Programmeringsundervisning i matematikk på vgs.

Utfordringer ved programmeringsundervisning i matematikk ved en skole hvor alle forhold ligger til rette for det.

Kristine Gaustad Heimdal

Lektorprogrammet

30 studiepoeng

Institutt for lærer- og skoleforskning

Det utdanningsvitenskapelige fakultet

15.09.2020

Programmeringsundervisning i matematikk på vgs.

Utfordringer ved programmeringsundervisning i matematikk ved en skole hvor alle forhold ligger til rette for det.

2020

Programmeringsundervisning i matematikk på vgs.

Kristine Gaustad Heimdal

http://www.duo.uio.no

Trykk: Reprosentralen, Universitetet i Oslo

Sammendrag

Målet med dette masterstudiet er å undersøke om PRIMM (Predict-Run-Investigate-Modify-Make) er en god undervisningsmodell for programmeringsundervisning i matematikkfaget. I tillegg ønsker jeg å identifisere og vurdere utfordringer som lærere må være klar over for at undervisningen skal bli vellykket. Med bakgrunn i dette og koding av funn, formulerte jeg to forskningsspørsmål:

1. Hvor god er PRIMM som undervisningsmodell for undervisning i programmering i matematikk på vgs?
2. Hvilke utfordringer må lærere være klar over for at undervisning i programmering skal bli vellykket?

Informantene er én matematikklærer som også veileder i digital læringsteknologi. 25 elever fra en 1T-klasse og 13 elever fra en R1-klasse, begge klasser på en teknologisatsende skole. Over halvparten av elevene hadde erfaring med programmering før de begynte på vgs.

Undervisningsmodellen er testet for å se om PRIMM er en egnet modell i forbindelse med innføring av programmering i matematikkfaget. Undervisningsmodellen skal tilrettelegge for at elevene utvikler kompetanse i algoritmisk tenkning, problemløsning og samarbeid, kompetanser som alle er viktige i arbeidslivet. Undervisningsmodellen skal være et verktøy spesielt til hjelp for lærere som har liten eller ingen erfaring i å undervise i programmering i matematikkfaget. PRIMM kan også brukes etter implementering, men prosjektet fokuserer på å gi lærere med lite erfaring i programmeringsundervisning en modell å ta utgangspunkt i. Modellen er ikke alene det som skal til for at implementeringen skal bli vellykket. Lærerens faglige og didaktiske kompetanse er helt vesentlig for å utnytte modellens muligheter.

Studien viser at programmeringsundervisning i matematikkfaget kan by på utfordringer. Funn viser at elevene hadde utfordring med det sosiale aspektet i parprogrammering og at tekniske utfordringer tar mye tid fra læreren. Positivt funn viser at programmering gir elever økt motivasjon til å arbeide med matematikkfaget. Det viser seg også at det ikke er programmeringen elevene har størst utfordring med i oppgaveløsningen men det matematikkfaglige.

Forord

Denne masteroppgaven markerer slutten på min studietid på Universitetet i Oslo. Jeg vil med det takke Helmer Aslaksen for veiledning og inspirasjon gjennom denne oppgaven. Takk til alle venner for alt faglig og sosialt, dere har gjort studietiden fantastisk. Takk til Jørgen Dokken for støtte.

Kristine Gaustad Heimdal
Oslo, September 2020

Innholdsfortegnelse

1 Innledning

Matematikklærere står overfor en spennende oppgave når programmering innføres med Fagfornyelsen høsten 2020. I matematikkfaget er temaer fjernet til fordel for økt fokus på algoritmisk tenkning og dybdelæring (Kjerneelementgruppen i matematikk, 2017). Det er også tatt inn programmering som ett nytt teknologisk verktøy. Det er diskutert og argumentert for hvorfor programmeringen skal inngå i matematikkfaget. Argumenter er at programmering skal inspirere elever til å velge teknologiske fag i høyere utdanning (Norstein & Haara, 2018) og gjøre matematikkfaget på vgs. mer moderne. Samfunnet endres og teknologien omgir oss hele tiden. Barn og unge som vokser opp i dag har stort sett kjennskap til digitale medier, men det betyr ikke at de har tilstrekkelig grunnleggende kompetanse tilpasset den teknologiske utviklingen som preger dagens arbeidsmarked.

Det er kritisert at norsk skole har lært elever til å være forbrukere og ikke utviklere av teknologi (NOU 2013:2, 2013, s.102). Sanneutvalget (Sanne et al., 2016) la frem en faggjennomgang i 2016 som pekte på at alle elever trenger en utdannelse som gjør dem rustet for et samfunn integrert med teknologi. Elevene trenger å lære hvordan Norge skal utvikle økonomisk lønnsom teknologi og kunne reflektere, lese og forstå avisartikler med temaer om teknologi. Utvalget foreslo å innføre teknologi og programmering som et eget obligatorisk fag, men det ble bestemt at programmering skal inngå i matematikkfaget uten at timetallet økes. Dette er noe som lærere har reagert på.

Reaksjonene på å innføre programmering i matematikkfaget har vært at det allerede er stofftrengselen i faget. Temaer er fjernet fra matematikkfaget men det er ikke estimert timetall på innføring og integrering av programmeringsverktøy i faget. I 2015 ble GeoGebra, et *dynamic geometry program*, innført som et digitalt verktøy uten at lærerne fikk nødvendig opplæring i bruk av og hvordan utnytte potensialet til programmet i undervisning. GeoGebra er et godt undervisningsverktøy til å illustrere grafiske funksjoner og geometri. Som eksamensverktøy må elevene lære seg kommandoer for å bruke CAS[1] og løse oppgaver. Elevene må også til enhver tid ha oppdatert programvare på sine PCer og ved oppgradering

[1] Computer Algebra System - dataprogram for å regne symbolsk matematikk. CAS (2017, 14. februar). I Wikipedia. Hentet fra https://no.wikipedia.org/wiki/CAS CAS. Hentet: 12. august.2020

kan det oppstå installasjonsproblemer. Før avgjørelsen ble tatt om innføring var det gjennomført lite forskning på hvordan det digitale *dynamic geometry program* verktøyet har fungert og påvirket matematikkfaget i norsk skole. Med GeoGebra var det hvordan oppgavene ble gitt på eksamen som gjordet det til et mål for hvordan GeoGebra ble brukt i undervisningen.

Tilsvarende kan man tenke seg vil gjelde ved innføring av programmering i matematikkfaget. Basert på denne erfaringen er det mange lærere som nå lurer på hvordan programmering skal vurderes. UDIR har ikke gitt klar informasjon om hvordan programmering vil inngå i eksamen og lærere reagerer på at denne diskusjon i liten grad har blitt prioritert.
Erfaring fra dette masterstudiet er også at innføring i og bruk av Python, programmeringsverktøyet som ble brukt til å løse matematikkoppgaver i dette studiet, tok vesentlig lenger tid enn avsatt undervisningstid på det matematikkpensum som nå er tatt ut av matematikkfaget i den nye læreplanen.

1.1 Bakgrunn for studiet

Denne masteroppgaven er tilknyttet prosjektet *Undervisningsopplegg for programmering i realfagsmatematikk* (Heretter kalt UPR). Hovedmålene til UPR-prosjektet er å undersøke «effekten» av programmeringsundervisning på vg1 og vg2 ved bruk av PRIMM-modellen (Predict-Run-Investigate-Modify-Make) og å utvikle forskningsbaserte undervisningsopplegg der programmering gir støtte til dybdelæring i matematikk. UPR er et designbasert forskningsprosjekt der undervisningsopplegget er basert på den engelskutviklede undervisningsmodellen PRIMM.

Sentance og Waite (2017b) skriver at elever kan oppleve programmering som vanskelig og lærere som har liten erfaring i programmeringsundervisning kan oppleve det som en utfordring å støtte og undervise elevene i programmering. Det er per dags dato lite forskning på støttemateriell og didaktisk metoder. Det er også lite tilgjengelig forskning på programmeringsundervisning og hvordan bruke programmering i matematikkundervisningen.

Jeg ønsker å undersøke om PRIMM er en egnet undervisningsmodell for programmering i matematikkfaget og om PRIMM er et godt verktøy for lærere som ikke har eller har lite

erfaring med programmeringsundervisning i matematikk på vgs. Vil bruk av PRIMM kunne gjøre forberedelse til undervisning lettere? I oppgaven vil jeg diskutere og argumentere funn ut ifra faglige argumenter og påstander fra informantene. Jeg håper at denne oppgaven vil bidra til at lærere gis innsikt i noen av de muligheter programmering kan gi og gjøre lærere tryggere på å undervise i programmering i matematikkfaget. Jeg håper også at dette studiet av PRIMM vil gjøre lærere bedre forberedt på utfordringer de kan møte. Jeg har brukt data fra en lærer og to realfagsmatematikk-klasser på vgs. (T1 og R1). Begge klassene jobbet med samme tema (funksjoner), undervisningsplan og oppgaver men brukte forskjellig versjon av Python editor.

1.2 Forskningsspørsmålene

I dette kvalitative studiet har jeg formulert to forskningsspørsmål:

1. Hvor god er PRIMM som undervisningsmodell for undervisning i programmering i matematikk på vgs.?
2. Hvilke utfordringer må lærere være klar over for at undervisning i programmering skal bli vellykket?

For å besvare forskningsspørsmålene har jeg i forkant av datainnsamlingen fulgt den ene forsøksklassen i innføringsprosessen i programmering. Jeg har lest fagartikler, avisartikler og fulgt debatter for å forberede forskningen. Informantene i dette studiet var valgt ut av lederen for prosjektet som også er en av informantene. En bredere presentasjon av informantene gis i kapittel 5.1.1.

1.3 Oppbygning av oppgaven

Kapitlene i denne masteroppgaven er i rekkefølgen: Programmering i skolen, PRIMM, Teori, Metode, Resultater, Diskusjon og Konklusjon. Jeg vil begynne med kort historisk overblikk over programmering i skolen, etterfulgt av relevant tidligere forskning på PRIMM, deretter relevant teori. I metodekapittelet presenteres hvilke metoder jeg brukte i forbindelse med forskningsdesign, metoder og analysemetoder. Jeg vil også diskutere metodens validitet og relabilitet og presentere funn knyttet til intervjuet og spørreundersøkelse. Funnene blir så

diskutert opp mot relevant teori. Til slutt presenteres konklusjon på forskningsspørsmålene og tanker og innspill til videre forskning.

2 Programmering i skolen

Fremtidens arbeidsmarked vil bli preget av ny teknologi og fremtidige elever skal være med på å utvikle denne teknologien. Som en konsekvens av dette har programmering blitt en *21st-century skill* som politikere og næringsliv ser som en nødvendighet å innføre i læreplanen (Bocconi, Chioccariello & Earp, 2018; Forsström & Kaufmann, 2018).

Programmering i skolen er ingen ny idé. På 60-tallet kom Seymour Papert (1993) med programmeringsspråket LOGO. Elever skulle bruke datamaskiner, med LOGO som verktøy, til å utvikle kreativitet, algoritmisk tenkning og bli bedre problemløsere (Papert, 1993). Idéen var å la elevene programmere en robotskilpadde til å bevege seg rundt. Ideen til Papert ble videreutviklet og Scratch[2] ble et produkt av LOGO. Programmeringsinnføring støttes av EU som anbefaler alle EU-land å lære elever programmering på skolen. Ett av hovedargumentene for innføring er det Forsstrøm og Kaufmann (2019, s.19) skriver at elevene skal lære å styrke sentrale ferdigheter som logisk tenkning og problemløsning i teknologitette samfunn. Programmering bidrar til å trene elever i å resonnere, presentere, modellere problemer med symboler og algoritmer, beskrive kodingsprosessen og se etter strukturer og regulariteter.

Programmering i Norge, Finland og Sverige

Norge er det siste landet i Norden til å innføre programmering i skolen og som de andre nordiske landene har vi valgt å innføre programmering i matematikkfaget. Finland innførte programmering både i matematikkfaget og i andre fag for 1-9 klassingene og målet er at elevene skal utvikle algoritmisk tenkning og lage enkle programmer. I Finnland teller programmering og algoritmisk tenkning i sluttvurderingen i 9.-klasse (Sevik, 2016). Sverige har valgt å innføre programmering i matematikk- og teknologiske fag og elevene skal med programmering kunne skape, anvende og teste og forbedre algoritmer.

Men det er lite som skulle tilsi at programmeringen skulle bli implementert i matematikkfaget i norsk skole når vi leser Sanne et al. (2016) sin rapport. Sanner et al. (2016) anbefaler at programmering bør implementeres som et eget fag. Slik ble det ikke og fra og med høsten

[2] Scratch brukes av kodeklubber for å lære barn programmering. Det er gratis og et visuelt programmeringsspråk for læring. Scratch. I Wikipedia. Hentet fra: https://no.wikipedia.org/wiki/ScratchScratch. (4. juli. 2019). Hentet 23. august. 2020

2020 skal programmering gradvis integreres i læreplanene for matematikkfaget. Hovedendringen i læreplanen for matematikk er at programmering og algoritmisk tenkning skal inngå i faget og for å få plass til programmering måtte tema tas ut. Temaene sannsynlighet, geometri og differensiallikninger ble fjernet fra matematikkfaget, temaer som hadde passet godt sammen med programmering.

Programmering i England

I England lærer elever fra 11 årsalderen tekstbasert programmering og det ble innført som et eget computational science fag: *Computing*, i 2014 (Balanskat & Engelhardt, 2015). Faget har som et mål å øke elevenes programmeringskompetanse. Elevene skal forstå hva algoritme er, hvordan programmer virker og benytte minst to programmeringsspråk (Sevik, 2016). Da ICT-faget (norsk IKT) ble innført hadde mange lærere lite eller ingen erfaring med å undervise i programmering. PRIMM har sitt utspring fra programmeringsinnføringen i Computing-faget i England.

2.1 Computational thinking og algoritmisk tenkning

I Norge brukes ikke «computational thinking» (heretter CT) som begrep i læreplaner eller kjerneelementer. Wing (2006) var tidlig ute med å promotere CT som en ferdighet alle burde inneha. Det for å lære elever til å tenke strukturert slik at de skal vite hvordan en datamaskin skal programmeres for å utføre en gitt handling.

> «Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science.». (Wing, 2006, s.33)

Hun lanserte begrepet i kontekst med informatikkstudier og beskriver computational thinking med bruk av fem konsepter; tenke abstrakt ved å luke ut unyttig informasjon, bryte opp problemet, velge riktig fremgangsmåte, lage steg-for-steg fremgangsmåte for å gjøre løsningen gjenkjennbar og vurdere sluttprodukt.

Algoritmisk tenkning er den norske oversettelsen av computational thinking (Sevik, 2016).

Algoritmisk tenkning er ett av kjerneelementene i matematikk fellesfag og det er beskrevet i *Utforskning og problemløsning* at «Algoritmisk tenking er viktig i prosessen med å utvikle strategier og fremgangsmåter og innebærer å kunne bryte ned et problem i delproblem som kan løses systematisk.» (Kunnskapsdepartementet, 2018a, s.15). Algoritmisk tenkning er nært knyttet til programmering og matematikk. Algoritmisk tenkning brukes i programmering av datamaskiner for å løse et problem og det gjerne ved å følge en oppskrift og semantikk.

Algoritmisk tenkning er et godt verktøy for å løse problemer, velge fremgangsmåte og riktig verktøy. Bocconi et al. (2018) skriver i sin rapport at i de nordiske landene har tolkningen av algoritmisk tenkning beveget seg i en annen retning enn computational thinking i England. Norge har valgt å definere egne konsepter under den algoritmiske «tenke-paraplyen». Uttrykket inkluderer digital kompetanse som en grunnleggende ferdighet som inngår i matematikkfaget. (Bocconi et al., 2018). Arbeidsgruppen som utarbeidet kjerneelementene i matematikk argumenterte for at skal det fokuseres på algoritmisk tenkning i matematikkfaget er det naturlig at programmering inngår i kompetanseelementet.

Utdanningsdirektoratet (2019) gir ikke én definert definisjon på algoritmisk tenkning og viser til at det er mange forskjellige beskrivelser. I artikkel *Algoritmisk tenkning* (Utdanningsdirektoratet, 201, s. 1) beskriver UDIR egenskapene til den algoritmiske tenkeren. De beskriver det som å tenke ut en mulig løsningsvei og kunne bruke riktig teknologiske kompetanse for å få en datamaskin til å løse problemet. Det er altså ikke bare det å kunne programmere som innebærer å tenke algoritmisk som også Figur 1 illustrererFigur 1

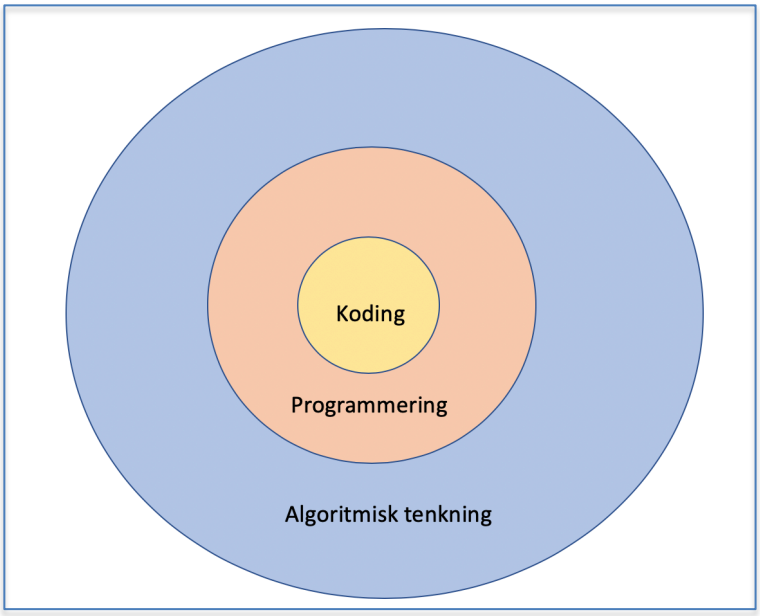
Figur 1: Illustrasjon om den algoritmiske tenkeren. Hentet Utdanningsdirektoratet (2019). Tilpasset fra Barefoot Computing (UK).

2.2 Programmering er ikke bare koding

Programmering er brukt som en term på aktiviteten å skrive instrukser i et kodespråk for å få en datamaskin til å utføre en oppgave (Sevik, 2016, s.9). Programmering er bygget opp som et produkt av tre ferdigheter; algoritmisk tenkning, programmering og kode. Figur 2 viser at koding utgjør den minste delen av det å programmere, hoveddelen er algoritmisk tenkning. Ser vi figur 2 som en helhetlig prosess kan programmering deles inn fremgangsmåter-instrukser som gjennomføres i rekkefølge; dele opp problemet, gjenkjenne regler og formler, lage algoritme, kode algoritmen og til slutt evaluere og vurdere sluttproduktet. Programmeringsprosessen for å få datamaskinen til å utføre ønsket handling kan skrives i forskjellige språk. Java, Python, C++ er eksempler på programmeringsspråk og det er Python som er brukt i denne masteroppgaven. Kodingen utføres i.h.h.t syntaksen i programmeringsspråket som er valgt å bruke for å gi instrukser til datamaskinen.

For å kunne diskutere undervisning i programmering må man vite hva det innebærer å programmere. Noen forskere ser på det å programmere som å kode datamaskiner og gjenstander til å utføre ønskede prosesser. Andre forskere beskriver programmering som en

ferdighet i å tenke rekursivt, kjenne igjen mønster og implementere programkoden på datamaskin for å løse problemet (Balanskat & Engelhardt, 2015).



Figur 2: Egenlaget oversikt over hva programmering er. Etter ide fra Liv Oddrun Voll. Naturfagsenteret.

3 PRIMM

I dette masterstudiet har vi testet undervisningsmodellen PRIMM på to norske klasser våren 2020. PRIMM er en undervisningsmodell utviklet til bruk i det obligatoriske faget Computing på engelske skoler. Som det er i Norge hadde de færreste lærerne i England, som skulle undervise i ICT på innføringstidspunktet, erfaring i å undervise i tekstbasert programmering. Derfor ble det forsket på og utviklet en modell som kunne være et godt rammeverk for lærerne til planlegging av programmeringsundervisning.

PRIMM-metoden er et designbasert rammeverk med fem stadier (predict, run, investigate, modify, make). PRIMM er basert på og utviklet gjennom tre forskningsområder: *use-modify-create,* nivåer av abstraksjon og lesing og forståelse av kode (Sentance & Waite, 2017a). Basert på de tre forskningsområdene ble PRIMM designet, testet og implementert i engelsk skole og det er denne versjonen av PRIMM som er brukt i dette masterstudie.

I de neste kapitlene beskrives de tre forskningsområdene og resultat fra tidligere forskning på bruk av PRIMM og en beskrivelse av PRIMM-modellen.

Use-Modify-Create
Use-Modify-Create er et rammeverk for å støtte opp om elevenes progresjon i programmering. Elevene jobber kontinuerlig med egen forbedring og progresjon. De første oppgavene elevene skal gjøre er å lese og modifisere programmer som er laget av andre. Elevene lager til slutt sine egne programmer i en iterativ prosess der de avgrenser, tester og analyserer som i PRIMMs predict, run og investigate.

Abstraksjonsnivå
Rammeverket for å utvikle elevenes abstraksjonsnivåer består av fire deler: utførelse, program, algoritme og problem. Rammeverket fokuserer på at elevene skal vite hvilket nivå de jobber på og hva de kan overføre mellom nivåene. Uansett hvilket stadium og nivå du er på i PRIMM er abstraksjon i fokus.

Lesing og forståelse av kode

Å lære programmering er en sekvensiell og kumulativ prosess. Elevene må akkumulere kunnskap for å få det store bilde for programmering (Sentance & Waite, 2017a). Elevene begynner med å lese og skrive av en programkode som de så gjør begrensede endringer i som for eksempel konstruere en for-løkke. Det er først mot slutten av prosessen at elevene kan skrive kodebiter helt på egenhånd. For som forskerne sier kreves det ganske god kodeforståelse før elevene skal klare å skrive kode på egenhånd (Sentance & Waite, 2017, s.1).

3.1 Tidligere studier på PRIMM

Sentance og Waite (2017) gjennomførte to pilotstudier og en storskalastudie med lærere som alle var nye i å undervise i tekstbasert programmering på videregående skole i England (K-12 education). Studiene tar for seg implementeringen og evalueringen av metoden brukt i klasserommet.

Det første pilotstudie ble gjennomført med 15 ikke-spesialiserte computational science lærere som deltok på et 8 ukers kurs i «hvordan undervise i programmering for klasse 6-8» (10-13 år). Ved endt kurs besvarte kursdeltakerne en spørreundersøkelse. Resultatene viste at lærerne mente at alle stadiene i PRIMM var pedagogisk gode. 47% mente modellen var veldig god og nyttig. 80% av lærerne svarte at de ville fortsette å bruke PRIMM i undervisning og 13% at de sannsynlig ville fortsette å bruke PRIMM.

Det andre pilotstudie ble gjennomført med 7 erfarne IKT-lærere rekruttert til å bruke PRIMM i deres klasser. Studiet hadde tre steg: opplæring for lærerne i PRIMM, 4 til 6 ukers bruk av PRIMM i klasserom og til slutt et intervju. I det samme pilotstudie besvarte forsøkselevene og lærerne en før- og etter multiple choice-spørreundersøkelse for å identifisere tegn på læring.

I storskalaundersøkelsen deltok 500 elever i alder 11 til 14 år (7-9 klasse) og 113 lærere som underviste disse elevene. Sentance, Waite og Kallia (2019) brukte en kvasi-eksperimentell tilnærming for å undersøke elevenes prestasjon med bruk av PRIMM som rammeverk. Lærerne brukte PRIMM i undervisningen i 3 til 4 måneder og elevene gjennomførte en test

ved oppstart og avslutning av undersøkelsen. Det viste seg at 492 elever hadde signifikant forbedring på testen gjennomført etter undervisning med bruk av PRIMM sammenliknet med referansegruppen på 180 elever som ikke ble undervist i.h.h.t. til PRIMM. Lærerne ble intervjuet etter at forsøket var ferdig. Konklusjonen var at PRIMM er en god undervisningsmodell for elevene fordi de fikk en bedre forståelse av programmering. Lærerne merket økt læring og bedret samarbeidsevne hos elevene. Lærerne mente også at muligheten for differensiering var bra og at elevene aktivt fikk brukt relevante begreper i muntlig arbeid. Modellen gjorde timene effektive og de fleste lærerne ville fortsette å bruke den. De konkluderer med at modellen kan brukes til undervisning i ICT-faget.

Basert på erfaring fra de tre studiene har forskerne i England konkludert med at PRIMM er en metode som er nyttig for lærere å bruke i undervisningen. Lærerne har også tilgjengelig undervisningsmateriell på en offentlig nettside. Forskerne ønsker å videreutvikle PRIMM og at det gjennomføres flere forskningsprosjekter.

3.1.1 PRIMMs fem stadier

PRIMM har fem stadier som er basert på de tre forskningsområdene beskrevet tidligere. Stadiene må benyttes i den kronologiske rekkefølgen de er presentert, men alle stadier må ikke gjennomføres. Med støtte i PRIMM skal elevene gjøre parprogrammering. Parprogrammering er å jobbe sammen to-og-to på én PC og blir nærmere beskrevet i kapittel 3.2.

Predict: I dette stadiet skal elevene i par og i plenum studere en kort kodesnutt, gjerne skrevet av læreren, uten å kjøre koden. Parprogrammering gir mange muligheter. Læreren kan velge å la elevene sitte to-og-to og diskutere en kort tid før de så diskuterer høyt i klassen. De kan bruke hele timen på forskjellige kodesnutter som elevene skal diskutere i grupper og/eller i plenum. Elevene oppfordres til å se etter tegn/hint som henviser til hvilke(n) funksjonen(er) koden gjør.

Run: I dette stadiet foreslår utviklerne av PRIMM at elevene skal «copy-paste» eller laste koden inn i editoren slik at det ikke blir skrivefeil som det kan bli hvis elevene selv skriver av programkoden. Vi valgte å la elevene skrive programkoden inn i editoren på sin egen PC og

jeg vil diskutere begrunnelsen senere, men kort fortalt er det for at elevene skal bli kjent med tastaturet for koding.

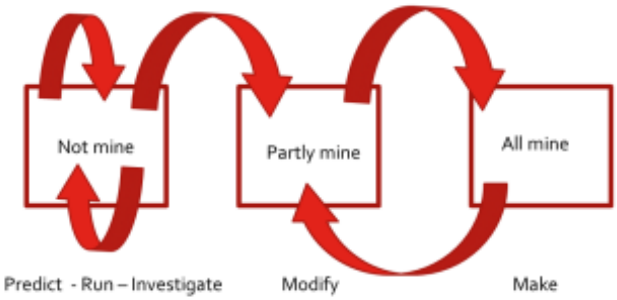
Investigate: I dette stadiet valgte vi å la elevene arbeide sammen to-og-to for å skrive forklaring på hver linje i koden. «Hva gjør eller skjer på hver linje i koden?». Utviklerne av PRIMM foreslår å legge inn feil i koden for å la eleven øve på «de-bugging[3]» eller klippe opp koden i biter fra et papir for å la elevene sette det sammen som i et puslespill (kan leses på primming.wordpress.com). Med tanke på at programmering inngår matematikkfaget i Norge har vi ikke like mye tid å bruke på programmering som de har i England. Derfor valgte vi å ikke gjøre slike aktiviteter.

Modify: Med utgangspunkt i den ferdige koden elevene har skrevet av inn i editoren skal de så modifisere koden for å løse oppgavene. Koden de får er gjerne ganske enkel og etterhvert skal elevene lære å gjøre koden mer avansert med f.eks løkker og iterasjoner. Stadiet er differensiert og elevene kan selv velge hvor avansert kode de selv vil skrive. I dette stadiet blir koden elevene fikk av læreren «partly mine» (Figur 3) ved at elevene selv stegvis forbedrer koden og elevene trenes i å legge til egne små kodebiter og se virkningen av kodebitene i et større program (Sentance, Waite & Kallia, 2019). Koden vil bli mer og mer elevens eierskap og elevene beveger seg over til neste stadiet.

Make: I dette stadiet skriver elevene helt sin egen kode og får oppgaver som er tilrettelagt for det. Elevene kan bruke biter av den originale koden men hensikten er at de skal få mulighet til å fordype seg i temaet og skrive egen kode. Dette stadiet er viktig for å gjøre koden til «all mine» (Figur 3). Elevene må starte med å planlegge og legge en løsningsstrategi, lage en passende algoritme og så kode denne, teste og kjøre programmet for å se om det fungerer. Stadiet oppmuntrer til kreativ tenking og at elevene får eierskap til programmet.

Figur 3 viser progresjonsrekkefølgen, venstre til høyre, for PRIMM. Pilene som går fra høyre til venstre er stegene elevene kan gå tilbake hvis de står fast eller som i *make* stadiet finne inspirasjon til egen kode. Boksene viser eierskapet elevene har til sine løsninger.

[3] Søke etter feil - feilsøking



Figur 3: Illustrasjon tatt fra www.primming.wordpress.com. Oversikt over eierskap av kode og eksempel på fremgang mellom stadiene i rammeverket.

3.2 Parprogrammering

I PRIMM settes to elever sammen for å arbeide med oppgavene, kalt parprogrammering. Parprogrammering foregår slik at to elever sitter sammen på én datamaskin og en elev skriver på tastatur og en elev vurderer. Wiliams og Kessler (2003) gjorde et forsøk der de gjennomførte tester med to typer grupper - parprogrammering og SBS[4]. De erfarte at parprogrammering var mest effektivt da SBS ga mindre samarbeid mellom partnerne enn parprogrammering. Ashbacher (2002) har gjort en vurdering av hva som kan gjøres bedre ut ifra funnene i parprogrammeringsforsøket til Wiliams og Kessler (2002).

Ashbacker (2002) tar utgangspunkt i ferdighetsforskjellene til partnerne, funn som er identifisert som hovedproblemet ved parprogrammeringen. I forsøket til Wiliams og Kessler (2002) var elevene gruppert i par etter elevenes ferdigheter:

- Ekspert – ekspert
- Ekspert – gjennomsnittlig
- Ekspert – nybegynner
- Nybegynner – nybegynner

[4] «Side-by-side Programming»: To personer jobber med hver sin del av programmet på hver sin PC. Går sammen for å diskutere kode og programmet og «brainstorming». Hentet fra Wiliams og Kessler (2003).

Ashbacker (2002) skriver etter å ha analysert hvilke bakgrunnskunnskaper elevene som jobbet sammen har at man bør unngå visse parsammensetninger. Å sette ekspert sammen med nybegynner er en sammensetning man bør unngå fordi nybegynneren ikke klarer å holde følge eller å bidra, og dette kan føre til at eksperten «ekskluderer» partneren. Analysen av forsøket til Wiliams og Kessler (2002) viste også at det ikke var plassert elever med gjennomsnittlig- og nybegynnererfaring sammen. Det var heller ikke satt elever med gjennomsnittlig – gjennomsnittlig kompetanse i samme par. Fordelen med gjennomsnittlig - nybegynner er at alle har vært nybegynner en gang i tiden. Kompetansegapet mellom gjennomsnittlig og nybegynner vil ofte heller ikke være så stort at de ikke klarer jobber å jobbe greit sammen, så denne parsammensetningen kan være en god kombinasjon.

Andre mulige utfordringer er at partnerne er av forskjellige kjønn, fra ulike kulturer, en dominant skribent, egoistisk- eller blyg personlighet (Ashbacker, 2002, s. 180). Dette er en amerikansk studie så det kan forventes at segregering, kulturforskjeller og to av ulikt kjønn er et større problem der enn man kan forvente i Norge.
Det er ingen forskning som viser at menn er flinkere i programmering enn kvinner. Men det er flere gutter enn jenter som tar de tyngre realfagene som fysikk og teknologi (Ingeniørenes stemme, 2018).

Med kultur mener Ashbacker at de kommer fra forskjellige programmeringskulturer. I Norge kommer elevene til en videregående skole fra flere forskjellige ungdomsskoler. Det lærerne på en ungdomsskole mener er god programmeringsundervisning, vaner og kompetansemål kan lærerne på andre ungdomsskoler og på vgs. ha en annen oppfatning av. Ulik kompetanse kan skape et skille i klassen på videregående skole. Egoisme eller trang til å styre kan man ikke unngå å møte på i en elevgruppe likedan som i arbeidslivet, men kan bearbeides.

4 Teori

I dette kapittelet defineres begrepet dybdelæring og vi diskuterer matematisk kompetanse som en del av dybdelæringen i matematikkfaget. Avslutningsvis belyses programmering og problemløsning i matematikk.

4.1 Matematikkopplæring

Hva som sees på som god matematisk kompetanse har endret seg fra å regne aritmetisk til det som i dag sees på som fremtidsrettet matematikk. NOU 2014:7 (2014, s.33) viser til syv forutsetninger for god læring; deltakelse, kommunikasjon, dybdeforståelse, utfordringer, differensiering, læringsmiljø med hensyn på følelser og motivasjon og tankeprosess. Ut ifra forutsetningene og NOU 2014:7 er det et mål at elevene skal tilegne seg læringskompetanse tilpasset det 21. århundrets kompetanse i metakognisjon og selvregulering. Relevant læringsstrategi tilrettelegger for å få elevene til å tro på egen evne til å lære, motivasjon og støtter elevenes utvikling i læringsstrategier (NOU, 2014:7, s. 11).

4.2 Dybdelæring

På 1920-tallet var instrumentell undervisning vanlig slik at elevenes kunnskap ble tilpasset det industrielle samfunnet. I dag er arbeidsmarkedet mer teknologisk komplekst og næringslivet skal være kostnadseffektivt og etisk korrekt og til dette er instrumentell undervisning ikke egnet (Sawyer, 2006). Sawyer (2006) skriver videre at dagens elever må få opplæring med fokus på dybdekunnskap i å løse komplekse problemer og tenke kreativt. Det for å ha kompetanse til å kunne generere nye ideer, teorier og kunnskap. I tillegg konstaterer han at elever trenger å uttrykke matematikk skriftlig og muntlig for å kunne forstå matematisk tankegang. I den nye læreplanen er det økt fokus på dybdelæring for å trene elevene i nettopp Sawyers (2006) syn på fremtidens kompetanser. Dybdelæring er, etter Marton og Säljö (1976), inndelt i to læringsstrategier: «deep level-processing» og «surface level-processing». Disse to uttrykkene kan oversettes til norsk som dybdelæring og overflatelæring. Elever som pugger regler og fakta uten å se sammenhenger plasserer de under overflatelæring. De elevene som lærer å se sammenhenger og prøver å bygge forståelse

plasseres under dybdelæring. Elevene i dybdelæringskategorien blir motiverte til å gjøre det bra på skolen og det viser seg at de får en indre motivasjon til å lære og forstå (Marton & Säljö, 1976). Tabell 1 er en egenlaget tabell over kjennetegn på dybdelæring og overflatelæring basert på Sawyers (2006, s.4) forskning.

Dybdelæring	Overflatelæring
Elever relaterer nye ideer og begreper til tidligere kunnskap og erfaringer.	Elever jobber med nytt lærestoff uten å relatere det til hva de kan fra før.
Elever organiserer egen kunnskap i begrepssystemer som henger sammen.	Elever behandler lærestoff som adskilte kunnskapselementer.
Elever ser etter mønstre og underliggende prinsipper.	Elever memorerer fakta og utfører prosedyrer uten å forstå hvordan eller hvorfor.
Elever vurderer nye ideer og knytter disse til konklusjoner.	Elever har vanskelig for å forstå nye ideer som er forskjellige fra det de har møtt i læreboken.
Elever forstår hvordan kunnskap blir til gjennom dialog og vurderer logikken i et argument kritisk.	Elever behandler fakta og prosedyrer som statisk kunnskap overført fra en allvitende autoritet.
Elever reflekterer over sin egen forståelse og sin egen læringsprosess.	Elever memorerer uten å reflektere over formålet eller over egne læringsstrategier.

Tabell 1: Dybdelæring vs. overflatelæring. Laget etter Ludvigsenutvalgets (NOU 2014:7 s. 36) oversettelse fra Sawyer (2006, s. 4).

Dybdelæring defineres i retningslinjene for utforming av læreplaner som «det å gradvis utvikle kunnskap og varig forståelse av begreper, metoder og sammenhenger i fag og mellom fagområder» (Kunnskapsdepartementet, 2018b, s.9). Dybdelæring handler om å trene elever i å forstå og bruke kunnskap på tvers av fag og på forskjellige problemer. Når økt fokus på dybdelæringen, programmeringen og algoritmisk tenkning blir kritisert går kritikken på at timetallet for matematikkfaget ikke er økt. Stofftrengselen er en negativ faktor for realisering av relasjonell forståelse ved dybdelæringen. Dybdelæring krever tid og planlegging av aktiviteter slik at elevene skal kunne bygge forståelse. -For det å pugge læringsstoff er ikke

en strategi for dybdelæring. Elevenes tilnærming til stoffet bør være slik at de ikke får instrumentell overflatelæring.

4.2.1 Kognitiv tilnærming til dybdelæring

Gilje, Landfald og Ludvigsen (2019*)* skriver i sin artikkel at kognitiv læringsteori vektlegger utvikling av langtidshukommelsen. Pugg og memorering er derfor en lite egnet læringsstrategi for dybdelæring. Best mulig kognitiv læring for elevene forutsetter at det elevene skal lære settes i en relevant og forståelig kontekst med kjerneelementene i matematikk. Er ikke oppgavene relevante kan de oppleves som isolerte deler istedenfor at de inngår i det store bildet som elevene kan kjenne seg igjen i. Dette kan sees i sammenheng med Piagets konstruktivisme. Säljö (2013) beskriver Piagets syn på hvordan elevene selv utvikler og bygger kunnskap. Kunnskapen utvikles av elevene gjennom to prosesser som han kalte for assimilasjon og akkomodasjon. Elevene fyller på med erfaringer og endring av fakta, får kunnskap gjennom en undersøkende arbeidsmåte og nysgjerrighet uten å være ukritisk til autoritære (læreren) personer (Säljö, 2013). Kognitivismen tok slutt på 1980-tallet da populasjonen ble mer mangfoldig. Ikke alle i samfunnet passet til Piagets syn på utviklingsprosess. Istedenfor begynte Lev S. Vygotskijs (1896-1934) utviklingspsykologi om et sosiokulturelt perspektiv å blomstre.

4.2.2 Sosiokulturell tilnærming til dybdelæring

Den nyanserte metoden for hvordan dybdelæring skjer i et klasserom er annerledes enn med Piagets. Sosiokulturelt perspektiv handler om at elever lærer gjennom kommunikasjon i klasserommet, samarbeid og av lærerens støtte for å appropriere kunnskap og ferdigheter (Säljö, 2013). Det er med Vygotskijs analyse av den proksimale utviklingssone og læreren samt medelever som *scaffolding* at elever utvikler kunnskapen via dialog og instruksjon. Schoenfeld (1987, s. 210) beskriver at eleven på et begrenset nivå skal klare å arbeide alene og å utvide sin nivåsone ved å nyttiggjøre ekstern påvirkning og samarbeide med lærer og medelever. Fordelen med sosiokulturell tilnærming til dybdelæring er at elevenes kognitive utvikling og sosiale tilnærming kombineres som en prosess til læring.
-Men det er uansett viktig at læreren legger opp undervisningen i h.h.t kjerneelementene (Gilje et al., 2019).

I retningslinjene for utforming av læreplaner skriver Kunnskapsdepartementet (2018b, s.9):

«Det innebærer at vi reflekterer over egen læring og bruker det vi har lært på ulike måter i kjente og ukjente situasjoner, alene eller sammen med andre»

Dette viser til samspill mellom det kognitive og sosiokulturelle perspektivet ved dybdelæringen. Sawyer (2006) skriver tilsvarende at dybdelæring handler om å utvikle en forståelse for et fagområde gjennom problemløsning, matematisk tenkning, analyse og refleksjon. Refleksjon gjelder både for egen læring og for elevenes vurdering av om informasjonen de har er reell eller ikke. Piagets kognitive kunnskap handler om at elevene skal kunne reflektere over om informasjonen de har er sann eller ikke (Säljö, 2013). I dagens samfunn, der informasjonen er lett tilgjengelig er slik reflekterende og evaluerende ferdighet viktig. Dybdelæring bidrar til at elevene blir kompetente medborgere (Gilje et al., 2019).

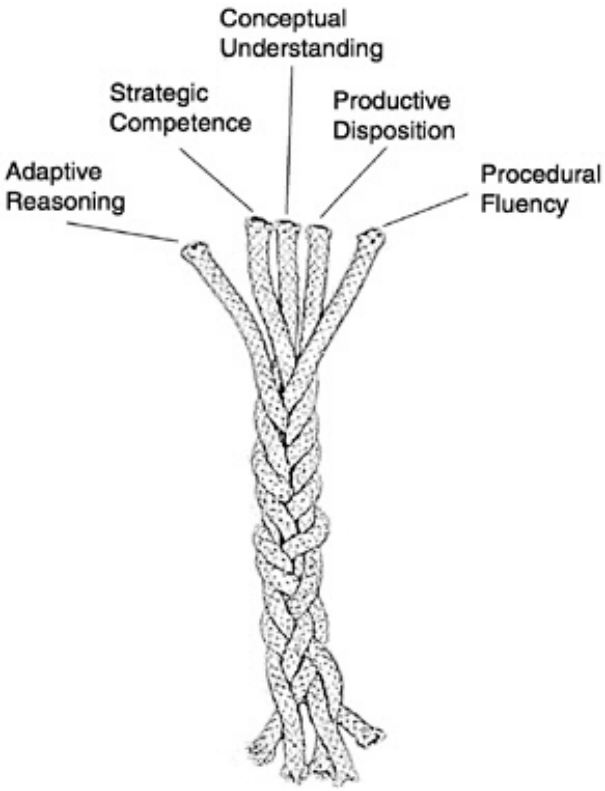
4.2.3 Dybdelæring i matematikk

Hva dybdelæring er i matematikk er vanskelig å definere. I Tabell 1 om dybdelæring og i Kilpatrick, Swafford og Findell (2001) sin flette som beskriver matematisk kompetanse, vises én sammenheng som kan være en måte å definere dybdelæring på. I NOU 2015:8 – *Fremtidens skole* (2015, s.57) beskrives matematisk kompetanse ved hjelp av Kilpatricks et al. (2001) sin flette med de fem tråder (*Figur 4*). Hver av trådene representerer én kompetanse og Kilpatricks et al. (2001) sitt design med fletten viser at for å utvikle matematisk kompetanse må elevene ha kompetanse i alle fem trådene. For å utvikle kompetanse i én tråd må de bruke kompetanse fra de fire andre tråd-kompetansene. De fem trådene er:

1. «Adaptive reasoning» å reflektere, vurdere gyldighet, argumentere, begrunne sammenhenger og se logiske fremgangsmåter og løsninger.
2. «Strategic competanse» det NOU 2015:8 henviser til som *anvendelse*, altså kompetanse i problemløsning ved å kunne formulere matematisk, representere løsningen og utvikle løsningsstrategier.

3. «Conseptual understanding» ferdighet i å forstå matematiske konsepter og se sammenhenger mellom konsepter og hvordan idéer bygger på hverandre samt kunne veksle mellom ulike representasjoner.
4. «Productive disposition» evnen å se nytten og verdien av matematikk.
5. «Prosedural fluency» kompetanse i å gjennomføre oppgaver nøyaktig, fleksibelt og på en effektiv måte.

I tillegg har Schoenfeld (1987) trukket inn metakognisjon fordi det handler om å tilegne seg kunnskap og forståelse gjennom tekning.



Figur 4: Kilpatrick et al. (2001, s.117) Matematisk kompetanseflette

Disse fem kompetansene kan bidra til det som i Tabell 2 karakteriserer dybdelæring i matematikk. Det de to presenterte modellene (Kilpatricks flette og Sawyers kjennetegn på dybdelæring) ikke viser er hva som bidrar til nødvendig egenskap for dybdelæring, matematisk kompetanse og vektlegging på kommunikasjon. Av kompetansefletten kan det tolkes at kommunikasjon er et underliggende tema i «adaptive Reasoning» da elevene skal argumentere, resonnere og beskrive matematikk.

Kommunikasjon er en del av læreplanen i Fagfornyelsen. Læring skjer når elevene kommuniserer med hverandre og lærere både skriftlig og digitalt. Elevengasjement er viktig og bør fokuseres på i undervisningen slik at elevene utvikler matematisk kompetanse og økt dybdelæring. Elever som ikke engasjerer seg i for eksempel gruppearbeid på grunn av egne kompetansehull eller liten interesse for samarbeid vil ha redusert mulighet til å utvikle de andre kompetansene.

4.2.4 Problemløsning

Det er påpekt at det har vært lite forskning på problemløsning i skolesammenheng (Lesh & Zawojewski, 2007). Etter denne påpekningen har Schoenfeld blant annet publisert artikler som omhandler problemløsning i matematikk. Problemløsning blir definert som en matematisk kompetanse som kan brukes for å omforme et problem til matematisk uttrykk og til å vurdere gyldigheten til svaret på problemet (Lesh et al., 2007). Fokus på problemløsning og matematisk kompetanse er viktig for at dagens elever skal bidra i utvikling av moderne teknologi tilpasset dagens samfunn. I NOU 2015:8 (2015) står det:

> «Dagens arbeidsliv stiller høye krav til kompetanse, utdanning, omstillingsevne og tverrfaglig samarbeid. Det er sannsynlig at omfanget av oppgaver som krever kompleks problemløsning og kommunikasjon vil øke fremover og at en rekke rutinepregede og manuelle oppgaver vil erstattes av teknologi.» (NOU, 2015, s. 20).

Dette understreker viktigheten av at elever lærer problemløsning og utvikler gode løsningsstrategier. Men hva er egentlig problemløsning? Historisk sett har problemløsning blitt sett på som «praktisk regning». Før M87 ble innført var problemløsning et eget hovedområde på ungdomsskolen. I L97 ble det sett på som en arbeidsmåte og modellering, i LK06 ble det igjen et eget hovedområde. I Fagfornyelsen er problemløsning og modellering sentrale kjerneelementer.

Som Lesh et al. (2007) diskuterer; med mer fokus på problemløsning i skolen og endringer i læreplanen vil det bli større fokus på forskning på problemløsning. Forskere definerer problemløsning forskjellig, men det er spesielt én definisjon som har befestet seg i matematikkdidaktikken. George Pòlya definerer fire instrukser for problemløsninsstrategier som omhandler hvordan elever skal gå frem for å løse et problem (Lesh & Zawojewski,

2007; Nordlander & Nordlander, 2009; Polya, 1957). Instruksene er: først forstå problemet, så legge en plan, deretter gjennomføre planen og til slutt evaluere og vurdere svaret. For hver instruks anbefaler Pòlya (1957) at man stiller seg noen spørsmål om gjennomføringen. For første instruks er det viktig at problemløseren stiller seg spørsmålet «hva er problemet?», «Hva skal gjøres?» og «Hva er dataene?». Til andre instruks bør problemløseren stille seg spørsmål om man har sett et liknende problem tidligere og kan problemet løses på en enklere måte enn sist. Schoenfeld (1987, s.192) sier «Never use a difficult technique before checking to see whether simple techniques will do the job». Problemløserne må stille seg spørsmålet om det de står overfor nå er noe de har sett før eller med hjelp fra lærer finner ideer til å løse problemet. Instruks tre er å gjennomføre planen. Pòlya (1957) sier her at det er viktig at man fullfører planen som man planla i instruks to. Den siste instruksen, punkt 4, er et punkt som ofte ikke blir prioritert av uerfarne problemløsere. Dette punktet er viktig for å lære av strategiene og metodene som er valgt, vurdere hva som kunne vært gjort på en bedre måte og hva som kan benyttes igjen. Kritikk til Pòlyas fire instrukser er at de sier lite om hva man skal gjøre hvis man står fast. Schoenfeld (1987) advarer og sier at Pòlyas fire instrukser forutsetter at elevene har et visst matematisk kunnskapsnivå for at de skal kunne gjøre mest mulig på egenhånd. Schoenfeld (1987) bruker også metaforen at man skal se for seg at de fire instruksene er fire «personer» som diskuterer en problemløsning. Teknikken kan settes i en sosial sammenheng og er en teknikk elevene må læres opp i. Schoenfeld (1987) viser til at kommunikasjon og sosiokulturelle læringsteori, det å diskutere med andre, bidrar til læring.

Det er viktig å ha gode strategier for å lykkes med problemløsning. Lesh et al. (2007, s. 770) bruker Lester og Kehles sine beskrivelser av gode problemløsere som at elever med høy problemløsningskompetanse lettere finner en løsningsstrategi. De elevene som evaluerer og regulerer sin arbeidsmåte vet at problemløsning ikke er en lineær prosess. De vet at man må «gå» frem og tilbake i strategiene for å komme frem til en god løsning og det bidrar til videreutvikling av matematiske kompetanse.

Kompetansetrådene i Kilpatricks flette avhenger av hverandre for å utvikle hele «fletten» av matematisk kompetanse. Evner eleven å se sammenheng i kunnskapen de besitter har de større mulighet for å lykkes med problemløsning og å få en dypere matematisk forståelse - dybdelæring (Kilpatrick, Swafford & Findell, 2001a).

4.2.5 Problemløsning og programmering

Boccooni, Chioccariello, Dettori, Ferrari og Engelhardt (2016, s. 21) skriver at algoritmisk tenkning og programmering har flere likheter i problemløsninsstrategier. Tabell 2 viser problemløsningsstrategiene til Pòlya (1957) og algoritmisk tenkning ved programmering og gjenkjenner flere likheter for instruksene. Begge tankeprosessene bygger på løsningsstrategier for planlegging, gjennomføring og vurdering av resultat. En annen likhet er at elever må reflektere over hvilken fremgangsmåte som er mest relevant å bruke på problemet de står overfor (diSessa, 2018; Norstein & Haara, 2018; Papert, 1993; Sanne et al., 2016; Wing, 2006). En annen artikkel av Loksa et al. (2016) viser til at å bruke programmering i problemløsningsoppgaver bidrar til økt produktivitet, selvstendighet, mestringsevne og mer effektivt tankesett. Programmering og problemløsning oppfordrer til kreativitet da det ikke er én eksakt vei til målet. Programmering krever systematikk, algoritmisk tenkning og analysekompetanse. Det er gjerne flere mulige løsningsstrategier som kan løse samme problem noe som også gjelder for problemløsning med hjelp av programmering.

	Pòlyas instrukser	Programmerings-«instrukser»
1	Forstå problemet	Dele opp problemet
2	Planlegge	Kjenne igjen regler, formler og lage algoritme
3	Gjennomføre planen	Kode algoritme
4	Evaluere og vurdere	Evaluere og vurdere

Tabell 2: Sammenlikning mellom Pòlyas fire instrukser og programmerings- «instrukser» hentet fra kapittel 2.2

Ved programmerer får man umiddelbar tilbakemelding på om programmet gir rett løsning eller ikke (feil i programmet f.eks syntaksen). For å løse egne programmeringsfeil må man «ta ett skritt tilbake» og finne stedet i programmet hvor koden ble feil - «debugging» eller feilsøking som det heter på norsk. Som ved problemløsning er det å finne feil i koden i programmet en essensiell del av det å programmere for å komme frem til løsningen. Papert (1997) skriver at strukturerte feilsøkingsrutiner er en ting elevene lærer av og forbedrer seg på, elevene må finne nye løsningsveier når de arbeider med problemløsning. Men klarer ikke

eleven å finne feilen eller gjennomføre feilsøking kan det, etter min mening, ha negativ innvirkning på elevene siden de kan miste motivasjon til å jobbe videre. Derfor er lærerens støtte i feilsøking viktig.

4.3 Motivasjon

Flere av informantene fortalte at de fikk økt motivasjon av å jobbe med programmering. Generelt erfares at noen elever får mindre motivasjon og selvtillit ettersom de blir eldre. Det er faktisk vist at på videregående skole er elevers motivasjon og selvtillit lavere enn på barneskolen (Liljedahl & Hannula, 2016). Det er i nyere tid at det er økt fokus og oppmerksomhet på motivasjon i matematikkfaget og begrepene indre og ytre motivasjon er sentrale begreper innen motivasjon (Liljedahl & Hannula, 2016). Ytre motivasjon er gjerne det elevene kjenner på når de vil nå et mål, for eksempel karakter, for å komme inn på drømmestudiet. Indre motivasjon er motivasjon for selve faget og gleden av å lære. Manger (2013) kaller det for egenmotivasjon. Ofte får elevene indre motivasjon av den ytre motivasjon, og det er den indre motivasjonen som fører til et ønske om å forstå faget.

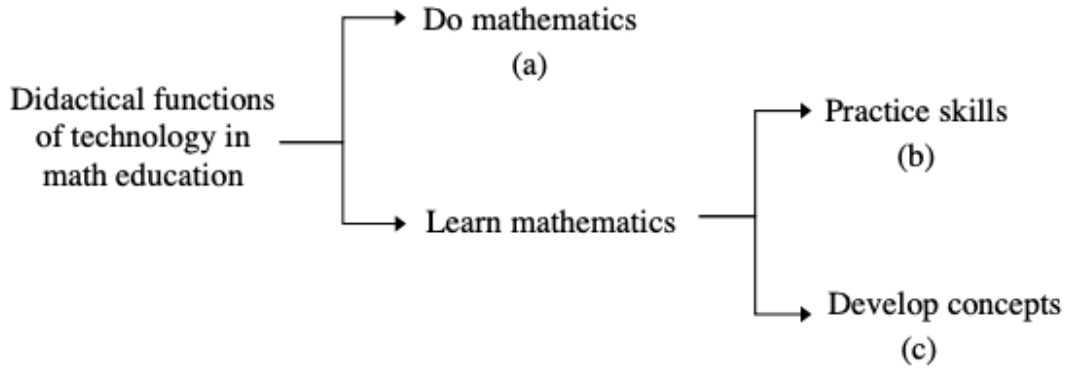
I matematikkfaget snakkes det om instrumentell relasjonell forståelse (I-rasjonelle) og «sosial relasjonell» forståelse (S-rasjonelle) (Mellin-Olsen, 1981, s.357). Instrumentell relasjonell forståelse er bygget på instrukser og beskjeder. Sosial relasjonell forståelse kan sees i sammenheng med sosiokulturell teori der motivasjon er et resultat av menneskelig deltakelse og engasjement med oppgaver (Manger, 2013). Dette er knyttet opp mot den indre- og ytre motivasjonen. Den I-rasjonelle er nært knyttet opp mot ytre-motivasjon og S-rasjonelle mot indre-motivasjon (Mellin-Olsen, 1981). Av dette følger at for å oppnå indre motivasjon trenger elevene en ytre-motivasjon og at elever oppnår relasjonell forståelse i matematikk ved først å opparbeide en instrumentell forståelse. For eksempel, beskriver Skemp (1976) at når elevene lager programmene på egen hånd for å løse matematiske problemer, vil de i mindre grad pugge rutiner og regler til fordel for definisjoner og forståelse.

Motivasjon kan støttes av Kilpatrick et al. (2001a) kompetansetråd *Procedural Fluency* som definerer motivasjon som en kompetanse. Har elevene lav motivasjon kan det blokkere for videre utvikling og for å se nytten av matematikk (Nordlander & Nordlander, 2009). Liljedahl et al. (2016) viser til forskning der det er gitt modellerings- og

problemløsningsoppgaver som har gitt en positiv effekt på elevenes motivasjon. Med programmering kan den indre motivasjonen for eksempel komme av at de får bruke programmering som et verktøy eller det at programmering er nytt og oppleves som meningsfylt. For andre kan programmering være en ytre motivasjon fordi programmering er noe som kan bli krevet av dem i fremtidig utdannelse og/eller arbeid.

4.4 Teknologiske verktøy

Digitale verktøy har lenge vært en del av norske læreplaner. Med innføring av programmering er det nå 4 verktøy elevene skal lære seg uten at timetallet i faget øker. Fordelen med flere verktøy er mulighetene elevene får til å vise matematiske temaer fra forskjellige innfallsvinkler som Kjerneelementgruppen også poengterer. Å ha forskjellige digitale verktøy i skolen er med og bidrar til at elevene får økt og allsidig erfaring med digitale verktøy. Drijvers (2013) skriver om tre didaktiske teknologifunksjoner som kan inngå i matematikkundervisning vist som valgtre i Figur 5.



Figur 5: Didaktiske funksjoner av teknologi i matematikkundervisning. Drijvers (2013, s.3)

Teknologiske verktøy bidrar til at elevene bruker mindre tid på det tekniske ved matematikken og isteden fokuserer mer på løsningsstrategien for å løse problemet («Do mathematics»). (Stacey & William, 2013, s. 748). Bruk av teknologiske verktøy kan være tidsbesparende og gi mer nøyaktige beregninger i for eksempel oppgaver med funksjoner når elevene benytte digitalt verktøy som GeoGebra, CAS eller programmering. Med «Practice skills» eller «Develop concepts» kan teknologi bidra til matematikklæring gjennom å øve på ferdigheter eller utvikle konseptuell forståelse. Bruk av apper og nettsider som tilbyr

oppgaver kan gi elevene trening i regning og forståelse og har den fordel at de får tilbakemelding umiddelbart. Dette bidrar til mengdetrening.

Det er også ulemper med alle de forskjellige teknologiske verktøyene. En ulempe er, som Stacey et al. (2013) skriver, at elever og lærere må vende seg til alle verktøyene. En annen ulempe er at de elevene som er fra sterke sosiale og sosioøkonomiske familier stort sett har en fordel når det kommer til bruk av hjelpeverktøy. Dette kan være fordi disse elevene har mulighet for å få hjelp og tilgang på digitalt utstyr hjemme.

GeoGebra som undervisningsverktøy og elevverktøy er to forskjellige ting. Som undervisningsverktøy har GeoGebra en fordel som et visuelt geografisk verktøy som kan gi elevene et bilde på matematikken. GeoGebra har som Albaladejo, Garcia og Codina (2015) skriver den fordel at det bidrar til samarbeidskultur og bedre forståelse når elevene bruker vokabularet til å diskutere seg frem til matematiske konsepter. GeoGebra som eksamensverktøy kan være komplisert fordi elevene må lære kommandoer som de fort ender opp med å pugge. Oppgaver med bruk av GeoGebra gitt til eksamen satt fokus på det å undervise i verktøyet og ikke bare å bruke det som undervisningsverktøy. Når elevene ikke har lært seg kommandoer fører det fort til at elevene bruker oppskriftsmanualer som er tilpasset oppgaver med GeoGebra gitt på tidligere eksamener.

UDIR har i liten grad deltatt i diskusjon om hvordan programmering skal bli vurdert og det har lærere legitimt reagert på og vist skeptisk til før innføringen. Mange lærere lurer på hvordan det skal undervises i programmering og hvordan programmering skal inngå i eksamen.

Black Box - White Box

Digitale verktøy er et viktig redskap i arbeid med å løse praktiske oppgaver og bidrar til at elevene blir mer matematisk utforskende (Greefrath, Hertleif & Siller, 2018). Det er ikke alle verktøy som er til like god nytte i alle matematikktema. Drijvers et al. (2010) henviser til Buchberger (1990) og skriver om begrepene «White Box» og «Black Box». White Box beskrives som teknologi der elevene må bruke relasjonell matematikkforståelse for å kunne løse et problem med teknologiverktøy. Hvis eleven ikke kan matematikkteorien og bruker innpuggede rutiner på teknologiverktøyet er teknologiverktøyet Black Box (Drijvers et al., 2010, s. 94). Python er et eksempel på White Box. Et eksempel på Black Box er CAS.

Drijvers et al. (2010, s.13) beskriver CAS som et katastrofeverktøy når elevene skal lære matematiske temaer og at Black Box-verktøy en ulempe å bruke når elevene skal lære. For som flere forskere sier er fordelen med digitale verktøy at det styrker elevenes læring ved å engasjere dem i matematiske ideer (Goos, Galbraith, Renshaw & Geiger, 2003, s.73, Drijtev et al, 2010, s. 86). Å bruke Black Box-verktøy kan virke mot sin hensikt hvis det gjør løsningsstrategien til en instrumentell fremgangsmåte uten at det bidrar til relasjonell forståelse. GeoGebra er et nyttig undervisningsverktøy for å gi en visuell forklaring men kategoriseres som Black Box fordi alt er styrt av kommandoer.

5 Metode

I dette kapittelet presenteres studiet før metoden så beskrives. I resten av kapittelet går jeg nærmere inn på metodevalget og hvordan data har blitt samlet inn og bearbeidet. Jeg vil også beskrive datautvalget og til slutt reflektere over validitet, relabilitet og etiske hensyn.

5.1 Programmeringsundervisning i realfagsmatematikk

I denne masteroppgaven er dataene fremskaffet i samarbeid med UPR-prosjektet *Undervisningsopplegg for programmering i realfagsmatematikk*. UPR-prosjektet er et designbasert forskningsprosjekt og undervisningsopplegget er utviklet i et samarbeide mellom prosjektleder (lærer) og meg (student). Prosjektets endelige målsetning er å se på effekten av programmeringsundervisning og utvikle forskningsbaserte undervisningsopplegg ved bruk av PRIMM for å støtte dybdelæring i matematikk på vgs. Fokuset i dette masterprosjektet er å vurdere PRIMM som undervisningsmodell og om PRIMM er egnet til å bruke i matematikkundervisningen. Prosjektet er gjennomført på kort tid og masterstudiet utgjør 30 studiepoeng, så det har ikke vært mulig å utføre en fult kvantitativ studie. Utvalget vil ikke være representativt for alle elever. Studiet vil vise hvordan «noen» elever opplever programmeringsundervisningen i matematikk og utfordringer som oppstod under forsøket. Prosjektleder skal dele erfaringer fra dette utviklingsarbeidet i kurs, veiledning og workshops i regi av utdanningsetaten i fylket skolen tilhører. UPR har mottatt såkornmidler for å fremme innovasjon i lærerutdanning og skole.

5.1.1 Utvalget

Deltakerne i dette studiet er totalt 38 elever og en matematikklærer. Klassene var en 1T-klasse med 25 elever og 13 elever i en R1-klasse og klassene var valgt ut av læreren. 1T-elevene var elever på en teknologilinje og det satses stort på teknologirettet undervisning og matematikk. R1-elevene går på en robotikklinje som har koding og programmering som en del av programfagene. Begge klassene og læreren var fra samme videregående skole. Læreren, som også var prosjektleder, underviste R1-klassen på fast basis i matematikk. 1T-

elevene underviste hun ikke på fast basis i noen fag. Læreren hadde ved gjennomføring av prosjektet fått lov til å bruke noen samfunnsfagstimer til å undervise 1T-elevene i programmering med micro:bit. Det utgjøre én time (45min) i uken.

Læreren samarbeidet med utdanningsdirektoratet om opplæringsopplegg for lærere i programmeringsundervisning i matematikkfaget. Elevene var spesifikt valgt fordi de representerer elever med interesse for teknologi og har hatt innføring i programmering. Når dette prosjektet ble iverksatt var det ikke mange skoler som hadde startet med innføring av LK20 og programmering.

5.2 Forskningsdesign

I dette masterstudiet har jeg valgt å gjennomføre en kvalitativ studie med spørreundersøkelse og intervju. Studiet baserer seg på metodeutvikling av ny læringsteknologi som i seg selv er et designbasert forskningsprosjekt. Designet og spørsmålene stilt i dette masterprosjektet er utviklet i samsvar med forskningen til Sentance et al. (2019) som er beskrevet i kapittel 3.

5.2.1 Observasjon

Siden jeg skulle utarbeide oppgaver og planlegge undervisningsopplegg tilpasset klassen var det nødvendig at jeg først var observatør i noen timer i 1T-og R1-klassen. Som observatør måtte jeg tilpasse meg situasjonen og ikke bli for nær gruppen (Fangen, 2011). Fordeler med observasjon er det Fangen (2011) skriver at som observatør får man samlet data som intervjuobjekter kanskje ikke vil komme til å dele i et senere intervju. Observasjon har den fordel at jeg kan samle kommentarer fra lærer og elever underveis. Kommentarene og observasjonsdataene analyserte jeg og baserte spørsmålene i intervjuguide og spørreskjema på disse. Intervjuet kan da utdype de allerede innhentede data.

Under observasjonen observerte jeg også interessante hendelser som bidro i utforming av forskningsspørsmål nummer to. Fangen (2011, s.99) støtter dette og forteller at observasjon muliggjør *abduktiv tenkning* slik at forskeren underveis kan skape nye forskningsspørsmål av observasjoner man finner interessant.

5.2.2 Intervju

Siden dette er en kvalitativ studie er intervju av læreren en viktig del av forskningsdesignet for å «[...]få innsikt i informantens egne erfaringer, tanker og følelser» (Dalen, 2011, s. 13).

Med intervjuet ønsker man å få mer detaljerte og fremmende svar som kan transskriberes, kodes og bearbeides. Jeg har valgt å bruke et semistrukturert intervju og fordelen med det er å begrense avsporinger. Jeg valgte denne strukturen på intervjuet fordi jeg hadde spørsmål basert på mine observasjoner i timene, elevsvar fra spørreskjemaet og forskningsspørsmål.

Jeg ville også at læreren skulle snakke fritt og dele flest mulig tanker uten at jeg stilte spørsmål. Og når læreren delte sine tanker kunne jeg komme med oppfølgingsspørsmål. For å skape tillit må intervjuobjektet oppfatte at intervjuer er interessert og engasjert (Dalen, 2011; Kleven, 2014). Det er viktig at intervjuobjektet føler seg trygg, derfor fant intervjuet sted på lærerens kontor og det var intervjuer (meg) og informanten som var tilstede.

Intervjuet ble gjennomført tre uker etter den siste av de to forsøkstimene. Strukturen på intervjuet er basert på erfaring fra første analyse av spørreundersøkelsen og forskningsmålene. Siden det hadde gått noe tid mellom siste forsøkstime og gjennomføring av intervjuet brukte jeg litt tid på en kort beskrivelse av temaer som skulle besvares og jeg forklarte hvorfor jeg ønsket svar på spørsmål knyttet til disse temaene. Grønmo (2015) forteller at *retrospektive spørsmål* kan være vanskelige å besvare sammenliknet med samtidsspørsmål siden tema som spørsmålene gjelder har skjedd for en liten stund siden. Noen av spørsmålene er *fremtid spørsmål* om hva som kan forventes med programmeringen i matematikk (Grønmo, 2015).

Intervjuguidens struktur og utforming

Dalen (2011) beskriver at utarbeidelse av intervjuguide bør følge «traktprinsippet»: begynn med overfladiske/åpne spørsmål, dette for å «varme opp» til resten av intervjuet. Deretter suksessivt rette spørsmålene mot det man er ute etter å finne. De første spørsmålene er om lærerens erfaring med programmering. Videre følger spørsmål om tidsbruk og om «programmeringsregnskapet» går opp med tiden det tar å gjennomgå resten av matematikkpensum. Deretter stilles det spørsmål om gjennomført undervisning og om

PRIMM etterfulgt av spørsmål angående læringsutbytte med programmering og overføringsverdier. Det var også spørsmål om hvordan klassen har oppført seg og om elevene ble påvirket av at jeg var tilstede. Avslutningsvis spør jeg om fremtidig bruk av programmeringsundervisning og vurderingsformer.

Intervjuguiden er lagt ved som vedlegg 1 og bestod av 20 spørsmål som læreren ikke hadde fått se på forhånd. Intervjuguiden var utarbeidet sammen med veileder og pilotert på to av mine medstudenter. Under piloteringen fikk jeg testet at teknisk utstyr og spørsmålsformuleringene var gode nok.

5.2.3 Lydopptaker

For å dokumentere intervjuet benyttet jeg lydopptaker. Fordelen med lydopptak er at jeg kan fokusere fullt på det informanten har å si og komme med eventuelle oppfølgingsspørsmål (Dalen, 2011). Med lydopptak unngår man å miste verdifullt datamateriale og med tanke på lengden på intervjuet var lydopptak viktig slik at informantens meninger kom riktig frem.

Intervjuet ble transkribert og analysert i etterkant med NVivo 12 som gjør det lettere å gjøre grundig analyse og gå tilbake å se og høre igjen hva informanten svarte. Når jeg forberedte meg til intervjuet visste jeg på forhånd at jeg var ute etter svar på tidsbruk, undervisningsmodell og overføringsverdi. Med lydopptak kom det frem andre interessante temaer ved analysearbeidet og temaer ble vinklet fra flere sider.

Intervju med lydopptaker kan for flere virker ubehagelig. Så før intervjuet startet jeg med å informere om anonymitet og sikkerhet for at opptaket ikke ville bli misbrukt eller komme på avveie. Godkjenning om bruk av lydopptaker kan sees i søknaden til NSD – *norsk senter for forskningsdata* i vedlegg 3.

5.2.4 Elevenes spørreskjema

Elevene besvarte et digitalt spørreskjema etter å ha gjennomført undervisningstimene i forsøket. Universitet i Oslo har et eget nettskjema[5] som de anbefaler. Nettskjema er tilrettelagt for å lage spørreskjema, enkel distribuering og å lage statistikk ut ifra besvarelsene. Spørreskjemaet var lagt opp slik at elevene ikke kunne identifiseres. IP-adresse ble ikke lagret.

Spørreskjema med strukturert utspørring blir gjerne forbundet med kvantitative forskningsdata (Grønmo, 2015). Med hensyn på at det i dette prosjektet var mange informanter ble strukturert spørreskjema valgt for å få inn svar fra alle elever. Spørsmålene var strukturert i bestemt rekkefølge og avgrenset til bestemte tema basert på prosjektets mål og tidligere observasjoner. Spørsmålene var både åpne og lukkede, på åpne spørsmål var det fritekstsvar og på lukkede var det gitt svaralternativer (Grønmo, 2015).

Spørreskjemaet, vedlegg 2, har totalt 15 spørsmål der noen er åpne og noen er avkrysning. Det er gitt én avkrysningsmatrise med 9 spørsmål. To av spørsmålene hadde ja/nei svaralternativ og ut ifra hva elevene svarte fikk de et åpent spørsmål for å utdype svaret sitt. Spørsmålene omhandlet undervisnings- og oppgavestruktur, innhold, metoder, motivasjon, samarbeid og erfaringer. Spørsmålene som ble gitt var målrettet mot prosjektmålet og skulle bidra med å avklare hva jeg hadde observert tidligere i året. Avkrysningsmatrisen brukte jeg for å gjøre det lettere for elevene å svare og det er også lettere å gjennomføre analyseprosessen.

Fordelen med faste svaralternativer er at innhentede svar kan gjøres om til tall og statistikk maskinelt. 7 spørsmål var åpne spørsmål fordi jeg ville ha mer inngående svar på hvorfor de hadde ment noe. De åpne spørsmålene ga elevene mulighet til å utdype svarene sine og gi meg informasjon. Det er viktig at spørsmålene er strukturerte, har relevans og ikke er for omfattende og for mange, slik at flest mulig fullfører spørreskjemaet.

Fordelen med å bruke spørreskjema er at det er en effektiv måte å inkludere mange i en undersøkelse og det kan innhentes mye data på en gang. Nettskjema kan raskt og enkelt

[5] www.nettskjema.no

gjøres om til statistikk. Negativt er at elevene ikke får mulighet til å uttrykke seg eller komme med sine egne synspunkter på spørsmål med multiple choice avkryssing (Grønmo, 2015). Men med for mange spørsmål og lang undersøkelse kan deltakerne falle fra.

Før første runde med datainnsamling ble det gjennomført pilottesting med fem stykker. De frivillige testerne var tre medstudenter på lektorstudie som selv holder på med masteroppgave, én medstudent på et bachelorstudier i informatikk og den siste er ferdig utdannet lektor. Ut ifra tilbakemeldingene fra testerne fikk jeg som forsker gode innspill fra «nye øyne». De fant formuleringsfeil og ga forslag til omforming av spørreskjema. Ulempe med spørreskjema er at spørsmål kan virke ledende og respondentene kan velge svaralternativ etter hva de tror forskeren vil ha som svar. Spørreskjemaet kan også inneholde kvalitetsfeil.

5.3 Gjennomføring

Dette masterstudiet kom i gang ved at prosjektlederen kontaktet instituttet med informasjon om prosjektet og ønske om en masterstudent. Et møte med prosjektleder, veileder og masterstudent ble gjennomført og innsamlingsmetode for data planlagt. Prosessen begynte med at jeg observerte undervisningen med micro:bit sammen med 1T-klassen i en periode og diskuterte mulige forbedringer av undervisningen med prosjektleder. I R1-klassen deltok jeg som observatør i en undervisningstime før selve forsøkstimen.

Undervisningsopplegget ble redigert og videreutviklet i.h.h.t. det prosjektleder og jeg diskuterte etter observasjon samt i faggruppen prosjektlederen hadde med kollegaer på skolen. Matematiktemaet for oppgaven er funksjoner og datainnsamlingen ble planlagt før første forsøkstime. Oppgavene ble bearbeidet av meg i diskusjon med veileder og godkjent av prosjektleder. 1T-klassen brukte MU[6] som editor. R1 var tidligere kjent med Jupyter[7] og læreren valgte at de skulle bruke den editoren.

Oppgavene ble gitt forskjellig til de to klassene. 1T-klassen fikk oppgavene på et skjema i ItsLearning der de kunne kopiere koden rett inn i tekstboksen etter hvert spørsmål. R1-

[6] https://codewith.mu
[7] https://jupyter.org

klassen fikk oppgavene via Google forms og de skulle også kopiere inn løsningen i en tekstboks etter spørsmålet. Fordelen med ItsLearning er at det er et verktøy elever er kjent med. Det negative med ItsLearning var at det var kronglete for læreren å se hvem som hadde jobbet sammen fordi bare én av elevene trengte å levere oppgavebesvarelse. Fordelen med Google er at elevene selv kan skrive inn hvem de jobbet sammen med og dette ga læreren en bedre oversikt.

Ved første møte i begge klasser informerte jeg elevene om prosjektet, gjennomføringen og taushetsplikten. Læreren og jeg delte ut samtykkeskjema og samlet disse inn. Besvarelsen av spørreskjemaet var for 1T-klassen i november og i januar for R1-klassen. Grunnen til tidsforskjellen for gjennomføring var at prosjektlæreren ikke hadde mer tid tilgjengelig før tentamen til jul. Intervju av læreren ble gjennomført i februar, tre uker etter gjennomført spørreskjema med R1-klassen. Ettersom undervisningen i R1-klassen ble gjennomført god tid etter 1T-klassen kunne vi gjøre nødvendige justeringer i undervisningsopplegget basert på erfaring fra 1T-klassen. Dette kan ha påvirket elevenes erfaring med undervisningen og påvirket hvordan de har besvart spørreskjemaet.

I undervisningsøktene for datainnsamlingene var jeg tilstede som forsker og oppholdt meg helt bak i klasserommet. Før timen begynte informerte jeg en gang til om grunnen til at jeg var tilstede. Læreren startet undervisningstimen med å fortelle hva de skulle gjøre i timen og begynte undervisningen med PRIMM. I begge forsøkstimene altså 1T-klassen og R1-klassen ble undervisningsøkten gjennomført som planlagt dog litt modifisert i R1-klassen basert på erfaring fra 1T-klassen. Klassene hadde det samme tema og fikk samme kode og oppgaver. Gjennom hele timen satt jeg som observatør bakerst i klasserommet og deltok ikke i undervisningen. På slutten av undervisningsøkten svarte elevene på spørreskjemaet, det tok 15 minutter.

5.4 Analyseprosessen

Dette kapittelet beskriver analyseprosessen av dataene jeg samlet inn i prosjektet. Analysearbeidet begynner ved prosjektoppstart med planlegging av hvilke tema og metode som skal brukes. Analyseprosessen begynte med at jeg så på statistikken og elevsvarene fra spørreundersøkelsen. I denne oppgaven brukes det tematisk innholdsanalyse som tilnærming til analysen. Tematisk innholdsanalyse har fordel med å kunne avdekke viktige konsepter og temaer for å videreutvikle eller validere et allerede eksisterende rammeverk. Tematisk analyse er veldig vanlig ved kvalitative data og gir friere muligheter for å analyse gjennomføringen (Fauskanger & Mosvold, 2014). Dette prosjektet tar utgangspunkt i tidligere forskning på undervisningsmodellen og jeg har brukt kodene fra tidligere forskning til dette masterprosjektet.

5.4.1 Transkribering og koding av intervju

Miller og Glassner (2004, s. 126) skriver at intervju er en god måte å diskutere informantens meninger og til å genere data som gjenspeiler deres opplevelser. For å gjenspeile lærerens utsagn transkriberte jeg intervjuet med Nvivo 12. Nvivo har fordelen at du kan spille av lydklippet og markere når utsagn ble sagt med kode for å lettere kunne gå tilbake og sammenlikne data. NVivo 12 har også et eget analyseprogram der man kan se hvor mange ganger en kode er satt ved analysearbeidet og kodene kan flyttes til under en passende kategori.

Jeg utarbeidet på forhånd kodekategorier basert på forskningen til Sentance et al. (2019). Analysekategoriene presenteres i Tabell 3. Jeg brukte også induktiv koding mens jeg gjennomgikk transkriberingen for å se etter nye funn og mønstre som skilte seg fra kategoriene. Kodene tidsbruk og teknisk var nye funn og som ble plassert under kategorien implementering. Jeg lagde en ny kategori om læring etter å ha gjennomgått transkriberingen flere ganger. Mine kategorier og Sentance et al (2019) sine kategorier brukes i vurdering og sammenlikning i diskusjonsdelen.

Kategori:
Implementering
Ferdigheter
Stadier
Differensiering og vurdering
Fremtidig bruk
Emosjonell

Tabell 3: Kodene basert på tema

5.4.2 Analyse av spørreundersøkelsen

Spørreundersøkelsen ble analysert etter Repley (2016) sin anbefaling - å skrive stikkord om interessante funn i margen på utskriftene av elevsvarene. Jeg hadde allerede, ved konstruksjonen av spørreundersøkelsen, sammenfattet hvilke kategorier jeg skulle benytte. Kategoriene er de samme som er brukt i intervjuanalysen i forskningen til Sentance et al. (2019). Etter gjentatte systematiske sammenlikninger av data, tre runder med analysearbeid og koding av spørreundersøkelsen skrev jeg et sammendrag av det jeg mente var de interessante funnene. Sammendraget brukte jeg i utviklingen av intervjuguiden som læreren brukte.

Kodingen er gjennomført på bakgrunn av forskningsspørsmålene, basert på interessante funn underveis i mitt analysearbeid og så tematisert. Det opprinnelige forskningsspørsmål nr. to ble endret underveis i analyseprosessen etter å ha oppdaget tendenser i svarene til elevene som negative utsagn om samarbeidspartner. Jeg fant det nødvendig å inkludere ulemper som en kategori der også teknisk svikt inngår.

Alle tabeller og de kommentarene jeg ser som mest informative og viktige for å belyse forskningsspørsmålene er tatt med i resultatene (Kapittel 6). En viktig del av analyseprosessen var å holde fokuset på hvert enkelt spørsmål og kategori for å unngå å ta inn for mange antagelser (Repley, 2016).

5.5 Forskningskvalitet

I dette kapittelet skal jeg diskutere kvalitet, relabilitet, gyldighet og troverdighet av datamaterialet. Jeg vil også ta opp de etiske perspektivene rundt forskningen.

5.5.1 Validitet

Validitet sier noe om gyldigheten av resultatene. I denne masteroppgaven har jeg benyttet *mixed methods* for å sammenlikne kvalitative og kvantitative data på samme forskningsspørsmål (Patton, 1999, s.1194). Forskningsresultatet blir sikrere når det benyttes flere metoder på samme forskningsspørsmål (Befring, 2015). Metoden er godt egnet fordi det avdekker komplementære aspekter av samme hendelser. Hensikten med spørreundersøkelsen og intervjuet var å belyse forskningsspørsmålene og besvare delspørsmålene og derfor er metodevalgene valide til formålet ved å være brobygger og gi et oversiktlig analysebilde. Oppgaven har troverdighet og gyldighet da forskjellige metoder er benyttet og informantenes uttalelser og formuleringer er eksakt gjengitt og dokumentert. I intervjuet ble det brukt lydopptaker for å kunne transkribere nøyaktig det informanten og jeg sa under intervjuet. Underveis i prosjektet har læreren og jeg vært i dialog og jeg har deltatt som observatør i timene. Dette har vært med på å skape en trygghet og relasjonsbygging slik at læreren kunne svare fritt under intervjuet. Fordel med intervju er at intervjuobjektet ikke gis mulighet til å unngå å svare og kanskje tilpasse svarene til spørsmål og dette øker validiteten til studiet. Intervjuet ble gjort to uker etter siste undervisningstime som var med R1-klassen. Under intervjuet passet jeg på å komme med innledning til hver ny kategori av spørsmål for å friske opp minnet. Det er da viktig som forsker å være nøytral og ikke skape sammenhenger som kan påvirke meningene til informanten. Patton (1999) sier at forskerens bias på emnet kan virke negativt på analysens validitet og troverdighet. Dette ved at forskerens erfaringer, teorigrunnlag før forsøket og opplevelser kan påvirke kvaliteten på data og analyse. Som den eneste forskeren i denne masteroppgaven måtte jeg under analysearbeidet passe nøye på troverdigheten og validiteten på funnene mine. I de tidligere forskningene deltok flere forskere noe som er med på å hemme en enkelt forsker bias (Patton, 1999, s. 1195).

Utvalget var ikke tilfeldig valgt og kan derfor Innvirke på validiteten på resultatene. Utvalget var valgt etter hva læreren hadde tilgang til av elever og som kunne være informantelever. Vi trengte elever som kunne litt om programmering. Det er vanskelig å finne elever da ikke alle

skoler har begynte med slik undervisning i matematikkfaget fordi Fagfornyelsen inntrer først høsten 2020. R1-klassen ble valgt fordi informantlæreren underviser der i matematikk. 1T-klassen ble også valgt ut som en del av prosjektet og dette ble avtalt med lærerne for trinnet og rektor. Elevene kan føle et press til å delta ved at læreren er pådriver. Det skal sies at jeg observerte at elevene ikke følte noe press ved at jeg var tilstede. Læreren svarte også, under intervjuet, at oppførselen og deltakelsen til elevene ikke ble endret selv om jeg var tilstede. Jeg hadde på forhånd prøvd å skape en trygg relasjon ved å være tilstede i flere timer og også småprate litt med noen av elevene.

Mulige kvalitetsfeil ved elevens spørreundersøkelse er at spørsmålene ikke var godt nok formulert og at noen av spørsmålene kunne oppfattes litt "forvirrende". Eksempel på spørsmål som kan lede til kvalitetsfeil er «Jeg liker at vi må forklare koden til hverandre» og «Jeg likte ikke at vi måtte forklare koden til hverandre». Det er spørsmålstekst som er nesten helt identiske og kan derfor misoppfattes. Elevenes svar ga likevel en oversikt over hva elevene syntes om den delen av undervisningen. For å unngå misforståelser med avkrysningsspørsmålene gjorde jeg det mulig for elevene å utdype spørsmål skriftlig. Jeg var tilstede under spørreundersøkelsen så om elevene hadde spørsmål fikk vi avklart det med en gang.

En annen kvalitetsfeil er hvis elevene har løyet eller svart med tull. Skjemaet var anonymt så fordelen er at elevene kan svare åpent og de har forhåpentlig svart ærlig da de også nettopp har hatt undervisningen. Det kan også være en fordel at læreren fortalte elevene at dette er et prosjekt for å teste ut programmeringsundervisning. Læreren informerte også at de har blitt valgt fordi de allerede har kunnskap på området. Dette kan ha gjort at de har følt en stolthet ved å delta.

Det kan være at noen elever tilfeldig har krysset av for å bli fort ferdig med undersøkelsen. Det at jeg lot elevene svare skriftlig ga en indikasjon på om noen elever ikke tok det seriøst og det var særdeles få tullesvar på undersøkelsene. På skriftlige spørsmål med mulighet for å utdype med egen tekst kan man anta at de som ikke hadde utdypet besvarelsen ikke hadde noe å svare utover avkrysningen. Ulempe med spørreskjema er ordskala som alternativ fordi det kan gi tolkningsusikkerhet for respondenter som ikke er flytende i norsk. Man får også en begrenset mulighet med svaralternativer på spørsmål og har man for mange muligheter kan det virke overveldende eller forvirrende.

5.5.2 Reliabilitet

Reliabilitet i forskning viser til hvor troverdige funnene er. Resultatene er basert på koding gjort av meg og på koder brukt i tidligere studier. Reliabiliteten til kodene styrkes ved av funnene er av samme art som i teorien (Fauskanger & Mosvold, 2014). En viktig del av koding og funn er at andre forskere også skal kunne bruke de samme kodene og få tilnærmet likt resultat. Andre forskere skal kunne følge og forstå resultater og argumentasjon (Repley, 2016) så om det blir ulike funn ved flere forsøk kan det oppfattes som tilfeldig hendelser. Undervisningsplanen, intervjuguide og spørreskjema er vedlagt som vedlegg så om andre forskere vil teste resultatene er det mulig å reprodusere min forskning.

Målefeil i spørreundersøkelsen og intervjuet kan være en reliabilitetsrisiko. Jeg ser i etterkant av undersøkelsene, selv med pilotstudier, at det er spørsmål som er noe dårlig formulert. Selv om det mest sannsynlig ikke har gitt stort utslag på analysearbeidet kan det ligge en reliabilitetsfeil i svarene da informantene kan ha misforstått spørsmålet.

5.5.3 Etikk

Det etiske perspektivet har vært viktig i denne masteroppgaven for å bevare anonymiteten til informantene. Jeg ser ikke på kjønn på elevene da det ikke er relevant for denne oppgaven, så fokuset har vært på anonymitet ved å unngå navn, skole og fylke skolen ligger i. Det som har gjort dette vanskelig er ved beskrivelsen av klassene da linjene er særegne for skolen. Elevene fikk muntlig informasjon før forsøket om hva det innebar å delta i forsøket, deres rett ved å trekke seg fra undersøkelsen og hva det innebar å delta. Elevene ble informert om hvilken informasjon som blir lagret, hva informasjonen skulle bli brukt til og at informasjon vil bli slettet og makulert ved prosjektslutt i henhold til NSD, Norsk Senter for Forskningsdata, sine retningslinjer.

Ved prosjektstart ble det sendt inn meldeskjema for behandling av personopplysninger i masteroppgave til NSD. Prosjektets håndtering, spørreskjema og intervjuguide ble godkjent. Elevene fikk utdelt informasjonsskriv og samtykkeerklæring for å delta i forskningsprosjektet i henhold til NSD sin mal. Dette måtte elevene skrive under på hvis de ville delta – alle elevene i klassene deltok.

Informantlærer var også prosjektleder. Dette gjorde det ekstra vanskelig å holde anonymiteten men jeg mener at den har blitt opprettholdt. Under intervjuet ble det brukt lydopptaker etter godkjenning fra NSD. Intervjuet er lagret på kryptert lagringsplass i.h.h.t. UiOs anbefaling og blir slettet ved prosjektslutt. Læreren fikk muntlig beskrivelse av datahåndtering og mulighetene til å trekke seg. Læreren skrev under på samtykkeerklæring etter NSDs mal som vil bli makulert ved prosjektslutt. Begge meldeskjemaene er vedlagt som vedlegg 3.

6 Resultater

I dette kapittelet vil jeg presentere funnene fra intervjuet og spørreskjemaet organisert etter forskningsspørsmålene 1) Er PRIMM en god modell og 2) ulemper og fordeler.

6.1 Informantenes erfaring med programmering

Elevene som var informanter er relativt interesserte i teknologi og programmering da de har valgt å gå på linjer med stort fokus på teknologi. I 1T-klassen hadde 56% av elevene erfaring med programmering før de begynte på videregående skole. I R1-klassen hadde 61.5% erfaring med programmering før de begynte på videregående skole. Svarene på spørreskjemaet viste at Python, Arduino, micro:bit og Scratch var de programmeringsspråkene flest av informantene hadde erfaring med.

Spørreundersøkelsen viste altså at flere av elevene har noe erfaring med programmering fra tidligere. På spørsmålet om hva de tenker om å programmere i matematikktimene svarte de fleste elevene i 1T-klassen positivt på det:

- «Ganske smart. Det kombinerer teknologi og matte som er ganske bra og nyttig»
- «Det er mer spennende enn vanlig matte»
- «Jeg syntes at det er bra fordi det blir mere variasjon i matematikktimene»
- «Jeg tenker at det hjelper meg med å forstå både matte og programmering mer enn jeg kunne før»

Men, spesielt en kommentar var interessant: «jeg synes det er bra, men jeg trenger tid til å jobbe med andre ting heller». Det var også én elev som skrev: «jeg tenker det vil bli komplisert fordi det er ganske nytt å gjøre.».

I R1-klassen ga mange elever tilsvarende kommentarer som i 1T-klassen. Alle er ganske positive til å innføre programmering i matematikkfaget, men det kommenteres av én elev at «jeg syns det er vanskelig, men lærerikt». Ellers har flere elever kommentert «tilsvarende» disse to svarene:

- «Veldig bra verktøy i matte, da mye av programmering er matte»
- «Problemløsning og matte henger godt sammen, og er derfor meget godt passende.»

Læreren har sin erfaring med programmering fra sin utdannelse som ingeniør der hun tok fag som krevde numeriske beregninger. Hun har også erfaring fra IT-grunnkurs og Matlab og forteller at hun valgte noen av fagene i utdannelsen sin fordi de fikk programmere og hun syntes det var gøy. Hun forteller at informantelevene fra R1-klassen vil ha mer erfaring med programmering enn vanlige R1-elever. 1T-klassen har bare brukt micro:bit på videregående skole, men det er flere i den klassen enn i andre klasser som kan litt programmering fra før slik at spennet på programmeringserfaring i informant klassene kan være større enn det vil værer i andre klasser

6.2 Resultater ved bruk av PRIMM som rammeverk

Første forskningsspørsmål i prosjektet var å undersøke om PRIMM kunne fungere som en modell for lærere med lite eller ingen erfaring i programmeringsundervisning. PRIMM er som sagt tidligere utarbeidet av Kings University i London for å være et rammeverk for lærere med lite eller ingen erfaring i programmeringsundervisning.
Resultatene fra forskning i England viser at PRIMM har vært et vellykket utformet rammeverk til bruk både i grunnskolen og videregående skole. Videre presenteres relevante resultater og utsagn gitt fra lærer og elever om PRIMM og stadiene: Predict, Run, Investigate, Modify og Make.

6.2.1 Lærerens utsagn om stadiene i PRIMM

Læreren forteller om fordelene ved å anvende stadiene i PRIMM rammeverket i timene og spesielt om fordelen med strukturen og gjennomføringen.

«Jeg syns det fungerer bra for å holde fokuset på hva de skal lære. Fordelen med PRIMM er at man kommer så fort i gang. Fordi man kan få presentert ganske mye kjapt, den klassiske er at du som lærer bare skriver et program foran elevene og forklarer alt, men det blir «too

much». Da er de kanskje mer opptatt av å skrive av også, så det er lurere å bare vise hele koden og ta tak i noe av det, og så kan de skrive av alt og så kan de prøve selv og undersøke om de forstår alt.»

Flere lærere bruker oppgaver i programmering tilgjengelig på nett for å tilegne seg kunnskap i å løse matematikkoppgaver med programmering. Men for elever kan nettlæring være en terskel og det er lett å bli distrahert. Lærerne kommenterer den lærerstyrte delen av PRIMM-modellen:

«Jeg føler at denne modellen er god på den måten at den er mer lærerstyrt enn tutorial på nett der man ikke har noe kontroll på om de faktisk leser det som står mellom eksemplene for elevene skal bare bli ferdige. Da er alle med på gjennomgangen og er med på å ta pauser for å tenke litt og diskutere.»

Læreren kommenterte at predict-stadiet er bra for alle elevene uansett hvor dyktige de er i programmering fra før:

«De som trenger litt forklaring på hva en «for-løkke» er får det, men ofte oppdager jeg at de som tror de kan det egentlig ikke har skjønt hva de gjør, de har bare gjort det så mange ganger at de tror de vet det. De vet f.eks. ikke hva en «range» gjør, eller hva en «int» betyr eller hvorfor du må ha punktum. Jeg tror det kan være fint for de også at de flinke får en oppklaring. Også kan de føle seg flinke ved å klare å svare på spørsmålene i klassen»

Investigate-stadiet, der elevene forklarte koden, fortalte læreren at hadde fungert godt. Hun mener også at å forklare kode er egnet som hjemmearbeid. Hun har tidligere gitt i lekse at elevene skulle lage en video der de forklarte hva koden gjør: «det fungerte å forklare koden og elevene fikk vist hva de hva de kunne og ikke kunne.». Læreren kunne se på alle videoene og få en oversikt over hva elevene hadde forstått og elevene fikk en mulighet til å teste seg selv.

Modify-stadiet av PRIMM indikerte læreren at ikke alle elevene rakk å bli ferdig med og alle fikk derfor ikke prøvd seg på *make*-stadiet. Hun har noen tanker om hvorfor elevene ikke ble ferdig med modify-stadiet:

«Jeg fikk ikke alle til å levere inn, de brukte for lang tid så økten var kanskje litt for kort. Så jeg kunne kanskje vært enda klarere for det fungerte mye bedre i 1T da de skulle svare underveis på oppgavene. Jeg var ikke tydelig nok med beskjedne i R1, så det var veldig mange som ikke leverte inn noe selv om de kanskje hadde prøvd litt så leverte de ikke inn.»

Lærerens uttalelse om *make*-stadiet for de elevene som ble ferdig med modify-stadiet:

«I R1-klasseen var det noen elever som gikk videre ved å ta utgangspunkt i min kode og løste den med likninger der man kan få flere svar og da jeg har skrevet litt tungvint kode for å hente ut de ulike svarene men noen elever ser da «men kan vi ikke bare bruke løkker». Så det er noen som blir motivert og tar det videre og da blir det at det er programmering de gjør og ikke matematikk akkurat der. Men det gir jo en del forståelse.»

Undervisningsopplegget var lagt opp slik at elevene skulle levere oppgavene fortløpende ved å «copy paste» koden inn i tekstsvarboksen i Google-skjemaet for R1-elevene og ItsLearning-skjema For 1T-elevene. Læreren utdypet fordelen med denne måten:

«Det er også greit at de leverer inn oppgavene som vi gjør digitalt for da forsvinner ikke ark som det ofte ellers gjør i bøker. Eller at de deler link til det de har gjort og jeg kan gå inn senere og se hva de har gjort.»

P*redict*-stadiet, Tabell 4 viser svarprosenten på tre utsagn gitt til begge klassene.
Tabellen viser elevsvarene som prosentvise svar. Elevene skulle krysse av i en tabell som henviser til stadiene i PRIMM på utsagn om undervisningen. Elevene skulle gradere utsagnene ut ifra hvordan de likte undervisningen på en skala fra 1 til 6, der 1 var «ikke enig» og 6 «helt enig».

1T						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Å få vist frem kode på tavlen gjør at jeg lærer hvordan koden fungerer.	**0%**	**4%**	**8%**	**32%**	**40%**	**16%**
Jeg likte at vi må forklare koden til hverandre.	**4%**	**12%**	**8%**	**12%**	**36%**	**28%**
Jeg likte ikke at vi må forklare koden til hverandre.	**20.8%**	**16.7%**	**29.2%**	**16.7%**	**4.2%**	**12.5%**
R1						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Å få vist frem kode på tavlen gjør at jeg lærer hvordan koden fungerer.	**0%**	**0%**	**15.4%**	**23.1%**	**30.8%**	**30.8%**
Jeg likte at vi må forklare koden til hverandre.	**0%**	**0%**	**7.7%**	**15.4%**	**15.4%**	**61.5%**
Jeg likte ikke at vi må forklare koden til hverandre.	**36.4%**	**36.4%**	**0%**	**9.1%**	**9.1%**	**9.1%**

Tabell 4:Svarprosent fra elevsvar om predict-stadiet. Karakter 1 beskriver «ikke enig» og 6 «helt enig».

For *run*-stadiet varierte besvarelsene fra elevene i 1T- og R1-klassen noe, elevene i 1T-klassen var mer fornøyd med å kunne skrive koden selv. Tabell 5 viser hva elevene mente om utsagnene.

1T						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg likte at vi må skrive koden selv (ikke copy paste inn).	**4%**	**8%**	**4%**	**12%**	**20%**	**52%**
R1						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg likte at vi må skrive koden selv (ikke copy paste inn).	**7.7%**	**7.7%**	**0%**	**15.4%**	**7.7%**	**61.5%**

Tabell 5: Svarprosent fra elevsvar om run-stadiet. Karakter 1 svarer til «ikke enig» og 6 «helt enig».

Investigate-stadiet, en elev skrev at det var lærerikt å sitte sammen to-og-to og kommentere hver kodelinje. Informantelever fortalte også at å være to gjør det mer effektiv og så nytten i å ha en samarbeidspartner spesielt hvis en stod fast.

Stadiet etter *investigate* er *modify*-stadiet der elevene skulle modifisere koden for å løse oppgaver. Elevene ble gitt ett utsagn og deres besvarelse vises i Tabell 6.

1T						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg lærte mer ved å modifisere koden for å løse de vanskeligere oppgavene.	**12%**	**0%**	**20%**	**24%**	**24%**	**20%**
R1						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg lærte mer ved å modifisere koden for å løse de vanskeligere oppgavene.	**0%**	**7.7%**	**0%**	**30.8%**	**46.2%**	**15.4%**

Tabell 6: Svarprosent på elevsvar i modify-stadiet. Karakter 1 svarer til «ikke enig» og 6 «helt enig».

Det siste stadiet i PRIMM er at elevene på egenhånd skal skrive en kode for å løse utfordringsoppgavene. I R1-klassen ble de færreste ferdig eller kom ikke noe særlig i gang, det var bare én gruppe som leverte utfordring to. I 1T-klassen var det tre grupper som leverte utfordring to. *Tabell 7* viser resultat over hvor langt de kom i stadiet.

1T						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg fikk mulighet til å lage min egen kode fra bunnen av på slutten.	**30%**	**4%**	**24%**	**12%**	**8%**	**16%**
R1						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg fikk mulighet til å lage min egen kode fra bunnen av på slutten.	**0%**	**46.2%**	**15.4%**	**15.4%**	**7.7%**	**15.4%**

Tabell 7: Svarprosent for elevsvar i make-stadiet. Karakter 1 svarer til «ikke enig» og 6 «helt enig».

De fleste elevene var fornøyde med lærerens gjennomgang i timen.

Tabell *8* viser elevenes svar.

1T						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Hvordan synes du lærerens gjennomgang var i dag?	**0%**	**8%**	**16%**	**16%**	**44%**	**24%**
R1						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Hvordan synes du lærerens gjennomgang var i dag?	**7.7%**	**7.7%**	**0%**	**15.4%**	**38.5%**	**46.2%**

Tabell 8: Svarprosent fra elever om undervisningsgjennomgangen. Karakter 1 svarer til «ikke enig» og 6 «helt enig»

Som læreren kommenterte følte hun at tiden ikke strakk helt til og at hun hadde for store ambisjoner for hva de skulle rekke. Ser vi på svarprosentene på elevsvarene på spørsmålet om de fikk nok tid til å jobbe med oppgavene er det en forskjell på 1T- og R1-klassen. Flertall av elevene i begge klasser mente de fikk nok tid, men i R1-klassen var det 30.8% som mente det gikk for fort frem. Tabell 9 viser at det bare er 7.7% i R1-klassen og 16% i 1T-klassen som mente de ikke fikk nok tid fordi de ikke har forstått programmeringen

Fikk du tiden du trengte til å følge undervisningen?	**1T**	**R1**
Ja	**64%**	**53.8%**
Nei, gikk for fort.	**16%**	**30.8%**
Nei, har ikke helt forstått det å programmere i Python.	**16%**	**7.7%**
Nei, annen grunn.	**4%**	**7.7%**

Tabell 9: Svarprosent på utsagn om tid til rådighet.

Ut ifra elevenes besvarelse ser vi at elevene er fornøyde med hvordan undervisningen er gjennomført, men at det var litt knapt med tid. Elevene ga litt forskjellige svar på om de likte å forklare koden til hverandre. Det kan ha en sammenheng med at samarbeidet fungerte dårlig eller om elevene hadde hatt tekniske utfordringer. Dette vil det bli sett nærmere på i diskusjonskapittelet.

Læreren hadde tilrettelagt for bruk av to forskjellige editorer slik at elevene kunne bruke den editoren de var vant med fra tidligere. 1T-klassen brukte Mu og R1-klassen brukte Jupyter. Begge editorene, Mu og Jupyter, fikk noen elever tekniske komplikasjoner med noe som førte til at de ikke fikk tid til å arbeide med det de egentlig skulle. Læreren forteller videre at skoleeieren har gitt de (på den tiden vi arbeidet med prosjektet) en for gammel versjon av programmet de bruker. Elevene kan derfor ikke installere nyere versjon selv. Læreren utdyper rundt dette og jeg har valgt å sensurere skolens datasystem for å holde på anonymiteten til skoleeier:

«Nå i R1 prøver jeg å introdusere Sympy og det er litt mer jobb for plutselig er det PC-er det ikke er forhåndsinstallert på, av en eller annen grunn, og da må vi gjøre det og da krasjer det med [redacted] og da blir det plutselig vanskelig.»

Tiden det tar å installere de nødvendige programmene på elevenes PCer for å kunne bruke Python kan sammenliknes med tiden det tar å installere GeoGebra. Det er kjent at i begynnelsen tar tekniske «ting» tid, både for elever og lærere spesielt når programmer skal tas i bruk for første gang og ved installasjon på PCer som ikke har en versjon av programmene installert. Under intervjuet ble læreren spurt om «*programmeringsregnskapet*». Altså hvor mye tid som vil bli brukt på tekniske løsninger og innføringen i Python og hvordan dette vil gå opp med tiden til rådighet i matematikkfaget. Læreren tenker seg at det vil ta tid i starten:

«Og så tar det jo litt tid sånn i starten for du må få lastet ned programvare og det er litt teknisk dette programmet vi velger å bruke, -og det tar jo litt tid. Men det tar jo ikke mere enn én økt kanskje å få det tekniske på plass.»

Læreren har estimert at tiden på å installere og introdusere grunnleggende Python-syntax vil være 45 minutter på installasjon og 45 minutter på introduksjon. Hun kommenterer at tiden som brukes på installasjon og introduksjon må brukes av matematikktimen: «Og det er jo

ekstra som man kanskje ikke hadde gjort ellers.». På spørsmålet om i hvilken klasse det hadde «kostet» mest tid å introdusere tekstbasert programmering svarte læreren:

«Det må ha vært 1T-klassen. Men der har vi ikke brukt mattetimen, men andre timer. Vi har tatt tiden fra litt forskjellige fag men integrert det inn i samfunnsfag som var det faget vi tok litt tid av ved å lage valg-o-mat på micro:bit.»

Læreren peker at det er det tekniske som vil ta tid i starten. Fordelen for dagens elever er at de kan en del fra før, men det vil ta ett års tid å komme inn i notasjonen og bibliotekene til Python:

«Det blir å prøve og vise dem at det ikke er noe ekstra de må lære. Man må holde det litt enkelt, man må ikke forklare alt men strippe koden mest mulig for å ikke bruke flest mulig spesialtegn. Det greie med Python er at det er ganske enkel notasjon, man trenger ikke vise dem hvor man finner «krøllparentes».»

Læreren sier at unge i dag sitter en del foran pc-en og kan det med å skrive på tastatur og det å finne tegnene de skal bruke ligger i fingrene allerede. Men, det betyr ikke at de vet hva tegnene betyr og hva det gjør i tekstbasert-programmering. Læreren hadde brukt 45 minutter i uken i 1T-klassen på å lære elevene programmering. Hun har også planer om å bruke 45 minutter i uken i matematikktimen i R1-klassen på å gjøre oppgaver med Python. Planen til læreren er at i fremtiden skal hun kun bruke Python og ikke vise CAS i GeoGebra eller GeoGebra til elevene. Hun begrunner dette valget med:

«Da blir det ikke mer tid som går da, men hvis du skal gjøre begge deler så går det jo mer tid. Da kan man si at man «taper» tid men jeg tror ganske mange av dem vil bruke Python frivillig hvis de enten kan bruke GeoGebra/CAS eller Python - altså de som er i R1 som er interessert, si ⅔ av den klassen.»

Som en del av PRIMM inngår arbeidsmåten parprogrammering. Læreren trekker frem at fordelen med at elevene jobber i par med én PC er at «sannsynligheten for at de har en fungerende pc blir større».

Begge klassene var vant med å kunne samarbeide med medelever i matematikktimene. Matematikklæreren, som til vanlig underviser i 1T-klassen, organiserer undervisningen slik at

elevene nesten alltid gis anledning til å samarbeide, helst med den de sitter nærmest og opplever at de aller fleste også gjør det. Av og til deler læreren inn i grupper på 3-4 elever som arbeider sammen om én eller flere oppgaver.

Informantlæreren underviser til vanlig i matematikk i R1-klassen. Hun organiserer timene til vanlig slik at det er kun én individuell oppgave i hver time de øvrige oppgavene gjøres i samarbeid. Elevene samarbeider vanligvis med den eleven de sitter sammen med, men de blir ikke stoppet om de går til en annen medelev i rommet.

Elevene hadde delte meninger om hvordan det var å samarbeide om oppgavene. Flere av elevene i begge klassene uttalte at de ikke fikk noe hjelp fra partneren sin. Det var enten fordi partneren ikke ville bidra eller at partneren ville gjøre alt selv.

Fra 1T-klassen gjengis noen av kommentarene til de som ikke likte å samarbeide:

- «Partneren min forstod ingenting og ville ikke jobbe»
- «Det var fordi personen jeg satt med ikke var interessert i å samarbeide med meg og ville gjøre alt selv. Derfor mister jeg motivasjonen til å jobbe. Jeg fikk heller ikke lært så mye pga. han hadde pc-en nesten hele tiden for seg selv»
- «Var for vanskelig, skjønte ikke så mye»

Og til noen elever som syntes det var positivt å samarbeide:

- «Fordi man kunne spørre den andre om man lurte på noe.»
- «Jeg liker å gjøre ting selv, men akkurat i slike tilfeller hvor man skal tenke litt, foretrekker jeg å tenke sammen med noen.»
- «Jeg får hjelp når jeg står fast på hva jeg skal gjøre.»
- «Jeg likte det fordi hvis jeg ikke vet om en ting, kan partneren min vite det. Det er enklere å komme frem til et svar sammen.»

Det var 84% som svarte «Ja» på at de likte å samarbeide med partneren om oppgaver og 16% svarte «Nei». På et annet spørsmål på spørreskjemaet kom det frem meninger fra elever vedrørende samarbeid som: «Det kunne ha vært gøy, men i mitt tilfelle ville jeg heller vært alene.».

I R1-klassen svarte de også ganske likt på hvordan de trivdes med å jobbe i par der, 84.6% av elevene svarte «Ja» på at de likte arbeidsmåten og15.5% svarte «Nei», altså 2 stk. De positive tilbakemeldingene var alle om fordelen med å samarbeide og at de kunne diskutere og lære av hverandre. Jeg gjengir to tilbakemeldinger som beskrev det jeg ser som positivt og gjenspeiler det de andre positive meldingene også beskrev:

- «Liker å snakke, tenker best når jeg kan kommentere på alt jeg gjør verbalt.»
- «Fordi føler man kan diskutere og skjønne mer og lære av hverandre».

Fra de negative tilbakemeldingene er det interessant å se at flere kommenterte negativt på det å samarbeide enn det er som svarer «Nei» i Tabell 10:

- «Læringspartneren min fulgte ikke med og jeg synes det er bedre å jobbe alene med programmering så lenge jeg får hjelp hvis det er noe jeg sliter med»
- «Det er fint hvis man jobber med noen som ikke tuller.»
- «Jeg likte ikke par siden man kan gjøre mer ting alene + kjappere alene»
- «Syntes det var ok, men mener vi burde lært mere grunnleggende Pyton før det»

Læreren forteller vedrørende samarbeidet at det er lettest at elevene arbeidet med eleven som sitter nærmest fordi det gjerne er en plan med hvor elevene sitter i klasserommet.

6.3 Informantenes meninger om dybdelæringen med PRIMM.

Kjerneelementgruppen i matematikk (2017) ønsker at arbeidet med «tall og tallforståelse» gjøres grundig og over en lengre periode slik at det blir mindre behovet for repetisjon. De mener dette vil gi mer rom i timeplanene slik at elevene kan møte flere tilnærminger til stoffet som for eksempel til programmering. Informantene ser positivt på dette og påpeker at det vil bli varierende arbeidsmetoder og mer utforskende.

6.3.1 Lærerens syn på dybdelæringen

Læreren ser for seg at hun kommer til å bruke Python-programmering på temaene sannsynlighet, funksjoner og numeriske metoder. Men hun ser ikke for seg at programmering vil hjelpe på algebrakunnskapene. For oppgaver i matematikken mener læreren at bruk av programmering kan bedre matematikkoppgavene ved at: «Det kan bli mer utforskende».

I dag brukes GeoGebra mye i skolen fordi det inneholder graftegning-funksjon og CAS. GeoGebra er typisk i bruk ved temaene funksjoner og differensiallikninger. Læreren mener det er fordeler med å kunne vise elevene flere innfallsvinkler. Hun utyper at fordelene med programmering er at det er kodebasert og derfor mer forståelig og spesielt i temaet funksjoner som elevene jobbet med i prosjektet. Hun mener at programmeringen i Python gir en matematisk tankegang som er annerledes enn GeoGebra:

«Det har vi andre mattelærere snakket litt om angående funksjoner at elevene sliter skikkelig med å forstå hva den grafen er. For når de jobber i vanlig graftegneprogrammer, så skriver du inn funksjonen og «boff» er grafen der og det betyr ikke at de har skjønt hva som skjer eller hva grafen viser men at det er bare en mengde punkter. Når vi bruker Python så må man sette opp hvor mange punkter man skal ha. Det er litt mer synlig hva som skjer i programmet. Så vi tenker at det kanskje gir mer forståelse av funksjoner når man legger opp til sånne oppgaver. Det er litt for «lett» i graftegneprogrammer - det blir bare å skrive inn, det blir bare metode. Det kan bli ekstremt instrumentelt at elevene kan tegne grafer i GeoGebra, men de klarer ikke å skjønne hva de holder på med.»

Fra lektorutdanningen i realfag på Universitetet i Oslo vet vi at det å vise forskjellige fremgangsmåter og metoder bidrar til å bedre matematikkforståelsen. Læreren påpeker at det er viktig at lærerne er didaktisk interessert i egenutvikling og interessert i programmering. Slik at programmering ikke blir en ting de underviser i rett før eksamen. Undervisningen må ikke gjøres for instrumentell. Det er du som lærer som må vise at bruk av verktøyene GeoGebra, gjøre det «for hånd» og med Python-programmering gir samme løsning.

Læreren forklarer at hun hadde observert at elevene syntes det var matematikken i oppgavene som var vanskelig, ikke programmeringen. Selv har hun kommenterer at en del av pakkene til Python kan bli for vanskelig i starten i forhold til CAS i GeoGebra som elevene er vant med:

> «Ja, eller jeg bruker Python som CAS. Men, da må vi bruke *SymPy*[8] og det er ikke på nybegynner listen egentlig, så jeg må prøve å løse det for det finnes ikke så mye ferdig materiale og det står ikke noe i de nye lærebøkene -de norske. Så da må man nok lage mer opplegg selv. Men det er jo da det virkelig blir nyttig for nå kan elevene løse likninger med ferdig verktøy og det er egentlig ikke noe vanskelig.»

SymPy er et Python-bibliotek som gir deg muligheten til å arbeide med symbolsk beregninger. Det er en åpen-kode slik at hvem som helst kan laste det ned (SymPy Development Team, 2020). SymPy kan for eksempel importeres på de tre måtene som er vist i Figur 6: Tre eksempler på importering av sympy. Med eksempel nummer 1 importerer du hele biblioteket og det kan gi komplikasjoner hvis du importerer andre biblioteker som også inneholder noen av de samme funksjonene. Eksempel nummer 2, linje to i Figur 7, importeres Sympy som sp-benevning for å bruke bibliotek Sympy sine funksjoner. Eksempel nummer 3 importerer kun de ønskede funksjonene til koden din. Læreren poengterer at bruken av ferdige biblioteker hemmer elevene i å bygge forståelse, de lærer mer av å skrive fremgangsmåten selv. Det er ikke mye læring i å «klikke og lime» i et ferdig læringsprogram.

```
1) from sympy import *
2) import sympy as sp
3) from sympy import exp, solve, Poly, cos
```

Figur 6: Tre eksempler på importering av sympy.

Python-programmering kan bidra til dybdelæring og læreren poengterer at det har sine fordeler når det gjelder utvikling av elevenes algoritmiske tenkning og utvikling av problemløsninsstrategier:

[8] Sympy er et Python-bibliotek som gir deg muligheten til å arbeide med symbolsk beregninger. https://www.sympy.org/en/index.html

«Det er jo det matematikk egentlig dreier seg om, å bruke egen logikk og det tror jeg programmering er veldig nyttig til om det er Python eller noe annet. At elevene må planlegge, de kan ikke bare begynne å skrive uten å ha tenkt ut en strategi og logisk løsning, det å bryte ned oppgavene i moduler er veldig viktig. Selv er jeg litt opptatt av at de skal finne «bitene» selv. Ved bruk av GeoGebra er det bare «magi» når tallene skrives inn (Black Box) også forstår ikke elevene hvorfor de får feil i CAS fordi de ikke skjønner hvorfor de noen ganger må skrive semikolon og noen ganger ikke. De har for eksempel ikke forståelse for definisjoner og ulikheter og mulighet for å ta det opp igjen f.eks. «hvorfor har man et likhetstegn her og to likhetstegn her».

Læreren understreker at det er vanskelig å si på nåværende nivå i prosjektet om programmering har gitt elevene en dypere forståelse.

6.3.2 Utsagn fra elever om dybdelæringen

Elevene fikk spørsmål om de hadde fått noen dypere forståelse for andregradsfunksjoner ved å programmere oppgavene. I 1T-klassen svarte 64% av elevene «Ja» og i R1-klassen svarte 84.6% av elevene «Ja». Noen av elevene kommenterte videre hvorfor de mente de hadde fått dypere forståelse. Fra 1T gjengis sitatene:

- «Det hjalp litt men ikke så mye fordi makkeren min var ikke samarbeidsvillig. Men jeg så litt hvordan andregradsfunksjoner kan løses på pc.»
- «Jeg kunne utforske funksjonen mer enn vanlig»
- «Hvis jeg jobber på forskjellige måter lærer jeg bedre»
- «Jeg fikk endre forskjellige ting av funksjonen i programmet så jeg fikk se hva de ulike tingene gjør»
- «Fordi jeg fikk et bedre inntrykk av grafer»

Fra R1 gjengis følgende kommentarene på hvorfor elevene mente de fikk dypere forståelse:

- «Viser at hvor nøyaktig en graf er, er basert på hvor mange punkter som er kalkulert.»
- «Måten man kan endre på koden og se resultater gir meg god forståelse»
- «Man lærer hva Geo gjør»

Fra besvarelsene til elevene i 1T-klassen som svarte «nei» gjengis kommentarene som jeg så som relevante for hvorfor elevene ikke syntes programmering bidro til bedre forståelse:

- «For vanskelig»
- «Jeg forsto ikke akkurat mer av andregradsfunksjoner. Jeg har en tanke om hvordan

Fra besvarelsene til elevene i R1-klassen som mente de ikke bidro til bedre forståelse gjengis følgende:

- «Meh kom ikke langt fordi jeg ikke liker Jupyter»
- «Fordi jeg brukte mye tid på å forstå kodene»
- «Kom ikke så langt, føler jeg.»
- «Litt usikker på hva som skjedde»

6.4 Differensiering

Læreren synes at oppgavene var godt differensiert fordi de flinkeste i matematikk og programmering gjennomførte oppgavene raskt og fikk også prøvd seg på oppgavene i «make»-stadiet. Hun legger til at selve PRIMM-modellen også er godt differensiert og fungerte bra for å holde elevene fokusert på det de skulle lære.

I 1T-klassen var det 88% som syntes oppgavene var utfordrende nok. 12% mente oppgavene ikke var utfordrende nok og de kommenterte at de ikke gjorde oppgavene, eller at de bare ikke likte å gjøre oppgaver. Av de elevene som svarte ja gjengis noen elevmeninger om hva de likte best med oppgavene:

- «Vi brukte timen til å forstå hvorfor vi gjør det vi gjør og det er viktig.»
- «At de fyller alle eksempler og spørsmål som du har.»
- «Hvordan jeg må vise at jeg forstå oppgave»
- «Jeg har erfaring med programmering, men ikke Python, derfor hadde jeg en bred forståelse på hva hver linje gjorde, men jeg visste ingenting om fundamentet til hvordan man koder i Python. Ved å gjøre oppgavene, lærte jeg litt om Python og hva som gjør dette språket ulik fra andre språk.»

- «Vi måtte utforske koden og skjønne hva den gjorde»
- «At vi fikk «programmere»»

Det var flere elever som kommenterte at de likte oppgavene fordi de fikk programmere. Selv om de fleste av informantene liker å programmere setter de pris på at de får utfordringer som bidrar til bedre forståelse. Altså ikke bare i programmering men også en annerledes måte å løse matematikkoppgaver på.

For elevene i R1-klassen var det 76.9% som svarte «Ja» på at oppgavene var utfordrende nok og 23.1% svarte «nei». De som ikke likte oppgavene kommenterte at oppgavene var: for lette, monotone eller for lite koding. Fra besvarelsene til de elevene som svarte «ja» gjengis fire elevsvar om hva de syntes var best med oppgavene:

- «At vi fikk i oppgave å finne uttrykkene selv»
- «Prøvde å gi meg bedre og dypere forståelse»
- «Finne ut av syntaxene.»

Læreren poengterer at det vil være «lettere» å differensiere undervisning og oppgaver i en klasse med mindre faglig spenn på elevene. Hun sier at det vil være lettere å differensiere i en klasse der «ingen» kan noe programmering fra før. Men det skal bemerkes at elevene i 1T-klassen ikke hadde jobbet med programmering for å løse matematikkoppgaver tidligere. Læreren mener det er mattekunnskaper det står på når de skal løse oppgaver.

6.5 Motivasjon og samarbeid

Motivasjon og mestring henger sammen i matematikkfaget og det var lite variasjon i motivasjonen til elevene i begge klassene. Det var én elev som svarte negativt på spørsmålet om de ble motiverte til å jobbe med matematikk når de fikk programmere. De øvrige elevene svarte positivt. 5 elever i R1-klassen svarte:

- «Jeg ble motivert til å lære meg generell programmering, eventuelt også i matte-sammenheng, fordi det kan være lettere enn GeoGebra.»

- «Ja, det å lære Python er interessant og morsomt, og gjør det mer spennende å jobbe med matte»
- «Ja, samme ting som å bruke LaTeX for å føre oppgaver enn å bruke annet. Dette gjør det mer spennende og versatile.»
- «Ja ble mere motivert fordi programmering er morsomt og gir mere forståelse i matte»
- «Ja, kan bruke det til å lære mer programmering»

I T1-klassen var det tre elever som svarte negativt på om de ble motiverte til å jobbe med matematikk når de fikk bruke Python.

- «Nei, forstår ikke gjennom Python»
- «Nei, jeg synes det er lettere å bruke GeoGebra til å tegne grafer»
- «Nei»

Noen av de som var positive svarte:

- «Ja fordi det er noe nytt.»
- «Ja, fordi det var mer å spennende å gjøre noe jeg interesserer meg for istedenfor å bare gjøre masse regnestykker.»
- «Ja. Siden jeg liker å programmere, blir det mer spennende å kunne programmere i mattetimen, i tillegg til at jeg lærer et nytt kodespråk.»
- «Ja men samtidig så må man pugge et helt nytt bibliotek, umulig å vite hva man skal skrive for å gjøre noe.»

Læreren forteller at i R1-klassen var det noen elever som gikk videre til *make*-stadier og ble motivert av å kunne lage sin egen kode for å løse oppgavene. Hun kommenterer at i informant-klassene er elevenes motivasjon som oftest i programmeringen og ikke i matematikken. Hun påpeker videre at selv om elevene er mest motivert til å programmere er det hva de gjør av matematikk i koden som er viktig: «Målet er jo at de skal jobbe med algoritmisk tenkning og matematikk.».

Elevene besvarte et Ja/Nei-spørsmål vedrørende erfaring med parprogrammeringen med påfølgende tekstspørsmål der elevene skrev mer utypene om hvorfor de svarte «Ja» eller «Nei». Tabell 10 viser svarprosenten.

Likte du at dere jobbet i par med oppgaver?	**1T**	**R1**
Ja	**84.%**	**84.6%**
Nei	**16%**	**15.4%**

Tabell 10: Svarprosent fra 1T og R1: Likte du at dere jobbet i par med oppgaver?

For 1T-klassen gjengis følgende utsagnene for positiv tilbakemelding:

- «Var gøy fordi man kunne reflektere»
- «Det var kult Jeg får hjelp når jeg står fast om hva jeg skal gjøre.»
- «Fordi man lærer av hverandre»
- «Jeg likte det fordi hvis jeg ikke vet om en ting, kan partneren min vite det. Det er enklere å komme seg til et svar sammen.»
- «Jeg får hjelp når jeg står fast om hva jeg skal gjøre.»

Under er det listet opp noen utsagn fra T1-elevene som var negative:

- «Partner hjalp ikke men itpd» («itpd» - ikke tenk på det, red.am.)
- «Jeg synes det var positive og negative sider med det.»
- «Man lærer programmering som kommer til å bli nyttig senere, men jeg trenger mer tid på andre ting.»
- «Jeg vet ikke, Partneren min forstår ingen ting og ville ikke jobbe»
- «Det var fordi personen jeg satt med ikke var interessert i å samarbeide med meg og ville gjøre alt selv. Derfor mister jeg motivasjonen til å jobbe.»
- «Jeg fikk heller ikke lært så mye pga. han hadde pc-en nesten hele tiden for seg selv.»

Fra R1- klassen gjengis følgende utsagnene som utdyper de positive svare:

- «Var fint å finne fram til ting ved å kombinere kunnskapen når man kan prate med hverandre om man lurer på noe.»

- «Det er en mulighet for oss å teste ut ulike ting i programmet og diskutere programmering.»
- «Fordi føler man kan diskutere og skjønne mer og lære av hverandre fordi i samarbeid kan man utveksle kunnskap og analysere oppgavene.»

Negative tilbakemeldinger fra R1-klassen:

- «Syntes det var ok, men mener vi burde lært mere grunnleggende Pyton før det»
- «Jeg likte ikke par siden man kan gjøre mer ting alene + kjappere alene»
- «Det er fint hvis man jobber med noen som ikke tuller.»
- «Læringspartneren min fulgte ikke med og jeg synets det er bedre å jobbe alene med programmering så lenge jeg får hjelp hvis det er noe jeg sliter med»

6.6 Videre bruk av PRIMM

Alt i alt er informantene positive til bruk av Python i matematikkfaget. De er innforstått med at det vil være en omfattende omstilling for lærere og elever i starten på videregående skole og det må prøves og feiles. Læreren påpeker at PRIMM derfor er en god modell i starten:

«Man må teste ut litt ting, tenker det er en god innføring å bruke PRIMM, men når man har holdt på en stund kan man gjøre om litt og tilpasse. Man kan bruke PRIMM hele året eller valgfri hvordan man skal gjøre oppgavene. Jeg syns det fungerer bra for å holde fokuset for hva de skal lære»

Oppgavene i PRIMM er strukturert slik at elevene skal skrive inn kode fortløpende under spørsmålene. En utfordring var at elevene ikke kopierte inn koden i svarfeltet eller at editoren ikke fungerte. Læreren kommenterte på det at hun prøver å finne en editor som er bedre egnet for alle klassetrinn. Hun trekker inn at elever ofte har gode forslag og skal høre med dem hva de tenker. Som en forbedring av hvordan elevene får «utlevert» og besvarer oppgavene foreslår læreren en forbedring på eksisterende læringsarena:

«Så om de kunne fått bygd inn en editor i ItsLearning eller noe liknende slik at du kunne laget din egen tutorial eller oppgavehefte. For da kan man gjøre mer med oppgavene etterpå for den konsolideringen på slutten fikk jeg ikke helt til.»

Hvordan eksamenen blir er også avgjørende for hvordan og i hvilken grad det blir undervist i Python-programmering fremover. Læreren mener at det ikke vil bli et krav å skulle bruke kodebasert pseudiokode[9]. Det er fordi Fagfornyelsen 2020 pålegger ikke å undervise i programmeringsspråket Python. Læreren kommenterer at:

«De kan ikke få en oppgave om å finne feilen i et språk/syntax for det er ikke sånn læreplanen er. Det skal være en kompetanse og ikke verktøyet de skal testes i. Men de kan teste om de forstår hvorfor Pseudokode er feil»

[9] Uformell beskrivelse av et dataprogram eller en algoritme som kan realiseres i ulike programmeringsspråk. (https://snl.no/pseudokode)

7 Diskusjon

Med utgangspunkt i mine funn skal jeg diskutere forskningsspørsmålene:

- Er PRIMM en undervisningsmodell som kan brukes til programmering i matematikk på vgs.?
- Hvilke utfordringer må lærere være klar over for at undervisning i programmering skal bli vellykket?

Jeg vil, med fokus på resultatene, først diskutere stadiene i PRIMM, så differensiering og motivasjon. Deretter diskuterer jeg utfordringer med innføringen av Python-programmering i matematikk-klasser som har stort fokus på tilrettelegging for teknologibruk sammenlignet med andre studiespesialiserende klasser. Jeg vil avslutningsvis ta opp fremtidige suksesskriterier.

Fra informantdataene erfarer vi at modellen er godt strukturert men det var noen negative kommentarer om samarbeid og noen om tekniske problemer. Elevene påpeker hvor viktig det er at læreren er interessert i programmering, noe som kan tyde på at det ikke er selve modellen som er det viktigste. Det er viktig at læreren er interessert i å lære programmering, lære bort programmering og gi elevene en variert undervisning ved å vise forskjellige innfallsvinkler til matematiske temaer.

7.1 PRIMM som undervisningsmodell

Basert på informantenes utsagn og meninger skal jeg i dette delkapittelet diskutere stadiene i PRIMM. Norske elever vil ha mindre tid til rådighet til programmering enn elever i England, så tid er en viktig faktor for en vellykket undervisning. Det er derfor behov for en god forskningsbasert undervisningsmodell som er tilrettelagt for norske undervisningsforhold med fokus på tid og læringsutbytte. Med utgangspunkt i informantenes syn, tilbakemelding og erfaringer skal vi se på de fem stadier i PRIMM.

7.1.1 PRIMM som undervisningsmodell

Informantene mener at programmering er en viktig kompetanse som de trenger i arbeidslivet. De anser det som et skritt i riktig retning at programmering innføres i læreplanen i norsk skole. Noen informanter mener også at programmering passer godt i matematikkfaget fordi matematikk og problemløsning «hører» sammen. Dette støttes av forskere som hevder at programmering forbedrer problemløsningsferdighetene. Forskere sier også at programmering har overføringsverdi til algoritmisk tenkning (Mayer, Dyck & Vilberg, 1986, s.609). Programmering er en strukturert og organisert måte å tenke på og denne tenkemåten kan også overføres til andre fag som matematikk. Det er også forskere som sier det ikke lar seg påvise at kompetanse i programmering gir bedre ferdigheter i problemløsning (Mayer et al., 1986). Men det er gjennomført ny forskning og «computational thinking» er blitt et begrep (Wing, 2006) som er oversatt til norsk som «algoritmisk tenkning» (Sevik, 2016). Fra kapittel 6 vet vi at «computational thinking» handler om mer enn bare kognitive ferdigheter som: arbeidsmåter, digital kompetanse og tolkning og analyse for å anvende programmering og matematikk.

Forskere er usikre på om det er en sammenheng mellom programmering og problemløsning. Selv om informantene mente det var en sammenheng mellom programmering og problemløsning og noen forskere er uenige i det, er ikke dette nødvendigvis en konflikt. For hva informantene mener er hva de føler der og da, nemlig at de får bedre problemløsningskunnskap. Det kan være at den måten informantelevene lærer programmeringen på gjør at de føler det har en sammenheng med problemløsning i matematikk. Det kan være andre ting som førte til at elevene fikk bedre problemløsningsferdigheter som hvordan du lærer programmering og problemløsning. De som mener det er liten sammenheng mener det ikke er en åpenbar overgang. De benekter nok ikke at det er en sammenheng men at overføringsverdien er liten.

Informantene er positive til PRIMM som undervisningsmodell. Selv om PRIMM er utviklet for brukt i Computational Science i England kan vi trekke parallell mellom resultatene til Sentance et al. (2019) og utsagnene til informantene i denne masteroppgaven. PRIMM har flere stadier og elevene skal forklare koden, utprøve, evaluere og endre kode gjennom samarbeid og muntlig aktivitet (Sentance et al., 2019). På internett finnes det utallige

nettressurser utviklet for å lære programmering men som krever selvstendig fremdrift og læringsvilje. Med nettlæring, som læreren også uttaler seg, må elevene være selvstendige og strukturerte. Læreren poengterer også at det vil være vanskelig for en lærer å følge opp elevenes progresjon hvis man brukte nettlæringsressurser. PRIMM derimot er en ressurs som en grunnmodell å ta utgangspunkt i for planlegging av undervisning.

7.1.2 Stadiene i PRIMM

Samfunnet endrer seg og det er behov for nye undervisningsmodeller tilpasset samfunnsutviklingen, læringsbehovet og læringsmåter. Ludvigsen og Mørch (2011) beskriver at undervisning med digitale hjelpemidler bør være læring som en sosiokognitiv prosess. Videre skriver de at elevene må lære å identifisere nye måter å løse oppgaver på. PRIMM er en undervisningsmodell som baserer seg på elevsamarbeid, muntlig aktivitet og tett oppfølging og støtte av lærer. Vygotskijs sosiokulturelle læringsteori støtter opp om hvordan elevene best lærer med interaksjoner og støtte av lærer. Lærerens egeninteresse i faget, støtte og assistanse til elevene er viktig for at elevene skal nå neste ferdighetsnivå.

Til det første stadium i modellen; *predict,* er tilbakemeldingene fra informantene veldig positive. Læreren var særlig fornøyd med at stadiet er lærerstyrt slik at elevene får en felles gjennomgang og elevene får en oversikt over arbeidet de skal gjøre. Stadiet gir muntlig trening i å forklare og undersøke siden elevene skal forklare koden verbalt. Predict stadiet gir elevene gode muligheter i å bruke sine muntlige ferdigheter og å lære av å forklare hverandre. Lye og Koh (2014) skriver at det ikke har vært fokusert nok på muntlig aktivitet i programmeringsundervisningen. I PRIMM-modellen derimot er det fokus på muntlig aktivitet. Læreren mener den muntlige delen i PRIMM har sin fordel i at uansett elevenes kunnskapsnivå så kan elevene hjelpe hverandre til en bedre forståelse. Elever bygger på egen forståelse og kan avdekke eventuelle egne misoppfatninger når de forklare til medelever. Dette støttes av Vygotskijs teori; at læring skjer gjennom sosial aktivitet (Schoenfeld, 1987).

Det er også forsket på at det å lese tekst høyt bidrar til at elevene blir mer fokusert, husker bedre og forbedrer tekstforståelsen (Swidan & Hermans, 2019). Dette er argumenter som støtter opp om at elever burde lese koden høyt og samarbeide for å lære. I artikkelen til

Swidan og Hermans (2019, s. 178) skrive de videre om hvordan «reading aloud» vil føre til at elever som er ferske i programmering vil huske syntax og dermed kode bedre.

Run-stadiet; alle informantene mener at å skrive av koden selv var positivt. I forskningen til Sentance og Waite (2017b) vil de at elevene skal «copy-paste» koden for å unngå at de skriver feil. Vi valgte å gjøre det motsatte. Læreren argumenterte for valget med at elevene lærer å skrive kode og får det inn i fingrene ved å skrive på et tastatur. Men med avskrivning av kode vil det være noen elever med lese- og skrivevansker som kan lese og skrive av koden feil. Det kan løses ved å gi disse elevene unntak noe hver enkelt lærer må avgjøre.

Investigate-stadiet gir elevene en ekstra gjennomgang med «reading aloud» og økt kognitiv forståelse ved å la de kommentere hver linje i programmet sammen. En av elevene poengterte: «to hjerner tenker bedre enn en», og har da forstått den fordelen muntlig aktivitet og samarbeid om oppgaver gir også når det gjelder å nå neste kunnskapsnivå.
De fleste elevene likte å bli vist kode på tavlen og å diskutere og forklare i klassen. Noen elever kan føle at dette er en unødvendig aktivitet hvis de selv mener de har god kontroll på hva kodedelene betyr. Selv om elevene kan fortelle hva kodedelen gjør betyr ikke det alltid at de har forstått konseptet og hva hele koden utfører. Å skrive kommentarer på hver linje, selv for den «letteste»-kodedelen, gir også elevene en liten repetisjon. I 1T-klassen varierte det noe om elevene likte å forklare koden høyt til partner eller ikke. Selv om elevene ikke «liker» å forklare høyt betyr det ikke at de ikke ser det som en nyttig læringsaktivitet. Fra kapittel 3.2 vet vi at gruppesammensetningen kan påvirke hvordan det arbeides i en gruppe. Grunnen til at elever ikke liker å forklare for hverandre kan være at kunnskapsforskjellen mellom elevene i gruppen gjør at noen føler ubehag når de ikke å svare riktig.

Informantene hadde varierende meninger om *modify*-stadiet. R1-elevene ytret at de fikk en følelse av læring når de modifiserte koden de hadde fått for å løse oppgavene. I 1T-klassen, jfr.

1T						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg lærte mer ved å modifisere koden for å løse de vanskeligere oppgavene.	**12%**	**0%**	**20%**	**24%**	**24%**	**20%**
R1						
Karakter:	**1**	**2**	**3**	**4**	**5**	**6**
Jeg lærte mer ved å modifisere koden for å løse de vanskeligere oppgavene.	**0%**	**7.7%**	**0%**	**30.8%**	**46.2%**	**15.4%**

Tabell 6, varierer meningene og tekstsvarene viser at for noen bidro programmeringen til økt forvirring og andre svarte at det var matematikken de ikke helt forstod. Det kan tyde på at hvis elevene står overfor to oppgaver og de ikke har kontroll på begge kan det innvirke negativt på hele aktiviteten. Oppgavene i *modify*-stadiet fikk de aller fleste gruppene gjort. Men det var ytterst få som fikk levert ferdig produkt for *make*-stadiet.

Tabell 7 viser at det er lite variasjon blant elevene i de to klassene på hvor mange som fikk gjort oppgavene i *make*-stadiet. Det kan ha sammenheng med det læreren sier om lite tid til rådighet og at hun selv følte hun kunne gitt tydeligere informasjon om innhold ved utlevering av oppgavene. Flertallet av elevene mente at lærerens gjennomgang var bra. Oppgavene som var utarbeidet til *make*-stadiet var ikke nummerert med kalt «Utfordring 1» og «Utfordring 2». Elevene kan ha antatt at det var frivillig å løse oppgavene eller så ble oppgaven for vanskelige. Skal man bruke PRIMM til dybdelæring er det i *make*-stadiet elevene får fordypet seg. For som en elev kommenterte angående oppgavene følte hun at de bare endret på variabler og hun kan ha følt at det ikke bidro til noen fordypning. Det må sies at oppgavene kan gjøres mer avanserte ved at elevene selv må endre mer i koden for å fkomme frem til løsninger.

Med Pythonprogrammering får elevene beskjed på skjermen hvis programmet de har laget ikke fungerer. Elevene får altså en rask tilbakemelding på hva som er galt og de kan fortsette å arbeide videre sammen for å løse problemet. I store klasser kan det være krevende for en lærer å være innom alle grupper og gi tilbakemeldinger. Svarene på spørreundersøkelsen viser at elever også slet med teknisk løsning og trengte assistanse fra lærer for å løse dette.

Ved oppstart vil det også være noen som trenger mer hjelp enn andre for å komme i gang. Med Python og parprogrammering kan elevene sammen prøvd å finne feilen og rette den opp noe som er en fordel med parprogrammering i metodeverket PRIMM.

Lærer og elever var positivt til modellen. Informantene sier at læreren må være interessert og engasjert for å gjøre undervisningen spennende og nyttig. Min tolkning av dette er at det er ikke modellen i seg selv som gjør undervisningen god. Modellen støtter, i mine øyne, opp under det Utdanningsdirektoratet sier om undervisningsvurdering; «Læraren skal vere i dialog med elevane om utviklinga deira i programmering og strategiar for å løyse problem» (Utdanningsdirektoratet, s. 6, 2019a). Selv om dialog mellom lærer og elev bør gjelde for hele matematikkfaget, så er PRIMM en modell som legger godt til rette for læring og dialog. Læreren kan se på besvarelsene og avdekke eventuelle misoppfatninger og vurdere hva de så må jobbe mer med.

7.2 Tidsperspektiv for innføring av programmering

Det var lite som tilsa at programmering ville inngå som teknologisk verktøy i matematikkundervisningen når man leser om Fagfornyelsen (Kjerneelementgruppen for matematikk, 2017a). Det er derimot fokusert på å redusere matematikkpensumet for å gi plass til nye elementer i læreplanen slik at alle innspillspersoner «blir tilfreds». Læreplanene gir ofte gode argumentasjoner for hvorfor endringer er nødvendige men, etter min mening, gir konklusjonene ikke alltid svar på argumentasjonen. Reduksjon av matematikkpensumet skal gi mer plass til implementering av dybdelæring og bedre fokus på tallforståelse gjennom hele utdanningsløpet (Kjerneelementgruppen for matematikk, 2017b). Innføring av programmering, i tillegg til verktøyene GeoGebra, CAS og regneark, vil medføre at elevene skal lære fire digitale verktøy i løpet av sin obligatoriske skolegang.

Det er argumentert for at programmering vil gi elevene en ny innfallsvinkel til matematikkfaget og trening i algoritmisk tenkning. Når programmering innføres i matematikkfaget er det viktig at «programmeringsregnskapet» går opp med resten av pensum i matematikk. Det vil ta tid å introdusere elevene for et nytt verktøy noe læreren også støtter

opp om. Det er noe variasjon i programmeringsregnskapet, som læreren utarbeidet for de to klassene, for hvor mye tid som ble brukt på innføring og det må poengteres at elevene i disse to klassene er over gjennomsnittlige opptatt av teknologi.

For mange elever vil tekstbasert programmering være noe helt nytt og ukjent og tiden det vil ta å implementere vil variere fra klasse til klasse. Læreren var optimist på at «programmeringsregnskapet» skulle gå opp sett i sammenheng med resten av matematikkpensumet. Hun så for seg 2x45 minutter for installasjon og innføring av de viktigste syntaksene, gjennomgang av terminaler og kjøring av programmer. For 1T-klassen brukte hun timer fra samfunnsfag til å jobbe med micro:bit og tverrfaglig oppgaver men det vil mest sannsynlig ikke være anledning til på andre skoler. R1-elevene har allerede kunnskap om programmering da de har det som et programfag så der brukte ikke læreren mye tid på innføring. Fremover vil nok programmeringsregnskapet som er beregnet for 1T-klassen være mest realistisk siden elevene som kommer fra ungdomsskoler som har jobbet godt med programmering ikke vil trenge så mye tid på oppstart.

Elevene som begynner på ungdomsskolen høsten 2020 vil lære blokkbasert programmering og det vil gi elevene bedre grunnlag for overgangen til tekstbasert programmering. Elever som har foreldre med erfaring fra informatikk, hatt programmering som valgfag på ungdomsskolen eller gått på *Lær kidsa koding* vil ha et fortrinn når de begynner på vgs. Kompetansevariasjon blant elevene kan gjøre innføring i programmering mer krevende og generer behov for differensiert undervisningen og varierende tidsbruk. Som læreren forteller har hun selv brukt en del tid på å lære seg det elevene skal lære og brukt tid på å lage passende oppgaver. Man kan ikke utelukke at selv om matematikkpensumet er redusert, så vil det ta tid å lære programmering både for elever og lærere. Programmering kan ikke bare ses i et vakuum fra resten av pensumet.

7.3 Dybdelæringen

Om Python vil fremme dybdelæring eller ikke har informantene ingen klare uttalelse om. Det er for lite grunnlag og for tidlig i prosjektet til å konkludere på dette. Dette støtter også læreren. Selv om elevene har uttrykt at forskjellige fremgangsmåter gir bedre forståelse er det fagpersoner som vil avgjøre hvordan programmeringen gir eller vil bidra til dybdelæringen.

Fordelen med implementering av programmering i matematikkfaget er at programmering gir nye undervisningsmuligheter og som noen informanter sier at undervisningen vil bli mer variert og utforskende. Med programmering kan elevene løse større og mer komplekse problemer enn tidligere fordi programmeringen gjør at de får mer nøyaktige data og kan bruke større tallmengder.

En elev ga tilbakemelding på læringsutbytte og uttalte at ved å bruke Python fikk hun sett at funksjoner består av mange punkter når de brukte for-løkker sammenlikning med bruk av GeoGebra da det bare er «magi» som læreren kaller det, elevene skriver inn funksjonsuttrykket og grafen vises. «Magien» er det som kalles Black Box altså at elevene ikke får brukt matematikken bak modelleringen (Kapittel 4.4). Programmering som *kan* gi elevene en mer relasjonell forståelse og føre til bruk av færre instrumentelle rutiner og regler kalles White Box. Skemp (1976) støtter opp om dette og skriver at elever bruker definisjoner og sin forståelse når de løser matematiske problemer på egenhånd. Ser vi på *make*-stadiet i PRIMM er dette et godt eksempel på dybdelæring da elevene må kunne matematikken for å løse de vanskeligste problemene med egenskrevne programmer. Nødvendige ferdigheter vil elevene tilegne seg etterhvert som de programmerer og læreren kan etter hvert som elevene tilegner seg ny kunnskap gi mer komplekse oppgaver.

En elev uttalte at «Problemløsning og matte henger godt sammen og passer derfor meget godt sammen.». Problemløsning og programmering er, som vi vet fra teorikapittelet, basert på ganske like problemløsninsstrategier. Programmeringen kan derfor hjelpe elevene i å trene på problemløsning og med det tenke logisk og kanskje bygge bedre matematisk forståelse. En fordel med å integrere programmeringen i matematikkfaget er at elevene kan løse mer avanserte oppgaver med bruk av programmering enn med andre verktøy og elevene vil få mulighet til å arbeide med mer virkelighetsnære og komplekse matematiske oppgaver.

Undervisningen blir mer variert når et matematikktema blir gjennomgått fra forskjellige innfallsvinkler og med et bredere repertoar av metoder. Dette stemmer overens med det dybdelæringen handler om, å anvende kunnskap. Det bekrefter også en elev med utsagnet: «hvis jeg jobber på forskjellige måter lærer jeg bedre». Men, om undervisningen blir så variert som det er tenkt etter Kjerneelementgruppens begrunnelse for flere innfallsvinkler, kan man bare spekulere i. Det kan bli tidkrevende å gjennomgå temaer fra flere innfallsvinkler og som om læreren forteller vil hun gradvis gå over til å kun å bruke Python.

Flertallet av informantelevene hadde erfaring med programmering fra tidligere. Det er noe som antagelig ikke vil gjelde for de fleste elever ved innføring av Fagfornyelsen høsten 2020. Flertallet av norske elever vil bruke mer tid på å lære seg programmeringen og dette kan, sett i mine øyne, føre til mindre tid til fordypning i matematikk det første året med Fagfornyelsen. Det må også vurderes om det er hensiktsmessig å vise flere innfallsvinkler da det tar tid med dybdelæring i hvert tema både for fordypning og tverrfaglig arbeid.
Elevene som mente programmeringen ikke bidro til dypere forståelse kan ut fra tilbakemeldingene også ha utfordringer med å forstå Python. *Tabell 7* viser at veldig få elever ferdigstilte eller fikk begynt å lage sin egen kode og til å fordype seg i temaet. Dette er et interessant funn fordi informantlæreren er engasjert og elevene interesserte i å lære og gjerne bruke programmering. Man skulle tro at de ville kommet lenger med oppgavene. For en gjennomsnitt-klasse kan man da anta at enda færre elever vil få muligheten til å fordype seg.

Det kan virke som at det i Fagfornyelsen er lagt opp til at de faglig flinkeste elevene vil klare seg best i starten av innføringen av Fagfornyelsen. Hvis elevene som går på en teknologisk faglig sterk skole og på teknologiske linjer har problemer, kan man bare se for seg hvordan det vil være for gjennomsnittseleven. En annen fallgruve kan være, som lærerne forteller, at for å få løst noen oppgaver i Python må elevene laste inn biblioteker som for eksempel Sympy. Et problem med Sympy er for eksempel at elevenes datamaskiner ikke har de nødvendige oppdateringer for å kunne benytte seg av nødvendige biblioteker. For som læreren sier så er det skolesystemet som ikke oppdateres hurtig nok med det elevene trenger. Læreren poengterer også at GeoGebra, som elevene bruker nå, er et instrumentelt verktøy med ferdig kode. Ved å utlevere ferdig kode mister elevene litt av læringspoenget, programmeringen er borte og verktøyet blir instrumentelt. Læreren snakker om vektorer, at det blir for komplisert og man mister den forståelsen hvis vi bare importer et script fra et bibliotek. Arbeidet med å få elevene på det nivået at de forstår konseptet er en omfattende oppgave for lærere og spesielt for de lærerne med lite erfaring i programmeringsundervisning. Lærerne må bruke mye tid på planlegging og å utarbeide varierte og utforskende oppgaver for slik undervisning.

Når hovedansvaret for opplæring i og bruk av programmering legges til matematikkfaget kan det føre til at programmering blir nedprioritert eller det blir undervist i på slutten av skoleåret fordi læreren mangler kompetanse eller ser programmering som ukultur (Sanne et al, 2016,

s.76). En fordel med at matematikkfaget har fått hovedansvaret for programmering er at matematikkfaget blir mer inkludert i andre skolefag som i dette prosjektet hvor 1T-elevene også programmerte i samfunnsfag.

7.4 Digitale hjelpemidler

Læreren mener det vil gå med mye tid hvis de skal fortsette å lære elevene alle de teknologiske verktøy som er i faget i dag. Hun sier videre at etterhvert vil hun kun bruke Python til graftegning og som CAS. Python er et mer matematisk hjelpemiddel og kan gi elevene en annen vinkling enn GeoGebra og CAS har gjort til nå. Læreren poengterte at fremgangsmåten med GeoGebra som matematikkhjelpemiddel er mindre intuitivt enn med Python. Det var litt uenighet mellom elevene om Python var et bedre verktøy å bruke enn GeoGebra for plotting av funksjoner. Flertallet hadde Python som favoritt og som de sier er Python mer intuitivt å bruke, de får nytte av kodespråket og de lærer seg å programmere. Det blir også mer variert og mer interessant å lære når elevene ser en nytte i å kunne programmere. De som var negative svarte også at de ikke hadde forstått språket, implementeringen eller slet med matematikken. For å få optimal utnyttelse av de forskjellige verktøyene må elevene ha kunnskap om hvilke verktøy som er best egnet til å løse de ulike oppgaver. Skal elevene optimalt kunne utnytte alle de forskjellige verktøyene må de vite hvilket verktøy som vil løse oppgaven mest effektivt. Det er derfor viktig at læreren introduserer verktøyene på en hensiktsmessig måte slik at elevene har tilstrekkelig kunnskap til å velge riktig fremgangsmåte. Dette krever at læreren har kunnskap om de forskjellige verktøyene og bruker tid på å lære dette til elevene. Det er også viktig at undervisningen bygger videre på elevenes feil og misoppfatninger for å gjøre elevene bedre. Det å lære de forskjellige verktøyene vil ta mye tid fra selve faget – noe det etter min mening og lærerens tilbakemelding ikke er reelt med tid til i matematikkfaget.

For utvikling av abstrakt tenking er GeoGebra et godt verktøy til visuell fremstilling som for eksempel for funksjoner. Læreren velger å gå mer vekk fra å bruke GeoGebra og CAS til fordel for Python som er White Box. Til funksjoner er GeoGebra godt egnet som undervisningsverktøy da det er et *dynamic geometric program*. GeoGebra er laget til fremstilling av funksjoner og grafikk og tilbakemeldingene fra noen elever støtter opp om at GeoGebra er mest nyttig i arbeid med funksjoner. CAS ble tatt inn i GeoGebra men har ikke

helt slått an (Albaladejo et al., 2015) siden GeoGebra ikke er laget for CAS. Da er det bedre å bruke Python som CAS.

Det bør vurderes konsekvensene av å bare bruke Python versa å undervise i flere forskjellige verktøy med tanke på eksamensgjennomføring og timetallet til rådighet i matematikkfaget. GeoGebra har vist seg å være et godt verktøy til å undervise i funksjoner og geometri. GeoGebra kan bidra til å skape diskusjon i timen og bedre undervisningen ved å la elevene lære gjennom sosiale aspekter og kommunikasjonsverktøy (Albaladejo et al., 2015). CAS er et Black Box verktøy som brukes til matematiske beregninger men vil ikke bidra til økt matematisk forståelse ifølge informantlæreren. Python, som White Box verktøy for CAS, har den fordel at når elevene skriver feil i koden, vises programfeilen umiddelbart på elevenes skjerm. Elevene lærer å «dele opp» problemet, gå tilbake og se på hva som gjorde at programmet ikke fungerte, lese gjennom koden steg for steg og forstå hvor de gjorde feil - feilsøking[10]. Elever kan ha problemer med å forstå og lese programmerings-språket og/eller ikke har nok kunnskap om matematikktemaet til å finne feilen. Feilsøking tar tid og feilsøking vil også bruke timer av matematikkfaget.

7.5 Algoritmisk tenkning

Det sies at å utvikle strategisk tankegang som utvikler elevenes logikk og problemløsningskompetanse også har sammenheng med algoritmisk tenkning. Ved problemløsning må man bryte ned oppgaven i biter, analysere og velge riktig fremgangsmåte for å effektivt løse oppgaven, tilsvarende gjelder for programmering. Programmering er et verktøy for å trene elevene i algoritmisk tenkning fordi for å programmere må elevene tenke strukturert. Læreren understreker at poenget er ikke at elevene skal bli gode kodere men at de skal trene på algoritmisk tenkning samtidig som de programmerer, men elevene må kunne programmering og inneha kompetanse i algoritmisk tenkning for å kunne løse komplekse problemer.

[10] Store norske leksikon, s.v. «Feilsøking». Hentet_ 24.06.20 https://snl.no/feils%C3%B8king programmering. Hentet Store norske leksikon. (2018).

Læreren sier at det viktigste er at elevene lærer å tenke algoritmisk. Problemløsning og programmering har en felles arbeidsmetode der teste og feile inngår i prosessen for å nå sluttproduktet. I problemløsningsaktiviteter er det ikke løsningen som er viktigst, det er det prosessen frem til løsningen som er med stegene utforske, teste og feile. Tilsvarende gjelder for programmering, når elevene skal lage sin egen algoritme for å få koden til å kjøre er det prosessen som gjør at fremgangsmåten ikke blir instrumentell.

7.6 Motivasjon og samarbeid

Elevene setter pris på forståelse og utforskning når de jobber med programmeringsoppgavene og uttaler at de blir motivert til å jobbe med programmering og det er annerledes enn det de gjør i andre matematikktimer. Motivasjon for å lære programmering gir økt glede til å arbeide med matematikk og det bidrar igjen til økt indre motivasjonen. Elevene er engasjerte i hva som kreves av dem på arbeidsmarkedet og det gir en motivasjon til å lære noe nytt. Informantelevene besitter en indre og ytre motivasjon. De får en ytre motivasjon av at fremtidig arbeidsmarked ofte krever kunnskap i programmering noe elevene var fullt klar over. Den indre motivasjonen får delevene av at undervisningen er variert og relevant. De fleste av informantene har interesse for programmering og teknologi. Tilsvarende gjelder antagelig ikke foreløpig for en gjennomsnittlig skoleelev.

Generelt vil det nok være mer variasjon i motivasjonen blant elevene i en klasse. Det vil gi behov for variert og elevtilpasset undervisning slik at elevene får til programmering og opplever mestringsmotivasjon. Elever kan få motivasjon til å arbeide med matematikk av at programmering er et nytt verktøy til bruk for å løse matematikkoppgaver. Med PRIMM får elevene vist funksjoner fra en annen innfallsvinkel enn bare den rene matematiske. De får også arbeidet sammen med å løse problemer og som elevene sier; «å parprogrammere gir motivasjon og entusiasme». Ved parprogrammering må de sammen lage en algoritme for å løse problemet og de må selv, med veiledning av læreren, resonnere og skrive et program som gir rett resultat. Elevene får et eierskap til arbeidet de har gjort og det kan være med på å øke motivasjonen i faget.

Fra svarene på spørreundersøkelsen ser vi at det er utfordringer med parprogrammering i PRIMM. Jfr. Tabell 10 liker flertallet av elevene å jobbe i par til sammenlikning med

fritekstsvarene der det er flere elever som er negative til samarbeidspartneren sin i dette forsøket enn det var av «Nei»-svar. Jfr. Tabell 10 og ut ifra kommentarene til elevene var det flere i R1- klassen enn i 1T-klassen som var misfornøyde med partneren sin. En mulig grunn til dette kan være at læreren i 1T-klassen gjennom det første halvåret hadde brukt tid på programmering med PRIMM. 1T-elevene er da vant med parprogrammeringen og kunnskapsnivået på elevene kan ha vært mer homogent. I R1-klassen har de programmert med en annen lærer og kunnskapen kan variere mer med elevenes interesse og tilegnelse av stoffet.

Tilbakemeldingene fra informatelevene viser at de som var på «likt faglig nivå» arbeider mye bedre sammen med parprogrammeringen. Det kan se ut som at kunnskapsnivået på R1-elevene er mer variert og at variasjon kan føre til frustrasjon for begge parter da den ene enten må dra «lasset» eller ikke får bidra. Det var i begge klassene en del vandring rundt i rommet under oppgavene og man så at den ene ble sittende med arbeidet. Det kan også virke som at noen av elevene heller ville arbeide alene hvis de ikke jobbet med en som var bedre eller på samme kunnskapsnivå som eleven selv. Når en elev ikke aktivt får delta i prosessen kan eleven miste eierskapet til fremgangsmåten de har brukt på problemløsningen og entusiasmen og motivasjonen kan forsvinne. Hvert av parene brukte også bare én PC. Det er vanlig i parprogrammering, men hvis elevene er vant med å jobbe på egen maskin og alene kan det være et problem. Ved bruk av grafisk verktøy har det vist seg at å bruke én PC og parprogrammering har vært en god måte til å lære på (Albaladejo et al., 2015).

Dette er noe lærerne bør følge med på i starten av året og erfare hvordan og hvilke elever som samarbeider best. I matematikktimene er det ofte at elevene blir plassert slik at flinke elever kan hjelpe de svake elevene. I parprogrammering kan det se ut som at det kan være en dårlig løsning i og med at elevene klager over at partner ikke bidrar, samtidig som de og gjerne vil jobbe i par. Siden det er tekstbasert koding må det innføres rutiner for parprogrammering. Som vi vet fra kapittel 3.2 om parprogrammering må partnerne bytte på å være skribent og kontrollør (Cockburn & Wiliams, 2000). I begynnelsen må læreren trene elevene på å arbeide på denne måten slik som de må trene på andre typer gruppearbeid.

Bruk av PRIMM gir god mulighet for å skape gode rutiner. Læreren forteller at med PRIMM vet elevene til enhver tid hva som skal skje og dette gir en trygghet og stabilitet i undervisningen. Ettersom elevene får mer erfaring med PRIMM kan det bidra til at elevene

arbeider bedre sammen da de vet hva som kreves av hverandre. For som matematikklærerne forteller er eleven vant med å samarbeide i matematikkundervisningen.
Bruk av PRIMM og parprogrammering kan bidra til at elevene hjelper hverandre og at flere elever når neste kunnskapsnivå.

7.7 Fremtidige forutsetninger

For at implementeringen av programmering i matematikkfaget skal bli vellykket spiller to faktorer en spesiell viktig rolle. - Eksamen og lærerkompetanse.

Eksamen

Læreren mener at det ikke kan bli gitt oppgaver på eksamen som må løses ved bruk av Python. Det er metodefrihet og det står ikke i læreplanen eller i kjerneelementene hvilket språk det skal programmeres i. Suurtamm et al. (2016) skriver at tradisjonelt sett har eksamen vært instrumentell og undervisningen relasjonell. Hvis fremtidig eksamen skal gjennomføres som matematikkeksamen har vært til nå, der det står «Bruk CAS», må det fremover legges opp til at programmeringen er en pseudiokode for å tilfredsstille kravet til metodefrihet. Det blir også for teknisk om det skal skrives i ett spesielt språk på eksamen.

Eksamen er styrende for hvordan det blir undervist i pensumet og etter at GeoGebra ble gitt på eksamen ble lærerne mer opptatt av å undervise i GeoGebra. Nå har Utdanningsdirektoratet kommet med ny IKT-komponent (programmering) uten å si hvordan det skal fungere på eksamen. Det er en svakhet at fokuset har vært på innføringen av programmering og at det ikke er diskutert hvordan programmering skal vurderes nasjonalt og lokalt.

Lærerkompetanse

Det kan skje at læremiddelprodusentene tar styring på innholdet og arbeidsformene hvis lærerne ikke har den nødvendige kunnskapen til å undervise i programmering i matematikkfaget. Lærebøker og eksamen har tidligere påvirket hvordan lærere underviser og planlegger undervisningen (Sivesind & Buchmanns, 2013, s.314). Informantlæreren sier det er spennende å følge utviklingen av programmeringsoppgavene i lærebøkene fremover. Hun tenker det kan bli dårlige oppgaver i læreboken fordi programmering er et tema som *må* være

med og som kan bli veldig instrumentelt. Hvor vellykket programmering i matematikkfaget blir vil avhenge av lærerens tid, kompetanse og eget arbeid. Læreren kommenterer at programmeringsoppgavene fort kan blir banale, lite undersøkende og ganske instrumentelle fordi Fagfornyelsen 2020 ikke gir retningslinjer for programmeringsoppgaver.

Det er viktig at lærerne får den opptreningen og videreutdanningen som er nødvendig for at alle elever skal få lik opplæring. Fra ungdomsskolen, jfr. Kapittel 6.1, var det i 1T-klassen 56% av elevene som hadde erfaring med programmering fra før de begynte på vgs. og 61.5% i R1-klassen. Disse elevene har et fortrinn med å være trent i algoritmisk tenkning og en viss forståelse for hva programmering er. Elevene kan nødvendigvis ikke kodespråket men det går raskere å lære fordi de innehar grunnleggende forståelse som kan knyttes opp mot Python, metakognitiv læring.

Selv om læreren har nødvendig kompetanse er det enighet blant informantelevene at lærerens engasjement er viktig for god undervisning. Informantelevene fikk god veiledning og læreren bisto elevene med feilsøking i koden når det var nødvendig. Informantlæreren brukte mye tid på å lage oppgaver og hun utviklet da samtidig egen programmeringskunnskap. Lærerens mål var at programmeringen ikke skulle bli en separert del av undervisningen men integrerer i matematikkfagets premisser. Bruk av PRIMM tilrettelegger for å bruke oppgaver som ikke er fra læreboken, oppgaver lærerne selv føler eierskap til og som er godt tilpasset temaet det undervises i.

PRIMM gir lærere en metode de kan ta utgangspunkt i når de forbereder undervisning. Lærerne kan etterhvert gjøre metoden til «sin egen» samtidig som de utvikler seg faglig. Skolen skal forberede elevene til det arbeidsmarkedet de en dag skal møte og for å klare dette må lærerne videreutdannes faglig og didaktisk. P*rofesjonsfaglige digitale kompetanse* (PfDK) støtter opp om at lærerne skal profesjonsutvikle seg i takt med det som kreves av elevene innen grunnleggende digitale ferdigheter (Furberg & Lund, 2007, s.2; Kelentric et al, 2017, s11). Samarbeid blant lærere, erfaringsutveksling, utarbeide opplegg og oppgaver er viktig for at innføringen blir best mulig og at alle elevene gis like muligheter til å erverve kompetanse på vgs.

8 Konklusjon

Dette kapittelet oppsummerer de viktigste funnene fra diskusjonskapittelet og besvarer forskningsspørsmålene. Jeg vil også ta for meg styrker og begrensninger med studiet før jeg avslutningsvis kommer med forslag til videre forskning.

8.1 Oppsummering av funn

Positive funn:

- Et av hovedfunnene har vært at PRIMM som undervisningsmodell for programmering i matematikk på vgs. har vært vellykket. PRIMM er en god mal for læreren ved planlegging av undervisningen. Vi gjorde en endring i metoden til å la elevene skrive koden inn i editoren selv i motsetning til opprinnelig modell der elevene «copy-paster» koden, resultatene var positive. Argumentene for at elevene skal skrive inn koden i editoren er at dette trener elevene i å bruke tastaturet med «spesialtegn».
- Det er i siste stadiet *make* at elevene får mulighet til å fordype seg.
- Et annet positivt funn er at siden stadiene i PRIMM skal utføres i fast rekkefølge vil det gi elevene forutsigbarhet.
- PRIMM gir muntlig aktivitet. De aller fleste elevene likte å forklare kode ved muntlig dialog. De får dermed testet sin forståelse på partner.
- Programmeringsundervisningen bidro til å gi elevene økt motivasjon til å arbeide både på grunn av ny arbeidsform og at det er relevant for arbeidslivet.
- Programmering er nytt for lærere og elever. Det er derfor viktig å gi klare beskjeder og å ha god nok tid til å gjennomføre alle stadiene.

Utfordringer:

- Parprogrammering. For å få et godt samarbeid og godt utbytte av parprogrammering er fokus på gruppesammensetning et veldig viktig aspekt. Erfaringen fra dette prosjektet viser at når elevene ikke har en partner som er interessert og/eller på samme faglige nivå kan det skape irritasjon eller skjevhet i arbeidsgjennomføringen. Parprogrammeringen er en stor del av PRIMM så det å sette sammen par som jobber godt sammen er vesentlig for at elevene skal lære.

I dette masterprosjektet har det ikke vært nok tid til å kunne måle læring. Men, de resultater vi har fått viser at det var matematikken elevene slet med og ikke programmeringen.

- Som ved alle andre innføringer av nytt teknologisk verktøy tar dette tid. I dette prosjektet er det jobbet tverrfaglig ved at det også ble brukt timer fra samfunnsfag i programmeringsinnføring. På andre skoler vil det kanskje ikke være anledning til dette, en mulighet er å ha en fagdag eller sette av timer spesielt til oppstart med programmering. En ulempe er at det kan ta tid fra andre temaer i matematikkfaget.
- Informantelevene er ganske interessert i teknologi og mange hadde erfaring med programmering fra før de begynte på vgs. I begge klassene ble det likevel tekniske utfordringer med installasjon av nødvendig programvare. Læreren fikk ikke gått rundt til alle gruppene for å veilede og hjelpe fordi hun måtte prioritere å bistå gruppene som hadde tekniske problemer. Tekniske problemer er noe man ikke kan klare å eliminere men editor og det tekniske utstyret som skal brukes må være klargjort og i orden til skolestart.

8.2 Styrker og begrensninger med studiet

Styrker

En styrke med studiet er at undervisningsmodellen PRIMM er basert på tidligere forskning og det er relativt godt forsket på PRIMM ved to pilotstudier og en storskalastudie. De tidligere studiene hadde også som mål å finne en modell godt egnet for lærere som ikke har erfaring fra programmeringsundervisning. Lærere skal ha en mal å gå uti fra når de planlegger undervisningen og som også skal gi mulighet til å gjøre små tilpasninger.

En annen styrke med dette masterstudiet er at det baserer seg på analysekategorier som er relevante også for selve matematikk faget og ikke bare «computational science» som det er tilpasset for i England. En fordel med kodene i analysearbeidet for intervjuet og spørreundersøkelsen er at det benyttes nesten like koder. De som er like er de som også ble brukt fra tidligere forskning.

Begrensninger

En begrensning ved metodebruken i denne masteroppgaven er antall informanter. Det burde vært flere informanter og testklasser. Men da dette prosjektet startet var det veldig få klasser som hadde startet med programmering. Masterprosjektet måtte også tilpasses lærerens tid og for at vi skulle få gjennomført forsøkene var to klasser det vi fikk til.

En svakhet med utvalget er at det ikke er representative for resten av landet. Informantelevene er over gjennomsnittet interessert i teknologi og flertallet hadde erfaring med programmering fra før de begynte på vgs. Informantelevenes programmeringserfaring kan ha innvirket på opplevelsen av lærerens gjennomgang og på deres evne til oppgaveløsing. Prosjektlederen/læreren for innføring av programmering i.h.h.t. til Fagfornyelsen var også ansatt på skolen der informantelevene gikk og det kan ha ført til at elevene kan ha følt et press til å delta og at svarene på intervjuet kan værer vektet i retning av at PRIMM vil fungere, læreren har et bias.

En annen begrensning er metodene som er brukt. Selv om det er *mixed method* med intervju og spørreundersøkelse burde jeg muligens ha gjennomført dybdeintervju med noen av elevene for å få mer nøyaktige svar. Det var søkt NSD om intervju av et utvalg elever og laget intervjuguide som ble godkjent, men på grunn av tidsmangel ble dette ikke gjennomført. Det ville uansett gått noe tid mellom gjennomføring av intervjuene og forsøksundervisningen. Dette ville påvirket elevenes svar fordi deres erfaring med gjennomføring av undervisningen hadde blitt «lagret» bak i hukommelsen.

8.3 Videre forskning

Forslag til videre forskning er å teste ut PRIMM i storskalaundersøking. Med tanke på at programmeringsundervisning vil inngå i matematikkfaget i flere år fremover vil det være interessant å forske ytterligere på bruk og effekt av denne undervisningen.
Storskalatesting over lengre tid vil kunne gi informasjon om hvordan det bidrar til dybdelæringen og om det bør gjøres endringer i PRIMM. Det vil med storskalatesting over tid være mulig å validere funnene gjort i denne masteroppgaven.
Videre forskning på bruk av PRIMM kan føre til at det blir laget oppgaver tilpasset resten av pensum i matematikk. Oppgavene kan testes og deles med kollegaer.

Litteraturliste

Albaladejo, I. M. R., Garcia, M. M. & Codina, A. (2015). Developing mathematical competencies in secondary students by introducing dynamic geometry systems in classrooms. *Education ans Science, 40*(177), 43-58.

Ashbacher, C. (2002). Pair Programming Illuminated by Laurie Wiliams and Robert Kessler. *Journal of Object Technology, 1*(5), 179-180. https://doi.org/10.5381/jot.2002.1.5.r2

Balanskat, A. & Engelhardt, K. (2015). Computing our future. *European Schoolnet*, 1-87. Hentet fra http://www.eun.org/documents/411753/817341/Computing+our+future_final_2015.pdf/d3780a64-1081-4488-8549-6033200e3c03

Befring, E. (2015). Vitenskapelige tradisjoner og verdier. I E. Befring (Red.), *Forskningsmetoder i utdanningsvitenskap* (s. 20-27). Oslo: Cappelen Damm Akademisk.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A. & Engelhardt, K. (2016). *Developing Computational Thinking in Compulsory Education - Implications for policy and practice*. Luxembourg. Hentet fra https://www.researchgate.net/publication/312039564_Developing_Computational_Thinking_in_Compulsory_Education_Implications_for_policy_and_practice

Bocconi, S., Chioccariello, A. & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. *Report prepared for the Nordic@BETT2018 Steering Group.* https://doi.org/https://doi.org/10.17471/54007

Buchberger, B. (1990). Should Students Learn Integration Rules? *ACM SIGSAM Bulletin, 24*(1), 10-17. https://doi.org/10.1145/382276.1095228

CAS. (14. februar, 2017). I *Wikipedia*. Hentet 23. august 2020. https://no.wikipedia.org/wiki/CAS. Hentet fra https://no.wikipedia.org/wiki/CAS

Cockburn, A. & Wiliams, L. (2000). The Costs and Benefits of Pair Programming, 12. Hentet fra https://www.researchgate.net/publication/2333697_The_Costs_and_Benefits_of_Pair_Programming

Dalen, M. (2011). *Intervju som forskningsmetode. En kvalitativ tilnærming* (2. utg.). Oslo: Universitetsforlaget

diSessa, A. A. (2018). Computational Literacy and "The Big Picture" Concerning Computers in Mathematics Education. *Mathematical Thinking and Learning, 20*(1), 3-31. https://doi.org/10.1080/10986065.2018.1403544

Drijvers, P., Kieran, C., Mariotti, M.-A., Ainley, J., Andresen, M., Chan, Y. C., … Meagher, M. (2010). Integrating Technology into Mathematics Education: Theoretical Perspectives. I *Mathematics Education and Technology-Rethinking the Terrain* (s. 89-132).

Fangen, K. (2011). Deltagende observasjon IK. Fangen & A.-M. Sellerberg (Red.), *Mange ulike metoder* (s. 37-56). Oslo: Gyldendal Akademisk

Fauskanger, J. & Mosvold, R. (2014). Innholdsanalysens muligheter i utdanningsforskning. *Norsk pedagogisk tidsskrift, 98*(Issue), s. 127-139.

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

Furberg, A. & Lund, A. (2016). En profesjonsfaglig digitalt kompetent lærer? Muligheter og utfordringer i teknologirike læringsimgivelser. I R. J. Krumsvik (Red.), *Digital læring i skole og læreutdanning* (2. utg., s. 26-43). Oslo: Universitetsforlaget.

Gilje, Ø., Landfald, Ø. F. & Ludvigsen, S. (2019, 29.09.2018). Dybdelæring – historisk bakgrunn og teoretisk tilnærminger. *Utdanningsnytt.no*. Hentet fra https://www.utdanningsnytt.no/fagartikkel-forskning-pedagogikk/dybdelaering--historisk-bakgrunn-og-teoretiske-tilnaerminger/171562

Goos, M., Galbraith, P., Renshaw, P. & Geiger, V. (2003). Perspectives on technology mediated learning in secondary school mathematics classrooms. *The Journal of Mathematical Behavior, 22*(1), 73-89. https://doi.org/10.1016/s0732-3123(03)00005-1

Greefrath, G., Hertleif, C. & Siller, H.-S. (2018). Mathematical modelling with digital tools—a quantitative study on mathematising with dynamic geometry software. *Zdm, 50*(1-2), 233-244. https://doi.org/10.1007/s11858-018-0924-6

Grønmo, S. (2015). Strukturert utspørring IS. Grønmo (Red.), *Samfunnsvitenskapelige metoder* (s. 190-211). Bergen: Fagbokforlaget.

Ingeniørenes stemme. (2018). Likestilte kvinner velger bort realfag. Hentet Hentet: 8. juli. 2020 2018, April fra https://www.ingstemme.no/utdanning-likestilling-talentsenter-i-realfag/likestilte-kvinner-velger-bort-realfag/101246

Kelentric, M., Helland, K. & Arstorp, A.-T. (2017). Rammeverk for lærerens profesjonsfaglige digitale kompetanse. I(s. 70). Senter for IKT i utdanning.

Kilpatrick, J., Swafford, J. & Findell, B. (2001a). The strands of mathematical proficiancy. I *Adding It Up: Helping Children Learn Mathematics* (s. 115-157). National academies press.

Kilpatrick, J., Swafford, J. & Findell, B. (2001b). The Strands of mathematical proificiency. I *Adding it up*. National academies press.

Kjerneelementgruppen for matematikk. (2017, 21.09.2017). Kjerneelementer i matematikk, men hvorfor programmering? Hentet fra https://udirbloggen.no/kjerneelementer-i-matematikk-men-hvorfor-programmering/

Kjerneelementgruppen for matematikk. (2017b). Kjerneelementer i matematikk, men hvorfor programmering? Hentet fra https://udirbloggen.no/kjerneelementer-i-matematikk-men-hvorfor-programmering/

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

Kunnskapsdepartementet. (2018). *Retningslinjer for utforming av nasjonale læreplaner for fag i LK20 og LK20S: Til bruk for læreplangrupper som er oppnevnt av Utdanningsdirektoratet eller Sametinget*. Oslo: Regjeringen. Hentet fra https://www.regjeringen.no/contentassets/3d659278ae55449f9d8373fff5de4f65/retningslinjer-for-utforming-av-nasjonale-og-samiske-lareplaner-for-fag-i-lk20-og-lk20s-fastsatt-av-kd.pdf

Lesh, R. A. & Zawojewski, J. S. (2007). Problem Solving and Modeling. I F. K. J. Lester (Red.), *Second handbook of research on mathematics teaching and learning* (bd. 2, s. 763-804). Charlotte, NC: Information Age:

Liljedahl, P. & Hannula, M. S. (2016). Research on Mathematics-Related Affect: Examining the structures of affect and taking the social turn. I A. G. Gutiérrez, G. C. Leder & P. Boero (Red.), *The Second Handbook of Research on the Psychology of Mathematics Education: The Journey Continues* (s. 417-446). Rotterdam: Sense Publisher.

Loksa, D., Ko, A. J., Jernigan, W., Oleson, A., Mendez, C. J. & Burnett, M. M. (2016). *Programming, Problem Solving, and Self-Awareness*. Innlegg presentert ved Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems.

Ludvigsen, S. R. & Mørch, A. I. (2011). Theoretical Bases og Computer Supported Learning. I I. V. G. Aukrust (Red.), *Learning and Cognition* (s. 46-51). Burlington: Elsevier Science.

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

Manger, T. (2013). Motivasjon for skulearbeid IR. J. Krumsvik & R. Säljö (Red.), *Praktiskpedagogisk utdanning: En antologi* (3. utg., s. 145-169). Bergen: Fagbokforlaget.

Marton, F. & Säljö, R. (1976). On qualitative differences in learning: I. Outcome and process. *British Journal of Educational Psychology, 46*(1), 4-11. https://doi.org/https://doi.org/10.1111/j.2044-8279.1976.tb02980.x

Mayer, R. E., Dyck, J. L. & Vilberg, W. (1986). Learning to program and learning to think: what's the connection? *Communications of the ACM, 29*(7), 605-610. https://doi.org/10.1145/6138.6142

Mellin-Olsen, S. (1981). Instrumentalism as an educational concept. *Educational Studies in Mathematics, 12*(3), 351-367. https://doi.org/10.1007/BF00311065

Miller, J. & Glassner, B. (2014). The «inside» and the «outside». I D. Silverman (Red.), *Qualitative Research* (2. utg., s. 125-199). London: SAGE Publications.

Nordlander, M. & Nordlander, E. (2009). Influences of students´ attitudes and beliefs on the ability of solving mathematical problems with irrelevant information. I J. Mass & W. Schlögelmann (Red.), *Beliefs and attitudes in mathematics education* (s. 165-178). Rotterdam: Sense Publishers.

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

NOU 2013:2. (2013). *Hinder for digital verdiskapning*.

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

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

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

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

Polya, G. (1957). *How To Solve It* (2. utg.)Princeton University Press.

Repley, T. (2016). Some pragmatics of Qualitative data analysis. I D. Silverman (Red.), *Qualitative Research* (4. utg., s. 331-355). Thousand Oaks: SAGE.

Sanne, A., Berge, O., Bungum, B., Jørgensen, E. C., Kluge, A., Kristiensen, T. E., … Voll, L. O. (2016). *Teknologi og programmering for alle - En faggjennomgang med forslag til endringer i grunnopplæringen*. Oslo: Utdanningsdirektoratet. Hentet fra https://www.udir.no/globalassets/filer/tall-og-forskning/forskningsrapporter/teknologi-og-programmering-for-alle.pdf

Sawyer, R. K. (2006). The New Science of Learning. I R. K. Sawyer (Red.), *The Cambridge Hanbook of the Learning Sciences* (2. utg., s. 1-10). Cambridge: Cambridge University Press.

Schoenfeld, A. H. (1987). What´s all the fuss about metacognition? I A. H. Schoenfeld (Red.), *Cognitive science and mathematics education* (s. 189-215). Hillsdale: NJ: Lawrence Erlbaum Associates.
Scratch. (4. juli. 2019). I *Wikipedia*. Hentet 23. august. 2020 fra: https://no.wikipedia.org/wiki/Scratch. Hentet
Sentance, S. & Waite, J. (2017a). PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. I *Proceeding of the 12th Workshop in Primary and Secondary Computing Education: WIPSCE'17*. Nijmegen:
Sentance, S. & Waite, J. (2017b). *Proceedings of the 12th Workshop in Primary and Secondary Computing Education*. Abstract hentet fra https://kclpure.kcl.ac.uk/portal/files/79583213/version_pure_primm_1.pdf
Sentance, S., Waite, J. & Kallia, M. (2019). Teaching computer programming with PRIMM: a sociocultural perspective. *Computer Science Education, 29*(2-3), 136-176. https://doi.org/10.1080/08993408.2019.1608781
Sevik, K. (2016). *Notat om programmering i skolen*. Utdanningsdirektoratet.no: Senter for IKT i utdanningen. Hentet fra https://www.udir.no/globalassets/filer/programmering_i_skolen.pdf
Sivesind, K. & Bachmann, K. (2013). Læreplaner - fra dokumenter til dokumentasjon. I R. J. Krumsvik & R. Säljö (Red.), *Praktiskpedagogisk utdanning: en antologi* (2017. utg., bd. 3, s. 304-342). Bergen: Fagbokforlaget.
Stacey, K. & Wiliam, D. (2013). Technology and assessment in mathematics. I I. M. A. Clemets, A. J. Bishop, C. Keitel, J. Kilpatrick & F. K. S. Leung (Red.), *Third international handbook of mathematics education* (s. 721-751). New York: Springer.
Suurtamm, C., Thompson, D. R., Kim, R. Y., Moreno, L. D., Sayac, N., Schukajlow, S., … Vos, P. (2016). *Assessment in mathematics education* (1. utg.). Doordrecht: Springer.
Swidan, A. & Hermans, F. (2019). *The Effect of Reading Code Aloud on Comprehension*. Innlegg presentert ved Proceedings of the ACM Conference on Global Computing Education.
SymPy Development Team. (u.å). What is SymPy. (Hentet 1. Juni 2020). Hentet fra https://www.sympy.org/en/index.html
Säljö, R. (2013). Støtte til læring - tradisjoner og perspektiver. I R. J. Krumsvik & R. Säljö (Red.), *Praktisk pedagogisk utdanning: rm antologi* (3. utg., s. 53-79). Bergen: Fagbokforlaget.
Utdanningsdirektoratet. (2019). *Algoritmsik tenkning*. Hentet fra https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/
Utdanningsdirektoratet. (2019a). *Læreplan i matematikk fellesfag Vg1 teoretisk (matematikk T)*. Hentet fra https://data.udir.no/kl06/v201906/laereplaner-lk20/MAT09-01.pdf
Williams, L. & Kessler, r. (2002). *Pair Programming Illuminated*. Boston: MA.
Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3). https://doi.org/10.1145/1118178.1118215

Vedlegg 1 – intervjuguide

Intervjuguide til lærer:

Start:

- Hva er din fagbakgrunn?
- Hvor mye erfaring har du med programmering?

Tema: «Programmeringsregnskapet»

- Hvor mange timer (ca.) i matematikkundervisningen ser du for deg det skal undervises i programmering (Python) i ett skoleår? (De første årene)
    - Eller: Hvor mange timer/uker vil gå med på å lære elevene Python i matematikkundervisningen.
- Hvor mange timer i uken har du brukt?
- Hvor mange timer i uken har du brukt på å lære elevene programmering i matematikkundervisningen?
    -Av 1T og R1 - klassene. Hvor har det "kostet" mest på å lære vekk Python.
- Hvilken av fagene (1T/R1) har det mest «pay-off?» - mer forståelse for matematikken?

Tema: Programmering

- Hvilken gevinst tror du elevene får ved å lære seg Python?
    -Algoritmisk tenkning?
    - matematisk forståelse?
    - Hva er det Python mest konkret kan hjelpe elevene med i matematiske temaer? (Algebra, problemløsning, funksjoner)
- Hva ser du for deg Python vil påvirke matematikkundervisningen? - Positivt og negativt
- Kommer du til å bruke Python programmering som et mål eller undervisningsmiddel?

Tema: Modellen

- Spørsmål om fasene i PRIMM.
- Hva tenker du kan utvikles til det bedre med PRIMM?
    - Du kommer til å bruke PRIMM videre i matematikkundervisningen?
- Har du noen spesielle erfaringer du har tatt med deg fra prosjektet? negative/positive
    - er det noen som er mer viktigere enn andre?
- Fra da jeg observerte i R1, så ble det litt uroligheter, tenker du at det burde vært mer lærerstyrt.
- Holdt det med 45 min?
- Hva er dine oppfatninger fra da du fremstilte opplegget for elevene ang. deres reaksjoner på opplegget?
- Hva tenker du om samarbeidet i parprogrammeringen?

Tema: Differensiering

- Det var litt forskjellige hva elevene hadde av forkunnskaper på programmering i de to klassene. Hvordan påvirket dette undervisningen?
- Var det lettere for deg ang programmeringen i R1 enn i 1T?

- Hvordan følte du metoden vi brukte var godt differensiert? Og oppgavene?

Avslutning

Når tror du programmering er på det andvendelsesnivået som GeoGebra er i dag?

Har du noen spørsmål til meg?

Vedlegg 2 - Spørreskjema

Spørreskjema elever

*

Jeg har erfaring med programmering fra før jeg begynte på videregående skole

- ○ Ja
- ○ Nei

Hvis ja; på spørsmålet over:

ⓘ Dette elementet vises kun dersom alternativet «Ja» er valgt i spørsmålet «»

Jeg har erfaring med (flere svar mulig):

- ☐ Java
- ☐ Python
- ☐ Arduino
- ☐ micro:bit
- ☐ C++
- ☐ C+
- ☐ Scratch

*

Syns du det er krevende å lære deg programmering? (hvorfor/hvorfor ikke)

*

Hva tenker du om å programmere i matematikkundervisningen?

Tenk tilbake på mattematikk-undervisningen du nettopp har hatt *

Likte du at dere jobbet i par med oppgaver?

- ja
- nei

*

Hvorfor likte du det?/hvorfor likte du det ikke?

*

Følte du at du fikk dypere forståelse om andregradsfunksjoner ved å programmere?

- ja
- nei

Hvis "ja" på forrige spørsmål: *

Dette elementet vises kun dersom alternativet «ja» er valgt i spørsmålet «»

Hva er grunnen til at programmeringen bidro til din forståelse om funksjoner

Hvis "nei" på forrige spørsmål: *

Dette elementet vises kun dersom alternativet «nei» er valgt i spørsmålet «»

Hva er grunnen til at programmering i timen ikke bidro til bedre forståelse på funksjoner

*

Fra skala 1-6, der 1 er dårlig og 6 er veldig bra. Hvordan synes du lærerens gjennomgang var i dag?

- [ ] 1
- [ ] 2
- [ ] 3
- [ ] 4
- [ ] 5
- [ ] 6

*

Hva likte du best med å bruke programmering til å løse oppgaver i matematikktimen?

*

Fikk du den tiden du trengte til å følge undervisingen?

- ◯ ja
- ◯ nei, gikk for fort
- ◯ nei, har ikke helt forstått det å programmere i python
- ◯ nei, annen grunn

Oppgavene *

Var oppgavene utfordrende nok?

- ◯ ja
- ◯ nei

Hvis "ja" på utfordrende oppgaver *

Hvis "ja" på utfordrende oppgaver *

Dette elementet vises kun dersom alternativet «ja» er valgt i spørsmålet «Oppgavene»

Hva likte du best med oppgavene?

Hvis "nei" på oppgavesprømål?

Dette elementet vises kun dersom alternativet «nei» er valgt i spørsmålet «Oppgavene»

Hva likte du ikke med oppgavene?

*

Ble du motivert til å jobbe med matematikk når du fikk bruke python?
Uttyp

Til slutt: *

Hva tenker du om at programmering skal inn i matematikkundervisningen fra høsten 2020?

Litt om undervisningen

Her skal dere krysse av på hvor tilfreds dere var med forksjellige deler av undervisningen. 1 er ikke enig og 6 er helt enig.

	1	2	3	4	5	6
Å få vist frem en kode på tavlen gjør at jeg lærer hvordan koder fungerer	○	○	○	○	○	○
Jeg liker at vi må forklare koden til hverandre	○	○	○	○	○	○
Jeg liker at vi må skrive koden selv (ikke copy paste inn)	○	○	○	○	○	○
Jeg lærer å bruke funksjoner i python ved å skrive koden selv	○	○	○	○	○	○
Jeg likte ikke ikke at vi måtte forklare koden til hverandre	○	○	○	○	○	○
Å få gitt en matteoppgave og så løse den med python gjorde at jeg forstod *tema* mye mer.	○	○	○	○	○	○
Jeg likte at vi kunne prøve ut forskjellige variabler for å løse opgaver	○	○	○	○	○	○
Jeg lærte mer ved å modifisere koden for å løse de vanskeligere oppgavene	○	○	○	○	○	○
Jeg fikk mulighet til å lage min egen kode fra bunnen av på slutten	○	○	○	○	○	○

Vedlegg 3 – NSD

Vil du delta i forskningsprosjektet
"Undervisningsopplegg for programmering i realfagsmatematikk"?

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å *forske på undervisningsmetode som kan brukes til å undervise programmering i matematikk*. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål
Dette er et samtykkeskjema for min masteroppgave i matematikk-didaktikk.
Formålet med prosjektet er å studere effekten av programmeringsundervisningen på videregående skole ved bruk av PRIMM-modellen. Vi ønsker å utvikle en forskningsbasert undervisningsopplegg hvor elevene skal bruke programmering for å støtte dybdelæring i realfagsmatematikk.
Omfanget er begrenset til en matematikklasse og de anonyme dataene vi innhenter her er med på å utvikle undervisningsopplegget på 90 minutter som dere eventuelt skal være med på.
Jeg skal skal se på om undervisningsopplegget har en effekt på om dere lærer mer, og om modellen vi bruker kan utvikles til å brukes som støtteverktøy i matematikkundervisning.

Hvem er ansvarlig for forskningsprosjektet?
Institutt for lærerutdanning og skoleforskning, Universitetet i Oslo,
[redacted]

Hvorfor får du spørsmål om å delta?
Det er veldig få skoler som har begynt med å bruke python i matematikkundervisningen. Og siden dere har litt erfaring med pythonprocrammering fra før, både i matematikkundervisning og utenom, ønsker jeg å ha dere med i prosjektet.

Hva innebærer det for deg å delta?

- Hvis du velger å delta i prosjektet, innebærer det at du fyller ut et spørreskjema. Det vil ta deg ca. 15 minutter. Spørreskjemaet inneholder spørsmål om Hvordan du likte undervisningen, Hva du tenker om programmering, og dine bakgrunnskunnskaper om programmering. Dine svar fra spørreskjemaet blir registrert elektronisk
- Du samtykker også for å delta i observasjon. Altså at jeg (Kristine) sitter bakerst i klasserommet og observerer hele klassen. Det jeg vil se etter er om hvordan dere interagerer med hverandre om oppgavene og gjennomfører undervisningen. Notater jeg tar fra observasjonen er fullstendig anonymt og jeg vil ikke gå til læreren deres for å gi bort informasjon om det jeg har observert.

Det er frivillig å delta
Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du når som helst trekke samtykke tilbake uten å oppgi noen grunn. Alle opplysninger om deg vil da bli anonymisert. 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.

- De som vil ha tilgang til opplysningene du gir er meg (Kristine Heimdal), min veileder (Helmer Aslaksen)
- Samtykkeskjemaene med din signatur vil bli oppbevart innelåst og makulert etter at prosjektet er ferdig. Dataene fra observasjon og spørreskjema vil bli lagret på forskningserver på Universitetet i Oslo.
- Spørreskjema er utgitt av Universitetet i Oslo og lagres på deres server.

Observasjonsnotater, og analyse, samt svarene dere gir på spørreskjema vil bli analysert og brukt i min (Kristine Heimdal) masteroppgave. Du som deltaker vil ikke bli gjenkjent i masteroppgaven.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?
Prosjektet skal etter planen avsluttes 01.06.2020. All innsamlede data jeg får fra dere, og samtykkeskjema vil bli slettet og makulert.

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

- innsyn i hvilke personopplysninger som er registrert om deg,
- å få rettet personopplysninger om deg,
- få slettet personopplysninger om deg,
- få utlevert en kopi av dine personopplysninger (dataportabilitet), og
- å sende klage til personvernombudet eller 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 *Institutt for lærerutdanning og skoleforskning, 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:

- Institutt for lærerutdanning og skoleforskning, Universitetet i Oslo ved Helmer Aslaksen, tlf: 22844482, epost: helmer.aslaksen@ils.uio.no.
- Vårt personvernombud: Personvernombud@uio.no
- NSD – Norsk senter for forskningsdata AS, på epost (personverntjenester@nsd.no) eller telefon: 55 58 21 17

Med vennlig hilsen

Helmer Aslaksen	Kristine Heimdal
Prosjektansvarlig	*student*
(Forsker/veileder)	

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet *Undervisningsopplegg for programmering i realfagsmatematikk*, og har fått anledning til å stille spørsmål. Jeg samtykker til:

- å delta i elektronisk spørreskjema
- å delta i observasjon
- Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet, ca. *01.06.2020*

(Signert av prosjektdeltaker)

Vil du delta i forskningsprosjektet
"Undervisningsopplegg for programmering i realfagsmatematikk"

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å *forske på undervisningsmetode som kan brukes til å undervise programmering i matematikk*. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål
Dette er et samtykkeskjema for min masteroppgave i matematikk-didaktikk.
Formålet med prosjektet er å studere effekten av programmeringsundervisningen på videregående skole ved bruk av PRIMM-modellen. Vi ønsker å utvikle en forskningsbasert undervisningsopplegg hvor elevene skal bruke programmering for å støtte dybdelæring i realfagsmatematikk.
Omfanget er begrenset til en matematikklasse og de anonyme dataene vi innhenter her er med på å utvikle undervisningsopplegget på 90 minutter som du eventuelt skal være med på.
Jeg skal skal se på om undervisningsopplegget har en effekt på om elevene lærer mer, og om modellen vi bruker kan utvikles til å brukes som støtteverktøy i matematikkundervisning.

Hvem er ansvarlig for forskningsprosjektet?
Institutt for lærerutdanning og skoleforskning, Universitetet i Oslo,
[redacted]

Hvorfor får du spørsmål om å delta?
Det er veldig få skoler som har begynt med å bruke python i matematikkundervisningen. Og siden dere har litt erfaring med pythonprocrammering fra før i matematikkundervisning ønsker jeg å ha deg med i prosjektet.

Hva innebærer det for deg å delta?
Beskriv metode (spørreskjema, intervju, observasjon etc.), omfanget, hvilke opplysninger som samles inn og hvordan opplysningene registreres (elektronisk, notater, lyd-/videoopptak), f.eks.:

- *Hvis du velger å delta i prosjektet, innebærer det at du er villig til å delta i ett intervju etter forsøket er gjort. Det vil ta ca 20-30 minutter å gjennomføre. Du vil bli spurt om faglig bakgrunn, om hvordan du syns modellen fungerte, og om du så læring hos elevene. Dine svar vil bli tatt opp på lydopptaker*
- Du samtykker også for å delta i observasjon. Altså at jeg (Kristine) sitter bakerst i klasserommet og observerer deg gjennomføre undervisningsopplegget. Det jeg vil se etter er om hvordan du kommuniserer med elevene om oppgavene og gjennomfører PRIMM-modellen. Notater jeg tar fra observasjonen er fullstendig anonymt.

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du når som helst trekke samtykke tilbake uten å oppgi noen grunn. Alle opplysninger om deg vil da bli anonymisert. 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.

- De som vil ha tilgang til opplysningene du gir er meg (Kristine Heimdal), min veileder (Helmer Aslaksen)
- Samtykkeskjemaene med din signatur vil bli oppbevart innelåst og makulert etter at prosjektet er ferdig. Dataene fra lydopptaket og observasjon, og selve lydopptaket vil bli lagret på forskningserver på Universitetet i Oslo.

Transkripsjon av intervju vil bli analysert og brukt i min (Kristine Heimdal) masteroppgave. Du som deltaker vil ikke bli gjenkjent i masteroppgaven.

Hva skjer med opplysningene dine når vi avslutter forskningsprosjektet?
Prosjektet skal etter planen avsluttes 01.06.2020. All innsamlede data jeg får fra deg, og lydopptak vil bli slettet og makulert.

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

- innsyn i hvilke personopplysninger som er registrert om deg,
- å få rettet personopplysninger om deg,
- få slettet personopplysninger om deg,
- få utlevert en kopi av dine personopplysninger (dataportabilitet), og
- å sende klage til personvernombudet eller 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 *Institutt for lærerutdanning og skoleforskning, 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:

- Institutt for lærerutdanning og skoleforskning, Universitetet i Oslo ved Helmer Aslaksen, tlf: 22844482, epost: helmer.aslaksen@ils.uio.no.
- Vårt personvernombud: Personvernombud@uio.no
- NSD – Norsk senter for forskningsdata AS, på epost (personverntjenester@nsd.no) eller telefon: 55 58 21 17.

Med vennlig hilsen

Helmer Aslaksen	Kristine Heimdal
Prosjektansvarlig	*student*
(Forsker/veileder)	

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet *Undervisningsopplegg for programmering i realfagsmatematikk*, og har fått anledning til å stille spørsmål. Jeg samtykker til:

- å delta i *intervju*
- å delta i *observasjon*

Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet, ca. *01.06.2020*

(Signert av prosjektdeltaker, dato)

NSD sin vurdering

Prosjekttittel
Undervisningsopplegg for programmering i realfagsmatematikk

Referansenummer
285510

Registrert
21.10.2019 av Kristine Gaustad Heimdal - krisghe@uio.no

Behandlingsansvarlig institusjon
Universitetet i Oslo / Det utdanningsvitenskapelige fakultet / Institutt for lærerutdanning og skoleforskning

Prosjektansvarlig (vitenskapelig ansatt/veileder eller stipendiat)
Helmer Aslaksen, helmer.aslaksen@ils.uio.no, tlf: 22844482

Type prosjekt
Studentprosjekt, masterstudium

Kontaktinformasjon, student
Kristine Heimdal, homewall@online.no, tlf: 95927764

Prosjektperiode
23.08.2019 - 15.09.2020

Status
02.06.2020 - Vurdert

Vurdering (3)

02.06.2020 - Vurdert
NSD har vurdert endringen registrert 02.06.2020.

Vi har nå registrert 15.09.2020 som ny sluttdato for forskningsperioden.

I tilfelle det skulle bli aktuelt med ytterligere utvidelse av den opprinnelige sluttdato, må vi vurdere hvorvidt det skal gis ny informasjon til utvalget).

NSD vil følge opp underveis ved ny planlagt avslutning for å avklare om behandlingen av personopplysningene er avsluttet.

Lykke til videre med prosjektet!

18.02.2020 - Vurdert
NSD har vurdert endringen registrert 17.02.2020

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 den 18.02.2020.

Behandlingen kan fortsette.

Endringen består av at en i tillegg til elektronisk spørreskjemaundersøkelse og ikke-deltakende observasjon også ønsker å gjennomføre gruppeintervju med elever 16-17 år.

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

Lykke til med prosjektet!

Kontaktperson hos NSD:

Tlf. Personverntjenester: 55 58 21 17 (tast 1)

28.10.2019 - 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 den 28.10.2019.

Behandlingen kan starte.

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://nsd.no/personvernombud/meld_prosjekt/meld_endringer.html

Du må vente på svar fra NSD før endringen gjennomføres.

TYPE OPPLYSNINGER OG VARIGHET
Prosjektet vil behandle alminnelige kategorier av personopplysninger frem til 01.06.2020

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 behandles 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
Så lenge de registrerte kan identifiseres i datamaterialet vil de ha følgende rettigheter: åpenhet (art. 12), informasjon (art. 13), innsyn (art. 15), retting (art. 16), sletting (art. 17), begrensning (art. 18), underretning (art. 19), dataportabilitet (art. 20).

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.

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/eller rådføre dere med behandlingsansvarlig institusjon.

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

Lykke til med prosjektet!

Kontaktperson hos NSD: Gry Henriksen
Tlf. Personverntjenester: 55 58 21 17 (tast 1)

Vedlegg 4 – Undervisningsplan og oppgaver

UNDERVISNINGSPLAN

Dato:	Klasse: 1STT	Varighet: 90 min	Time/tidspunkt:
Fag: 1T	Kompetansemål (K06) eller fagfornyelsen:		

Læringsmål:	Vurdering:
• lage og tolke funksjonar som beskriv praktiske problemstillingar, analysere empiriske funksjonar og finne uttrykk for ein tilnærma lineær funksjon • bruke digitale hjelpemiddel til å drøfte polynomfunksjonar, rasjonale funksjonar, eksponentialfunksjonar og potensfunksjonar Nye læringsmål: • kommer 18.11	• Levere inn kode samtidig som oppgavene på ItsLearning • (Liten klasse) Gå rundt til hver gruppe
Andre mål for timen: • Kommunikasjon • Algoritmisk tenkning	Utstyr/hjelpemidler: • pc
Undervisningssituasjon (elevenes læreforutsetninger): •	Viktige begreper: • for-løkker • lister • funksjoner
	Andre notater: Timen brukes til å forstå funksjoner, arbeide med python som er i kjerneelementene 2020

PLAN FOR TIMEN

Timing:	Læreraktivitet	Elevaktivitet
5 min	oppstart	
20 min	Lærer viser opplegg, vise eksempler oppgaver, konsolidering	muntlig aktivitet
10 min - predict	Viser kode, spør elever på slutten	jobber i par, finne ut hva koden gjør. Begge presenterer for hverandre.
10 min (run, investigate)	lærer går rundt	Elever skriver inn kode hver for seg, kjører koden til oppgaven, kommentarer i koden og «outputet»
40 (modify)	oppgaver Enten utvide programmet eller lage nye.	elever jobber hver for seg eller i grupper med oppgaver. Kommentarer i koden hva den gjør.
5 min	avslutning	elever laster opp koden sin i teams slik at lærer kan se forståelsen.

Oppgaver:

Oppgave 1: Kopier inn denne koden i terminalen, se om den kjører og kommenter hva hver linjene gjør. Forklar om funksjonen har et toppunkt eller bunnpunkt. Kopier så koden dere har kommentert inn igjen i tekstboksen under.

```
from pylab import *

a = 1
b = -2
c = 2

for a in [-2,-1,0,1,2]:
        x = np.arange(-10, 10, 0.1)
        f=a*x**2 + b*x + c
        plot(x, f, label = a,)

grid(True)
ylim(-10,10)
xlabel('x-aksen')
ylabel('y-aksen')
axhline()
axvline()
legend()
show()
```

Oppgave 2:
Endre på den globale verdien c og forklar hva som skjer. Først endrer dere c til et negativt tall, og observerer hvordan den beveger seg på grafen. Endre c til 0 og observer hva som skjer. Til sist endrer dere c til å være et positivt tall. Eksempel -10, 0, +10. Hvis dere ikke ser forskjell, prøv med større verdier.
Forklar hva som skjer med grafen når c endrer verdi fra negativt, null og positivt.
Forklar i tekstboksen under.

Oppgave 3
Nå skal du endre verdiene til a. Først sett a lik et negativt tall, så 0 og deretter et positivt tall. For eksempel -10, 0, og +10. Hvis dere ikke ser forskjell, prøv med større verdier.
Hva skjer med grafen når du endrer verdiene til a? Hvordan beveger den seg på grafen? Endrer grafen seg?
Forklar under:

Oppgave 4
Nå skal dere endre verdiene på b. Prøv å endre verdien til b først et negativt tall, så null og deretter positivt tall. Eksempel -10, 0, +10. Observer hver gang hvordan grafen endrer seg.
Kan dere se hvilken akse grafen flytter på seg i koordinatsystemet?
Kan dere også se hvordan den flytter seg i koordinatsystemet? Eksempel om bunnpunktet flytter seg i en bestemt bue.
Går funksjonen gjennom et punkt (0,1) på y-aksen konstant?
Kan man argumentere for at det alltid går gjennom dette punktet uavhengig hva a og b er?
Beskriv under:

Utfordring 1:

Ved å bare se på grafen, er det noen steder grafen vokser raskere?

Utfordring 2:

Prøv å lag et program/kode der dere skal plotte en tredjegradslikning. Kopier inn koden under. (Kopier inn det dere fikk til selv om dere ikke ble ferdig.)