med mange feil (Polya, 1971, s. 9). Når man utfører planen, så vil man ha en skisse over løsningen, der man fyller inn detaljene, en prosess som vil kreve tålmodighet (Ibid). Ved å se over resultater og løsningen som ble utført, kan eleven verifisere resultatet, og konsolidere kunnskap (Polya, 1971, s. 15).

I forholdet mellom problemløsning og matematikk, påpeker Lesh og Zawojewski (2007) at mange elever ikke ser forholdet mellom matematikken som læres i skolen, og den som læres i såkalte problemløsningsmiljø, eller problemløsningskontekster (Lesh & Zawojewski, 2007, s. 781). I denne sammenheng påpekes det at man bør gå utenfor skolematematikken for å se på hvordan elevene bruker matematikk i problemløsning. Begrepsutvikling og

problemløsningsferdigheter trekkes frem som i høyere grad mer kontekstavhengig enn tidligere antatt, der det også påpekes at begrepene har en sentralt tilknytning til sosiale sammenhenger. Denne utviklingen vil ikke finne sted, uten en utvikling av følelser, oppfatninger, verdier, disposisjoner og andre komponenter som tilsammen skaper en «…*problemsolving persona*» (Lesh & Zawojewski, 2007, s. 779).

Det fremheves at det i yrker som baserer seg på utstrakt bruk av matematikk og problemløsning, vil mange ikke se sammenhengen mellom matematikken som forekommer gjennom problemløsning i yrkesutøvelsen og matematikken de bruker og lærer på skolebenken (Lesh & Zawojewski, 2007, s. 781). Ingeniører, teknikere og andre yrker fremheves som yrker der man arbeider med matematikk i problemløsningssituasjoner. I denne sammenheng mener Lesh og Zawojewski (2007) at man må formulere en ny definisjon av problemløsning, på bakgrunn av at man må identifisere og forstå forskjellen mellom matematikken i skolen, og den som skjer på arbeidsplassen (Ibid). Her trekkes det frem en ny definisjon av såkalte *Model Elicting Activeties*. Fremfor å først lære matematikk for deretter å gjennomføre problemløsning, så vil læringen av matematikk skje gjennom modelleringen som tar sted i problemløsningen (Lesh & Zawojewski, 2007, s. 783). Læringen starter ved å bygge konseptuelle systemer, ved å forstå ekte situasjoner der det er viktig å skape, revidere og tilpasse en matematisk måte å tenke på (Ibid).

I forbindelse med matematikk og problemløsning trekkes situert kognisjon og communities of practice frem som lovende forskningsområder. Situert kognisjon, også kalt situert læring, tar utgangspunkt i at læringen og problemløsningen tar sted i en kontekst. Den matematiske tenkningen skjer som en respons til problemer i konteksten, der matematikken skapes for å møte disse problemene i konteksten. Denne matematikken tenderer til å være meget forskjellig fra det som læres i skolen (Lesh & Zawojewski, 2007, s. 787). Det trekkes frem at det i store deler av forskning på situert kognisjon kommer frem at problemløsende aktiviteter i lokale kontekster utvikler matematiske redskaper og forståelse. I denne sammenheng påpekes det at aktørene som tar del i disse aktivitetene ikke ser at matematikken de bruker, utvikler og tilpasser, er relatert til matematikken de lærer i skolen. (Gainsburg, 2003b; Leave, 1998) i (Ibid). Videre så fremheves det at aktørenes syn på matematikk som brukes i virkeligheten, er meget forskjellig fra begrepene og prosedyrene som vektlegges i skolen (Lesh & Zawojewski, 2007, s. 787).

Det påpekes en kobling til situert kognisjon ved å trekke frem begrepet communities of practice, som tar for seg at kunnskap er sosialt situert (Lesh & Zawojewski, 2007, s. 789). Communities of practice tar utgangspunkt i områder og yrker der spesialister må arbeide sammen for å utvikle løsninger til komplekse problemer (Ibid). I denne sammenheng trekkes problemløseren frem som en gruppe, og ikke som en enkel aktør. I en skolesituasjon kan et problem være så krevende at elever må arbeide sammen for å løse det, og kan dermed ses på som et community of practice. Dette fenomenet kan finner sted i en hel klasse eller i en liten gruppe, med veiledning av en lærer eller mellom elevene. Mellom elevene kan det dannes strukturer og lederskap, der interaksjoner innad i gruppen gir muligheter for forståelse, fremfor at forståelsen kommer fra interaksjon med læreren. I en slik kontekst vil kunnskapen som oppstår bli sett på som utviklende, og lært/ikke lært (Lesh & Zawojewski, 2007, s. 790).

2.7 Algoritmisk tankegang:

Sentralt i forholdet mellom matematikk og programmering trekkes det å tenke i algoritmer og prosedyrer frem som et viktig mål i matematikken (Ejsing-Duun & Misfeldt, 2015, s. 2524). I denne sammenheng trekkes det frem et stort potensiale for å lære matematikk i programmering gjennom algoritmisk tankegang (Ibid).

Algoritmisk tankegang tar for seg elevenes ferdighet til å arbeide med algoritmer, der begrepet algoritmer forstås som systematiske beskrivelser av problemløsningsstrategier og årsak-virkning forhold (Ejsing-Duun & Misfeldt, 2015, s. 2526). En oppskrift trekkes frem som et godt eksempel på en algoritme (Ibid). Den algoritmiske tankegangen tar for seg det å utvikle, utføre, samt å få maskiner til å utføre slike algoritmer (Ibid). Videre trekkes det linjer til at en algoritme forbindes med en oppskrift, men også objektene som blir håndtert i oppskriften (Ibid). Når man vanligvis løser en problemløsningsoppgave, så gjøres dette stegvis, slik som en algoritme, som deretter løser problemet, men i programmering kan vi se på stegene som selve koden (Hazzan, Lapidot & Ragonis, 2014, s. 77). Den algoritmiske tankegangen vil være prosessen rundt det å lage programmeringskode.

På bakgrunn av en analyse av forholdet mellom matematisk tenkning og algoritmisk tankegang, trekker Donald Knuth (1985), ut følgende: Algoritmisk tankegang relateres til representering, redusering, abstrakt argumentering, informasjonsstrukturer og algoritmer, der matematikken kan være tilstede i alle disse områdene. Samtidig påpekes det at andre aspekter er tilstede ved manipulering av formler, opptreden av funksjoner og generalisering. På bakgrunn av dette påpekes det at algoritmisk tankegang har en sterk relasjon til matematisk

tenkning, men at det vektlegges andre aspekter enn det som gjøres i andre typer matematisk tenking (Ejsing-Duun & Misfeldt, 2015, s. 2524).. I denne sammenheng poengterer Knuth at matematisk tenkning ikke er et isolert konsept, men at man som matematiker bruker flere måter å tenke på. (Knuth, 1985, s. 180).

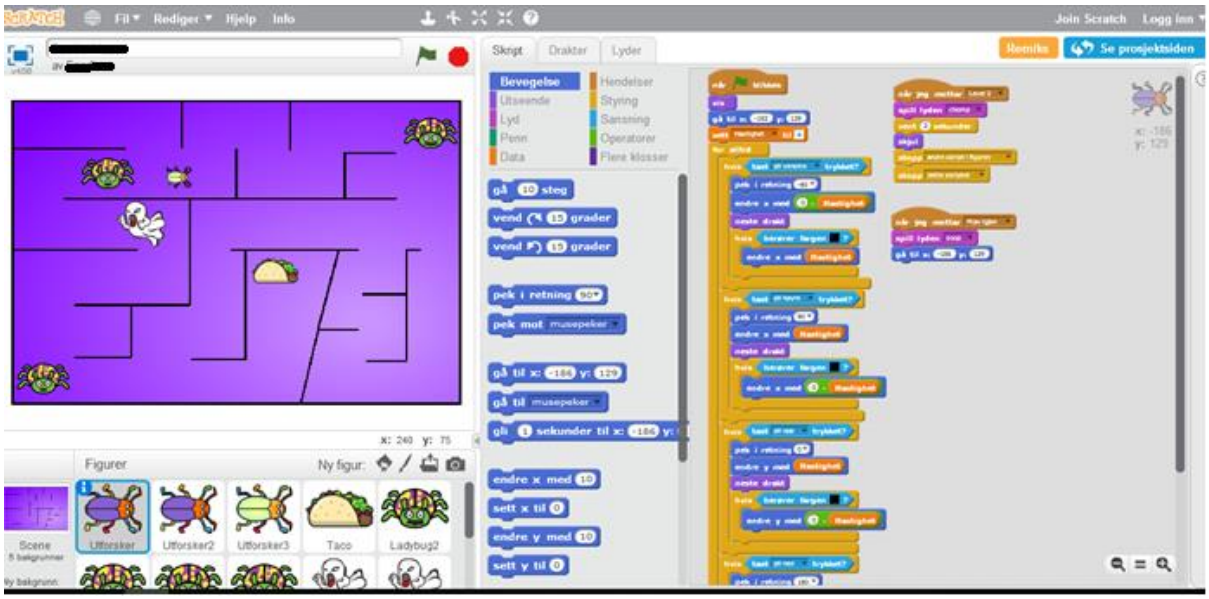
I forbindelse med algoritmisk tankegang vil det være viktig å trekke frem begrepet algoritme i matematikken, som definers som: *«An algoritm comprises a step-by-step set of instructions in logical order that enable a spesefic task to be accomplished»* (Thomas, 2014, s. 36). Ideen bak en algoritme er sterkt forbundet med det som defineres som prosedyrer i matematikken, siden prosedyrer kan gjennomføres ved å bruke algoritmer. Her er det viktig med et skille mellom algoritme og prosedyrer. For å kvalifiseres som en algoritme, må en prosedyre karakteriseres av nøyaktighet og generalitet. Med dette menes det at algoritmen alltid vil generere et rett svar, samt også alltid vil generaliseres til alle forekomster av problemet, eller klassen av problemer (Bass, 2013, s. 324). I matematikken påpekes det flere ulemper med algoritmer, der det er lettere å lære bort algoritmer enn det å lære begreper, eller konsepter. Videre vil det være mulig å utføre en algoritme, uten at man forstår hvordan den fungerer (Thomas, 2014, s. 37). En videre ulempe med en algoritme er steg-for-steg naturen, ved at den ikke gir noen avvik fra metoden. På denne måten kan ikke algoritmer oppfordre til allsidig tenkning (Ibid). I denne sammenheng vil jeg trekke inn prosedyrebasert kunnskap, som kan trekkes inn i arbeidet med det å følge en algoritme.

I sammenheng med algoritmer og programmering, trekkes begrepet algoritmics frem i matematikken, som defineres som design og analysen av algoritmer (Knuth 2000) i (Lagrange, 2014, s. 32). Dette området tar ikke for seg det å utøve algoritmen, men heller refleksjon av hvordan algoritmer er oppbygd og hvordan de fungerer. Typiske spørsmål på bakgrunn av refleksjon vil være effektiviteten, kompleksiteten og likeverdigheten til en algoritme.

2.8 Scratch:

Sentralt i forskningsprosjektet er programmeringsspråket Scratch. Dette er et gratis programmeringsspråk der man kan lage sine egne spill, animasjoner og interaktive fortellinger, samt lage andre programmer som er relatert til disse temaene (Saez-Lopezs, 2016, s. 130). Programmet baserer seg på blokk-koding, der det tilbyr mer enn 100 blokker fordelt over åtte kategorier (Ibid). Programmet er lettere å bruke enn tradisjonelle programmer siden elevene må bruke fargerike blokker for å lage script, der det påpekes at man må tenke

kreativt og samarbeide når man programmerer (Ibid). Programmet er laget for at barn og unge enkelt skal lære seg programmering ved å brukekodeklosser der programmeringskoden er ferdig skrevet (kidsakoder.no, 2016). Scratch har en lav begynnerterskel med vekt på en visuell innfallsvinkel, og kan sammenlignes med det å bygge legoklosser. For å lage programmeringskode, lager man en sammensetning av kodebrikker, et script, der det må være et samsvar mellom brikkene for at koden skal fungere.



Figur 1- Eksempel fra Scratch. Dette er et eksempel på en labyrint som elevene får tilgang til i prosjektet, som programmeringslærerne har laget.

Programmet tar utgangspunkt i at man beveger seg i et rom med bruk av koordinater og grader, der koordinatene kan skrus av og på. Det påpekes i denne sammenheng at man ofte ikke vil øve på en spesifikk matematisk aktivitet, men at man får prosessert den matematiske aktiviteten gjennom programmeringen i Scratch (Taylor, Harlow & Forret, 2010, s. 567).

Kodebrikkene man bruker for å programmere, er inndelt i forskjellige fargekoder etter hvilken funksjon de har. Med brikkene kan elevene lage løkker, et script som kan gjentas flere ganger. Videre kan man lage variabler, der man kan lagre informasjon man kan bruke i flere settinger, for eksempel ved å lagre en verdi som kan brukes i flere script. Man kan også lage et script til å fungere som en funksjon, der man fokuserer på forhold mellom elementer. Et eksempel på dette er at hvis a skjer, så skjer b.

Category	Notes		Category	Notes
Motion	Moves sprites and changes angles and change X and Y values		Events	Contains event handlers placed on the top of each group of blocks
Looks	Controls the visuals of the sprite; attach speech or thought bubble, change of background, enlarge or shrink, transparency, shade		Control	Conditional if-else statement, "forever", "repeat", and "stop"
Sound	Plays audio files and programmable sequences		Sensing	Sprites can interact with the surroundings the user has created and can import from PicoBoard or Lego WeDo
Pen	Draw on the portrait by controlling pen width, color, and shade. Allows for turtle graphics.		Operators	Mathematical operators, random number generator, and-or statement that compares sprite positions
Data	Variable and List usage and assignment		More Blocks	Custom procedures (blocks) and external devices control

Figur 2- Kategoriene for de forskjellige kodebrikkene, hentet fra Wikipedia.

3 Metode

Dette kapittelet vil inneholde de metodiske valgene jeg har tatt underveis i prosjektet, for å kunne svare på det primære forskningsspørsmål og de påfølgende underspørsmål.

3.1 Teoretiske perspektiv:

I mitt forskningsspørsmål spør jeg: *På hvilken måte vil innhold og arbeidsmåter i programmering valgfag kunne gi en overføringsverdi til matematikkfaget?* I denne sammenheng er det viktig å trekke frem Alan Schoenfeldt (2007) som påpeker at når man skal begrunne sine metoder, vil det være meget viktig å være bevisst over sine teoretiske antagelser, hva de vil representere i dataen og konklusjonene man tar ut fra de (Schoenfeld, 2007, s. 69). Her vil jeg poengtere at jeg ser etter overføringsverdien mellom programmering valgfag og matematikkfaget, der jeg tar utgangspunkt i hva eleven påpeker er matematikk i valgfaget. Her vil teorien rundt situert læring være sentral, hvorvidt det som læres er spesifikt for konteksten det læres i. (Andreson et al, 1996, s. 5). Begrepet situert læring stammer fra kognitiv psykologi, der flere av studiene rundt tema har utgangspunkt i dette feltet. Dette vil være i samråd med det teoretiske perspektivet i kognitiv psykologi, som tar for seg alt teoretisk grunnlag vedrørende elevers tolkninger og forståelse av verden, og at det tar utgangspunkt i at man kan trekke ut elevenes kognitive interne prosesser og strukturer (Cobb, 2007, s. 19).

3.2 Metodevalg og forskningsstil

For at min studie skal være relevant til det overordnede forskningsspørsmål, og de påfølgende underspørsmål, har jeg valgt å gjennomføre et kvalitativt studie. Dette begrunnes med at et kvalitativt studie vil ha en høyere grad av fleksibilitet fremfor et kvantitativt studie (Christoffersen & Johannesen , 2012, s. 17). Dette vil gi meg muligheten til å avdekke og følge opp fenomen som oppstår under forskningen. På denne måten kan jeg gå i dybden i datainnsamlingen, for å avdekke disse fenomenene.

Tidligere forskning rundt temaet programmering og matematikk tar for seg en større andel av casestudier. I denne sammenheng vil jeg trekke frem at programmering valgfag innføres som en prøveordning i den norske skolen. Dette vil si at prøveprosjektet gjennomføres for første gang, som en del av tre totale prøveår på 146 skoler i Norge, hvorav et flertall av skoler er i Tromsø. (Regjeringen.no, Koding blir valgfag på 146 skoler, 2016). Jeg mener det ligger et stort potensiale i å forske på denne prøveordningen, og da særlig på overføringsverdien til

matematikkfaget. Tar vi dette i betraktning, sammen med forskningsspørsmålene og det teoretiske perspektiv som ligger til grunn, vil jeg trekke frem casestudie som den mest hensiktsmessige forskingsstilen.

Et casestudie vil ta for seg et enkelt tilfelle, som vil illustrere et mer generelt prinsipp (Cohen et al, 2007, s. 253). I denne sammenhengen vil det enkle tilfellet være at programmering valgfag gjennomføres for første gang i den norske skolen. Det generelle prinsippet vil være overføringsverdien mellom programmering og matematikk. Man er som forsker en større del av casestudiet, der man kan fange unike egenskaper som ellers kan mistes i større datainnsamlinger. Disse unike egenskapene kan være nøkkelen til å forstå et viktig fenomen (Cohen et al, 2007, s. 256). Casestudier er i realiteten sterke og kan bygge på uforutsette situasjoner samt ukontrollerbare variabler som oppstår, noe som er sentralt siden prøvevalgfaget ikke hadde blitt gjennomført enda.

For å svare på forskningsspørsmålet i casestudiet valgte jeg å gå frem med rammeverket rundt «intended, enacted og attained» (Stein et al, 2007, s. 321) for å finne overføringsverdien mellom matematikk og programmering. I casestudiet deltar jeg i undervisningen, der jeg setter meg grundig inn i planer, føringer, oppgaver og styringsdokumenter, og dekker dermed tiltenkt innhold. Ved å være med i programmeringsvalgfaget vil jeg bruke observasjon for å dekke det utførte innhold. Her vil jeg kunne observere elevene i den konkrete undervisningssituasjonen. For å samle data innenfor oppnådd innhold, hvor jeg ser på hvilken overføringsverdi elevene sier de sitter igjen med, vil jeg bruke intervju som metode. Den viktigste egenskapen til et casestudie er at jeg som forsker kan fordype meg i feltet, og samle inn forskjellige typer data (Yin, 2014, s. 119). Ved å bruke flere metoder vil jeg dermed kunne svare på de forskjellige underspørsmålene. Videre vil det å bruke flere metoder gi mulighet til å kunne belyse data fra flere perspektiv.

Observasjon

Observasjon er en naturlig del av et casestudie hvor jeg som forsker skal følge undervisningen over lengre tid og dermed være tilstede i undervisningen, og kunne observere elevene i en naturlig setting (Cohen et al, 2007, s. 408). Et viktig valg i denne sammenheng vil være i hvilken grad jeg skal være deltagende i undervisningen. Det skilles hovedsakelig mellom det å være nøytral som observatør og det å være observatør ved å delta som deltager i situasjonen. Dette vil ikke bare ha mye å si for i hvilken grad man påvirker det som skjer i valgfaget, men også hvordan man oppfatter det som skjer i valgfaget (Cohen et al, 2007, s. 404). Min rolle i

klasserommet ble et sted mellom disse ytterpunktene, der jeg deltok i klasserommet og observerte i bakgrunnen, men ble i begynnelsen sett på som en av de voksne og ofte kalt for lærer. Dette er noe Cohen et al (2007) påpeker vil være en av fordelene med å være i en setting over lengre tid, der omgivelsene blir vant til din tilstedeværelse (Ibid).

Observasjon skilles videre mellom ytterpunktene strukturert og ustrukturert observasjon. Ved strukturert observasjon vil en se etter klare og tydelige fenomener og kategorier, mens en ved ustrukturert observasjon vil trekke ut kategorier fra observasjonene som finner sted (Cohen et al, 2007, s. 397). I dette prosjektet gjennomføres det semi-strukturert observasjon, som vil befinne seg mellom ytterpunktene, med klare tema for hva som observeres, samtidig som man kan ha rom for å tolke fenomen som oppstår i klasserommet.

Intervju

For å kunne samle data om hva som er det oppnådde innhold, gjennomføres det intervju i etterkant av observasjon. Jeg vil presisere at det også ble gjennomført et intervju i forkant av observasjonen, grunnet tidligere nevnt problemstilling og forskningsfokus, nemlig å kartlegge elevenes oppfatninger av matematikk. Hovedvekten av datamaterialet vil imidlertid komme fra etter-intervju, der jeg kartlegger hvilket innhold som er oppnådd.

Christoffersen og Johannesen (2012) skiller hovedsakelig mellom tre typer intervju: Strukturert, semi-strukturert og ustrukturert (Christoffersen & Johannesen, 2012, s. 79). I forbindelse med omfanget av masteroppgaven og min faglige bakgrunn, mener jeg at det ikke er passende med strukturerte intervju, eller ustrukturerte intervju. Et bedre alternativ vil være å benytte meg av semi-strukturerte intervju, der jeg tar utgangspunkt i en intervjuguide hvor tema, spørsmål og rekkefølge kan variere (Ibid). På denne måten kan jeg både ha en overordnet plan fra intervjuguiden, samtidig som jeg er fleksibel, og kan følge opp fenomen som framkommer i intervjusituasjonen.

Cohen (2007) påpeker at gruppeintervju kan være nyttig med barn, da det oppfordrer til interaksjon fremfor å bare respondere til den voksnes spørsmål. (Cohen et al, 2007, s. 374). Dette var hovedargumentet for å gjennomføre gruppeintervju, da gruppeintervju kunne føre til diskusjon mellom informantene med en større mulighet for å belyse viktige fenomener ved casen. Videre vil gruppeintervju føre til at jeg kunne intervjue et større utvalg av informanter.

3.3 Praktisk om datainnsamling:

3.3.1 Rammefaktorer og Utvalg

Ved at prøveprosjektet innføres som et eget valgfag i ungdomskolen, la dette naturlige føringer for hvor jeg ville finne informanter til å svare på forskningsspørsmålet. Andre føringer vil være rammene til et casestudie som innebærer det å se elevene i kontekst, at forskeren skal være en del av casen, samt kunne se helheten av casen. Det ble på bakgrunn av dette tatt en beslutning om å gjennomføre datainnsamling så tidlig som mulig, høsten 2016. På denne måten fikk jeg følge elevene over lengre tid, samt fikk en god ramme rundt det å gjennomføre observasjon og intervju. Dette passet godt med begrensningen på en master på 30 studiepoeng, der en lengre datainnsamling ikke ville være gjennomførbar våren 2017. Videre vil en tidligere datainnsamling gi tid til å fordype seg i feltet, samt kunne gi gode marginer for å tilpasse seg det som dukker opp underveis.

Mange av ungdomsskolene i Tromsø hadde fått innvilget prøveprosjektet i programmering, noe som ga gode muligheter for tilgang på programmeringsklasser og informanter. Ved å delta på et innføringskurs til fremtidige programmeringslærere, kom jeg i kontakt med en programmeringslærer som fungerte som en døråpner, og ga tilgang til programmeringsklassen vedkommende skulle undervise. Prosjektet viste seg å være for stort i forhold til oppstart og nylig innføring av det nye valgfaget. Dette førte til at jeg fikk valget om å kutte ned prosjektet drastisk eller avslutte samarbeidet. Jeg begynte å lete etter et nytt utvalg, der jeg fikk kontakt med en programmeringslærer i nettverket mitt, som var villig til å inngå et samarbeid. Etter kontakt med administrasjon ved skolen, ble det signert kontrakt med skoleeier og en av to programmeringslærere.

Programmeringsklassen hadde mange elever fra alle klassetrinn i valgfaget, med totalt 35 elever. Grunnet omfanget av prosjektet måtte det leveres ut informasjonsskjema og innhentes samtykke. Dette viste seg å være en utfordrende faktor da det var en dobbeltime i uken med valgfag, hvor det av den grunn var vanskelig å følge opp innsamlingen av skjema hos elevene. Godt samarbeid med programmeringslærerne gjorde at det ble innhentet nok skjema til å gjøre et tilstrekkelig utvalg. Grunnet disse faktorene kom ikke prosjektet i gang før tidlig oktober. For å garantere en tilstrekkelig datainnsamling over fire måneder, ble prosjektet dermed forsinket med en måned fram til tidlig februar.

Cohen et al (2007) påpeker at i en casesammenheng vil utvalget være desto mer representativ, jo større utvalget er, samt at det vil være mer sannsynlig at den som observerer vil bli en

naturlig del av konteksten (Cohen et al, 2007, s. 263). Grunnet størrelsen på programmeringsklassen, ble en ramme på 12-18 informanter satt. Dette begrunnes også ved at det gjennomføres gruppeintervju. På bakgrunn av at det bare kom inn 21 samtykkeskjema, ble utvalgsstrategien basert på frivillighet (Cohen et al, 2007, s. 116). Deretter ble et utvalg satt i samarbeid med programmeringslæreren, der totalt 17 informanter ble tatt ut i utvalget. Det vil være vanskelig å si hvilken strategisk utvalg som vil være optimal uten å ha møtt de potensielle informantene, men et godt utgangspunkt vil være et utvalg med maksimal variasjon (Christoffersen & Johannesen, 2012, s. 78), noe vi vektla da vi satte sammen gruppene ut fra mangfold i feltet. Jeg gikk ut ifra at feltet ville være homogent, og at det dermed ville være viktig å fange ytterpunktene i feltet. Faktorer som kjønn, klassetrinn og kompetanse var av betydning. At informantene arbeidet sammen i grupper ble også vektlagt i utvelgelsesprosessen. I intervjuet ble gruppene sammensatt for å fremme mest mulig mangfold og diskusjon i blant informantene. Fem informanter trakk seg fra prosjektet i datainnsamlingsperioden, dermed ble det i etterkant av andre intervju igjen 12 informanter.

3.3.2 Gjennomføring av observasjon:

Rammeverket TRU Math, med sine fem kategorier, ble brukt som utgangspunkt til observasjonsskjema. Dette begrunnes med at de fem kategoriene som rammeverket består av

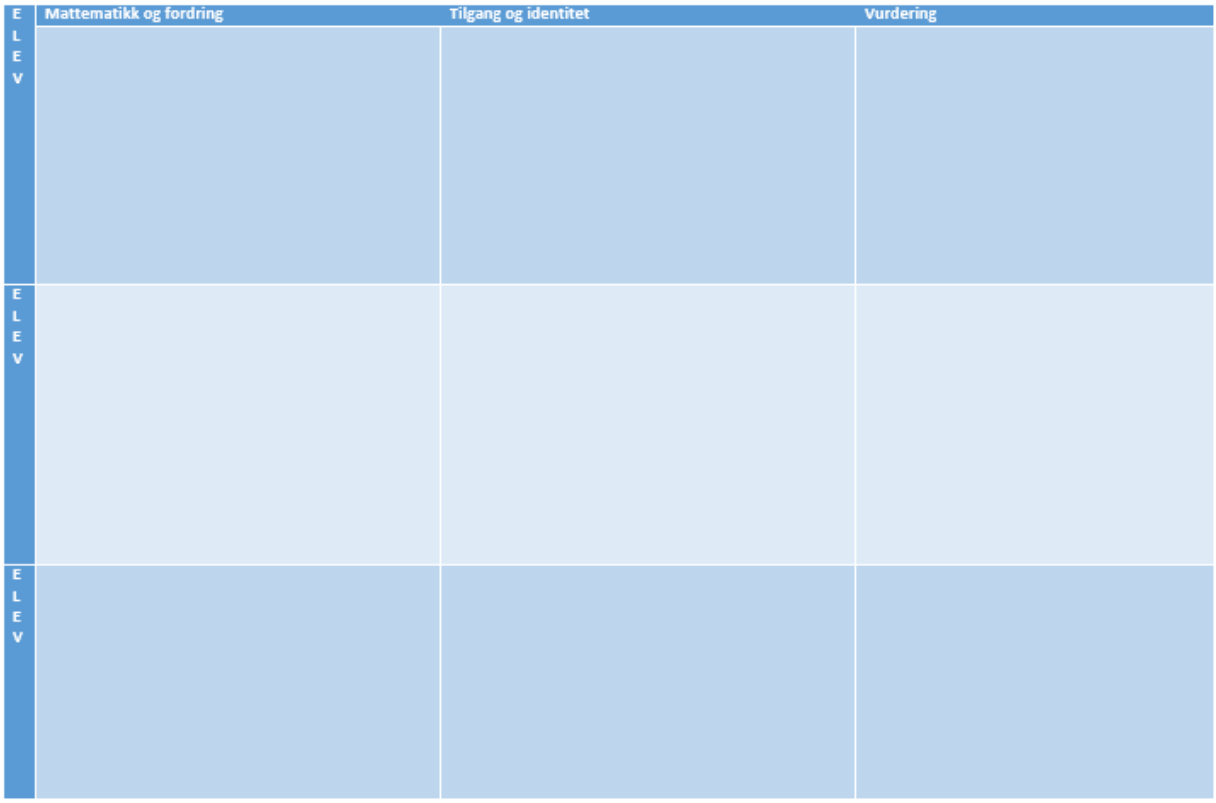
The Five Dimensions of Powerful Mathematics Classrooms

The Mathematics	Cognitive Demand	Equitable Access to Mathematics	Agency, Ownership, and Identity	Formative Assessment
The extent to which classroom activity structures provide opportunities for students to become knowledgeable, flexible, and resourceful mathematical thinkers. Discussions are focused and coherent, providing opportunities to learn mathematical ideas, techniques, and perspectives, make connections, and develop productive mathematical habits of mind.	*The extent to which students have opportunities to grapple with and make sense of important mathematical ideas and their use. Students learn best when they are challenged in ways that provide room and support for growth, with task difficulty ranging from moderate to demanding. The level of challenge should be conducive to what has been called "productive*	*The extent to which classroom activity structures invite and support the active engagement of all of the students in the classroom with the core mathematical content being addressed by the class. Classrooms in which a small number of students get most of the "air time" are not equitable, no matter how rich the content: all students need to be involved in meaningful ways.*	*The extent to which students are provided opportunities to "walk the walk and talk the talk" – to contribute to conversations about mathematical ideas, to build on others' ideas and have others build on theirs – in ways that contribute to their development of agency (the willingness to engage), their ownership over the content, and the development of positive identities as thinkers and learners.*	*The extent to which classroom activities elicit student thinking and subsequent interactions respond to those ideas, building on productive beginnings and addressing emerging misunderstandings. Powerful instruction "meets students where they are" and gives them opportunities to deepen their understandings.*

Figur 3-De fem dimensjonene til rammeverket TRU Math

er sentrale for i hvilken grad elevene vil gå ut av klasserommet som effektive matematiske tenkere og problemløsere (Shonfeld, 2016). Innenfor hver kategori fantes det tre forskjellige underkategorier som fungerte som ytterpunkt for god og dårlig oppnåelse innenfor hver kategori.

Rammeverket ble brukt som utgangspunkt for et semi-strukturert observasjonsskjema, der hovedvekten av skjema ble lagt på de to første kategoriene; matematikk og fordring. Dette begrunnes i at disse to kategoriene er essensielle for at elevene skal kunne bruke matematikk i programmering. Deretter brukes de andre kategoriene påfølgende. På denne måten kunne jeg gjennomføre observasjonen med klare rammer for hva jeg skulle se etter.



Figur 4-Observasjonskjema med utgangpunkt i TRU Math Dimensjonene

I forkant av observasjonen, som forberedelse, satte jeg meg grundig inn i styrende dokumenter og planer som la føringer for valgfaget, og kartla alle oppgavene som elevene skulle gjennomføre, for dermed å være forberedt for mulige situasjoner som kunne oppstå i klasserommet. Observasjonen som fant sted i klasserommet innebar å følge opp gruppene med informanter, der jeg noterte observasjonene fra gruppen i hver rubrikk. I avslutningen av observasjonen ble fire grupper med informanter trukket ut, for å få detaljerte data.

Hovedpoeng på Matematikken og fordring	Hovedpoeng tilgang, identitet og vurdering
Logg og journal: utfordringer, faktorer o.l.	Notater, refleksjoner og ideer

Figur 5-Baksiden av observasjonsskjema, basert på Spradley (1979) sine fire typer data.

I forbindelse med observasjon påpekes det flere utfordringer, der man kan risikere å få «observer bias», eller det å «go native» (Cohen et al, 2007, s. 404) I den sammenheng påpekes det av Spradley (1979) i (Cohen et al, 2007, s. 407) at man som observatør observerer fire typer data: Feltnotater som gjøres i felt, utvidede notater i etterkant, en journal der man kan samle refleksjoner, ideer og utfordringer, og avslutningsvis en pågående tentativ logg over endringer, analyse og tolkning. Ved å innføre en bakside i observasjonsskjema, ble disse kravene møtt, der jeg som observatør både underveis og i etterkant skrev grundige notater over det jeg hadde observert.

3.3.3 Gjennomføring av 1. intervju

Utformingen av intervjuguide begynte tidlig høsten 2016, siden jeg skulle gjennomføre første intervju tidlig i valgfaget. Målet med intervjuguiden tok utgangspunkt i et assortert og bredt forskningsfokus, der målet var å kartlegge elevenes oppfatninger av matematikk. I denne sammenheng tok jeg utgangspunkt i fire tema fra McLeod (1992) i (Kloostermann, Raymond & Emenaker, 1996) som vil ta for seg bredden til elevenes oppfatningene av matematikk. Det ble supplert med et nytt tema fra to andre forskningsstudier som jeg mente var hensiktsmessig. Temaene ble: Like matematikk og skolen, Nytte av matematikk, Selvtillit i matematikk, Matematikk alene eller sammen, og Matematikkhjerne. Til spørsmålene ble det laget en liste med mest sannsynlige oppfølgningsmål.

Det ble gjennomført fire gruppeintervju med fire informanter i tre av gruppene, og fem i den fjerde. Intervjuene ble gjennomført i undervisningstiden i valgfaget, grunnet at dette var mest hensiktsmessig for både elevene og programmeringslærerne. Det ble gjennomført to gruppeintervju på en dobbeltøkt, hvor det til sammen tok to uker for å bli ferdig med førintervjuet. Intervjuene varte litt over 20 minutter, som var i tråd med det som var planlagt. Det ble gjennomført prøveintervju i forkant for å forsikre meg om at det ble nok tid til gjennomføring. I intervjuet ble det brukt båndopptaker og feltnotater for bedre å kunne følge elevene, samt kunne sikre bedre data til analysen.

3.3.4 Gjennomføring av 2. intervju:

Ved utformingen av intervjuguide til 2. intervju satt jeg inne med gode erfaringer fra første intervju. Videre hadde jeg hatt muligheten til å tilpasse intervjuet til det jeg hadde observert i valgfaget. I forkant av 2. intervju hadde elevene grunnlag for å danne seg erfaringer med valgfaget, noe som også ble tatt høyde for i intervjuguiden. Intervjuet ble delt inn i tre faser: Tilbakeblikk og erfaringer, matematikk i programmering, og elevenes syn på matematikk og programmering. Tilbakeblikk og erfaringer fungerte godt som en overgangsfase, der elevene måtte fortelle hva de hadde gjort i valgfaget. Videre fikk elevene presentert elevoppgaven i Scratch-prosjektet. Andre fase tar for seg matematikken i programmeringsfaget med bakgrunn i det vi hadde gjennomgått i forrige tema. Siste fase inneholdt spørsmål om hvilket syn elevene hadde på matematikkfaget. Her ble det laget oppfølgingsspørsmål der deres syn på programmeringsfaget ble satt opp mot deres synet på matematikken. På denne måten kunne jeg få frem forskjeller og likheter som elevene måtte ta stilling til.

Det ble gjennomført intervju i samme struktur som i første intervju, med fire grupper, fordelt over to dobbeltøkter med valgfag. På grunn av praktiske faktorer, slik som sykdom og fravær, ble rekkefølgen av intervjuene forandret for å passe med utvalget. I kombinasjon med at flere elever trakk seg, var det igjen 12 elever i utvalget, med tre elever i hver gruppe. Det ble brukt båndopptaker og feltnotater i intervjuet.

3.4 Analysemetode

Ved at prosjektet hadde en datainnsamling over fire måneder, ble det lagt opp til en tentativ analyseprosess underveis. Dette innebar at jeg satte meg nøye inn i styringsdokumenter, planer og oppgavene elevene arbeidet med. Dette ble analysert fortløpende igjennom perioden, sammen med refleksjoner og observasjoner fra undervisningen. Flere av disse tentative analysene ble brukt til å bygge opp 2. intervju, samt å tilpasse observasjonsskjema.

Jeg vil påpeke at den tentative analysen ikke er grunnlaget for analysen for å svare på forskningsspørsmålet, men et tiltak for å styrke relabiliteten i forskningsprosjektet.

I etterkant av datainnsamlingsperioden satt jeg igjen med data fra både observasjonsskjema, før- og etterintervju, feltnotater fra intervju, samt førende dokumenter, elevoppgaver og styringsdokumenter fra valgfaget. I denne sammenheng startet jeg en analyseprosess der jeg tok utgangspunkt i den tematiske analysen. Tematisk analyse er en metode for å identifisere, analysere og reportere mønster i data, der tema representerer viktige fenomen i forhold til forskningsspørsmålet, og representerer i noen grad mønster eller mening utfra datasettet (Braun & Clarke, 2006, s. 6).

Stegene i den tematiske analysen ble fulgt, der første steg innebærer prosessen med å fordype seg i datamaterialet, og bli kjent med bredden og dybden av dataene. (Braun & Clarke, 2006, s. 15). Det påpekes at dette skjer i høy grad når man selv gjennomfører en transkripsjon av datamaterialet. Cohen et al (2007) påpeker at transkripsjon er et kritisk steg i intervjuet, der potensialet er stort for å miste data (Cohen et al, 2007, s. 365). Siden det å transkribere er å oversette et intervju fra et verbalt system til et skriftlig system, vil man kunne miste mye i prosessen. Det påpekes derfor at man som forsker bør være seg bevisst denne overgangen, og bør veie det opp mot nytten det vil gjøre i en analysesituasjon (Cohen et al, 2007, s. 367). Jeg valgte likevel å ta i bruk metoden da jeg lettere ville få oversikt over datamaterialet og dermed forenkle den videre analyseprosessen.

I etterkant av transkriberingen ble transkripsjonene gjennomgått grundig sammen med lydfilene for å forsikre meg at det ikke ble gjort feil. Deretter gjennomgikk jeg observasjonsskjema, dokumenter og øvrige feltnotater. I denne prosessen startet jeg å skrive ned notater for å bli kjent med dataene, og startet deretter med å kode dataene. I denne sammenheng skiller man mellom induktiv eller teoretisk tematisk analyse, som avhenger av i hvilken grad man vil knytte kodene og temaene opp mot dataene eller teori. Jeg valgte en mellomting, der jeg baserte meg på teori fra forskningsfeltet, samt tidligere forskning. På den måten kunne jeg være fleksibel og trekke ut viktige tema fra datamaterialet.

For å svare på det primære forskningsspørsmålet, ble analysen oppdelt etter rammeverket tiltenkt, utført og oppnådd innhold, for å svare på de tre underspørsmålene. Analysen i det tiltenkte innholdet ble gjennomført for å svare på *hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil vi kunne finne i planer og føringer til programmeringsvalgfaget?* Tre områder ble trukket frem i analysen: forsøkslæreplanen i

Programmering Valgfag, planer og føringer for Scratch-prosjektet og elevoppgaven i Scratch-prosjektet.

I analysen av det utførte nivå, ble det tatt utgangspunkt i å svare på *hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil vi kunne finne i utøvelsen av programmeringsvalgfaget?* Her tok jeg utgangspunkt i dataene fra observasjonsskjema. Yin (2014) påpeker at

"Any casestudy finding or conclusion is likely to be more convincing and accurate if it is based on several different sources of information, following a similar convergence" (Yin, 2014, s. 120).

For å styrke mine analytiske slutninger valgte jeg å trekke inn data fra etterintervjuet. Følgende tema ble trukket frem:

Varierte løsninger i Scratch-prosjektet:
Følger ikke elevoppgaven
Utvider og endrer på elevoppgaven
Følger ikke elevoppgaven

For å svare på *hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil elevene påpeke på bakgrunn av å delta i programmeringsvalgfaget?* så ble det gjennomført en analyse av det oppnådde nivå. Her tok jeg utgangspunkt i det elevene sier de sitter igjen med etter gjennomføring av valgfaget. Følgende tema fra etterintervju ble trukket frem:

- **Informantene kommenterer matematisk innhold i programmeringen**
- **Elevene kommenterer matematikken i programmeringen:**
- **Matematikk hjelper i programmeringen, matematikk i programmeringen hjelper ikke i matematikkfaget:**
- **Elevene kommenterer arbeidsmåtene:**
- **Elevene påpeker samarbeid som viktig i programmering:**

3.5 Relabilitet, validitet og metodekritikk

Gjennom mitt studie ble det gjennomført flere tiltak for å styrke relabiliteten og validiteten. Relabilitet kan defineres som hvor pålitelig dataene er, som knyttes til hvor nøyaktig undersøkelsens data er med. (Christoffersen & Johannesen, 2012, s. 23). Validitet kan defineres som hvor generaliserbare dataene er, og hvor relevante de er for undersøkelsen (Cohen et al, 2007, s. 134). Ved å ha en lengre datainnsamlingsperiode fikk jeg mulighet til å tilpasse meg det som oppsto i undervisningen, der jeg brukte baksiden av observasjonsskjema aktivt. Dette førte til at jeg endret forskningsspørsmål under prosjektet, og som en følge ble relabiliteten til første intervju lav. Mine data fra intervju baserer seg dermed på 2. intervju. Et videre tiltak var å basere intervjuguiden til 2. intervju på erfaringer fra 1. intervju, samt mine observasjoner. Det ble gjennomført prøveintervju på elever som ikke var med i utvalget, men som hadde levert samtykkeskjema. Videre har mine veiledere gitt meg tilbakemelding og veiledning i flere viktige stadier i studien, samt lest kritisk gjennom mine slutninger og data. Jeg gjennomførte transkripsjonene selv, der jeg før analysen hørte igjennom lydopptakene samt leste transkripsjonene, og slik forsikret meg om at det var riktig.

Min rolle som deltagende observatør gjør at observasjonene er påvirket av meg. Observasjonene vil påvirkes av hvordan jeg tolker situasjonene, hva jeg ser i situasjonene og hvordan jeg ser observasjonene. Jeg velger dermed å presentere mine data og analytiske slutninger så nøye som mulig, slik at leseren selv får gjøre sine egne tolkninger (Stake, 1995, s. 63). Et videre tiltak er å trekke inn data fra både intervju og observasjon for å styrke validiteten.

I intervjusituasjonen skiftet jeg miljø rundt elevene ved å ta ut elevene til grupperom. I denne sammenhengen påpeker Schoenfeldt (2007) at *«People will do things in some circumstances that they might do differently (or not at all) in other circumstances»* (Schoenfeld, 2007, s. 87). Endring av miljø vil være en påvirkende faktor. Dette mener jeg vil gjelde i høyere grad for 1. intervju, der elevene ikke var kjent med meg. I forkant av 2. intervju hadde jeg vært en del av valgfaget, der elevene hadde hatt muligheten til å bli trygge på meg. Dette begrunnes med at flere av elevene kalte meg for lærer. Miljøskiftet for 2. intervju vil dermed i høyere grad være materialistisk.

Jeg har gjennomført undersøkelsen på en liten gruppe elever. Dette er ikke representativt for en større del av befolkningen. I denne sammenheng vil jeg trekke fram en av ulempene til en casestudie der resultatene ikke kan være generaliserbare annet enn der andre lesere, eller

forskere ser deres applikasjon (Cohen et al, 2007, s. 256). Dette setter krav til hvordan jeg kan generaliserer mine funn, og i hvilken grad de er representative. I denne sammenheng påpekes det at et enkelttilfelle kan overføres til flere tilfeller som det representerer (Ibid). Casestudier kan gjøre teoretiske uttalelser, men det påpekes at disse må være støttet av dataen som presenteres i casen (Cohen et al, 2007, s. 254). Mine beskrivelser ligger til rette for at jeg kan sammenligne mine funn med andre caser og på denne måten dra nytte av mine funn og tolkninger.

3.6 Etiske betraktninger

Det vil være meget viktig at jeg kan gi noe tilbake til alle aktørene i prosjektet. I denne sammenheng påpeker Cohen et al (2007) at en av fordelene til en casestudie er at den er forståelig til de som leser den, og at den kan gi mye overføringsverdi til aktørene i casen (Cohen et al, 2007, s. 256). I tillegg er det viktig å påpeke at jeg som forsker indirekte kan være med å bevisstgjøre deltakerne om sentrale elementer, som kan påvirke feltet på en positiv måte.

I undersøkelsen har jeg tatt utgangspunkt i tre forskningsetiske retningslinjer, som er utarbeidet av den nasjonale forskningsetiske komité for samfunnsvitenskap og humaniora (Christoffersen & Johannesen, 2012, s. 41). Første retningslinje tar for seg at informanten har rett til selvbestemmelse og autonomi, som vil si at informanten har bestemmelse over egen deltakelse. Informanten skal få tilstrekkelig informasjon, gi frivillig samtykke for å kunne delta, og skal når som helst kunne trekke seg (Ibid). Den andre retningslinje går ut på å respektere informantens privatliv (Ibid). Dette betyr at informanten selv bestemmer hvilke opplysninger som skal brukes, og når som helst kan bestemme at opplysninger ikke skal brukes i oppgaven. Her vil det også være viktig å bevisstgjøre informanten om at all informasjon skal behandles konfidensielt og at opplysninger skal anonymiseres slik at de ikke kan spores tilbake til informanten. Den siste retningslinje er at jeg som forsker skal unngå skade.

For å imøtekomme disse retningslinjene, ble det gjennomført et grundig forarbeid. En sentral del av forarbeidet var informert samtykke, der informantene ble grundig informert om formålet med undersøkelsen, om sine rettigheter, hvem som har tilgang på empirien som skulle samles inn, hvordan empirien skulle behandles, samt annen generell informasjon om undersøkelsen. I denne sammenheng ble det gitt ut et informasjonsskriv i forkant av første intervju, der informanten måtte gi samtykke. Grunnet at elevene er umyndige, samt at data ble

samlet elektronisk, mente jeg at foresatte måtte gi samtykke for at elevene skulle være med i prosjektet. Jeg vil påpeke at lærere, samt skolen fikk utlevert informasjonsskriv i forkant, både formelt og uformelt i form av mail.

Empiri som ble samlet inn fra intervju og observasjoner vil være private data som kan identifisere deltakere i undersøkelsen. Derfor har det vært viktig med konfidensialitet, som innebærer at dataene ikke kan avsløre deltakerne. For å sikre konfidensialitet ble opplysninger som skole og navn, anonymisert. Informantene ble tildelt nummer, som kun har eksistert på en kodenøkkel som ble låst inn i et skap. Empirien fra båndopptakene ble innelåst ved masterkontoret, og skal destrueres med en gang masteroppgaven er gjennomført. Disse forhold ble informantene grundig informert om.

Innenfor personvernloven har man meldeplikt hvis opplysningene man samler inn er personopplysninger, eller at opplysningene er lagret elektronisk (Christoffersen & Johannesen, 2012, s. 43). Ved personopplysninger menes opplysningen som kan spores opp til informanten. Informasjonen jeg har innhentet har ikke involvert sensitive data, men siden jeg har brukt båndopptaker har jeg vært meldepliktig til Norsk Samfunnsvitenskapelig Datatjeneste AS (NSD) (Ibid).

4 Analyse og diskusjon

I dette kapitlet vil jeg presentere mine funn ut fra analysen som ble gjort av datamaterialet. For å svare på det primære forskningsspørsmålet og de påfølgende underspørsmål, gjennomførte jeg en analyse med utgangspunkt i rammeverket: tiltenkt, utført og oppnådd innhold. Jeg velger dermed å strukturere mitt analysekapittel etter dette rammeverket. I første delkapittel analyseres det tiltenkte innhold med utgangspunkt i: Forsøkslæreplanen i programmering valgfag, planer og føringer for Scratch-prosjektet og oppgave i Scratch-prosjektet. I påfølgende delkapitler vil jeg presentere situasjoner fra datamaterialet som gir et klart bilde over mine funn. I kapitlet som omhandler det utførte innhold, presenteres det funn med utgangspunkt i det som utspilte seg i klasserommet, der det kom frem at elevene brukte varierte løsninger i Scratch-prosjektet. Når det gjelder det oppnådde innhold presenteres det funn ut fra det elevene sier de sitter igjen med, der jeg trekker ut totalt fem sentrale tema.

4.1 Tiltenkt innhold:

4.1.1 Forsøkslæreplan i programmering valgfag:

Det naturlige første steg i analysen av det tiltenkte innholdet, vil være å se på de offisielle styringsdokumentene som legger de øvrige føringene for faget. Når jeg kom inn som forsker i faget var forsøkslæreplanen det eneste førende dokumentet som jeg fikk tilgang til. I en nøye analyse av hvordan matematikk forekommer i forsøkslæreplanen i Programmering Valgfag, ser vi allerede spor i formålet av valgfaget:

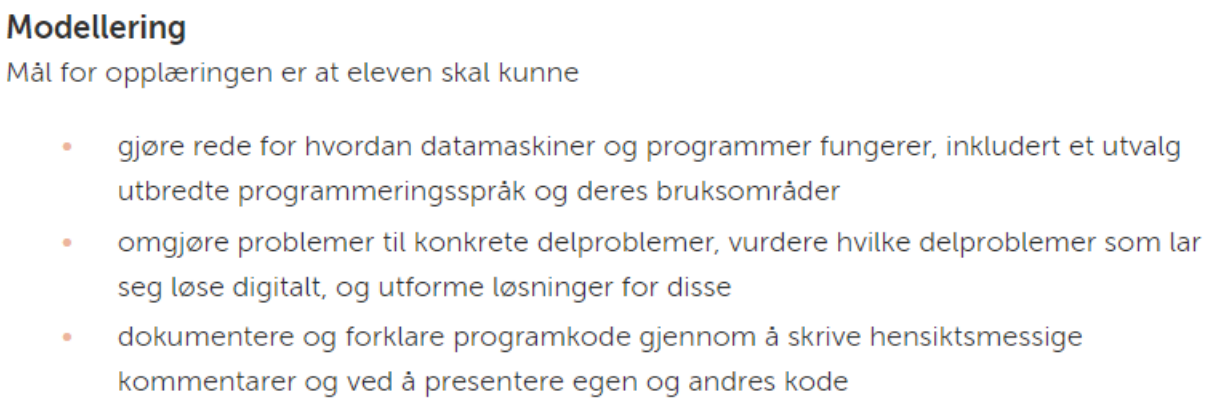
«Å gi elever grunnleggende ferdigheter i programmering er med på å forberede dem for fremtidige realfaglige jobber, i tillegg til å øke forståelsen for naturvitenskapelige og matematiske problemer.» (Udir.no, Forsøkslæreplan i koding valgfag, 2016).

Med formålet i Programmering Valgfag som bakteppe, vil jeg trekke inn formålet til matematikkfaget i skolen. I matematikkfaget påpekes det at faget griper inn i viktige samfunnsområder, slik som teknologi og økonomi, der en solid kompetanse i matematikk er en forutsetning for utviklingen av samfunnet (Utdanningsdirektoratet, 2013). En sammenstilling av formuleringen av formålene til begge fag gir rom for å hevde at programmering valgfag er en god arena, samt ligger innenfor et samfunnsområde i form av informasjonsteknologi, der man i aller høyeste grad får bruk for matematikk, både i nåtid og framtid.

Videre i forsøkslæreplanen, finner vi et begrep som defineres som algoritmisk tankegang (Ibid). Dette defineres som:

«*Valgfaget programmering handler om å lage programkode, det vil si et sett med regler og uttrykk for å styre digitale enheter. I dette inngår prosessen fra å identifisere problemer og utforme mulige løsninger, til å lage kode som kan forstås av en datamaskin, systematisk feilsøke og forbedre denne koden, og dokumentere løsningen på en forståelig måte. Det omfatter alle nivåer fra å forutse og analysere hva et program skal gjøre, til å kjenne igjen mønstre, eksperimentere og evaluere mulige løsninger, og samarbeide med andre. Summen av disse ferdighetene kalles algoritmisk tankegang.*» (Ibid).

Ut fra dette sitatet kan vi se et mønster i forsøkslæreplanen, der man strukturerer valgfaget i to hovedområder, henholdsvis modellering og koding (Ibid). Modellering som hovedområde defineres i forsøkslæreplanen som alle stegene man må ta for å løse problemer ved hjelp av programmering, noe som også kalles for algoritmisk tankegang i læreplanen.Her presiseres det at man må ha kunnskap om hvilke problemer som er egnet å løse ved hjelp av datamaskiner, hvordan disse kan deles inn i underproblemer, og deretter hvordan løsninger



Figur 6- Kompetansemål fra forsøkslæreplan i programmering valgfag

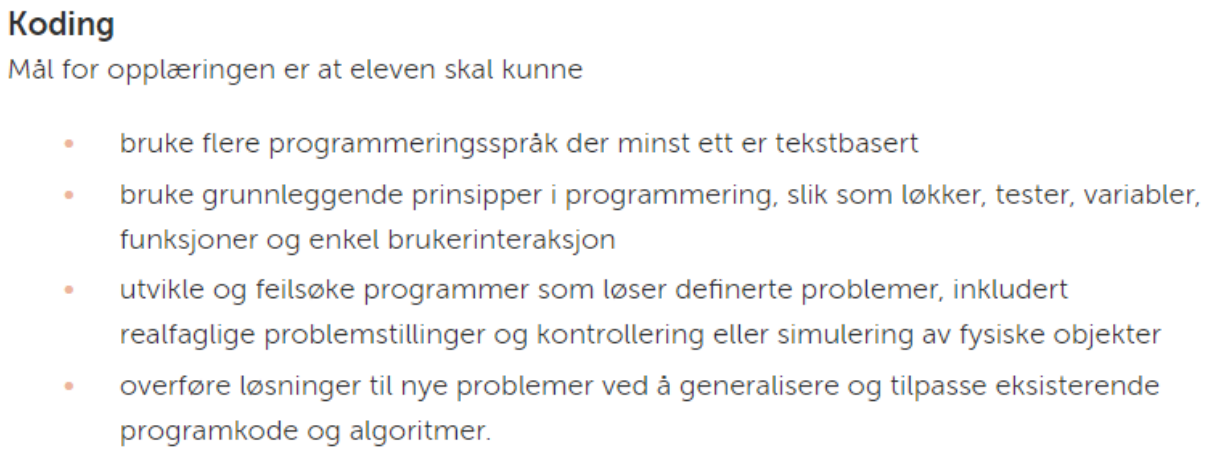
kan utformes. (Ibid). Følgende kompetansemål blir presentert i modellering som hovedområde i forsøkslæreplanen:

Følgende begrep gjør seg gjeldende i modelleringsdelen av læreplanen: den algoritmiske tankegangen, problemløsning og modellering samt begrepet algoritme. Algoritmisk tankegang i læreplanen fremheves som en større problemløsende prosess, der det endelige målet vil være å lage programkode i form av regler og uttrykk for å styre digitale enheter. Denne definisjonen er i overensstemmelse med tidligere definisjoner der algoritmisk tankegang beskrives som en systematisk beskrivelse av problemløsningen (Ejsing-Duun,

2015, s. 2526). Denne prosessen trekker frem modellering som hovedområde, der modellering av matematiske fenomener fremheves som en viktig del av området. I denne sammenheng vil jeg hevde at det vil eksistere en overføringsverdi mellom matematikkfaget og programmeringsvalgfaget. Lesh og Zawojewski (2007) trekker frem begrepet *Model Eliciting Activities*, som er en forståelse av at læringen av matematikk skjer gjennom modelleringen som finner sted i problemløsningen, i stedet for at man først lærer matematikken for så å utføre problemløsning (Lesh & Zawojewski, 2007, s. 783). Her kan elevene bygge opp konseptuelle systemer ved å forstå ekte situasjoner der det er viktig å skape, revidere og tilpasse en matematisk måte å tenke på (Ibid).

En videre overføringsverdi vil komme frem i forbindelse med begrepet algoritme. Igjennom problemløsningen som finner sted i den algoritmiske tankegangen, der elevene vil måtte forholde seg til matematiske fenomen, påpekes det at elevene arbeider med algoritmer som et viktig potensiale (Ejsing-Duun & Misfeldt, 2015, s. 2524). Dette vil kunne føre til at elevene vil måtte designe og analysere algoritmer, som igjen innebærer at man må reflektere over hvordan algoritmene er oppbygd, samt hvordan de fungerer (Lagrange, 2014, s. 32). Når elevene følger en oppskrift både i form av en algoritme og en prosedyre, påpeker Hiebert og Lefevre (1986) at elevene oppnår, samt benytter seg av, prosedyrebasert kunnskap (Hiebert & Lefevre, 1986, s. 14). Når elevene arbeider med innhold som kan relateres til matematikk, vil en overføringsverdi ikke bare eksistere i form av prosedyrebasert kunnskap, men også i form av at elevene vil måtte designe og analysere algoritmer, samt forholde seg til matematikk i problemløsningsprosessen. Det er her det vil kunne oppstå forbindelser til konseptuell kunnskap. Ut fra diskusjonen om hvorvidt det er en overføringsverdi til enten matematikkfaget eller regning som grunnleggende ferdighet, vil jeg hevde at det ved å skape forbindelser vil eksistere en overføringsverdi til matematikkfaget. Forbindelser vil også være av betydning for om kunnskapen er situert, og dermed i hvilken grad elevene vil se at det er matematikk de bruker (Andreson et al, 1996, s. 8).

Til forskjell for modellering, vil koding som hovedområde innbefatte det å utvikle programmer ved hjelp av ulike programmeringsspråk, samt forstå grunnleggende prinsipper innenfor det å programmere. Koding vil innebære simulering av fysiske objekter, slik som roboter eller baller i bevegelse. Det påpekes også at det dreier seg om beregninger og simuleringer som er basert på problemstillinger av en matematisk og naturfaglig art (Udir.no, Forsøkslæreplan i koding valgfag, 2016). Det finnes følgende kompetansemål finnes i hovedområdet koding:



Koding

Mål for opplæringen er at eleven skal kunne

- bruke flere programmeringsspråk der minst ett er tekstbasert
- bruke grunnleggende prinsipper i programmering, slik som løkker, tester, variabler, funksjoner og enkel brukerinteraksjon
- utvikle og feilsøke programmer som løser definerte problemer, inkludert realfaglige problemstillinger og kontrollering eller simulering av fysiske objekter
- overføre løsninger til nye problemer ved å generalisere og tilpasse eksisterende programkode og algoritmer.

Figur 7- Kompetansemål fra forsøkslæreplan i programmering valgfag7

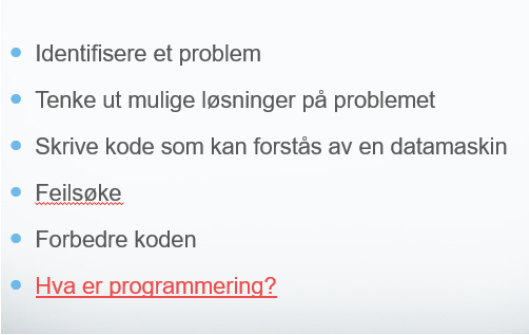
I koding som hovedområde vil jeg trekke inn Polyas (1971) problemløsningsprosess, der jeg vil hevde at Polyas siste steg kommer tydelig frem. Ved at elevene må overføre løsninger til nye problemer, samt utvikle og feilsøke programmer som løser definerte problemer, vil elevene benytte seg av steget: å se tilbake (Polya, 1971, s. 6). I forkant av dette har elevene laget script og algoritmer gjennom den algoritmiske tankegangen, der de forrige stegene i problemløsningsprosessen har skjedd. I sammenheng med å se tilbake eksisterer potensialet med å reflektere rundt algoritmer, ved at elevene må generalisere og tilpasse eksisterende algoritmer.

Ut fra kompetansemålene vil jeg videre trekke frem begrepene løkker, variabler, funksjoner og generalisering. I sin analyse av den algoritmiske tankegangen, påpeker Knuth (1985) aspektene ved manipulering av formler, oppførselen til funksjoner og generalisering. Funksjoner og hvordan man arbeider med funksjoner, vil være sentral i forholdet mellom matematikk og programmering. Funksjoner ble i matematikken definert som situasjoner som samvirker (Van de Walle et al, 2006, s. 242). Videre trekkes kontekster samt geometriske mønster inn som en viktig del av det å introdusere funksjoner. Variabler vil også være sentralt

og kjennetegnes ved at de uttrykker likninger og uttrykk i generaliseringer og mønster. I denne sammenheng vil jeg trekke inn at det vil eksistere et potensiale i arbeidet med generaliserende aktiviteter i algebra, der man kan forme uttrykk og likninger som er objekter av algebra (Kieran, 2007, s. 713). Denne aktiviteten innebærer videre arbeidet med variabler, ukjente og likhet.

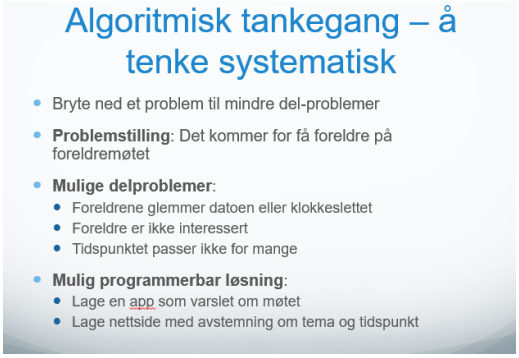
4.1.2 Planer og føringer for Scratch-prosjektet:

I løpet av min tid som forsker i programmeringsfaget, hadde elevene gjennomgått tre forskjellige tema i valgfaget. De første ukene hadde elevene gjennomgått et opplegg fra www.code.org for å lære enkle prinsipper i programmering. Deretter hadde elevene et lengre prosjekt i Scratch der de skulle lage et spill i form av en labyrint. I slutten av datainnsamlingsperioden hadde elevene tre uker introduksjon til micro:bit, en enhet som kan brukes til å simulere flere objekt som f.eks. en robot. I gjennomføringsperioden av intervjuene ble elevene delt opp i tre grupper der de fikk velge mellom micro:bit, Scratch eller å lære programmeringsspråket Python. Grunnet at jeg observerte og gjennomførte før- og etterintervjuet i gruppen som valgte å arbeide med Scratch har jeg vektlagt å behandle dette i forskningsstudien.



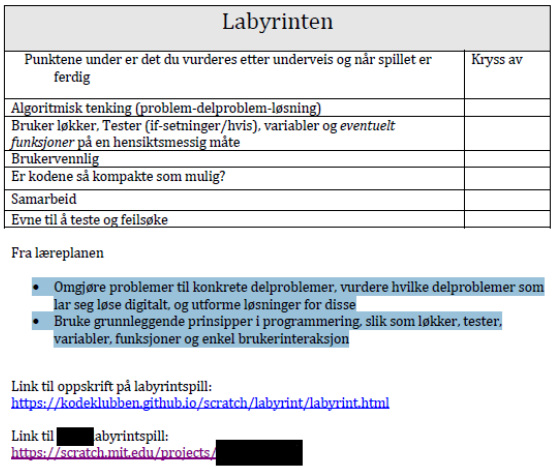
Figur 8- Første bilde i PowerPoint

Utgangspunkt for førende dokument ble en Powerpoint for Scratch prosjektet, siden det ikke var tilgang på noe annet. Problemløsning kommer frem som en sentral del av valgfaget, da dette er det første bildet eleven får presentert i forbindelse med Scratch-prosjektet. I dette mener jeg problemløsningsprosessen til Polya (1971) kommer frem, med stegene: Forstå problemet, lage en plan, utføre planen og se tilbake (Polya, 1971, s. 6). Dette gir et godt eksempel på prosessen rundt å programmere, der målet er å lage script. I denne sammenheng er algoritmisk tankegang sentralt, som kommer i neste bilde.



Figur 9- Andre bilde i PowerPoint

Den algoritmiske tankegangen blir beskrevet som det å tenke systematisk som en del av problemløsningsprosessen. Man skal dele et problem inn i flere delproblemer som elevene deretter må løse for å lage script. Elevene må i denne sammenhengen både komme med flere løsninger på problemer og hvordan man må gå frem for å programmere løsningen. Denne oppgaven elevene må gjøre, mener jeg gir et mer nyansert syn på begrepet algoritmisk tankegang som et resultat av en større problemløsningsprosess, ved å dele inn i delproblem for deretter å lage en stegvis oppskrift som datamarken skal forstå. Etter at flere delproblemer løses, kommer løsningen til problemstillingen frem ved at programmeringen fungerer. Deretter gjør man feilsøking, som vil innebære å endre og forbedre på script. Dette innbefatter å endre på algoritmen fra den algoritmiske tankegangen som vil si å endre på regler og uttrykk som får programmet til å fungere.



Labyrinten	
Punktene under er det du vurderes etter underveis og når spillet er ferdig	Kryss av
Algoritmisk tenking (problem-delproblem-løsning)	
Bruker løkker, Tester (if-setninger/hvis), variabler og *eventuelt funksjoner* på en hensiktsmessig måte	
Brukervennlig	
Er kodene så kompakte som mulig?	
Samarbeid	
Evne til å teste og feilsøke	

Fra læreplanen

- Omgjøre problemer til konkrete delproblemer, vurdere hvilke delproblemer som lar seg løse digitalt, og utforme løsninger for disse
- Bruke grunnleggende prinsipper i programmering, slik som løkker, tester, variabler, funksjoner og enkel brukerinteraksjon

Link til oppskrift på labyrintspill:
https://kodeklubben.github.io/scratch/labyrint/labyrint.html

Link til [redacted]abyrintspill:
https://scratch.mit.edu/projects/[redacted]

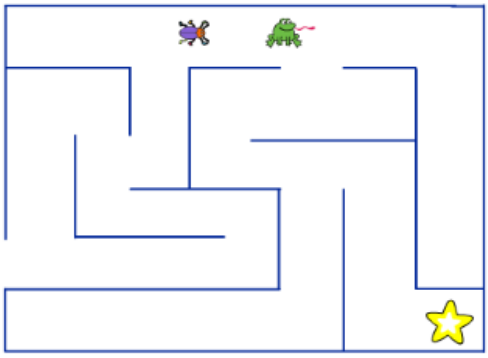
Figur 10- Vurderingskriterier for Scratch-prosjektet

I Scratch-prosjektet skulle elevene lage en labyrint med tre forskjellige nivå, dette sammen i par. I vurderingskriteriene til prosjektet ser vi at algoritmisk tankegang blir vektlagt som et selvstendig kriterium, der problemløsning blir vektlagt som en del av dette. Dette viser at man lærer å forstå at det inngår i en problemløsningsprosess i den algoritmiske tankegangen. Dette er i overenstemmelse med hvordan begrepet er forstått i tidligere forskning og i forsøkslæreplanen. Videre vil jeg poengtere at de andre vurderingskriteriene reflekteres i oppbyggingen av forsøkslæreplanen ved å reflektere det fjerde steget i Polyas problemløsningsprosess, som var tydelig tilstede i koding som hovedområde.

Elevene fikk mulighet til å følge en detaljert oppskrift for konstruksjonen av en labyrint i starten av prosjektet, som de fikk utlevert sammen med vurderingskriteriene. Elevene ble oppmuntret til å være kreative og bruke oppskriften som utgangspunkt for å bygge videre på.

Elevene ble også oppmuntret til hente inspirasjon fra andre labyrinter, der de kunne kopiere og videreutvikle programmeringskoder. Dette mener jeg beviser at det her vil ligge et potensiale i at elevene vil måtte designe og analysere algoritmer, som vil innebære et overføringspotensiale til matematikkfaget. Her oppfordres elevene til å endre og tilpasse på eksisterende script, og dermed også algoritmer. Her vil det være viktig med en presisering av algoritme i programmeringen, der algoritme forstås som en systematisk beskrivelse av problemløsningsprosessen der prosessen innebærer algoritmisk tankegang. Resultatet av prosessen vil være script, men et script vil ikke være en algoritme i seg selv. Algoritmen vil være funksjonen i de digitale enhetene som scriptet utfører.

4.1.3 Elevoppgave i Scratch-prosjektet:



Figur 11- Eksempel på et ferdig nivå ved å følge elevoppgaven

I forbindelse med Scratch-prosjektet får eleven utlevert en oppskrift. Oppskrift forstås forskjellig fra begrepene prosedyre og algoritme. Resultatet av å følge oppskriften i Scratch-prosjektet vil være et ferdig script, som vil innebære algoritmer. Oppskriften refereres til som elevoppgaven i denne oppgaven. Denne består av fem deler med en steg-for-steg beskrivelse for hver del. Resultatet av å følge elevoppgaven vil være et ferdig nivå av labyrinten som elevene skal lage, med en utforsker, skatt og fiende. I hvilken grad elevene tenker algoritmisk kan variere ut fra hvordan de velger å bruke elevoppgaven for å løse prosjektet. I denne sammenheng vil jeg påpeke at selv om elevene ikke vil arbeide med en spesifikk matematisk aktivitet, så vil elevene få prosessert den matematiske aktiviteten gjennom Scratch (Taylor et al, 2010, s. 567). Når vi dermed ser etter matematisk innhold i elevoppgaven, vil det være viktig å fremheve at elevene vil få prosessert den igjennom interaksjon med Scratch.

Første steg:

Figur 12- to muligheter for å algoritmen bak bevegelsen til utforskeren

I første steg av elevoppgaven i Scratch må elevene utvikle script som tar for seg bevegelsen til utforskeren i spillet. Utgangspunktet for bevegelsen kan sies å være sentrum av to akser, der utforskeren befinner seg i sentrum. Når en av piltastene trykkes, vil utforskeren bevege seg på aksen så

mange enheter som blir angitt. Videre vil rotasjonen ta utgangspunkt i sentrum av utforskeren. Her vil det ligge et potensiale for at elevene arbeider med rotasjon og enheter i et koordinatsystem.

Andre og tredje steg:

Figur 13- Kodesekvens fra steg tre

I tredje steg må elevene programmere at utforskeren ikke skal gå igjennom veggene til labyrinten som de har laget i steg to. Her vil elevene arbeide med en variabel for hastighet som de må sette inn i kodesekvensen. Videre må eleven bruke grader/rotasjon for å komme seg bort fra veggen. Her ligger det et potensiale for arbeid med matematikk i måten de bruker en variabel på, som kan overføres til hvordan man bruker en variabel i matematikken. *Fjerde steg:*

Figur 14-kodesekvens for fjerde steg

I steg fire må elevene lage en skatt som utforskeren skal ta. Når skatten blir tatt, starter spillet på nytt. Her vil elevene møte en utfordring. Når spillet starter på nytt igjen, vil utforskeren være på den posisjonen den var da den tok skatten. Elevene må dermed sette inn koordinater utforskeren skal starte på hver gang startflagget trykkes. Her må elevene forholde seg til koordinater i et større koordinatsystem ut ifra hvordan de har valgt å tegne labyrinten.

Femte steg:

Figur 15-kodesekvens for femte steg

I steg fem skal elevene lage en fiende. Her kommer det frem flere elementer av matematikk som elevene må forholde seg til. Elevene må sette inn koordinater for posisjonen der fienden skal settes inn. De må forholde seg til rotasjon og grader, ikke bare i hvilken retning man skal bevege seg, men også hvordan fienden skal bevege seg når den kommer i kontakt med labyrinten. Elevene må i tillegg forholde seg til variabler, men denne gangen er det to stykker i kodesekvensen. I siste del av sekvensen, hvis elevene velger å ha det med, så programmerer de at fienden skal bevege seg i tilfeldige retninger. Her programmerer de tre muligheter, enten

– 90 grader, 0 grader, eller 90 grader. Dette vil skje en av 25 ganger, som berører sannsynlighetsregning.

4.2 Utført innhold

I første undervisningssøkt gjennomgår programmeringslærerne rammene for prosjektet. Lærerne trekker frem algoritmisk tankegang og at problemløsning er en sentral del av programmeringen. De påpeker at dette blir vektlagt i vurderingen av prosjektet. Videre spesifiserer lærerne at elevene skal lage en labyrint med tre forskjellige nivå, der lærerne presiserer at elevene bør følge oppskriften for å lage første nivå. Dette kommer frem i de senere observasjoner av undervisningsøktene, der oppskriften blir fulgt av en stor del av elevene i starten av prosjektet. Programmeringslærerne fremhever videre at elevene skal være kreative og lage de to neste nivåene i labyrinten. Senere i presentasjonen av Scratch-prosjektet gjør programmeringslærerne elevene oppmerksomme på at Scratch inneholder matematikk i form av algebra, der de eksempelvis trekker frem et koordinatsystem og forklarer de forskjellige funksjonene dette kan ha. Videre settes det krav til samarbeid og selvstendighet, der elevene arbeider i grupper på to og to. Gruppene blir fulgt opp omtrent en gang for hver undervisningsøkt, der lærerne følger opp progresjonen i prosjektet.

4.2.1 Varierte løsninger i Scratch-prosjektet:

I gjennomføringen av Scratch-prosjektet kommer det frem at elevene har varierte løsninger, der det matematiske innholdet i høy grad reflekteres i løsningene til elevene. I alle observasjonene er måten elevene forholder seg til elevoppskriften sentral, der vi kan dele inn i det å følge oppskriften og det å utvide og endre på elevoppgaven.

4.2.1.1 Følger oppskriften i Scratch:

I observasjonene i avslutningen av Scratch-prosjektet kommer jeg i samtale med informant 6 og 7, der jeg ber de vise meg labyrinten sin. De forteller at de har fulgt elevoppgaven under hele valgfaget, og at de har møtt utfordringer. I første nivå fungerer labyrinten som den skal, men i de andre to nivåene, som har en helt annen oppbygning, blir elementene plassert på samme plass som i første nivå. Dette poengterer informantene er meget utfordrende, da flere av elementene henger seg fast i veggen. Dette løses ved å gå ut av spillet og dra elementene dit de vil de skal være. Dette kommer også frem i intervjuet.

> *E6:* Bare gjør det rett av bruksanvisningen, og det den sier, så klarer du det. Det var noe vi gjorde helt feil, som fikk den til å gå igjennom vegger og være helt fucked.

E6: Vår labyrint var helt fucka.

Andrej: Hva var det som var vanskelig med deres labyrint da?

E6: Det var det at den der skiten dreiv og gikk igjennom veggene.

Andrej: Den gikk igjennom veggene?

E7: Ja, vi klarte ikke å fikse det.

Senere i intervjuet, når vi prater om matematikken i programmeringen, kommer det frem at:

E6: Jeg er fortsatt drit i matte.

E7: Samme her.

E6: Hver mattetime så lærer jeg ikke en drit.

Diskusjon

Ut fra både observasjoner og intervju, kommer det frem at informantene følger elevoppgaven i Scratch under hele prosjektet. Dette reflekteres i resultatet, der labyrinten er velfungerende i første nivå, men at elevene møter utfordringer i de to neste nivåene. Utfordringene informant 6 og 7 kom i kontakt med kan sies å være knyttet opp til matematikk ved at elevene ikke forsto at de måtte endre koordinater for å sette inn elementene i et nytt nivå. Videre ser vi i intervjuet at elevene ser på seg selv som dårlige i matematikk.

Når elevene følger elevoppgaven gjennom hele prosjektet, vil de få et ferdig nivå. Når de derimot skal lage to nye nivå, vil de i noen grad måtte gå bort fra elevoppgaven og lage nytt script der de vil møte utfordringer. Den største utfordringen elevene måtte løse innebar matematikk der informantene måtte bruke koordinater. Dette klarte ikke informantene og jeg vil hevde at de ikke har nok kunnskap til å løse disse utfordringene.

Det eksisterer en mulighet for at elevene kan lage forbindelser mellom den prosedyrebaserte kunnskapen og konseptuell kunnskap, i form av at de blant annet klarer å koble koordinater i utfordringen. I denne sammenheng spesifiseres det i forhold til konstruksjonen av forbindelser at man kan ha mangler i kunnskapsbasen, og dermed ikke kunne konstruere forbindelser. (Hiebert & Lefevre, 1986, s. 17). Med utgangspunkt i hvordan informantene formidler det, vil jeg hevde at dette er meget sannsynlig. Dette stemmer med tidligere forskning der det fremheves at det er en tydelig sammenheng mellom lave programmeringsferdigheter og lave ferdigheter i matematisk kompetanse (Gomes et al, 2006, s. 6). Videre påpekes det en sammenheng mellom dårlige problemløsningsferdigheter i matematikk med dårlige ferdigheter i programmering (Pacheco et al, 2008, s. 6).

4.2.1.2 Utvider og endrer på elevoppgaven i Scratch-prosjektet:

Under observasjonen kom det tydelig fram at flere av informantene utvidet eller endret på elevoppgaven. Jeg har valgt å trekke frem to grupper som jeg mener representerer disse to løsningene.

Elevene utvider den eksisterende algoritmen

I observasjonene i avslutningen av Scratch-prosjektet prater jeg med informant 9 og 10, der jeg ber dem vise meg hva de mener har vært matematikk i valgfaget og om noe var utfordrende. Informantene viser til en utfordring med utforskeren på bakgrunn av at den henger seg opp i veggen i labyrinten de har laget. Gjennom feilsøking og mye arbeid forklarer informantene at de klarte å løse denne utfordringen med å endre på henholdsvis variabelen til hastigheten til utforskeren og den øvrige algoritmen til det å gå i veggen. Det poengteres at i forhold til det å gå inn i veggen, må være en større enhet ut fra veggen. Løsningen var å sette hastigheten til 0 når utforskeren møter veggen, deretter sette hastigheten til -2 enheter i forhold til den enheten hvor utforskeren beveget seg mot veggen, i stedet for å bruke grader og rotasjon til å snu[1]. Videre hevder begge informantene at de har vært innom matematikk i prosjektet, der de fremhever prosent, geometri, koordinater og regning med tall. Geometri har kommet frem ved å arbeide med grader da de referere til det å rotere i interaksjonen med objekter[2]. De påpeker videre at de har brukt betydelig med regning i flere av sekvensene der de har brukt tall. Deretter trekker informantene frem utfordringer med koordinater, der de påpeker at de brukte mye tid på å feilsøke og forstå utfordringen, og at de etter hvert klarte å løse utfordringen. Dette kommer godt frem i intervjuet:

> *E10:* Bare å finne ut riktig pluss og minus og x og y og sånn der, også måtte vi feilsøke ekstremt mye.»
>
> *E10:* Ehm, når det var hvor på koordinatsystemet vi skulle sette, og hvordan, og hvor startpunktet var, om når de kom til det, liksom for eksempel minus 35, 29, ja liksom startpunktet og så skulle komme seg til et annet, så var det vanskelig å få til at det skulle skje noe når du kom til liksom den.»

Elevene kommenterer fremgangsmåten senere i intervjuet, der:

> *E10:* Det var egentlig ikke basert på vanskelighetsgrad. Det var basert på hvor fort du ble irritert. Fordi det var litt sånn noen veier som man trodde man kunne gå, som man ikke kunne gå og masse forskjellige ting. Så det var litt sånn litt gøy.»

[1] Se steg 4 i elevoppgave
[2] Se steg 1 og 3 i elevoppgave

E9: Det var litt når man kom til et sted, så gjorde man det som var riktig for oss da, og da, også da fant vi ut at vi var på et annet sted i oppskriften, så den funket liksom ikke sånn helt optimalt for den enkelte personen vi lagde da.»

Diskusjon

Sentralt i observasjonene og intervjuet var hvordan informantene påpeker at de brukte matematikk. Elevene mener at de bruker regning, geometri og prosent som en del av det å programmere, men det jeg vil trekke frem som interessant er at matematikken kommer tydelig frem når de skal løse utfordringer i programmeringen. Når informantene går bort fra elevoppgaven, møter de utfordringer, og da særlig i form av matematikk. Ut fra det de beskriver i utfordringen med veggen, kan vi argumentere for at elevene arbeider med genererende aktiviteter, der de må løse en utfordring i forhold til bevegelse inn og ut fra veggen ved at de må endre på variablene hastighet[3]. Dette løser elevene med å endre på hele scriptet og funksjonen det representerer. Dette kan vi se i sammenheng med at Kieran (2007) påpeker at uttrykk ved generalitet som oppstår fra geometriske mønster og numeriske sekvenser samt uttrykk av reglene som styrer numeriske forhold, er et godt eksempel på generaliseringsaktiviteter i algebra (Kieran, 2007, s. 713).

Videre vil jeg fremheve at elevene arbeider problemløsende med matematikken som finner sted i oppgaven. Dette kan vi knytte opp til den algoritmiske tankegangen, som er prosessen rundt problemløsningen. Ut fra det informantene beskriver, måtte de se tilbake, finne en løsning, og deretter gjennomføre løsningen, som vil si at de følger stegene til Polya. Et godt eksempel på dette er hvordan elevene løste utfordringen med koordinater, der de påpekte at de måtte feilsøke for å løse utfordringen.

Her mener jeg vi er inne på et overføringspotensiale til matematikkfaget, der matematikken kommer frem i problemløsningsprosessen elevene må arbeide med, ved at de må løse problemer som har opprinnelse og tilhørighet i matematikken. Denne prosessen skjer som et resultat av den algoritmiske tankegangen, hvor målet er ferdig script.

Med de satte rammene for prosjektet, utfordres elevene til å lage to nye nivåer i spillet. Dette fører til at elevene må arbeide problemløsende. Elevene beskriver at de gjorde dette ved å følge elevoppgaven så lenge det fungerte for dem, men at de etter hvert gikk bort fra den.

[3] Se steg 3 i elevoppgaven

Bruker elevoppgaven i Scratch-prosjektet for å lære seg programmering

Etter observasjonene i avslutningen av Scratch-prosjektet snakker jeg med informant 15 og 16, der jeg ber dem om å vise meg labyrinten. Informantene viser meg spillet der de har delt inn labyrinten i lukkede rom der utforskeren må bevege seg med bruk av portaler for å komme seg til de forskjellige rommene. Det blir stegvis vanskeligere for hvert nivå man løser. Elevene trekker frem store utfordringer med å programmere portaler og bruken av koordinater som de må bruke lengre tid på å løse. Informantene påpeker videre at de har utfordringer med hastighet i møte med veggen, som de også klarer å løse, der de spesifiserer at dette er særdeles utfordrende i møte med portaler.

Informantene kommenterer fremgangsmåten i intervjuet:

> *E15:* Ehm, først, først når jeg startet med den, så fulgte jeg oppskriften. Og liksom lagde sånn på gruppa. Men etter hvert, mens tiden begynte å gå gikk jeg mer og mer fra oppskriften, og så startet jeg på nytt et par ganger fordi selve kodingen var ganske klumsete, og liksom ikke særlig effektiv. Så jeg startet på nytt flere ganger for å gjøre det mer og mer effektiv, og mindre sånn, mindre bortkastet koding.
>
> *E16:* ja –
>
> *E15:* Tilslutt så gjorde jeg bare, så brukte jeg ikke oppskrifta til å lage det i det hele tatt. Jeg bare gjorde det selv.
>
> *E16:* Vi brukte liksom det der til å lære oss det «basics», og så når vi kunne det, så begynte vi liksom å gjøre som E15 sier, å lage våre egne.

Diskusjon

Informant 15 og 16 velger å løse prosjektet ved å først ta utgangspunkt i elevoppgaven. Elevoppgaven brukes til å lære seg det grunnleggende som er relevant for spillet, for så å starte på nytt. Deretter lager informantene tre helt nye nivå der jeg vil hevde at de gjennomfører en kreativ løsning på prosjektet. I denne sammenheng vil jeg trekke inn fordelene med forbindelser mellom prosedyrebasert og konseptuell kunnskap. Det påpekes at man kan lagre prosedyrene i et konseptuelt system, og på denne måten kan man evaluere om prosedyrer er nyttige eller ikke, velge bort en prosedyre eller bedømme utfallet av en prosedyre (Hiebert & Lefevre, 1986, s. 12). Denne form for kunnskap vil ha en sterk sammenheng med å analysere nytten og bruken av algoritmer (Thomas, 2014, s. 37). Slik det fremkommer i observasjoner og intervju, vil jeg hevde at informantene skaper forbindelser ved å lære seg det grunnleggende og bygger videre på denne kunnskapen. I denne prosessen arbeider elevene med matematikk som de påpeker har vært utfordrende. Her vil jeg fremme at

informantene har klart å knyttet arbeidet med koordinater opp mot matematikken som kan vitne om sterke forbindelser til konseptuell kunnskap. Dette vil innebære arbeid med algoritmer, der elevene må reflektere rundt bruken og nytten av algoritmer når de skal løse utfordringene som de påpeker oppstår med arbeidet med koordinater. Informantene hevder videre at denne utfordringen har oppstått i forbindelse med å gå inn i veggen, noe som gjorde at de fant løsningen.

4.2.1.3 Følger ikke elevoppgaven

I mine observasjoner i Scratch-prosjektet, kommer det tydelig frem at flere av gruppene arbeidet på tvers av de fastsatte gruppene. Det ble notert ved gjentatte anledninger i observasjonene at informant 3 og 4, som er på samme gruppe, samarbeidet med flere elever fra andre grupper gjennom prosjektet. I intervjuet kommenterer de:

Andrej:	Ehm, fulgte dere oppskriften?
E3:	Nei, vi fulgte den ikke.
Andrej:	Dere fulgte den ikke?
E4:	Hmm, nei.
E4:	Jeg fulgte egentlig mest hvordan de andre hadde gjort det. Jeg søkte på hvordan andre hadde gjort det og så hentet informasjon fra andre labyrinter.

Diskusjon

Både intervju og observasjoner viser at elevene valgte å ikke følge elevoppgaven som de fikk utlevert i Scratch-prosjektet. Informantene påpeker at de løser oppgaven ved å tilpasse eller endre på løsninger fra andre, både fra valgfaget og fra internett. Dette kan tyde på at elevene har samarbeidet på tvers av gruppene, noe som åpner opp for at communities of practice kan trekkes inn, altså at kunnskapen er sosialt situert (Lesh & Zawojewski, 2007, s. 789). Communities of practice kan fremheves ved at flere av informantene har arbeidet sammen med et problem som har vært krevende og der de måtte arbeide jevnlig sammen for å løse det (Ibid). Dette kan kobles med tidligere forskning, som påpeker at elevene samarbeidet bra i Scratch-prosjekt, der man fikk utviklet god kompetanse i kommunikasjon (Koptelov & Taube, 2015, s. 95).

4.3 Oppnådd innhold

I kapitlet om det oppnådde innhold presenteres det funn ut fra det elevene sier de sitter igjen med, der jeg trekker ut totalt 5 tema som er gjeldende. Jeg vil presentere episoder fra intervjuet som vil gi en oversikt over mine funn, med en påfølgende diskusjon.

4.3.1 Informantene kommenterer matematisk innhold i programmeringen

I intervjuet spør jeg alle informantene om hvilken matematikk de har brukt i programmeringen. Dette på bakgrunn av at vi har hatt en gjennomgang av hva informantene har gjort i valgfaget frem til nå, med fokus på Scratch-prosjektet.

Andrej: Hvilken matematikk mener dere er i valgfaget?

E3: Funksjoner.

Andrej: Funksjoner?

E3. Koordinatsystem.

E4: Ja!

Andrej: Koordinater?

E4: Det, er jo litt på funksjon, ja.

Andrej: Hm?

E4: Mest funksjoner og koordinatsystem.

E4: Det er litt sånn algebra, bare at man ganger litt ting forskjellig, sånn her: gang det med hundre, jeg vet ikke, noe sånn der. For eksempel, jeg vet ikke.

Dette gir et godt inntrykk av det som kommer frem i intervjuene, der informantene ramser opp forskjellige tema. Her påpeker informantene at det er koordinater og funksjoner. Informantene nevner algebra, men beskrivelsen ligner mer på aritmetikk. For å forsikre meg om at elevene ikke bare ramser opp tema, ber jeg de komme med eksempler:

E4: Ehm.

E3: Prosent.

E4: Ja, prosent.

E3: Sett størrelse til prosenten man hadde lyst til det skulle være.

Andrej: Jaha. Noe mer matematikk dere husker ble brukt da?

E3: Hastighet.

Andrej: Hastighet ja.

E3: Hvor mange steg du skulle gå fra. Hvor mange piksler.

Andrej: Ja, brukte dere hastighet i programmeringen?

E4: Ja.

Andrej: Husker dere det?

E4: Det var en sånn der blokk som der sett hastighet. Eh, ja. Og så videre.

Andrej. Jaha.

E4: Åsså, vi brukte også koordinatsystem, fordi vi måtte jo sette figuren et sted på brettet.

E3: De fire regneartene brukte vi og.

Andrej: Dere brukte dem?

E3: Vi brukte de for å regne poengsum og sånn.

I denne sekvensen utdyper informantene flere av temaene som ble ramset opp tidligere, samtidig som det trekkes frem flere nye tema. Begrepet hastighet trekkes frem, i sammenheng med at de brukte blokk, noe som tyder på at de arbeidet med variabler. I forbindelse med at elevene ramser opp tema, kommer det frem i gruppe 2 at:

E8: Jeg vet at det er veldig mye matematikk, men på en måte så vet jeg ikke hva

E8: Jeg, jeg tenker ikke på det.

E7: Legger ikke merke til det.

E5: Jeg gjør det helt automatisk. Uten å tenke på det.

I etterkant av at informantene har ramset opp tema, kommer det frem fra gruppe 2 at de ikke klarer å trekke frem eksempler. Dette er interessant med tanke på at de på forhånd hadde trukket frem tema i programmeringen.

Senere i intervjuet til gruppe 1 kommer det frem:

E3: Vi har vært borte i matematikk hele tiden, mens vi har vært bort i programmering nesten hele tiden.

Andrej: Du har vært borti matematikk hele tiden?

E3: Nesten hele tiden.

Andrej: Du også?

E2: Ja. Man bruker det jo nesten til all programmeringen.

Andrej: Man bruker det til all programmeringen?

E2: Jada, nesten. Det er jo tall hele greia.

Andrej: Det er tall hele greia?

E3: Det er grunnleggende.

Andrej: Det er grunnleggende?

E4: Ja

E2: Mhm

E3: Skal man være litt mer avansert så har vi gått inn i funksjoner og prosent.

Andrej: Det har både vært grunnleggende og avansert da?

E3: Ja

E3: Mest grunnleggende.

I denne sekvensen kommenterer informantene at matematikken har vært grunnleggende, samtidig som de trekker frem matematikken som avansert, der de nevner begreper som prosent og funksjoner.

Diskusjon

Ut fra sekvensene kommer det tydelig frem at flere av informantene klarer å påpeke hva som er matematikk i programmeringen. Informantene trekker frem flere av de matematiske elementene som kom frem i det utførte innhold. Elevene kommenterer funksjoner og koordinatsystem, prosent og regning, der flere av elevene klarer å utdype dette med eksempler rundt disse. Informantene trekker også frem algebra som begrep der variabler også blir nevnt, men sannsynligvis mener de aritmetikk. Dette kan tyde på at algebra er et vanskelig begrep for elevene, og i tillegg at elevene egentlig mener arbeid med koordinater, funksjoner og variabler. Videre kommenterer informantene at de har vært borte i matematikk hele tiden, i den forstand at matematikken er grunnleggende, der de samtidig påpeker at det forekommer avansert matematikk i form av funksjoner og koordinater. Ut fra det elevene sier angående funksjoner og algebra, kan det tyde på at elevene snakker om arbeid med koordinater. Videre kan arbeid med variabler og funksjoner tyde på at elevene refererer til genererende aktiviteter i algebra, der Kieran (2007) påpeker at uttrykk ved generalitet som oppstår fra geometriske mønster og numeriske sekvenser, samt uttrykk for reglene som styrer numeriske forhold, framtrer som et godt eksempel på generaliseringsaktiviteter i algebra (Kieran, 2007, s. 713).

Ut fra funnene kan vi si at elevene ser en forbindelse mellom matematikk og programmering. Dette tyder på at læringen ikke er spesifikk for situasjonen den læres i (Andreson, 1996, s. 5), og at den kan overføres til andre situasjoner ved at elevene erkjenner og poengterer at de bruker matematikk. I denne sammenhengen vil jeg trekke frem informant 7 og 8, som ikke klarer å poengtere hvilken matematikk som forekommer i valgfaget. Dette tyder i høy grad på

at læringen har vært situert, ved at de ikke klarer å identifisere matematikken. Disse slutningene baseres på det som kommer frem i det utførte innhold.

4.3.2 Elevene kommenterer matematikken i programmeringen:

I intervjuene spør jeg elevene hva de mener om matematikken som finner sted i programmeringen, og hvor mye matematikk de mener det har vært. Her kommer det frem følgende:

> *E15:* Man kan se det på den her måten: Det er matte, men det er ikke alt som er lett å se, å kunne forklare at det er ren matte. Noe er liksom enklere hvis det står to tall og pluss til, så er det jo enkelt å si det er matte, men av og til så når man ser igjennom det, så virker det som at det ikke er særlig mye matte i det, på grunn av måten som det er kamuflert, hvis du kan si det sånn.

Senere i samme sekvens, kommer det frem:

> *E15:* Den er ganske lett fordi den er egentlig sånn helt vanlig, helt grunnleggende matte helt i starten, så hvis man er god i matte, så er det ganske enkelt hvis man skjønner hva som egentlig skjer.

I denne sekvensen trekkes matematikken i programmeringen frem som enkel. Samtidig fremheves matematikken som kamuflert der det er vanskelig å påpeke at det er ren matematikk. Videre kommer det frem at hvis man er god i matte, så er det enklere å skjønne hva som skjer i programmeringen.

Hos gruppe 2 kommer det frem:

> *E6:* Matematikk skal jo egentlig være vanskelig.
>
> *E7:* Fniser.
>
> *Andrej:* Synes dere noe var vanskelig da?
>
> *E6:* Det var enkel matematikk.
>
> *E7:* Det er sånn førsteklassematematikk.

I denne forbindelse vil jeg trekke frem at det senere i intervjuet til gruppe 1 kommer frem at:

> *E3:* Du har et veldig stort koordinatsystem.
>
> *Andrej:* Jaha.
>
> *E3:* I programmering i forhold til matematikken. For der har du masse piksler, små piksler du kan flytte på, men i matte har du veldig store ruter og.
>
> *Andrej:* Ja, jeg skjønner.
>
> *E3:* Så det kan være lettere i matteboka, enn i programmering.
>
> Andrej: Det kan være lettere?
>
> *E3:* Ja, fordi du jobber med mye mindre tall.

Diskusjon:

I første sekvens kommer det frem en nyansering om at matematikken er grunnleggende i programmeringen. Begge sitater fra første sekvens er meget interessante, da de styrker argumentasjonen for at det skapes forbindelser fremfor at læringen er situert. Videre kommer det frem at hvis man er god i matematikk, så er det lettere å skjønne hva som skjer. Dette er interessant, da tidligere forskning påpeker en sammenheng mellom lave problemløsningsferdigheter i matematikk og lave programmeringsferdigheter (Pacheco et al, 2008, s. 6). Det kan tyde på at informantene ser denne sammenhengen. På en annen side påpeker informant 6 og 7 at matematikken er grunnleggende og enkel. Dette er interessant med tanke på at informantene ikke klarte å løse utfordringen med koordinater, men samtidig kunne fremheve eksempler fra matematikk i programmering. Dette vil tyde på at læringen er situert, og at elevene dermed påpeker at matematikken er grunnleggende.

I denne sammenheng vil jeg trekke inn forbindelsene mellom prosedyrebasert kunnskap og konseptuell kunnskap der det å ha en kunnskapsbase er et av grunnlagene til at læringen kan overføres, som i dette tilfellet er å kunne trekke ut matematikken i programmeringen. Det kan tyde på at det er dette informant 15 påpeker. Her kan vi komme med eksemplet om arbeid med koordinatsystemer i programmeringen der det kommer frem hos flere av gruppene at dette var utfordrende og at de kobler det til matematikken. Denne koblingen gjør ikke informant 6 og 7.

4.3.3 Matematikk hjelper i programmeringen, matematikk i programmeringen hjelper ikke i matematikken:

Senere i intervjuet spør jeg informantene om de mener matematikkfaget har hjulpet i programmeringen, eller om at programmeringen har hjulpet i matematikkfaget:

> E15: Man må ikke, men det kan vær, det hjelper veldig ofte å kunne det, på grunn av måten liksom programmering er lagd. Så kommer det til å være situasjoner hvor man trenger å kunne matte for å komme videre i selve programmet.
>
> E17: Ja.

I sitatet ser elevene en kobling med matematikken i form av at man kommer bort i situasjoner der man trenger matematikk for å kunne programmere. Senere i intervjuene kommer det frem:

> E15: Matten er jo der i løpet av hele veien, helt fra starten til slutten av vanskelighetsgradene. Matte blir bare å fortsette å være der, det blir bare mer komplisert og vanskeligere å sette inn over tid.

I forbindelse med at matematikk hjelper i programmering, Så vil jeg trekke frem en sekvens fra informant 9 og 10:

E9: Nei, jeg tror ikke. Jeg tenkte ikke over at det var geometri, før man begynte å ha om geometri nå på skolen, utenom programmering.

E10: Mhm

Andrej: Oja, så det var i etterkant av programmeringen at dere hadde om geometri og sånn. Det her med grader og slikt, kunne dere det før dere begynte med oppgaven?

E9: Jeg tror at alle sammen hadde ganske basic om det på barneskolen og.

E10: Mhm

Det påpekes av informantene at de i etterkant, i den neste matematikktimen, la merke til og forsto at det var geometri de hadde anvendt. Samtidig påpeker de at det er grunnleggende matematikk. Senere i intervjuet kommer det frem:

E9: Hvis alle hadde trengt å lære seg programmering, og det hadde vært noe som hadde vært så logisk så hadde folk sett det og lagt det inn som et eget vanlig fag, eller i hvert fall hatt timer.

E10: Da ville sikkert matte og programmering slått seg sammen.

E9: Ja

Andrej: Dere mener det? At matte og programmering ville slått seg sammen?

E9: I hvert fall i neste generasjon. Fordi at vår generasjon blir å ha nytte av programmering, fordi da blir datamaskinene å ha kommet så sterkt da. Den er sterkere enn den står nå, fordi at det blir bare mer og mer. Det er veldig mange som jobber med å få bedre teknologi sånn sett, så.

Diskusjon:

Det kommer frem i intervjuet at informantene poengterer at matematikk hjelper i programmering, men at matematikken i programmeringen ikke hjelper i matematikkfaget. Informant 15 mener at grunnet måten programmering er bygget opp på, så vil man senere trenge matematikk i situasjoner for å komme videre. Dette kan tyde på at det snakkes om utfordringer knyttet til matematikk som man møter på, der det senere kommer frem at matematikken vil være en del av alle vanskelighetsgradene i programmeringen. Her kommer vi tilbake til at elevene påpeker at matematikken er grunnleggende, men slik som det fremstår i intervjuet, så fremhever informantene også at matematikken vil være en naturlig del av programmeringen, der det eksisterer matematikk som er mer utfordrende. Dette vil jeg fremheve som en interessant nyansering som jeg mener viser at informantene har et reflektert forhold til matematikken i programmeringsfaget. Dette vil styrke argumentet for at elevene

skaper forbindelse mellom matematikk og programmering ved at de konstaterer en overføringsverdi i form av at de trenger matematikk for å programmere.

I sammenheng med at matematikken påpekes som en del av programmeringen, vil jeg trekke frem informant 9 og 10, når de i matematikkundervisningen på skolen kommer frem til at det er geometri de bruker i programmeringen. Dette funnet er interessant, da dette er et betydelig eksempel på at læringen har vært situert i konteksten til programmeringen, men at elevene klarte å se sammenhengen til matematikkfaget i etterkant. Informantene fremhever samtidig at matematikken er grunnleggende. I denne sammenheng oppstår det en diskusjon om hvorvidt overføringsverdien kan vinkles mot regning som grunnleggende ferdighet, eller til matematikkfaget. Ut fra definisjonen av regning som grunnleggende ferdighet, vil all matematikk i andre fag være regning som grunnleggende ferdighet. For dermed å kunne si i hvilken grad overføringsverdien tenderer mot matematikkfaget, må det skapes forbindelser til den konseptuelle kunnskapen til elevene, der kunnskapen kobles til matematikken. Sentralt for alle informantene er at de ser på matematikken som sentral i programmeringen, selv om de påpeker at mye av matematikken er grunnleggende. Slik det fremstår i intervjuet, så skaper flere av elevene forbindelser med reflekterte nyanser til matematikkfaget. Dermed vil jeg hevde at matematikken elevene fremhever gir en overføringsverdi til matematikkfaget. Her vil jeg spesielt framheve informant 9 og 10, som konkluderer med at programmeringsfaget ville ha vært en del av matematikkfaget, hadde programmeringsfaget vært mer relevant i vår tid.

4.3.4 Elevene kommenterer arbeidsmåtene:

I intervjuene kommenterer elevene arbeidsmåtene de har arbeidet med. Dette kommer godt frem blant informant 9 og 10:

Andrej: Hvordan har det vært å arbeide med matematikken i programmering valgfag? Det pratet vi egentlig litt om, men har dere noen kommentarer til det?

E9: Mer komplisert enn i vanlig matematikk, fordi at det er så mye man må tenke på.

E10: Ja.

E9: Når man gjør matte i programmering så gjør man det for å klare noe. Når man gjør matte som et fag, så er det litt mer for å løse en oppgave, og det er det eneste. Men i programmering så er det bare en del av det.

E10: Og hvis man gjør den matten feil, så kan det hende at det ødelegger hele greia.

Her påpeker informantene at når man gjør matematikk i programmering, så gjør man det for å oppnå eller mestre noe, mens man derimot i matematikkfaget kun skal løse en oppgave.

Videre fremheves det som kritisk å gjøre feil i programmeringen. Det kommer frem senere i intervjuet at:

> *Andrej:* Hvordan har det vært å arbeide med matematikken i programmeringen fremfor vanlig undervisning. Så hvis dere vil si det igjen.
>
> *E10:* Ehm. At det var mer komplisert, fordi det var mer å tenke på. Og hvis vi kanskje gjorde en feil, så kunne det hende at vi ødela hele greia. Og man måtte gjøre hele greia på nytt. Når man gjør vanlig matte så er det bare liksom det, det er ikke noe mer matte liksom.
>
> *Andrej:* Mhm.
>
> *E10:* Jeg tror. Når vi for eksempel får vite en metode i matte, så bruker vi å holde oss til en metode. Fordi vi ikke lærer alle metodene, for vi synes når en metode passer bra til oss så bruker vi den. Når det gjelder Scratch så bruker vi mange metoder. Vi kan ikke bruke samme metoden til alt, fordi vi har forskjellige ting vi må lage å sånn der.

Her fremhever elevene at man i matematikkfaget lærer en metode som er gunstig. I matematikken som finner sted i programmeringen, brukes forskjellige metoder, grunnet at man må få programmet til å fungere og det er der det er kritisk å gjøre feil.

I sammenheng med å kommentere arbeidsmåtene, og i forhold til forskjellen i arbeidsmåtene i henholdsvis matematikk og programmering, vil jeg trekke frem følgende:

> *E3:* Artigere enn å sitte i en bok å holde på med dataen og skrive.
>
> *E2:* Ja, det er artig, det er artig å kunne bruke det man lærer og ikke bare lære hele tiden.
>
> *Andrej:* Jaha.
>
> *E4:* Ässå kan man få bruk for det man har lært, til å komme med mer selv i stedet for å følge en oppskrift til punkt og prikke.

Senere i intervjuet kommer det frem:

> *Andrej:* Var det noen ting dere ikke kunne som dere lærte i programmeringen?
>
> *E3:* Nei, vi kom ikke så langt.
>
> *E4:* Ikke noe annet enn at man fikk lære at man fikk bruke det. At man kunne bruke matematikken i programmering.
>
> *Andrej*: Hvordan har det vært da? Det å bruke matematikken i programmering på den her praktiske måten?
>
> *E3:* Artigere enn å jobbe i boka, og lettere.
>
> *E2:* Ja!

Diskusjon:

Når elevene klarer å poengtere en forskjell mellom arbeidsmåtene i programmering og matematikkfaget, vises det tydelig at de har blitt påvirket av arbeidsmåten i programmeringen. I denne sammenheng vil jeg trekke frem fremgangsmåtene til informantene i Scratch-prosjektet, som vil være grunnlaget for deres svar. I tidligere funn kommer det frem at elevene arbeider problemløsende med matematikk ved å løse sentrale utfordringer som oppsto gjennom programmeringen. I denne prosessen arbeidet elevene med å skape og manipulere algoritmer, der prosessen kalles for algoritmisk tankegang. Dette kan stemme godt overens med at elevene poengterer at det har vært komplisert og utfordrende å arbeide med matematikken hvor man bruker matematikken for å mestre noe. Dette reflekteres i deres svar og settes opp mot arbeidsmåter i matematikken. I matematikk fremheves det at man forholder seg til en metode, der man løser et problem. Dette kommer også frem hos gruppe 1 som påpeker at man følger en oppskrift til punkt og prikke i matematikken, mens det i programmeringen åpnes for flere muligheter. I matematikken vil elevene kunne dele en problemløsningsoppgave opp i ulike deloppgaver, som enten kan løses ved hjelp av kjente algoritmer i matematikken, eller at elevene selv vil måtte finne egnede løsningsmetoder. I programmeringen derimot vil elevene arbeide problemløsende gjennom den algoritmiske tankegangen og må skape, samt analysere nytten og bruken av algoritmer. I denne prosessen arbeider elevene problemløsende med matematikk.

Når målet er å lage programkode igjennom den algoritmiske tankegangen og få digitale enheter til å fungere, vil en feil i den algoritmiske tankegangen være kritisk. Dette kommer tydelig frem hos informantene når de poengterer at det i programmeringen er avgjørende å gjøre feil, mens det i matematikken ikke er så viktig. Her kan vi trekke inn tidligere forskning som påpeker at elevenes motivasjon og interesse økte for matematikk ved å engasjere seg i å utvikle spill (Koptelov & Taube, 2015, s. 94). Et synlig bevis på dette vises tydelig i gruppe 1, der informantene gir uttrykk for at de sitter igjen med positive inntrykk av arbeidsmåtene når de brukte matematikk. I sammenheng med at elevene opplever arbeidsmåten som positiv, påpekes det at man heller bør få elevene engasjert og motivert i kognitive prosesser som kan overføres, og ikke fokusere på hvilke kognitive prosesser et problem trekker frem. (Andreson et al, 1996, s. 9). I tidligere forskning påpekes det at ved kreativitet og tilgang til teknologi, som f.eks. ved å lage spill, kunne man gi elevene mulighet til å lære seg viktige tema som de opprinnelig hadde negative holdninger til (Koptelov & Taube, 2015, s. 90). Ut fra slik gruppe 1 svarer, tyder det på at dette stemmer overens med tidligere forskning.

En viktig bemerkning er at elevene ser en forbindelse til matematikkfaget, der de klarer å se at arbeidsmåtene innebærer bruk av matematikk. Dette viser at elevene klarer å se at det de har gjort, dvs. arbeidsmåtene de har anvendt, kan knyttes opp mot matematikkfaget. Elevene klarer dermed å se utover konteksten av programmering, som dermed vil tyde på at det ikke er situert, og dette vil igjen styrke at det eksisterer forbindelser til matematikkfaget. Dette vitner videre om at det eksisterer en overføringsverdi til matematikkfaget, ved at elevene evner å gjøre viktige refleksjoner rundt arbeidsmåtene i begge fag.

4.3.5 Elevene påpeker samarbeid som viktig i programmering:

I intervjuet tar jeg opp samarbeid, både i programmering, i matematikkfaget og i forbindelse med matematikken i programmeringen. Elevene reflektere over dette slik:

E3: Fordi det er vanskelig å gjøre alene, og er man flere som tenker, så finnes det flere løsninger.

E2: Man kan se på, eller løse oppgaven på, fordi man har flere synsvinkler å tolke problemet på.

E4: Ja, jeg er enig

E3: Kommer an på hva man foretrekker. Hvis man foretrekker å jobbe alene, så jobber man alene med matematikken. Men det er fortsatt lurt å jobbe flere, for det finnes ofte flere løsninger.

Her vil jeg supplere fra gruppe 4:

E15: Man kan programmere alene, men det er en god ide å programmere med andre. Kanskje ikke direkte på det samme, men å dele kunnskapene sine. Det er veldig viktig.

E17: Det er veldig vansk… Det er vanskelig, så du burde spørre andre om hjelp å sånn der.

Andrej: Ja

E17: Å hjelpe andre, om de trenger det.

Her vil jeg også trekke frem gruppe 2:

E5: Programmering, man gjør det i grupper.

E8: Ja, det er lurt å gjøre det i lag.

E5: Ja, det er veldig lurt å gjøre det i lag.

E7: Da kan man, da kan man feilsøke i lag.

E8: Det er best når man er på samme nivå.

Diskusjon:

Jeg har valgt å trekke frem sitater fra forskjellige grupper, fordi de impliserer så ulike vinklinger på samarbeid. Informantene påpeker at samarbeid gjør seg gjeldene i programmeringen, og da særlig i tilknytning til matematikken. Ut fra det som kommer frem i intervjuene, vil jeg hevde at begrepet communities of practice gjør seg gjeldende, hvilket refererer til læringsmiljø der mennesker av lik status arbeider sammen for å forbedre sin tilegnelse av kunnskap (Andreson et al, 1996, s. 9). Communities of practice tar også for seg at interaksjonene i gruppen fører til kunnskap og forståelse (Lesh & Zawojewski, 2007, s. 790). I intervjuene kommer det frem at man ved å samarbeide kan komme frem til flere svar, der informant 15 påpeker at man deler kunnskapen sin med gruppen. Videre poengterer informantene i gruppe 2 at det er lurt å programmere sammen og på samme nivå. Her vil jeg trekke frem informant 7 som fremhever at man kan feilsøke sammen. Dette tyder på at informanten ser at problemløsning er sentralt, samt at samarbeid er viktig. Dette er interessant med tanke på at informant 6 og 7 ikke klarte å løse utfordringen med labyrinten, og at de på bakgrunn av dette ser samarbeid og problemløsning som viktig.

Sentralt blant intervjuene er at communities of practice gjør seg gjeldende for elevene, både mellom gruppene, men også i klassen. Dette kan jevnføres med tidligere forskning, der det trekkes frem at elevene samarbeidet bra ved å arbeide med spill i Scratch og fikk utviklet god kompetanse i kommunikasjon (Koptelov & Taube, 2015, s. 95). Ut fra følgende diskusjon vil jeg dermed hevde at det eksisterer en overføringsverdi til matematikkfaget i form av communities of practice, der elevene arbeider problemløsende med matematikk i programmeringsfaget.

5 Oppsummering av tiltenkt, utført og oppnådd innhold.

I mitt studie stiller jeg forskningsspørsmålet: *På hvilken måte vil innhold og arbeidsmåter i programmering valgfag kunne gi en overføringsverdi til matematikkfaget?* For å svare på dette vil jeg i dette kapitlet oppsummere diskusjonen som kom frem i analysen, og svare på de tre underspørsmålene.

I styringsdokumentene og føringene til Programmering Valgfag kommer det en overføringsverdi i form av at elevene må arbeide problemløsende gjennom algoritmisk tankegang, som kan tenkes på som prosessen rundt det å lage ferdig script. Prosessen kan knyttes opp mot Polyas (1971) problemløsningsprosess, der det legges særlig vekt på det å se tilbake i form av feilsøking og forbedring av script. Gjennom denne prosessen ligger det et potensiale for at elevene arbeider med matematikk. Dette gir igjen et potensiale for en overføringsverdi ved at elevene vil reflektere rundt algoritmer, der de analyserer bruken, samt nytten av disse. I denne prosessen vil elevene kunne utvide og utvikle det Hiebert og LeFevre (1986) kaller for konseptuell kunnskap, som vil gi en overføringsverdi til matematikkfaget. Både i form av arbeidsmåten som tar sted i problemløsningsprosessen, men også det matematiske innholdet som kommer frem. Videre fremheves et potensiale for å arbeide med genererende aktiviteter i form av funksjons- og variabelbegrepet i programmeringen, som også vil kunne gi en overføringsverdi til matematikkfaget. Dette potensialet reflekteres i føringene som programmeringslærerne lager for valgfaget, der de gjennomfører et Scratch-prosjekt. I dette prosjektet må elevene lage en labyrint på tre nivåer, ved å arbeide parvis, der de får utlevert en elevoppgave, som elevene bes følge. I elevoppgaven kommer det frem flere matematiske tema. Disse funnene gir dermed et grunnlag for å svare på: *Hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil vi kunne finne i planer og føringer til programmeringsvalgfaget?* Jeg vil fremheve at det eksisterer en overføringsverdi i føringene og planene til valgfaget, gjennom problemløsning som tar sted i den algoritmiske tankegang. I dette ligger et potensiale for at elevene reflekterer rundt algoritmer, samt kommer i kontakt med matematisk innhold. Her vil elevene kunne skape forbindelser til den konseptuelle kunnskap, som forbindelsen mellom dette og prosedyrebasert kunnskap..

I utøvelsen av Scratch-prosjektet kommer det frem at elevene bruker varierte løsninger, der elevoppgaven er sentral. Overføringsverdien som ble påpekt i planene og føringene gjør seg gjeldene. Rammene for prosjektet, der elevene må bygge et spill med tre nivå, mener jeg

legger opp til at elevene må gå bort fra elevoppgaven, der elevene møter utfordringer av matematisk art. For å løse utfordringene må elevene bruke problemløsning, der Polyas problemløsningsprosess gjør seg gjeldende. I denne prosessen påpeker flere av elevene at det er matematikk de må bruke for å løse utfordringene. Det fremhevesat elevene må gjøre koblinger til den konseptuelle kunnskap, samt koblinger til den prosedyrebaserte kunnskapen. Dette gir en sterk overføringsverdi til matematikkfaget. Elevene som ikke klarer å løse utfordringene klarer ikke å gjøre forbindelser til den konseptuelle kunnskapen, noe som stemmer med tidligere forskning som påpeker en sammenheng mellom lave ferdigheter i programmering og lav matematisk kompetanse (Gomes et al, 2006, s. 6). Videre kommer det frem at flere av elevene samarbeider på tvers av gruppene, der begrepet Communities of practice gjør seg gjeldende, hvor kunnskapen er sosialt situert (Lesh & Zawojewski, 2007, s. 789).

Ut fra følgende funn vil jeg svare på delspørsmålet: *Hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil vi kunne finne i utøvelsen av programmeringsvalgfaget?* Arbeidsmåtene kommer frem i utførelsen av valgfaget, ved at elevene arbeider problemløsende med matematikk i prosessen rundt den algoritmiske tankegang. I denne prosessen må elevene løse utfordringer knyttet opp mot matematikk, der matematisk innhold kommer frem. Elevene må feilsøke og derav analysere algoritmer. Denne overføringsverdien knyttes opp mot at elevene må skape forbindelser til den konseptuelle kunnskap, samt forbindelser mellom konseptuell og prosedyrebasert kunnskap. Elevene som ikke gjør koblinger, klarer ikke å løse utfordringene.

Informantene klarer å fremheve matematisk innhold fra Scratch-prosjektet. Dette tyder på at læringen ikke er situert, der læringen ikke er spesifikk for situasjonen den læres i (Andreson et al, 1996, s. 5). Elevene som fulgte elevoppgaven, og ikke skapte forbindelser til konseptuell kunnskap, klarer ikke å påpeke det matematiske innholdet i prosjektet. Dette tyder på at elevene gjør forbindelser mellom den konseptuelle og prosedyrebaserte kunnskapen, som fører til at læringen ikke er situert. Dette gir en overføringsverdi til matematikkfaget, og ikke regning som grunnleggende ferdighet, grunnet at overføringsverdien skapes ved forbindelser. I denne sammenheng påpeker elevene at matematikken er grunnleggende, samtidig som at matematikken er usynlig. Likevel trekker elevene frem at det er flere matematiske utfordringer som kommer frem i prosjektet, der de blant annet kommenterer arbeid med koordinater. Dette kom også frem av observasjonene. Dette kan tyde på at elevene ser at det

er matematikk de bruker, men at de mener det matematiske innholdet som er i programmeringen er grunnleggende matematikk. Samtidig kommer det frem fra informantene at matematikken er en sentral del av programmeringsfaget, og at man vil trenge dette for å få progresjon i programmeringen. Noen informanter går så langt som å foreslå at programmeringsfaget bør slå seg sammen med matematikkfaget. programmeringen, noe som er interessant.

Elevene kommenterer arbeidsmåtene med matematikken som tar sted i programmeringen, der flere kommenterer at i matematikken forholder man seg som oftest til en metode, mens man i programmeringen må forholde seg til flere. Dette mener jeg kan kobles opp mot arbeid med algoritmer i matematikken, der det kommer frem at man i programmeringen må analysere algoritmer, der man gjør koblinger til den konseptuelle kunnskapen. Slik det kommer frem, så vil jeg påpeke at elevene blir påvirket av dette, der flere opplever arbeidsmåten som lystbetont. Dette stemmer med tidligere forskning som påpeker at elevenes motivasjon og interesse økte for matematikk ved å engasjere seg i å utvikle spill (Koptelov & Taube, 2015, s. 94). Videre kommer det frem at elevene fremhever samarbeid som en viktig egenskap i programmeringen, og i matematikken som tar sted i programmeringen. I dette eksisterer det igjen en overføringsverdi til matematikkfaget gjennom begrepet communities of practice, som refererer til læringsmiljø der mennesker av lik status arbeider sammen for å forbedre deres tilegnelse av kunnskap (Andreson et al, 1996, s. 9).

Ut fra disse funn kan vi svare på underspørsmålet: *Hvilken overføringsverdi til matematikkfaget, i form av innhold og arbeidsmåter, vil elevene påpeke på bakgrunn av å delta i programmeringsvalgfaget?* Elevene som gjorde forbindelser til den konseptuelle kunnskapen, klarer å fremheve det matematiske innholdet i valgfaget, som viser at læringen ikke var situert. Videre blir elevene påvirket av arbeidsmåtene som tar sted i forbindelse med matematikken i valgfaget, der elevene må feilsøke og arbeide problemløsende. påpeker at det i matematikken kun er en arbeidsmåte å forholde seg til, men flere i programmering. Dette begrunnes med at det er kritisk å feile i programmeringen. Noen elever fremhever at arbeidsmåtene i matematikk i programmering er mer lystbetont enn i matematikkfaget. En overføringsverdi ligger dermed i at elevene ser nytten av matematikken i programmeringsfaget, ved både å trekke frem innholdet, men også arbeidsmåtene. Videre eksisterer det en overføringsverdi ved at elevene opplever samarbeid som viktig i sammenheng med begrepet communities of practice.

6 Veien videre

Overføringsverdien mellom programmering valgfag og matematikkfaget har vært et interessant tema å forske på. Jeg vil påpeke at programmering valgfag enda er en prøveordning som gjennomføres for første gang i den norske skolen. Mitt studie har belyst mulighetene for en overføringsverdi til matematikkfaget, på et tema jeg føler enda er ungt. Ser vi det i sammenheng med dagens teknologiske utvikling vil programmering i skolen komme for å bli.

Det å programmere kan spire til motivasjon og engasjement fra elevene, noe som kommer frem i datamaterialet. Dette er noe som burde forskes mer på, samt brukes i undervisningen. Videre mener jeg det burde forskes mer på begrepsutvikling i matematikk gjennom det å programmere, både i programmering valgfag, men også i spesial lagde opplegg med begrepsutvikling som mål. I datamaterialet kom det frem at elevene samarbeidet, der begrepet communities of practice gjør seg gjeldende. Dette er et tema jeg mener det burde forskes mer på, der det tyder på at elevene kan ta over rollen til læreren i høyere grad en slik det kan forekomme i matematikkfaget. Når valgfaget har blitt gjennomført over flere år, mener jeg det vil være viktig å forske på deres oppfatninger og holdninger på matematikkfaget, og hvordan disse kan bli påvirket ved å delta i programmering valgfag.

7 Litteraturliste

Andreson, J. Lynne, M. Herbert, S. (1996). Situated Learning and Education. *Educational Researcher, Vol 25, No. 4*, s. 5-11.

Astad, H. (2013). *Pisa tall: Vi må gjøre matte gøy*. Kidskoder.no Hentet fra: http://kidsakoder.no/2013/12/09/pisa-tall-vi-ma-gjore-matte-goy/ Lest (24.08.16)

Bass, H. (2013). Computional Fluency, Algorithms, and Mathematical Proficiency: One Mathematicians`s Perspective. *National Council Of Teachers Of Mathematics*, 322-327.Hentet fra: http://staff.washington.edu/klomax/FAQs_files/Bass%20Computational%20Fluency.pdf Lest (10.03.17)

Braun, V & Clarke, V (2006). Using thematic analysis in psychology. *Qualitative Reasearch in Phychology*, S. 77-101.

Brown, Q. Morgan, w. Kusic, D. Garbaraine, E. Fromm, E & Fontecchio, A. (2008). *Computer Aided Instruction as a Vehicle for Problem Solving: Scratch programming Enviroment in the middle Years Classroom.* Philadelphia: Drexel Universety.

Gomes, A. Birgotte, E. Carmo, A. & Mendes, A. (2006). Mathematics and programming problemsolving. *3rd E-learning Conference* , S. 1-5.

Christoffersen, L. Johannesen, A. (2012). *Forskningsmetode for lærerutdanningene.* Oslo: Abstrakt forlag.

Cobb, P. (2007). Putting Philosophy to work, Coping with multiple theoretical perspectives. I F. Lester, *second handbook of research on mathematics teaching and learning*. NCTM. S. 3-38

Cohen, J. M. (2007). *Research Methods in Education.* New York: Routledge.

Ejsing-Duun, s. Misfeldt, M. (2015). Learning mathematics through programming: An instrumental approach to potentials and pitfalls. *CERME 9 - Ninth Congress of the European Society for Research in Mathematics Education* Prauge: CRME. S. 2524-2530

Hazzan, O. Lapidot, T. Ragonis, N. (2014). *Guide to Teaching Computer Science, An Activity-Based Approach.* London: Springer-Verlag.

Hiebert, j & Lefevre, P. (1986). Conceptual and Procedural Knowledge in Mathematics: An Introductory Analysis. I Hiebert, *Conceptual and Procedural Knowledge*. New York: Routledge. S. 1-23

kidsakoder.no. (2016). *Scratch*. kidsakoder.no Hentet fra: http://oppgaver.kidsakoder.no/scratch/ Lest (29.04.17)

Kieran, C. (2007). Learning and Teaching Algebra at the Middle School Through College Levels: Building Meaning for Symbols and Their Manipulation. I F. Lester, *Second Handbook of Research on Mathematics Teaching and Learning* NCTM. S. 707-762

Kloostermann, P. Raymond, A & Emenaker, C. (1996, September). Students Beliefs about mathematics: A three-year Study. *The Elementary School Journal*, S. 39-56.

Knuth, D. (1985). Algorithmic thinking and mathematical thinking. *The American Mathematical Monthly, Vol. 92, No. 3*, S. 170-181.

Koptelov, A. Taube, S. (2015). Learning mathematics and Critical thinking via Computer games design. *Journal of Mathematical Sciences 2*, S. 88-96.

Lagrange, J.-B. (2014). Algorithmics. I S. Lerman, *Encyclopedia of Mathematics Education* Dordrecht: Springer Netherlands. S. 32-36.

Lesh, R & & Zawojewski, J. (2007). Problemsolving and moddeling. I F. K. Lester, *Seccond handbook of research on mathematics teaching and learning*. NCTM. S. 763-804

Matematikksenteret.no. (2014). *regning i alle fag*. www.matematikksenteret.no Hentet fra: http://www.matematikksenteret.no/content/2029/Regning-i-alle-fag Lest (14.03.17)

Pacheco, A. Gomes, A. Henriques, J. Almeida, A & Mendes, A. (2008). Mathemathics and programming: Some studies. *International Conference on Computer Systems and Thechnologies - Comsystech`08*, Hentet fra: http://delivery.acm.org/10.1145/1510000/1500963/a77-pacheco.pdf?ip=129.242.197.171&id=1500963&acc=ACTIVE%20SERVICE&key=CDADA77FFDD8BE08%2E9FFE78EC01B95AD0%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=936814658&CFTOKEN=64220317&__acm__=1494861913_5c510b5979d415196f22dff1f82bb267

Polya, G. (1971). *How to solve it.* New Jersey: First Princeton Paperback Printing.

Regjeringen.no. (2013). *Fra matteskrekk til mattemestring*. Hentet fra Regjeringen.no: https://www.regjeringen.no/globalassets/upload/kd/vedlegg/grunnskole/strategiplaner/matematikk_aug_2011.pdf Lest (25.08.16)

Regjeringen.no. (2013, 12 03). *PISA 2012: Svakere resultater i matematikk og naturfag*. Hentet fra Kunnskapsdepartementet: https://www.regjeringen.no/no/aktuelt/pisa-2012-svakere-resultater-i-matematik/id747180/ Lest (25.08.16)

Regjeringen.no. (2013). *Satsing på realfag*. Hentet fra Regjeringen.no: https://www.regjeringen.no/no/tema/utdanning/grunnopplaring/artikler/et-felles-loft-for-realfagene/id279649 Lest (25.08.16)

Regjeringen.no. (2016). *Koding blir valgfag på 146 skoler*. Hentet fra Regjeringen.no: https://www.regjeringen.no/no/aktuelt/koding-blir-valgfag-pa-146-skoler/id2481962/ Lest 25.08.16)

Saez-Lopezs, J. Roman-gonzalez, M. Vazquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study usin "scratch" in five schools. *Computers & education 97*, S. 129-141.

Schoenfeld, A. H. (2007). Method. I F. Lester, *Second handbook of research on mathematics teaching and learning*. NCTM. S. 69-105

Shonfeld, A. &. (2016). *An introduction for Robust Understanding (TRU) Framework.* Hentet fra map.mathshell.org: http://map.mathshell.org/trumath/intro_to_tru_20161223.pdf Lest (30.09.16)

Shonfeld, A. &. (2016). *The Teaching for Robust Understanding (TRU) Observation Guide for Mathematics.* Hentet fra map.mathshell.org: http://map.mathshell.org/trumath/tru_observation_guide_math_v5_20161125.pdf lest (30.09.16)

Stake, R. (1995). *The art of case study research.* Thousand Oaks: Sage Publications.

Stein, M. Remmiliard, J. Smith, M. (2007). How curriculum influences student learning. I F. K. Lester, *Second handbook of research on mathematics teaching and learning*. NCTM. S. 319-369

Taylor, M. Harlow, A. Forret, M. (2010). Using a Computer Programming Enviroment and an Interactive Whiteboard to Investigate some mathematical Thinking. *International Conference on Mathematics Education Research 2010*, S. 561-570.

Thomas, M. (2014). Algorithms. I S. Lerman, *Encyclopedia of Mathematics Education*. Dordrecht: Springer Netherlands. S. 36-37

TIMSS. (2001). *Framgang men langt fram.* Hentet fra uv.uio.no: http://www.uv.uio.no/ils/forskning/prosjekt-sider/timss-norge/TIMSS/2011/timss_2011_web.pdf Lest (26.08.16)

Udir.no. (2013). *Fortsatt en vei å gå.* Hentet fra Udir.no: http://www.udir.no/contentassets/478ff813bbdd4a6298f9a9ea646c48e3/pisa-2012-norske-resultater.pdf Lest (25.08.16)

Udir.no. (2016). *Forsøkslæreplan i koding valgfag.* Hentet fra udir.no: http://www.udir.no/globalassets/filer/lareplan/forsok/forsokslareplan-programmering-som-valgfag.pdf Lest (24.09.16)

Utdanningsdirektoratet. *Læreplan i mateamtikk fellesfag: Føremål.* www.udir.no Hentet fra: https://www.udir.no/kl06/MAT1-04/Hele/Formaal Lest (14.03.17)

Utdanningsdirektoratet. (2013). *Regning*. Udir.no Hentet fra: https://www.udir.no/laring-og-trivsel/lareplanverket/grunnleggende-ferdigheter/regning/ lest (14.03.17)

Van de Walle, J. Bay-Williams, J. Lovin, L. Karp, K. (2006). *Teaching student-Centered Mathematics Developementally Appropriate Instruction for grades 6-8.* New Jersey: Pearson Education.

Wikipedia.org. (2017, 04 14). *Scratch (programming language)*. www.wikipedia.org Hentet fra https://en.wikipedia.org/wiki/Scratch_(programming_language) Lest (20.03.17)

Yin, R. (2014). *Case Study Research.* Thousand oaks: Sage Publications.

8 Vedlegg

8.1 Vedlegg - Informasjonsskriv

Forespørsel om deltakelse i forskningsprosjektet

Konsekvenser av koding valgfag - en kvalitativ studie av elevers syn på matematikk

Presentasjon av prosjektet:

Dette er et mastergradsprosjekt i regi av universitetet i Tromsø V/ student Andrej Verstad.
Prosjektet omhandler koding valgfag, der jeg vil se på konsekvensene valgfaget har for matematikkfaget i skolen. En av begrunnelsene for at valgfaget gjennomføres er at det vil gjøre matematikk interessant og aktuelt hos elevene, noe jeg er nysgjerrig på om stemmer. Derfor har jeg stilt forskningsspørsmålet:
På hvilken måte påvirker koding valgfag elevenes syn på matematikkfaget?

Gjennomføring av prosjektet:

For å finne ut hvordan elevene påvirkes av koding valgfag, Vil jeg basere meg på intervju og observasjon som metode. Dette innebærer at jeg i starten av valgfaget gjennomfører et gruppeintervju, der vi fokuserer på elevenes syn på matematikk. Deretter gjennomføres det observasjon av undervisningen, med vekt på hvordan matematikk forekommer i valgfaget, og hvordan elevene arbeider med dette. Prosjektet avsluttes med et nytt gruppeintervju, som likt det de hadde i forkant, men som kan innebære elementer fra observeringen.

Spørsmålene som stilles i intervjusituasjonen vil basere seg på synet på matematikk, og vil ikke innebære spørsmål om elevens bakgrunn eller andre personlige spørsmål. I intervjuet vil jeg bruke båndopptaker, noe som gir meg muligheten til å være mer involvert i intervjuet. Under observasjonen vil jeg ta båndopptak av en eller flere arbeidsøkter i valgfaget. Dette for å få frem hvordan elevene arbeider med en gitt oppgave. Jeg vil presisere at elevene som intervjues vil bli anonymisert før masteroppgaven blir publisert, og det vil ikke være mulig å gjenkjenne hvilke elever jeg har intervjuet eller observert. Ved forespørsel kan både foresatte og elever få se intervjuguide.

Formelle avklaringer:

Det er hentet inn tillatelse av rektor og faglærere i programmering valgfag til å gjennomføre undersøkelsen. I tillegg er prosjektet meldt inn til Norsk Samfunnsvitenskapelige Datatjeneste (NSD) som ivaretar personvernet ved Universitetet i Tromsø. Lyd-opptak fra båndopptaker uten sender vil bli lagret i en ekstern lagringsenhet som låses inn i skap ved universitetet. Personopplysninger slik som navneliste, vil bare jeg ha tilgang til, og vil bli låst inne i skap ved universitetet. Etter prosjektslutt vil alt av datamateriell (dvs. feltnotater, Lyd-opptak og personopplysninger) blir destruert. Prosjektet vil bli avsluttet våren 2017.

Jeg vil presisere at det vil være frivillig å delta i prosjektet, og at man uten begrunnelse, på et hvert tidspunkt trekke seg fra undersøkelsen.

Dersom du har noen spørsmål til studien, ta kontakt med Andrej

Med vennlig hilsen

Andrej Verstad

e-post: Andrejv1@hotmail.com

Telefon: 92xxxxxx

Veileder:

Ove drageset

e-post: ove.drageset@uit.no

Telefon:7766xxxx

Svarslipp:

Kryss av:

Jeg godtar deltakelse i prosjektet

☐

Jeg godkjenner ikke deltakelse i prosjektet

☐

Dato og underskrift foresatte:

_____

8.2 Vedlegg – Godkjenning NSD

Ove Gunnar Drageset
Institutt for lærerutdanning og pedagogikk UiT Norges arktiske universitet

9006 TROMSØ

Vår dato: 02.09.2016 Vår ref: 49436 / 3 / ASF Deres dato: Deres ref:

TILBAKEMELDING PÅ MELDING OM BEHANDLING AV PERSONOPPLYSNINGER

Vi viser til melding om behandling av personopplysninger, mottatt 12.08.2016. Meldingen gjelder prosjektet:

49436	*Konsekvenser av koding valgfag - en kvalitativ studie av elevers syn på matematikk*
Behandlingsansvarlig	*UiT Norges arktiske universitet, ved institusjonens øverste leder*
Daglig ansvarlig	*Ove Gunnar Drageset*
Student	*Andrej Verstad*

Personvernombudet har vurdert prosjektet og finner at behandlingen av personopplysninger er meldepliktig i henhold til personopplysningsloven § 31. Behandlingen tilfredsstiller kravene i personopplysningsloven.

Personvernombudets vurdering forutsetter at prosjektet gjennomføres i tråd med opplysningene gitt i meldeskjemaet, korrespondanse med ombudet, ombudets kommentarer samt personopplysningsloven og helseregisterloven med forskrifter. Behandlingen av personopplysninger kan settes i gang.

Det gjøres oppmerksom på at det skal gis ny melding dersom behandlingen endres i forhold til de opplysninger som ligger til grunn for personvernombudets vurdering. Endringsmeldinger gis via et eget skjema, http://www.nsd.uib.no/personvern/meldeplikt/skjema.html. Det skal også gis melding etter tre år dersom prosjektet fortsatt pågår. Meldinger skal skje skriftlig til ombudet.

Personvernombudet har lagt ut opplysninger om prosjektet i en offentlig database, http://pvo.nsd.no/prosjekt.

Personvernombudet vil ved prosjektets avslutning, 30.06.2017, rette en henvendelse angående status for behandlingen av personopplysninger.

Vennlig hilsen

Katrine Utaaker Segadal

Amalie Statland Fantoft

Kontaktperson: Amalie Statland Fantoft tlf: 55 58 36 41

Dokumentet er elektronisk produsert og godkjent ved NSDs rutiner for elektronisk godkjenning.

NSD – Norsk senter for forskningsdata AS Harald Hårfagres gate 29 Tel: +47-55 58 21 17 nsd@nsd.no Org.nr. 985 321 884
NSD – Norwegian Centre for Research Data NO-5007 Bergen, NORWAY Faks: +47-55 58 96 50 www.nsd.no

8.3 Vedlegg – Bekreftelse på endringsmelding

BEKREFTELSE PÅ ENDRING

Hei, viser til endringsskjema registrert hos personvernombudet 09.09.2016.

Vi har nå registrert at personlig intervju endres til gruppeintervju, samt at det skal gjennomføres observasjoner av undervisningsopplegg. Observasjonen vil basere seg på organiseringen av valgfaget, arbeidsmåtene hos elevene og hvordan de løser de forskjellige metodene. Informasjon om observasjoner vil tilføyes informasjonsskrivet.

Det er fint om du sender det oppdaterte informasjonsskrivet til meg.

Vi har også registrert at utvalget utvides til 12-20 personer.

Personvernombudet forutsetter at prosjektopplegget for øvrig gjennomføres i tråd med det som tidligere er innmeldt, og personvernombudets tilbakemeldinger. Vi vil ta ny kontakt ved prosjektslutt.

Vennlig hilsen,

--

Amalie Statland Fantoft
Rådgiver | Adviser
Seksjon for personvernTjenester | Data Protection Services
Tlf: (+47) 55 58 36 41

NSD – Norsk senter for forskningsdata AS | NSD – Norwegian Centre for Research Data
Harald Hårfagres gate 29, NO-5007 Bergen
Tlf: (+47) 55 58 21 17
postmottak@nsd.no www.nsd.no

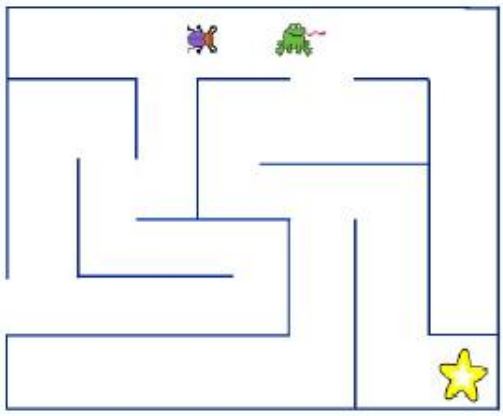
8.4 Vedlegg – Elevoppgave i Scratch-prosjekt

Labyrint

Introduksjon Scratch Lærerveiledning

Introduksjon

I dette spillet vil vi kontrollere en liten utforsker mens hun leter etter skatten gjemt inne i labyrinten. Dessverre er skatten beskyttet av den skumle froskekongen. Vi vil lære hvordan vi kontrollerer figurer, og hvordan vi kan programmere figurer til å bevege seg selv.



Steg 1: Hvordan styre figurer med piltastene

Vi begynner med å se på hvordan vi kan styre figurer med piltastene. For å få til dette vil vi bruke `Hendelser`*-klosser som merker når man trykker på tastaturet.*

Sjekkliste

- [ ] Start et nytt prosjekt.
- [ ] Slett kattefiguren ved å høyreklikke på den og velge `slett`.
- [ ] Legg til en ny figur. Klikk på -knappen og velg en figur du har lyst til å styre rundt. Vi har brukt `Dyr/Beetle`-figuren.
- [ ] Gi den nye figuren navnet `Utforsker` ved å klikke på `i`.

Vi begynner med å la figuren bevege seg oppover skjermen når vi trykker på `pil opp`-tasten.

- [ ] Legg til følgende skript på `Utforsker`-figuren din.

Prøv å trykk på `pil opp`-tasten. Beveger utforskeren din seg oppover skjermen? Nå må vi lage lignende skript for de andre tastene.

Test prosjektet

Klikk på det grønne flagget.

- [ ] Beveger utforskeren din seg rundt slik du hadde forventet?
- [ ] Kan du forandre hvor raskt utforskeren flytter seg?

Tallet `5` i 

-klossene bestemmer hvor raskt utforskeren flytter seg rundt. Vi vil gjerne eksperimentere litt for å se hvilken fart som passer best i spillet vårt, men for å endre farten må vi bytte tallet i fire forskjellige skript. Det blir for mye jobb!

Sjekkliste

Vi vil i stedet bruke en **variabel** som kan styre farten til `Utforsker`-figuren.

- [ ] Lag en ny variabel ved å gå til `Data`-kategorien og klikk `Lag en Variabel`.
- [ ] Kall variabelen `hastighet`, og velg at den bare skal gjelde `For denne figuren`.
- [ ] Til slutt, fjern avhukingen ved siden av den nye 

-klossen for at variabelen ikke skal vises på scenen.

Nå må vi endre i skriptene våre slik at bruker 

-variabelen.

- [ ] Lag først et nytt skript som setter verdien av 

til `10`.

- [ ] Deretter endrer vi de fire skriptene vi allerede har laget slik at de bruker 

.

Test prosjektet

Klikk på det grønne flagget.

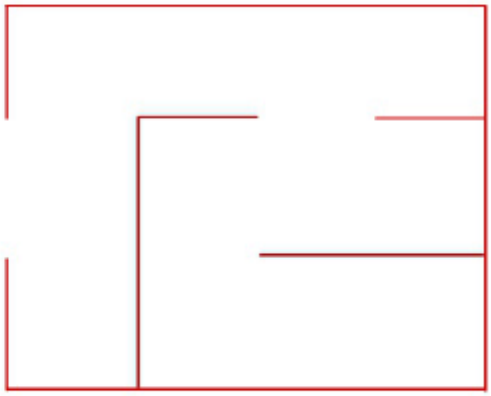
- [ ] Beveger utforskeren din seg fortsatt rundt slik den gjorde tidligere?
- [ ] Forandrer hastigheten til utforskeren seg hvis du endrer verdien av `hastighet` og klikker på det grønne flagget igjen?
- [ ] Velg en hastighet du synes passer.

Steg 2: Vi tegner vår egen labyrint

Nå som vi kan bevege utforskeren vår rundt omkring på skjermen, skal vi gi henne en utfordring! Vi vil tegne en labyrint som hun kan bevege seg rundt inni.

Sjekkliste

- [ ] Velg `/` nederst til venstre på skjermen for å tegne en ny bakgrunn. Pass på at du faktisk tegner en ny **bakgrunn**, og ikke en ny figur.
- [ ] Gi den nye bakgrunnen navnet `Labyrint`.
- [ ] Velg en farge du liker og tegn en liten labyrint. Det er viktig at alle veggene i labyrinten har samme farge (vi oppdager hvorfor snart). Du kan velge selv hvordan labyrinten skal se ut, den trenger ikke en gang å ha rette vegger!



Dette er et eksempel på en liten og enkel labyrint. Du kan selv velge hvordan din labyrint skal se ut! Men ikke bruk for lang tid på å tegne labyrinten nå, for vi vil jo fortsette å programmere. Du kan i stedet komme tilbake og tegne en mer avansert labyrint etter at du er ferdig med spillet!

Tips

Dersom du vil tegne rette vegger er det enklest å bruke linjeverktøyet, `\`. Du kan i tillegg holde inne `shift`-knappen for at linjene skal bli helt rette.

Test prosjektet

Klikk på det grønne flagget.

- [ ] Kan du bevege utforskerfiguren din rundt inne i labyrinten?
- [ ] Dersom figuren din er for stor kan du gjøre den mindre ved å trykke på -knappen på toppen av skjermen.
- [ ] Hva skjer dersom figuren din går på veggen i labyrinten?

Steg 3: Utforskeren kan ikke gå gjennom veggen

Selv om vi har tegnet en flott labyrint bryr ikke utforskeren seg noe om den. Hun kan bare gå gjennom veggene. Det skal vi gjøre noe med nå

Sjekkliste

For å oppdage når `Utforsker`-figuren vår går gjennom veggen på labyrinten vil vi bruke en **berører fargen** -kloss. Denne klossen merker om en figur kommer borti en spesiell farge. Her er det viktig at vi har tegnet alle veggene i labyrinten i samme farge.

- [ ] Vi legger **berører fargen** -klossen inn i skriptet vi allerede har laget som setter **hastighet** -variabelen.

- [ ] For å få riktig farge i  -klossen klikker du først på den lille firkanten hvor fargen vises. Deretter flytter du musepekeren slik at den peker på en vegg i labyrinten din. Da forandres fargen i den lille firkanten. Klikk igjen for å velge denne fargen.

Test prosjektet

Klikk på det grønne flagget.

- [ ] Blir utforskeren stoppet når hun prøver å gå gjennom veggen?
- [ ] Skjønner du hvordan skriptet sier at utforskeren ikke kan gå gjennom veggen?

Tips

En måte vi kan bruke for å begrense hvor en figur kan gå, er å tvinge den til å ta et skritt tilbake når den gjør noe feil. I koden

vil figuren først snu seg helt rundt (180 grader), deretter ta et skritt, og til slutt snu seg rundt igjen slik at den peker i samme retning som da den startet.

Steg 4: På leting etter skatten

Nå kan vi bevege oss rundt i labyrinten. Men det blir jo fort kjedelig om vi ikke har noe å gjøre inne i labyrinten. La oss se om vi kanskje finner en skatt!

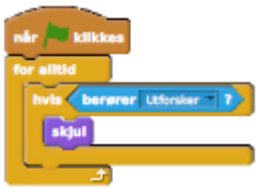
Sjekkliste

- [ ] Legg til en ny figur. Du kan velge en figur fra biblioteket ved å trykke eller tegne en figur selv ved å trykke . Vi brukte figuren `Ting/Star1`.
- [ ] Gi den nye figuren navnet `Skatt`.
- [ ] Dra skatten rundt inne i labyrinten din, og gjem den et sted den er vanskelig å komme til.

Vi skal nå lage litt kode som oppdager når utforskeren finner skatten. Her har vi faktisk et valg: Vi kan lage et skript på `Utforsker` som sjekker om hun berører `Skatt`, eller vi kan gjøre det omvendt, vi kan lage et skript på `Skatt` som sjekker om den berører `Utforsker`.

I dette tilfellet spiller det liten rolle hva vi velger, men om vi tenker oss at vi kanskje vil lage flere skatter senere kan det være litt enklere å lage skriptet på `Skatt`.

- [ ] Pass på at figuren `Skatt` er markert, og skriv følgende kode:



Test prosjektet

Klikk på det grønne flagget.

- [ ] Forsvinner skatten når utforskeren finner fram til den?

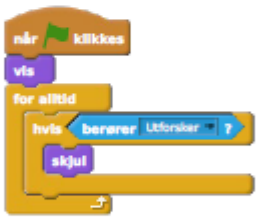
- [ ] Hva skjer når du prøver å starte spillet på nytt etter å ha funnet skatten? Hvor har skatten blitt av?

Sjekkliste

Det er et problem i spillet vårt. Etter at utforskeren har funnet skatten en gang, forblir skatten borte.

- [ ] Vi må passe på at skatten vises på begynnelsen av spillet. Endre skriptet på `Skatt` ved å legge til 

helt i begynnelsen.



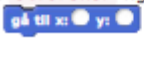
Vi har enda et problem: Når vi starter spillet på nytt står utforskeren fortsatt der den fant skatten sist. Det blir ikke veldig spennende.

- [ ] Klikk på `Utforsker`-figuren.

- [ ] Legg til en 

-kloss rett etter 

-klossen.

- [ ] For å finne ut hvilke tall vi vil bruke for `x` og `y` kan vi gjøre følgende. Dra utforskeren til et sted det er fint å starte fra. Se øverst i høyre hjørne. Sammen med `Utforsker`-figuren står det `x` og `y` og to tall. Dette er posisjonen til figuren akkurat nå. Skriv disse to tallene inn i 

-klossen.

- [ ] Hele skriptet vil nå se slik ut (dine tall for `x` og `y` vil være forskjellige):

Test prosjektet

Klikk på det grønne flagget.

- [ ] Forsvinner fortsatt skatten når utforskeren finner fram til den?
- [ ] Virker spillet slik det skal når du starter det på nytt etter å ha funnet skatten?

Steg 5: Froskekongen vokter i gangene

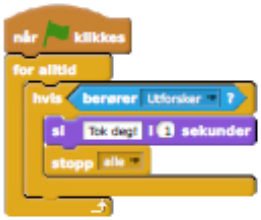
Nå skal vi gjøre spillet vanskeligere. Froskekongen vandrer rundt i labyrinten og passer på skatten.

Sjekkliste

- [ ] Legg til en ny figur. Vi brukte `Dyr/Frog`. Gi den navnet `Froskekonge`.
- [ ] Plasser den nye figuren et sted i labyrinten. Gjør den mindre eller større om nødvendig.

Vi begynner med å la `Froskekonge` merke at den fanger utforskeren. Dette blir veldig likt hvordan `Skatt` merket at den ble funnet.

- [ ] Legg til følgende kode:



Linjen  gjør at skriptet på `Skatt` slutter å kjøre. Det betyr at vi klarer ikke å få tak i skatten etter at vi har blitt tatt av `Froskekonge`.

Test prosjektet

Klikk på det grønne flagget.

- [ ] Hva skjer om utforskeren kommer borti froskekongen?

- [ ] Hva skjer når du finner skatten etter å ha blitt tatt av froskekongen?

Sjekkliste

Til sist skal vi få froskekongen til å bevege seg rundt i labyrinten.

- [ ] Start et nytt skript på `Froskekonge`-figuren. Igjen kan du bytte ut tallene for `x` og `y` med noe som passer for din labyrint.

- [ ] Før vi lar `Froskekonge` begynne å bevege seg lager vi en `hastighet` -variabel også for ham. Klikk på `Data`, og deretter `Lag en Variabel`. Kall variabelen `hastighet` og la den gjelde kun `For denne figuren`. Tilslutt, fjern avhukingen på variabelen.

- [ ] Vi kan nå utvide skriptet slik at froskekongen går fram og tilbake. Vi får ham til å snu når han treffer veggen på nesten samme måte som vi hindrer utforskeren i å gå gjennom veggen.

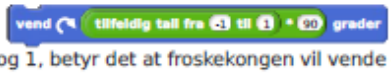
Helt tilslutt kan vi gjøre det enda vanskeligere ved å la froskekongen av og til endre retning.

- [ ] Legg til kode som lar `Froskekonge` snu seg tilfeldig rundt i labyrinten:

Disse to siste klossene ser litt kompliserte ut. La oss se litt nøyere på dem.

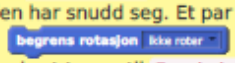
- [ ] Klossen 

sier at vi skal gjøre *noe* cirka èn av 25 ganger.

- [ ] Dette *noe* er 

. Tegnet `*` betyr gange, slik at om vi velger tilfeldig mellom tallene -1, 0 og 1, betyr det at froskekongen vil vende -90, 0 eller 90 grader. Det vil si at den svinger mot venstre, fortsetter rett frem eller svinger mot høyre.

Tips

Du kan av og til oppleve at `Froskekonge` setter seg fast i veggen. Dette er fordi `Froskekonge` fortsatt berører labyrintveggen etter at den har snudd seg. Et par ting du kan prøve for å forbedre dette er å gjøre `Froskekonge`-figuren mindre, legge en 

-kloss øverst i `Froskekonge`-skriptet, eller velge en figur som er *rundere* (prøv også å viske bort tunga til `Froskekonge` om du bruker `Dyr/Frog`-figuren).

Test prosjektet

Klikk på det grønne flagget.

- [ ] Klarer du å få tak i skatten?

- [ ] Om du synes spillet er for lett eller vanskelig er det mange måter du kan endre dette på! Prøv å lag froskekongen større eller mindre. Prøv å endre hastigheten på både utforskeren og froskekongen. Om du endrer tallet 25 i det siste skriptet vi laget for `Froskekonge` vil han endre retning oftere eller sjednere.

- [ ] Du kan også prøve å lage flere skatter. Prøv å høyreklikk på `Skatt`-figuren og velg `Lag en kopi`.

Lagre prosjektet

Da var vi ferdig med labyrint-spillet!

Nå kan du gå på skattejakt! Hvis du vil kan du dele spillet med familie og venner ved å trykke `Legg ut`.

8.5 Vedlegg – Intervjuguide førintervju

Fase 1: Introduksjon og rammesetting 2-5 minutter

Uformell prat

Informasjon

Formål og bakgrunn

Taushetsplikt, anonymitet

Informere om opptak, notater osv.

Er noe uklart, noen spørsmål

Fase 2: Overgangsfase, erfaringer 5 minutter

Tema: *Det å like skolen og matte:*

1)

Liker dere skolen?

Hvorfor liker du skolen? Hvorfor liker du ikke skolen?
-Hvilken hovedfaktor er det som får deg til å like/ikke like skolen?

2)
Hva tenker dere på når dere tenker på begrepet matte?

-Hvorfor liker dere matte? Hvorfor liker dere ikke matte?
-Har det alltid vært slikt?

Hvilken faktor var det som endret dette, og når ble dette endret?

Fase 3: Nøkkelspørsmål:

Tema: *Nytte av matte*

1) Føler dere at det er viktig å lære matte?
- Hvorfor er matte viktig? Hvorfor er matte ikke så viktig?

Når bruker man matte? Hvordan bruker man matte?
- Tror du at du kommer til å bruke matte i fremtiden?

Hvordan vil du bruke matte i fremtiden?

Hvilken type matte vil du bruke i fremtiden?

Kjenner du noen som bruker matte til vanlig? Hvem er dette?
Hvordan bruker de matte? Hvilken type matematikk bruker de?

Tema: *Matte alene eller sammen:*

Er matte noe man kan gjøre med andre eller er det noe du liker å gjøre selv?

Hvilken arbeidsmåte liker du best å jobbe med i mattematikk?

Hvordan synes du læreren jobber med mattematikk?

Tema: *Selvtilitt i matte:*

Gjør dere det bra i matte?
-Er det noen områder dere gjør det bedre i enn andre?

Var du bedre i matte før, eller var du dårligere?

Tema: *Matte hjerne:*

Tror du alle kan lære matte?

Er det noen som bare ikke er smarte nokk til å være god i matte, eller kan alle lære matte hvis de prøver hardt nokk? ikke navn

hvorfor tenker du at det finnes elever som ikke gjør det veldig bra i matte? eller hvordan vet du at alle kan lære matte?

Tror du alle kan lære matte på samme måte?

Fase 4: Oppsummering. 2-5 minutter minutter

Oppsummere funn

Har jeg forstått deg riktig?

Er det noe du vil legge til?

Er det noen spørsmål?

8.6 Vedlegg - Observasjonsskjema

	Mattematikk og fordring	Tilgang og identitet	Vurdering
ELEV			
ELEV			
ELEV			

Bakside:

Hovedpoeng på Matematikken og fordring	Hovedpoeng tilgang, identitet og vurdering
Logg og journal: utfordringer, faktorer o.l.	Notater, refleksjoner og ideer

8.7 Vedlegg – Intervjuguide, etterintervju

Fase 1: Rammesetting, 1 min
Uformell prat

Informasjon

Formål og bakgrunn

Taushetsplikt, anonymitet

Informere om opptak, notater osv.

Er noe uklart, noen spørsmål

Fase 2: overgangsfase:

Recall:
Da vil jeg veldig gjerne at vi skal starte med å gjennomgå hva som har skjedd i valgfaget frem til nå. Dere trenger ikke å ha fokus på matematikk eller sånnt, bare sånn gennerelt, og hva dere synes om det.

Hva har dere gjort i valgfaget frem til nå? -Hvilke hovedtema er det dere har gjort frem til nå? - Hva gjorde dere i de forskjellige temaene?

- Hva synes dere om det?

Her har dere en kort presentasjon av oppgave i scratch.

Husker dere den?

Hva gjorde dere i prosjektet?

Fulgte dere oppskriften? Laget dere noe annerledes?

Laget dere noe mer?

Hva synes dere var utfordrende?

Oppfølgingspørsmål til recall

Hvilken typer matematikk er det dere mener er i valgfaget?

Har dere noen eksempler?

Hvorfor er dette matematikk?

Hvor mye matematikk har dere hatt i programmeringsfaget? – Hvilke tema mener dere er brukt.

Hvordan har det vært å arbeide med matematikken i programmering valgfag?

Hvordan har det vært å arbeide med matematikken i programmering valgfag fremfor i vanlig undervisning?

Hvordan synes dere miljøet har vært i valgfaget. Nå tenker jeg med tanke på det faglige.

Spissede spørsmål: 12 min

Ut fra det vi sa i stad, så lurer jeg på:

Hvordan hjelper dette deg i matematikken? Hvordan hjelper det dere har gjort i programmering valgfag dere i matematikken?

Hjelper programmering i matematikken Eller hjelper matematikken i programmeringen?

Hvordan hjelper dette deg videre?

Har dere hørt om grunnleggende ferdigheter slik som regning? Forstår dere forskjellen mellom matematikk og regning som grunnleggende ferdghet?

Når dere har jobbet med programmering, så har dere jo fulgt en oppskrift? Hvordan er det når dere følger en oppskrift i matematikken?

Fase 3: Nøkkelspørsmål: 20 min

Liker dere/ liker ikke forsatt matematikk? Hvorfor?

Liker dere programmering da? Hva med matematikken i programmeringen?

Gjør dere det bra i matematikk? Gjør dere det bra i programmering?

Matematikk er noe man bare gjør alene? Hva med programmering da? Er Programmering noe man gjør alene?

Alle kan lære matematikk? Kan alle lære programmering da?

Alle kan lære matematikk hvis de prøver hardt nokk? Kan alle lære programmering hvis de prøver hardt nokk da?

Det er viktig å lære matematikk? Hvorfor er det viktig å lære matematikk?

- Er det viktig å lære programmering? Hvorfor er det viktig å lære programmering? Er det noen sammenheng?

Fase 4: Oppsummering. Maks 30 min, helst 25

Oppsummere funn

Har jeg forstått deg riktig?

Er det noe du vil legge til?

Er det noen spørsmål?