Praktisk innføring av programmering i matematikk

En kvalitativ studie knyttet til arbeidet med innføringen av programmering i et 1T-klasserom

Jonatan Glen Joseph

MDID4009 - Masteroppgave i matematikkdidaktikk

30 studiepoeng

Institutt for lærerutdanning og skoleforskning

Det utdanningsvitenskapelig fakultet

Vår 2022

Praktisk innføring av programmering i matematikk

En kvalitativ studie knyttet til arbeidet med innføringen av programmering i et 1T-klasserom

Jonatan Glen Joseph

2022

Praktisk innføring av programmering i matematikk

Jonatan Glen Joseph

http://www.duo.uio.no/

Trykk: Reprosentralen, Universitetet i Oslo

Sammendrag

Denne studien er et matematikkdidaktisk forskningsarbeid der det overordnede temaet er programmering i norsk matematikkundervisning. Med innføring av fagfornyelsen 2020 ble programmering en del av matematikkfaget. I skoleåret 2021/22 er programmering relativt nytt for mange matematikklærere og nytt for årets 1T-elever som ikke har hatt programmering på ungdomstrinnet.

Innføringen av programmering i den nye læreplanen i matematikk har ført til problemstillinger som per i dag er relativt lite avklarte i diskusjonen rundt den nye læreplanen, men som lærere i skolen likevel må forholde seg til i praktisk arbeid allerede nå. Disse problemstillingene er knyttet til hvor stor del av matematikkundervisningen som vil gå med til at elevene lærer å programmere, hvordan man skal strukturere matematikkundervisningen med programmering, og om man skal lære elevene programmering som fokuserer på grunnleggende forståelse av programmeringsspråkene eller til bruken i matematikk.

Hensikten med denne kvalitative studien er primært å belyse hvordan det arbeides med innføringen av programmering i et 1T-klasserom etter den nye læreplanen i matematikk, og bidra til innsikt i hvordan implementeringen av programmering i matematikk 1T har foregått i praksis. I studien har jeg intervjuet én matematikklærer i 1T og observert hans matematikkundervisning med programmering første termin av skoleåret 2021/22.

Funnene i denne studien viser at det er mulig å bruke programmering som et nyttig verktøy til å lære matematikk, men at arbeidet med å integrere programmering og «vanlig» matematikk i 1T-faget har vist seg å være utfordrende. Læreren betraktet det som for ambisiøst at elevene for dette årskullet jobber frem fullstendig kjørbare programmer i typiske problemstillinger som tas opp. Læreren valgte å nedprioritere undervisning av den tekniske arten, og heller fokusere på bruken til programmering i matematikk. Problemet med manglende elevkunnskaper om grunnleggende programmering gjorde det vanskelig å oppnå integrasjon mellom programmering og matematikk.

.

Abstract

This study is a mathematics educational research work where the main theme is programming in Norwegian mathematics teaching. With the introduction of the curriculum renewal in 2020, programming became part of the mathematics subject. In the school year 2021/22, programming is relatively new for many mathematics teachers and new for this year's upper secondary school students, who have not had programming at the lower secondary level.

The introduction of programming in the new curriculum in mathematics has led to issues that are currently relatively unresolved in the discussion around the curriculum, but which teachers in schools still must deal with in practical work already now. These issues are related to how much of the mathematics teaching will help students learn to program, how to structure mathematics teaching with programming, and whether to teach students programming that focuses on basic understanding of programming languages or their use in mathematics.

The purpose of this qualitative study is primarily to shed light on how work is done with the introduction of programming in a 1T classroom according to the new curriculum in mathematics and contribute to insight into how the implementation of programming in mathematics in upper secondary level has taken place in practice. In the study, I have interviewed one mathematics teacher and observed his mathematics lessons with programming during the first term of the school year 2021/22.

The findings of this study show that it is possible to use programming as a useful tool to learn mathematics, but that the work of integrating programming and "ordinary" mathematics in the 1T subject has proved to be challenging. The teacher considered it too ambitious for the students in this year group to work out completely executable programs in typical issues that are addressed. The teacher chose to downplay teaching of the technical details related to coding, and rather focus on the use of programming in mathematics. The problem of lack of student knowledge about basic programming made it difficult to achieve integration between programming and mathematics.

Forord

Fem år som lektorstudent ved Universitet i Oslo rundes av med ferdigstillingen av masteroppgaven. Det har vært en utfordrende, men lærerik opplevelse å kunne fordype seg i fagfornyelsen. I løpet av studietiden har jeg lært mye, utviklet meg faglig og personlig og møtte mange flotte mennesker. Nå ser jeg frem til mange lærerike og spenningsfylte år som matematikk- og naturfagslærer.

Jeg vil rette en stor takk til veilederen og favorittforeleseren min Arne Hole. Dine gode råd og innspill har bidratt til at jeg kommet meg i mål. Takk for god veiledning.

Takk til lærerinformanten min og klassen som deltok i prosjektet. Takk til mine gode venner Umar og Shamshad for gode lattere gjennom hele studietiden. Til slutt må jeg takke min familie som alltid har stilt opp for meg. Min kjære mor og min kjære søster har vært mine to viktigste støttespillere i et tøft løp. Jeg er utrolig takknemlig for deres hjelp og tålmodighet.

Jonatan Glen Joseph

Mai 2022

Innhold

1 Innledning

I Norge har vi sett at utviklingen av nye læreplaner påvirkes av internasjonale trender (Grønmo & Hole, 2017, s. 49). Programmering har i noen år vært en del av læreplanene i Storbritannia, Sverige, Danmark, Finland og flere andre land som et eget fag, eller som en del av matematikkopplæringen (Bocconi et al., 2018, s. 1). Fagfornyelsen og de nye læreplanene i Norge både i matematikk og naturfag, inkluderer programmering. Implementering skal bidra til at matematikkopplæringen i Norge utvikler elevenes ferdigheter og kompetanser som kreves i det 21. århundre.

I NOU-rapporten *Hindre for digital verdiskaping* publisert i 2013 ble den manglende programmeringskompetansen blant det norske folk påpekt (NOU 2013: 2, s. 10). I NOU-rapporten *Fremtidens skole – Fornyelse av fag og kompetanser* levert av Ludvigsen-utvalget i 2015, ble det understreket at skolene bør tilrettelegge for fremtidens kompetansebehov. I Norges offentlige utredninger kommer det frem at Norge nå er inne i den såkalte fjerde industrielle revolusjon (NOU: 2020:2, s. 13). Den fjerde industrielle revolusjonen kjennetegnes av teknologiens fremmarsj som blant annet betyr nye forretningsmodeller, digitalisering og automatisering. Digitale ferdigheter som å programmere stilles som et behov både for arbeidsgivere og arbeidstakere.

1.1 Bakgrunn for valg av tema

I NOU-rapporten *Fremtidige kompetansebehov III – Læring og kompetanse i alle ledd* blir både programmering og algoritmisk tenkning nevnt eksplisitt gjennom følgende sitat:

> *«Ved å lære såkalt algoritmisk tenkning lærer man tankegangen og konseptene bak mye av informasjonsteknologien. Algoritmisk tenkning skjer ofte innenfor en kontekst av programmering, men trenger ikke å gjøre det. Å lære algoritmisk tenkning ser ut til å kunne styrke evnene til matematisk, kreativ og kritisk tenkning (OECD 2019c). Å lære programmering ser ut til å kunne øke kompetansen i andre nærliggende fag, som matematikk (Scherer mfl. 2019).»* (NOU: 2020:2, s. 43)

Fagfornyelsen og de nye læreplanene er nå innført for fullt i elevenes og lærernes hverdag i de fleste trinn, og innføres i tredje trinn på videregående skole høsten 2022. I de overnevnte landene har man erfart at suksessen til implementering av programmering er avhengig av lærerens tilnærming (Bocconi et al., 2018). I takt med endringene i fagfornyelsen stilles det

større krav til kompetansen blant matematikklærere, eksempelvis programmering og algoritmisk tenkning. Innføringen stiller krav til enda bredere og omfangsrikere matematisk kunnskap både faglig og didaktisk.

Å styrke lærernes digitale kompetanse understrekes i NOU-rapporten *Fremtidige kompetansebehov III – Læring og kompetanse i alle ledd* (NOU: 2020:2, s. 43). Personalet ved lærerutdanninger i landet har uttrykt det som viktig å utvikle den digitale kompetansen blant lærere, og har fått midler fra myndighetene til å utvikle lærernes digitale kompetanse (NOU: 2020:2, s. 43). Videre påpeker de at utvikling fører til nye arbeidsmetoder og en endring i lærerrollen (NOU: 2020:2, s.43). Likevel anser lærerutdanningene seg selv som middels gode i tilretteleggingen av å utvikle læreres profesjonsfaglige digitale kompetanse, og indikerer et stort behov for utviklingen av tilretteleggingen (Daus et al., 2019, referert i NOU: 2020:2, s. 43). Dette stemmer overens med at flere lærere ikke hadde programmeringskunnskaper før innføringen av den nye læreplanen, og den manglende kjennskapen til hvordan programmering skal integreres i matematikk (Johansen, 2020; Vogt, 2021).

I en allerede så hektisk hverdag må lærere møte utfordringene knyttet til å lære seg programmering og å undervise det. Problemstillingene og nytteverdiene knyttet til programmering i norsk skole, og behovet for forskning av programmering i den norske skoleopplæringen har vært bakgrunnen for valget av temaet.

1.2 Studiens formål og problemstilling

Det er utvilsomt et stort behov for programmering i det moderne arbeidslivet med tanke på teknologiens fremmarsj. Matematikk har fått hovedansvaret for programmering i den norske skole og åpner for mer problemløsing, algoritmebehandling og utforsking i faget. For å få til en vellykket integrering av programmering i matematikkfaget er man avhengig av at elevene ser nytteverdien av å programmere og hvordan det kan brukes som et verktøy til å hjelpe dem å forstå matematikken (Johansen, 2020). Fokuset kan ikke kun være på programmering, men sammenhengen mellom programmering og matematikk (Kaufmann & Stenseth, 2020, s. 1045).

Likevel oppstår det praktiske utfordringer med denne integreringen. Det er mange lærere som har manglende kompetanse i programmering og hvordan man skal undervise det (Vogt, 2021). Majoriteten av elevene i skolen har dette skoleåret eller det foregående skoleåret programmert for aller første gang. Dette kan føre til at undervisning med programmering kan bli for

programmeringsteknisk. Å lære å programmere er tidskrevende og kan gå på bekostning av andre aspekter og temaer i matematikkfaget.

Det fins foreløpig svært lite forskning om programmering i den norske skole på videregående nivå. Å svare på hvordan man skal undervise programmering i matematikkfaget på en god måte er krevende. Det er derfor et behov for forskning som tar for seg hvordan det fungerer i praksis å implementere programmering i matematikkfagene på videregående skole. I denne studien ønsker jeg å rette søkelys mot faktorer og forhold som påvirker denne implementeringen. På bakgrunn av dette har jeg formulert følgende problemstilling:

Hvordan arbeides det med programmering i matematikk i et 1T-klasserom etter LK20?

Problemstillingen er omfattende, og jeg har valgt tydeliggjøre hva jeg vil undersøke ved å dele den opp i to forskningsspørsmål. De to følgende forskningsspørsmålene tar for seg to forskjellige sider av problemstillingen:

1) *Hvilke praktiske utfordringer er det med arbeidet med programmering i matematikk 1T?*
2) *I hvilken grad er det problematisk at årets elever har hatt lite programmering på ungdomstrinnet?*

Det første forskningsspørsmålet tar for seg å kartlegge hvilke formål som ligger til grunn for implementeringen av programmering i matematikk 1T, og hvordan 1T-læreren i utvalget for studien arbeider med å få til denne implementeringen i praksis. Programmering i norsk skole er nytt både for læreren og elevene i utvalget. Som nevnt ovenfor, fins det manglende kompetanse i programmering og hvordan man skal undervise det blant lærere og lite empiri rundt programmering i norsk skole. Forskningsspørsmålet vil ta for seg de praktiske utfordringene som følger med integreringen av programmering i matematikkopplæringen.

Et aspekt av de praktiske utfordringene ved implementeringen vil jeg drøfte i lys av det andre forskningsspørsmålet. Majoriteten av elevene som har matematikk 1T skoleåret 2021/22 møter på programmering for aller første gang, og kan bidra til problematikk både for elevene og lærere. Etter noen år med programmering i læreplanen vil utfordringene bli mindre og andre enn det de er i dag. En del av problematikken rundt programmering dette skoleåret skyldes at det er nytt både for elevene og lærerne, og derfor tenker jeg det er nødvendig å drøfte dette aspektet når jeg forsøker å besvare problemstillingen.

Hensikten med studien er primært å belyse arbeidet med programmering i et 1T-klasserom etter den nye læreplanen i matematikk, og gi innsikt i hvordan implementeringen av programmering i matematikk 1T har foregått i praksis.

1.3 Tidligere forskning

Å få innsikt og oversikt over tidligere forskning i implementering og integrering av programmering i matematikkundervisning, vil gi meg en større forståelse for forskningsfeltet jeg skal undersøke, samtidig som jeg vil danne et grunnlag for empiri og hvilke funn og resultater som har blitt gjort i forskningsfeltet. Jeg vil også se på hvordan det forskes på programmering i matematikkundervisningen, og hvilke tilnærminger som er brukt. Det fins flere artikler om hvordan programmering implementeres i matematikkundervisningen, og i disse artiklene brukes det både kvalitative og kvantitative tilnærminger, i tillegg til mixed methods. Å få en oversikt over et utvalg av artikler vil kunne bidra til at jeg får innsikt i hva slags funn jeg kan forvente med tanke på hvilke tilnærmeringer jeg tar i bruk på mitt utvalg.

For å tilegne en større forståelse for forskningsfeltet som kan knyttes til å besvare problemstillingene oppgaven tar for seg, vil jeg rette fokus mot tidligere forskning som går mest mulig direkte på det jeg undersøker, nemlig programmering i norsk videregående skole. Programmering i skolen er ikke et helt nytt fenomen, men tidligere forskning gjort på programmering i matematikk på vgs-nivå i Norge er som nevnt begrenset. Derfor vil dette delkapittelet primært omhandle forskning om programmering i norsk skole på andre trinn som bruker andre programmeringsspråk enn det jeg undersøker i mitt prosjekt, samt forskning gjort i andre land som tar for seg rammefaktorer, forhold og kontekster i programmering i skoleopplæringen.

Forsström og Kaufmann (2018) sin studie tar for seg en litteraturgjennomgang hvor elevers læring av matematikk gjennom programmering undersøkes. En motivasjon for elevene til å programmere er programmeringens tilkobling til virkeligheten og virkelighetsnære problemstillinger (Forsström & Kaufmann, 2018, s. 25-26). I flere av studiene som ble undersøkt, ble elevenes evne til problemløsing og matematisk tenking tilsynelatende positivt påvirket. Man kan imidlertid ikke ta for gitt at programmering var årsaken til den positive sammenhengen, med tanke på at resultatene er avhengige av læringsaktiviteter og en rekke andre faktorer.

1.3.1 Programmering i norsk skole

Studien "Computer programming in the lower secondary classroom: learning mathematics" ble publisert i 2017 og er et eksempel på forskning i programmering i norsk matematikkundervisning (Lie et al., 2017). Det ble forsket på sammenhengen mellom programmering og matematisk læring, mer spesifikt matematisk tenking og problemløsing hos en gruppe elever mellom 10-11 år. Funnene i studien er gjort på grunnlag av samspillet mellom elevene og mellom elevene og læreren når det ble arbeidet med programmering (Lie et al., 2017). I studien ble det forsket på 45 elever i en klasse og hvilke læringsprosesser som er tilstedeværende når det blir gjennomført et undervisningsopplegg med det blokkbaserte programmeringsspråket Scratch i sammenheng med matematisk tenking og problemløsing.

Scratch er utviklet av Media Lab på Massachusetts Institute of Technology (MIT) og brukes ofte som en inngangsport til en mer abstrakt programmeringsverden. Scratch består av en samling av grafiske programmeringsblokker som brukeren kan sette sammen, fjerne fra hverandre eller rekombinere til å lage programmer (MIT, 2022; Taylor et al., 2010, s. 562). Blokkene er formet slik at de gir mening syntaktisk, og programmeringsfeil unngås. Boksene kan settes sammen på ulike måter og i forskjellige sekvenser for å se hva som skjer i programkoden. Man kan også inkludere målinger og geometri i koden, som for eksempel lengde og koordinater for å bestemme bevegelser og effekter. Ved å bruke Scratch kan barn lære matematiske begreper og konsepter ved kreativ tenkning (MIT, 2022; Taylor et al., 2010, s. 562).

For å operasjonalisere elevenes læringsprosesser og matematiske tekning har forskerne tatt utgangspunkt i Niss og Jensen (2002) sitt rammeverk for matematisk kompetanse, Sfards (1991) beskrivelse av matematisk forståelse og tenkning og Wings (2006) beskrivelse av algoritmisk tenkning. Forskerne observerte i undersøkelsen at matematisk og algoritmisk tenkning var til stede hos elevene når de programmerte (Lie et al., 2017). I studien inkluderes det flere eksempler der elevene viser flere delkompetanser av matematisk kompetanse gjennom programmering: representasjonskompetansen, resoneringskompetansen og kommunikasjonskompetansen.

1.3.2 Utfordringer og problemstillinger

I studien til Kaufmann og Stenseth (2020, s. 1045) konkluderer forskerne med at en suksessfull integrering av programmering i matematikkundervisning krever en lærer med tilstrekkelige programmeringsferdigheter og programmeringskompetanse. I norsk skole har man lærere med varierende programmeringskompetanse. Fagfornyelsen krever mye nytt av lærere, og det er kanskje mange som ikke er godt nok rustet til de nye oppgavene (Vogt, 2021). Disse utfordringene ble støtt på av matematikklærere i Sverige når den nye læreplanen i matematikk i Sverige ble innført i 2018. Læreplanen i matematikk i Sverige har flere likheter med LK20. I studien til Stigberg og Stigberg (2019) ble fire svenske matematikklæreres undervisning fulgt det første semesteret programmering var innført som en obligatorisk del av matematikkfaget. Lærerne uttrykte at manglende programmeringskompetanse og pedagogikk rundt programmering i matematikkundervisning bydde på en rekke utfordringer.

En utfordring som påpekes er tiden det tar å lære og lære bort programmering, og hvordan man skal få elevene til å oppleve nytteverdien av programmering (Vogt, 2021). Kaufmann og Stenseth ser faren for at matematikkundervisningen blir av programmeringsteknisk og syntaktisk fokus (Johansen, 2020). At programmering har blitt integrert i matematikkfaget uten at timeantallet i matematikk har økt, har ført til at avgjørelsen om integreringen har vært kontroversiell. Integreringen medbringer en rekke didaktiske spørsmål, herunder har jeg forsøkt å formulere noen av dem:

- *Hvor stor del av matematikkundervisningen vil gå til at eleven lærer å programmere?*
- *Vil elevenes programmeringsferdigheter gjøre at de lærer matematikk raskere i lys av alle læringsmålene, med tanke på at det går tid på programmeringsopplæring?*
- *Skal man lære elevene programmering som fokuserer på grunnleggende forståelse av programmeringsspråkene eller på bruken i matematikk?*
- *Hvilke temaer i læreplanen i matematikk kan man bruke programmering til, på en nyttig måte?*
- *Hvordan kan man strukturere matematikkundervisningen med programmering?*
- *Hvilke eventuelle nye temaer kan det være naturlig å introdusere i skolematematikken, nå som programmering inngår i læreplanen?*
- *Hvilke temaer i matematikk kan introduseres i fremtiden når elevene blir bedre i programmering?*

- *Hvilke oppgaver egner seg best når man skal lære elevene å programmere i matematikkundervisningen?*
- *Hvordan skal elevene bruke programmering på eksamen?*

Dette er problemstillinger som per i dag er relativt lite avklarte i diskusjonen rundt den nye læreplanen, men som lærere i skolen likevel må forholde seg til i praktisk arbeid allerede nå.

1.4 Struktur for oppgaven

Studien består av sju kapitler. Det første kapittelet tar for seg bakgrunnen for valg av studiens tema, studiens formål, problemstilling og forskningsspørsmål. Videre inneholder kapittelet tidligere forskning i programmering i norsk skole og matematikkundervisning.

Kapittel 2 tar for seg det teoretiske grunnlaget for oppgaven. Her presenteres teori rundt den nye læreplanen i matematikk, programmering og algoritmisk tenkning. Hensikten med kapittelet er å presentere hva som ligger til grunn for Utdanningsdirektoratets implementering og beskrivelse av programmering og algoritmisk tenkning. Til slutt i kapitelet presenteres studiens teoretiske rammeverk og teorien som ligger til grunn for rammeverket. Rammeverket er et selvkonstruert teoretisk rammeverk inspirert av Kilpatrick, Swafford og Findell (2001) sine fem tråder av matematisk kyndighet og Niss og Højgaard Jensen (2002) sine åtte delkompetanser av matematiske kompetanse.

I kapittel 3 gjør jeg rede for den metodiske tilnærmingen som er brukt i arbeidet med datamaterialet og analysearbeidet i studien. Kapittelet belyser refleksjoner jeg har gjort rundt metodiske spørsmål til prosjektet og min rolle som forsker, som kan ha hatt innvirkning på studiens resultater og kvalitet. Til slutt i kapittelet belyses refleksjoner rundt forskningens kvalitet i lys av prosjektets reliabilitet, validitet og forskningsetiske hensyn.

I kapittel 4 presenteres resultatene av datainnsamlingen og analyse av materialet.

I kapittel 5 presenteres og drøftes studiens hovedfunn. Problemstillingen og forskningsspørsmålene besvares i lys av teori og funn fra analysen og uttalelsene i kapittel 4 og tidligere forskning i feltet.

Avslutningsvis i kapitel 6 trekker jeg konklusjoner og kommenterer studiens avgrensninger og veien videre i forskningsfeltet.

2 Teori

I dette kapittelet vil jeg ta for meg det teoretiske grunnlaget for oppgaven.

2.1 Den nye læreplanen i matematikk

Matematikk som fag i den norske skole har et ansvar når det gjelder ferdigheter samfunnsborgere i landet bør ha (Utdanningsdirektoratet, 2019a). Matematikkfaget skal forberede elevene til videre studier og arbeid der matematisk kunnskap er en forutsetning. En slik holdning til matematikk kommer frem i den nye læreplanen i matematikk under «fagets relevans og sentrale verdier»:

> *«Kompetanse om teoretiske og praktiske anvendelser av matematikk forbereder elevene til videre arbeid og utdanning som stiller krav om matematisk forståelse.»* (Utdanningsdirektoratet, 2019a)

Teknologiens fremmarsj og arbeidsgiveres kravstilling til ferdigheter innenfor teknologi i dagens samfunn, gir en indikasjon på kompetansebehovet i dette århundret. Digitale ferdigheter har blitt betegnet som en av de fem grunnleggende ferdighetene i matematikk siden 2006 i den norske skole (Utdanningsdirektoratet, 2019b). Grover og Pea (2013) referert i Forsström og Kaufmann (2018, s. 19) påstår at programmeringens tilknytning til algoritmisk tenkning, gjør at programmering er viktig i elevers utvikling til å bli problemløsere og kreative tenkere, og er at vanlig argument i diskusjonen om innføringen av programmering i læreplaner i matematikk (Stenseth et al., 2019, s. 7). Relasjonen mellom programmering og algoritmisk og matematisk tenkning, gjør at matematikk har fått et betydelig stort ansvar for elevenes programmeringskompetanse etter fullført skolegang i mange land. I den forbindelse har programmering og algoritmisk tenkning blitt vektlagt og fått plass i fagfornyelsen som ble innført høst 2020 (Utdanningsdirektoratet, 2019a). Dermed følger Norge i fotsporene til naboland Sverige og Finland, som allerede har programmering som en integrert del av matematikkopplæringen (Bocconi et al., 2018, s. 1).

I den nye læreplanen i matematikk har det blitt introdusert fem kjerneelementer i matematikklæreplanen. Kjerneelementene i matematikk skal gi en indikasjon på hva som er det viktigste faglige innholdet i opplæringen, slik at man får informasjon om hva som skal til for å mestre faget og hvordan man kan bruke det i videre studier og i arbeidslivet

(Utdanningsdirektoratet, 2019c). Algoritmisk tenkning og programmering har en sentral sammenkobling med kjerneelementet *utforsking og problemløsing*, nemlig ferdighetene som Grover og Pea (2013) påpeker elevene kan utvikle når de tenker algoritmisk og programmerer. Dette kommer tydelig frem i beskrivelsen av kjerneelementet:

> «*Algoritmisk tenkning er viktig i prosessen med å utvikle strategier og fremgangsmåter for å løse problemer og innebærer å bryte ned et problem i delproblemer som kan løses systematisk. Videre innebærer det å vurdere om delproblemene best kan løses med eller uten digitale verktøy.*» (Utdanningsdirektoratet, 2019c):

2.2 Begrepet programmering og programmering i skolen

Programmering handler om å lage en programkode for blant annet datamaskiner. Aktiviteten innebærer å gi datamaskinen instrukser for å utføre en oppgave (Sevik et al., 2016, s. 9). Programmering blir tradisjonelt sett på som det å skrive programkode, beskrive et program og utvikle løsninger. Programmering knyttes ofte til begrepet algoritmisk tenkning. Aktiviteten innebærer å dele opp et problem i mindre bestanddeler, og bruke programmeringsfunksjoner og kommandoer til å angripe og løse bestanddelene. Dette gjør problemet mer forståelig.

Mye av forskningen gjort på programmering i klasserommet er utøvd på elever i aldersgruppen 6-16 år (Forsström & Kaufmann, 2018, s. 21). Forskningen ser på bruken av blant annet programmeringsspråket Scratch (MIT, 2022) og programmerbare roboter, som Lego Mindstorms (LEGO, 2020) i klassrommet. I denne studien vil jeg fokusere på programmering på vg1-nivå. På vgs-nivå i norsk skole er det primært det tekstbaserte programmeringsspråket Python som benyttes. Valget i å bruke Python i skolen og andre sammenhenger utenfor skolen, kan henge sammen med at Python er lettere å introdusere for nybegynnere i programmering og bruke til å løse matematiske problemer, sammenlignet med andre språk.

Programmering er ingen ny idé i verken matematikk eller i skolen. På 60-tallet ble programmeringsspråket LOGO introdusert med hensikt til å bli brukt i skolesammenheng av elever og lærere. Seymour Paperts tanke bak å bruke LOGO i skolen var å tilby nye og varierte måter å lære matematikk på (Papert 1993a og Papert 1993b). Tanken var at elevene skulle bli bedre problemløsere. På 80-tallet ble undersøkelser som tok for seg programmering i skolen gjennomført. Gjennomføringen av undersøkelsene og idéene medførte ikke den forandringen i utdanningen man kan ha forespeilet seg. Pea og Kurland (1984, s. 159-162) var kritiske til at

programmering med LOGO ville gjøre elevene til bedre problemløsere, og påpekte mangel på empirisk forskning rundt hvorvidt programmering i undervisningen gir ferdigheter i problemløsing og matematisk forståelse. Man blir ikke bedre problemløsere gjennom programmering av seg selv (Lye & Koh, 2014, s. 58).

I dag er situasjonen annerledes. Digital kompetanse legges vekt på i skolen, arbeidslivet og den digitale fremtiden. Å lære programmering er mer tilgjengelig, og ikke bare tilgjengelig for spesielt interesserte som det en gang var. Programmering er en obligatorisk del av elevenes hverdag i skolens utdanning og er blitt integrert i matematikkfaget i mange land (Selvik et al. 2016, s. 6). Selv om situasjonen er en annen i dag, tar vi med Paperts idéer fra 60-tallet videre. Scratch, som er et programmeringsspråk det er gjennomført en rekke nyere undersøkelser på, er basert på LOGO (Lye & Koh, 2014, s. 52). I forskningen undersøkes det faglige utbyttet og overføringsverdien programmering i skolen kan bidra med.

2.3 Algoritmisk tenkning

Forskningsprosjektet mitt tar for seg programmering i matematikkundervisning. For å fremheve matematikken i programmering er det imidlertid viktig å diskutere begrepet algoritmisk tenkning, og sammenkoblingen mellom algoritmisk tenkning og programmering. Jeg vil nå se nærmere på Utdanningsdirektoratets beskrivelse av algoritmisk tenkning og hvilket utgangspunkt som har blitt brukt for denne beskrivelsen.

Som tidligere nevnt, blir det at elevene skal tenke algoritmisk brukt til å støtte at programmering skal integreres i matematikkopplæringen Det finnes forskjellige definisjoner og beskrivelser av algoritmisk tenkning. Utdanningsdirektoratet understreker i sin beskrivelse sammenkoblingen mellom algoritmisk tenkning og programmering. Utdanningsdirektoratet beskriver algoritmisk tenkning som en gren innenfor problemløsing, og legger frem at dette innebærer å løse problemer ved hjelp av programmering:

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

Algoritmisk tenkning brukes ut ifra beskrivelsen som en problemløsningsmetode for både mennesker og teknologi, altså datamaskiner. Metoden går ut på å angripe delproblemer for å bedre forstå helheten. Sånn sett vil algoritmisk tenkning innebære å fokusere på bruken av variabler, setninger, kommandoer, betingelser og løkker, som i dette tilfelle kan ses på som delproblemer eller bestanddeler man trenger for å få et program til å løse et større problem for oss (Stenseth et al., 2019, s. 8).

I forbindelse med beskrivelsen av begrepet «algoritmisk tenkning» som norsk oversettelse av «computational thinking» har Utdanningsdirektoratet en utviklet figur med tittel «den algoritmiske tenkeren» som er tilpasset fra Barefoot Computing (UK), og som illustrerer hva Udir legger i begrepet algoritmisk tenkning. På nettsiden til Barefoot Computing finner man figuren Udir tar utgangspunkt i. Her finner man også informasjon og beskrivelser av nøkkelbegreper og arbeidsmåter for algoritmisk tenkning, deres plass i den engelske læreplanen og hvorfor dette er viktig. Det virker som Udir har tatt inspirasjon av denne informasjonen og disse beskrivelsene med tanke på at informasjonen og beskrivelsene stemmer overens med kompetansebehovet slik dette beskrives i den nye læreplanen i matematikk, og i NOU-rapporten *«Fremtidige kompetansebehov III – Læring og kompetanse i alle ledd»*. (Utdanningsdirektoratet 2019a, NOU: 2020:2).

Under har jeg plassert figuren til henholdsvis Udir og Barefoot Computing (UK), slik at det er lettere å observere likhetstrekk.

Figur 2.1 – Den algoritmiske tenkeren. Figur hentet fra «Algoritmisk tenkning» av Utdanningsdirektoratet (2019d). https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/

Figur 2.2 – The Computational Thinker: Concepts og Approaches. Figur hentet fra «Computational Thinking: Concepts and Approaches» av Barefoot Computing (2014). https://www.barefootcomputing.org/concept-approaches/computational-thinking-concepts-and-approaches?CatItemUrl=computational-thinking-concepts-and-approaches

Det har tidligere vært utfordringer med å finne en operasjonell definisjon av begrepet algoritmisk tenkning. En av grunnene til dette er at det har vært vanskelig å skille med algoritmisk tenkning og problemløsing. Dette har skapt en forvirring rundt begrepet og stilt spørsmål til hvordan man skal fremstille begrepet for elevene i skolen. Denne problemstillingen har blitt tatt opp av blant annet Jeanette Wing. Wing (2006) understreker at algoritmisk tenkning er en ferdighet som er fordelaktig for alle å utøve, og ikke bare informatikere. Algoritmisk tekning handler om å tenke som en informatiker, noe som innebærer å utvikle konseptuell forståelse og å tenke abstrakt. Algoritmisk tenkning er noe fundamentalt som fins hos alle, og innebærer mer enn programmering og informatikk. Wings formulering av algoritmisk tenkning innebærer å formulere et større problem som kan løses i form av å dekomponere problemet og en trinnprosess.

Grover og Pea (2013) uttrykker enighet med artikkelen til Wing om algoritmisk tenkning. Grover og Pea diskuterer potensialet elever i skolen har til å utvikle problemløsingsferdigheter,

kreativitet og evne til kritisk tenkning gjennom algoritmisk tenkning. Essensen og hovedtrekkene i forklaringene av algoritmisk tenkning til Udir, Wing, Grover og Pea er sammenfallende.

2.4 Programmering og algoritmisk tenkning/matematikk

I oppgaven har det blitt argumentert for implementeringen av programmering i matematikkopplæringen i den norske skole. Det fins også gode argumenter for å ha programmering som et eget fag, som det eksempelvis er i Storbritannia. Landene som har implementert programmering i matematikkfaget har lagt vekt på elevenes muligheter for utvikling av algoritmisk tekning, og har vært essensielt for at programmering har fått en plass i matematikkopplæringen. Noe som kan henge sammen med at mye tidligere forskning om algoritmisk tenkning i matematikk diskuterer hvordan programmering kan fungere som en slags inngangsport for algoritmisk tenkning inn i matematikken. Mange av disse studiene tar for seg hvordan programmeringens tilknytning til algoritmisk tenkning bidrar til å utvikle elevenes ferdigheter i problemløsing og kreativ tenkning (Forsström & Kaufmann, 2018, s. 19).

Når man programmerer må man nemlig gjennom algoritmer skrive en kode som programmet og datamaskinen skal tolke. For å utforme en velfungerende kode må man dekomponere problemet til mindre bestanddeler og trinn, for å gjøre det lettere å forstå og angripe problemet.

Udirs tilnærming for fremme programmering og algoritmisk tenkning for elevene i skolen, er å legge vekt på relasjonen mellom programmering og algoritmisk tenkning/matematikk. I læreplanen i matematikk blir programmering nevnt i fagets sentrale verdier, kjerneelementer og kompetansemål. Digitale ferdigheter blir sett på som en av de grunnleggende ferdighetene elevene skal utvikle i matematikk. Det gir elevene muligheter for flere og varierte muligheter for læring av matematikk. Med en læreplan som tilbyr mer utforskende, åpne oppgaver og problemløsing, kan programmering og algoritmisk tenkning tilby flere løsningsstrategier, modelleringer, representasjonsmuligheter, automatiseringer og kommunikasjon i matematikk.

2.5 Rammeverk for matematisk kompetanse brukt i min studie

I den nye læreplanen i matematikk kommer det frem at elevene viser og utvikler kompetanse når de praktiserer kjerneelementene (Utdanningsdirektoratet, 2019e). Kjerneelementene i den

nye læreplanen bruker formulereringer og beskrivelser som ligner de man kan finne i rammeverkene til Kilpatrick et al. (2001) og Niss og Jensen (2002) (Utdanningsdirektoratet, 2019c).

For å undersøke hvilke aspekter ved matematisk kompetanse som kommer frem i programmeringstimene til klassen jeg skal undersøke, benytter jeg et eget sammensatt rammeverk for matematisk kompetanse inspirert av Kilpatrick, Swafford og Findell (2001) sine fem tråder av matematisk kyndighet og Niss og Højgaard Jensen (2002) sine åtte inndelinger av matematisk kompetanse. Disse to rammeverkene for matematisk kyndighet/kompetanse består av inndelinger som er sterkt tilknyttet til hverandre. Rammeverkene har forskjeller og likheter i deres tilnærminger til matematisk kyndighet/kompetanse, noe som gjør at rammeverkene styrker hverandre. I dette delkapittelet vil jeg ikke gi en separat detaljert presentasjon av innholdet i hvert av de to rammeverkene, men heller en beskrivelse av aspektene jeg har bygget på, for hver av kategoriene i det egenkomponerte rammeverket.

2.5.1 Konseptuell forståelse

Kategorien *conseptual understanding* i trådmodellen til Kilpatrick et al. (2001) handler om å forstå og dirigere matematiske idéer, metoder og konsepter for å se viktige matematiske sammenhenger og ikke bare isolerte fakta (Kilpatrick et al., 2001, s. 118-120). Kategorien handler om å lære ved å utvikle forståelse og danne en dypere forståelse som gir et bedre grunnlag for matematisk læring. Dette innebærer å forstå når en matematisk idé og sammenheng er viktig og nyttig, og å anvende ulike matematiske representasjoner.

Dette er også et aspekt ved Niss og Jensen (2002) beskrivelse av *representasjonskompetansen*. Kompetansen handler om å reflektere over hvilken representasjon som er hensiktsmessig for ulike matematiske problemer og situasjoner (Niss & Jensen, 2002, s. 56-58). Representasjonskompetansen har en sterk tilknytning til *problemhandlingskompetansen* og *modelleringskompetansen* til Niss og Jensen (2002). *Problemhandlingskompetansen* handler om å formulere og løse matematiske problemer (Niss & Jensen, 2002, s. 49), mens *modelleringskompetansen* handler om å danne et bilde av virkeligheten gjennom matematikk (Niss & Jensen, 2002, s. 52).

Aspektene ved *konseptuell forståelse* i trådmodellen til Kilpatrick et al. (2001) og *representasjonskompetansen* i inndelingen til Niss og Jensen (2002) sin beskrivelse av matematisk kompetanse, er gjeldende i flere av kjerneelementene i læreplanen i matematikk.

Konseptuell forståelse og *representasjonskompetansen* er nært tilknyttet kjerneelementene *utforsking og problemløsing*, *modellering og anvendelsen* og *representasjon og kommunikasjon*, ut ifra beskrivelsene til Udir (2019c).

2.5.2 Beregning

Kategorien *procedural fluency* i trådmodellen til Kilpatrick et al. (2001) har blant annet å gjøre med beregninger. Kategorien vektlegger elevenes kompetanse om matematiske operasjoner og prosedyrer. Dette innebærer å håndtere matematiske utregninger, ulike matematiske representasjoner, symbol og formaliteter og hjelpemidler (Kilpatrick et al., 2001, s. 121-123). Å ha kompetanse om matematiske operasjoner og prosedyrer betyr å kunne utføre dem effektivt og fleksibelt. Denne kompetansen brukes som et sterkt hjelpemiddel i å gi flyt i problemløsing. *Beregning* henger i stor grad sammen med *konseptuell forståelse*, eksempelvis når forstår forskjeller og ulikheter ved regnemetoder.

Beregning kan også knyttes til to av delkompetansene rammeverket til Niss og Jensen (2002), nemlig *symbol- og formalismekompetansen* og *hjelpemiddelkompetansen*. *Symbol- og formalismekompetansen*, som er nært forbundet *representasjonskompetansen*, handler om å håndtere symbolholdige uttrykk og utsagn i matematiske operasjoner og prosedyrer (Niss & Jensen, 2002, s. 58-59). *Hjelpemiddelkompetansen* omhandler kunnskap om hjelpemidler og hvordan man betjener dem. *Hjelpmiddelkompetansen* er også nært forbundet *representasjonskompetansen* og *symbol- og formalismekompetansen*, ved at «ethvert hjelpemiddel involverer en eller flere representasjoner» og forståelse av symboler og formalisme i matematikk (Niss & Jensen, 2002, s. 62). *Beskrivelsene* av *symbol- og formalismekompetansen* og *hjelpemiddelkompetansen* inneholder sentrale aspekter som man finner i kompetansen *beregning*.

Aspektene ved *beregning* i trådmodellen til Kilpatrick et al. (2001) og *symbol- og formalismekompetansen* og *hjelpemiddelkompetansen* i inndelingen til Niss og Jensen (2002) sin beskrivelse av matematisk kompetanse, er gjeldende i samtlige kjerneelementer i læreplanen i matematikk. Aspektene er sentrale i samtlige kjerneelementer, da alle kjerneelementene i læreplanen i matematikk involverer håndtering og kunnskap om matematiske operasjoner, prosedyrer, hjelpemidler og symboler (Utdanningsdirektoratet, 2019c).

2.5.3 Strategisk kompetanse

Kategorien *strategisk kompetanse* (*strategic competence*) er en annen av komponentene i trådmodellen til Kilpatrick et al. (2001). Kategorien handler om strategiene og fremgangsmåtene i å løse et problem. Her vektlegges ikke bare det å formulere et matematisk problem, mens også å kunne løse det ved å utvikle en eller flere metoder (Kilpatrick et al., 2001, s. 124-129. Kompetansen styrkes jo mer man formulerer og løser et matematisk problem. Derfor er det viktig at man møter på flere og varierte problemer som skal formuleres og løses ved hjelp av en eller flere strategier og fremgangsmåter. Økt *strategisk kompetanse* er derfor proporsjonal med økt *konseptuell forståelse* og økt kompetanse i *beregning*. *Strategisk kompetanse* inneholder ikke bare aspekter i *konseptuell forståelse* og *beregning*, men også i representasjonskompetansen til Niss og Jensen (2002). Kompetansen innebærer å representere matematiske problemer, løsninger og strategier (Niss & Jensen, 2002, s. 52-53).

Strategisk kompetanse kan knyttes til flere av kjerneelementene i LK20, spesielt *modellering og anvendelser* og *representasjon og kommunikasjon*. Men også kjerneelementene *utforsking og problemløsing* og *resonnering og argumentasjon* da strategisk kompetanse styrkes ved å utforske og løse flere og varierte matematisk problemer, samt resonnere seg frem til en kognitiv modell for å løse et problem (Utdanningsdirektoratet, 2019c).

2.5.4 Resonnering

Resonnering handler blant annet om å vurdere strategier for å løse problemer i matematikk. I vurderingen av strategier legger det vekt på både egne og andres matematiske resonnementer. I trådmodellen til Kilpatrick et al. (2001) beskrives en bredere forståelse av resonnering i matematikk: «*Adaptive reasoning refers to the capacity to think logically about the relationships among concepts and situations*» (Kilpatrick et al., 2001, s. 129-131). Kapittel til å tenke logisk om forholdene i konsepter og situasjoner behøver ikke å gjelde kun for formell bevisføring. *Adaptive reasoning* innebærer også uformelle og intuitive resonnementer, argumenter, beskrivelser, begrunnelser og refleksjoner. I modellen presenteres kompetansen *resonnering* som limet i matematikk og viser vei for læring» (Kilpatrick et al., 2001, s. 129-131). Dette gjør at *resonnering* har en sentral plass i modellene til Kilpatrick et al. (2001) og Niss og Jensen (2002), og kan knyttes til de andre delkompetansene i modellene.

Resonneringskompetansen er en av delkompetansene i modellen til Niss og Jensen (2002). Niss og Jensen understreker viktigheten av å forstå hva et matematisk bevis er og forskjellene mellom et matematisk bevis og andre former for matematiske resonnementer. *Resonneringskompetansen* legger vekt på å gjennomføre uformelle og formelle bevis, og å omforme matematiske resonnementer til gyldige bevis (Niss & Jensen, 2002, s. 54).

Kjerneelementet *resonnering og argumentasjon* legger i stor grad vekt på prinsippene og beskrivelsene i delkompetansene *adaptive reasoning* og *resonneringskompetansen* i hver av modellene. *Resonnering* nevnes også i kjerneelementene *representasjon og kommunikasjon* og *abstraksjon og generalisering*. *Resonnering* beskrives som limet som holder alt sammen i matematikk i Kilpatrick et al. (2001, s. 129-131) og har en sentral plass i læreplanens beskrivelse av matematisk kompetanse (Utdanningsdirektoratet, 2019c).

2.6 Instrumentell og relasjonell matematisk forståelse

Matematisk forståelse kan deles inn instrumentell og relasjonell forståelse (Mellin-Olsen, 1981). Instrumentell forståelse handler om at man klarer å løse matematikkoppgaver fordi man gjenkjenner oppgavetypen og vet hvilke formler og regler man skal bruke (Skemp, 2006, s. 92). Denne formen for forståelse innebærer ingen dypere struktur. Relasjonell forståelse handler om å vite hvordan man skal løse et problem og vite hvorfor det gjøres slik (Skemp, 2006, s. 92). Relasjonell forståelse krever en dypere struktur der man ser sammenhenger i faget.

Skemp (2006, s. 94) presenterer et eksempel som tydeliggjør forskjellen mellom instrumentell og relasjonell forståelse. Hvis man skal komme seg fra startpunkt A til sluttpunkt B, vil man klare det ved å følge instrukser som viser veien fra A til B og gjøre akkurat det instruksene sier. Men gjør man en tabbe vil man ikke finne komme seg til B, hvis man ikke klarer å spore seg tilbake til den riktige stien. Dette forstås som en person med instrumentell forståelse av stien. En person med relasjonell forståelse har et mentalt bilde av området, og vil kunne navigere seg fra hvilket som helst startpunkt til hvilket som helst sluttpunkt i området. Dermed vil personen vite hvor han/hun er selv om personen går feil, og navigere seg frem til sluttpunktet.

I matematikkfagets kjerneelementer, sentrale verdier og overordnede del kommer det frem at læreplanen legger vekt på relasjonell forståelse av matematikk (Utdanningsdirektoratet, 2019aogc). Fokuset er ikke løsningene, men heller hvordan man kommer frem til løsningene

og matematikkfagets overføringsverdi. Samtidig understreker Skemp (2006, s. 93) at det ofte er vanskeligere og mer tidskrevende for elevene å oppnå relasjonell forståelse.

3 Metode

Metodene som er brukt for å besvare prosjektets problemstilling og forskningsspørsmål er intervju og observasjon, og disse to metodene representerer to ulike måter å analysere datamateriale på. I dette kapittelet vil jeg gjøre rede for den metodiske tilnærmingen som er brukt i arbeidet med datamateriale og analysearbeidet i masterprosjektet.

I løpet av høsten 2021 har jeg gjennomført to intervjuer med en faglærer i matematikk 1T og observert fem av hans undervisningsøkter som har involvert programmering. Observasjonsperioden strakk seg utover høstsemesteret fra 12.09.2021 til 15.10.2021. Det første intervjuet ble gjennomført dagen før første observasjon, og det andre ble gjennomført i første skoleuke etter juleferien.

I løpet av prosessen med å samle inn data til prosjektet har jeg reflektert rundt flere metodiske spørsmål til prosjektet og min rolle som forsker. Valgene og avgjørelsene tatt i forbindelse med gjennomføringen av masterprosjektet, kan ha hatt innvirkning på studiens resultater og kvalitet. Derfor vil jeg i dette kapittelet utdype avgjørelser og utfordringer som har ligget til grunn for innsamlingen av data og inngått i arbeidet med å belyse problemstillingen. Jeg vil gi en beskrivelse og begrunnelse for valg av metode og utvalg. Videre vil jeg diskutere utarbeidelsen av intervjuguidene, gjennomføringen av intervjuene, transkripsjonene og utarbeidelsen av analyseverktøyet. Jeg vil reflektere rundt forskningens kvalitet i lys av prosjektets reliabilitet, validitet og forskningsetiske hensyn mot slutten av kapittelet, men også underveis i kapittelet når jeg utdyper metodiske forbehold.

3.1 Metodisk tilnærming

I didaktisk forskning skiller vi mellom kvantitative og kvalitative forskningsmetoder. Kvantitativ forskning er vanlig å gjennomføre hvis man ønsker en statistisk representativ undersøkelse (Furseth & Everett, 2012, s. 149). Et kvantitativt utvalg er som regel større enn et kvalitativt utvalg, noe som kan bidra til at man får mye data. En vanlig kvantitativ forskningsmetode brukt i didaktisk forskning er spørreundersøkelser. I spørreundersøkelser svarer informantene i de representative utvalgene på de samme spørsmålene, og forskeren kan bruke avanserte bearbeidingsmetoder, som for eksempel statistikkprogrammer, for å gjennomføre analyser av datamaterialet. Likevel er det i kvantitative forskningsmetoder vanskelige å undersøke begrunnelsene til svarene til folk da man ikke har muligheten til å stille

dybdespørsmål, med mindre man bruker «mixed methods», som er en kombinasjon av kvantitative og kvalitative forskningsmetoder. (Furseth & Everett, 2012, s. 149; Frønes & Pettersen, 2021, s. 176). Å trekke konklusjoner ut fra data som ikke går i dybden kan svekke forskningen validitet (Frønes & Pettersen, 2021, s. 176)

For å styrke mitt prosjekts validitet har jeg på bakgrunn av oppgavens rammer valgt en kvalitativ tilnærming ved å studere et lite utvalg (Patton, 1999, s. 1197). En slik tilnærming er vanlig i didaktisk forskning og egner seg best når man skal forske på «menneskers erfaringer og oppfatninger» (Christoffersen & Johannessen, 2012, s. 17). Ved å observere og intervjue læreren får man mulighet til å stille flere og mer utdypende spørsmål sammenliknet med en spørreundersøkelse. Informanten har dermed fått mulighet til ytre sine erfaringer, meninger og opplevelser, og gitt meg datamateriale og funn som ikke kan trekkes ut av lukkede svar og tall (Dalland, 2017, s.52). Det har bidratt til å gi meg en bedre forståelse av det jeg har undersøkt og et dypere grunnlag å trekke konklusjoner på.

I kvalitativ forskning er det viktig som forsker å være bevisst over rollen sin. Jeg har tolket og analysert all data, teori og annen informasjon med mitt syns- og utgangspunkt, dermed vil forskningsprosessen og mine funn være avhengige av subjektive vurderinger (Mellin-Olsen & Lindén, 1996, s. 21). Hvordan jeg har forstått det teoretiske grunnlaget og datamateriale som står til grunn for analysen stammer av min forforståelse. Forforståelse innebærer at man som forsker har en forståelse av det som forskes på før selve forskningsprosessen starter (Dalen, 2011, s. 16). Det er derfor vesentlig at valgene jeg har tatt og synspunktene mine blir gjort til syne i oppgaven.

Et valg jeg stod ovenfor var å velge hva som skulle utgjøre mitt utvalg og hvordan jeg skulle implementere observasjon og intervju som metode. I kapittel 1 har jeg presentert tidligere forskning i programmering i matematikkopplæringen. Hvilke funn som har blitt gjort og hvilke metoder som har blitt brukt har i den tidligere forskningen styrt mitt valg av metoder og utvalg. I den tidligere forskningen har både kvantitative og kvalitative tilnærminger blitt foretatt til ulike formål. Flere av studiene er gjort på storskalanivå, men det er likevel flere studier gjort på et mindre utvalg som har forsøkt å kartlegge de matematiske kompetansene elevene sitter igjen med etter å ha arbeidet med programmering i matematikkundervisningen. For eksempel i studiene «Computer programming in the lower secondary classroom: learning mathematics» (Lie et al., 2017) og «Using a Computer Programming Environment and an Interactive Whiteboard to Investigate Some Mathematical Thinking» (Taylor et al., 2010) hadde begge forskergruppene som formål å forske på programmering i matematikkundervisning og

programmerings sammenheng med matematisk læring. I begge disse forskningsprosjektene ble funnene gjort hovedsakelig på bakgrunn av observasjonsdata, opptak og intervjudata. På bakgrunn av studiene valgte jeg hvilke metoder jeg skulle bruke, og det kom tydeligere frem at å utvikle et rammeverk for matematisk kompetanse og læring ville gjøre det lettere for meg å kategorisere, analysere og presentere mine funn.

Jeg valgte altså å supplere ved å benytte meg av to ulike kvalitative metoder, intervju og observasjon. Første innsamling av data skjedde i det første intervjuet. Forskningsintervjuet ga meg innsikt i lærerens tanker og meninger om programmering i matematikkopplæringen, mens observasjonene ga meg ny og dypere innsikt og informasjon over hva lærerens faktisk gjorde og hva som faktisk skjedde i klasserommet i programmeringsundervisningene. Det andre forskningsintervjuet ga meg ny og mer innsikt og informasjon om det jeg observerte. At lærerens synspunkter er spesielt interessante i en situasjon der programmering nettopp er innført, og at ting endres underveis, bidro til at jeg ville intervjue læreren før og etter observasjonsperioden. Å supplere datamaterialet ved bruken av begge intervjuene og metodene kan være mer fordelaktig for kvaliteten av datamaterialet (Furseth & Everett, 2012, s. 129).

3.2 Kvalitative forskningsintervju

Studien tar for seg implementeringen av programmering i en matematikklærers hverdag. Programmering i den norske skolen er et tema som er forsket relativt lite på, og kvalitative forskningsintervju egner seg godt for å få kunnskap om et tema det fins lite forskning på, da metoden fokuserer på dybde fremfor bredde. Ved å bruke kvalitative forskningsintervju ville jeg få innsikt lærerens opplevelser, tanker, meninger og erfaring rundt programmeringsundervisning (Larsen, 2017, s. 98).

Som forsker er det viktig å vurdere forskningsmetoders svakheter og styrker. Eksempelvis trekkes subjektiviteten i kvalitative forsknings intervju frem av mange som en svakhet ved den kvalitative metoden. Likevel trekker Dalen frem subjektiviteten som en styrke fremfor som en svakhet, da metoden subjektiviteten kan gi mer «fyldig og beskrivende informasjon» om temaet fremfor en kvantitativ undersøkelse (Dalen, 2011, s. 13).

3.3 Utarbeidelse av intervjuguidene

Intervjuene var verken lukkede eller åpne, men semistrukturerte. Dalen beskriver semistrukturerte intervjuer som ideelle og en slags mellomting av åpne og lukkede intervju (2011, s. 26). Intervjuene var ikke åpne med tanke på at jeg valgte å fokusere samtalen mot temaer jeg hadde valgt ut på forhånd, og nyansene og fortolkningene jeg var ute etter (Dalen, 2011, s. 26). Jeg sørget for at intervjuene heller ikke var lukkede, for da hadde intervjuene lignet mer på en kvantitativ undersøkelse og mistet fleksibiliteten og åpenheten som lå til grunn i intervjuene. Noe som går imot hensikten til kvalitative intervju som metode.

Som nevnt er min forforståelse av temaet essensielt i hvordan studien gjennomføres. Tidligere har jeg hatt utfordringer med å implementere programmering i min undervisning, og har fått inntrykk av at flere matematikklærere har opplevd det samme. Dette synspunktet spilte en betydelig rolle i utarbeidelsen av intervjueguiden til det første intervjuet med læreren. I det første intervjuet med læreren var jeg ute etter hans tanker rundt implementeringen av programmering i læreplanen i matematikk, hva hans formål med å implementere programmering i sin undervisning er og hvordan han skal oppnå dette. Likevel prøvde jeg å være objektiv i min rolle både som intervjuer og observatør, ved å notere og registrere hendelser og utsagn som motstridde min forforståelse. Å ikke gjøre dette kunne ført til en holistisk feilantagelse ved «å tolke hendelser og utsagn ut fra en feilaktig forforståelse» (Dalen, 2011, s. 99).

Den andre intervjuguiden ble utarbeidet i lys av hva jeg observerte og å følge opp hva som ble sagt i det første intervjuet. For eksempel ville jeg følge opp om læreren følte han oppnådde det han beskrev som formålet med å implementere programmering i sin undervisning i det første intervjuet. Et eksempel på et spørsmål som baserte seg på hva jeg observerte var: «Hvordan fremmet dine undervisningstimer i programmering matematisk læring hos elevene?».

3.4 Gjennomføring av intervjuene

Intervjuene ble gjennomført henholdsvis september 2021 og januar 2022 på lærerens arbeidskontor på skolen der han underviser. Intervjuene varte 30-35 minutter hver. Dette var passe lengde fordi det var nok til at jeg fikk svar på det jeg lurte på, i tillegg til at det var plass for utdypelser fra informanten og oppfølgingsspørsmål fra meg.

Innledningsvis i intervjuene informerte jeg læreren om hvordan intervjuene skulle foregå, hva de skulle dreie seg om og at det ville bli tatt lydopptak. Jeg informerte om at jeg skulle stille han spørsmål tilknyttet hans undervisning og tanker om programmering i matematikkundervisningen. Intervjuguidene ble ikke fulgt punkt til prikke, men fungerte som en rød tråd under intervjuene. For å holde meg til de relevante temaene jeg ville ha innsikt i, brukte jeg intervjuguidene som en slags sjekkliste på om jeg hadde fått stilt spørsmålene jeg ville ha svar på. Under selve intervjuet var jeg fleksibel med tanke på at jeg ga bekreftende tilbakemeldinger og stilte oppfølgingsspørsmål, i tillegg til at intervjuobjektet fikk muligheten til å få repetert spørsmålene og spørre om ting han lurte på (Larsen, 2017, s. 99).

I det andre forskningsintervjuet la jeg spesielt merke til hvor stor rolle min uerfarenhet spilte i det første intervjuet. I det andre intervjuet var jeg som intervjuer i bedre stand til å kommentere og følge opp svarene til intervjuobjektet. Jeg vurderte til enhver tid underveis i intervjuet hvordan jeg kunne skape en samtale som på mest mulig vis var relevant for å belyse problemstillingen og forskningsspørsmålene mine. Dette gjorde jeg ikke i det første intervjuet. Jeg fokuserte i større grad på å stille alle spørsmålene i intervjuguiden slik de var formulert, selv om flere av spørsmålene allerede var besvart. I etterkant av det første intervjuet oppdaget jeg noen mangler i svarene til informanten som lett kunne være oppklart ved å kommentere svarene eller stille oppfølgingsspørsmål. I forkant av første intervju burde jeg utprøvd intervjuguiden på en medstudent eller en annen lærer med tanke på at jeg aldri har intervjuet noen i en forskningskontekst før, og at jeg ville funnet ut av spørsmål jeg ikke hadde trengt å inkludere (Krumsvik, 2019, s. 130).

3.4.1 Lydopptak

Etter informantens tillatelse ble det gjort lydopptak av begge intervjuene. Lydopptakene ble gjort med Universitetet i Oslos (UiO) diktafon-app for mobil. Hovedårsaken til at jeg tok lydopptak var at jeg ville transkribere intervjuene og inkludere mest mulig detaljer, med tanke på at analysen tar utgangspunkt i transkripsjonene (Dalen, 2011, s. 96). Pauser og tonefall blir tatt opp under lydopptak, og gir meg informasjon om svarene. I tillegg ga lydopptak meg muligheten til å vie oppmerksomheten min til samtalen fremfor å notere, noe jeg vurderte som vesentlig med tanke på min bakgrunn som intervjuer.

En svakhet ved lydopptak er at intervjuobjektet kan endre sin atferd når personen er vitende om at det blitt tatt opptak. Jeg vurderte risikoen av at et slikt tilfelle kunne oppstå som lav, og at

lydopptak ville ha liten påvirkning på svarene jeg fikk under intervjuene på bakgrunn av mitt bekjentskap til informanten.

I lydfilene er informantens anonymitet ivaretatt og inneholder ingen sensitiv informasjon. Lydfilene ble lagret konfidensielt og vil slettes når oppgaven er ferdigstilt 1. juni 2022.

3.4.2 Transkribering og oversiktsanalyse av intervjuene

Jeg transkriberte intervjuene ut fra lydopptakene som ble foretatt. I resultatdelen av studien vil jeg presentere relevante utsagn i intervjuene, og legge ved utdrag fra transkriberingen.

Transkribering kan innebære å skrive ned hva som blir sagt i et intervju, og muliggjør å lese intervju (Sollid, 2013, s. 132). Jeg transkriberte så mye som mulig slik at alle spørsmål og utsagn ble notert detaljert i skriftlig form. Dette innebar å transkribere intervjuene ordrett, inkludere pauser og etablere notasjon for ord som ble lagt ekstra trykk på. For meg var det lettere å avgjøre hva i datamaterialet som var relevant for oppgaven, og hva jeg ville inkludere i resultat- og analysedelen når jeg hadde intervjuene i skriftlig form.

Selv om man i lydopptak bevarer hva som ble sagt i et intervju, tonefall og pauser, mister man tilgangen til informantens kroppsspråk og ansiktsuttrykk. Jeg innså viktigheten i å transkribere intervjuene like etter at de var gjennomført, da jeg var ute etter å transkribere intervjuene på en måte som gjenspeilet intervjusituasjonene så nøyaktig som mulig. Dette innebærer å fortolke svarene til informanten i lys av kroppsspråk, ansiktsuttrykk og andre faktorer som påvirker meningen bak svarene til informanten. Noe som lot seg enklest lot seg gjøre rett etter intervjuene var gjennomført, da jeg hadde mest informasjon om intervjuene.

Etter transkriberingen gjennomførte jeg en analyse av intervjudataene, der jeg kategoriserte datamaterialet i lys av studiens formål. En slik analyse ga meg muligheten til å se hvilke sammenhenger som påvirker implementeringen av programmering i matematikk, og hvordan læreren betrakter denne implementeringen.

3.5 Utvalg og kontekst

Før rekrutteringsprosessen var jeg fast bestemt på å velge et lite utvalg for å utnytte de kvalitative forskningsmetodenes styrker. Kvalitative forskningsmetoder er ikke metodene som gir opphav til mye data, men heller dyp data. Med dype data menes rike beskrivelser, utdypelser og refleksjoner som man får ved å intervjue og observere noen. Med min uerfarne

forskningsbakgrunn og mitt valg av tema, som ikke er forsket mye på i den norske skolen, avgjorde jeg det som hensiktsmessig å velge et snevret omfang og utvalg. Ved denne tilnærmingen ville jeg unngå å kontrollere mange faktorer og et dataomfang som ikke er i tråd med oppgavens rammer. I tillegg ville det gi meg muligheten til å gå inn i dybden på utvalget.

Utvalget i mitt forskningsprosjekt består av én matematikklærer i 1T. Det var essensielt å reflektere over rekrutteringsprosessen med tanke på at læreren jeg velger utgjør hele utvalget mitt. Utvalget mitt kan kalles et strategisk utvalg, eller et kriteriebasert utvalg. Det kan også betraktes som et bekvemmelighetsutvalg, men da i betydningen «ikke representativt utvalg» (Frønes & Pettersen, 2021, s. 188-190).

Læreren er skolens avdelingsleder i realfag og underviser kun den ene klassen jeg observerte timene til. Det var det første skoleåret med den nye læreplanen for klassen. Skoleåret 2021-2022 var for majoriteten i klassen det aller første møte med programmering i undervisning. Læreren hadde kun undervist i programmering det foregående skoleåret, 2020-2021, også i matematikk 1T. I forkant av prosjektets start hadde jeg et ønske om å studere hvordan implementeringen av programmering i matematikkopplæringen virker fra bunnen av. At programmering var nytt for elevene og å undervise det var nytt for læreren, var hovedårsaken til at jeg valgte en matematikk 1T-lærer som utvalg.

Mitt bekjentskap til læreren før rekrutteringsprosessen startet var en hovedgrunnene til at jeg valgte han som informant. Det kan være til fordel å slippe å etablere en relasjon mellom forsker og informant fra bunnen av (Sæther & Gleiss, 2021, s. 87).

Tidligere visste jeg at læreren har hatt en betydelig rolle i skolens arbeid med å integrere programmering i skolens matematikklasser. Dermed var jeg bevisst på at han hadde en formening om implementeringen av programmering i læreplanen i matematikk, og gjorde meg spent på hvilke funn jeg ville gjøre meg ved å intervjue han og observere undervisningstimene hans.

En viktig faktor i å rekruttere informanter til intervju er å ivareta informantenes motivasjon til å bli intervjuet (Oppenheim, 1992, s. 82). Denne motivasjonen påvirker intervjudataen og kan dermed påvirke kvaliteten ved et forskningsprosjekt. Lærerens bidrag i skolens arbeid med å implementere en ny læreplan i skolehverdagen, gjorde meg sikrere på lærerens motivasjon og at jeg ville få data som var til hjelp for å besvare oppgavens problemstilling og forskningsspørsmål.

Et informasjonsskriv og samtykkeskjema ble delt ut til samtlige i klassen inkludert læreren, etter at Norsk Samfunnsvitenskapelige Datatjeneste (NSD) hadde godkjent mitt forskningsprosjekt. Før jeg observerte undervisningstimene til klassen introduserte jeg meg selv og ga en kort innføring på hva forskningen min ville dreie seg om. Jeg understrekte at jeg kom til å være en ikke-deltakende observatør av deres matematikktimer med programmering og deres anonymitet vil bli ivaretatt.

Anonymiteten til læreren og elevene i klassen vil i denne studien bli ivaretatt. I studien har jeg referert til læreren som «matematikklæreren», «læreren», «informanten» eller «han».

3.6 Observasjon som forskningsmetode

Intervju er en metode mye brukt i didaktisk forskning, da man får beskrivelser, begrunnelser og refleksjoner rundt læreres undervisningspraksis. Men for å få innsikt og et innblikk i hva som skjer i klasserommet, må man observere (Sæther & Gleiss, 2021, s.206). I mitt prosjekt er jeg ikke bare ute etter lærerens refleksjoner om programmering i matematikkundervisningen, men også hvordan implementeringen av programmering foregår i praksis. Dette vil også gi leseren en mer direkte tilgang til konteksten, og knytte intervjudataen til konteksten (Sæther & Gleiss, 2021, s. 102).

Når man bruker en kvalitativ forskningsmetode som observasjon, er det viktig å være klar over hvordan forskerens forståelse påvirker forskningen. Gjennom observasjon får forskeren direkte tilgang til mennesker og deres handlinger, selv om man ikke kan gi andre direkte tilgang til den virkeligheten man har observert. I likhet med intervju som forskningsmetode, blir forskerens forståelse påvirket av begrepene hen tar utgangspunkt i (Sæther & Gleiss, 2021, s.206). Dermed blir observasjonen påvirket av mine presupposisjoner.

3.7 Gjennomføring av observasjonene

Å observere hva som skjer i et klasserom er komplekst, og det er derfor viktig å strukturere arbeidet med observasjonene. For å redusere kompleksiteten tok jeg hensyn til min observasjonsstruktur, rollen min som observatør, hvor lenge og ofte jeg skal observere og hvordan jeg skal avgrense observasjonene mine (Sæther & Gleiss, 2021, s. 103).

3.7.1 Observasjonsstruktur og avgrensning av observasjon

Min observasjonsstruktur var semistrukturert med tanke på at jeg på forhånd hadde avgrenset hva som skulle observere i lys av mine problemstilling og mine forskningsspørsmål, men likevel hadde en utforskende tilnærming (Kleven, 2014, s. 34; Sæther & Gleiss, 2021, s. 104). I mitt prosjekt ville jeg at observasjonene skulle fokusere på innholdet i programmeringstimene, hva læreren gjør og hans rolle og hvordan læreren legger til rette for læring av matematikk gjennom sin programmeringsundervisning.

3.7.2 Observatørrolle

Observatørollen min var «tilstedeværende, ikke-deltagende observatør» i alle undervisningstimene jeg observerte. I observasjonsperioden følte jeg at elevene i klassen aksepterte min rolle som observatør, og det var sjeldent elevene stilte meg spørsmål. Jeg introduserte meg selv og informerte om at jeg skal sitte bakerst i klasserommet observere deres programmeringstimer i forbindelse med masteroppgaven min, før jeg delte ut samtykkeskjemaet. Læreren til klassen informerte meg også om at han ikke la merke til noen endringer i elevenes atferd når jeg observerte og ikke observerte.

3.7.3 Tidsaspektet i observasjonen

På forhånd hadde jeg bestemt meg for å observere alle klassens timer som involverte programmering fra skolestart i august og siste skoledag i desember. I denne perioden observerte jeg undervisningsøkter på 90 minutter og fagdager i matematikk på fire undervisningstimer.

Læreren informerte meg om at programmering regelmessig ville dukke opp i timene hans. Dette så jeg på som en styrke, da jeg som uerfaren forsker tenkte at flere observasjonsøkter og lengre varighet fører til bedre datamaterialet. Det viste seg senere at programmering ikke dukket opp like ofte i undervisningen til læreren som han først antok og ønsket. Grunnen til at observasjonsperioden tok slutt 15.10, var fordi klassen nesten ikke hadde programmering i timene perioden etter 15.10 og frem til jul. I løpet observasjonsperioden hadde jeg kun observerte fem undervisningsøkter som involverte programmering, og var bekymret over at dette vil gi et dårlig datamateriale. Men disse fem observasjonsøktene viste seg å utgjøre en rimelig mengde med datamateriale, med tanke på at «materialet ikke bare skal produseres, men også analyseres» (Sæther & Gleiss, 2021, s. 108).

3.8 Analyseverktøy og analyseprosessen

Datamaterialet fra observasjonene består av håndskrevne notater gjort i øktene med programmeringsundervisning. Observasjonene vil bli presentert økt for økt, etterfulgt av en analyse av hver økt. Utgangspunktet for analysen er notatene og egne teoretiske sammensatte rammeverk.

Analysen i et prosjekt er avhengig av problemstillingen og forskningsspørsmålene. Fordelen med en kvalitativ metode er at analysen av datamateriale kan gjøres på flere fleksible og ulike måter (Furseth & Everett, s. 144).

Kjerneelementene i den nye læreplanen bruker formulereringer og beskrivelser som ligner de man kan finne i rammeverkene til Kilpatrick et al. (2001) og Niss og Jensen (2002) (udir), jf. kapittel 2. Jeg vurderte det som nødvendig å gjøre meg kjent med begge rammeverkene for å styrke min forståelse og fortolkning av matematisk kompetanse. Ved gjøre meg kjent med begge rammeverkene observerte jeg sammenhengen mellom dem og hvordan de styrker hverandre.

På bakgrunn av dette så jeg det som nødvendig i analysen å introdusere to rammeverk som analyseverktøy. Et rammeverk som vurderer om det er programmering og/eller matematikk i øktene, og et annet rammeverk som vurderer den matematiske kompetansen som kommer til uttrykk i øktene. Det siste rammeverket ble konstruert med tanke på at formålet med programmering er å bruke det til å lære matematikk.

3.8.1 Rammeverk 1 – en induktiv tilnærming

Formålet med det første rammeverket er å identifisere om det hovedsakelig er programmering for programmeringens skyld i lærerens økter, programmering for matematikk eller en blanding av disse. Brukt som analyseverktøy, baserer det første analyseverktøyet seg av to hovedkategorier: *syntaks/teknisk* og *algoritmisk tenkning/matematikk.*

Hovedkategorien *syntaks/teknisk* innebærer at det som har blitt observert, er undervisning i syntaks eller «ren» programmering. Eksempler på dette kan være undervisning som omfatter *print*-kommandoen, hvor man skriver inn en kode eller hvor og hvordan man kjører et program.

Hovedkategorien *algoritmisk tenkning/matematikk* innebærer at det som har blitt observert, er undervisning der programmering blir brukt for at elevene primært skal lære matematikk som er *ny på 1T-nivå*. Med andre ord at de lærer matematikk de åpenbart ikke kan fra før av. Eksempler på dette kan være å arbeide med å programmere abc-formelen for at elevene skal få en dypere forståelse av annengradsfunksjoner, eller å programmere Newtons metode for at elevene skal lære om numeriske metoder i matematikk. Å regne ut enkle regnestykker i Python, eller å lage et program som avgjør om et tall er et partall eller oddetall, vurderes ikke som at elevene lærer matematikk som de åpenbart ikke kan fra før av eller matematikk som er ny på 1T-nivå.

Analysen består av at hver økt tilegnes en kode som beskriver i hvor stor grad økten bærer preg av *syntaks/teknisk*, og en kode som beskriver i hvor stor grad økten bærer preg av *algoritmisk tekning/matematikk*. Analysen inneholder tre koder: kode *0* betegner *ingen observerbar grad*, kode *1* betegner *til en viss grad* og kode *2* betegner i stor grad.

De to hovedkategoriene vil jeg dele inn i *intendert plan* og *realisert plan.* Hver hovedkategori vil få en kode for *intendert plan* og en kode for *realisert plan*. *Intendert plan* tar for seg min fortolkning av i hvor stor grad læreren har intendert å undervise «ren» programmering og programmering for matematisk læring. *Realisert plan* tar for seg om elevene kun lærer programmering i økten, eller matematikk gjennom programmering. *Realisert plan* er elevenes opplevelser i lys av mine fortolkede observasjoner.

For eksempel kan en økt bestå av at læreren har intendert å bruke programmering som et verktøy i å lære et matematisk tema. Dermed kan denne økten tilegnes kode *2* i kategorien *algoritmisk tenkning/matematikk* for *intendert plan*, fordi lærerens intenderte plan er at elevene skal lære det matematiske temaet. Hvis elevene, ut fra min fortolkning, har utviklet matematisk forståelse for temaet slik læreren intenderte, vil økten også tilegnes kode *2* i kategorien *algoritmisk tenkning/matematikk* for *realisert plan*.

I en slik økt kan intensjonen til læreren være at elevene skal utvikle programmeringskunnskap til en viss grad, men at læring av programmering ikke er hovedfokuset. Dermed vil økten tilegnes kode *1* i kategorien *syntaks/teknisk* for *intendert plan*. Likevel kan det hende at elevene, ut fra min fortolkning, blir opphengt kun i programmeringen, selv om det ikke er intendert fra lærerens side. For eksempel kan det hende at hovedfokuset blir på å lære seg syntaks og kommandoer i Python. I dette tilfellet vil økten tilegnes kode *2* i kategorien *syntaks/teknisk* for *realisert plan*.

I en økt vil elevene oppleve ulik grad av mestring og ha ulik utvikling av forståelse i matematikk og programmering. Når jeg har kodet *realisert plan* i de to hovedkategoriene har jeg tatt utgangspunkt i majoriteten av elevene i klassens opplevelser. I klassen jeg observerte la jeg ikke merke til store faglige nivåforskjeller. I samtale med læreren i slutten av oktober, ble jeg informert om at han hadde fått det samme inntrykket. Jeg opplevde i programmeringsøktene at hvis noe var vanskelig for noen, delte de fleste den samme opplevelsen, og det samme hvis noe var enkelt.

3.8.2 Rammeverk 2 – en deduktiv tilnærming

Formålet med det andre rammeverket brukt som analyseverktøy er å identifisere matematisk kompetanse elevene uttrykker i lærerens programmeringstimer. Her vil jeg primært vurdere den matematiske kompetansen som elevene åpenbart ikke hadde fra før av og som er ny på 1T-nivå. Som nevnt i kapittel 2, valgte jeg å utvikle et eget rammeverk bestående av fire kategorier relatert til trådmodellen til Kilpatrick et al. (2001) og Niss og Jensen (2002) sine inndelinger for matematisk kyndighet/kompetanse. De fire kategoriene er oversettelser av fire av kategoriene i trådmodellen til Kilpatrick et al. (2001), men inneholder berikelse hentet fra rammeverket til Niss og Jensen (2002).

De fire kategoriene i rammeverket er:

1) *Konseptuell forståelse*: Denne kategorien tilsvarer conceptual understanding i trådmodellen til Kilpatric et al. (2001), se kapittel 2. Her vektlegger jeg det å forstå og dirigere matematiske idéer og konsepter. Dette innebærer å ta hensyn til når en matematisk idé og sammenheng er viktig og nyttig, og å bruke og forstå ulike matematiske representasjoner, jf. kapittel 2.5.1 (Kilpatrick et al., 2001, s. 118-120).

2) *Beregning*: Denne kategorien er ment til å fange opp kompetanse om matematiske operasjoner og prosedyrer. Her legges det vekt på å håndtere matematiske utregninger, ulike matematiske representasjoner, symbol og formaliteter og hjelpemidler, jf. kapittel 2.5.2 (Kilpatrick et al., 2001, s. 121-123; Niss & Jensen, 2002, s. 56-62).

3) *Strategisk kompetanse*: Her vektlegger jeg det å forstå hvordan man skal løse et problem, altså strategiene og fremgangsmåtene. Dette innebærer å formulere et matematisk problem, og kunne løse det ved å utvikle en eller flere metoder, jf. kapittel

2.5.3 (Kilpatrick et al., 2001, s. 124-129; Niss & Jensen, 2002, s. 52-53).

4) *Resonnering*: Denne kategorien er hovedsakelig ment til å fange opp at elevene vurderer strategier for løse problemer i matematikk. Dette innebærer å vurdere egne og andres matematiske resonnement, jf. kapittel 2.5.4 (Kilpatrick et al., 2001, s. 129-131; Niss & Jensen, 2002, s. 54).

Numerisk bruker det andre rammeverket de samme kodene som det første: kode *0* betegner *ingen observerbar grad*, kode *1* betegner *til en viss grad* og kode *2* betegner i stor grad. I likhet med det første rammeverket, vil analysen bestå av at hver økt tilegnes en kode til hver av de fire trådene, tilegnelsen betegner hvor mye hver av trådene kommer til uttrykk i økten.

For eksempel kan en økt tilegnes kode *2* for *konseptuell forståelse* og *beregning*, kode *1* for *strategisk kompetanse* og *kode 0* for *resonnering*. Det tilsvarer at trådene *konseptuell forståelse* og *beregning* fremkommer i stor grad i økten, *strategisk kompetanse* fremkommer til en viss grad og *resonnering* ikke fremkommer i observerbar grad.

3.9 Kvaliteten på studien

Forskningens reliabilitet og validitet er avgjørende for dens troverdighet. Reliabilitet og validitet er indikatorer på kvalitet i forskning i den forstand at reliabilitet diskuterer forskningens gyldigheter, og validitet diskuterer forskningens troverdighet.

3.9.1 Studiens reliabilitet

Reliabilitet tar for seg gyldigheten i et forskningsprosjekt. Det innebærer hvorvidt man ville fått samme resultater som en selv dersom en annen forsker hadde gjort det på samme måte (Sæther & Gleiss, 2021, s. 202). I kvalitative studier er det viktig å understreke metodiske refleksjoner og avgjørelser gjort i prosjektet, og det gjelder å være åpen og beskrivende om innsamlingsprosessen (Sæther & Gleiss, 2021, s. 204). Teorien og metodiske forbehold som ligger til grunn i studien er gjort i lys av min forståelse og fortolkning (Creswell & Miller, 2000, s. 127). Ettersom fortolkning og forståelse er knyttet til forskeren, kan man diskuterer hvorvidt mine funn ville blitt gjort av andre forskere.

I prosjektet har jeg valgt en kvalitativ tilnærming der forskeren og innsamlingsmetodene kan innvirkning på datamateriale (Solbakken, 2019, s. 44). Data innsamlet gjennom intervju og

observasjon er avhengig av min forståelse av begreper jeg tar utgangspunkt i og mine presupposisjoner (Sæther & Gleiss, 2021, s.206). I intervjuene og observasjonene ville det vært utfordrende for en annen forsker å skape de samme forholdene som i min undersøkelse. En annen forsker ville nok ikke valgt de samme oppfølgingsspørsmålene som jeg mener passer inn, og en annen observatør vil gjøre andre observasjoner (Silverman, 2011, s. 43). Komponentene i analyseverktøyet er sammenflettet og sammenhengende, og kan bidra feilaktig tolkninger og utfordringer når man identifiserer en riktig komponent. På bakgrunn av dette, og valg av metode og utvalg, vil analysen være unik for meg.

I kvalitative studier er det tilnærmet umulig for en forsker å være fullstendig nøytral, noe som vil påvirke datamaterialet og analysen (Larsen, 2017, s. 93). Reliabiliteten i studien styrkes ved å gi leserne en detaljert beskrivelse av metodevalgene, innsamlingsprosessen og konteksten (Johnson, 2013, s. 306). Underveis i metodekapittelet har jeg reflektert over svakheter og styrker i metodene jeg har benyttet i lys av tidligere forskning og problemstillingen jeg har valgt. Tilnærming av utvalg og analyseverktøy ble diskutert med veileder og noen medstudenter, det Creswell og Miller (2000) definerer som «peer debriefing», og har gitt avgjørende bidrag til forskningen min.

3.9.2 Studiens validitet

En studies ytre validitet innebærer om dens funn og konklusjoner kan overføres og ha samfunnsrelevans til en kontekst som strekker seg utover den i studien (Sæther & Gleiss, 2021, s. 207). Ytre validitet henger derfor sammen med reliabilitet. I kvantitative studier vil statistiske generaliseringer gjøre at resultatene kan gjelder for et større utvalg eller andre kontekst, og på denne måten styrke studiens ytre validitet. Betegnelser for generalisering og ytre validitet i kvantitative studier er forskjellige fra betegnelsene i kvalitative studier (Creswell & Miller, 2000, s. 124).

Funnene i mitt prosjekt gjelder kun for prosjektets utvalg og kontekst, og kan ikke bekreftes å gjelde for andre lærere og klasser i matematikk (Furseth & Everett, 2012, s. 149). Funnene er dermed begrenset til mitt utvalg, men kan bidra til idéer for videre forskning.

Selv om generalisering ikke er formålet med kvalitative studier, kan man likevel diskutere ytre validitet i kvalitative studier (Johnson, 2013, s. 305). Alle forskere har ulik bakgrunn og erfaring, og vil derfor tolke og analysere datamateriale ulikt (Johannessen et al., 2010, s. 229). I kvalitative studier diskuterer man derfor en annen type ytre validitet og generalisering, nemlig

analytisk eller teoretisk generalisering. Denne type validitet handler om at man utvikler kategorier eller typologier som kan gjelde for andre settinger enn den som er undersøkt i denne kvalitative studien (Sæther & Gleiss, 2021, s. 207).

Basert på mine intervjuer og observasjoner av læreren og hans klasse, har jeg vært ute etter å utarbeide kategorier eller typologier over programmeringsundervisningen i klassens matematikkopplæring. Men disse kategoriene eller typologiene kan være relevante for andre matematikklasser og lærere, med tanke på at programmering er nytt for de fleste skolene i Norge. Læreren i utvalget mitt refererer ofte til at hans kolleger har følt og opplevd mye av det samme som han når det kommer til programmeringsundervisning, og kan tyde på at funnene kan gjelde for andre enn kun læreren i forskningsprosjektet mitt. Derfor har jeg valgt å beskrive utvalget og konteksten detaljert, slik at man kan vurdere hvem funnene kan gjelder for (Dalen, 2019, s. 96; Johnson, 2013, s. 306).

Å velge et utvalg bestående av flere lærere kunne gitt meg ny og beriket innsikt, og et sammenlikningsgrunnlag. Men på grunn av mitt ønske om å etablere en dypere forståelse av lærerens praksis, oppgavens omfang og unngå omfattende datamengder, unnlot jeg å gjøre dette.

Indre validitet handler om at studien i sin metodiske tilnærming besvarer forskningsspørsmålene (Johnson, 2013, s. 291). I denne studien er dette relativt uproblematisk, siden observasjon av elevers arbeid i matematikktimer og intervju med lærer kan betraktes som naturlige metoder, gitt min problemstilling.

3.10 Forskningsetiske betraktninger

All forskning skal være gjennomført med hensyn til etiske prinsipper, og en forsker skal alltid utøvde etisk dømmekraft (Befring, 2016, s. 28-32). Før jeg kunne starte datainnsamlingen, høsten 2020, måtte jeg fylle ut et meldeskjema som jeg delte med min veileder og sendte det inn til NSD. Søknaden ble godkjent noen dager senere, 16.08.2021, og NSD informerte om at «prosjektet vil være i samsvar med personvernlovgivningen».

I gjennomføringen av prosjektet ble det lagt stor vekt på å oppbevare datamateriale på en sikker og konfidensiell måte, hente inn informert samtykke og anonymisere alle involverte i forskningsprosjektet, jf. kapittel 3.4.1 (Befring, 2016, s. 32).

Elevene og læreren i klassen ble informert om innholdet i prosjektet og deltakernes rettigheter muntlig og skriftlig, i form av informasjonsskrivet som ble delt ut til alle sammen. Det var viktig for meg å få frem at deltakelsen var frivillig, og at det ikke ville få noen konsekvenser å trekke seg fra deltakelsen. Informasjonen ble delt en uke i forkant av første observasjon 12.09.2021. Elevene over 16 år og læreren samtykket selv, mens elevene under 16 år trengte samtykke fra foreldre eller foresatte.

Læreren jeg forsket på har mye erfaring med å bli observert, og har deltatt i forskningsprosjekter til masterstudenter i lærerutdanninger tidligere. Dette gjorde meg trygg på at læreren ikke ville trekke seg på grunn av usikkerheter knyttet til meg som forsker, og at han visste hva han samtykket for.

Intervjuene ble tatt opp med UiOs diktafon-app, som krypterer dataene på telefonen og sender lydfilene til UiOs *Nettskjema*. Transkripsjonen ble gjort så detaljert som mulig for å sikre kvalitet og unngå misforståelser (Universitetet i Oslo, 2021).

4 Resultater

I dette kapittelet presenteres resultatene fra observasjonene og intervjuene, og analyse av dette materialet. Datamaterialet består av to intervjuer med læreren og observasjonsnotater av programmeringsundervisningene. Analysedelen består av to deler.

I første del vil jeg presentere datamaterialet fra observasjonen. Øktene er omtalt i hvert sitt underkapittel. Hvert underkapittel vil bestå av en oppsummering fra hver økt, og det mest essensielle fra notatene. Etter oppsummeringen vil analysen av observasjonen til økten presenteres. Analysen presenteres i form av to tabeller for hver økt som er laget ut ifra *Rammeverk 1* og *Rammeverk* 2. Den første tabellen skal gi en oversikt over om det er programmering og/eller matematikk i øktene, og den andre tabellen skal gi en oversikt over den matematiske kompetansen som kommer til uttrykk i øktene. Kriteriene for disse vurderingene er beskrevet i kapittelet om metode, jf. kapittel 3.8.

I andre del vil jeg presentere datamaterialet fra intervjuene. Jeg har kategorisert dataene fra begge intervjuene i fire overordnede temaer på bakgrunn transkripsjonene og kontekstene i intervjuene. Å trekke frem noen temaer fra intervjuene kan bidra til å se hvilke faktorer som spiller inn på implementeringen av programmering i matematikkundervisningen. De tre temaene er: *lærerens formål med å implementere programmering i undervisningen*, *programmering for matematikk i praksis* og *problemstillinger knyttet til prioritering og tidsbruk*. Hvert underkapittel vil bestå av en beskrivelse av hvert intervjutema, samt relevante uttalelser, sitater og kommentarer av læreren.

4.1 Observasjon

I dette delkapittelet presenteres resultatene fra observasjonene av programmeringsøktene. Fokuset vil være på hvilket fagstoff som har blitt gjennomgått, aktiviteter og hendelser i undervisningstimene.

4.1.1 Økt 1

Sammendrag av økta

Første økt var en fagdag satt av til programmering, og varte i fire undervisningstimer.

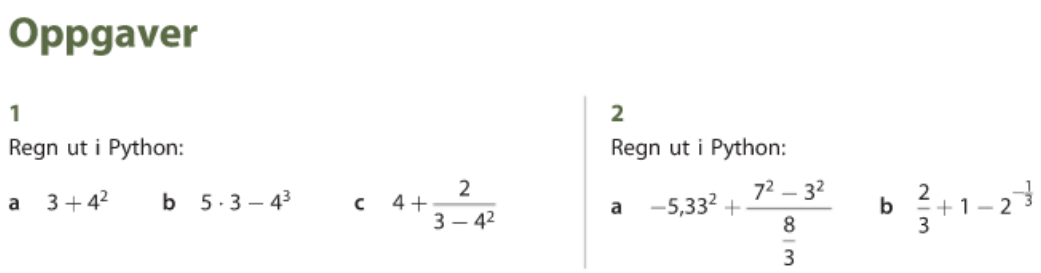
Det var elevenes første økt der programmering var involvert. Læreren begynte timen med å introdusere planen for dagen og at programmering var på agendaen. De første 15 minuttene gikk hovedsakelig ut på introduksjonen til timen og at elevene tok frem sin PC. Resten av første time fikk elevene til å laste ned programmet «Spyder», som er programmet klassen bruker for å kode i Python. Etter stor pågang om hjelp til å laste ned Spyder, fikk samtlige til slutt åpnet programmet på sin PC.

Læreren starter andre time med å snakke om hvordan programmering skal involveres i matematikkundervisningen og i 1T-faget, og at de kommer til å møte på programmering regelmessig i matematikktimene. Læreren forteller elevene at programmering skal integreres i deres matematikkopplæring, og at formålet er at det skal brukes som et nyttig verktøy til å lære matematikk. Videre snakker han om nyttighetsverdien til programmering. Han nevner at man finner programmering i praktisk talt alle realfagsstudier, og at et annet formål med programmering er dens studieforberedende komponent.

Deretter snakker læreren generelt om hva programmering er og kobler dette til algoritmisk tenkning. Som eksempler på algoritmer trekker han frem at å legge sammen to tall er en algoritme, og å regne ut areal for ulike figurer. Han understreker at programmering er viktig for å håndtere store data, løse likninger og modellering, og nevner at det dukker opp over alt.

Videre følger en forklaring av hvordan oppsettet i Spyder og programmeringsspråket Python fungerer. Læreren viser til kodeeditoren og konsollen gjennom et eksempel på prosjektoren. Eksemplet er et program som skriver ut teksten «2+3» og et program skriver ut summen 2+3. Læreren forklarer at programmet leses linje for linje i kodeeditoren, og at når programmet kjøres dukker resultatet opp i konsollen. Han forklarer hva som skjer i koden linje for linje, print-kommandoen, syntaks og forskjellen mellom tall og tekst i Python. Deretter lager han programmer som introduserer hvordan man regner ut potenser, multipliserer og dividerer større uttrykk, før han ber elevene jobbe med oppgaver i boka og skrive sine første programmer.

Læreren veileder og hjelper elevene med oppgavene i Figur 4.1 Det blir fort mange hender i været og læreren ber elevene kikke i boka og spørre læringspartner.



Oppgaver

1
Regn ut i Python:

a $3+4^2$ **b** $5\cdot 3-4^3$ **c** $4+\dfrac{2}{3-4^2}$

2
Regn ut i Python:

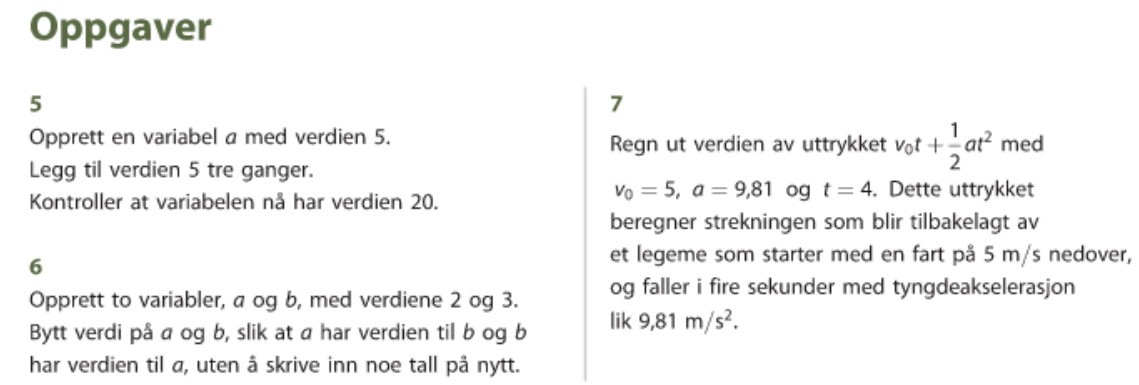
a $-5{,}33^2+\dfrac{7^2-3^2}{\frac{8}{3}}$ **b** $\dfrac{2}{3}+1-2^{-\frac{1}{3}}$

Figur 4.1 – De første programmeringsoppgavene som elevene fikk i Økt 1. Figur hentet fra programmeringsoppgavene i læreboka Mønster 1T.

Fra figur 4.1 ser vi at oppgave 2, tatt fra klassens lærebok i matematikk, Mønster 1T, bruker fortegnet «minus» feil (se Figur 4.1). Liten avstand fra minustegnet til 5,33 betyr at dette er et fortegn, og ikke et regnetegn. Ergo står det her $(-5{,}33)^2$, og det er ikke det de vil frem til. En matematiker ville her typisk skrevet $-(5{,}33)^2$ for å unngå usikkerhet. Denne oppgaven fokuserer på syntaks i Python sammenliknet med syntaks i matematikk.

Noen få elever klarte oppgavene uten hjelp. Av de som trengte hjelp var det mest typiske de slet med dette: å kjøre programmet, punktum istedenfor komma, skrive potenser i Python, hvor resultatet kommer opp, mangel på parentes og et mellomrom for mye, samt andre syntaksfeil. Avslutningsvis i andre time holdt mange av elevene på med andre ting, mens noen fortsatt programmerte.

Innledningsvis i tredje time gjennomgikk læreren oppgavene, og elevene fulgte med på projektoren og kopierte kodene på maskinene sine. Læreren presiserte regnerekkefølger og å regne i print-kommandoen. Læreren introduserte deretter delbarhet, før han gikk videre til variabler. Her viste han eksempler på hvordan man kunne tilegne variabler verdier og oppdatere variabler, samt addere og multiplisere dem. Elevene fikk resten av tredje time og fjerde time til å jobbe med oppgavene i Figur 4.2.



Oppgaver

5
Opprett en variabel *a* med verdien 5.
Legg til verdien 5 tre ganger.
Kontroller at variabelen nå har verdien 20.

6
Opprett to variabler, *a* og *b*, med verdiene 2 og 3.
Bytt verdi på *a* og *b*, slik at *a* har verdien til *b* og *b* har verdien til *a*, uten å skrive inn noe tall på nytt.

7
Regn ut verdien av uttrykket $v_0t+\frac{1}{2}at^2$ med $v_0=5$, $a=9{,}81$ og $t=4$. Dette uttrykket beregner strekningen som blir tilbakelagt av et legeme som starter med en fart på 5 m/s nedover, og faller i fire sekunder med tyngdeakselerasjon lik 9,81 m/s^2.

Figur 4.2 – Programmeringsoppgavene i variabler som elevene fikk i Økt 1. Figur hentet fra programmeringsoppgavene i læreboka Mønster 1T.

I arbeidet med oppgavene i Figur 4.2 var det ikke oppgaveteksten eller matematikken elevene slet med å forstå. Det var heller syntaksen i kodene som var problemet. De typiske feilene var: mangel på multiplikasjonstegn, å glemme print-kommandoen, punktum istedenfor komma og hvordan man skulle skrive «v_0» i Python. Mange skjønte heller ikke hvorfor det kun ble skrevet ut et tall i konsollen, og hvor resten av koden ble av. Læreren avslutter fagdagen med å forklare siste oppgave i Figur 4.2. Han forklarer at det eneste som skrives ut er det som står i print-kommandoen, og hvorfor det kun skrives ut et tall i konsollen.

Analyse av økta

Resultatene fra den kvantitative analysen av økt 1 er presentert i tabellene under.

Programmering og matematikk i økt 1		
	Intendert plan	Realisert plan
Syntaks/teknisk	2	2
Algoritmisk tenkning/matematikk	0	0

Tabell 4.1 – Tabell over koder som identifiserer om det er undervisning programmering og/eller matematikk i økt 1. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

Økt 1 var en introduksjon i grunnleggende programmering. Fra tabell 4.1 ser vi at i denne økten var lærerens intensjon at elevene primært skulle lære å programmere, og det var det som ble realisert.

Matematisk kompetanse uttrykt i økt 1	
Konseptuell forståelse	0
Beregning	0
Strategisk kompetanse	0
Resonnering	0

Tabell 4.2 – Tabell over koder som identifiserer den matematiske kompetansen som kommer til uttrykk i økt 1. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

Intensjonen til læreren denne økta var å gi elevene en innføring i programmering, og ikke nødvendigvis å lære dem matematikk som er ny på 1T-nivå. Økta regnes ikke som undervisning der programmering blir brukt for at elevene primært skal lære matematikk som er ny på 1T-

nivå. Derfor ser vi fra tabell 4.2 at elevene ikke uttrykte noen av de fire kategoriene relatert til matematisk kompetanse til å lære matematikk som er ny på 1T-nivå.

4.1.2 Økt 2

Sammendrag av økta

Økta ble gjennomført omtrent to uker etter første økt med programmering, altså økt 1. Denne økten bestod av to undervisningstimer, men det var den siste timen som var satt av til programmering. Elevene uttrykte et behov for repetisjon, og store deler av timen ble satt av til å repetere det de lærte fra første økt med programmering. Læreren repeterte noen grunnleggende kommandoer i Python, litt om kodeeditoren og konsollen, delbarhet og variabler.

Deretter gikk læreren videre til å gjennomgå avrunding og formatering i utskrift i Python. Elevene kopierte koden læreren skrev på prosjektoren, der det ble vist hvordan man kan bruke print-kommandoen til å formatere utskriften og skrive ut både tekst og variabler til skjerm. Videre gjennomgikk han datatypene streng, heltall og flyttall. Han forklarte kodene til disse, hvorfor man må skille mellom dem og når hver av dem er hensiktsmessig å bruke.

Avslutningsvis arbeidet elevene med oppgaver og så på programmeringsvideoer i fellesskap i henhold til temaene som ble gjennomgått.

Analyse av økta

Resultatene fra den kvantitative analysen av økt 2 er presentert i tabellene under.

Programmering og matematikk i økt 2		
	Intendert plan	Realisert plan
Syntaks/teknisk	2	2
Algoritmisk tenkning/matematikk	0	0

Tabell 4.3 – Tabell over koder som identifiserer om det er undervisning i programmering og/eller matematikk i økt 2. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

Økt 2 bestod hovedsakelig av en repetisjon av det som ble gjennomgått av programmering i første økt. Fra tabell 4.3 ser vi at formålet med økta var at elevene skulle lære å programmere, og det var også det som ble realisert.

Matematisk kompetanse uttrykt i økt 2	
Konseptuell forståelse	0
Beregning	0
Strategisk kompetanse	0
Resonnering	0

Tabell 4.4 – Tabell over koder som identifiserer den matematiske kompetansen som kommer til uttrykk i økt 2. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

I økt 2 fortsatte elevenes innføring i programmering. I økta var ikke matematikk hovedfokuset, men syntaks og grunnleggende programmering. Med tanke på at økt 2 var en programmeringsøkt, uttrykte ikke elevene noen av de fire kategoriene relatert til matematisk kompetanse for å lære matematikk som er ny på 1T-nivå. I likhet med økt 1 regnes ikke økta som undervisning der programmering blir brukt for at elevene primært skal lære matematikk som er ny på 1T-nivå. Fra tabell 4.2 ser vi at elevene ikke uttrykte noen av de fire kategoriene relatert til matematisk kompetanse for å lære matematikk som er ny på 1T-nivå.

4.1.3 Økt 3

Sammendrag av økta

Økta ble gjennomført en uke etter økt 2. Denne økta bestod av to undervisningstimer satt av til vekstfart og programmering. Læreren startet timen med å gjennomgå vekstfaktor på tavla. Deretter gjennomgikk han innhenting av opplysninger fra brukeren og vilkår i programmering, altså if/elif/else-tester. Læreren gjorde to oppgaver på projektoren der kodene inneholdt if-tester input-kommandoen.

Videre fikk elevene en oppgave. Oppgaven bestod av at de skulle innhente informasjon fra brukeren, og deretter regne ut verdien av et hus etter en viss endring ved hjelp av vekstfaktor. Elevene skulle lage et program som spør brukeren om verdien av huset før en endring, og regner ut verdien i fremtiden eller verdien for tidligere år. For å løse oppgaven må elevene hente ut informasjon om den prosentvise økningen eller reduksjonen, og hvilket årstall man skal regne ut den nye verdien for. De trenger også en if-test for å avgjøre om de skal regne ut verdien i fremtiden eller et tidligere år.

Elevene jobbet for det meste i par. Før timen var slutt var det få som lagde et velfungerende program som regnet ut nye verdier av huset. Selv om flere i klassen virket motiverte av den virkelighetsnære faktoren i oppgaven, var majoriteten av klassen opphengt i

programmeringsbiten av oppgaven. Mange slet med hvordan de skulle hente inn informasjon fra brukeren, og lurte på hvorfor de fikk feilmeldinger når de skrev inn formelen for å regne ut verdien etter endring. Feilene hos de fleste bestod av syntaks og at opplysninger som hentes inn fra bruker ikke blir gjort om til et flyttall.

Flere valgte å jobbe med oppgaver om input og if-tester, fordi de ikke hadde skjønt hvordan man henter inn opplysninger fra brukeren og hva en if-test gjør.

Selv om veldig få fikk en velfungerende kode, var det likevel noen elever som resonnerte seg frem til at programmet må spørre brukeren om verdien for huset, prosentendringen (for å regne ut vekstfaktor) og hvilket årstall man skulle regne ut verdien av huset for. Alt i alt, brukte elevene tiden de hadde på oppgaven for det meste til å lære seg å hente inn opplysninger fra brukeren i Python og kode if-tester, fremfor å diskutere vekstfaktor og matematikken i oppgaven.

Analyse av økta

Resultatene fra den kvantitative analysen av økt 3 er presentert i tabellene under.

Programmering og matematikk i økt 3		
	Intendert plan	Realisert plan
Syntaks/teknisk	2	2
Algoritmisk tenkning/matematikk	2	1

Tabell 4.5 – Tabell over koder som identifiserer om det er undervisning iprogrammering og/eller matematikk i økt 3. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

Formålet med økta var at elevene skulle lære om vekstfart og if-tester. I tabell 4.5 ser vi at det ble observert at elevene hadde et stort fokus på å lære seg syntaks og kommandoer i Python, og til en viss grad utviklet matematisk forståelse for vekstfart med programmering som verktøy, slik læreren intenderte.

Matematisk kompetanse uttrykt i økt 3	
Konseptuell forståelse	1
Beregning	2
Strategisk kompetanse	1
Resonnering	2

Tabell 4.6 – Tabell over koder som identifiserer den matematiske kompetansen som kommer til uttrykk i økt 1. (Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)

I observasjonen av Økt 3 kom kategoriene *beregning og resonnering* i stor grad til uttrykk hos elevene. Kategoriene resonnering og strategisk kompetanse kom til en viss grad til uttrykk. *Resonneringskomponenten* ble i stor observerbar grad fanget opp ved at elevene i økta vurderte strategier for løse problemer i matematikk, både ved å vurdere egne og andres matematiske resonnement. *Beregningskomponenten* ble i likhet med *resonneringskomponenten* observert i stor grad i økt 3. Observasjonene fanget opp at elevene viste og utviklet kompetanse om matematisk operasjoner og prosedyrer. Her ble de utfordret i å håndtere matematiske utregninger, ulike matematiske representasjoner og hjelpemidler. Konseptuell forståelse og strategisk kompetanse er tilegnet kode *1* i økt 3. I økta ble elevene utfordret i å dirigere matematiske idéer og konsepter, og å forstå hvordan man skal løse et problem, altså strategiene og fremgangsmåtene, men viste dette til en viss observerbar grad.

4.1.4 Økt 4

Sammendrag av økta

I den siste økta jeg observerte var temaet løsningsformelen for andregradslikninger og å skrive et program i forbindelse med temaet. Læreren startet økta med en utforskende oppgave der elevene skulle finne ut av hvor mange nullpunkter fire andregradsfunksjoner har, regne ut verdien av $b^2 - 4ac$ til hver av funksjonene, og se sammenhengen mellom antall nullpunkter og verdien av $b^2 - 4ac$.

Elevene ble deretter bedt om å skrive et program som finner nullpunktene til $f(x) = x^2 + 2x + 3$. Programmet skulle skrive ut nullpunktene til funksjonen og fortelle brukeren hvor mange nullpunkter funksjonen har. Det var flere som stod fast. Læreren hintet om at koden må inneholde en if/elif/else-test som tar for seg de tre ulike tilfellene av diskriminantens verdi. Etter hintet var det flere elevgrupper som resonnerte seg frem til at koden må teste om diskriminanten er positiv, negativ eller null, og knyttet dette til antall nullpunkter en annengradsfunksjon kan ha. Mens elevene programmerte var det flere av dem som utviklet en relasjonsforståelse for sammenhengen mellom diskriminantens verdi og antall nullpunkter til andregradsfunksjoner (Skemp, 2006, s. 92).

Likevel slet de fleste med programmeringsbiten selv om de forstod matematikken bak oppgaven. Flere av elevene skrev if-testen med feil syntaks og glemte ofte et

multiplikasjonstegn. Det endte med at læreren løste oppgaven på prosjektoren der han han forklarte koden linje for linje og viste riktig hva som er riktig syntaks. Læreren hadde tenkt at elevene skulle utvide programmet de hadde laget slik at programmet finner nullpunktene til en vilkårlig annengradsfunksjon, men gjennomgikk dette i plenum. I løpet av gjennomgangen uttrykte elevene at de ikke skjønte hvordan input-kommandoen brukes i koden og hvordan man gjør om input til et flyttall. Deler av timen gikk til matematikk, men det var likevel en del undervisning i grunnleggende programmering.

Analyse av økta

Resultatene fra den kvantitative analysen av økt 4 er presentert i tabellene under.

Programmering og matematikk i økt 4		
	Intendert plan	Realisert plan
Syntaks/teknisk	1	2
Algoritmisk tenkning/matematikk	2	2

Tabell 4.7 – Tabell over koder som identifiserer om det er undervisning i programmering og/eller matematikk i økt 4. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

I økt 4 hadde læreren intendert å bruke programmering som et verktøy til å lære sammenhengen mellom antall nullpunkter og diskriminantens verdi i andregradsfunksjoner, uten at læring av programmering er hovedfokuset. Elevene utviklet matematisk forståelse for temaet slik læreren intenderte, men brukte store deler av økten, og litt mer enn intendert til å forstå programmeringsbiten.

Matematisk kompetanse uttrykt i økt 4	
Konseptuell forståelse	2
Beregning	2
Strategisk kompetanse	2
Resonnering	2

Tabell 4.8 – Tabell over koder som identifiserer den matematiske kompetansen som kommer til uttrykk i økt 4. (*Kode 2: I stor observerbar grad. Kode 1: Til en viss grad observerbart. Kode 0: Ikke observert.)*

I økta kom alle de fire kategoriene for matematisk kompetanse i stor observerbar grad til uttrykk hos elevene. I likhet med i økt 3, ble *resonnerings*- og *beregningskomponenten* i stor grad observert i økt 4. Elevene vurderte kontinuerlig strategier for løse problemer i matematikk, samtidig som de håndterte matematiske utregninger, ulike matematiske representasjoner og

hjelpemidler. Sammenlignet med økt 3, kom det tydeligere frem i økt 4 at elevene dirigerte matematiske idéer og konsepter og forstod hvordan man skal løse et problem. Derfor er *konseptuell forståelse* og *strategisk kompetanse* også tilegnet kode *2*.

4.2 Intervju

I dette delkapittelet presenteres datamaterialet fra intervjuene. I presentasjonen vil jeg trekke frem noen overordnede temaer som har spilt inn på implementeringen av programmering i matematikkundervisningen til læreren. Temaene skal jeg nå gå nærmere inn på, og de vil bli presentert med utgangspunkt i transkripsjonene og kontekstene i intervjuene.

4.2.1 Lærerens formål med å implementere programmering i undervisningen

Det overordnede temaet «lærerens formål med å implementere programmering i undervisningen» tar for seg lærerens uttalelser om formålene han har, og hvordan han intenderte å realisere disse formålene. Læreren uttalte seg i begge intervjuene om hva han mener formålet med programmering i matematikkundervisningen og læreplanen er. For han er formålet å bruke programmering som et nyttig verktøy i prosessen av å lære matematikk:

> *«Målet er ikke å fremme programmering som noe essensielt i faget, men synliggjøre nytten og bruken. Det handler om å ufarliggjøre møte med koding. De vil jo kanskje møte på matematisk metode som skal programmeres senere i livet i form av MATLAB og håndtering av store datasett osv. I min undervisning vil jeg fremme programmering i lys av det som står i læreplanen. Jeg vil at elevene først og fremst skal se nytten av programmering når man skal lære matematikk. Dette innebærer at de utvikler grunnleggende programmeringsferdigheter og programmeringskompetanse, før man kan bruke programmering til å øke forståelse i matematikk, utforsking og problemløsing og lære å løse problemer som krever programmering som løsningsstrategi.»*

Læreren trekker frem det er matematikken som er hovedfokuset i faget, og ikke programmeringen. Han trekker frem dette i lys av at man fort kan ende opp med å vurdere elevenes programmeringsferdigheter i timene og vurderingssituasjoner, og ikke deres forståelse i faget. Han legger til at man ikke skal ha for stort fokus på syntaks og grunnleggende programmering i undervisningen, men heller vri oppmerksomheten mot det algoritmiske og matematiske aspektet i programmering.

«I teamet har vi blitt enige om at elevene ikke skal bli eksperter i programmering, men heller bruke ferdige maler, bruke dem og tolke kode, fremfor å gå i dybden på syntaks. Det er jo selvfølgelig en viss syntaks som elevene må kunne. De må vite hvordan programmet fungerer ved å lese linje for linje. De må vite hva det betyr å skrive a=3, og at det er forskjellig fra å sette a==3. Det er noen sånn helt grunnleggende ting. At de skal vite hva en if-setning gjør da, eller hva en løkke gjør. Men det er ikke detaljene i det som gjelder. Ja, det fins mange måter å gjøre det på med programmering, men vi må passe på at vi ikke tester dem på detaljer. Skal du teste dem i det at de vet at man skal ha en løkke som går 10 ganger? Det er viktigere å prøve å vri det mot den algoritmiske delen og matematiske delen av det.»

Slik jeg tolker det, er læreren her inne på prinsippet om å ikke gå for detaljert til verks når det gjelder syntaks, og at elevene ikke trenger å skrive kompilerende programmer. Selv om elevene kanskje må vite hvordan man skal «lage en løkke som går 10 ganger» eller en if/else-setning til for eksempel å finne ut av hvilke tall fra 1-100 som er delbare på tre, retter læreren heller oppmerksomheten mot matematikken, som er delbarhet i dette eksempelet, fremfor det programmeringstekninske/syntaksen. Disse prinsippene ble observert i hans undervisning. Læreren prioriterte bort å gjennomgå detaljert syntaks, da både hans undervisningopplegg i programmering og oppgavene som ble gitt fra boka, hovedsakelig la vekt på å studere, tolke, utvide og reprodusere ferdigskrevne koder. Kodene elevene måtte skrive fra bunnen av, var ofte koder elevene kunne finne lignende eksempler av i boka.

For å realisere formålet til læreren, som er programmering til å lære matematikk, understreket han at det er viktig at elevene har programmering regelmessig. Dette ved at man har drypp av programmering omtrent ukentlig. Han forteller videre at programmering vil bli glemt og miste sin hensikt hvis man ikke integrerer det naturlig og regelmessig i matematikkundervisningen. Men han er likevel usikker på hvordan han vil integrere programmering, og at han vil unngå at programmering havner på sidelinja. Videre diskuterer han utfordringer knyttet til hans kompetanse i programmering og programmeringsundervisning.

«Vi kan ikke bare ha programmering hver fagdag, som tilsvarer hver fjerde uke, som vi gjorde i fjor. Det blir for sjeldent. Det vil ende opp med at man lærer matematikkfaget og at programmering havner litt sånn på siden. Det vil gjøre at man åpenbart mister hensikten med programmering, men det er ganske usikkert for meg hvordan jeg skal integrere programmering i timene mine. Man kan ikke jobbe bare med programmering, det blir for mye for elevene. Man trenger å bruke tid på mattedelen av faget. Vi lærere

lærer fortsatt hvordan vi skal få til integreringen av programmering på en god måte. Man ligger litt i forkant av elevene, men det er det. Jeg føler ikke jeg er sterk nok i programmering, og at det er flere lærere som føler på det med manglende kompetanse i programmering og undervisningen av det. Jeg har kun hatt et lite kurs i programmering. Hva er en pen måte å løse ting ved programmering? Og spesielt hvordan man underviser det. Jeg kan selv tenke hva som er vanskelig, men man har litt mindre empiri å gå på.»

4.2.2 Programmering for matematikk i praksis

I 4.2.1 har jeg forsøkt å kartlegge lærerens hovedformål med å integrere programmering. Det overordnede temaet «programmering for matematikk i praksis» tar for seg lærerens uttalelser og tanker om hvordan realiseringen av disse formålene har gått.

Det andre intervjuet jeg gjennomførte med læreren fant sted etter observasjonsperioden. Intervjuet dreide seg hovedsakelig om hvordan programmeringsundervisningen foreløpig har gått skoleåret 2021/22. Ifølge læreren har hans programmeringsundervisning hjulpet elevene med å lære matematikk i liten grad. Å realisere formålene han hadde og at «elevene skal formulere og løse problemer ved blant annet programmering» har ikke klassen fått til, ifølge han.

«Det vil jeg si at vi ikke har fått til. Altså, vi har gjort innlæringsoppgaver som går på veldig grunnleggende utregninger, og bruke variabler og få til utskrift og sånt. Så nei, vi har ikke formulert problemene selv, de har jo bare løst sånne grunnleggende og lukkede problemer. Vi har ikke noe åpne ting foreløpig. Men vi vet at vi har jobba med det (programmering), og vi er på det nivået vil jeg si nå.»

«Programmeringsundervisningen har ikke hjulpet elevene med å lære matematikk så mye foreløpig. Det er mer riktig å si at programmeringen og matten har vært to parallelle ting foreløpig, istedenfor å integrere programmering som et verktøy til å lære matte. Mye av tiden gikk på å lære dem veldig grunnleggende programmering fordi programmering var helt nytt for dem. Vi fikk ikke begynne med løkker og det å definere funksjoner og sånt, og knyttet dette til temaer i 1T. Da tenkte jeg at det kunne hjelpe dem med å forstå numeriske metoder i matematikk, dette med halveringsmetoden og numerisk derivasjon. Det tenker jeg er noen konkrete ting man kunne gi en økt forståelse til ved programmering. Hva som faktisk skjer og grubler litt.»

Selv om læreren uttrykker at han ikke har vært positiv til sin programmeringsundervisning så langt dette skoleåret, fremsnakker han økt 4. I økt 4 ble programmering brukt til å utvikle en forståelse av sammenhengen mellom antall nullpunkter og verdien av $b^2 - 4ac$ til en andregradsfunksjon. Det de jobbet med var blant if/elif/else-setninger for tilfellet negativt, positivt og 0 under rottegn. Læreren trekker frem at elevene resonnerte og grublet seg frem til sammenhengene, både selvstendig og med læringspartner. Han ser på økta som den mest suksessrike økta med programmering. Dette på grunn av at han følte at hans formål og ønske om at elevene skal bruke programmering som et nyttig verktøy i matematikk, og til å hjelpe dem med å lære matematikk ble oppfylt.

> *«Foreløpig har vi vært så grunnleggende med programmeringsbiten at vi ikke har gjort noe spenstig opplegg. Det mest suksessrike hittil var da vi brukte programmering til å lære elevene om sammenhengen mellom nullpunkter og diskriminanten i abc-formelen. Jeg tror elevene ser nytteverdien å kunne programmere, men fikk inntrykk at nytteverdien av programmering i faget. De tok et program for å løse annengradslikninger med abc-formelen, og så skulle vi endre på koden sånn at man kunne skrive inn koeffisientene selv osv. Elevene måtte jo selv reflektere og gruble for å skjønne matematikken bak det hele. For å skrive en slik kode som de gjorde den økta (økt 4) så må man jo kunne syntaksen, kommandoene og programmeringsbiten til en viss grad. Det tenker jeg ikke er det aller viktigste, men heller matematikken. Å jobbe litt mer med sånne ting når vi programmerer tenker jeg kan være mer nyttig for elevene.»*

4.2.3 Problemstillinger knyttet til prioritering og tidsbruk

I intervjuene uttalte læreren seg om hvorfor han ikke har vært positiv til sin egen programmeringsundervisning så langt dette skoleåret. Uttalelsene omhandlet blant annet bortprioriteringen av programmering og utfordringene ved å undervise det. Bortprioriteringen og utfordringene har ført til han ikke har nådd formålet om å bruke programmering som et verktøy til å lære matematikk slik han ville, og hatt programmering mindre og sjeldnere enn han først antok. Læreren påpeker at noen av grunnene til at man lett kan bortprioritere programmering er: holdningene lærerne har til programmering, manglende kunnskap om gode innganger og undervisningsformer elevene har godt utbytte av i programmering, nivået elevene skal være på i programmering, og at programmering og matematikk har følt ut som «to parallelle ting og ikke samme sak».

«Sett i sammenheng med hvordan skoleåret har utviklet seg så langt, er jeg ikke positiv til hvordan programmeringen har gått. Det er lett å behandle det mest som det første som ryker litt da. Litt på grunn av holdninger blant lærerne også. Fordi det føles ut som to parallelle ting. Vi vil jo at matte og programmering skal være samme sak, det tror jeg nok at vi kan, men der er ikke vi. Da er lett å ta det vekk for å lette litt på ting. Jeg syns vi matematikklærere på skolen har prøvd å diskutere litt ulike oppgaver og innfallsvinkler og sånn, men jeg føler det er flere som usikre på nivået og innganger og sånt. Det er ikke sånn at summen av delene blir bedre når vi jobber sammen, det blir nesten enda mer krevende å samarbeide, fordi mange er usikre.»

I løpet av skoleåret opplevde læreren at matematikk og programmering føltes ut som to parallelle ting, og at å dekke elevenes kunnskapshull i matematikk og det rent matematiske ble en prioritet. Ut fra lærerens uttalelser og det jeg observerte av hans timer virket det som elevene så på programmering og matematikk som to forskjellige ting. Han påpekte at han fikk inntrykk av at elevene som ikke opplevde nok mestring i matematikk, opplevde enda mindre mestring når de programmerte. Ut fra mine fortolkninger av observasjonene var det mange av elevene som ble oppgitt når de fant ut agendaen for dagen var programmering, og det virket som mange følte at de måtte over på et annet tankesett enn når de arbeidet med «vanlig» matematikk.

«Den jobben vi gjorde i starten hvor vi klarte å holde litt sånn trykk oppe på begge deler (matematikk og naturfag), klarte vi ikke etter hvert. En del hadde kunnskapshull fra i fjor allerede. For at det ikke skulle bli alt for mye var det jo det som ble prioritert bort da. Det har jo en pris da. Men vi vet jo at det er der, og det er noe vi må ta tak igjen, men ikke helt fra scratch nå da. Men kanskje være veldig kyniske i programmeringa og hvilke deler av programmeringa vi tenker at de må kunne. Målet var jo at det skulle oppleves som noe integrert, men det var et tydelig skille i starten. Okei, vi begynner med programmering, vi skal lære det grunnleggende før vi skal lære om å bruke det inn i mattefaget. Vi satte av tid til dette i starten før vi følte oss tvunget til å prioritere det bort en stund. Etter hvert så ble det ikke regelmessig i timene. Et sånt bytte fra matte til programmering er ikke bare å ha litt programmering 20 min hver time, det handler om å bytte mind-set. Hvis det er veldig tydelig at elevene trenger mer støtte i et tema, er det vanskelig å si nå skal vi gjøre noe helt annet, hvor nøyaktig de samme elevene, og sikkert flere, syns dette med programmering er tungt også. Så å gå fra noe man føler lav mestring på til noe man føler enda mer lav mestring på føles bare som en feil beslutning. Og da er det sånn at det føles plutselig som det er lenge til sommeren, vi kan få til det

ene og det andre etter hvert. Kanskje det gir enda strengere behov for å ha en realistisk tempoplan som setter av tid til de ulike temaene i pensum og som setter av tid til programmering også. Det er erfaringer som vi som fagteam må diskutere.»

Videre påpekte læreren at han er overrasket over ambisjonsnivået både i læreplanen i matematikk og oppgavene i læreboka Mønster 1T, og at dette kan ha bidratt til at elevene opplever programmering som vanskelig. De «numeriske delkapitlene», som han referer til som delkapitlene i læreboka som tar for seg programmering av numeriske metoder, trekker han frem som de vanskeligste delkapitlene elevene har i matematikk 1T. Han mener at man er for ambisiøs hvis man skal lære elevene å programmere fra bunnen av i tillegg til at de mestrer majoriteten av oppgavene i læreboka. Denne problemstillingen tror og håper han blir bedre om 2-3 år, med tanke på at elevene i senere kull vil ha hatt programmering i matematikk på ungdomskolen.

«Jeg er litt overrasket over ambisjonsnivået både i læreplanen og de oppgavene i læreverket vi bruker har. De numeriske delkapitlene i Mønster er jo de vanskeligste, der måtte til og med jeg tenke meg om, og jeg kan se for meg hvordan elevene kommer til å ha det. De tar for gitt en del ting da, syns jeg. Blant annet at elevene tar til seg programmeringen like lett som matematikken. Jeg føler at det er enda større strekk i kunnskapsnivåene der enn det er i den rene matematikken da, hvis vi man kan kalle det det. At det er noen som er utrygge med det å bruke et program og forholde seg til alle disse begrepene med variabel og syntaks, og skrive i et vindu, og kjøre, og få resultater i et annet vindu, og skjønne den greia der. Det blir for ambisiøst å tro at de skal lære programmering fra Scratch i tillegg til å forstå og løse programmeringsoppgavene i boka. Jeg har en forventning og et håp om at det vil bli bedre om 2-3 år når elevene har fått programmert litt på grunnskolen. Vi har snakka litt om det. Hvordan skal den grunnopplæringa vi gir dem på videregående i programmering se ut? Den må vel etter hvert justeres når nivået blir bedre.»

5 Funn og drøfting

I dette kapittelet vil jeg trekke frem og drøfte studiens hovedfunn gjort på bakgrunn av analysen og uttalelsene presentert i kapittel 4. Hvordan disse funnene besvarer problemstillingen og forskningsspørsmålene mine, vil jeg ta opp i kapittel 6.

5.1 Hovedfunn 1: Full detaljering i syntaks nedprioriteres

I intervjuet sier læreren at syntaktiske detaljer i programmering er noe han har prioritert ned. Dette betyr at elevene ikke arbeider med å taste inn kode i matematikkundervisningen. Avgjørelsen har han tatt i samråd med sine kolleger og på bakgrunn av erfaringer fra skoleåret 2020/21. Da opplevde de at å lære elevene programmering fra bunnen med full detaljering i syntaks gikk på bekostning av å lære matematikk. I likhet med kullet fra 2020/21, er de fleste av skoleårets 1T-elever ferske i programmering. Men dette skoleåret har planen vært å tone ned på innføringen syntaktiske detaljer, og heller kun fokusere på det syntaktiske elevene trenger for å bruke programmering til å lære matematikk. Blant annet sier læreren (jf. kapittel 4.2.1):

> *«I teamet har vi blitt enige om at elevene ikke skal bli eksperter i programmering, men heller bruke ferdige maler, bruke dem og tolke kode, fremfor å gå i dybden på syntaks.»*
>
> *«Ja, det fins mange måter å gjøre det på med programmering, men vi må passe på at vi ikke tester dem på detaljer. Skal du teste dem i det at de vet at man skal ha en løkke som går 10 ganger? Det er viktigere å prøve å vri det mot den algoritmiske delen og matematiske delen av det.»*

Konklusjonen er at læreren for dette årskullet betrakter det som for ambisiøst at elevene jobber frem fullstendig kjørbare programmer i typiske problemstillinger som tas opp.

Som nevnt i kapittel 2, peker litteratur som tar for seg programmering og programmeringsundervisning mot at det er et stort potensiale for å lære matematikk gjennom programmering (Taylor et al., 2010; Lie et al., 2017). Flere av studiene i Forsström og Kaufmann (2018) sin litteraturgjennomgang konkluderte med at elevenes evne til problemløsing og matematisk tenking ble positivt påvirket når de programmerte. Likevel påpeker forskerne at man ikke kan ta for gitt at dette skjer hver gang elevene programmerer, og at den positive sammenhengen mellom programmering og problemløsing er avhengig av

læringsaktiviteter og en rekke andre faktorer (Forsström & Kaufmann, 2018; Misfeldt & Ejsing-Duun, 2016).

Mine funn i denne studien bekrefter dette. I økt 1 og 2 bestod læringsaktivitetene stort sett av at læreren løste oppgaven på prosjektoren der han han forklarte koden linje for linje og viste hva som er riktig syntaks, og at elevene skrev av kodene på sin datamaskin og stilte spørsmål (se kapittel 4.1.1 og 4.1.2). Valget av denne typen læringsaktiviteter var sannsynligvis gjort på bakgrunn av at lærerens intensjon om at elevene primært skulle lære å programmere, og ikke nødvendigvis å lære matematikk som er ny på 1T-nivå i de to første øktene (se tabell 4.1 og 4.3).

I økt 3 og 4 flettet læreren inn «vanlig» matematikk. Selv om læreren intenderte at elevene skulle lære å programmere i disse øktene også, var fokuset på fullstendig detaljering i syntaks betydelig tonet ned, og sammenliknet med økt 1 og 2 ble andre læringsaktiviteter gjennomført (se kapittel 4.1.3 og 4.1.4). I de kvantitative analysene av økt 3 og 4 ser man at elevene lærte matematikk som var ny på 1T-nivå, selv om det var i mindre grad enn læreren intenderte i økt 3 (se tabell 4.5 og 4.7).

I økt 3 introduserte læreren temaet vekstfaktor i matematikk, og innhenting av opplysninger fra brukeren og vilkår i programmering, altså if/elif/else-tester, i programmering (se kapittel 4.1.3). I økta ble programmering brukt som et verktøy til å jobbe med temaet vekstfaktor. Selv om elevene måtte lære seg ny programmering var ikke syntaksen og det programmeringstekniske hovedfokuset, men heller hva de ulike testene gjør og når de er hensiktsmessige å bruke i matematikk. Elevene jobbet gruppevis med en oppgave der de skulle innhente informasjon fra brukeren, og deretter regne ut verdien av et hus etter en viss endring ved hjelp av vekstfaktor og programmering. Den virkelighetsnære faktoren oppgaven hadde i å skrive et program som regner ut verdien til et hus, virket å engasjere elevene. Å koble programmering og matematikk til virkelighetsnære problemstillinger er viktig for å motivere elevene til å programmere, noe læreren klarte i denne økta (Forsström & Kaufmann, 2018, s. 25-26).

Som nevnt i kapittel 2, vil å lære matematikk gjennom programmering, være avhengig av prinsippene som ligger til grunn for lærerens tilnærminger til integreringen av programmering i matematikkundervisningen og didaktiske grep (Forsström & Kaufmann, 2018, s. 25-26).

En tilnærming læreren i denne studien har tatt i bruk er at hans undervisning har et hovedfokus på matematikk fremfor det programmeringstekniske og syntaks (jf. kapittel 4.2). Dette stemmer med mine observasjoner i klasserommet (jf. kapittel 4.1. Observasjonene av øktene peker mot

at både hans undervisningopplegg i programmering og oppgavene som ble gitt fra boka, hovedsakelig la vekt på å studere, tolke, utvide og reprodusere ferdigskrevne koder (se kapittel 4.1). På denne måten unngår man at elevene må skrive kodene fra bunnen av, noe som gir større sjanse for syntaksfeil, samt at hovedfokuset ikke går ut på å finne disse feilene, men å forstå matematikken. Det betyr at innholdet i programmeringsdelingen går mer over til den algoritmiske delen og algoritmisk tenkning.

Allerede i 1T-elevenes første økt med programmering dette skoleåret understreket læreren hvorfor elevene skulle lære seg programmering og programmeringsfunksjonalitet i matematikk. I økta forklarte han elevene programmeringens kobling til algoritmisk tenkning og at formålet med programmering er at det skal brukes som et nyttig verktøy til å lære matematikk (se kapittel 4.1.1). Å få til en vellykket integrering av programmering i matematikkfaget består i stor grad av å få elevene til å se nytteverdien av å programmere og hvordan det kan brukes som et verktøy til å hjelpe dem å forstå matematikken (Johansen, 2020).

I økt 4 kom det tydelig frem hvordan læreren forsøkte å vri programmeringsdelingen mer over til den matematiske delen og algoritmisk tenkning. Analysen av økt 4 peker mot at lærerens intenderte plan var å bruke programmering som et verktøy til å lære sammenhengen mellom antall nullpunkter og diskriminantens verdi i andregradsfunksjoner, uten at læring av syntaks i programmering er hovedfokuset (se tabell 4.7). I økt 4 var planen å ikke gjennomgå ny programmering og at elevene hadde lært programmeringen og syntaksen de trengte den økta i de foregående tre øktene. I økta ble det i stor grad observert at elevene utviklet en relasjonsforståelse for sammenhengen mellom diskriminantens verdi og antall nullpunkter til andregradsfunksjoner (Skemp, 2006, s. 92). Elevene, ut ifra min fortolkning, utviklet matematisk forståelse for temaet slik læreren intenderte, og økten ble tilegnet kode *2* i kategorien *algoritmisk tenkning/matematikk* for *realisert plan*. Dette kan tyde på at elevene så nytteverdien av å programmere og hvordan det kan brukes som et verktøy til å hjelpe dem å forstå sammenhengen mellom antall nullpunkter og diskriminantens verdi i andregradsfunksjoner, som er viktig når man skal integrere matematikk og programmering i undervisningen (Johansen, 2020). Læreren sa i intervjuet at han følte at han fikk vist elevene nytteverdien av programmering i matematikkfaget og hvordan programmering kan brukes til å lære matematikk i denne økta (se kapittel 4.2.2):

> *«Det mest suksessrike hittil var da vi brukte programmering til å lære elevene om sammenhengen mellom nullpunkter og diskriminanten i abc-formelen. Jeg tror elevene*

ser nytteverdien å kunne programmere, men fikk inntrykk at nytteverdien av programmering i faget.»

«Elevene måtte jo selv reflektere og gruble for å skjønne matematikken bak det hele. For å skrive en slik kode som de gjorde den økta (økt 4) så må man jo kunne syntaksen, kommandoene og programmeringsbiten til en viss grad. Det tenker jeg ikke er det aller viktigste, men heller matematikken. Å jobbe litt mer med sånne ting når vi programmerer tenker jeg kan være mer nyttig for elevene.»

Læreren uttrykte at han suksessfullt klarte å vri innholdet i programmeringsdelen av økta mer over til den algoritmiske delen og matematisk tenkning, noe som stemmer overens med mine observasjoner og min analyse av økta. I økta kom alle de fire kategoriene i mitt rammeverk for matematisk kompetanse i stor observerbar grad til uttrykk hos elevene (se tabell 4.8). *Resonnerings-* og *beregningskomponenten* ble i stor grad observert i økt 4 ved at elevene stadig vurderte strategier for løse problemer i matematikk og håndterte ulike matematiske utregninger, matematiske representasjoner og hjelpemidler. Kategoriene *konseptuell forståelse* og *strategisk kompetanse* ble også tilegnet kode *2*. Det ble i stor grad observert at elevene dirigerte matematiske idéer og konsepter og forstod hvordan man skal løse et problem.

Liknende funn gjorde Lie et al. (2018) i deres studie om sammenhengen mellom programmering i Scratch og matematisk læring. I studien inkluderes det eksempler der elevene viste flere av delkompetansene i Niss og Jensen (2002) sitt rammeverk for matematisk kompetanse gjennom programmering, deriblant representasjonskompetansen, resoneringskompetansen og kommunikasjonskompetansen. Å klare å vri innholdet i programmeringsdelen mer over til den algoritmiske delen og matematisk tenkning er nok lettere i Scratch enn i Python, med tanke på at Scratch er blokkbasert og Python er tekstbasert, gitt at elevene tilnærmet lærer programmering fra bunnen. Likevel viser funnene i økt 4 at det er mulig å gjennomføre en undervisningssøkt der programmering brukes til å lære matematikk selv om tekstbasert programmering er nytt for dem.

Det er nemlig denne relasjonen mellom programmering og matematikk, og hvordan man kan bruke det i videre studier og i arbeidslivet som har gjort at programmering har blitt en integrert del av matematikkopplæringen i norsk skole (NOU: 2020:2, s. 43; Utdanningsdirektoratet, 2019a). Å mestre og anvende matematikk krever ifølge Utdanningsdirektoratet at elevene arbeider med kjerneelementene. I læreplanen kommer det frem at algoritmisk tenkning og programmering har en sentral sammenkobling med kjerneelementet *utforsking og*

problemløsing, slik Grover og Pea (2013) også påpeker (Utdanningsdirektoratet, 2019c). Å eksponere elevene for undervisning der de skal utvikle relasjonell forståelse, er essensielt i å gjøre dem til utforskere og problemløsere (Skemp, 2006). I økt 4 klarte læreren å bruke programmering til å gjennomføre undervisning der elevene måtte utvikle relasjonell matematisk forståelse og konsepter ved utforsking og problemløsing.

5.2 Hovedfunn 2: Integrering av programmering og «vanlig» matematikk i matematikkundervisningen er utfordrende

Læreren uttrykker at en ønskelig integrering av programmering i matematikkundervisningen består av at elevene programmerer fullstendig kjørbare programmer i typiske matematiske problemstillinger som tas opp, samt utvikler den algoritmiske og matematiske forståelsen man kan oppnå gjennom programmering (Forsström & Kaufmann, 2018; Johansen, 2020). Ettersom læreren, for dette årskullet, betrakter denne ønskelige integreringen som for ambisiøs, har læreren valgt å nedprioritere full detaljering i syntaks og heller vridd programmeringsdelingen mer over til den algoritmiske delen og algoritmisk tenkning, som diskutert i hovedfunn 1 (jf. 5.1). Å vri programmeringsdelingen mer over til den algoritmiske delen og algoritmisk tenkning har likevel bydd på faglige og praktiske utfordringer.

Noe av utfordringene ligger i at elevene trenger grunnleggende programmeringskunnskaper for å kunne lære matematikk gjennom programmering og se nytten av programmering i faget (Forsström & Kaufmann, 2018). Å kunne bruke programmering som løsningsstrategi i utforsking og problemløsing, krever at man over kommer en slags «terskel» i programmeringsferdigheter og programmeringskompetanse. Dette understreker også læreren i intervjuet (se kapittel 4.2.1):

> «*Jeg vil at elevene først og fremst skal se nytten av programmering når man skal lære matematikk. Dette innebærer at de utvikler grunnleggende programmeringsferdigheter og programmeringskompetanse, før man kan bruke programmering til å øke forståelse i matematikk, utforsking og problemløsing og lære å løse problemer som krever programmering som løsningsstrategi.*»

De kvantitative analysene av øktene jeg observerte peker mot at læreren intenderte å bruke både økt 1, 2 og 3 til å lære elevene grunnleggende programmeringskunnskaper. I økt 1 og 2 var fokuset primært «ren» programmering, mens i økt 3 ble det observerte at fokuset både var «ren»

programmering og algoritmisk tekning/matematikk. I økt 4 ble det observert i stor grad at utviklet elevene matematisk forståelse for temaet slik læreren intenderte. Problematikken rundt dette ligger i at i tre av de fire øktene klassen hadde i programmering første halvåret i 2021/22 ble fokuset primært på grunnleggende programmering, og at det ikke ble observert i stor grad at elevene lærte matematikk som er ny på 1T-nivå før økt 4.

Analysene peker også mot at alle disse fire øktene involverte fokus på programmering i stor observerbar grad, både når læreren intenderte et hovedfokus på programmering og når han ønsket et delvis fokus på programmering. Det kan tyde på at elevene blir opphengt i programmeringsbiten selv når læreren intenderer læring av matematikk, noe som kan komme av at majoriteten av elevene i dette årskullet aldri hadde programmert før. Dette kom frem i økt 3 og 4. I økt 3 intenderte læreren fokus på både læring av matematikk gjennom programmering og læring av grunnleggende programmeringskunnskaper (jf. kapittel 4.1.3). I økta ble det for mange nye kommandoer og setninger i programmering elevene måtte ta hensyn til. Den realiserte planen ble at elevene hadde et stort fokus på å lære seg syntaks og kommandoer i Python, og kun til en viss grad utviklet matematisk forståelse for det matematiske temaet.

I økt 4 ønsket læreren at elevene skulle lære å bruke programmering som et verktøy til å lære sammenhengen mellom antall nullpunkter og diskriminantens verdi i andregradsfunksjoner, uten at læring av programmering er hovedfokuset (jf. kapittel 4.1.4). Planen om å lære sammenhengen mellom antall nullpunkter og diskriminantens ble realisert, men elevene var fremdeles opphengt i å lære seg syntaks og kommandoer i Python, i større grad enn det læreren intenderte.

Å fremme programmering som et verktøy som hjelper elevene å lære matematikk viser seg å være enda vanskeligere når elevene må lære programmering fra bunnen av. Som nevnt ovenfor peker de kvantitative analysene mot at elevene, ut ifra mine fortolkninger, stort sett ikke lærte matematikk som er ny på 1T-nivå slik læreren intenderte utenom i økt 4. Store deler av samtlige økter gikk ut på at elevene lærte seg grunnleggende programmering, noe læreren mener gikk på bekostning av å lære elevene matematikk ved hjelp av programmering (se kapittel 4.2.2):

> *«Programmeringsundervisningen har ikke hjulpet elevene med å lære matematikk så mye foreløpig. Det er mer riktig å si at programmeringen og matten har vært to parallelle ting foreløpig, istedenfor å integrere programmering som et verktøy til å lære matte. Mye av tiden gikk på å lære dem veldig grunnleggende programmering fordi*

programmering var helt nytt for dem. Vi fikk ikke begynne med løkker og det å definere funksjoner og sånt, og knyttet dette til temaer i 1T.»

«Altså, vi har gjort innlæringsoppgaver som går på veldig grunnleggende utregninger, og bruke variabler og få til utskrift og sånt. Så nei, vi har ikke formulert problemene selv, de har jo bare løst sånne grunnleggende og lukkede problemer. Vi har ikke noe åpne ting foreløpig. Men vi vet at vi har jobba med det (programmering), og vi er på det nivået vil jeg si nå.»

Problemet med manglende elevkunnskaper om grunnleggende programmering gjorde det altså vanskelig å oppnå integrasjon mellom programmering og matematikk.

5.3 Hovedfunn 3: Elevenes og lærerens manglende kompetanse i henholdsvis programmering og undervisning i programmering

Programmering er helt nytt for mange av elevene i 1T dette skoleåret. Elevene i denne studien følger for første gang LK20 dette skoleåret. For lærerens del, var skoleåret 2020/21 første gang han underviste i programmering. Læreren forteller at han har manglende kompetanse i programmering, hvordan man underviser det og hvordan man integrerer det i matematikk 1T-opplæringen. Som nevnt, er læreren skolens avdelingsleder og forteller at hans kollegaer opplever det samme rundt programmering. Læreren og flere av hans kollegaer har kun gjennomført et lite kurs i programmering. Dette kommer frem i intervjuet (se kapittel 4.2.1):

«Vi lærere lærer fortsatt hvordan vi skal få til integreringen av programmering på en god måte. Man ligger litt i forkant av elevene, men det er det. Jeg føler ikke jeg er sterk nok i programmering, og at det er flere lærere som føler på det med manglende kompetanse i programmering og undervisningen av det. Jeg har kun hatt et lite kurs i programmering. Hva er en pen måte å løse ting ved programmering? Og spesielt hvordan man underviser det. Jeg kan selv tenke hva som er vanskelig, men man har litt mindre empiri å gå på.»

NOU-rapporten *Hindre for digital verdiskaping* publisert i 2013 understreker den manglende programmeringskompetansen blant Norges befolkning (NOU 2013: 2, s. 10). Dette gjelder også lærere i den norske skole, som har manglende kompetanse innen programmering og hvordan det skal integreres i matematikk (Johansen, 2020; Vogt, 2021). I intervjuet går læreren inn på

hvordan hans manglende kompetanse i programmering og hvordan man underviser i det kan føre til flere utfordringer for elevene. Han forteller at det ikke bare er manglende programmeringskompetanse på skolen, men at det generelt er utilstrekkelig med kursing i programmering for lærere og didaktikk om tilpasning i matematikkfagene.

I det første intervjuet var læreren innstilt på at de skulle ha programmering mer regelmessig enn forrige skoleår. Det foregående skoleåret dukket programmering omtrent hver fagdag (hver fjerde uke), noe læreren mente var for sjeldent og gjorde at programmering mistet sin hensikt i faget (se kapittel 4.2.1):

> *«Vi kan ikke bare ha programmering hver fagdag, som tilsvarer hver fjerde uke, som vi gjorde i fjor. Det blir for sjeldent. Det vil ende opp med at man lærer matematikkfaget og at programmering havner litt sånn på siden. Det vil gjøre at man åpenbart mister hensikten med programmering, men det er ganske usikkert for meg hvordan jeg skal integrere programmering i timene mine.»*

Programmering dukket ikke opp like ofte i undervisningen til læreren som han først antok og ønsket. I løpet av observasjonsperioden hadde jeg kun observerte fem undervisningsøkter som involverte programmering I praksis har det vært vanskelig å integrere programmering som en naturlig del av matematikk 1T-faget, noe som har ført til at han har behandlet programmering som «det første som ryker» og at elevene ikke har hatt programmering regelmessig som han først planla og ønsket. Den manglende kompetansen har ført til en negativ holdning blant læreren og hans kollegaer. (se kapittel 4.2.3):

> *«Sett i sammenheng med hvordan skoleåret har utviklet seg så langt, er jeg ikke positiv til hvordan programmeringen har gått. Det er lett å behandle det mest som det første som ryker litt da. Litt på grunn av holdninger blant lærerne også. Fordi det føles ut som to parallelle ting. Vi vil jo at matte og programmering skal være samme sak, det tror jeg nok at vi kan, men der er ikke vi. Da er lett å ta det vekk for å lette litt på ting. Jeg syns vi matematikklærere på skolen har prøvd å diskutere litt ulike oppgaver og innfallsvinkler og sånn, men jeg føler det er flere som usikre på nivået og innganger og sånt. Det er ikke sånn at summen av delene blir bedre når vi jobber sammen, det blir nesten enda mer krevende å samarbeide, fordi mange er usikre.»*

At klassen ikke har hatt programmering regelmessig dette skoleåret skyldes ikke kun manglende programmeringskompetanse blant lærerne, men også blant elevene. At elevene lærer programmering fra bunnen av, byr på flere utfordringer. Målet er å bruke programmering

som et verktøy i matematikkfaget, men dette krever grunnleggende programmeringskompetanse. Å tilegne seg denne grunnleggende kompetansen er mer tidskrevende når man ikke har programmert før. Læreren beskriver det som problematisk at elevene skal lære seg å bruke programmering på ambisjonsnivået til læreplanen og læreboka, uten å ha hatt programmering tidligere:

> *Det blir for ambisiøst å tro at de skal lære programmering fra scratch i tillegg til å forstå og løse programmeringsoppgavene i boka.*

Læreren opplevde delkapitlene om numeriske metoder, deriblant Newtons metode og numerisk derivasjon, som de vanskeligste i læreboka. I disse delkapitlene har programmering en sentral rolle. Delkapitlene krever at elevene kan grunnleggende programmering og anvende det i matematikk. Han følte at disse delkapitlene og programmering generelt ville være utfordrende og krevende for elevene i klassen, da flere av dem allerede opplever faget som tungt nok og har kunnskapshull i flere temaer. Læreren beskrev det som at han opplevde at elevene gikk fra noe de føler lav mestring i, til noe de føler enda mindre mestring i, når elevene måtte programmere. Ifølge læreren, tar ikke den nye læreplanen i matematikk og læreboka hensyn til kompetansenivået til årets 1T-elever i programmering og det tar tid å lære seg (se kapittel 4.2.3):

> *«Jeg er litt overrasket over ambisjonsnivået både i læreplanen og de oppgavene i læreverket vi bruker har. De numeriske delkapitlene i Mønster er jo de vanskeligste, der måtte til og med jeg tenke meg om, og jeg kan se for meg hvordan elevene kommer til å ha det. De tar for gitt en del ting da, syns jeg. Blant annet at elevene tar til seg programmeringen like lett som matematikken. Jeg føler at det er enda større strekk i kunnskapsnivåene der enn det er i den rene matematikken da, hvis vi man kan kalle det det.*
>
> *«Et sånt bytte fra matte til programmering er ikke bare å ha litt programmering 20 min hver time, det handler om å bytte mind-set. Hvis det er veldig tydelig at elevene trenger mer støtte i et tema, er det vanskelig å si nå skal vi gjøre noe helt annet, hvor nøyaktig de samme elevene, og sikkert flere, syns dette med programmering er tungt også. Så å gå fra noe man føler lav mestring på til noe man føler enda mer lav mestring på føles bare som en feil beslutning.»*

Programmering føltes dermed som en ekstra ting elevene måtte forholde seg til i faget, fremfor et hjelpemiddel eller verktøy som kan hjelpe dem i faget. Elevene tok ikke til seg programmering så lett som læreplanen og læreboka kanskje legger opp til. Læreren la merke til

at det var vanskeligere for elevene lære seg programmering enn den «rene» matematikken. At programmering var vanskelig for elevene, tidskrevende og vanskelig for læreren å integrere i faget kan være grunnene til at han behandlet programmering som det første som går bort.

Læreren stiller seg positivt til at situasjonen blir bedre når elevene har hatt programmering på ungdomsskolen (se kapittel 4.2.3):

> *«Jeg har en forventning og et håp om at det vil bli bedre om 2-3 år når elevene har fått programmert litt på grunnskolen. Vi har snakka litt om det. Hvordan skal den grunnopplæringa vi gir dem på videregående i programmering se ut? Den må vel etter hvert justeres når nivået blir bedre.»*

Et skoleår der programmering har vært nytt både for elevene og læreren, har betydd at det har vært en fare for et fokus på programmering i matematikkundervisningen av teknisk art (Johansen, 2020). Selv om læreren valgte å nedprioritere undervisning av den tekniske arten i fordel for nytteverdien i matematikk, gikk det allikevel mye tid på det tekniske og syntaktiske. Som nevnt tidligere, ser man for eksempel i økt 3 at det ble realisert et større fokus på den tekniske arten ved programmering enn intendert (jf. tabell 4.5). At det tar lengre tid en intendert å lære elevene det syntaktiske og tekniske er potensielt en grunn til at programmering har blitt behandlet som det første som nedprioriteres. Å få elevene til å se nytteverdien av programmering i matematikk krever at de utvikler en relasjonell forståelse (Grover & Pea, 2013). Samtidig understreker Skemp (2006, s. 93) at det ofte er vanskeligere og mer tidskrevende for elevene å oppnå relasjonell forståelse. Om noen år vil elevene ha hatt programmering på ungdomsskolen, samt bedre forkunnskaper og et sterkere grunnlag i programmering når de starter på videregående. Dette vil føre til at mer tid av matematikkundervisning kan gå på å synliggjøre nyttigheten av programmering i matematikk fremfor den tekniske arten.

6 Konklusjon og videre forskning

6.1 Konklusjon

Problemstillingen i denne studien er (se kapittel 1.2):

> *Hvordan arbeides det med programmering i matematikk i et 1T-klasserom etter LK20?*

For å besvare problemstillingen har jeg tatt utgangspunkt i to forskningsspørsmål:

1) *Hvilke praktiske utfordringer er det med arbeidet med programmering i matematikk 1T?*
2) *I hvilken grad er det problematisk at årets elever har hatt lite eller ingen programmering på ungdomstrinnet?*

Når det gjelder problemstillingen, så besvares den primært av hovedfunn 1 og 2. Disse viser at læreren for dette årskullet betrakter det som for ambisiøst at elevene fra bunnen av koder fullstendig kjørbare programmer i typiske problemstillinger som tas opp. Læreren har i sin matematikk undervisning hatt et hovedfokus på matematikk fremfor det programmeringstekniske og syntaks. Det betyr at han har forsøkt å vri innholdet i programmeringsdelingen mer over til den algoritmiske delen og matematikk, og at full detaljering i syntaks nedprioriteres. Observasjonene av øktene hans peker mot han la vekt på at elevene skulle studere, tolke, utvide og reprodusere ferdigskrevne kode. Denne tilnærmingen har læreren valgt for å synliggjøre nyttigheten av programmering i matematikkfaget for elevene og bruke programmering til å få elevene til å forstå faget bedre.

Å få til dette har vist seg å være vanskelig i praksis, når elevene samtidig må lære programmering fra bunnen av. Ut ifra mine fortolkninger, lærte elevene stort sett ikke matematikk som er ny på 1T-nivå slik læreren intenderte utenom i én av de fire øktene jeg observerte. Læreren konkluderer med at programmering og matematikk har vært to parallelle ting dette skoleåret.

Funnene i denne studien viser at det er mulig å integrere programmering i matematikkopplæringen, men også at arbeidet med å integrere programmering og «vanlig» matematikk i 1T-faget har vist seg å være utfordrende. For å besvare første forskningsspørsmål har jeg identifisert praktiske utfordringer som følger med integreringen av programmering i matematikkopplæringen dette skoleåret. Funnene peker mot at elevene trenger grunnleggende

programmeringskunnskaper for å kunne lære matematikk gjennom programmering og se nytten av programmering i faget. Læreren måtte arbeide med at elevene overkom en slags «terskel» i programmeringsferdigheter og programmeringskompetanse før de tilnærmer seg matematiske problemer. Å overkomme denne «terskelen» førte til at mer tid av matematikkundervisningen med programmering gikk mer over til den tekniske arten og på bekostning av matematikken, slik Kaufmann og Stenseth så faren for (Johansen, 2020). Selv om læreren intenderte å nedprioritere det tekniske og syntaktiske ved programmeringen ble elevene opphengt i nettopp dette. Manglende elevkunnskaper om grunnleggende programmering gjorde det vanskelig å oppnå integrasjon mellom programmering og matematikk.

Når det gjelder forskningsspørsmål 2, som er *i hvilken grad er det problematisk at årets elever har hatt lite eller ingen programmering på ungdomstrinnet?,* er det korte svaret *i stor grad*, og svaret utdypes primært av hovedfunn 3. Hovedfunn 1 og 2 kan oppfattes som konsekvenser av hovedfunn 3. Det er nemlig den manglende elev- og lærerkompetansen i henholdsvis programmering og programmeringsundervisning som har ført til integreringen av programmering og matematikk har vært utfordrende, samt at full detaljering i syntaks ble nedprioritert.

Læreren la merke til og uttrykte i intervjuet at årets elever i 1T har opplevd programmering som en ting til elevene må lære i faget, istedenfor et verktøy som hjelper dem å forstå faget bedre. Han erfarte at ambisjonsnivået til læreplanen og læreboka og nivået til elevene var forskjellige. Flere av elevene opplevde programmering som vanskeligere og mer utfordrende enn den rene matematikken. Problemstillingene rundt tidsbruken og den manglende elev- og lærerkompetansen i henholdsvis programmering og programmeringsundervisning, gjorde at programmering ble behandlet som det første som «ryker» i faget dette skoleåret.

6.2 Studiens begrensninger

I denne studien har jeg undersøkt hvordan det arbeides med implementeringen av programmering i én 1T-klasse. Ettersom funnene mine er gjort på bakgrunn av mine fortolkninger og min forståelse, er de kun gjelende for prosjektets utvalg og kontekst, og kan ikke påstås å gjelde for andre lærere og klasser i matematikk. I tillegg kan det diskuteres hvorvidt funnene ville blitt annerledes om studien var blitt gjort av andre forskere.

Selv om funnene mine er begrenset til mitt utvalg, kan de bidra til idéer for videre forskning. Formålet med denne studien har ikke vært å etablere funn som gjelder for alle matematikklærere. Mitt formål har primært vært å belyse problemstillinger knyttet til arbeidet

med implementeringen av programmering i matematikkundervisning, som kan gjelde for andre settinger enn utvalget i denne studien.

I min studie har jeg intervjuet én lærer, og observert han og hans klasse. Likevel er det læreren og hans tilnærming til innføringen av programmering i matematikkopplæringen, som har vært hovedfokuset. Å inkludere metodiske tilnærmeringer som tar for seg elevenes og flere læreres perspektiv til innføringen av programmering, ville gitt beriket grunnlag til å besvare spørsmålene diskutert i denne studien.

Denne studien er også begrenset til dagens situasjon og kontekst. I dag er programmering i matematikkopplæringen nytt for de fleste lærere og elevene. Studien er gjennomført i midten av en prosess med innføring av en ny læreplan. Om noen år vil lærere ha en beriket innsikt i hvordan den nye læreplanen og dens verdier og prinsipper skal praktiseres i klasserommet. Elevene som starter på videregående skole, vil ha bedre forkunnskaper og et sterkere grunnlag i programmering enn elevene skoleåret 2021/22. Problemstillingene jeg har drøftet knyttet til elevenes og lærerens manglende kompetanse i henholdsvis programmering og undervisning i programmering, vil antakelig være betydelig tonet ned, og i fremtiden vil nok nye problemstillinger diskuteres.

6.3 Forslag til videre forskning

I fremtiden ville det vært interessant å velge et utvalg bestående av flere lærere, og dermed danne et sammenlikningsgrunnlag. I større undersøkelser er det viktig å inkludere elevenes perspektiver, erfaringer og opplevelser rundt implementeringen av programmering. Et større og mer variert utvalg kan bidra til mer representative funn. I skrivende stund er vi bare i startfasen av en prosess med innføring av ny læreplan. Dermed er det nødvendig å observere hvordan elever og lærere forholder seg til denne innføringen.

I videre forskning kan man gjøre undersøkelsene større ved å inkludere flere skoler, studieprogrammer, lærere med mye og liten erfaring i programmering. I dag arbeider skolene med deres tilnærminger til hvordan de skal innføre av den nye læreplanen på deres skole. Det utarbeides lokale læreplaner som tar for seg blant annet hvordan de planlegger å integrere programmering i matematikkundervisningen på deres skole. Å gjennomføre en større studie som ser på ulike skoler sin tilnærming, kan gi innsikt i hva som fungerer og ikke fungerer.

Det vil være interessant å følge opp situasjonen og utviklingen om noen år. Om noen år vil nok premissene med hensyn til vurderingsordning være avklart. Elevene vil ha hatt programmering på ungdomstrinnet i henhold til LK20, og lærere vil ha fått mer erfaring med hva som fungerer og ikke. Da vil nok undervisningen, holdningene og problemstillingene og utfordringene være annerledes enn det de har vært i denne studien.

I videre forskning kan det være aktuelt å undersøke elever med ulik programmeringsbakgrunn sin tilnærming til programmering i 1T. På arbeidsplasser brukes ulike programmeringsspråk, og det samme gjelder på ungdomstrinnet i norsk skole. På ungdomsskolen brukes blant annet det blokkbaserte programmeringsspråket Scratch, det tekstbaserte Python, mens noen kombinerer disse eller bruker kanskje andre språk. Hvordan vil elever med Scratch fra ungdomstrinnet klare programmering i 1T sammenliknet med de som har hatt Python? Dette er et eksempel på en problemstilling som vil være interessant å undersøke.

Litteraturliste

Befring, E. (2015). Forskningsetikk. I E. Befring (Red.), *Forskningsmetoder i utdanningsvitenskap* (s. 28-35). Cappelen Damm Akademisk.

Bocconi, S., Chioccariello, A. & Earp, J. (2018*). The Nordic approach to introducing Computational Thinking and programming in compulsory education.* https://doi.org/10.17471/54007

Christoffersen, L. & Johannessen, A. (2012). *Forskningsmetode for lærerutdanningene* (2. utgave). Oslo: Abstrakt forlag.

Creswell, J.W. & Miller, D.L. (2000). Determining validity in qualitative inquiry. *Theory into practice: Getting good qualitative data to improve educational practice, 39*(3), 124- 130. https://doi.org/10.1207/s15430421tip3903_2

Dalen, M. (2011). *Intervju som forskningsmetode – en kvalitativ tilnærming* (2 utg.). Universitetsforlaget.

Daus, S., Per Olaf Aamodt, P. O. & Tømte, C. E. (2019). *Profesjonsfaglig digital kompetanse i lærerutdanningene. Undersøkelse av tilstand, holdninger og ferdigheter ved fem grunnskolelærerutdanninger*. NIFU: 2019: 13. https://nifu.brage.unit.no/nifu-xmlui/bitstream/handle/11250/2602702/NIFU-rapport2019-13rev.pdf?sequence=6ogisAllowed=y

Dalland, O. (2017). *Metode og oppgaveskriving* (6. utg.). Oslo: Gyldendal Akademisk.

Forsström, S. E. & Kaufmann O. T. (2018). A Literature Review Exploring the use of Programming in Mathematics Education in International Journal of Learning, *Teaching and Educational Research*, *17* (12), 18-32. https://doi.org/10.26803/ijlter.17.12.2

Frønes, T. S. & Pettersen, A (2021). Spørreundersøkelser i utdanningsforskning. I C. Dalland og E. Andersson-Bakken (Red.), *Metoder i klasseromsforskning: Forskningsdesign, metoder og analyser* (s. 167-205). Universitetsforlaget.

Furseth, I. & Everett, E. (2012). Kap. 9: Kunsten å holde stø kurs. Å lage en god analyse. I *Masteroppgaven. Hvordan begynne og fullføre* (s. 145-161). Universitetsforlaget.

Grover, S. & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Johannessen, A., Tufte, P. A. & Christoffersen, L. (2010). *Introduksjon til samfunnsvitenskapelig metode* (4. utg.). Oslo: Abstrakt.

Johansen, A.-K. (2020, 11. juli). *Programmering vil bli en utfordring for lærere.* Forskning. fra https://forskning.no/a/1711838

Johnson, B. R. (2013). Valditity of Research Results in Quantitative, Qualitative and Mixed Research. I B. R. Johnson & L. Christensen (Red.), *Educational Research: Quantitative, Qualitative, and Mixed Approaches* (s. 277-316). Los Angeles: Sage.

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

Kleven, T. A. (2014). Data og datainnsamlingsmetoder. I T. A. Kleven (Red.), *Innføring i pedagogisk forskningsmetode* (s. 27-47). Bergen: Fagbokforlaget.

Krumsvik, R. J. (2019). *Kvalitativ metode i lærarutdanninga* (1. utg.). Bergen: Fagbokforlaget.

Larsen, A. K. (2017). Om samfunnsvitenskapelig metode. I A. K. Larsen (Red.), *En enklere metode. Veiledning i samfunnsvitenskapelig metode* (2. utg., s. 17-31). Bergen: Fagbokforlaget.

LEGO. (2020). *Mindstrorms.* https://www.lego.com/nb-no/themes/mindstorms/about

Lie, J., Hauge, I. O. & Meaney, T. J. (2017). Computer programming in the lower secondary classroom: learning mathematics. *Italian journal of educational technology, 25*(2), 27-35.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior*, *41*, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Mellin-Olsen, S. (1981). Instrumentalism as an educational concept. *Educational Studies in Mathematics*, *12*(3), 351-367.

Mellin-Olsen, S. & Lindén, N. (1996). *Samtalen som forskningsmetode : Tekster om kvalitiv [i.e. kvalitativ] forskningsmetode som del av pedagogisk virksomhet*. Landås: Caspar forlag.

Misfeldt, M. & Ejsing-Duun, S. (2015). Learning mathematics through programming: An instrumental approach to potentials and pitfalls. In K. Krainer & N. Vondrová, (Eds.) Proceedings of the 9th Conference of the European Society for Research in Mathematics Education (CERME 9) (s. 2524–2530).

Massachusetts Institute of Technology Media Lab. (2022). *Scratch—Imagine, Program, Share*. Scratch. https://scratch.mit.edu/

Niss, M. & Jensen, T. H. (2002). *Kompetencer og matematiklæring: Idéer og inspiration til udvikling af matematikundervisning i Danmark (Vol. 18).* Copenhagen, DK: Undervisningsministeriet. http://www.gymnasieforskning.dk/wp-content/uploads/2013/10/Kompentecer-og-matematikl%C3%A6ring1.pdf

NOU: 2013: 2. (2013). *Hindre for digital verdiskaping.* Fornyings-, administrasjons- og kirkedepartementet. https://www.regjeringen.no/contentassets/e2f0d5676e144305967f21011b715c16/no/pdfs/nou201320130002000dddpdfs.pdf

Papert, S. (1993a). *Mindstorms: Children, computers and powerful Ideas* (2. utg.). New York: BasicBooks.

Papert, S. (1993b). *The children's machine: rethinking school in the age of the computer*. New York: BasicBooks

Pea, R. D. & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, *2*(2), 137–168. https://doi.org/10.1016/0732-118X(84)90018-7

Patton, M. Q. (1999). Enhancing the quality and credibility of qualitative analysis. *Health Services Research, 34*(5), 1189-1208. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1089059/

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

Skemp, R. (2006). Relational Understanding and Instrumental Understanding. *Mathematics Teaching in the Middle School, 12*(2), 88-95.

Solbakken, S. S. (2019). *Statistikk for nybegynnere* (1. utg.). Fagbokforlaget.

Sollid, H. (2013). Intervju som forskningsmetode i klasseromsforskning. I: M. Brekke & T. Tiller (red.). *Læreren som forsker. Innføring i forskningsarbeid i skolen*. Oslo: Universitetsforlaget, s. 124-137.

Silverman, D. (2011). Designing a research project. I D. Silverman (Red.), *Interpreting qualitative data* (s. 27-56). Thousand Oaks, California: SAGE publications ltd.

Stenseth, B., Kaufmann, O. T. & Forsstöm, S. E. (2019). Programmering og matematikk. *Tangenten – tidsskrift for matematikkundervisning, 30*(2), 7–12.

Kaufmann, O. T. & Stenseth, B. (2020). Programming in mathematics education. *International Journal of Mathematical Education in Science and Technology, 52*(7), 1029-1048.

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

Taylor, M., Harlow, A. & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia - Social and Behavioral Sciences, 8*, 561–570. doi: 10.1016/j.sbspro.2010.12.078

Universitetet i Oslo (2021, 16. des). *Nettskjema diktafon-app*. https://www.uio.no/tjenester/it/adm-app/nettskjema/hjelp/diktafon.html

Utdanningsdirektoratet. (2019a). *Fagets relevans og sentrale verdier*. Udir. https://www.udir.no/lk20/mat09-01/om-faget/fagets-relevans-og-verdier?lang=nob

Utdanningsdirektoratet. (2019b). *Grunnleggende ferdiheter*. Udir. https://www.udir.no/lk20/mat09-01/om-faget/grunnleggende-ferdigheter?lang=nob

Utdanningsdirektoratet. (2019c). *Kjerneelementer*. Udir. https://www.udir.no/lk20/mat01-05/om-faget/kjerneelementer

Utdanningsdirektoratet. (2019d). *Algoritmisk tenkning*. Udir. https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/

Utdanningsdirektoratet. (2019e). *Kompetansemål og vurdering*. Udir. https://www.udir.no/lk20/mat09-01/kompetansemaal-og-vurdering/kv42?lang=nob

Vogt, Y. (2021, 1. feb). *Programmering blir allemannseie i skolen*. Apollon. https://www.apollon.uio.no/artikler/2021/1_utdanning_programmering.html

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *12*(3), 33-35. doi:10.1145/1118178.1118215

Vedlegg

Vedlegg 1: Prosjektgodkjenning fra Norsk senter for Forskningsdata (NSD)

Vurdering (1)

16.08.2021 - Vurdert
Det er vår vurdering at behandlingen av personopplysninger i prosjektet vil være i samsvar med personvernlovgivningen så fremt den gjennomføres i tråd med det som er dokumentert i meldeskjemaet med vedlegg 16.8.2021. Behandlingen kan starte.

NSD minner om at det må legges til rette for at de som ikke ønsker å delta ikke registreres på lydopptak.

DEL PROSJEKTET MED PROSJEKTANSVARLIG (la stå ved studentprosjekt).
Det er obligatorisk for studenter å dele meldeskjemaet med prosjektansvarlig (veileder). Det gjøres ved å trykke på "Del prosjekt" i meldeskjemaet. Om prosjektansvarlig ikke svarer på invitasjonen innen en uke må han/hun inviteres på nytt.

TYPE OPPLYSNINGER OG VARIGHET
Prosjektet vil behandle alminnelige kategorier av personopplysninger frem til 31.12.2022.

LOVLIG GRUNNLAG
Prosjektet vil innhente samtykke fra de registrerte til behandlingen av personopplysninger. Vår vurdering er at prosjektet legger opp til et samtykke i samsvar med kravene i art. 4 og 7, ved at det er en frivillig, spesifikk, informert og utvetydig bekreftelse som kan dokumenteres, og som den registrerte kan trekke tilbake. Lovlig grunnlag for behandlingen vil dermed være den registrertes samtykke, jf. personvernforordningen art. 6 nr. 1 bokstav a.

PERSONVERNPRINSIPPER
NSD vurderer at den planlagte behandlingen av personopplysninger vil følge prinsippene i personvernforordningen om:

• lovlighet, rettferdighet og åpenhet (art. 5.1 a), ved at de registrerte får tilfredsstillende informasjon om og samtykker til behandlingen
• formålsbegrensning (art. 5.1 b), ved at personopplysninger samles inn for spesifikke, uttrykkelig angitte og berettigede formål, og ikke viderebehandles til nye uforenlige formål • dataminimering (art. 5.1 c), ved at det kun behandles opplysninger som er adekvate, relevante og nødvendige for formålet med prosjektet

• lagringsbegrensning (art. 5.1 e), ved at personopplysningene ikke lagres lengre enn nødvendig for å oppfylle formålet

DE REGISTRERTES RETTIGHETER
NSD vurderer at informasjonen om behandlingen som de registrerte vil motta oppfyller lovens krav til form og innhold, jf. art. 12.1 og art. 13.

Så lenge de registrerte kan identifiseres i datamaterialet vil de ha følgende rettigheter: innsyn (art. 15), retting (art. 16), sletting (art. 17), begrensning (art. 18) og dataportabilitet (art. 20).

Vi minner om at hvis en registrert tar kontakt om sine rettigheter, har behandlingsansvarlig institusjon plikt til å svare innen en måned.

FØLG DIN INSTITUSJONS RETNINGSLINJER
NSD legger til grunn at behandlingen oppfyller kravene i personvernforordningen om riktighet (art. 5.1 d), integritet og konfidensialitet (art. 5.1. f) og sikkerhet (art. 32).

For å forsikre dere om at kravene oppfylles, må dere følge interne retningslinjer og eventuelt rådføre dere med behandlingsansvarlig institusjon.

MELD VESENTLIGE ENDRINGER
Dersom det skjer vesentlige endringer i behandlingen av personopplysninger, kan det være nødvendig å melde dette til NSD ved å oppdatere meldeskjemaet. Før du melder inn en endring, oppfordrer vi deg til å lese om hvilke type endringer det er nødvendig å melde: https://www.nsd.no/personverntjenester/fylle-ut-meldeskjema-for-personopplysninger/melde-endringer-i-meldeskjema Du må vente på svar fra NSD før endringen gjennomføres.

OPPFØLGING AV PROSJEKTET
NSD vil følge opp ved planlagt avslutning for å avklare om behandlingen av personopplysningene er avsluttet.

Kontaktperson hos NSD: Lisa Lie Bjordal
Lykke til med prosjektet!

Vedlegg 2: Informasjonsskriv og samtykkeskjema

Vil du delta i forskningsprosjektet

Praktisk innføring av programmering i matematikk?

Dette er et spørsmål til deg om å delta i et forskningsprosjekt. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål

Dette er en undersøkelse jeg skal bruke i min masteroppgave om matematikkundervisning ved Universitetet i Oslo. Formålet er å undersøke hvordan det arbeides med programmering i kurset Matematikk 1T. Dette er interessant fordi programmering er et nytt tema i matematikkfaget etter Fagfornyelsen i 2020.

Hvem er ansvarlig for forskningsprosjektet?

Institutt for lærerutdanning og skoleforskning ved Universitetet i Oslo er ansvarlig for prosjektet.

Hvorfor får du spørsmål om å delta?

Jeg ønsker å observere matematikkundervisning i en 1T-klasse.

Hva innebærer det for deg å delta?

Hvis du velger å delta i prosjektet, innebærer det at du tillater at jeg observerer klassens matematikkundervisning i noen uker og tar lydopptak. Kun jeg og veileder vil ha tilgang til lydopptakene, og de vil bli slettet når prosjektet er slutt vinteren 2022. Alt vil anonymiseres og det vil ikke være mulig å identifisere deg eller skolen i den ferdige masteroppgaven.

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

Ditt personvern – hvordan vi oppbevarer og bruker dine opplysninger

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

tilgang til datamaterialet og personopplysningene. Navnet og kontaktopplysningene dine vil jeg erstatte med en kode som lagres på egen navneliste adskilt fra øvrige data.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?

Alle personopplysninger og lydopptak slettes når prosjektet avsluttes/oppgaven er godkjent, noe som etter planen er vinteren 2022.

Dine rettigheter

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

innsyn i hvilke personopplysninger som er registrert om deg, og å få utlevert en kopi av opplysningene,

å få rettet personopplysninger om deg,

å få slettet personopplysninger om deg, og

å sende klage til Datatilsynet om behandlingen av dine personopplysninger.

Hva gir oss rett til å behandle personopplysninger om deg?

Vi behandler opplysninger om deg basert på ditt samtykke.

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

Hvor kan jeg finne ut mer?

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

Universitetet i Oslo ved Jonatan Glen Joseph (jonatangjo@hotmail.com) eller Arne Hole (arne.hole@ils.uio.no).

Vårt personvernombud: Roger Markgraf-Bye (personvernombud@uio.no)

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

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

Med vennlig hilsen

Arne Hole Jonatan Glen Joseph

(Forsker/veileder) (Student)

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet *Praktisk innføring av programmering i matematikk*, og har fått anledning til å stille spørsmål. Jeg samtykker til:

å delta i observasjon av klassens undervisning i matematikk

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

---

(Signert av prosjektdeltaker, dato)

Vedlegg 3: Intervjuguide

Veiledende spørsmål til semi-strukturert intervju

- Har du blitt mer positiv eller negativ i løpet av denne høsten til programmering har blitt inkludert i læreplanen?
- Hva er så langt din erfaring med å undervise programmering?
- Hva tenker du er formålet med å inkludere programmering og algoritmisk tenkning i læreplanen i matematikk?
- Føler du at du har nok kunnskap om programmering for å kunne legge til kunne legge til rette for at elevene skal oppnå kompetansemålet som tar for programmering?
- Hvordan har din programmeringsundervisning hjulpet elevene til å lære matematikk?
- I læreplanen står det at elevene skal formulere og løse problemer ved blant annet programmering og algoritmisk tenkning, hvordan har du fått til dette i din undervisning? Og hvis du ikke har fått det til, hvorfor ikke?
- Hva har vært ditt formål med å inkludere programmering i undervisninga di? Og hvordan har du egentlig integrert programmering i undervisninga?
- Hva overrasket deg mest med å undervise programmering denne høsten?
- Hva overrasket deg minst med å undervise programmering?
- Tror du det blir lettere å undervise 1T-elever som har programmert i ungdomsskolen?
- Hvordan har deg og dine kolleger jobbet med å inkludere programmeringa? Hva er du førnøyd/misfornøyd med i dette arbeidet?
- Hva føler du har manglet denne høsten for at du kan legge til rette for programmering slik du ønsker, med de formålene du har hatt?
- Føler du at matematikk har fått ansvaret for programmeringen?
- Hva kommer du til å gjøre annerledes dette halvåret?
- Hvilke temaer i 1T føler du kan kobles til programmering, temaer dere har hatt frem til nå og temaer dere skal ha dette semesteret?
- Er det noe du vil legge til?