Programmering som inngangsport til algoritmisk tenking

Ein mixed methods studie av 1P-elevars erfaringar, forståing og syn knytt til programmering i matematikkundervisninga

Stian Hirth

Masteroppgåve i matematikkdidaktikk MAT399K

Matematisk institutt

UNIVERSITETET I BERGEN

20. desember 2020

Forord

Ei lang reise er ved vegs ende, og det arbeidet som eg har lagt ned i løpet av mine 5 år på Universitetet i Bergen koker ned til denne oppgåva. Eg ser tilbake på ei utruleg fin tid prega av glede, frustrasjon, meistring og heftige skippertak. Heldigvis har sjølvkontrollen vore såpass god at eg dette semesteret har klart å arbeide jamt og trutt med denne oppgåva, og eg er fornøygd og stolt over resultatet. Dette er ei oppgåve som ikkje kunne ha blitt til utan hjelp og støtte frå eit knippe personar som eg i dette forordet ønsker å takke.

Den største takken går til elevane som stilte opp for å delta i masterprosjektet mitt. Elevane sette av tid og innsats, og utan desse elevane hadde denne oppgåva ikkje vore mogeleg å få til. Tusen takk! Lærarane i klassen der dette prosjektet vart gjennomført fortener også ein stor takk. Ikkje berre inviterte dei meg inn i klassen med opne armar, men dei bidrog også med nyttige tips og innspel undervegs i arbeidet. Det sette eg veldig pris på.

Å gå laus på eit masterprosjekt er stort og overveldande, og då hjelper det å ha personar med ekspertise og erfaring i ryggen. I den forbindelse ønsker eg å takke hovudrettleiaren min, Johan Lie, for støtte, ro og eksepsjonell god rettleiing i løpet av prosjektet sin gang. Det har vore trygt og godt å vite at eg alltid kunne ta kontakt med deg om det var noko eg lurte på. Eg ønsker også å takke birettleiaren min, Bettina Dahl Søndergaard, som har vore ein svært verdifull ressurs då kunnskapane mine frå STAT110 svikta. Det kvantitative datamaterialet i denne oppgåva hadde ikkje vore like fint framstilt utan di hjelp.

I løpet av mine 5 år i Bergen har eg vorte kjend med utruleg mange fine personar, og ein stor del av desse personane er dei som eg har lært å kjenne på lektorstudiet. Eg ønsker derfor å takke mine eminente medstudentar, som gjennom samarbeid, diskusjonar, kos og moro har gjort livet i Bergen til ein fest. Spesielt ønsker eg å takke Andreas som heldigvis alltid var like dårleg forberedt som meg, Linnea som hadde kontroll på det eg ikkje hadde kontroll på, Jon Asgeir som stod for studiets høgdepunkt; praksisperioden med deg var skamnice, Martha som aldri gjekk lei då eg bad ho forklare same konsept 30 gonger, Ingvild som var hovudbidragsytaren til at lektorgjengen faktisk møttest utanfor realfagsbygget, og Kristin som alltid lyste opp tilværet med sin utømmelege kjelde av fun-facts.

Til slutt ønsker eg å rette ein takk til resten av vennane, kollegaane og familien min som har støtta meg undervegs i studiet og masterarbeidet. Denne sida blir for liten til å nemner dykk alle, men nokon som må nemnast er mamma og pappa, som alltid gjorde det klart at eg var velkommen heim til Voss om eg trengte ei avkopling i arbeidet, og Sindre, Espen og Jørgen som sørga for at eg gjekk og vasa i dosarland når eg skulle ha øvd til eksamen.

Stian Hirth

Bergen, 20.12.2020

Samandrag

Målet med denne studien var å kartlegge tre aspekt knytt til 1P-elevar sitt fyrste møte med bruk av programmering i matematikkundervisninga; deira *erfaringar* med programmering før undervisninga tek til, og deira *forståing* av programmering og *syn* på programmering etter at undervisninga som inkluderte programmering vart avslutta. Oppgåva kartlegg dette frå eit elevperspektiv.

Undervisninga gjekk føre seg i totalt 4 økter, der øktene varte enten i 1 time og 30 minutt eller 2 timar og 45 minutt.

Studien tek i bruk ein kombinasjon av kvantitative og kvalitative forskingsmetodar (mixed methods). For å måle dei tre aspekta som studien har som mål å kartlegge gjennomførte elevane ei spørjeundersøking før programmeringsundervisninga tok til og ei spørjeundersøking etter at programmeringsundervisninga var gjennomført. Basert på svara i desse undersøkingane vart eit representativt utval av elevane plukka ut til å delta i forskingsintervju for å klarare kaste lys over dei tre aspekta. Observasjon av undervisninga har også vorte nytta som metode.

Oppgåva legg blant anna til grunn Jean Piaget sine teoriar kring forståing og kunnskap. Det vart også tatt i bruk kartlegging av elevsyn gjennom elevane sin kognitive dimensjon, emosjonelle dimensjon, og dimensjon knytt til motivasjon.

Det vart funne at 21% av elevane i klassen hadde tidlegare erfaring frå det som dei hugsar og/eller oppfattar som programmering. Funna i studien tyder på at undervisningsøktene har bidratt til ei generell auka konseptuell forståing av programmering hos elevane, men at den prosedyremessige forståinga av programmering generelt sett ikkje synast å ha auka i særleg stor grad. I tillegg tyder funna i studien på at elevane sitt syn på programmering i matematikkundervisninga generelt sett er prega av låg grad av fornøyelse, minkande grad av motivasjon, og låg, men etter kvart noko stigande, sjølvtillit i arbeidet.

Studien har produsert datamateriale som tyder på at elevane i klassen saknar eit sterkare fokus i undervisninga på nytteverdien av programmering som eit verktøy i arbeidet med matematikk. Det er sannsynleg at elevane sine vanskar med å sjå denne nytteverdien har gjort utslag på funna i undersøkinga.

Innhaldsliste

1 INNLEIING

1.1 Bakgrunn for val av tema

Algoritmisk tenking er eit omgrep som gjer sin debut i læreplanane i matematikk som vert innført som følgje av fagfornyinga i skuleåret 2020-21. I matematikk 1P er algoritmisk tenking nemnt under kjerneelementet «Utforsking og problemløysing», og det vert her beskrive på følgande måte:

«Algoritmisk tenking er viktig i prosessen med å utvikle strategiar og framgangsmåtar for å løyse problem og inneber å bryte ned eit problem i delproblem som kan løysast systematisk» (Utdanningsdirektoratet, 2020a, side 2)

Innføringa av dette temaet i klasserommet kan gjerast på fleire ulike måtar, og ein av dei er programmering i kodeprogram. Programmering kan nyttast som eit verktøy i klasserommet, både for å fremme forståinga av algoritmsik tenking, men også som eit hjelpemiddel for å fremme forståinga av dei temaa der programmering vert nytta. Dette kan vere ei utfordring både for lærarar og elevar. Lærarar er plikta til å følgje læreplanen, og må derfor nytte programmering der det krevst, sjølv om dette ikkje nødvendigvis er noko som læraren har tidlegare erfaringar med. For elevane, som kanskje allereie syns det er nok å tenke på innanfor matematikken, kan innføringa av programmeringa i klasserommet verke overveldande og til dels unødvendig. Dette var tankar som slo meg då eg fekk vite at den algoritmiske tenkinga skulle innførast som del av matematikkfaget i dei nye læreplanane.

Eg bestemte meg for å ta tak i dei tankane som eg hadde omkring algoritmisk tenking og programmering, og nytte dei til noko produktivt. Desse tankane var frøet som har vekst til å forme denne masteroppgåva. Lærarane som sett i gang matematikkundervisning som inkluderer algoritmisk tenking og programmering i skuleåret 2020-21 er på eit sett historiske, då desse lærarane er dei fyrste som sett i gang med denne undervisninga oppfordra av læreplanane i den norske skulen. Sidan eg skulle skrive mastergraden min i haustsemesteret 2020 augna eg mogelegheita til å få delta i eit av desse klasseromma der den nye undervisninga skulle ta til, for å forsøke og kartlegge enkelte element knytt til elevane si oppfatning av denne undervisninga.

Dei elementa som eg har valt å forsøke og kartlegge i denne masteroppgåva er elevane si opphavelege erfaring med bruk av programmering, og deira forståing og syn knytt til programmering etter undervisningssekvensar som inkluderer bruk av verktøyet. Det kan tenkast å vere nyttig for andre lærarar å sjå kva desse målingane resulterer i etter bruk av undervisninga som læraren i dette prosjektet har nytta, samtidig som det kan bidra til å drive den matematikkdidaktiske forskinga på dette feltet vidare.

1.2 Tidlegare forsking

Før prosjektet vert presentert kan det vere nyttig å studere den tidlegare forskinga som er blitt gjort på dette feltet. Dette for å få ei oversikt over kva resultat som er vorte produsert i fagfeltet tidlegare, og opne opp for samanlikningar og anvendingar av forskinga som allereie eksisterer på feltet. Dette kapittelet er delt inn i to; i delkapittel 1.2.1 vert noko av den tidlegare forskinga på algoritmisk tenking i klasserommet presentert, mens delkapittel 1.2.2 tek føre seg forskinga gjennomført på programmering i klasserommet.

1.2.1 Algoritmisk tenking

Det finst mange studiar som omhandlar implementeringa av algoritmisk tenking i undervisninga, men dei fleste av desse (spesielt dei studiane der det er samla inn datamateriale omkring situasjonar der ein prøver ut opplegg kring algoritmisk tenking i klasserom) tek føre seg programmering som del av den algoritmiske tenkinga. Noko av denne forskinga vert gjennomgått i delkapittel 1.2.2. Det vil her bli gjennomgått nokon moment frå tekstane som tek føre seg implementeringa av algoritmisk tenking i klasserommet, men der programmering ikkje står sentralt. Meir om den algoritmsike tenkinga følger i kapittel 2.5.

I doktoravhandlinga til Joshua Levi Weese blir det sett fokus på at det kan vere vanskeleg å skilje algoritmisk tenking og problemløysing (Weese, 2017, side 12). I delkapittel 2.5.2 vil samanhengen mellom den algoritmiske tenkinga og problemløysing bli utforska nøyare, men for denne delen av oppgåva er nok å seie at algoritmisk tenking er ein måte å løyse problem på. Avhandlinga til Weese tek føre seg den algoritmiske tenkinga i amerikanske læreplanar, men

sidan problemløysing også er ein sentral del av dei norske læreplanane i matematikk (Utdanningsdirektoratet, 2013, side 2) kan ein dra parallellar til norske klasserom. Mange av aspekta og øvingane ved den algoritmiske tenkinga er ikkje nødvendigvis så unike at dei berre kan brukast innanfor fagfeltet informatikk (Weese, 2017, side 12), noko som gjer at hovudmomenta i den algoritmiske tenkinga moglegvis har vorte undervist i norske skular i lengre tid.

I 2009 vart det i USA sett i gang eit prosjekt der målet var å komme fram til ein operasjonell definisjon for det engelske uttrykket for algoritmisk tenking, «computational thinking» (meir om linken mellom det norske og engelske uttrykket i delkapittel 2.5.1), for at emnet med klarheit kunne implementerast i dei engelske læreplanane (Barr og Stephenson, 2011, side 50). Til dette arbeidet vart det sett saman ei gruppe bestående av 26 undervisarar og administratorar.

I arbeidet fram mot ei beskriving av computational thinking vart det kartlagt fleire interessante aspekt. Det blir blant anna poengtert at denne typen tenking gjer seg gjeldande i all anna form for resonnering, og at ein med dette verktøyet kan løyse alle slags problem på tvers av fagfelt; til dømes kvantefysikk, avansert biologi og utvikling av nyttige digitale hjelpemiddel (Barr og Stephenson, 2011, side 51). Gruppa spådde også at elevar som vert sett ovanfor læringssituasjonar der mogelegheitene for å nytte seg av computational thinking var i overflod etter kvart vil vise større grad av evne til å løyse problem på ein meir kompetent (eng: *fluid*) måte (Barr og Stephenson, 2011, side 51).

1.2.2 Programmering

Det finst ein god del forsking som tek føre seg bruk av programmering i klasserommet. I dette delkapittelet vil eit utval frå denne forskinga bli presentert.

I noko av den aller fyrste forskinga som vart gjennomført på bruk av programmeringsverktøy i undervisningssamanheng vart programmeringsverktøyet Logo tatt i bruk. I ein studie gjennomført av Douglas Clements og Michael Battista i 1989 vert bruken av Logo innanfor fagfeltet geometri undersøkt. Denne studien vart gjennomført på 48 tilfeldig utvalde tredjeklassingar, og gjekk føre seg over 26 veker (Clements og Battista, 1989, side 455).

Deltakarane vart delt inn i to grupper, der ei av gruppene fekk undervisning som inkluderte bruk av Logo, mens kontrollgruppa fekk undervisning som inkluderte andre digitale verktøy.

Gruppa av tredjeklassingar som vart underviste ved bruk av Logo møttest i 45-55 minutt tre gonger i veka, mens kontrollgruppa møttest ei gong i veka. Det vil seie at den eksperimentelle gruppa møttest tre gonger så ofte som kontrollgruppa over ein periode på 26 veker (Clements og Battista, 1989, side 452-454). I forkant av denne undervisninga vart det gjennomført ein matematisk test for å kartlegge eventuelle forskjellar i nivå på dei to gruppene, men poengscoren til dei to gruppene på denne testen var så å seie identiske (Clements og Battista, 1989, side 455). Etter undervisningsperioden vart det gjennomført testar og individuelle intervju med elevane. Både testane og intervjua hadde som mål å kartlegge elevane sine kompetansar innanfor geometri. Testen innehaldt blant anna spørsmål om vinkeldefinisjonar, geometriske objekt og aritmetikk.

Alt i alt vart det kartlagt at dei elevane som hadde nytta logoprogrammering i undervisninga hadde ein tendens til å gi svar som var meir matematisk samanhengande og abstrakt, samanlikna med kontrollgruppa (Clements og Battista, 1989, side 466). Resultata til eksperimentgruppa var uniformt høgare enn kontrollgruppa sine resultat. Forfattarane konkluderer frå dette at resultata støtter opp under bruken av logo i undervisning, som eit rammeverk for læring om geometri, gitt at undervisningsmiljøet der logo vert implementert er strukturert på ein måte som guidar elevane sine erfaringar innan dette rammeverket (Clements og Battista, 1989, side 466).

To namn frå matematikkdidaktikken som også har forska på bruk av programmering i matematikkundervisning i er Uri Leron og Anna Sfard. I 1996 vart det i *International Journal of Mathematical Learning* publisert ein tekst av Leron og Sfard som tek utgangspunkt i to oppgåver, ei gitt til studentar i eit matematikkfag og ei gitt til studentar i eit programmeringsfag. Sjølv om oppgåva som omhandla programmering var tydeleg meir kompleks enn matematikkoppgåva viste det seg at eit større tal elevar klarte å løyse programmeringsoppgåva. Teksten gjennomgår, frå ein undervisar sitt synspunkt, kva forklaringa bak dette kan vere. Det teksten konkluderer med er at klassiske arbeidsmetodar innanfor matematikk ofte ikkje har same kapasitet til å mobilisere studentar si evne knytt til å aktivere tankegangar rundt problemløysing som det bruk av programmering har. Teksten går også inn på kva haldningar elevar har til matematikk og programmering som to distinkte fagområde, og korleis desse handlingane kan ha verka inn på resultatet som viser at elevane i større grad klarte å løyse programmeringsoppgåva samanlikna med matematikkoppgåva (Sfard og Leron, 1996, side 193-194).

I *Advanced Mathematical Thinking and the Computer* (Dubinsky og Tall, 1991) vert det sett på korleis PCar kan inkluderast i matematikkundervisninga, og ein ser i denne teksten at sjølve programmeringshandlinga kan fremme forståing av matematisk kunnskap. Dette er fordi at konstruksjonen av kode i eit programmeringsspråk kan sjåast på som ein parallell til konstruksjonen av ein elev sine underliggande matematiske prosessar (Dubinsky og Tall, 1991, side 234-235). Når dei matematiske prosessane vert konkretisert i eit kodeprogram er det større sjanse for at elevane som arbeider med koden enklare får implementert desse prosessane som ein del av sin kunnskap (Dubinsky og Tall, 1991, side 235).

Aspektet rundt konstruksjon av matematisk kunnskap ved bruk av programmering vert også støtta opp under av Leron og Dubinsky sin tekst «An Abstact Algebra Story» frå 1995, som beskriv korleis programmering vart implementert som ein del av pensumet i eit universitetskurs om abstrakt algebra. Den høge graden av abstraksjon i emnet gjer det særs vanskeleg å forstå, og programmering vert som eit resultat av dette brukt som eit forsøk på å gjere fagstoffet i emnet meir forståeleg. Det vert i teksten sett fokus på at det er under konstruksjonen av programma at læring og forståing oppstår. Dette er fordi det matematiske konseptet, samtidig som det vert konstruert i kode, vert konstruert i studenten sitt sinn (Leron og Dubinsky, 1995, side 230). Ein ser både i denne teksten, og Dubinsky og Tall sin tekst, at det er denne dualiteten, med konstruksjonen av kode som beskriv eit konsept, og konstruksjonen av kunnskap rundt dette konseptet hos elevane, som kan verke fordelaktig ved å dra inn programmering i matematikkundervisninga. Det at forståinga av matematiske konsept kan verte styrka ved bruk av programmering ser ein også evidens på i Sfard og Leron sin tekst. Resultata frå observasjonen i matematikk- og programmeringsfaga i denne teksten viser at bruk av programmering gjer at studentane i større grad klarer å nytte matematiske konsept til å løyse oppgåver.

Det er også blitt gjennomført forsking på implementering av programmering i matematikkundervisninga i nyare tid. Studien i «Computer Programming in the Lower Secondary Classroom: Learning Mathematics» (Lie et al., 2017) tek føre seg bruk av programmeringsspråket Scratch i norsk matematikkundervisning, der elevane er frå 13 til 16 år gamle. Opplegget som teksten beskriv gjekk føre seg over tre veker, med økter som totalt utgjorde om lag fire timar per veke. Forskarane deltok i opplegget der elevane jobba med å løyse problemløysingsoppgåver ved bruk av Scratch, og samla inn kvalitativt datamateriale frå desse øktene ved bruk av lyd- og videoopptakarar (Lie et al., 2017, side 30). I studien vart det observert fleire eksempel på bruk av algoritmisk tenking for å løyse opne

problemløysingsoppgåver. Datamaterialet i oppgåva gir også indikasjonar på at programmeringsarbeidet utvida elevane sine kjennskapar til strukturelle og operasjonelle matematiske konsept knytt til programmering (Lie et al., 2017, side 33).

1.3 Problemstilling og forskingsspørsmål

I dette prosjektet vil det bli følgd ein 1P-matematikklasse som for fyrste gong får programmering inkludert som del av pensumet. Som beskrive i kapittel 1.1 vil elevane si erfaring, forståing og syn knytt til programmering bli forsøkt kartlagt. Oppgåva vil derfor bli sentrert rundt følgande problemstilling:

Kva erfaringar har elevar i ein 1P-klasse med programmering før undervisninga om programmering tek til, og kva er deira forståing og deira syn knytt til programmering som verktøy til bruk i matematikk 1P etter denne undervisninga?

Dette er ei omfattande problemstilling, og i eit forsøk på å gjere komponentane i den klarare vert det etablert tre forskingsspørsmål, der svara på desse vil gi eit fullgodt svar på problemstillinga i oppgåva. Problemstillinga kan brytast ned til desse tre individuelle forskingsspørsmåla:

1. Kva erfaringar har elevar i ein 1P-klasse med programmering før undervisninga tek til?
2. Korleis er elevane i ein 1P-klasse si forståing av programmering etter undervisninga?
3. Kva er elevane i ein 1P-klasse sitt syn på programmering etter undervisninga?

1.4 Oppbygging av oppgåva

Oppgåva er bygd opp av kapittel og delkapittel, og oversikten over desse er i finne i innhaldslista.

Kvart av dei 6 hovudkapitla tek føre seg ein konkret del av oppgåva, og desse er igjen organiserte i delkapittel som gir eit oversiktleg oppsett av hovudmomenta i kvart kapittel. Dette

kapittelet – kapittel 1 – innleiar oppgåva ved å argumentere for tema, summere opp tidlegare forsking og presentere problemstillinga som oppgåva er sentrert rundt. I kapittel 2 vert det gjort eit djupdykk i den aktuelle teorien som kan bidra til å støtte opp under oppgåva, og som vert brukt til å belyse problemstillinga tydlegare. Kapittel 3 presenterer metodane som har vorte nytta i prosjektet, og diskuterer også prosjektet sin reliabilitet, validitet og etikk. I kapittel 4 vert datamaterialet opparbeida ved bruk av metodane i prosjektet presentert og analysert. Kapittel 5 diskuterer datamaterialet frå kapittel 4 i lys av teorien som vart presentert i kapittel 2. Til slutt knyt kapittel 6 hovudmomenta frå diskusjonen saman til ein konklusjon som gir svar på problemstillinga i oppgåva.

Etter kapittel 6 kjem referanselista og ein oversikt over oppgåva sine vedlegg.

2 TEORI

2.1 Den kognitive utvikling

Den sveitsiske psykologen og biologen Jean Piaget arbeidde med spørsmål knytt til korleis individ utviklar kunnskap. På bakgrunn av dette arbeidet utvikla Piaget ein omfattande teori knytt til korleis personar utviklar seg intellektuelt, og det er den logiske resonnerande tenkinga som står sentralt i teorien (Evenshaug og Hallen, 2000, side 103, Kunnskapsdepartementet, 2016). Teorien til Piaget har ikkje vore fri for kritikk, blant anna har Vygotsky kritisert Piaget for å i for liten grad ta omsyn til dei sosiale og kulturelle aspekta som utviklar born sin kunnskap (Evenshaug og Hallen, 2000, side 116-117). Likevel har Piaget sin teori hatt enorm betydning for vår forståing av tenkinga si utvikling hos born og ungdom (Evenshaug og Hallen, 2000, side 103). Dette er grunnen til at det er inkludert eit delkapittel om Piaget, og hans teori. Når ungdommane i dette prosjektet nytter programmering som eit verktøy i matematikkfaget for fyrste gong kan det vere nyttig å kople inn Piaget for å undersøke om det kan vere mogeleg å forstå korleis elevane utviklar kunnskap i møte med dette verktøyet.

Hovudpilaren i Piaget sitt arbeid er at tileigning av kunnskap er ein prosess (Evenshaug og Hallen, 2000, side 104). Kunnskap vert utvikla ved å interagere med ytre ting og hendingar, samtidig som ein reviderer gamle oppfatningar som ikkje lenger held stand som eit resultat av dei interaksjonane som vert gjennomført. Det er gjennom arbeid med konkrete ting, som kan handlast med, at kunnskap vert konstruert (Evenshaug og Hallen, 2000, side 104). Piaget skil i sitt arbeid mellom to ulike typar kunnskap; *operativ* kunnskap, som vert tileigna gjennom logisk læring, og *figurativ* kunnskap, som er faktakunnskap opparbeida gjennom interaksjonar med ulike objekt (Evenshaug og Hallen, 2000, side 104)

Piaget beskriv det «å kunne noko» som å vere i stand til å handle overfor denne tingen (Evenshaug og Hallen, 2000, side 104). I den forbindelse definerer Piaget eit *skjema* som eit spesifikt handlingsmønster som opererer på miljøet for å nå eit mål. *Kognitive skjema* er indre representasjonar av desse handlingsmønstera. Det er i prosessar der personar endrar desse skjemaa at den intellektuelle utviklinga hos eit menneske skjer (Evenshaug og Hallen, 2000, side 104). Prosessen der skjemaa vert endra skjer i all hovudsak som to delprosessar som utfyller kvarandre; *assimilasjon*, der individet forsøker å tilpasse omgjevnadane til seg sjølv,

og *akkommodasjon*, der individet forsøker å tilpasse seg sjølv til miljøet (Evenshaug og Hallen, 2000, side 104). Begge desse prosessane bidrar til å endre våre mentale skjema til eit nivå av høgare tenking. Assimilasjon hos eit individ skjer som ein konsekvens av at personen møter kognitive konfliktar. Desse kognitive konfliktane fører til ein kognitiv ubalanse hos personen, noko som tvinger fram nye assimilasjonar hos personen, og ei naturleg reorganisering av tankar vil følgje etter (Alves, 2014, side 25).

Som nemnd tidlegare har teorien til Piaget vorte kritisert av blant anna Lev Vygotsky. Hovudforskjellen mellom Piaget og Vygotsky er summert opp i følgande sitat:

> *«(...)* [Piaget og Vygotsky] *differ in that , unlike Piaget, Vygotsky thought that the assimiliation of new information does not have to wait for an appropriate level of development but must, on the contrary, produce that development»* (Alves, 2014, side 24)

Vygotsky meiner altså ikkje at assimilasjon og reorganisering av tankar skjer som ein naturleg konsekvens frå kognitiv ubalanse, slik som Piaget. Til tross for denne forskjellen vert både Vygotsky og Piaget rekna for *konstruktivistar*. Ein konstruktivist meiner at kunnskap og sanning begge er produkt av menneskelege undersøkingar og oppfinningar, heller enn at kunnskap er noko som vert gitt til menneske frå tekst eller naturen (Olson, 2001, side 104). Ein annan konstruktivist som har bygd sine teoriar ved å ta utgangspunkt i både Piaget og Vygotsky sitt arbeid er den amerikanske psykologen Jerome Bruner. Bruner, Piaget og Vygotsky deler desse tre fundamentale meiningane (Olson, 2001, side 106):

1. Kognisjon vert best sett på som generisk eller som under utvikling.
2. Sinnet (eng: *the mind*) er konstruert av aktivitetane til subjektet.
3. Strukturane som vert konstruert hos personar openberrar seg i bevisstheita deira.

Jerome Bruner beskriv i sitt arbeid det han kallar for stillasbygging (eng: *scaffolding*). Dette omgrepet vert brukt for prosessen der ein undervisar minkar sin kontroll og hjelp i ein undervisningssituasjon der ein person som vert undervist etter kvart byrjar å meistre ei gitt oppgåve. Dette gjer at personen som vert undervist betre kan konsentrere seg om dei vanskelege aspekta ved det som han eller ho prøver å lære seg (Hyland, 2009, side 118). Denne måten å undervise på er ikkje vanskeleg å kjenne att i skulen. Læraren vil ofte gjennomgå eit og eit tema knytt til eit spesifikt konsept, og vil etter kvart i undervisninga ta for gitt at desse temaa er forstått når pensumet beveger seg vidare til andre tema som bygger på det som har blitt

gjennomgått. I ein slik situasjon gir læraren meir og meir slepp på kontrollen og hjelpa knytt til det overordna emnet, i eit forsøk på å få elevane til å ta til seg kunnskap rundt temaa som det bygger på.

Bruner følgde Vygotsky sitt syn på lingvistisk sosialisering (Bakhurst og Shanker, 2001, side 10). Dette synet bygger på vissheita om at språk og kultur som konsept ikkje kan skiljast frå kvarandre, i og med at dei er så sterkt knytt til kvarandre. Eit sitat som illustrerer denne tilknytinga er dette:

> *«Culture is the medium of language and language the vehicle of culture»* (Bakhurst og Shanker, 2001, side 10)

Eit anna viktig haldepunkt i Vygotsky sitt syn på lingvistisk sosialisering er at han avviser Jerry Fodor sin idé om at eit born si kognitive utvikling fortsett gjennom formuleringa av hypotesar. Vygotsky meiner at mange bestanddeler av eit born sine intellektuelle krefter vert absorbert frå kulturen, i motsetnad til at dei vert avleia av borna frå ein hypotese som er vorte uttenkt og testa (Bakhurst og Shanker, 2001, side 10).

2.2 Forståing innanfor matematikk og teknologi

Ein av tinga som denne studien har som mål å kartlegge er elevane si forståing av programmering som ein verktøy i matematikkfaget. I den forbindelse vil det på førehand vere aktuelt å ta føre seg forståing som konsept, og sjå på korleis forståing kan tolkast generelt, innanfor matematikkfaget og i møte med teknologi.

2.2.1 Generell forståing

Det finst mange måtar å tolke ordet «forståing», og Douglas Newton gir fleire eksempel på beskrivingar av ordet. Her er eit utval av beskrivingane (Newton, 2012, side 12):

1. Eit mentalt forsøk på å kople noko som er gitt med noko anna enn seg sjølv.

2. Prosessen der sinnet (eng: *the mind*) vel, puslar saman og sett i system dei relevante observerte faktum, og avviser det irrelevante, inntil det har blitt sydd saman eit logisk og rasjonelt 'teppe' (eng: *quilt*) av kunnskap.
3. Samankoplinga av faktum, relateringa av nyleg tileigna informasjon til det som allereie er kjend, vevinga av bitar av kunnskap til ein integrert og samanhengande heilskap.

Alle beskrivingane ovanfor tek for seg samankopling av faktum til eit større organisert system, og det er dette systemet som resulterer i forståing. Denne samankoplinga, enten det er samansyinga til kunnskapsteppet i beskriving 2 eller samanvevinga til heilskapen i beskriving 3, kan sjåast på som alternative måtar å forklare Piaget sin teori om utvikling av mentale kognitive skjema.

Newton hentar beskrivingane av forståing ovanfor frå andre verk, men gir også si eiga beskriving av forståing, der han samanliknar prosessen med å strikke:

> *«The act of knitting can parallel the process of understanding and the garment it produces is like the mental product that results»* (Newton, 2012, side 12)

I denne beskrivinga kan det mentale produktet produsert bli sett på som eit av Piaget sine kognitive skjema som vert produsert ved assimilasjon og akkommodasjon. Frå denne samanlikninga meiner Piaget at berre dei skjema som bidreg til å svare på spørsmålet om «kvifor» fortener å bli kalla forståing (Newton, 2012, side 12-13). Til dømes kan ein matematikkelev vite at verdien til pi er omtrent lik 3,14, men i følgje Piaget kan ein ikkje sjå på eleven sin kunnskap om pi som forståing før eleven til dømes kan forklare *kvifor* verdien til pi er omtrent lik 3,14. Denne typen forståing er verdifull fordi den er så fleksibel at den kan nyttast i fleire forskjellige typar situasjonar (Newton, 2012, side 13).

2.2.2 Matematisk forståing

Innanfor matematikkfaget kan forståing delast inn i to typar; konseptuell (eng: *conceptual*) og prosedyremessig (eng: *procedural*) forståing. Konseptuell forståing referer til ein elev si forståing av matematiske konsept (Sierpinska, 1994, side 6-7), som til dømes likskap, prosent og parallellitet. Den prosedyremessige forståinga refererer til ein elev si forståinga av tal, og operasjonar på desse tala (Sierpinska, 1994, side 48). Ein seier at ein elev har prosedyremessig forståing om han/ho er i stand til å operere på tal på ein korrekt måte for å oppnå eit ønska

resultat (Sierpinska, 1994, side 48). Ein kan også seie at prosedyremessig forståing er det same som å ha ei instrumentell forståing av algoritmar (Newton, 2012, side 16) (sjå delkapittel 2.5 for definisjon av algoritme). Konstruksjonen av forståing i matematikk kjem av evna til å kunne gjere logiske koplingar basert på tidlegare forståingar, og stiller krav til at eleven som gjer desse koplingane arbeider med matematikken på ein kreativ måte (Newton, 2012, side 16).

Eit omgrep som gjer seg gjeldande i samband med matematisk forståing er *fornuftsgrunnlag*. Ein elev sitt fornuftsgrunnlag for læring kan beskrivast som eleven si hensikt med å lære, og er avgjerande for korleis læringa til eleven går føre seg (Mellin-Olsen og Hoel, 1984, side 22 & 37). Fornuftsgrunnlaget er sett saman av fleire komponentar (Mellin-Olsen og Hoel, 1984, side 37):

- Eleven si interesse for lærestoffet
- Eleven si oppfatning av nyttigheita til lærestoffet
- Eleven si oppleving eller glede gitt av lærestoffet
- Lærestoffet si tilfredsstilling av eleven si nysgjerrigheit

Grovt fordelt kan ein dele inn i to ulike fornuftsgrunnlag; *instrumentelt fornuftsgrunnlag* (IFG) og *sosialt fornuftsgrunnlag* (SFG). Om ein elev har eit instrumentelt fornuftsgrunnlag lærer eleven fordi lærestoffet er ein del av skulen og er viktig for framtida, mens ein elev med eit sosialt fornuftsgrunnlag lærer fordi han syns lærestoffet er viktig (Mellin-Olsen og Hoel, 1984, side 41). I forbindelse med IFG vil elevane gi signal om at det viktigaste er å få riktige svar utan at dei nødvendigvis kan gjere reie for desse svara, mens elevar i forbindelse med SFG vil gi signal som tilseier at dei er meir opptatt av kunnskapane i seg sjølv, meir enn påskjøninga for å ha dei (Mellin-Olsen og Hoel, 1984, side 42).

I forbindelse med presentasjonen av fornuftsgrunnlag vert følgande tese lagt fram:

> *«Dersom elevens fornuftsgrunnlag for læring i hovudsak baserer seg på IFG, og dersom eleven får tilstrekkelig med negativ kommunikasjon frå skolen om sine skoleprestasjoner, vil eleven ikke lenger ha noe fornuftsgrunnlag for å lære på skolen. Læringen vil da opphøre.»* (Mellin-Olsen og Hoel, 1984, side 43-44)

I forbindelse med desse fornuftsgrunnlaga er det aktuelt å dra fram to typar forståing når det kjem til reglar innanfor matematikken; *regeloppfatning* og *strukturoppfatning*. Regeloppfatning viser til forståing kring korleis ein regel skal brukast i matematikken, mens strukturoppfatning

viser til forståinga kring regelen sin struktur (Mellin-Olsen og Hoel, 1984, side 32). Eit eksempel på desse to forståingane kan ein sjå ved å ta utgangspunkt i følgande brøk:

$$\frac{3 * 7}{6 * 3}$$

Ein elev som har regeloppfatning knytt til si forståing av forkorting av brøkar vil argumentere for at ein kan stryke 3-talet i teljar og 3-talet i nemnaren til brøken fordi ein alltid kan stryke like tal i teljar og nemnar i ein brøk. Ein elev som derimot har strukturoppfatning knytt til si forståing av forkorting av brøk vil argumentere for at ein kan separere brøken, og deretter sjå at brøken blir forkorta på denne måten:

$$\frac{3 * 7}{6 * 3} = \frac{7}{6} * \frac{3}{3} = \frac{7}{6}$$

2.2.3 Forståing i møte med teknologi

Boka «Teaching for understanding with technology» (Wiske et al., 2005) tek for seg korleis ein kan fremme forståing ved bruk av teknologi i undervisninga. Forfattarane definerer det å forstå eit emne på følgande måte:

> *«(…) being able to perform flexibly with the topic – to explain, justify, extrapolate, relate, and apply in ways that go beyond knowledge and routine skill. Understanding is a matter of being able to think and act flexibly with what you know.»* (Wiske et al., 2005, side 5)

Graden av forståing av eit emne relaterer seg altså til i kva grad ein er i stand til å nytte det ein forstår på fleksible måtar for å løyse problem som strekkjer seg forbi kunnskap og rutine. Ein kan her sjå grunnen til at Piaget fremmer viktigheita av at forståing er knytt til det å svare på spørsmål om kvifor. Definisjonen ovanfor tek utgangspunkt i forsking som viser at læring er ein aktiv prosess, i motsetnad til å berre absorbere informasjon eller utføre grunnleggande operasjonar (Wiske et al., 2005, side 5), noko som er heilt i tråd med Piaget, Vygotsky og Bruner sine tankar rundt kognisjon.

For at elevar skal kunne opparbeide seg forståing innanfor eit emne må ein stor del av arbeidet som elevane gjer vere aktivitetar som krev at elevane tenker kreativt (Wiske et al., 2005, side

7), noko som, poengtert tidlegare i teksten, også er essensielt når det kjem til konstruksjon av forståing innanfor matematikkfaget. For å engasjere elevane med denne typen arbeid er det eit krav at lærarane som underviser legg opp ein sekvens av aktivitetar som byggjer på elevane sine tidlegare kunnskapar og interesser, slik at elevane gjennom dette arbeidet kan opparbeide seg ny kunnskap, saman med evna til å nytte denne kunnskapen på stegvis meir sofistikerte måtar (Wiske et al., 2005, side 7). I dette arbeidet kan teknologi i klasserommet nyttast av læraren på to hovudmåtar; designe pensum som integrerer teknologi på ein slik måte at arbeidet fokuserer på å utvikle elevane si forståing, eller designe pensum på ein måte som gjer det klart korleis teknologi gir ein signifikant fordel når det kjem til læring (Wiske et al., 2005, side 12).

2.3 Matematikksyn

«Syn på matematikk» er eit omgrep som ofte vert nytta innanfor matematikkundervisning, og omgrepet refererer til den systematiske karakteristikken av tru/meiningar (eng: *beliefs*) innanfor matematikk (Sthephani et al., 2019, side 2). Når ein forskar sett ut for å kartlegge elevars syn på matematikkfaget kan dette gjerast på mange måtar, og mange aspekt knytt til matematikksyn kan undersøkast. I nyare forsking på dette feltet er det vanleg å kartlegge elevars syn på matematikk ved å kartlegge tre dimensjonar (Sthephani et al., 2019, side 2): den kognitive dimensjonen (sjølvsikkerheit, vurdering av eigen kompetanse, vanskar innanfor matematikk), den emosjonelle dimensjonen (fornøyelse innanfor matematikk) og dimensjonen knytt til motivasjon (meistringsmål, innsats). Kartlegging av desse tre dimensjonane vil kunne gi eit bilete av ein elev sitt syn på matematikk. Under er det lista nokon avklaringar av ord nytta i beskrivingane av dei tre dimensjonane (Sthephani et al., 2019, side 2):

- Meistringsmål refererer til ein representasjon av eleven si vilje til å konstruere kompetanse og fremme kunnskap gjennom meiningsfull læring.
- Innsats refererer til intensiteten av eleven sin motivasjon.
- Fornøyelse innanfor matematikk er ei haldning som er definert som ein følelsesmessig respons som omfattar positive eller negative følelsar av moderat intensitet og sannsynleg stabilitet.
- Vanskar innanfor matematikk vert sett på som ei tru rundt eigen kompetanse og suksess.

- Vurdering av eigen kompetanse er ei evaluering av kompetansen til å oppdrive eit ønska resultat på ein korrekt måte.
- Sjølvsikkerheit vert betrakta som ei samling av følelsar som relaterer til trua på seg sjølv, og på eigen effektivitet når det kjem til å prestere i ein sosial omgjevnad.

I ein studie henta ein inn data frå 276 elevar frå vidaregåande skular i Indonesia, og studerte korrelasjonen mellom dei ulike faktorane i dei tre dimensjonane (Sthephani et al., 2019, side 3). Studien avdekka låge korrelasjonar mellom faktorane, der den høgaste korrelasjonen var mellom meistringsmål og sjølvsikkerheit (Sthephani et al., 2019, side 3).

Alan Schoenfeld definerer ein elev si matematiske tru (eng: *belief*) som kjensler kring matematikk som disiplin og elevens forhold til matematikk (Schoenfeld, 1989a, side 338). I ein av Schoenfelds studiar vart det, ved bruk av spørjeundersøkingar, kartlagt ein god del aspekt knytt til matematikkundervisning hos amerikanske vidaregåande elevar. Dette vart gjort i eit forsøk på å undersøke korleis desse aspekta knyt seg til elevane si matematiske tru og oppførsel i klasserommet. Schoenfeld fann blant anna at elevane generelt sett ser på matematikk som ein objektiv disiplin som det er mogeleg å meistre gjennom hardt arbeid, og at matematikk i stor grad handlar om å memorere reglar og formlar (Schoenfeld, 1989a, side 343-344). I studien vart det stilt to spørsmål knytt til tidsbruk innanfor matematikk. På spørsmål om kor lang tid elevane meinte ei typisk matematikkoppgåve skal ta låg middelverdien på litt under 2 minutt, mens middelverdien låg på 12 minutt på spørsmål om kva som var ei rimeleg tidsmengd å bruke på ei matematikkoppgåve før ein gir opp, og innser at oppgåva er for vanskeleg til å løysast (Schoenfeld, 1989a, side 345). Desse relativt korte tidsmengdene meiner Schoenfeld kjem som ein konsekvens av årelange erfaringar med korte og konsise matematikkoppgåver, der målet med oppgåvene er å fremme ein følelse av meistring og fornøyelse hos elevane (Schoenfeld, 1989a, side 348).

Studien til Schoenfeld kartla også motivasjonen til matematikkelevane. Han fann at elevane i stor grad arbeider med matematikk for å gjere det bra i faget (Schoenfeld, 1989a, side 346). Samtidig vart det også avdekka at elevane syns pensumet er interessant, og at dei vel å arbeide med faget for å auke evna til å tenke betre generelt sett, i staden for å til dømes unngå å sjå dum ut eller for å imponere lærarane (Schoenfeld, 1989a, side 346-347).

I studien til Schoenfeld (1989a) ser ein at elevane i størst grad arbeider med matematikk utifrå ein *ytre* motivasjon, det vil seie ein motivasjon til å gjere noko fordi det leier til eit ønska resultat

(Ryan og Deci, 2000, side 55). I dette tilfellet arbeider elevane med matematikk i hovudsak for å oppnå eit godt resultat i faget og/eller for å betre tankeprosessane sine, noko som gjer at ein god karakter og/eller eit betre tankesett vert det ønska resultatet som driv motivasjonen til å arbeide med matematikk. Ein skil gjerne mellom to ulike typar ytre motivasjon; *kontrollert* og *autonom*. Kontrollert ytre motivasjon kjem som ein konsekvens av ei form for press eller ein følelse av å verte tvunge, mens autonom ytre motivasjon referer til handlingar gjennomført på eige initiativ (Skaalvik og Skaalvik, 2013, side 147). I Schoenfeld sin studie kan ein kjenne att elevane som arbeider med matematikk berre for å oppdrive eit godt resultat som å vere drive av kontrollert ytre motivasjon, mens elevar som ser kor viktig det er å arbeide med matematikk for å utvikle tankegangane sine som å verte drive av autonom ytre motivasjon. Ein ser også at enkelte av elevane syns pensumet er interessant (Schoenfeld, 1989a, side 346), noko som gjer at ein kan tenkje seg at desse elevane også delvis arbeider med faget fordi dei syns arbeidet i seg sjølv er interessant og fornøyeleg. Dette vil vere evidens for delvis indre motivasjon, altså ein motivasjon som vert drive av sjølve handlinga i seg sjølv (Ryan og Deci, 2000, side 55).

Sosiale normer og *sosiomatematiske normer* relaterer seg til elevane sine matematikksyn. Sosiale normer refererer til regularitetar i interaksjonsmønsteret som regulerer sosiale interaksjonar i klasserommet, mens sosiomatematiske normer er spesifikke til matematikkfaget (Çakır og Akkoç, 2020, side 20). Dei sosiale og sosiomatematiske normene evolverer ilag, og saman indikerer dei distinkte aspekt av ein klasse sin sosiale kontekst (Çakır og Akkoç, 2020, side 20). Det kan i eit klasserom oppstå normer knytt til forventningar rundt rolla som elevar og lærarar skal spele i ulike kontekstar (Çakır og Akkoç, 2020, side 22). Dette er grunnen til at oppførslar som vert observert i ein klasseromsituasjon ikkje ukritisk kan tolkast som indikasjonar på elevane sine syn på matematikk, då det også kan vurderast om deira handlingar er eit resultat av dei etablerte sosiale og sosiomatematiske normene som har blitt etablert i klasserommet.

2.4 Kunnskapsløftet 2020

Læreplanverket *Kunnskapsløftet* vart fyrst innført frå skuleåret 2006-2007 (Kunnskapsdepartementet, 2013, side 20), og vart revidert til skuleåret 2013-2014 (Kunnskapsdepartementet, 2013, side 70). Ved skulestart i skuleåret 2020-2021 vart dei nyaste delane av kunnskapsløftet tatt i bruk i fleire fag i den norske skulen. Arbeidet med denne fagfornyinga har bygd på stortingsmelding nr. 28 (Kunnskapsdepartementet, 2016). Denne fagfornyinga er ei vidareføring av Kunnskapsløftet i ny form, og målet med denne fagfornyinga er gitt her:

> *«Målet med en slik fagfornyelse er bedre læring og forståelse for elevene. Skolen og lærerne skal legge til rette for at elevene tilegner seg solid faglig kunnskap og forståelse, grunnleggende ferdigheter, og at de kan anvende det de lærer i ulike sammenhenger»* (Kunnskapsdepartementet, 2016, side 26)

I matematikkfaget på den vidaregåande skulen skal dei nye læreplanane i matematikk innførast gradvis, og dette arbeidet starta med at Vg1 tok i bruk dei nye læreplanane skuleåret 2020-21. Dette inkluderte matematikk 1T og 1P. I denne oppgåva er det ein 1P-matematikklasse som skal studerast. Grunnen til dette vert gitt i kapittel 3.6.

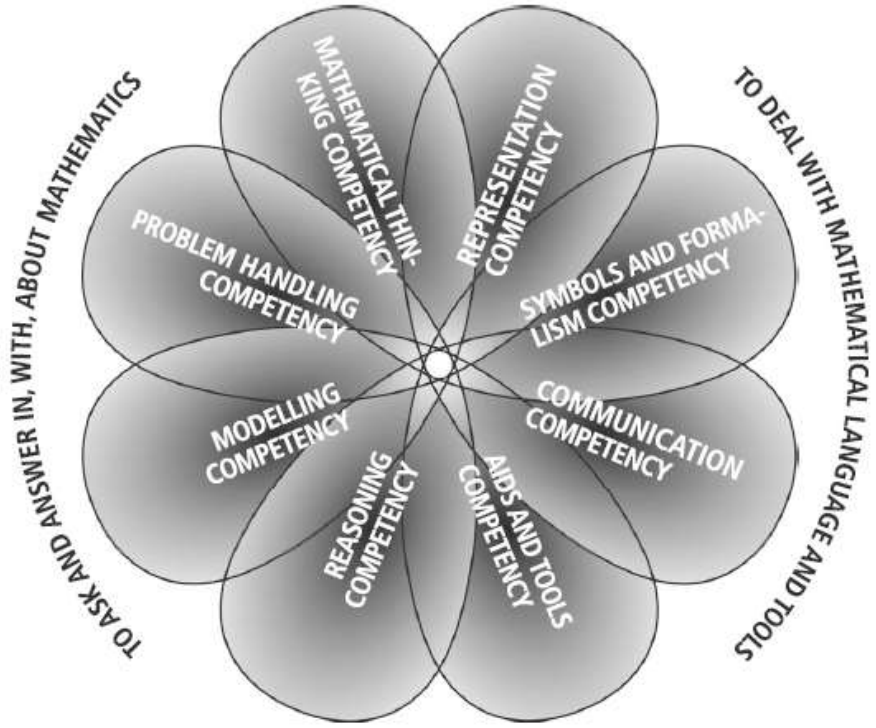
I dei nye læreplanane til matematikk 1P står det:

> *«Matematikk P er eit sentralt fag for å kunne forstå og beskrive forhold og samanhengar i samfunnet gjennom matematisk modellering. Matematikk P skal bidra til at elevane utviklar eit presist språk for kritisk tenking og matematiske problemløysingsstrategiar. Matematikk P skal førebu elevane på eit samfunn og arbeidsliv i utvikling gjennom praktisk bruk av matematikk»* (Utdanningsdirektoratet, 2020a, side 2).

Det som står sentralt i matematikk P er altså det å gi elevane som følger faget verktøy i matematikk som kan bli relevante for praktiske bruk vidare i livet deira.

2.4.1 Kjerneelementa i matematikk 1P

Noko som er nytt når det kjem til dei nye læreplanane er korleis dei beskriv det som er mest relevant i faget. Dette blir gjort gjennom såkalla *kjerneelement* som summerer opp dei viktigaste fokusområda i faget. I matematikk 1P er det lista 6 slike kjerneelement, og eg ønsker i denne delen av teksten å ta ein nærare kikk på desse. Eg ønsker også å knyte desse kjerneelementa opp mot Mogens-Niss sin oversikt over kompetansar innanfor matematikk. Desse kompetansane er samanfatta i denne figuren (Niss og Højgaard, 2019, side 19):



Figur 2.1 - Ein visuell representasjon av dei 8 kompetansane innanfor matematikk

Som ein ser er dei 8 kompetansane organiserte som kronblad på ein blom. Denne framstillinga er valt for å illustrere at desse kompetansane ikkje står åleine, men glir over i kvarandre. Alle kompetansane er i denne framsyninga overlappande med resten, noko som gir eit bilete på at ein til stadigheit i matematikklasserommet arbeider med fleire enn ein kompetanse. Figuren er organisert på ein slik måte at dei kompetansane som har mest med kvarandre å gjere har kronblad som i større grad overlappar med kvarandre.

Kjerneelementa i matematikk 1P:

- Utforsking og problemløysing

Dette kjerneelementet er det som vil ha størst fokus i denne oppgåva. Kjerneelementet beskriv korleis elevane skal arbeide problemløysande i matematikkfaget. I kjerneelementet står det at elevane skal utforske ved å leite etter mønster og samanhengar, og utvikle metodar som er nyttige for å løyse oppgåver der framgangsmåten ikkje er gitt. I dette kjerneelementet er det framgangsmåten som står sterkt i fokus, og den algoritmiske tenkinga er her nemnd som ein metode for å utvikle fruktbare framgangsmåtar for å løyse problem. Meir om den algoritmiske tenkinga i kapittel 2.5.

For dette kjerneelementet er dei mest relevante kompetansane frå Mogens-Niss sin oversikt problemløysingskompetansen og den matematiske tenkinga. Avhengig av korleis ein lærar legg opp til arbeid med problemløysing vil også dei andre kompetansane i varierande grad gjere seg gjeldande. Det er også verdt å nemne at National Council of Teachers of Mathematics (NCTM) lister «bruk av oppgåver/aktivitetar som legg til rette for utforsking, resonnering og problemløysing» som eit av dei 8 kjenneteikna på god matematikkundervisning (National Council of Teachers of Mathematics, 2014, side 10). Meir om problemløysing kjem i delkapittel 2.5.1.

- Modellering og anvendingar

Kjerneelementet tek føre seg modellar, og korleis dei kan brukast i matematikkfaget og i dagleglivet elles. Modelleringa knyt seg ikkje berre til det å kunne bruke modellar, men også til å kunne vurdere korleis ein modell skal nyttast, og å vurdere når ein modell er gyldig eller ikkje. Modellering knyt seg til dagleglivet, noko som kan verke motiverande for elevane.

Den viktigaste matematiske kompetansen er her, naturleg nok, modelleringskompetansen. I arbeidet med å vurdere når ein modell er gyldig og ikkje gjer også resoneringskompetansen og den matematiske tenkinga seg gjeldande. I framstillinga av matematiske modellar er det også naturleg å tenke at hjelpemiddel- og verktøyskompetansen er sentral.

- Resonnering og argumentasjon

Kjerneelementet knyt seg til evna å kunne forstå og utforme eigne matematiske resonnement. Elevane skal vere i stand til å forstå og vurdere matematiske tankerekker, og gyldigheita til desse.

I dette kjerneelementet er det resonneringskompetansen som står i sentrum. Det å kunne forstå, vurdere og utforme matematiske argument stiller også krav til eleven si evne til å forstå desse argumenta, så symbol- og formalismekompetansen er også viktig her.

- Representasjon og kommunikasjon

Fokuset ligg her på å kunne forklare og grunngje representasjonar, samt å kunne kommunisere matematikk i samtalar, argumentasjon og resonnement. Dette kjerneelementet stiller også krav til læraren, i det at han/ho må legge opp til samtalar i klasserommet som kan bidra til å styrke elevane si evne til å kommunisere matematikk på ein tilfredsstillande måte.

Dei to kompetansane som er viktigast her gir seg sjølv; representasjonskompetansen og kommunikasjonskompetansen. Mellom desse kronblada i oversikten til Mogens-Niss ligg symbol- og formalismekompetansen, så denne står også sentralt i det å kunne arbeide med matematiske representasjonar og kommunikasjon.

- Abstraksjon og generalisering

Abstraksjon knyt seg i matematikk 1P til å bruke symbola og dei formelle resonnementa, mens generalisering handlar om å oppdage samanhengar og strukturar i faget, og å kunne bruke desse til å løysa oppgåver. Bruk av algebra er omfatta av dette kjerneelementet.

Dette kjerneelementet nemner igjen resonnering, så resonneringskompetansen er her viktig. I og med at algebra er involvert vil symbol- og formalismekompetansen også komme til spille, samt den matematiske tenkinga rundt å bruke algebraen på ein fornuftig og korrekt måte.

- Matematiske kunnskapsområde

Dette kjerneelementet er veldig overordna, og stiller krav om å kunne trekkje og utforske samanhengar mellom dei ulike kunnskapsområda i faget. Det er her også viktig å kunne trekkje kunnskapsområda ut frå sjølve matematikkfaget, og vere i stand til å knyte dei opp mot kvardagen.

Sidan dette kjerneelementet er såpass overordna vil alle kompetansane til Mogens-Niss gjere seg gjeldande her.

2.5 Algoritmisk tenking

Dette masterprosjektet baserer seg på undervisninga av programmering som elevane hadde i forbindelse med den algoritmiske tenkinga. I dette delkapittelet vil omgrepet «algoritmisk tenking» bli studert nøyare, med den hensikta å belyse denne delen av matematikken betre.

2.5.1 Om algoritmisk tenking

Utdanningsdirektoratet beskriv algoritmisk tenking på følgande måte:

> *«Algoritmisk tenkning innebærer å tilnærme seg problemer på en systematisk måte, både når vi formulerer hva et er vi ønsker å løse og når vi foreslår mulige løsninger. Litt forenklet kan vi si at det er 'å tenke som en informatiker' når vi skal løse problemer eller oppgaver»* (Utdanningsdirektoratet, 2019, side 1).

Algoritmisk tenking er altså ein måte å tenke på og nærme seg problem på. Ein informatikkar er ein person som arbeider med informatikk, som av store norske leksikon (SNL) blir definert som:

«(…) vitenskapen om strukturen, driften og anvendelsen av datamaskiner og datamaskinsystemer» (Rossen, 2018)

Her ser ein altså koplinga mellom algoritmisk tenking og digital kompetanse, men også at digitale hjelpemiddel ikkje er eit nødvendig hjelpemiddel for å tenke algoritmisk. Informatikk er her nytta som ein måte å visualisere korleis ein kan sjå for seg algoritmisk tenking. I følgande sitat set informatikkprofessoren Jeannette Wing søkelys på at det å tenke som ein informatikkar (eng: *computer scientist*) handlar om meir enn å kunne programmere:

> *«Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction.»* (Wing, 2006, side 34)

Linken mellom algoritmisk tenking og digital kompetanse kan verte endå tydlegare om ein tek for seg den engelske ekvivalenten av uttrykket «algoritmisk tenking»; «computational thinking» (heretter referert ved CT). Dette uttrykket har utgangspunkt i det engelske ordet «computational». Dette ordet kan tenkast å ta utgangspunkt i det engelske ordet for datamaskin; «computer». Når ein bruker det engelske uttrykket kjem det heilt tydeleg fram at dette emnet kan koplast til digital kompetanse. Eit forslag til definisjon på CT er gitt her:

> *«(…) an individual's ability to recognize aspects of real-world problems which are appropriate for computational formulation and to evaluate and develop algorithmic solutions to those problems so that the solutions could be operationalized with a computer»* (Fraillon et al., 2018, side 27)

Her vert CT definert i samanheng med bruk av datamaskiner til å løyse problem, og å utvikle algoritmar som ei datamaskin kunne ha operert med. Likevel ser ein av definisjonen at CT kan nyttast uavhengig av datamaskiner, fordi det er knytt til ferdigheitene til personar. Datamaskiner er her dratt inn som del av definisjonen for å setje lys på dei typane problem CT kan bidra til å løyse, og kva løysingar som problem løyst ved hjelp av CT skal kunne gi.

CT kan likevel også tenkast å ta utgangspunkt i det engelske uttrykket for å rekne; «to compute». Om ein vel å tolke uttrykket med utgangspunkt i ordet «compute» vert fokuset på datamaskiner fjerna. International Working Group on Computational Thinking vektlegg dette aspektet i følgande sitat:

> *« (…) computational thinking "shares elements with various other types of thinking such as algorithmic thinking, engineering thinking and mathematical thinking"»* (Barr og Stephenson, 2011, side 50)

Frå dette sitatet ser ein også at det norske uttrykket «algoritmisk tenking» (eng: *algorithmic thinking*) vert inkludert som ein del av uttrykket CT. Dette viser enten at CT er meir omfattande enn den norske ekvivalenten, eller at den norske ekvivalenten har fått namnet algoritmisk tenking i mangel på ei god norsk omsetjing på det engelsk uttrykket. I denne oppgåva tek eg for meg algoritmisk tenking i ein undervisningssamanheng, og det vil då vere naturleg å nytte utdanningsdirektoratet si tolking av omgrepet. Utdanningsdirektoratet presiserer at omgrepet «algoritmisk tenking» i skulen skal tolkast som ei direkte omsetjing av det engelsk uttrykket «computational thinking» (Utdanningsdirektoratet, 2019, side 1), og algoritmisk tenking i den norske skulen vil som ein konsekvens av dette omfatte alt det som det engelske omgrepet omfattar.

Algoritmisk tenking er utleia frå ordet algoritme. Matematikkdidaktikaren Ragnar Solvang definerer algoritme på følgande måte:

> *«En algoritme er en framgangsmåte, et sett med instruksjoner som ved et endelig antall operasjoner fører til at en gitt oppgåve kan løses»* (Solvang, 1986, side 136)

Om ein ser algoritmisk tenking i samanheng med denne definisjonen ser ein at det å tenke algoritmisk kan tolkast som å finne ut kva for tankemåtar ein kan nytte for å produsere ein algoritme for å løyse eit problem. Dette kjem også tydeleg fram i definisjonen av CT frå Fraillon et al. (2018); *«(…) develop algorithmic solutions to those problems (…)»*. Algoritmisk tenking er altså ein måte menneske løyser problem på; og det handlar ikkje om å få menneske til å tenke som datamaskiner (Eickelmann, 2019, side 57).

2.5.2 Problemløysing og algoritmisk tenking

Kjerneelementet som tek for seg den algoritmiske tenkinga i matematikk 1P er namnsett til «Utforsking og problemløysing». I dette kjerneelementet er problemløysing beskrive slik:

> *«Problemløysing i matematikk P handlar om at elevane utviklar ein metode for å løyse eit problem dei ikkje kjenner frå før»* (Utdanningsdirektoratet, 2013, side 2)

Denne definisjonen inneheld ordet «problem». Før vi går vidare skal vi sjå litt på dette omgrepet. Solvang definerer eit problem på følgande måte:

> *En utfordring vil for en person være et problem dersom denne personen ikke har noen algoritme som vil gi en løsning når personen konfronteres med utfordringen»* (Solvang, 1986, side 137)

Her har Solvang definert eit generelt problem. Alan Schoenfeld definerer eit *matematisk* problem ved bruk av to krav, på følgande måte:

> *«For any student, a mathematical problem is a task (a) in which the student is interested and engaged and for which he wishes to obtain a resolution, and (b) for which the student does not have a readily accessible mathematical means by which to achieve that resolution»* (Schoenfeld, 1989b, side 87-88)

Definisjonane til Solvang og Schoenfeld skil seg ved krav (a) i Schoenfeld sin definisjon. Om ein tek utgangspunkt i desse to definisjonane av problem og matematiske problem så er forskjellen på dei to at eit matematisk problem omhandlar matematikk, og at det også engasjerer og interesserer eleven som arbeider med det. Det er verdt å observere at Schhoenfeld har definert matematisk problem for ein elev eller student, mens Solvang har definert eit problem for ein kva som helst person.

Eit omgrep som knyt seg sterkt til matematisk problemløysing er *undersøkingslandskap*. Eit undersøkingslandskap er ein stad ein seier elevar oppheld seg om dei er i ein situasjon der dei vert invitert til, og ikkje kan la vere, å stille spørsmål som «Kva no viss…?» og «Kvifor det?» når dei arbeider med matematiske oppgåver (Skovsmose, 1998, side 28). Eit undersøkingslandskap kan frå denne beskrivinga bli sett på som ein invitasjon til elevane til å gjennomføre utforsking i møte med oppgåver der dei ikkje kjenner framgangsmåten for å oppdrive eit svar, noko som ligg tett opptil det som utdanningsdirektoratet har beskrive som problemløysing i matematikk 1P. Eit sentralt poeng ved slike undersøkingslandskap er at det må verke freistande for elevar å utforske (Skovsmose, 1998, side 28). Frå denne poengteringa kan det sjåast ei kopling mellom undersøkingslandskap og Schoenfeld sin definisjon av eit matematisk problem, der krav (a) krev at eit matematisk problem verkar interessant og engasjerande for eleven.

Prosessen der elevane bestemmer seg for kva metodar dei ønsker å nytte for å løyse eit problem er sentral i aktiviteten problemløysing. Det er i denne samanhengen at den algoritmiske tenkinga kan komme i spel. Aspektet der algoritmisk tenking vert dratt inn som eit verktøy i problemløysingsarbeidet i skulen vil bli utforska nøyare i delkapittel 2.5.4.

NCTM har offentleggjort følgande utsegn knytt til problemløysingsoppgåver i matematikk:

> *«Effective teaching of mathematics engages students in solving and discussing tasks that promote mathematical reasoning and problem solving and allow multiple entry points and varied solutions»* (National Council of Teachers of Mathematics, 2014, side 17)

NCTM sett altså oppgåver som dreg inn problemløysing som eit krav for at matematikkundervisninga skal vere effektiv. Spørsmåla som vil vere naturlege å stille seg i denne samanhengen er kva som gjer problem gode? Den amerikanske matematikkdidaktikaren Magdalene Lampert er klar i si meining om at det viktigaste kriteriet for eit godt problem er at det engasjerer elevane i prosessen der dei sett opp og testar matematiske hypotesar for å løyse problemet (Breiteig, 2008, side 36). Kravet som Alan Schoenfeld (1989b) altså stiller til eit matematisk problem ser Lampert på som ein fordel ved eit generelt problem.

Utdanningsdirektoratet legg vekt på at eit viktig element av problemløysinga ligg i om elevane klarer å vurdere om løysingane dei får er gyldige eller ikkje (Utdanningsdirektoratet, 2013, side 2). Om elevane som arbeider med ei matematisk oppgåve vert engasjert i arbeidet kan også etterarbeidet med problemet vere fagleg verdifullt. I lys av arbeidet med eit matematisk problem kan samanlikningar av svar engasjere til ein fagleg diskusjon blant deltakarane (Breiteig, 2008, side 36).

Definisjonen til Ragnar Solvang av eit problem inneheld ordet «algoritme», som vi definerte i delkapittel 2.5.1. I møte med eit problem er det problemløysingsstrategiar som må nyttast for å løyse det, og Solvang definerer problemløysing på følgande måte:

> *«Problemløsing er å søke etter de handlinger som en må foreta for å løse et problem»* (Solvang, 1986, side 137)

Om ein ser dei to definisjonane til Solvang i samanheng med kvarandre kan ein sjå at når ein arbeider med problemløysing så er ein på søken etter ein algoritme, som i utgangspunktet ikkje er gitt, som kan løyse problemet på ein tilfredsstillande måte. Det er dette den algoritmiske tenkinga i grunn handlar om, og ein ser dermed at algoritmisk tenking og problemløysing i aller største grad kan knytast saman. Koplinga mellom problemløysing og den engelskspråklege ekvivalenten til algoritmisk tenking, CT, openberrar seg også når ein ser på definisjonane som er gitt av CT. Til dømes er problem nemnd i definisjonen til Fraillon et al. (2018), som vart gitt i delkapittel 2.5.1. Koplinga kjem også til syne i følgande beskriving av CT:

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

Om ein skal implementere problemløysing innanfor fagfeltet matematikk finst det ein del ekstra haldepunkt ein kan ta omsyn til. Om ein skal lære seg matematikk i løpet av problemløysingsprosessen er det viktig at ein legg opp til arbeid med problem som inviterer til konseptualisering, og at ulike dilemma som oppstår i arbeidet er viktige, og må representerast, utforskast, analyserast og forklarast ved bruk av matematiske ressursar (Santos-Trigo, 2019, side 67). Det er også viktig for fagleg framgang i matematikk at oppgåvene er av ein slik karakter at den som arbeider med dei støyter på utfordringar som gjer at dei må ta i bruk sine eigne krefter og problemløysingsevner for å løyse oppgåvene, jamfør Piaget sin teori kring møte med kognitive konfliktar. For å utvikle ein person si matematiske tenking i møte med problemløysingsoppgåver må oppgåvene legge opp til spørsmål, utfordringar og refleksjon (Santos-Trigo, 2019, side 67).

2.5.3 Programmering som ein del av algoritmisk tenking

Programmering er ein prosess som relaterer seg til å utvikle og implementere instruksjonar for dataprogram for å få maskina til å utføre spesifikke handlingar (Forsström og Kaufmann, 2018, side 19).

Programmering blir utført ved bruk av ulike programmeringsspråk. Ulike programmeringsspråk har ulike verkemåtar, noko som gjer at ein ikkje nødvendigvis er i stand til å programmere i fleire språk om ein har kjennskap til eit. Dette gjer at fagfeltet programmering stiller krav til kunnskap kring utvikling av spesialiserte algoritmar og logikk. Logikk i programmering er definert som lista med instruksjonar som fortel eit program korleis det skal operere (2018a), mens syntaks er eit sett med reglar for grammatikk og rettskriving som må følgast for å få eit program til å operere på ein ønska måte (2018b). Effektiv bruk av programmering stiller også krav til evna ein har til å analysere, forstå og løyse problem ved å verifisere krava ein sett til algoritmane ein bruker, og å kunne avgjere om algoritmane ein implementerer i spesifikke kodeprogram er korrekte handlingar (Forsström og Kaufmann, 2018, side 19).

Python er eit programmeringsspråk som har utvikla seg til å bli eit språk som ofte blir brukt til å introdusere programmering til nybegynnarar. Det er mange grunnar til at Python er eit godt programmeringsspråk å nytte i undervisningssamanheng. For det fyrste er syntaksen som vert nytta i Python svært «rein» (eng: *clean*), i den forstand at koden som må skrivast for å få program til å fungere er minimal og konsis. I tillegg til dette fokuserer Python på lesbar (eng: *readable*) kode (John Paul, 2014, side 1), noko som kan bidra til å gjere skrivinga og tolkinga av koden enklare. Dette gjer at studentar kan lære språkets verkemåte kjapt, og det gjer også undervisaren i stand til å implementere konsise programmeringshint i kodefragmenta (Myers og Sethna, 2007, side 75). Python er også eit kraftig og velfungerande programmeringsspråk. Dei innbygde datatypane gjer at elevar fort kan setje i gang med Python-programmering, utan å ta omsyn til å deklarere dei ulike datatypane, som i fleire andre programmeringsspråk er nødvendig for å få eit program til å fungere ordentleg (Myers og Sethna, 2007, side 75). Alle desse faktorane gjer at Python er eit ideelt programmeringsspråk for nybegynnande programmerarar å arbeide med.

Programmeringa knyt seg til den algoritmiske tenkinga på den måten at kodeprogram ofte vert nytta for å produsere løysingar på spesifikke problem. Ein av styrkane til programmering er at det veldig ofte finst fleire mogelege måtar å løyse same problem på i eit kodeprogram, noko som gjer at prosessen der ein vel framgangsmåte er svært aktuell. Som beskrive ovanfor er også måten ein sett opp algoritmar, og vurderingar av løysingar og løysingsmetode i ettertid eit sentralt konsept når det kjem til programmering, akkurat som dette også er svært sentrale konsept når det kjem til den algoritmiske tenkinga.

Ei anna styrke som gjer at programmering er nyttig innanfor fleire fagfelt er at ein ved hjelp av eit visst tal kodelinjer kan utføre mange operasjonar etter kvarandre. To eksempel på datastrukturar innanfor programmering som har som verkemåte å gjennomføre gjentatte operasjonar etter kvarandre er for- og while-løkker. Ved bruk av desse strukturane kan ein i kodeprogram utføre handlingar, som til dømes utrekningar eller tekstanalyse, som utan kodeprogram ville tatt lenger tid og kravd meir arbeid, samt hatt større sjanse for feil. Dette momentet ved programmering gjer at programmerarar som prøver å lage eit program som oppfyller visse krav på ei stor mengd datamateriale enkelt kan prøve seg fram utan å vere redd for at eit svært omfattande arbeid på materialet vil vere forgjeves om verkemåtane ikkje skulle fungere som ønska. I eit kodeprogram er det ofte snakk om å endre på nokon få verkemåtar i koden for å få eit ønskeleg resultat.

Ein av tinga som skil programmering frå mykje anna arbeid er at ein ved bruk av programmering har ein slags "samtalepartnar", i form av eit kodeprogram. Kodeprogrammet nytta gir nyttige og konsise tilbakemeldingar til programmeraren når han/ho arbeider med programmet. Tilbakemeldinga som vert gitt når programmeraren har gjort alt riktig er eit program som fungerer som det skal, mens om noko er feil i programmet så får programmeraren tilbakemelding om dette i form av eit program som ikkje fungerer som ønska, eller feilmeldingar. Slike direkte tilbakemeldingar får ein ikkje om ein arbeider med penn og papir. Om ein innanfor matematikken arbeider med penn og papir, og oppdagar på slutten av ei oppgåve at eit svar er feil får ein ingen tilbakemelding om kvar feilen er. Dette kan gjere at heile oppgåvearbeidet vert sett på som bortkasta arbeid. Innanfor programmeringa får ein i motsetnad ein tilbakemelding om logiske og syntaktiske feil, noko som gjer at feil (eller «bugs» som det er kalla i program) ikkje vert sett på som forgjeves arbeid, men nødvendige steg for å kunne produsere eit betre resultat og framtidig framgang (Sfard og Leron, 1996, side 192).

Det finst mange fordelar som programmering har ovanfor andre arbeidsmetodar. Eg har gjennomgått fleire ovanfor, og ønsker å gjennomgå ein viktig fordel til i denne delen av teksten. Programmering som verktøy kan konkretisere enkelte ting på ein måte som andre verktøy ikkje kan. Ved å skrive kode som omhandlar abstrakte konsept får ein programmerar framfor seg dei spesifikke verkemåtane til slike konsept, som ein kanskje ikkje ville fått på ein like klar måte om ein arbeida på andre måtar. Dette er kanskje spesielt nyttig innafor matematikken, der oppbygginga og verkemåten til fleire konsept ofte er så abstrakte at dei er vanskelege for elevar å forstå. Programmering gjer elevane i stand til å konstruere desse konsepta, i staden for å oppdage dei frå tekstbøker eller forelesningar. Denne konstruksjonen i kodeprogram gjer at elevane også konstruerer konsepta mentalt, noko som kan verke fordelaktig for arbeid med konseptet vidare (Leron og Dubinsky, 1995, side 230).

2.5.4 Algoritmisk tenking i skulen

I boka *Computational Thinking Education*, som tek føre seg korleis algoritmisk tenking kan implementerast i klasserommet, vert det satt fokus på at det er viktig å skilje mellom å bruke datamaskina som eit verktøy i klasserommet, og det å tenke algoritmisk (eng: *computationally*) (Kong og Abelson, 2019, side 4). Boka lister opp dei tre vanlegaste framgangsmåtane som vert brukt for å implementere det som læraren oppfattar som algoritmisk tenking i klasserommet

(National Research Council (U.S.). Committee for the Workshops on Computational Thinking., 2010, side vii):

1. Fokus på digitale lese- og skriveferdigheiter. Bruk av digitale hjelpemiddel til å lage nyheitsbrev, dokument, nettsider, multimedia-presentasjonar eller budsjett.
2. Fokus på konstruksjon av kode gjennom programmering. Bruk av programmeringsverktøy til å skrive kode.
3. Fokus på bruk av programmering i til dømes spel, robotar og programmering.

Det er nok naturleg for dei fleste lærarar, når dei høyrer uttrykket «algoritmisk tenking» for fyrste gong, å knytte dette til ordet "algoritme", som igjen enkelt kan bli knytt til datamaskiner. Som diskutert tidlegare er algoritmisk tenking eit meir omfattande tema enn å vite korleis ein kan bruke digitale hjelpemiddel, det er ein måte å arbeide på. Dette er viktig for lærarar å vite, og formidle til elevane, slik at den algoritmiske tenkinga blir med i matematikkundervisninga også etter at PC-skjermane er lagt ned og penn og papir kjem fram. Sidan det å tenke algoritmisk er ein måte å løyse problem på kan denne måten å arbeide på gjere seg gjeldande i så og sei alle område innanfor matematikken, og det vil vere læraren si oppgåve å sørge for at elevane ikkje forbinder denne arbeidsmetoden eksklusivt til bruk på datamaskiner.

Som alltid er det viktig for lærarar å motivere elevane til å arbeide med pensumet som vert arbeida med i timane, og når eit nytt tema skal arbeidast med kan det lønne seg å tenke igjennom korleis ein kan motivere elevane til å arbeide med det temaet spesifikt. Det kan i all hovudsak identifiserast tre ulike måtar dette kan gjerast på når det kjem til algoritmisk tenking (Eickelmann, 2019, side 54):

1. Fleirfagleg framgangsmåte:
   Fyrste framgangsmåte går ut på å gjere det forstått at algoritmisk tenking ikkje er eit emne som er avgrensa til matematikkfaget. Det er ein tenkemåte som, når forstått, kan nyttast i fleire ulike fag.
2. Informatikkframgangsmåte:
   Denne framgangsmåten tek basis i å formidle kor essensielt algoritmisk tenking er innanfor fagfeltet informatikk. Informatikk kan så nyttast som ein inngangsport til å diskutere kor viktig dette fagfeltet er for våre daglege liv.
3. Framgangsmåte som fremmer algoritmisk tenking som eit individuelt tema:

> Ved å ta i bruk denne framgangsmåten legg ein vekt på å framheve at den algoritmiske tenkinga er ein nøkkelkompetanse når det kjem til å kunne nytte ny teknologi på ein kompetent og reflektert måte.

Som ein ser av framgangsmåtane gitt over så er det ikkje til å komme bort ifrå at den algoritmiske tenkinga er knytt til bruk av digitale hjelpemiddel, men som sagt er det viktig å få fram poenget om at det ikkje knyt seg eksklusivt til bruk av teknologi. Dette er grunnen til at eg personleg held ein knapp på den fyrste framgangsmåten, som tek utgangspunkt i den algoritmiske tenkinga som problemløysingsmetode utanfor rammeverket med digitale hjelpemiddel.

Etter kvart som programmering har blitt anerkjent som ein ferdigheit som spesielt innanfor teknologi og realfag er viktig å kunne, har det vorte eit aukande press på å få inn programmering som eit fag i den norske skulen (Forsström og Kaufmann, 2018, side 19). I tillegg til at programmering er eit nyttig fagfelt i dagens samfunn gir kode også eit svært oversiktleg og verkeleg bilete på korleis ein algoritme kan sjå ut, samtidig som konstruksjonen av kode kan hjelpe elevane å bygge kunnskap rundt det å tenke algoritmisk.

3 METODE

3.1 Metodisk tilnærming

Føremålet med dette masterprosjektet var å kartlegge erfaringa med programmering, forståinga av programmering og synet på programmering til elevane i ein 1P-klasse i det programmering vart introdusert som eit verktøy i matematikkundervisninga. For å kartlegge dette vart det tatt i bruk fleire forskingsmetodar.

Elevane som meldte seg på prosjektet (sjå delkapittel 3.6 for meir om dette utvalet) svarte på to spørjeundersøkingar knytt til undervisninga i matematikk; ei undersøking før dei vart introdusert for programmering og ei undersøking etter at dei hadde gjennomført undervisningsopplegget som inkluderte programmering. I desse spørjeundersøkingane vart både kvantitative og kvalitative data samla inn.

Mens programmeringsundervisninga gjekk føre seg vart observasjon tatt i bruk som metode. Dette for å kunne dokumentere kva programmeringsundervisninga innehaldt, samt elevane sine reaksjonar i møte med denne undervisninga.

I etterkant av gjennomføringa av spørjeundersøkingane og observasjonen vart eit utval av elevane som hadde gjennomført begge spørjeundersøkingane valt ut til å delta i eit forskingsintervju (meir om dette utvalet i delkapittel 3.6). Føremålet med intervjua var å grundigare belyse enkelte av spørsmåla som var stilt i spørjeundersøkingane, samt å samle inn meir datamateriale knytt til problemstillinga, som ikkje vart samla i spørjeundersøkingane. Datamaterialet frå forskingsintervjua er derfor svært sentral i denne studien.

3.2 Kvalitativ og kvantitativ metode

I undervisningsforsking er det vanleg å skilje mellom *kvalitative* og *kvantitative* metodar. Val av forskingsmetode vil ofte ta utgangspunkt i problemstillinga og/eller forskingsspørsmålet ein prøver å svare på ved å bruke metodane, sjølv om det nokon gonger vil vere naturleg å bruke

begge metodane for å svare på ei problemstilling (Danielsen, 2013, side 138). I denne oppgåva ønsker eg i all hovudsak å samle inn kvalitative data, men også nokon kvantitative. Det er derfor naturleg å nytte både kvantitative og kvalitative forskingsmetodar i dette arbeidet.

Kvantitativ forskingsmetode samlar inn observasjonar i form av tal (Danielsen, 2013, side 137). Denne forskingsmetoden kan nyttast til å teste teoretiske hypotesar som har utgangspunkt i teori (Danielsen, 2013, side 138). Tala som er resultatet av gjennomføringa av ei kvantitativ undersøking kan analyserast, og nyttast til å støtte opp under eller svekke teorien som undersøkinga har tatt utgangspunkt i. Dersom resultata frå fleire undersøkingar peikar i same retning vil det føre til mindre usikkerheit rundt funna, og hypotesen vil bli ytterlegare styrka (Danielsen, 2013, side 138). Eit døme på ein kvantitativ forskingsmetode er ei spørjeundersøking med fastsette svaralternativ (Dalland, 2017, side 53). Dei fastsette alternativa gjer det mogeleg for forskaren å oppdrive kvantitative data i form av til dømes svarprosentar og middelverdiar. Dette vert gjort ved bruk av statistiske metodar.

Den kvalitative forskingsmetoden skil seg frå den kvantitative ved at resultata frå slike metodar er opplevingar og meiningar som ikkje direkte lar seg representere av tal (Dalland, 2017, side 52). Eit eksempel på ein kvalitativ metode er intervju utan faste svaralternativ (Dalland, 2017, side 53). I ei slik setting vil intervjuobjektet ha mogelegheita til å greie ut om erfaringar, meiningar eller liknande, og det er desse svara som vert rekna som det kvalitative datamaterialet.

Kvantitative metodar vert beskrive som å gå i breidda, det vil seie at metodane hentar inn eit lite tal opplysningar om mange einingar, mens kvalitative metodar går i djupna ved å samle inn mange opplysningar om få einingar (Dalland, 2017, side 53). Når både kvantitative og kvalitative metodar vert nytta kallar ein det for *mixed methods*, og det er mixed methods som vart brukt i dette prosjektet (Danielsen, 2013, side 139). Mixed methods har blitt trekt fram som eit nyttig verktøy i forsking. Mixed methods innehar ein fleksibilitet som reflekterer den endrande og integrerte delen av verda, og fenomena som vert studert (Cohen et al., 2011, side 26).

Dei kvalitative dataa som er nytta i dette prosjektet er delane av spørjeundersøkingane der svara elevane oppgjer er tal eller fastsette svaralternativ. Dei kvalitative dataa nytta er observasjonsloggen, transkripsjonane frå forskingsintervjua og delane av spørjeundersøkingane der svaret elevane gir er fritekst.

3.3 Spørjeundersøkingar

Spørjeundersøkingar er eit viktig forskingsverktøy som har som hovudmål å måle noko (Oppenheim, 1992, side 100). I dette prosjektet samla spørjeundersøkingane inn datamateriale som kan hjelpe til å belyse problemstillinga, samtidig som dei gav eit grunnlag for å velje ut elevar til intervju. Det var også eit poeng at svara frå spørjeundersøkingane danna eit grunnlag som kunne undersøkast nærmare i intervjua.

På førehand vart det gjennomført følgande tiltak, som har vist seg å bidra til å auke svarprosenten i spørjeundersøkingar (Oppenheim, 1992, side 104-105):

- Informasjonsskriv om prosjektet vart delt ut
- Ei forklaring på korleis og kvifor klassen er valt ut vart gitt
- Informasjon om prosjektleiar og prosjektinstitusjon vart gitt
- Lovnad om absolutt konfidensialitet og anonymitet vart gitt

3.3.1 Spørjeundersøking 1

I prosjektet vart det nytta to spørjeundersøkingar; ei undersøking før undervisningsøkta med programmering og ei anna etter. Spørsmåla stilt i den fyrste spørjeundersøkinga (vedlegg 3) hadde som mål å kartlegge følgande punkt:

- Tidlegare programmeringserfaring
  Kartlagt ved å kryssa «Ja» eller «Nei» for tidlegare programmeringserfaring, og eventuelt på kva programmeringsspråk dei har nytta tidlegare.

- Tankar kring tidsbruk innanfor matematikkfaget
  Dei to spørsmåla kring tidsbruk er dei same spørsmåla som Alan Schoenfeld nytta i sin studie (1989a). Elevane skriv inn ei tidsmengd dei føler ei typisk matematikkoppgåve skal ta å løyse, og ei tidsmengd dei føler er rimeleg å bruke på ei vanskeleg oppgåve. Svara på desse kan gi ein indikasjon på elevane sine matematikksyn.

- Kunnskapar og meiningar knytt til algoritmisk tenking
  Elevane vel det svaralternativet dei meiner best beskriv ordet «algoritme», og skriv inn det dei trur ligg i omgrepet «algoritmisk tenking». Dette vil kunne gi ein peikepinn på kva kunnskapar elevane sitt på om temaet før undervisninga og/eller kva dei tenker det handlar om.

- Matematikkfagets relevans for eleven
  Rangeringsspørsmål på ein skala frå 1 til 6. Spørsmåla kan koplast til elevane si forståing og syn på matematikkfaget.

- Tidlegare bruk og meiningar knytt til bruk av digitale hjelpemiddel i matematikkfaget
  Rangeringsspørsmål på ein skala frå 1 til 6. Vil kartlegge elevane si tidlegare erfaring med bruk av teknologi i matematikkfaget, og synspunkt rundt dette.

- Synspunkt kring nyttigheita til programmering i matematikkfaget
  Elevane skriv om dei trur bruk av programmering i matematikkundervisninga vil vere nyttig, og kvifor/kvifor ikkje. Dette er interessant for å sjå kva dei tenker om programmering i matematikkfaget i utgangspunktet, og om synet deira endrar seg etter at programmeringsundervisninga er vorte gjennomført.

- Plassering av algoritmisk tenking innunder eit eller fleire av kjerneelementa i matematikk 1P
  Dette spørsmålet vart gitt til elevane, men datamaterialet frå spørsmålet har ikkje vorte nytta i oppgåva, og vil derfor ikkje bli diskutert nærare her. Sjå delkapittel 5.4.1 for meir informasjon rundt dette.

3.3.2 Spørjeundersøking 2

På spørjeundersøking 2 (vedlegg 4) vert følgande punkt forsøkt kartlagt:

- Synspunkt rundt undervisninga i programmering
  To spørsmål med fritt svar; eit som spør om undervisninga gjekk føre seg som venta og eit som spør om dei syns undervisninga kunne blitt lagt opp på ein annan måte. Desse

spørsmåla kan bidra til å danne seg eit bilete av elevane si forståing og syn på programmering i matematikkfaget.

- Tankar kring tidsbruk innanfor programmering
  Identiske spørsmål kring tidsbruk frå spørjeundersøking 1, men no knytt til programmeringsoppgåver. Kan danne eit grunnlaget for ei samanlikning mellom elevane sine tankar kring tidsbruk på reine matematikkoppgåver og programmeringsoppgåver.

- Kunnskapar og meiningar knytt til algoritmisk tenking
  Dei to same spørsmåla som frå spørjeundersøking 1. Her vil ein sjå om elevane sine tankar kring algoritmisk tenking har endra seg i løpet av undervisningsperioden.

- Vurdering av programmering som verktøy
  Rangeringsspørsmål frå 1-6 som spør om elevane sine synspunkt kring programmeringa. Nytteverdi, frustrasjon i arbeidet, graden av læring og fornøyelse vert forsøkt kartlagt. Alle desse synspunkta kan vere med å danne eit bilete av elevane si forståing og elevane sitt syn på programmering.

- Andre erfaringar
  Eit ope spørsmål der elevane kan skrive inn andre erfaringar knytt til algoritmisk tenking.

Som beskrive tidlegare er målet med desse spørjeundersøkingane å danne eit grunnlag for å vidare kunne belyse problemstillinga i prosjektet. Det er gjort ei vurdering om å ikkje spørje om for mykje på spørjeundersøkingane, ettersom forsking tyder på at kvaliteten på svara gitt i ei spørjeundersøking kan gå ned om undersøkinga er for lang (Herzog og Bachman, 1981, side 558). Svara på spørjeundersøkingane har ein verdifull funksjon då dei dannar eit datamateriale som seier noko generelt om klassen, og dei vil vere eit fokus i intervjua med utvalet av elevar.

3.4 Observasjon

Eg ønska å inkludere i oppgåva beskrivingar av det pensumet som vart gjennomgått i timane som omhandlar programmering. For å kunne dokumentere dette tok eg i bruk observasjon som verktøy.

Observasjon handlar om å nytte sansane til å samle inn og danne seg inntrykk av det som skjer (Germeten og Bakke, 2013, side 110). I dette prosjektet var målet å vere mest mogeleg skjult for elevane for på den måten å kunne observere situasjonar og samtalar slik som dei ville ha gått føre seg naturleg, utan ein observatør i klasserommet. Denne typen observatør vert kalla ein *fullstendig observatør*, og ein ideell fullstendig observatør vil ha ingen grad av nærleik til dei som vert observert (Germeten og Bakke, 2013, side 114-115). Dette vart gjort ved å observere elevane frå bak i klasserommet, og ved å halde avstand til dei då eg observerte interaksjonane deira med lærar eller andre elevar. Reint praktisk var det vanskeleg å steppe inn i rolla som ein fullstendig observatør i dette klasserommet sidan eg tidlegare hadde hatt kontakt med fleire av elevane då eg introduserte prosjektet, og gav fleire av dei karakterkodar til deltaking i den fyrste spørjeundersøkinga. Det at eg hadde ein liten relasjon til fleire av elevane gjorde at nokon spurde meg om hjelp under programmeringsarbeidet då eg observerte, og då steppa eg midlertidig inn i rolla som *fullstendig deltakar*; ein observatør som inngår i hendingane til dei som vert studert (Germeten og Bakke, 2013, side 114). Under observasjonen gjorde eg meg skriftelege notat av det som vart observert.

Det finst fire aspekt ved metoden observasjon som gjer at resultatet frå metoden må betraktast som ei forenkling av verkelegheita (Germeten og Bakke, 2013, side 111). Desse aspekta knyt seg til observatøren sin hukommelse og sansane si avgrensing. Desse aspekta er (Germeten og Bakke, 2013, side 112-113):

1. Endring i proporsjonane i det sanseinntrykka og hukommelsen vert overført til papiret. Det som blir skrive ned vil aldri kunne beskrive ein situasjon så detaljert som den faktisk har utspelt seg.
2. Henda er treigare enn talen. Dette gjer at det vil vere bortimot umogeleg å nøyaktig overføre til papiret den samtalen som er blitt observert.
3. Det er umogeleg å observere alt som går føre seg i klasserommet. Sansane våre er berre i stand til å observere eit visst tal situasjonar om gongen, noko som gjer at observatøren kan gå glipp av verdifulle observasjonar i det noko anna vert observert.

4. Observatøren kan berre nøyaktig observere det han/ho har kunnskapar om. Det ein observerer ser ein i lyset av det ein har sett før, noko som gjer at enkelte observerte situasjonar kan bli noko forvridd.

3.5 Forskingsintervju

Forskingsintervjua samlar inn viktig datamateriale i denne masteroppgåva. Målet med eit forskingsintervju er å få fram betydninga av folk sine erfaringar, og å avdekke deira opplevingar av verda (Kvale et al., 2015, side 20). Intervju gjer respondentane i stand til å uttrykke seg på ein rikare og meir spontan måte enn dei kan gjere på spørjeundersøkingar (Oppenheim, 1992, side 81).

Ein viktig faktor i intervju er motivasjonen til respondentane (Oppenheim, 1992, side 82). Respondentane vert bedt om å gi av si tid, sine tankar, sin innsats og sitt privatliv utan å tilsynelatande tene noko på deltakinga. Om motivasjonen til respondentane som vert intervjua er låg kan dette påverke kvaliteten til responsane til intervjuobjekta (Oppenheim, 1992, side 82). For å auke motivasjonen for deltaking hos elevane i dette prosjektet vart det før intervjua presisert kvifor akkurat dei var plukka ut i staden for nokon andre, og det vart også presisert at deira deltaking i intervjuet bidreg til å produsere forskingslitteratur innanfor fagfeltet matematikkdidaktikk.

Før forskingsintervjua med elevane tok til vart det utarbeida ein intervjuguide som vart brukt som eit hjelpemiddel av meg som intervjuar (vedlegg 5). Spørsmåla i denne intervjuguiden vart utarbeida med mål om å kunne bruke datamaterialet frå intervjua til å belyse problemstillinga og forskingsspørsmåla i masteroppgåva.

Intervjuguiden er delt inn i 4 deler:

1. **Bakgrunnsinformasjon**: Kartlegging av eleven si tidlegare erfaring med programmering og deira motivasjon i matematikkfaget.

2. **Utforsking av svara i spørjeundersøkinga**: Ein diskusjon sentrert rundt eleven sine svar i spørjeundersøkingane.

3. **Programmeringsundervisning**: Ei utspørjing om eleven sine erfaringar og synspunkt knytt til den programmeringsundervisninga dei har delteke i.

4. **Programmering i matematikkundervisninga**: Generelle spørsmål om eleven sine tankar og meiningar om programmering som del av matematikkpensumet i skulen.

Ved presentasjon av denne intervjuguiden må det presiserast at sidan ein stor del av intervjua baserte seg på elevane sine svar i spørjeundersøkingane vart ikkje intervjuguiden følgt til punkt og prikke. Intervjuet vart derimot tilpassa elevane sine svar, som også var tilgjengeleg for elevane i løpet av intervjuet i tilfelle dei hadde gløymt delar av spørsmåla og/eller sine responsar på spørsmåla.

3.6 Utval av elevar

Då dette prosjektet tok til var det tiltenkt at opplegget skulle gjennomførast i ein matematikk 1T-klasse, og ikkje i ein matematikk 1P-klasse. Dette var fordi at fagfornyinga i matematikk 1T vart innført i skuleåret 2020-2021, og i denne læreplanen står programmering nemnt direkte (Utdanningsdirektoratet, 2020b, side 5). Når det gjeld læreplanen i matematikk 1P (Utdanningsdirektoratet, 2020a) så står ikkje programmering direkte nemnd. Fleire aktuelle lærarar vart kontakta, og av ulike grunnar var det ikkje mogeleg å få gjennomført dette prosjektet i ein 1T-klasse i løpet av haustsemesteret 2020. Ein lærar som skulle undervise i matematikk 1P dette skuleåret responderte likevel på førespurnaden. Denne læraren skulle i løpet av haustsemesteret ta i bruk programmering i 1P-undervisninga, som del av den algoritmiske tenkinga, og var open for å inkludere ein masterstudent i dette. Slik gjekk det føre seg at prosjektet vart gjennomført i ein 1P-klasse på ein vidaregåande skule på Vestlandet.

Elevane som deltok på spørjeundersøkingane var dei som på førehand hadde fylt ut samtykkeskjema på ein korrekt måte, og på den måten meldt seg på prosjektet (meir om dette i kapittel 3.10).

Det vart plukka ut 4 elevar til intervju. Desse vart plukka ut med omsyn på svara deira i spørjeundersøkingane. To av elevane som vart plukka ut hadde kryssa av for tidlegare erfaring

med å programmere, mens dei resterande to ikkje hadde kryssa av for tidlegare programmeringserfaring. Det vart også lagt vekt på utviklinga i løpet av undervisningsperioden, deira syn på nytteverdien til programmering, deira syn på nytteverdi av matematikk og deira grad av forståing i programmering. Dei 4 elevane hadde varierande svar, og utgjer saman eit relativt representativt utval av klassen.

3.7 Datainnsamling

Spørjeskjema nytta i dette prosjektet er vorte laga og distribuert ved bruk av programmet SurveyXact By Ramboll. Elektroniske spørjeskjema vart nytta av fleire grunnar. For det fyrste hindra restriksjonar knytt til smittespreiing av Covid-19 å dele ut spørjeskjemaet i papirform til elevane. Elektroniske spørjeskjema gjer dessutan datainnsamlinga og dataorganiseringa enklare, ettersom all informasjon vert henta inn og lagra digitalt. Dette gjer likevel at ein må ta ekstra omsyn til å lagre og oppbevare det sensitive datamaterialet på ein sikker og konfidensiell måte, og dette har i høgaste grad vore ein prioritet i dette prosjektet (sjå kapittel 3.10 for meir om dette).

Spørjeundersøkingane vart gjennomførte i byrjinga av to matematikktimar, etter avtale med lærar. Elevane fekk i forkant av begge undersøkingane utdelt sine karakterkodar. På den fyrste undersøkinga deltok 19 elevar, mens på den andre undersøkinga deltok 14 elevar. Færre elevar deltok på spørjeundersøking 2 grunna fråvær frå skulen.

Intervjua vart gjennomført i løpet av to matematikktimar, der ein og ein elev vart henta ut ifrå klassen. Intervjua med 4 elevar vart gjennomførte på eit grupperom i nærleiken av klasserommet. Intervjua tok mellom 18 og 24 minutt.

3.8 Validitet

Validitet handlar om i kva grad forskaren har klart å måle det som forskaren hadde intensjonar om å måle (Cohen et al., 2011, side 179). Faktorane som avgjer graden av validitet i eit forskingsprosjekt avhenger blant anna av typen forskingsmetode nytta i prosjektet. Ved bruk

av kvalitativ metode kan validiteten i prosjektet verta adressert ved å sjå på ærlegdommen, djupna og omfanget til det innsamla datamaterialet, mens det i kvantitativ forsking ofte vil vere meir aktuelt å ta føre seg nøye sampling, passande instrumentering og passande statistisk behandling av datamaterialet for å auke validiteten (Cohen et al., 2011, side 179).

I mitt prosjekt er mixed methods valt som metode, noko kan vere med på å styrke validiteten av funna (Cohen et al., 2011, side 26). Mange forskarar argumenterer blant anna for at intervju er meir valid enn spørjeundersøkingar fordi intervjuaren ser og snakkar til intervjuobjektet samtidig som det blir registrert ekte responsar, noko som minkar sjansen for misforståingar (Oppenheim, 1992, side 82). På ei anna side kan intervju vere meir partisk (eng: *bias*) og upåliteleg enn spørjeundersøkingar (Oppenheim, 1992, side 82). Når ein bruker både kvalitative og kvantitative forskingsmetodar i same prosjekt gjeld validitetskrava for kvar av dei to metodane brukt (Cohen et al., 2011, side 197).

Spørjeundersøkingane i dette prosjektet hadde som formål å måle haldningane og innstillingane til elevane. Det er vanskeleg å seie noko konkret om validiteten til denne typen spørsmål grunna mangelen på kriterium hos gruppa som skal undersøkast (Oppenheim, 1992, side 148). Elevgruppa som gjennomførte undersøkingane er samansett av mange forskjellige typar individ, noko som gjer at det beste ein kan håpe på i gjennomføringa av desse undersøkingane er ein ujamn korrespondanse mellom elevane sine handlingar og resultata av handlingsspørsmåla i undersøkinga (Oppenheim, 1992, side 148). Linken mellom haldningar og oppførsel er kompleks, og ein kan ikkje nødvendigvis føresjå oppførsel med utgangspunkt i haldningar, eller omvendt (Oppenheim, 1992, side 148). Desse faktorane svekker validiteten av resultata frå spørjeundersøkingane, noko som gjorde at forskingsintervju også vart nytta som metode i dette prosjektet.

Forskinga til Ron Garland (1991, side 4) konkluderer med at respondentar av spørjeundersøkingar ofte vel midtpunktet på ein skala. Dette vert gjort for å tilfredsstille intervjuaren, verke hjelpsam eller for å unngå å gi eit svar som kan oppfattast som sosialt uakseptabelt (Garland, 1991, side 4). For å unngå at elevane valte midtpunktet på rangeringsspørsmåla i desse undersøkingane vart rangeringsspørsmåla stilt på ein skala frå 1 til 6, der eit midtpunkt ikkje er tilgjengeleg.

I gjennomføring av intervju må fleire aspekt knytt til validitet vurderast. Sjølve intervjuhandlinga er svært essensiell, og validiteten her har med intervjuobjektet sin

truverdigheit og intervjuhandlinga sin kvalitet å gjere (Kvale et al., 2015, side 278). For å sikre validitet vart elevane i intervjurundane oppfordra til å svare så ærleg og nøyaktig som dei klarer, og det blei også før intervjuet nok ein gong presisert at alt datamaterialet vil bli anonymisert. Forskingsintervju vert ofte kritisert for at funna ikkje er valide fordi intervjuobjektet ikkje fortel sanninga under intervjuet (Kvale et al., 2015, side 281). For å kontrollere svara til elevane under intervjua vart kontrollspørsmål nytta. For å auke validiteten i intervjua vart det nytta lydopptakar, i motsetnad til å notere under intervjua. Dette sikra ei nøyaktig gjengiving av innhaldet i intervjua. Intervjua vart i etterkant transkribert, og lydopptaka sletta. I analysedelen vil også ein del av valideringsprosessen finne stad, ettersom at samlinga av alt datamateriale gjer ein i stand til å søke etter eventuelle skeivheiter som kan ugyldiggjere delar av det kvalitative datamaterialet (Kvale et al., 2015, side 279).

Generalisering refererer til om ein studie har evna til å produsere resultat som gjer det mogeleg å generalisere innanfor spesifikke grupper og fellesskap, situasjonar eller omstende (*intern* validitet), og vidare til spesifikke utanforliggande fellesskap, situasjonar eller omstende (*ekstern* validitet) (Cohen et al., 2011, side 181). Sidan det i dette prosjektet berre vert undersøkt ein 1P-klasse er den interne validiteten låg, og den eksterne validiteten endå lågare. Det vil seie at ein på bakgrunn av funna i dette prosjektet i låg grad er i stand til å generalisere funna for andre 1P-klassar, eller vidare til andre klasserom. Funna i studien er likevel ikkje utan nytteverdi, då dei kan nyttast som utgangspunkt eller datamateriale i andre liknande studiar.

3.9 Reliabilitet

Reliabilitet har med forskingsresultata sin konsistens og truverdigheit å gjere (Kvale et al., 2015, side 276). For at forsking skal vere påliteleg (eng: *reliable*) må den demonstrere at om den hadde vorte gjennomført på ei liknande gruppe og i ein liknande kontekst ville ein ha funne liknande resultat (Cohen et al., 2011, side 199).

Det finst ulike måtar å styrke reliabiliteten til gjennomføringa av ei spørjeundersøking på. I spørsmål som måler handlingar, slik som i dette prosjektet, vil ein kunne styrke reliabiliteten til resultata av undersøkingar ved å nytte skalaar til å måle dei haldningane som har størst viktigheit i undersøkinga (Oppenheim, 1992, side 147). Ved å nytte ein skala i ei undersøking

vil ein sikre at dei underliggande haldningane til kvart element i skalaen vert betrakta som vanlege, noko som vil bidra til å minke partiskheita og auke reliabiliteten til undersøkinga (Oppenheim, 1992, side 147). Dette er grunnen til at det er nytta skalaar på dei viktigaste spørsmåla i spørjeundersøkingane i dette prosjektet.

Ein intervjuprosess med stor grad av reliabilitet er lagt opp på ein slik måte at intervjuobjektet ikkje vil endre sine svar i eit liknande intervju med ein annan forskar (Kvale et al., 2015, side 276). Dette vil nok også vere litt avhengig av både intervjuobjekt og intervjuar, men ordvalet som ein intervjuar gjer i sitt oppsett av spørsmål er uansett eit essensielt tema når det kjem til reliabiliteten i eit intervju (Kvale et al., 2015, side 276). Oppenheim (1992, side 85) skriv at ord stilt i intervju kan tolkast på ulike måtar, og eksemplifiserer dette ved å seie at «å lese» kan tolkast på minst 3 ulike måtar: som å lese blad eller bøker frå start til slutt, som å bla igjennom eit blad mens ein sitt på eit venterom, eller som å kjøpe ei utgåve av ei bok eller eit magasin. Ulike tolkingar av omgrepet vil sjølvsagt resultere i ulike svar, noko som vil minke reliabiliteten i intervjuet (Oppenheim, 1992, side 85). For å styrke reliabiliteten i intervjua i dette prosjektet er det sett fokus på å ikkje stille leiande spørsmål, og å fortelje elevane kva som blir meint med intervjuspørsmåla om det skulle oppstå tvil om dette. Eksempelvis vil intervjua i denne studien utforske elevane sine oppfatningar og beskrivingar av omgrepa «algoritme» og «algoritmisk tenking», men etter at dette er vorte diskutert vil omgrepa verte definerte for elevane. Dette vert gjort for å sikre at omgrepa vert nytta i ein riktig kontekst vidare i intervjuet.

3.10 Forskingsetikk

Forskingsetikk refererer til dei mange verdiane, normene og institusjonaliserte einigheitene som utgjer og regulerer forskinga (Tangen, 2014, side 678). I dette prosjektet knyt etikken seg især til tre faktorar; innhenting av informert samtykke, anonymisering av prosjektdeltakarane, og sikker oppbevaring av datamaterialet.

Elevane vart i forkant informert munnleg om prosjektet og deira rettigheiter. Dei fekk i tillegg utdelt eit informasjonsskriv (vedlegg 2) som informerte om følgande:

- Masteroppgåva sine metodar og føremål
- Elevane sine rettigheiter og høve til å trekkje seg frå prosjektet

- Anonymisering av prosjektdeltakarar
- Sikker oppbevaring av datamaterialet
- Kontaktopplysningar til personar knytt til prosjektet

Informasjonsskrivet fungerer også som samtykkeskjema. Elevar under 16 år treng samtykke frå foreldre/føresette for å delta, mens elevar over 16 år kan samtykke sjølve. Elevane kan velje å melde seg til observasjon, spørjeundersøkingar og intervju med lydopptakar. Datainnhentinga gjekk føre seg i september og oktober 2020.

Prosjektet er meldt inn til norsk senter for forskningsdata (NSD), og vart godkjent 4. september 2020 (vedlegg 1). I tillegg vart leiinga ved skulen der prosjektet skulle gjennomførast informert om prosjektet via e-post før det vart sett i gang.

Spørjeundersøkingane vart gjennomført digitalt, og for å sikre anonymitet vart det tatt i bruk deltakarkodar. Kvar elev fekk i forkant av undersøkingane utdelt ein unik kode bestående av to bokstavar og eit tal. Desse kodane vart brukt i spørjeundersøkingane for å kunne kople saman dei to undersøkingane, og for å kunne vere i stand til å velje ut elevar til intervju med utgangspunkt i svara deira i undersøkingane. Bruk av slike kodar sikrar også at ingen elevnamn vert eksponert om datamaterialet som summerer opp svara på spørjeundersøkingane mot all tru skulle komme på avvege. Det kan også til ei viss grad verke tryggande for elevane å sleppe å identifisere seg ved bruk av namn på spørjeundersøkingane (Oppenheim, 1992, side 105).

Under intervjua vart det nytta ein lydopptakar som ikkje kunne koplast til internett. Transkripsjonen av intervjua vart gjennomført utan tilkopling til internett. Dette vart gjort for å vere absolutt trygg på at ingen utanforståande skulle få tak i intervjuopptaka som vert gjort i løpet av prosjektet. Etter att transkripsjonen av intervjua var gjennomført vart intervjuopptaka sletta.

Alt det digitale datamaterialet som kan koplast til elevane som deltok – det vere seg intervjuopptak, spørjeundersøkingar, deltakaroversikt, karakterkodar og transkripsjonar – vart oppbevart som krypterte filer på ei datamaskin som ikkje er tilgjengeleg for nokon som ikkje er delaktige i prosjektet. Dette vart også opplyst både munnleg og skriftleg til elevane før prosjektet starta opp. Datamaterialet som ikkje er digitalt – det vere seg underskrivne samtykkeskjema, lydopptakar og observasjonsnotat – vart oppbevart trygt, utilgjengeleg for andre enn prosjektleiar. Etter at prosjektet vart avslutta, og oppgåva fullført, vart alt datamaterialet knytt til prosjektet sletta eller destruert, jamfør informasjonsskrivet (vedlegg 2).

4 ANALYSE

4.1 Analytisk tilnærming

Det er nytta ulike måtar å analysere resultata frå dei ulike forskingsmetodane, basert på kva type datamateriale som er henta inn, og korleis dette datamaterialet er vorte henta inn.

Datamaterialet frå observasjonen av programmeringsundervisninga er handskrivne notat, og essensen frå desse notata vil bli presentert i delkapittel 4.2. Observasjonane vil bli presentert i underkapittel, der kvart underkapittel presenterer observasjonane frå ei matematikkøkt der programmering vart nytta.

Resultata frå spørjeundersøkingane vert direkte lagra i programmet SurveyXact. Etter at datamaterialet frå spørjeundersøkingane er vorte samla inn i SurveyXact vert det overført til eit rekneark. Dette gir ei systematisk oversikt over funna frå spørjeundersøkingane, og gir gode mogelegheiter for vidare analyse av svara.

Resultata frå spørjeundersøkingane er presenterte i delkapittel 4.3. Datamaterialet vert analysert ulikt, basert på om det er kvantitative eller kvalitative funn. Dei kvantitative funna frå spørjeundersøkingane vert presenterte i tabellar og diagram. Når det vert presentert kvantitative funn vert middelverdien og standardavviket til funna presentert på enkelte spørsmål. Middelverdien måler plasseringa til senteret av målingane, mens standardavviket kan tolkast som eit typisk eller representativt avvik frå middelverdien av målingane (Devore og Berk, 2012, side 34). Om ein antek at svara som elevane gir er omtrentleg normalfordelte kan ein nytte standardverdien til å finne eit intervall der om lag 68% av elevane sine svar ligg. Dette intervallet vert funne i ein avstand på eit standardavvik frå middelverdien av målingane (Devore og Berk, 2012, side 187). I denne analysen vert det antatt at elevane sine svar er normalfordelte. Både middelverdiane og standardavvika vert funne ved bruk av formlar i reknearket. I dei kvalitative funna i spørjeundersøkingane vert det sett på trendar og tendensar i svara til elevane. Der svara til elevane er såpass ulike at det ikkje er mogeleg å samle fleire svar under liknande kategoriar vil alle svara frå elevane verte presentert i ei liste.

Det er forskjell på talet på elevar som har gjennomført spørjeundersøkingane. Som beskrive i kapittel 3.7 er det 19 elevar som har gjennomført spørjeundersøking 1 og 14 elevar som har

gjennomført spørjeundersøking 2. Sidan samanlikning av responsane på spørjeundersøking 1 og spørjeundersøking 2 står sentralt i dette prosjektet vert resultata i delkapittel 4.3.1 for alle elevane som har gjennomført enten berre den fyrste eller begge undersøkingane presentert som gruppe 1, mens resultata frå dei elevane som har gjennomført begge undersøkingane vert presentert som gruppe 2. Gruppe 2 er altså ei delmengd av gruppe 1. Dette gjer at ein kan samanlikne resultata frå spørjeundersøking 1 og spørjeundersøking 2 for gruppe 2, sidan desse svara ikkje vil vere uavhengige. Sidan gruppe 1 og gruppe 2 inneheld 14 av dei same elevane vil forskjellane på dei to gruppene kome som eit resultat av dei 5 ekstra elevane som finst i gruppe 1.

Forskingsintervjua har etter gjennomføringa vorte transkriberte. Dette vart gjort for å enklare kunne analysere datamaterialet frå intervjua. I analysen av intervjua er det elevane sine meiningar, tankar og synspunkt som står sentralt. Desse vert forsøkt tolka frå transkripsjonane, og eksempel på intervjuepisodar vert presentert frå transkripsjonane. Dette vert gjort i delkapittel 4.4.

4.2 Observasjon

I dette delkapittelet vil resultata frå observasjonen av undervisningsøktene verte presenterte. Det vil vere fokus på pensumet som vart gjennomgått i timane, men enkelte reaksjonar og oppførslar frå elevane vil også verte inkludert. Bruk av programmering vart inkludert i matematikkundervisninga i kapittelet om matematiske formlar.

4.2.1 Økt 1

Økta varer i 1 time og 30 minutt.

Dette var elevane si fyrste økt der programmering var eit tema. Spørjeundersøking 1 vert gjennomført i fyrste del av økta, mens resten av økta går med på å installere programmet som skulle nyttast i vidare programmeringsarbeid. Bortsett frå å vise på projektor korleis dette skulle gjerast presenterer ikkje lærar noko anna fagstoff.

4.2.2 Økt 2

Økta varer i 2 timar og 45 minutt.

Læraren begynner denne timen med å snakke om algoritmisk tenking. Lærar innfører omgrepet «algoritme» for elevane, og forklarer at ein algoritme kan tenkast på som ei nøyaktig oppskrift på korleis ei oppgåve kan løysast. Her gir læraren to eksempel på algoritmar som elevane har nytta i matematikkfaget tidlegare; algoritmen for å løyse ei likning og algoritmen for å dele to tal på kvarandre.

Læraren går deretter over til å snakke om programmeringsspråket Python. Det fyrste som vert presentert her er oppsettet i Spyder, programmet som klassen skal nytte til å kode i Python. Læraren bruker projektor for å vise programmet i plenum. Ei forklaring av oppsettet med teksteditor og konsoll følger, og læraren forklarer at programmet les koden i teksteditoren linje for linje, og viser resultatet av koden i konsollen. Deretter viser læraren korleis nye filer vert lagra.

Det fyrste programmet læraren demonstrerer er programmet som skriv ut teksten «Hello World» til konsollen ved bruk av ei print-setning. Etter denne demonstrasjonen skriv elevane inn den same kodelinja i sine program. Her er det mange spørsmål som melder seg, og læraren går rundt for å hjelpe elevane. Elevane treng typisk hjelp til ein av to ting; dei har enten gløymt å køyre programmet, eller så har dei køyrt programmet, men skjønner ikkje at programmet har blitt gjennomført. Etter at elevane har fått tilstrekkeleg hjelp med programmet legg læraren til endå ei print-setning i programmet for å demonstrere at programma dei lager vert lest og utført linje for linje.

Det neste som vert gjennomgått er rekning ved bruk av kodeprogram. Her viser læraren korleis ein kan rekne direkte i konsollen og i teksteditoren. Det vert gjennomgått reknerekkefølge, samanskøyting av strengar, og rekning i print-setningar.

Variablar vert så gjennomgått. Her viser læraren korleis ein kan opprette variablar i program, lagre verdiar i variablane, og nytte desse til å rekne med i programmet. Eksempelet læraren viser i plenum er korleis ein kan opprette to variablar, lengde og breidde, nytte desse til å rekne ut arealet til eit rektangel, og deretter skrive dette arealet ut til brukaren. Etter denne gjennomgangen gir læraren elevane i oppgåve å lage eit program som i staden for å skrive ut arealet til brukaren, skriv ut omkrinsen av det same rektangelet. I løpet av dette arbeidet blir det

klart at det som gjer at elevane i størst grad ikkje får eit fungerande program er lite fokus på syntaksen i programma; det vere seg spesielt å skrive ord identisk i programma og unngå unødige innrykk og mellomrom i koden. Det er også ein del som også her gløymer å køyre programmet.

Etter ei stund med oppgåvearbeid blir det siste som læraren går igjennom i denne timen bruk av input-setninga. Etter ein gjennomgang av verkemåten til denne funksjonen får elevane i oppgåve å lage eit program som tek inn alderen til ein brukar ved bruk av ei input-setning, og skriv ut ei bursdagshelsing til denne personen. Mange av elevane har vanskar med å tolke forskjellen på print- og input-funksjonen, då dei føler at begge desse funksjonane er noko som blir skrive ut i konsollen når programmet blir køyrt. Mange overser då den vitale forskjellen at input-funksjonen hentar inn informasjon frå brukaren av programmet, mens print-funksjonen skriv ut informasjon gitt av programmeraren.

Figur 4.1 viser 4 av oppgåvene som vart gitt til elevane i løpet av denne timen (oppgåver vist med løyve frå lærar):

Oppgave 1

```
1   strekning = 100
2   fart = 60
3
4   tid = strekning/fart
5
6   print(tid)
```

- Her ser du utklipp av kodelinjene i et program. Hva gjør programmet?
- Skriv av koden og kjør programmet. Skjer det du forventet skulle skje?
- Endre på tallene i programmet og kjør programmet igjen. Hva skjer?
- Endre koden slik programmet beregner strekningen ut fra fart og tid.

Oppgave 2

Lag et program som beregner omkrets og areal til et kvadrat. Programmet skal bruke kvadratets sidelengde som input.

Oppgave 3

Lag et program som regner om temperatur fra grader Celsius til Fahrenheit og omvendt.

Oppgave 4

Lag et program som beregner omkrets og areal til en sirkel. Programmet skal bruke sirkelens radius som input.

Figur 4.1 - Eksempel på programmeringsoppgåver gitt til elevane i matematikk 1P

4.2.3 Økt 3

Økta varer i 1 time og 30 minutt.

Denne økta er ei arbeidsøkt; elevane arbeider med programmeringsoppgåver knytt til det dei arbeida med i førre økt, utan at nytt fagstoff vert presentert. I byrjinga av timen presiserer læraren viktigheita av å forstå kva ein algoritme er, og seier at elevane i oppgåvearbeidet skal skrive ned ein algoritme for kvar oppgåve før dei løyser oppgåva ved bruk av Python. Dette for å styrke elevane si kompetanse knytt til det å skrive og forstå algoritmar. Læraren gir her eit eksempel på korleis ein slik algoritme kan sjå ut på ei oppgåve som ber elevane om å lage eit program som skriv ut arealet av eit rektangel til brukaren. Læraren presiserer at stega tatt for å skrive algoritmen kan generaliserast til ei kva som helst oppgåve.

I løpet av oppgåvearbeidet er det stor pågang for å få hjelp frå lærarane. Det vert ofte observert at elevane støyter på ei feilmelding i programma sine, retter opp handa og venter ei god stund på hjelp utan å prøve og sjølv finne og rette opp feilen. I løpet av denne tida vert PCen svært ofte nytta til ufaglege aktivitetar.

Det som vert observert som dei vanlegaste feila hos elevane er at det er nokon ord som er skrive feil i forhold til måten dei er skrive på tidlegare i programmet, at det finst eit innrykk eller eit mellomrom for mykje i programmet, eller at elevane nytter print-setningar der input skulle ha vorte nytta, eller omvendt. Desse forvekslingane kan kome som ein konsekvens av at print-funksjonen og input-funksjonen sine syntaksar for ein uerfaren programmerar kan synast å vere nokså like, og det kan av den grunn verke som om dei har like bruksområde.

I tillegg til frustrasjon over feil vert det også observert frustrasjon over sjølve programmeringa som ein del av pensumet til elevane. Mange uttrykker at dei ikkje skjønner kvifor programmering er eit tema, då dei kunne ha nytte andre arbeidsmåtar for å finne svar på oppgåvene på ein mindre arbeids- og tidkrevjande måte.

4.2.4 Økt 4

Økta varer i 2 timar og 45 minutt.

Læraren presenterer i byrjinga av timen dei tre elementa som er tenkt gjennomgått i denne timen; datatypar, feilmeldingar og handlingsval.

Læraren presenterer tre datatypar for elevane; streng, heiltal og flyttal, og viser korleis ein kan omforma strengar til heiltal og flyttal i programmet. Elevane kopierer koden som vert gjennomgått på projektor i sine eigne kodeprogram i løpet av presentasjonen til lærar. Læraren gir etter gjennomgangen sin elevane ei oppgåve; eit kodeprogram vert vist på projektoren, og elevane skal diskutere verkemåten til dette programmet. I denne diskusjonen går læraren rundt, og hjelper dei som har hatt problem i løpet av gjennomgangen i plenum. Elevane diskuterer i byrjinga, men ein god del tid vert sett av til denne aktiviteten, noko som fører til at elevane etter kvart går over til ufagleg snakk og aktivitetar. Ein elev gir etter arbeidet ei korrekt oppsummering av korleis programmet fungerer. Elevane sett deretter i gang med oppgåvearbeid.

Etter oppgåvearbeidet og pause snakkar læraren om feil i kodeprogram, og skil mellom to hovudtypar; feil program som ikkje gir feilmelding og feil innskriving som gir feilmelding. Deretter viser læraren nokon typiske feilmeldingar som kan oppstå i programma, korleis desse skal tolkast, og korleis dei kan rettast opp i. Det vert også presisert frå lærar at feil er ein naturleg del av det å programmere, og at det er viktig at elevane sjølv prøver å rette opp i programma sine når slike feil oppstår, i staden for å be om hjelp frå lærar med ein gong. Elevane sett igjen i gang med oppgåvearbeid etter dette, og det merkast at elevane i mindre grad ber om hjelp frå lærar under arbeidet.

I samtale med lærar etter denne timen kom det fram at det blei gjort ei vurdering om at handlingsval, ved bruk av if/elif/else-logikk, ikkje skulle gjennomgåast i plenum. Dei elevane som læraren tolka som klare til å lære neste steg innan programmering vart sendt til NDLA sine nettsider der dei las om handlingsval på eiga hand.

4.3 Spørjeundersøkingar

Eg vil i dette delkapittelet presentere funna frå spørjeundersøkingane gjennomført i prosjektet. Eg vil ta føre meg kvar undersøking separat, og presentere funna frå spørjeundersøkingane eit og eit spørsmål om gongen. Begge spørjeundersøkingane ligg som vedlegg.

4.3.1 Spørjeundersøking 1

Dette delkapittelet summerer opp resultata frå den fyrste spørjeundersøkinga som elevane gjennomførte før undervisninga som nytta programmering (vedlegg 3).

4.3.1.1 Tidsmengd for ei typisk matematikkoppgåve

Under følger resultatet for dette spørsmålet. Der elevane har oppgitt tidsintervall, til dømes 3-5 minutt, vert middelverdien i dette intervallet nytta i utrekninga.

Tabell 4.1: Respondansar om tidsmengd på typisk matematikkoppgåve

Gruppe	*Elevar i gruppa*	*Lågaste verdi*	*Høgaste verdi*	*Middelverdi*	*Standardavvik*
1	19	20 sekund	5-10 minutt	2 min 49 sek	2 min 7 sek
2	14	20 sekund	6 minutt	2 min 40 sek	2 min 1 sek

Begge gruppene har ein middelverdi som er ein del høgare enn middelverdien Alan Schoenfeld (1989a, side 345) fann i sin studie. I den studien var middelverdien litt under 2 minutt. Om lag 68% av elevane i gruppe 2 har gitt eit svar som ligg mellom 39 sekund og 4 minutt og 41 sekund. Det at standardavviket for dei to gruppene er omtrent like stort som middelverdien seier at elevane har gitt svar med stor spreiing. Dette kan også observerast av høgaste og lågaste verdi.

4.3.1.2 Rimeleg tidsmengd før ein gir opp på ei matematikkoppgåve

Også her er middelverdien til tidsintervall nytta i utrekningane.

Tabell 4.2: Respondansar om rimeleg tidsmengd før ein gir opp på ei matematikkoppgåve

Gruppe	*Elevar i gruppa*	*Lågaste verdi*	*Høgaste verdi*	*Middelverdi*	*Standardavvik*
1	19	4 minutt	25 minutt	10 min 25 sek	5 min 28 sek
2	14	4 minutt	25 minutt	10 min 34 sek	6 min 17 sek

I Schoenfeld (1989a, side 345) sin studie var middelverdien på dette spørsmålet noko høgare; 12 minutt. Frå lågaste og høgaste verdi, samt dei relativt høge standardavvika, ser ein at elevane har gitt svar som strekkjer seg over eit langt intervall, så det er usemje hos elevane på dette spørsmålet. Om lag 68% av elevsvara i gruppe 2 ligg i intervallet frå 4 minutt og 17 sekund til 16 minutt og 51 sekund.

4.3.1.3 Tidlegare erfaringar med programmering

På dette spørsmålet har i alt 4 av dei 19 elevane svart at dei har tidlegare programmeringserfaring. Av desse 4 elevane har 2 av dei kryssa av på svaralternativet som indikerer at dei ikkje husker kva programmeringsspråk dei har nytta tidlegare, ein elev har kryssa av på «Andre», mens den siste eleven har erfaring med Python, Scratch og andre programmeringsspråk. Fordelinga av erfaring i gruppe 1 og gruppe 2 er summert opp i tabell 4.3.

Tabell 4.3: Respondansar om tidlegare programmeringserfaring

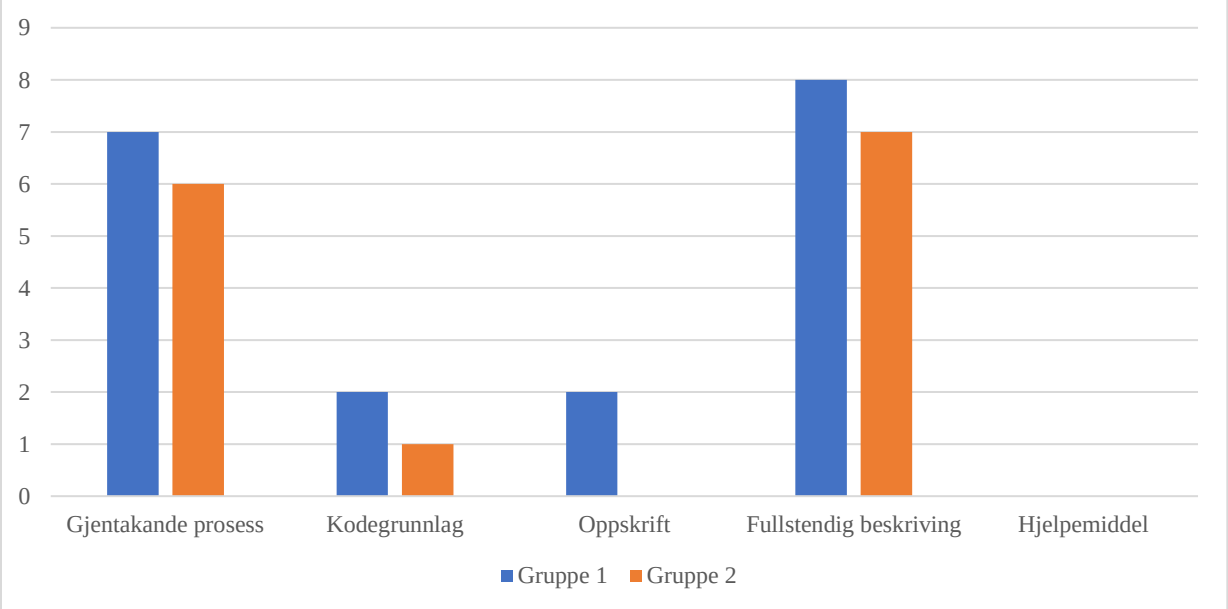
Gruppe	*Elevar i gruppa*	*Tidlegare erfaring frekvens*	*Tidlegare erfaring prosentdel*
1	19	4	21,0%
2	14	2	14,3%

Ein ser at det totalt i gruppe 1 er 4 elevar med tidlegare programmeringserfaring, og 2 av desse er også med i gruppe 2.

4.3.1.4 Definisjon av omgrepet «algoritme»

Her vil eg gjennomgå svara på spørsmålet som ber elevane velje det svaralternativet dei føler best beskriv omgrepet «algoritme». Talet på elevar som har svart dei ulike svaralternativa vert framstilt i eit stolpediagram, deretter kjem ein tabell med prosentverdiane knytt til kvart av svaralternativa.

Diagram 4.1: Respondansar om beskriving av algoritme. Tala på y-aksen er talet på elevar som har gitt det aktuelle svaralternativet.



Tabell 4.4: Respondansar om beskriving av algoritme

Gruppe	*Elevar i gruppa*	*Gjentakande prosess*	*Kode-grunnlag*	*Oppskrift*	*Fullstendig beskriving*	*Hjelpe-middel*
1	19	36,8%	10,5%	10,5%	42,1%	0,0%
2	14	42,9%	7,1%	0,0%	50,0%	0,0%

Ein ser av tabell 4.4 og diagram 4.1 at dei fleste elevane trur at ein algoritme er ein gjentakande prosess som alltid gir same resultat eller ei fullstendig beskriving av framgangsmåten for å løyse ei oppgåve. Det er verdt å merke seg at dette spørsmålet ikkje gir elevane mogelegheit til å hoppe over oppgåva utan å svare, eller å krysse av på at dei ikkje veit kva ein algoritme er.

4.3.1.5 Beskriving av omgrepet «algoritmisk tenking»

Av dei 19 svara som vart gitt på dette spørsmålet gir 10 elevar eit tomt svar; dvs. eit svar som fortel at eleven ikkje veit kva algoritmisk tenking er, til dømes «eg veit ikkje». Fordelinga av desse svara er vist i tabell 4.5:

Tabell 4.5: Respondansar med tomme svar på beskriving av algoritmisk tenking

Gruppe	*Elevar i gruppa*	*Tomme svar frekvens*	*Tomme svar prosentdel*
1	19	10	52,6%
2	14	7	50,0%

Lista under viser svara elevane gav som ikkje var tomme:

1. *«Er ikke sikker, men kanskje hvordan man tenker når det gjelder algoritmer»*
2. *«Jeg er veldig usikker på hva dette begrepet betyr, men tror det vil si at du tenker deg frem til hvordan en oppgave skal løses»*
3. *«Å tenke nye i en bestemt rekkefølge over lengre tid»*
4. *«At du tenker deg fram til hva du skal gjøre»*
5. *«Tenker deg frem til hva du skal gjøre og beskriver det?»*
6. *«Jeg tror at det er noe som blir gitt til deg, og at du gjennomfører det med samme resultat. I programmering er det lett forklart. Du gir kommandoer til et program, og den gjennomfører det med samme resultat om du ikke endrer kommandoene.»*
7. *«Jeg tror det er en måte å løse oppgaver på. Det kan være å dele opp oppgaven slik at du kanskje forstår den bedre.»*
8. *«At du tenker det samme om igjen, men kanskje på en litt annen måte enn første gangen.»*
9. *«Taktisk og systematisk tenkning.»*

Fleire av elevane uttrykker at dei ikkje veit kva algoritmisk tenking er, men gir likevel ei beskriving som dei tenker kan passe. Beskriving 7 legg vekt på å dele oppgåva opp i mindre oppgåver, som saman gir ei løysing på problemet. Dette ligg tett opptil slik algoritmisk tenking er beskrive i læreplanen (sjå kapittel 1.1). Sjølv om beskriving 2 er vag, er den riktig. Beskrivinga fastslår at algoritmisk tenking er ein tankeprosess nytta for å løyse oppgåver. Beskriving 6 dreg inn digitale hjelpemiddel (i form av programmering) som del av beskrivinga, noko som også er vanleg når ein definerer algoritmisk tenking, sjølv om beskrivinga i seg sjølv er unøyaktig. Dei fleste svara legg vekt på at det er ein måte å tenke på og/eller ein metode som vert nytta for å komme fram til korleis ei oppgåve skal løysast. Beskriving 9 er konsis og ikkje særleg konkret formulert, men inneheld likevel ordet «systematisk»; eit ord som også er nytta i utdanningsdirektoratet sin definisjon av algoritmisk tenking (Utdanningsdirektoratet, 2019, side 1).

4.3.1.6 Rangeringsspørsmål

Tabell 4.6 og 4.7 viser resultata frå rangeringsspørsmåla på spørjeundersøking 1. Tabellane inneheld resultata frå kvar si gruppe. På desse spørsmåla fekk elevane 4 påstandar, og skulle

plassere kvar av desse påstandane på ein skala frå 1 til 6. På denne skalaen indikerte 1 at eleven var heilt ueinig i påstanden, mens 6 indikerte at eleven var heilt einig.

Tabell 4.6: Respondansar på rangeringsspørsmåla frå gruppe 1 (N = 19)

Gruppe 1	*Aldri bruk for matematikk*	*Matte relevant?*	*Digitale hjelpemiddel ungdomsskule*	*Digitale hjelpemiddel VGS*
Middelverdi	2,37	2,67	2,42	3,29
Standard-avvik	1,38	1,29	1,14	1,60

Tabell 4.7: Respondansar på rangeringsspørsmåla frå gruppe 2 (N = 14)

Gruppe 2	*Aldri bruk for matematikk*	*Matte relevant?*	*Digitale hjelpemiddel ungdomsskule*	*Digitale hjelpemiddel VGS*
Middelverdi	2,14	2,54	2,50	3,08
Standard-avvik	1,41	1,08	1,24	1,75

Frå tabellane ser ein at standardavvika alle ligg mellom 1 og 2, noko som fortel oss at om lag 68% av elevane sine svar ligg i ein avstand på mellom 1 og 2 frå middelverdien. Alle middelverdiane ligg under midtverdien 3,5 på skalaen, noko som indikerer at elevane generelt sett er meir ueinige enn dei er einige i påstandane. Lågast scorar påstanden om at elevane ikkje kjem til å få bruk for matematikken dei lærar på skulen. Dette gir eit bilete av at dei fleste elevane ser nytteverdien av kunnskapane dei opparbeider seg i løpet av matematikkundervisninga i skulegangen sin. Påstanden om at elevane håpar på eit stort fokus

på bruk av digitale hjelpemiddel på VGS scorar høgast, men middelverdien ligg likevel på nedre del av skalaen. Det er verdt å merke seg at også standardavviket er høgast på denne påstanden, noko som fortel oss at elevane sine svar er mest spreidd på påstanden.

4.3.1.7 Nyttigheita av programmering i matematikkundervisninga

På dette spørsmålet er det delte meiningar i klassen. Svara på desse spørsmåla vart gitt i form av fritekst, og i tabell 4.7 er det vist ei tolking av desse svara. I denne tabellen er eit svar tolka som positivt om eleven som har gitt svaret har uttrykt at han/ho trur programmering vil vere nyttig, svaret er tolka som negativt om eleven som gitt svaret har uttrykt at han/ho ikkje trur at programmering vil vere nyttig, og svaret er tolka som ubestemt om eleven som har gitt svaret ikkje gir klart uttrykk for at han/ho trur programmering vil vere nyttig eller ikkje.

Tabell 4.7: Respondansar om nyttigheita av programmering i matematikkundervisninga

Gruppe	*Elevar i gruppa*	*Positiv*	*Negativ*	*Ubestemt*
1	19	9 (47,4%)	7 (36,8%)	3 (15,8%)
2	14	5 (35,7%)	7 (50,0%)	2 (14,3%)

Av dei elevane som har gitt negative svar er det to elevar utan tidlegare programmeringserfaring som har gitt svar som indikerer at dei, allereie før undervisninga har starta opp, har bestemt seg for at dei ikkje kjem til å skjønne programmeringa. Svara deira er lista nedanfor:

1. *«Nei, forstår det ikke»*
2. *«Nei, det er berre forvirrende»*

Av dei negative svara som inneheld grunngjevingar så baserer desse seg i all hovudsak på forståing og interesse.

Den største delen av elevane som har gitt positive svar grunngir dette med at programmering er eit fagfelt som vert meir og meir aktuelt, ettersom at samfunnet i aukande grad tek i bruk teknologiske løysingar. To eksempel på slike svar er gitt her:

1. *«Jeg tror det er nyttig ettersom att verden blir mer og mer digital»*
2. *«Jeg tror det vill være nyttig fordi programmering blir mer og mer relevant for oss. Vi bruker og matematikk hele tiden og overalt så de ville det vært fint å kunne litt om det.»*

Det kan tenkast at eleven i tilbakemelding 2 meinte å skrive «programmering» i staden for «matematikk» i den andre setninga, i og med at eleven forhåpentlegvis allereie kan litt om matematikk. Det at bruk av programmering kan bidra til å auke effektiviteten i arbeidet med matematikk vert også dratt fram som eit argument i dei positive svara.

Totalt 3 svar er vorte klassifiserte som ubestemte svar. Desse er gitt her:

1. *«Tror det vil være nyttig for de som tar t-matte og vil satse»*
2. *«Tror det kan forvirre mange elever, siden det er så nytt for mange.»*
3. *«Jeg tror programmering i faget vil bli en stor utfordring for meg fordi jeg allerede synes matte er vanskelig å forstå. Jeg tror ikke jeg kommer til å få bruk for programmering i matte senere i livet så jeg tror ikke det blir så nyttig. For meg er det best å skrive ned og selv regne ut mattematikkstykker på ark. Det gjør at jeg forstår fremgangsmåten på stykket bedre, og jeg husker bedre av å skrive selv. Jeg føler at programmering kan være litt stress fordi vi må kanskje huske på forskjellige formler, i tillegg til å forstå oppgaven. Jeg tror det vil gjøre det veldig forvirrende. Samtidig kan det være bra fordi vår generasjon bruker nesten ikke bøker lengre. Vi bruker nesten bare teknologi.»*

Eleven som har gitt tilbakemelding 1 indikerer indirekte at han/ho meiner at programmering ikkje vil vere nyttig i P-klasserommet. Eleven som gir tilbakemelding 2 svarer ikkje på spørsmålet som vert stilt, men får fram trua si på at programmering kan verke forvirrande. Eleven som har gitt tilbakemelding 3 uttrykkjer bekymringar knytt til bruk av programmering som eit ukjend verktøy, sidan eleven synest å meine at han/ho lærer best ved å bruke penn og papir. Samtidig skriv eleven at det kan vere bra å lære om programmering grunna ei auke i bruk av teknologi.

4.3.2 Spørjeundersøking 2

I dette delkapittelet er funna frå spørjeundersøkinga som elevane gjennomførte etter undervisninga om programmering presenterte (vedlegg 4). Alle elevane som gjennomførte denne undersøkinga hadde på førehand gjennomført spørjeundersøking 1, så her blir resultata presentert som ei gruppe. Resultata til gruppa som vert presentert her svarer til gruppe 2 frå delkapittel 4.3.1.

4.3.2.1 Undervisningsforventingar

Resultatet frå dette spørsmålet er framstilt i tabell 4.9. Her er svara der elevane svarte at undervisninga gjekk føre seg som forventa klassifiserte som positive, svara der elevane svarte at undervisninga ikkje gjekk føre seg som forventa klassifiserte som negative, og dei svara som ikkje kan puttast i ein av dei to kategoriane er klassifiserte som ubestemte.

Tabell 4.9: Responsar om undervisningsforventingar

Tal på elevar	*Positiv*	*Negativ*	*Ubestemt*
14	4	4	6

Dei positive svara gir ikkje meir grunngjeving enn at det heile gjekk føre seg som forventa.

Her er responsane frå dei elevane som har gitt negative svar:

1. *«Nei, eg trodde det skulle være bare kjedelig programmering, men det var faktisk gøy»*
2. *«Nei, det var mye enklere enn det jeg trodde. Undervisningen var også mer forenklet enn det jeg hadde forventet.»*
3. *«Nei, jeg trodde det var mye mer komplisert»*
4. *«NEI»*

Blant dei ubestemte responsane har 5 elevar svart at dei ikkje forventa noko før undervisninga starta, og at dei av den grunn ikkje kan svare på dette spørsmålet. Den siste ubestemte eleven har gitt følgande svar:

- *«Ja men jeg trodde de skulle forklare mer om hvordan de gjorde det og hvorfor det funket akkurat som det gjorde. Det var heller ikke like lett som jeg trodde det skulle bli. Det var mye som du måte huske for at helle programmet skulle funke. Og hvis du skrev en ting feil så funket det ikke.»*

4.3.2.2 Tidsmengd for ei typisk programmeringsoppgåve

Under følger resultatet for dette spørsmålet. Der elevane har oppgitt tidsintervall, til dømes 3-5 minutt, vert middelverdien i intervallet nytta i utrekninga.

Tabell 4.10: Respondansar om tidsmengd på typisk programmeringsoppgåve

Tal på elevar	*Lågaste verdi*	*Høgaste verdi*	*Middelverdi*	*Standard-avvik*
14	3 min	15 min	6 min 53 sek	3 min 11 sek

Middelverdien til den same gruppa elevar aukar med heile 157,7% frå det tilsvarande spørsmålet på spørjeundersøking 1, der programmering ikkje var involvert i oppgåva. Om lag 68% av elevane sine svar ligg i intervallet frå 3 minutt og 42 sekund til 10 minutt og 4 sekund. Legg merke til at middelverdien frå det tilsvarande spørsmålet på spørjeundersøking 1 ikkje ligg i dette intervallet. Elevane meiner altså at det trengs ein god del lenger tid for å løyse ei oppgåve som involverer programmering.

4.3.2.3 Rimeleg tidsmengd før ein gir opp på ei programmeringsoppgåve

Også her er middelverdien til tidsintervall nytta i utrekningane.

Tabell 4.11: Repondansar om rimeleg tidsmengd før ein gir opp på ei programmeringsoppgåve

Tal på elevar	*Lågaste verdi*	*Høgaste verdi*	*Middelverdi*	*Standard-avvik*
14	5 min	45 min	14 min 24 sek	9 min 40 sek

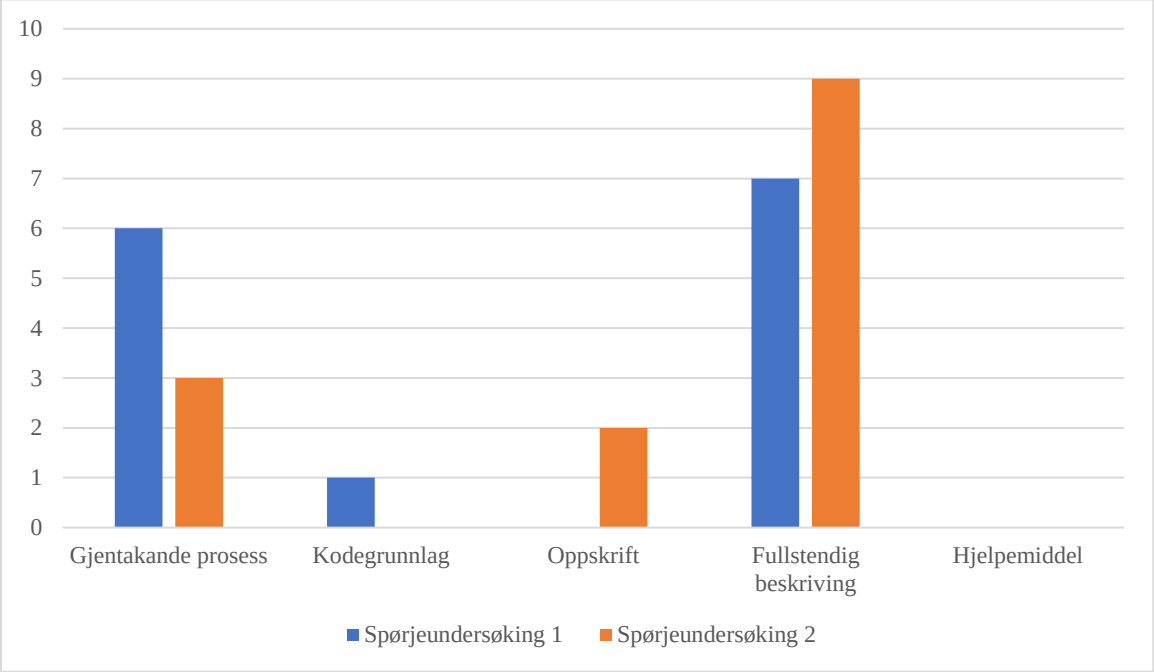
Elevgruppa aukar her svaret sitt med 36,2% frå middelverdien på det tilsvarande spørsmålet på spørjeundersøking 1. Også på dette spørsmålet ser ein ei stor spreiing i svara som er gitt; differansen til det høgaste og lågaste svaret på dette spørsmålet er på heile 40 minutt. Her gir den statistiske utrekninga at om lag 68% av elevsvara ligg innafor intervallet frå 4 minutt og 44 sekund til 24 minutt og 4 sekund.

Auken i tid på dette spørsmålet er ikkje like drastisk som auken på oppgåva om den typiske oppgåva. Likevel ser ein at elevane generelt sett set av lenger tid til ei oppgåve som inkluderer programmering.

4.3.2.4 Definisjon av omgrepet «algoritme»

Under følger resultata frå dette spørsmålet; fyrst i eit diagram som viser talet på elevar som har svart på kvar av svaralternativa, deretter ein tabell som viser prosentfordelinga på dei ulike svaralternativa. Diagram 4.4 samanliknar svara gitt av elevane (i gruppe 2) på spørjeundersøking 1 og spørjeundersøking 2.

Diagram 4.2: Respondansar om beskriving av algoritme. Tala på y-aksen er talet på elevar som har gitt det aktuelle svaralternativet.



Tabell 4.12: Respondansar om beskriving av algoritme

Frekvens	*Gjentakande prosess*	*Kode-grunnlag*	*Oppskrift*	*Fullstendig beskriving*	*Hjelpe-middel*
14	21,4%	0,0%	14,3%	64,3%	0,0%

Ein ser av diagrammet og tabellen at ein større del elevar svarer at ein algoritme er ei fullstendig beskriving av framgangsmåten for å løyse ei oppgåve. Færre elevar svarar at dei meiner ein algoritme er ein gjentakande prosess som alltid gir same resultat. Ingen trur lenger at ein algoritme er samlinga av kodegrunnlaget til eit dataprogram eller ein app, mens to elevar på spørjeundersøking 2 svarer eit alternativ som sto tomt på spørjeundersøking 1; at ein algoritme er ei oppskrift på korleis ei oppgåve blir løyst mest mogeleg nøyaktig.

4.3.2.5 Beskriving av omgrepet «algoritmisk tenking»

6 av elevane (42,9%) har gitt tomme svar på dette spørsmålet. Svara som ikkje er tomme på dette spørsmålet er gitt i denne lista:

1. *«Gjentakende tenking»*
2. *«En konkret forklaring på hvordan man skal løse en oppgave.»*
3. *«Systematisk og godt strukturert tenkning.»*
4. *«Tenke lurt»*
5. *«At du tenker på same måte på alle oppgaver du får.»*
6. *«En prosess til å tenke frem et svar steg til steg elnss»*
7. *«En måte å beskrive en fremgangsmåte på»*
8. *«En nøyaktig tankemåte eller en prosess av forskjellige situasjoner.»*

Det fyrste som er verdt å merke seg er at ein mindre del av elevane har gitt tomme svar på dette spørsmålet i forhold til det tilsvarande spørsmålet på spørjeundersøking 1, der prosentdelen tomme svar var 50,0%. Elevane verkar også sikrare på svara sine, i og med at ingen av dei på dette spørsmålet tek avstand frå spørsmålet ved å for eksempel å skrive «Eg veit ikkje, men eg trur…», som var tendensen på svara som vart samla inn på det tilsvarande spørsmålet på spørjeundersøking 1. Svara er også generelt sett kortare enn svara frå spørjeundersøking 1.

Beskriving 2 gir ei god beskriving av kva ein algoritme er, men den algoritmiske tenkinga er ikkje forklart. Beskriving 7 ligg litt nærare ei forklaring på den algoritmiske tenkinga, men også denne beskrivinga baserer seg på kva ein algoritme er. Resten av beskrivingane fortel om det å tenke. Den beskrivinga som ligg nærast ei korrekt beskriving av algoritmisk tenking er beskriving 6, som fokuserer på å finne eit svar steg for steg. Beskriving 3 er også god, då den gjer reie for at algoritmisk tenking er systematisk og strukturert, sjølv om beskrivinga i seg sjølv er noko vag.

4.3.2.6 Rangeringsspørsmål

Under følger tabell 4.13, som summerer opp svara elevane har gitt på rangeringsspørsmåla på denne spørjeundersøkinga. Elevane skal rangere 5 påstandar om programmering på ein skala frå 1 til 6, der 1 er heilt ueinig og 6 er heilt einig.

Tabell 4.13: Respondansar på rangeringsspørsmål

	Aldri bruk for programmering	*Nyttig verktøy*	*Frustrerande arbeid*	*Lærerikt arbeid*	*Moro arbeid*
Middelverdi	3,93	3,46	3,86	2,92	2,71
Standard-avvik	1,53	1,22	2,03	1,39	1,49

Middelverdiane til elevgruppa ligg mellom 3,93 og 2,71, noko som er nokså sentralt på skalaen. Den høgaste middelverdien finn ein på påstanden om at eleven aldri får bruk for programmeringa som vert undervist på VGS. Det vil seie at elevane generelt sett er skeptiske til at dei seinare får bruk for programmeringa som vert undervist. Til samanlikning var middelverdien 2,14 på påstanden frå spørjeundersøking 1, som påsto at eleven aldri får bruk for matematikken som vert undervist på VGS seinare i livet.

Verdiane frå tabell 4.13 viser at elevane syns programmering var tidvis frustrerande, og at middelverdien om lag ligg på midtverdien av skalaen på spørsmålet om dei syns programmering verka som eit nyttig verktøy. Dei lågaste middelverdiane finn ein på påstandane om at arbeid med programmering var lærerikt og moro. Det bør også merkast at standardavvika er høge, noko som gir evidens for stor spreiing i svara.

4.3.2.7 Vurdering av undervisning

5 av elevane har svart at dei ikkje ser nokon måte undervisninga kunne blitt lagt opp annleis på for at deira faglege utbytte skulle ha vorte betre. Av desse 5 legg ein elev til at han/ho syns opplegget var bra, mens ein annan elev legg til at han/ho syns opplegget var unødvendig.

Dei resterande 9 elevane har alle komme med forslag til korleis opplegget kunne ha blitt justert på ein måte som kunne ha fremma deira faglege utbytte. Desse responsane er lista opp her:

1. *«Gitt lekser, lagt ut videoer til elever som kan synes det er vanskelig å få med seg så mye informasjon på 1 time. Lagt opp arbeid som sjekket at alle elever henger med og at det også skal være lettere å henge med.»*
2. *«Ja senere når man skjønner det grunnleggende, burde det bli lagt mer vekt på hvordan man kan ha nytte av det»*
3. *«Ikke så lange timer med programmering.»*
4. *«Hvis oppgavene reflekterte mer til faktiske spørsmål vi vil så ovenfor i framtiden.»*
5. *«Jeg skjønte ingenting, kunne gjerne blitt lært opp bedre.»*
6. *«Ja. Lærerene kunne forklart mer spesifikt hvordan man gjorde tingene. Gitt oss oppgaver som lignet på det vi gjorde mer fordi eg skjønte ikke så mye av det.»*
7. *«Hvis oppgavene hadde vært gøyere»*
8. *«Lagt ut videoer, for det er mye å huske og da er det greit å ha noen videoer å se på»*
9. *«Mer repetisjon»*

Dei generelle trendane i tilbakemeldingane går på at oppgåvene kunne ha vore meir relevante, at det var mykje arbeid og lange timar, samt at fokus på repetisjonsaktivitetar kunne ha vore gunstig.

4.3.2.8 Andre erfaringar

5 elevar har svart på dette spørsmålet, og alle desse responsane omhandlar vanskegrada til programmering. Her er responsane:

1. *«Det var enklere enn jeg trodde, men det var frustrerende hvor lite som måtte til for å få feil.»*
2. *«Det var bare vanskelig å lage programmet riktig og legge til ting det var litt frustrerende fordi eg fikk det ikke til.»*
3. *«Nei, bare det at det var lettere enn jeg trodde, men tungvint i forhold til at det fint kunne gjøres på kalkulator»*
4. *«Hadde vansker med å forstå det meste»*
5. *«Hvor lett det er å faktisk lære noe.»*

Av desse tilbakemeldingane nemner 1, 3 og 5 at programmeringa/læringa var lett/lettare, mens 2 og 4 nemner at programmeringa var vanskeleg. Tilbakemeldingane 1 og 2 beskriv at programmering var frustrerande, mens tilbakemelding 3 beskriv programmering som tungvint. Beskriving 3 argumenterer for at programmeringa var tungvint med at oppgåvene kunne ha vore løyst ved bruk av kalkulator i staden for.

4.4 Forskingsintervju

Eg vil her presentere dei mest relevante funna frå forskingsintervjua som vart gjennomført med 4 elevar. Denne delen vert delt inn i 4 delkapittel (4.4.1 – 4.4.4), der eitt intervju vert gjennomgått per delkapittel. Elevane som har delteke i intervjua vert referert til som elev 1-4, og det kjønnsnøytrale pronomenet «hen» vert nytta for å anonymisere kjønnet til elevane. Direkte utdrag frå transkripsjonen av intervjua vert presentert i kursiv skrift, og i desse utdraga står *I* for «intervjuar» og *E* for «elev». Sidan transkripsjonane frå intervjua er direkte sitert vil dei bere preg av eit munnleg språk. Transkripsjonane vert berre endra på om det i intervjuet vart presisert namn som kan bidra til å svekke anonymiteten i prosjektet. Desse endringane vert då presenterte i klammer utan kursiv skrift.

4.4.1 Intervju 1

Elev 1 har tidlegare programmeringserfaring, i den forstand at hen har delteke på eit kodekurs på fritida ein gong. Eleven husker ikkje kva program som vart nytta til å kode på dette kurset. Motivasjonen for å arbeide med matematikk hentar denne eleven frå at hen ikkje vil stryke i faget.

På spørjeundersøking 1 svarer eleven at både ei typisk matematikkoppgåve og ei typisk programmeringsoppgåve bør ta rundt 5 minutt. Dette kom eleven fram til ved å tenke på kor lang tid hen bruker på dei enklaste oppgåvene innanfor matematikk og programmering. På spørsmål om ei rimeleg tidsmengd å bruke på ei vanskeleg oppgåve svarer hen 5 minutt på ei matematikkoppgåve som ikkje inkluderer programmering og 10 minutt på ei oppgåve som inkluderer programmering. Dette vert argumentert for på følgande måte:

I: *På første spørreundersøkelse er det eit tidsspørsmål til, og eg spør kva du tenker er ein rimelig tidsmengde på ei matteoppgåve før du gir opp, og innser at oppgåva er for vanskelig til å løysast. Du svarer 5 minutt der også. Fordi at?*

E: *Ehm…eg ikkje er ein person som bruker så masse energi og tid på noko eg føler eg ikkje er vits å bruke energi og tid på.*

I: *Så viss du ikkje får til ei oppgåva innen 5 minutt så går du gjerne vidare?*

E: *Ja.*

I: *Okei. Mens når eg spør om ei rimelig tidsmengde på ei programmeringsoppgåve då doblar du tida, og skriv at du kan sitte med den i 10 minutt.*

E: *Da er fordi at programmering er meir avansert, og då bruker eg lenger tid enn ei vanlig oppgåva.*

Eleven syns altså at programmering er meir avansert, og har av den grunn gitt ei høgare tidsmengd på spørsmålet om programmeringsoppgåva.

Elev 1 har kryssa av på at ein algoritme er ein gjentakande prosess som alltid gir same resultat på spørjeundersøking 1 fordi dette var det einaste svaralternativet som eleven forstod, men endra svaret til ein fullstendig beskriving for å løyse eit oppgåve på spørjeundersøking 2 fordi dette vart gjennomgått i matematikktimen. Når det kjem til å beskrive algoritmsik tenking

skreiv eleven noko som hen beskriv som «random» (*norsk*: tilfeldig) på fyrste spørjeundersøking, mens hen på spørjeundersøking 2 skriv at det er ein prosess til å tenke seg fram til eit svar. Dette vart grunngjeve på denne måten:

E: Ja, altså berre det at vi lærte jo at liksom at det er sånn steg for steg da. At du går liksom en sånn sammenhengende sånn…steg for å finne fram eit svar.

Elev 1 har kryssa av på 5 både på rangeringsspørsmålet om at hen aldri får bruk for matten hen lærar på skulen seinare i livet og på rangeringsspørsmålet om at matte er lite relevant for hen. Begge desse svara vert argumentert med at eleven ikkje føler at matematikkfaget er godt nok tilpassa dagleglivet. Eleven har kryssa av på 6, som indikerer at eleven er heilt einig, på utsegna om at hen aldri får bruk for programmeringa hen har lært på skulen. Argumentet eleven nyttar her er at reknestykka som dei nytta programmering til i klasserommet hadde vore mykje enklare å gjere på ein kalkulator. Dette er også grunnen til at eleven har kryssa av på 1 på utsegna om at programmering verker som eit nyttig verktøy.

Intervjudelen om digitale hjelpemiddel gjekk føre seg på følgande måte:

I: Dreiv de ikkje med noe excel eller geogebra eller noe?

E: Me dreiv med litt geogebra i niende klasse, og excel brukte vi en liten periode i tiende. Men det var aldri noe vi liksom…brukte.

I: Er du lei deg for det eller er du glad for da?

E: Glad for da, for geogebra er det verste eg har vore borti.

I: Okei, da er du ikkje fan av?

E: Nei.

I: Er da grunnen til at du har kryssa av på at du ikkje håper på mykje bruk av digitale hjelpemiddel på VGS?

E: Ja, eg har blitt traumatisert av geogebra.

I: Er det geogebra som går igjennom hovudet ditt når eg skriv digitale hjelpemiddel?

E: Mhm.

Eleven har på førehand altså ikkje eit godt forhold til bruk av digitale hjelpemiddel i matematikklasserommet.

På spørsmål om kva eleven syns var frustrerande med programmering svarer hen dette:

E: *Eg syns det var veldig vanskelig, og då mister eg motivasjonen og konsentrasjonen, og då lærer eg absolutt ingenting.*

Eleven uttrykker at hen syns programmering var enklare enn hen hadde sett for seg i utgangspunktet, men at heile opplegget virka unødvendig.

På spørsmål om korleis eleven opplever motivasjon når hen jobbar med programmering i forhold til penn og papir uttrykker eleven at hen får mest motivasjon av å jobbe med penn og papir fordi dette er verktøy som hen er vant til å bruke. Eleven uttrykker også at hen ser nytten av å bruke programmering i T-matematikken fordi dette matematikkfaget er vanskelegare, og har som mål å forberede elevar til vidare matematikk.

Dei siste to spørsmåla i intervjuet utspelte seg slik:

I: *Korleis vil du sei at arbeidet med programmering skil seg frå resten av matematikkundervisningen du har hatt tidlegare?*

E: *Eg syns det var veldig annerledes. Eg syns det er litt vanskelig å forstå seg på. Og, altså, alt annet føler eg at eg har ein liten form for… at det er viktig. Men det føler eg ikkje programmering hadde. Altså eg forstår kvifor vi lærer sånn volum og sånn, men eg skjønner ikkje kvifor vi skal bruke areal på ein PC.*

I: *(…) Vil du sei at algoritmisk tenking er meir eller mindre relevant for deg sammenlikna med resten av mattepensumet.*

E: *Eg vil sei det er mindre.*

I: *Fordi?*

E: *Fordi eg føler eg ikkje kjem til å få bruk for akkurat den typen algoritmisk tenking vidare i livet.*

4.4.2 Intervju 2

Elev 2 har tidlegare prøvd enkel robotprogrammering, men ikkje noko anna utover det. Eleven har alltid følt at hen beherskar matematikk godt, men valte P-matematikk over T-matematikk fordi hen syns matematikken på slutten av ungdomstrinnet var ganske vanskeleg. Eleven uttrykker at motivasjonen for å arbeide med matematikk kjem frå å få gode resultat, både for sin eigen del og for å tilfredsstille foreldras krav til eleven.

På tidsspørsmåla har eleven relatert dei til dei tidene hen ofte bruker på matematikkoppgåver, og har svart 30 sekund på ei typisk matematikkoppgåve og 4 minutt på ei typisk programmeringsoppgåve. Dette vert argumentert for på denne måten:

I: *Kva vurderingar gjorde du der?*

E: *Det var fordi det var mye sånn… selve sånn bare å skrive inn og de tingene der, det var jo ikke utfordrende. Eller å løse selve oppgaven. Men det var egentlig mer det at det var så… det er veldig detaljert program, og alt må jo være helt korrekt. Så då var det gjerne du må igjennom alt på nytt fleire ganger, rette på ting også er det ting du ikkje forstår, så ja…*

Eleven syns altså at programmeringsdelen var det utfordrande, og ikkje sjølve oppgåvene.

På spørsmål om kvifor eleven gjekk frå 4 minutt til 10 minutt på arbeid med ei vanskeleg matematikkoppgåve til ei vanskeleg programmeringsoppgåve svarer hen:

E: *(…) Også har eg gjerne…no når eg jobbet med det så har eg gjerne lyst å få det til, sant? Eg skjønnar ka eg driver med. Og eg har jo faktisk lyst til å faktisk skjønne ka eg skriver ned da. Så da har eg ofte brukt disse ekstra minuttene på å liksom få hjelp av læraren, og forstå programmet, og forstå ka eg gjør feil.*

Eleven hadde forventningar om at programmeringa skulle vere vanskelegare enn det viste seg å vere, fordi hen relaterte programmering til det som ofte blir vist i film.

Eleven har på begge spørsmåla om algoritme svart at det er ei fullstendig beskriving av framgangsmåten for å løyse ei oppgåve. På fyrste spørjeundersøking relaterte eleven dette til kva hen trudde algoritmisk tenking var, mens på andre spørjeundersøkinga svarte eleven det som hen hadde høyrt i undervisninga.

Eleven er ueinig i at hen aldri kjem til å få bruk for matematikken og programmeringa som vert undervist på skulen, fordi begge desse fagfelta er så integrert i vårt daglege liv. Dette er grunnen til at eleven også er einig i at programmering verker som eit nyttig verktøy.

Når det kjem til digitale hjelpemiddel har eleven erfaringar med Excel frå ungdomsskulen, men hen føretrekker å arbeide med penn og papir av denne grunnen:

E: Eg har liksom aldri mestret så mye som eg har på ark. For på ark kan eg liksom gjøre akkurat som eg vil. Å gjøre det på akkurat den måten eg vil. Men på PC må du liksom gjøre det på den måten du blir fortalt da. Og det PCen forstår. Så eg trur det er litt meir det.

Frå dette sitatet ser vi også kvifor eleven har kryssa av på 4 på utsegna om at arbeid med programmering var frustrerande; fleire reglar å forholda seg til når ein arbeider digitalt.

På spørsmåla om programmering var moro og/eller lærerikt seier eleven dette:

I: (...) Har du lyst til å kommentera nokon av dei? Lærerikt eller moro.

E: Altså eg lærte jo mye meir enn det…altså eg trodde egentlig frå begynnelsen at eg ikkje kom til å lære noe, og berre bli frustrert. Men eg lærte jo mye mer enn det eg hadde forventet. Så eg var jo fornøyd med det. Men moro… det var jo ikkje akkurat tema… det er ikkje det som eg syns…eg syns andre ting i matte er gøyare.

Eleven meiner at oppgåvene som vart gitt og gjennomgått i timane kunne vore meir aktuelle og ikkje så forenkla, fordi det hadde vore mykje kjappare og enklare å gjere dei oppgåvene som vart gitt ved hjelp av penn og papir.

Eleven uttrykker at sidan hen er vant til å arbeide på penn og papir så blir eleven motivert av arbeidsprosessen når dette er arbeidsmetoden hen nytter, mens det er sluttresultatet som motiverer mest når eleven arbeider med programmering. Det motiverer også eleven å arbeide med programmering fordi hen veit at fagfeltet blir meir og meir aktuelt.

På spørsmål om programmering i P-matematikk samanlikna med programmering i T-matematikk gjekk samtalen føre seg slik:

I: Okei. Så i T-matematikk er det direkte fokus på programmering. I P, som de har, så står ikkje programmering nevnt direkte. Det står algoritmisk tenking, også har [lærar] *valgt*

å gjennomføra programmering som del av algoritmisk tenking. Kvifor trur du at det i T-matte er større fokus på programmering?

E: *Fordi det er meir komplisert enn det vi, holdt på og si, det vi opplever det som. Eg tror det kan ligne litt mer på det som eg hadde forventet. Eg mener vi starter jo på null så du kan ikkje begynne med noe som er utfordrende med en gang. Men med T-matte elevar så er jo de der for å fokusere 100% på matten. Eller selvfølgelig, vi er jo der også for å lære matten, men de er jo der for å ta utfordringen da. Så då tenker eg at de kan gå mye bredere. Elevene vil forstå det og oppfatte det. Det er et litt meir selvstendig fag da. Eg tenker at her får vi jo hjelp til alt, mens der kan de jo gå litt dypere i det, med vanskelige tema.*

I: *Skulle du ønska at det var meir fokus på…altså at de skulle fortsetta litt meir med programmering no, eller er du glad for at no er me ferdige med den delen?*

E: *Eg føler jo egentlig at eg har lært det eg trenger å lære. I forhold til at viss eg hadde gått videre, då føler eg det hadde blitt meir komplisert, sant. Du trenger det til noe spesifikt.*

4.4.3 Intervju 3

Elev 3 har tidlegare erfaring med å programmere i Scratch og Python. Denne erfaringa kjem frå undervisninga i 8. og 9. klasse på ungdomsskulen, der hen følgde eit tilleggsfag i programmering. Utanom programmering har denne eleven ikkje andre erfaringar med bruk av digitale hjelpemiddel i undervisninga på ungdomsskulen. Eleven uttaler i intervjuet at hen ikkje lærte noko nytt i løpet av programmeringsundervisninga på VGS, men at undervisninga fungerte bra som repetisjon.

Eleven har på spørsmål om tidsmengder svart at ei typisk matematikkoppgåve og ei typisk programmeringsoppgåve tek same tid; om lag 5 minutt. Tidsmengdene på ei rimeleg tidmengd å nytte på ei oppgåve før ein går vidare er også lik for matematikk- og programmeringsoppgåva; 10 minutt. Desse tidsmengdene kjem frå elevens eigne erfaringar rundt arbeid med slike oppgåver.

Når ordet "algoritme" skulle definerast på fyrste spørjeundersøking svara eleven at det var ein gjentakande prosess som alltid ga same resultat. Dette var fordi hen relaterte spørsmålet til ein sorteringsalgoritme, som gjennomfører same handlingar inntil koden vert endra. Eleven endrar svaret sitt på spørjeundersøking 2 til ei fullstendig beskriving av framgangsmåten for å løyse ei oppgåve. Dette vart gjort fordi hen her tenkte at det finst andre typar algoritmar der den beskrivinga passa betre.

Eleven meiner at matematikk er ein sentral del av dagleglivet, og har derfor kryssa av på at hen er ueinig i at hen aldri kjem til å få bruk for matematikken som hen lærer på skulen seinare i livet og at matematikk er lite relevant.

Eleven har kryssa av på tre på utsegna om at hen aldri kjem til å bruke programmeringa hen lærer på skulen seinare i livet, og argumenterer for dette ved at hen ikkje har valt yrkesveg endå, og at programmering kan vere eit aktuelt val.

På spørsmål om programmering som verktøy blir følgande sagt:

I: *Programmering virker som eit nyttig verktøy, der har du kryssa av på fem. Noe du vil tilføye der?*

E: *For mesteparten er eg jo einig i at det kan vere eit godt hjelpemiddel, men samtidig kan det være helt unødvendig på andre områder. Så det var derfor eg tok på fem.*

Eleven svarer at arbeidet med programmering ikkje var frustrerande på VGS, men når hen tenker tilbake til undervisninga på ungdomsskulen så var det veldig frustrerande, ofte som eit resultat av «floker» i programma, som gjorde at dei ikkje fungerte som ønska.

Når det kjem til undervisninga så observerte eleven at det gjekk litt for fort for mange av dei andre elevane som aldri hadde programmert tidlegare, jamfør følgande intervjusituasjonar:

I: *Kunne undervisningen i programmering blitt lagt opp på ein annan måte, der har du svart nei, så du syns undervisningen i programmering var bra?*

E: *Den var helt grei. Det kunne kanskje ha vert litt forvirrende da, siden eg følte det gikk litt for kjapt. Fordi de fleste fikk ikkje helt med seg kvifor dette fungerer, eller kva vitsen er med det. Så eg følte…det eneste eg kunne påpeke var at det gikk litt for kjapt for de fleste.*

(…)

I: *(…) Gjekk undervisningen din seinare på ungdomsskulen sånn at fleire hang med etter kvart, eller?*

E: *Nei, vi begynte på scratch, så det var på en måte litt sånn….det var egentlig ganske lett å komme inn til det på den måten følte eg. Det var ikkje sånn rett på spyder, anaconda greiene med en gang.*

I: *Trur du at det kunne ha vore ein fordel her og, å starta i noko enklare, sånn som scratch?*

E: *Ja for eg mener scratch er ein veldig lett måte å vise programmering på. Det er på en måte sånn 3D-modell for programmering då, også kommer spyder og er 2D-modell. Så eg mener liksom at du kan se det gøye i programmering når du begynner på scratch med en gang, fordi det er så mange muligheter vi kan få til med scratch, og starte ut. Og eg føler då at gøy for de fleste viss de hadde klart å fått det til der.*

(…)

I: *Okei. Korleis føler du den overgangen frå scratch til python var? Var den tricky, eller?*

E: *For å starte ut så var den ikkje så, ka skal eg si…så utfordrende som eg trodde fordi det var jo egentlig bare å ta inn de prinsippene frå scratch inn i tekst. Så du måtte bare lære de ulike kodene, ka de betydde og ka de gjorde. Også bare sette de inn i et program. Det er egentlig alt som gjenstår etter at du har klart alt det visuelle i scratch.*

Når det kjem til motivasjon i matematikkfaget så uttaler eleven at det som motiverer hen er det å lære seg nye ting i faget. Når eleven blir spurd om motivasjon når hen arbeider med penn og papir i forhold til med programmering oppstår følgande intervjusituasjon:

I: *Okei, så føler du at du er meir eller mindre eller like motivert når du jobber i kodeprogram i forhold til når du jobber med penn og papir?*

E: *Eg føler meg meir motivert kan du si, fordi eg føler…eg liker responsen man får når man gjør noe feil, også som man får når man gjør noe riktig også. Så ja, eg liker den digitale måten litt betre.*

Eleven i dette intervjuet har også delteke i undervisning i matematikk 1T, og har som ein konsekvens av dette erfaringar frå begge matematikkemna på fyrsteåret på VGS. Samtalen rundt programmering i P-matematikk i forhold til T-matematikk utspelte seg slik:

I: *(…) Blei du overraska over at de skulle programmera i P-matten?*

E: *Ja, eg blei veldig overraska. Og grunnen til det er at eg trudde at P var veldig praktisk. Der det var bare «i dag skal vi igjennom denne grafen, og hvordan den blir relatert til det ekte livet. Og hvordan du kan bruke dette». Så eg var veldig overrasket. Trodde ikkje programmering skulle vere et så veldig fokusert tema her. Det virker som om det var viktig nok til at* [lærar] *bestemte å ha det med. Så det er eg faktisk litt takknemlig over.*

I: *(…) Vil du sei at algoritmisk tenking er meir eller mindre relevant enn resten av pensumet i matematikken?*

E: *Ehm, altså i forhold til det som vi har gått igjennom så langt, det eg ser, så mener eg at det ikkje er så særlig relevant, men det er viktig. Så…viss eg hadde vert i T så hadde eg sagt at det er veldig relevant, men no som eg er i P så ser eg at det ikkje er så relevant i forhold til resten av pensumet. Det vi går igjennom.*

4.4.4 Intervju 4

Elev 4 har høyrt om programmering tidlegare då hen fekk tilbod om å ta eit programmeringsfag på ungdomsskulen, noko denne eleven valte å ikkje gjere. Utanom dette har eleven ingen tidlegare erfaring med programmering.

Motivasjonen i matematikk hentar denne eleven frå følinga av meistring som oppstår når hen klarer å løyse vanskelegare og vanskelegare oppgåver i faget.

Av digitale hjelpemiddel har eleven nytta mykje GeoGebra og litt Excel på ungdomsskulen.

Eleven har svart at ei typisk matematikkoppgåve tek rundt 6 minutt, og i dette tidsrommet ligg det også ein del tid til å sjå over oppgåva, og rette opp eventuelle feil. Ei typisk programmeringsoppgåve meiner eleven tek 15 minutt, og dette knyt hen opp mot eigne erfaringar, og det faktumet at hen syns programmeringa var veldig vanskeleg. På ei slik oppgåve så tok det også lang tid å rette opp i eventuelle feil. På ei vanskeleg matematikkoppgåve kan

denne eleven bruke 25 minutt utan å gi opp, fordi hen vurderer seg sjølv som ein person som ikkje enkelt gir opp. Dette personelegdomsstrekket, kombinert med at eleven ikkje følte hen meistrar programmering, gjer at det tilsvarande svaret om ei programmeringsoppgåve blir høgt; 45 minutt.

Eleven syns programmeringsundervisninga var prega av litt for mange instruksjonar, og ikkje så mange forklaringar, noko som hen meiner er ugunstig. Eleven meiner også at oppgåvene som vart gitt kunne ha vore meir relevante enn det dei var.

På spørsmålet om algoritme kryssa eleven av på at det fyrst var ein gjentakande prosess som gir same resultat, deretter på at det er ei fullstendig beskriving av ein framgangsmåte for å løyse ei oppgåve. Eleven endra meining som eit resultat av det hen hadde funne ut på eiga hand og høyrt om i undervisninga.

På rangeringsspørsmåla svarer eleven at matematikk er relevant og at hen får bruk for det seinare fordi det er ein viktig del av dagleglivet. Programmeringa trur ikkje eleven at hen får bruk for, av den grunn at hen ikkje tenker å ta ei utdanning som inkluderer programmering. På spørsmålet om arbeidet med programmering var frustrerande oppstod følgande situasjon:

I: *(…). Kva som var frustrerande?*

E: *Når liksom…for å rette noe i programmet som var feil også prøve å spille det av igjen, så funket det enda ikkje. Så rettet eg det en gang til, så funket det endå ikkje. Så eg visste ikkje heilt kva som var feil, så det var litt frustrerende.*

I: *Meir frustrerande enn å jobba med penn og papir?*

E: *Ja.*

Om motivasjon ved bruk av penn og papir i forhold til programmering kjem følgande utsegner:

I: *(…) Når da kjem til å jobba med programmering og jobba med penn og papir sånn som de gjer til vanlig i matematikken. Kva syns du er mest motiverande av dei to tingene?*

E: *Penn og papir.*

I: *Fordi?*

E: *Det er mye enklere på penn og papir, du kan bare stryke vekk og skrive på nytt igjen. Og eg føler at siden du skriver det med hånden…det blir liksom en refleks i hånden da. Siden du skriver det og det og det. Men på PC er det jo taster. Så du får ikkje inn de same refleksene føler eg.*

I: *Er det noe som er motiverande med å jobba med programmering syns du?*

E: *Ja, det er jo motiverende når du får det til, og skjønner hele opplegget. Og kordan du skal rette feilene i programmet du lager.*

5 DISKUSJON

I dette kapittelet av oppgåva vil teorien bli kopla saman med analysen i ein diskusjon som har som hensikt å svare på problemstillinga som er utgangspunktet for studien:

Kva erfaringar har elevar i ein 1P-klasse med programmering før undervisninga om programmering tek til, og kva er deira forståing og deira syn knytt til programmering som verktøy til bruk i matematikk 1P etter denne undervisninga?

Denne oppgåva har også introdusert tre forskingsspørsmål som er utforma med utgangspunkt i problemstillinga:

1. Kva erfaringar har elevar i ein 1P-klasse med programmering før undervisninga tek til?
2. Korleis er elevane i ein 1P-klasse si forståing av programmering etter undervisninga?
3. Kva er elevane i ein 1P-klasse sitt syn på programmering etter undervisninga?

Føremålet med desse forskingsspørsmåla er å, gjennom å finne svar på dei, kunne gi eit svar på problemstillinga som oppgåva tek utgangspunkt i. Dette kapittelet er delt inn i tre hovuddelkapittel, der kvar av desse tek føre seg eit av forskingsspørsmåla. Diskusjonen frå desse delkapitla vil verte brukt til å gi eit svar på problemstillinga i kapittel 6.

5.1 Elevane si programmeringserfaring

Når ein lærar planlegg undervisning som har som mål å engasjere elevane må pensumet leggast opp slik at det bygger på elevane sine tidlegare kunnskapar og interesser (Wiske et al., 2005, side 7). I dette arbeidet er det ein føresetnad at lærarane kjenner til elevane sine kunnskapar og tidlegare erfaringar. Dette er grunnen til at det er aktuelt å kartlegge ein 1P-matematikklasse sine tidlegare erfaringar med bruk av programmering.

Frå tabell 4.3 ser ein at av dei totalt 19 elevane som gjennomførte den fyrste spørjeundersøkinga har 4 av elevane svart at dei har tidlegare programmeringserfaring. Dette svarer til 21,0% av elevane. Det kan likevel vise seg at dette talet ikkje nødvendigvis er heilt nøyaktig. Blant anna

vart det i intervjuet avdekka at elev 2 faktisk har tidlegare programmeringserfaring. Eleven har tidlegare prøvd det som eleven beskriv som «enkel robotprogrammering», men har til tross for dette kryssa «Nei» på spørsmålet om tidlegare programmeringserfaring. Erfaringa frå robotprogrammering kan komme til nytte når ein set i gang med programmering i ordinære kodeprogram, og kan av den grunn betraktast som relevant erfaring.

Så kvifor har elev 2 kryssa av på at hen ikkje har tidlegare programmeringserfaring? Svaret på dette spørsmålet finn ein kanskje seinare i intervjuet. Elev 2 uttaler at hen hadde forventningar om at programmeringa skulle vere vanskelegare enn det den var fordi eleven før undervisninga relaterte programmering til det som ofte blir vist på film. På film blir det ofte vist scener der såkalla «hackarar» programmerer på fleire dataskjermar samtidig, og kvar skjerm er full av fargerik tekst som ser ekstrem kompleks og avansert ut. Om elev 2 hadde denne førestillinga av programmering før undervisninga kan det tenkast at eleven kryssa av på «Nei» rett og slett fordi hen ikkje relaterte robotprogrammeringserfaringa som programmeringserfaring.

Elev 2 har på spørjeundersøking 2 skrive både på spørsmålet om undervisninga gjekk som forventa, og på spørsmålet om andre erfaringar, at programmering var enklare enn forventa. Dette kan ein nytte til å få ein indikasjon på om andre elevar har tenkt på same måte som denne eleven, og av same grunn har kryssa av på at dei ikkje har tidlegare programmeringserfaring, sjølv om dei i realiteten kanskje har det. På desse to spørsmåla er det fleire som har skrive at dei syns programmering var enklare enn dei i utgangspunktet hadde forventa, men ved nærmare inspeksjon kjem alle desse kommentarane frå elevar som har gjennomført intervju. Ein kan likevel tenke seg at det finst fleire som har drive med ei form for programmering tidlegare, men ikkje kryssa av for tidlegare programmeringserfaring fordi dei ikkje betraktar desse tidlegare erfaringane som programmering. Dette gjer at usikkerheita rundt dette spørsmålet aukar.

Elev 2 kan også ha kryssa av på inga programmeringserfaring grunna at hen gløymte vekk robotprogrammeringa som eleven har delteke i når spørjeundersøkinga vart gjennomført. Dette vil gjere at eleven tenker at hen ikkje har noko erfaring med å programmere.

To av elevane som kryssa av for tidlegare programmeringserfaring vart plukka ut til intervju; elev 1 og elev 3. Elev 1 har sin erfaring frå eit kodekurs som hen har delteke på tidlegare, mens elev 3 har si erfaring frå eit programmeringsfag som hen følgde i 2 år på ungdomsskulen. Begge desse elevane har relevant erfaring, og det er derfor bra at dei begge har kryssa av for at dei har tidlegare programmeringserfaring. Elev 4 har berre høyrt om programmering tidlegare, men har

ikkje tidlegare prøvd å programmere. Eleven har av den grunn kryssa av for at hen ikkje har tidlegare programmeringserfaring, noko som også kan betraktast som ei riktig vurdering.

5.2 Elevane si programmeringsforståing

Som presentert i delkapittel 2.1 så meiner Piaget at forståing skjer i assimilasjonsprosessar, der den som lærer endrar sine kognitive skjema knytt til spesifikke tema i møte med kognitive konfliktar. I denne oppgåva er fokuset elevane si forståing av programmering, og det er derfor naturleg å studere elevane sine kognitive skjema knytt til programmering. Dette vil vere hovudfokuset i dette delkapittelet. Delkapittel 5.2 er delt inn i fleire deler, der kvar del tek føre seg mindre delar knytt til elevane si forståing.

5.2.1 Undervisningsfokus

Av dei tre vanlegaste framgangsmåtane lista i delkapittel 2.5.4, som vert nytta for å implementere algoritmisk tenking i klasserommet, har læraren i denne klassen valt framgangsmåte 2, som fokuserer på konstruksjon av kode gjennom programmering (National Research Council (U.S.). Committee for the Workshops on Computational Thinking., 2010, side vii).

Det er tydeleg at læraren i denne klassen har designa eit pensum som integrerer programmeringa i undervisninga på ein slik måte at arbeidet fokuserer på å utvikle elevane si forståing av temaet der programmeringa vert nytta; bruk av matematiske formlar. Dette er ein av dei to måtane Wiske et al. (2005) skriv at teknologi kan implementerast i klasserommet for å auke elevane si forståing. Den andre måten går ut på å designa eit pensum som gjer det klart for elevane korleis bruk av teknologi gir ein fordel når det kjem til læringa. Det kjem fram, både i observasjonen, spørjeundersøkingane og intervjua, at mange av elevane saknar eit slik fokus i undervisninga.

Spesielt i økt 3 kjem det fram at elevane ikkje skjønner kvifor programmering skal nyttast på oppgåvene, sidan det jo hadde vore mykje meir effektivt å funne svara på andre måtar enn ved

bruk av programmering. På spørjeundersøking 2 ser ein eit sakn i fokus på nytteverdi av programmering frå eleven som har svart dette om programmeringsundervisninga:

> *«Ja senere når man skjønner det grunnleggende, burde det bli lagt mer vekt på hvordan man kan ha nytte av det»*

Elevane si oppfatning av nyttigheita til programmering vil bidra til å bestemme deira fornuftsgrunnlag for læring av programmering, noko som er avgjerande for korleis elevane si læring av programmering går føre seg (Mellin-Olsen og Hoel, 1984, side 22). Elevane si oppfatning av nyttigheita til programmering vert adressert direkte i dei fyrste to påstandane i rangeringsspørsmåla på spørjeundersøking 2 (delkapittel 4.3.2.6). På den fyrste påstanden – at eleven aldri seinare vil få bruk for programmeringa som vert undervist på skulen – ligg middelverdien på 3,93 av maksimalt 6. Dette vitnar om at mange av elevane ikkje ser føre seg at programmering er noko som dei vil ha nytte av seinare. Standardavviket på dette spørsmålet ligg på 1,53, noko som er høgt, og vitnar om at elevane har gitt spreidde svar på denne påstanden. Den andre påstanden distanserer seg i større grad frå eleven sjølv, og påstår at programmering verkar som eit nyttig verktøy. På dette spørsmålet ligg middelverdien på 3,46, noko som er svært nærme skalaen sitt midtpunkt på 3,50. Ein kan altså her sjå at elevane generelt sett er verken einige eller ueinige i om programmering verker som eit nyttig verktøy. Ein kan tenke seg at grunnen til dette er at elevane har opparbeida seg for lite erfaring til å kunne seie noko om nytteverdien til programmering som verktøy. Påstanden om nytteverdien til programmering har også det lågaste standardavvika av rangeringsspørsmåla på spørjeundersøking 2, noko som vitnar om ei relativt låg spreiing av svara.

På spørjeundersøking 2 vert det på spørsmålet om andre erfaringar skrive at programmering var tungvint, i og med at oppgåvene fint kunne ha vorte løyst ved hjelp av ein kalkulator. Denne utsegna høyrer til elev 1, som i sitt intervju innrømmer at hen har følt at alle føregåande tema i matematikkundervisninga, i større eller mindre grad, har vore viktige. Denne følelsen hadde ikkje eleven då det kom til bruk av programmering i matematikken. Også elev 2 uttaler i intervjuet at oppgåvene hadde vore mykje kjappare å løyse ved bruk av penn og papir samanlikna med programmering. Elev 3 meiner at programmering i matematikk 1P er av mindre relevans samanlikna med resten av pensumet i faget, og uttaler i tillegg i sitt intervju at hen observerte at dei fleste elevane i klassen ikkje forstod kva poenget med mykje av programmeringa som vart gjennomgått var.

Desse utdraga frå datamaterialet tyder på at det kan ha vore for lite fokus på kvifor programmeringa vart tatt i bruk i klasserommet, og kvifor ein nytta programmering til å løyse relativt enkle aritmetiske oppgåver. Mange av fordelane med å nytte programmering har blitt lista opp i delkapittel 2.5.3, men problemet med fordelane som er gitt i dette delkapittelet er at det ofte krevst eit større datamateriale enn det elevane arbeider med, eller meir kunnskap om programmering, for at dei skal gjere seg gjeldande. Ein kan derfor tenke seg at eit fokus på ein større heilskap, i den forstand at den enklaste programmeringa må lærast for å kunne vere i stand til å bruke programmering på ein fordelaktig måte, kunne ha vore gunstig i denne undervisninga. Dette kunne ha auka elevane si forståing av kvifor dei i nybegynnarfasen av programmeringsundervisninga treng å nytte programmering til å løyse oppgåver som kunne ha vorte løyst meir effektivt på andre måtar. Det kan også tenkast at det å vise elevane eit program som har praktisk nytte før sjølve undervisninga tek til kunne ha bidratt til å motivere elevane til å lære seg å programmere, samt bidratt til å bygge opp under elevane si tru frå delkapittel 4.3.1.7, om at programmering blir eit meir og meir relevant fagfelt.

Både i intervjua og på spørjeundersøking 2 så sett elevane fokus på progresjonen i undervisninga. Læraren nyttar det som Jerome Bruner beskriv som stillasbygging (Hyland, 2009, side 118) i si undervisning, der grada av kontroll og hjelp frå undervisaren minkar etter kvart som pensumet beveger seg vidare. Enkelte elevar kommenterer at dette skjer i eit for høgt tempo. Tilbakemeldingane 1 og 8 i delkapittel 4.3.2.7 kommenterer at det var mykje informasjon å ta inn over seg i undervisningssekvensane. Elevane som har gitt desse tilbakemeldingane søker også videoar knytt til lærestoffet, sannsynlegvis for å kunne gjennomgå pensumet i eit saktare tempo og/eller for å ha mogelegheita til å repetere fagstoffet. Det kan også verke som at eleven som har gitt tilbakemelding 9 syntest at progresjonen var litt for kjapp, då denne eleven sakna meir repetisjon i undervisninga. Elev 3 kommenterer i sitt intervju at hen registrerte at dei fleste elevane ikkje fekk med seg korleis tinga som vart gjennomgått fungerte, og trur av den grunn at det hadde vore ein fordel for forståinga til elevane om undervisninga hadde gått litt saktare fram.

5.2.2 Programmeringsoppgåver

Dei kognitive konfliktane (kapittel 2.1) som bidreg til utvikling av elevane sin kunnskap støyter dei i stor grad på i oppgåvearbeidet knytt til programmering. Som beskrive i delkapittel 4.2 så

er oppgåvearbeid ein sentral del av matematikktimane. Ei god kjelde til kognitive konfliktar vil vere problemløysingsoppgåver, i dette tilfellet matematiske problemløysingsoppgåver. Alan Schoenfeld sine to krav til problemløysingsoppgåver vart skrive om i kapittel 2.5.2, og dei er at (Schoenfeld, 1989b, side 87-88):

(a) eleven er interessert og engasjert i oppgåva, samt interessert i å finne ei løysing på den
(b) eleven ikkje har ein utlagt framgangsmåte/algoritme for å løyse oppgåva

Frå observasjonen kan det fastslåast at krav (b) på dei fleste oppgåvene vart tilfredsstilt. Oppgåvene som elevane arbeidde med omhandla det programmeringspensumet som var blitt presentert til dei, men gav ingen detaljerte forklaringar på korleis oppgåvene skulle løysast. Dette kan verifiserast ved å studere utvalet av oppgåver gitt til elevane frå figur 4.1.

Eleven som har gitt tilbakemelding 6 i delkapittel 4.3.2.7 ønsker at oppgåvene gitt til elevane skal likne meir på dei oppgåvene som læraren gjennomgår i plenum, og legg i dette eit ønske om ein klarare framgangsmåte eller algoritme som eleven kan følgje for å finne løysingar på oppgåvene. Det kan diskuterast kvifor læraren har valt å ikkje gi elevane oppgåver som kan bli løyst nøyaktig på same måte som dei læraren har vist, men ein av forklaringane til dette kan vere at læraren føler at det bryt med krav (b) frå Schoenfeld sin definisjon av eit matematisk problem. Om læraren berre hadde gitt oppgåver som kan løysast på same måte som den læraren viser til ville elevane indirekte ha mottatt ein framgangsmåte/algoritme for å løyse oppgåvene. Om dette hadde vore tilfelle ville ikkje oppgåvene lenger ha vorte rekna som matematiske problem, i følgje Schoenfeld sin definisjon. Også NCTM si utsegn knytt til effektiv matematikkundervisning ved bruk av oppgåver som fremmer matematisk resonnering og problemløysing frå delkapittel 2.5.2 ville ikkje ha gjort seg gjeldande, då den matematiske resonneringa og problemløysinga forsvinn i eit miljø der elevane har ein gitt framgangsmåte for å løyse oppgåvene dei blir gitt. Det at elevane si problemløysing fell vekk ved bruk av oppgåver der elevane følger ein gitt framgangsmåte vert også forsterka om ein ser på Solvang (1986, side 137) sin definisjon av problemløysing frå delkapittel 2.5.2, som fokuserer på elevane sin søken etter handlingar som må føretakast for å løyse eit problem. Kanskje den viktigaste grunnen til at læraren har gitt elevane oppgåver der dei ikkje har ein framgangsmåte for å løyse dei er at hen vil fremje den algoritmiske tenkinga hos elevane. Fraillon et al. (2018, side 27) sin definisjon av CT, som den algoritmiske tenkinga i den norske skulen er eit spegelbilete av, legg vekt på det å utvikle algoritmiske løysingar for problema som elevane blir gitt. Altså er det ein føresetnad at elevane sjølv tenker seg fram, nyttar og evaluerer ein

algoritme for å løyse oppgåvene dei vert gitt for at deira algoritmiske tenking skal utviklast. Det at elevane arbeider på kreative måtar for å finne algoritmar for å løyse oppgåvene vil kunne bidra til konstruksjon av forståing (Newton, 2012, side 16).

Krav (a) frå Schoenfeld sin definisjon av eit matematisk problem krev at elevane som arbeider med ei slik type oppgåve er engasjerte og interesserte i oppgåva. Arbeid med oppgåver som verkar engasjerande og interessante for elevane vil kunne freiste dei til å tre inn i eit undersøkingslandskap (beskrive i delkapittel 2.5.2) som inviterer til utforsking i oppgåvearbeidet. Det kan også føre til utviklinga av ein indre motivasjon for matematikk, noko som kan verke fordelaktig for elevane si forståing i faget. Interesse i arbeidet er også ein av komponentane som gir elevane sitt fornuftsgrunnlag, som diskutert i kapittel 2.2.2. Ei auka interesse for fagstoffet vil drive elevane mot eit sosialt fornuftsgrunnlag, noko som kan bidra til å auke elevane si forståing av lærestoffet.

Tilbakemeldingane 4 og 7 frå delkapittel 4.3.2.7 nemner oppgåvene som vart nytta som del av undervisninga. Eleven som har gitt tilbakemelding 4 meiner at oppgåvene kunne ha vore betre relatert til spørsmål ein vil stå ovanfor i framtida, mens eleven som har gitt tilbakemelding 7 meiner at oppgåvene kunne ha vore kjekkare. Begge desse tilbakemeldingane har med elevane si interesse og engasjement i oppgåveløysinga å gjere.

Argumentet om at oppgåvene med fordel kunne ha vore meir relevante for elevane kjem også fram under intervju av elev 2 og elev 4. I Fraillon et al. (2018, side 27) sin definisjon av CT vert det sett fokus på at CT vert nytta på problem henta frå den verkelege verda (eng: *real-world problems*). I dette ligg det at problema som skal løysast ved bruk av den algoritmiske tenkinga er problem som er relevante for elevane sine daglege liv, i og med at dei relaterer seg til verkelegheita. Oppgåvene som læraren ofte nytta til å demonstrere programmeringa omhandla areal og omkrins av geometriske objekt, og fleire av slike oppgåver vart også gitt til elevane under oppgåvearbeidet. To eksempel på slike oppgåver er oppgåve 2 og oppgåve 4 frå figur 4.1.

Sidan dei geometriske objekta som er utgangspunktet for oppgåve 2 og oppgåve 4 i figur 4.1 ikkje vert relatert til verkelegheita kan det verke som om nokon av elevane klassifiserer desse oppgåvene som lite relevante, noko som kan gjere at oppgåvene ikkje fangar interessa og engasjementet hos desse elevane. Dette kan i neste omgang svekke elevane si grad av problemløysing, som igjen minkar graden av kognitive konfliktar som fører til forståing.

Likevel er ikkje alle oppgåvene like abstrakte, då figur 4.1 viser at oppgåve 1 og oppgåve 3 hentar relevans frå den verkelege verda. Desse oppgåvene tek føre seg omgrepa tid, strekning, fart og temperatur, som alle relaterer seg til den verkelege verda. Det må også seiast at det er eit fåtal av elevane som har kommentert at oppgåvene med fordel kunne ha vore tettare knytt til den verkelege verda, noko som tyder på at dette ikkje har vore eit problem for mange.

Eleven som har gitt tilbakemelding 7 syns at oppgåvene kunne ha vore «gøyare». Her er det vanskeleg å vite kva eleven spesifikt legg i dette ordet, men det kan vere naturleg å anta at det er meint at oppgåvene i for liten grad byggjer på eleven sine interesser til å verke engasjerande. Det at arbeid verkar engasjerande skriv Wiske et al. (2005, side 7) er eit krav for at eleven skal kunne opparbeide seg ny kunnskap, og kunne utvikle evna til å nytte tidlegare kunnskap på nye måtar. Frå dette ser ein altså ei kopling mellom ei oppgåve si engasjerande effekt på ein elev og eleven si opparbeiding av kunnskap/forståing frå denne oppgåva. Tilbakemeldinga kan også tolkast dit at eleven føler at oppgåvene i liten grad gir glede i arbeidet med dei. Det at opplevinga av lærestoffet i for liten grad er prega av glede vil kunne bidra til å drive eleven mot eit instrumentelt fornuftsgrunnlag. Dette kan i ytste konsekvens føre til at læringa til eleven tek slutt (Mellin-Olsen og Hoel, 1984, side 43-44).

Tilbakemelding 7 er unik, det vil seie at berre denne eine eleven har gitt tilbakemeldinga om at oppgåvene ikkje var moro nok. Likevel kan ein få eit inntrykk av at elevane generelt sett ikkje syns arbeidet (deriblant oppgåvearbeidet) med programmering var særleg moro, då dette var det fleirvalsspørsmålet på spørjeundersøking 2 som hadde lågast middelverdi. Dette kan, som vi har sett, ha innverknad på elevane si forståing av programmering. Meir om fornøyelse i arbeidet i delkapittel 5.3.2.

5.2.3 Konseptuell forståing

Som presentert i kapittel 2.2 så går den konseptuelle forståinga ut på i kva grad ein elev har ei forståing av konsepta i det som vert arbeida med (Sierpinska, 1994, side 6-7). I dette delkapittelet vil eg ta føre meg dei omgrepa som i størst grad er vorte nytta i undervisninga, i spørjeundersøkingane, og i intervjua for å prøve og kartlegge elevane si konseptuelle forståing innanfor programmering.

I fleire av matematikktimane vart det observert ein del forvirring knytt til funksjonane print og input. Det elevane hadde særlege vanskar med var, som beskrive i kapittel 4.2, å tolke forskjellen på dei to funksjonane. Dette går nok mest på den prosedyremessige forståinga, sidan det var verkemåtane til dei to funksjonane som var vanskelege for å elevane å forstå, men det har nok litt med den konseptuelle forståinga å gjere også. «Print» og «input» er ikkje berre namna gitt til to programmeringsfunksjonar, men er omgrep som beskriv viktige konsept innanfor programmering. Noko av forvirringa som vart observert hos elevane i bruken av desse to funksjonane stammar nok frå det å ikkje klare å tolke print og input som konsept i ein programmeringskontekst.

Eit datamateriale som kan nyttast for å danne eit bilete av elevane si konseptuelle forståing innanfor programmering er svara på spørsmåla knytt til omgrepa algoritme og algoritmisk tenking på spørjeundersøkingane. Elevane blir bedt om å velje den beskrivinga dei føler passer best til ordet algoritme og blir bedt om å skrive kva dei trur ligg i omgrepet algoritmisk tenking på begge spørjeundersøkingane.

Elevane sine val av beskrivingar av omgrepet algoritme er nok det datamaterialet som i størst grad kan gi oss eit bilete på utviklinga av elevane si konseptuelle forståing innanfor programmering. Dette er fordi at omgrepet var eit sentralt tema i undervisninga som elevane deltok i. Som ein ser i kapittel 4.2 så fokuserte læraren mykje på dette omgrepet, og i oppgåvearbeidet vart elevane bedt om å skrive ein algoritme for kvar av programmeringsoppgåvene dei løyste.

I delkapittel 2.5.1 er omgrepet algoritme blitt definert av Solvang (1986, side 136). Frå denne definisjonen kan ein fastslå at det er svaralternativet «En fullstendig beskrivelse av framgangsmåten for å løse en oppgåve» som er rekna som det riktige alternativet på spørsmålet frå spørjeundersøkingane som ber elevane velje ei beskriving av algoritme. Ein ser av diagram 4.2 at 2 fleire elevar har valt dette svaralternativet på den andre spørjeundersøkinga samanlikna med den fyrste. Om ein samanliknar tabell 4.12 og tabell 4.4 ser ein at auken på dette svaralternativet har auka frå 50,0% til 64,3%, altså ei auke på 14,3%poeng. Tatt i betraktning at algoritmar var eit såpass sentralt tema i undervisninga er denne auken kanskje forventa.

Ein ser også at ingen elevar i gruppe 2 på spørjeundersøking 1 trudde at ein algoritme var «En oppskrift på hvordan et problem kan løses mest mulig nøyaktig», mens to elevar kryssa av for dette alternativet på spørjeundersøking 2. Dette svarer også til ei auke på 14,3%poeng. Denne

beskrivinga er ikkje heilt korrekt av primært to grunnar; ein algoritme treng ikkje nødvendigvis beskrive arbeidet med eit *problem* (sjå delkapittel 2.5.2 for definisjon), og ein algoritme treng ikkje å beskrive korleis ei oppgåve blir løyst *mest mogeleg nøyaktig*. Ein kan likevel fremme gode argument for at dette svaralternativet er det nest mest nøyaktige av dei fem; det knyt seg til oppgåveløysing og det fremmer ein framgangsmåte for å løyse ei oppgåve steg for steg. Om ein tek dette i betraktning så kan auken i svarprosenten på dette alternativet sjåast på som ei positiv utvikling av den konseptuelle forståinga til gruppa. Det vil heller ikkje vere feil å beskrive ein algoritme som ei oppskrift i staden for ei beskriving, og det bør også nemnast at læraren gjorde dette då hen beskreiv ein algoritme i økt 2. Dette kan også vere med på å forklare auken til dette svaralternativet.

Det nest mest populære svaralternativet på begge spørjeundersøkingane («En gjentagende prosess som alltid gir same resultat») droppar med 21,5%poeng frå spørjeundersøking 1 til spørjeundersøking 2. Dette er ei positiv utvikling, i og med at dette svaralternativet er dikta opp. Det er uvisst kvifor ein så stor del av elevane vel dette svaralternativet, men ei forklaring blir gitt av elev 3. Eleven valte dette svaralternativet på den fyrste spørjeundersøkinga fordi hen forbatt algoritme med ein spesifikk type sorteringsalgoritme i programmering. I intervjuet argumenterer hen for at ein sorteringsalgoritme vil gjenta same prosess inntil koden blir endra. Eleven endrar svaret sitt på spørjeundersøking 2 fordi hen tenker over at det finst fleire algoritmar, og generelt så meiner hen at dei betre blir kvalifisert som ei fullstendig beskriving av framgangsmåten for å løyse ei oppgåve.

Tre av elevane som er valt ut til intervju har på fyrste spørjeundersøking svarte at ein algoritme er ein gjentakande prosess, og har endra dette svaret til ei fullstendig beskriving på den andre spørjeundersøkinga. Grunngjevinga til elev 3 blir gitt i avsnittet ovanfor. Grunngjevingane til elev 1 og elev 4 er svært like; på fyrste spørjeundersøking gjetta dei, mens på andre spørjeundersøking svarte dei det som dei hadde blitt undervist at ein algoritme var. Elev 1 legg også til at hen tippa på spørjeundersøking 1 på ein gjentakande prosess fordi det var det einaste svaralternativet som eleven forstod. Elev 2 svara at hen trudde ein algoritme var ei fullstendig beskriving på begge spørjeundersøkingane. Eleven relaterte på fyrste spørjeundersøking algoritme til det hen trudde låg i det å tenke algoritmisk, mens eleven på andre spørjeundersøking svara det som var blitt undervist av lærar.

Alle elevane i intervjuutvalet viser eksempel på assimilasjon av sine kognitive skjema knytt til algoritmisk tenking. Assimilasjonen kjem som eit resultat av elevane si utvikling av den

konseptuelle forståinga si av algoritmar, som igjen er eit resultat av dei kognitive konfliktane som elevane har støytt på i undervisninga. Tre av elevane fastslår at dei fyrst og fremst veit kva ein algoritme er grunna forklaringar gitt i undervisninga, mens elev 3 har komme fram til dette ved å tenke breiare enn hen har gjort tidlegare.

Elevane sine beskrivingar av algoritmsik tenking vil i liten grad gi eit inntrykk av utviklinga av elevane si konseptuelle forståing innanfor programmering, då sjølve omgrepet ikkje er eit fokus i undervisninga. Elevane arbeider med algoritmisk tenking, men læraren nemner ikkje omgrepet for elevane, så om dei ikkje har høyrt dette omgrepet i andre samanhengar så er deira einaste møte med omgrepet frå informasjonsskrivet utdelt i forkant av gjennomføringa av masterprosjektet (vedlegg 2) og i spørjeundersøkingane og intervjua. Det kan likevel vere interessant å studere kva elevane legg i dette omgrepet, då det openbart knyt seg til omgrepet algoritme, som har vore eit svært sentralt tema i undervisninga.

Elevane gir generelt sett kortare og mindre usikre svar på spørsmålet om algoritmisk tenking på spørjeundersøking 2 samanlikna med spørjeundersøking 1, og i tillegg gir færre elevar tomme svar på spørsmålet. Dette kan tyde på at elevane føler seg sikrare kring algoritmar og algoritmisk tenking, kanskje delvis som eit resultat av ei auka konseptuell forståing. Det ein også ser på svara på spørjeundersøking 2 er at fleire av dei er tett knytt opp mot algoritmar, til dømes beskrivingane 2 og 7. Ein kan også tenke seg at beskriving 1 er gitt av ein elev som trur ein algoritme er ein gjentakande prosess, og knyt på denne måten den algoritmisk tenkinga opp mot det eleven meiner ein algoritme er. Beskriving 1 frå spørjeundersøking 1 knyt også algoritmar og algoritmisk tenking saman, men det er gjort på ein meir usikker og upresis måte samanlikna med beskrivingane i spørjeundersøking 2. Frå dette kan ein tenke seg at elevane i større grad ser at det er ei kopling mellom algoritmar og algoritmisk tenking etter undervisninga, sjølv om kjennskap til, og evna til å beskrive, omgrepet er låg.

5.2.4 Prosedyremessig forståing

Prosedyremessig forståing av programmering handlar om i kva grad elevane har forståing kring prosedyrane som vert nytta når ein programmerer. Ei svakheit i dette prosjektet er at det i for liten grad i datainnsamlinga er vorte fokusert på forskjellen mellom elevane si konseptuelle og prosedyremessige forståing (sjå delkapittel 5.4.2). Dette gjer at det spesielt manglar materiale til å direkte måle den prosedyremessige forståinga til elevane. I dette delkapittelet vil det bli

studert og diskutert punkt i datamaterialet som kan tenkast å vere kopla til elevane si prosedyremessige forståing av programmering.

Som diskutert tidlegare så kan elevane sine observerte vanskar knytt til bruk av print- og input-funksjonen til dels vitne om ei svak konseptuell forståing, men frå observasjonen kan det og slåast fast at denne forvirringa i tillegg har med ei svekka prosedyremessig forståing å gjere. Som beskrive i kapittel 4.2 så var det ein del elevar som sleit med å forstå forskjellen på desse to elementære programmeringsfunksjonane sine verkemåtar. Det vart også observert situasjonar der elevane får assistanse hos lærer, og innser at dei har nytta feil funksjon. Elevane dette omhandlar vert så observert å byte ut funksjonen som dei har nytta, med den funksjonen som læraren har foreslått at dei skal prøve, utan å be om ei vidare forklaring på kvifor det er sånn. Elevane dette omhandlar viser symptom på eit instrumentelt fornuftsgrunnlag for læring av programmering. Elevane som ukritisk byt ut print- eller input-funksjonen med den andre, og deretter går vidare til neste oppgåve, viser tendensar til regeloppfatning når det kjem til forståinga kring bruken av desse to funksjonane, i motsetnad til ei strukturoppfatning, som ville ha vore eit symptom på eit sosialt fornuftsgrunnlag for læring. Det vert også klart frå desse observasjonane at elevane det her er snakk om ikkje arbeider i Skovsmose (1998) sitt undersøkingslandskap, då arbeid i eit slik landskap ville ha fremma spørsmål og utforsking knytt til det som vert arbeida med. I delkapittel 2.5.2 ser ein at Santos-Trigo (Santos-Trigo, 2019, side 67) slår fast at utforsking, analysering og forklaring av dei dilemma som oppstår i det matematiske arbeidet er viktig for læringa. Når elevane søker assistanse frå lærar, og ukritisk følgjer lærarens råd, vil denne viktige delen av arbeidet forsvinne, dei kognitive konfliktane vert færre og svakare, noko som fører til mindre grad av forståing.

Det at elevane i mindre grad søker læraren si hjelp etter at dei har fått ei innføring i å rette feil i programma kan tyde på fleire ting. Elevane sin oppførsel kan tyde på ei auka forståing, eller i det minste eit ønske om å oppnå auka forståing, av feilmeldingar i programmeringa. Sitatet frå Wiske et al. (2005, side 5) i delkapittel 2.2.3 seier at forståing handlar om å ha evna til å tenke og handle fleksibelt med det ein veit. Sjølv om læraren berre viser nokre få eksempel på feil og feilmeldingar som elevane kan støyte på i arbeidet så viser elevane fleksibilitet ved at dei nytter grunnteknikkane læraren har demonstrert for å prøve og tolke, og rette opp i, sine eigne feil. Dette kan gi uttrykk for ein viss fleksibilitet hos elevane når det kjem til feil i programma. Det må likevel vurderast om denne oppførselen frå elevane ikkje gir indikasjonar på ei auka forståing, men heller kjem som eit resultat av ei sosial norm som er vorte etablert i klasserommet. Når oppgåvearbeidet startar opp, etter at læraren har gjennomgått korleis elevane

sjølv kan rette opp feila sine, kan det hende at elevane tenker at det er forventa at dei i mindre grad søker hjelp frå læraren, og at denne forventninga gir grunnlaget for ei sosial norm om å ikkje spørje om læraren si hjelp med å tolke og rette opp i feila som oppstår i programma, uavhengig om elevane forstår dei eller ikkje.

Elevane rangerte på den andre spørjeundersøkinga påstanden om at arbeidet med programmering var lærerikt. Sjølv om denne påstanden ikkje spesifiserer kva element ved programmeringa elevane her skal rangere vil nok den prosedyremessige forståinga til elevane vere ein stor del av det elevane oppfattar skal rangerast i denne påstanden. Middelverdien til elevgruppa er 2,92, og standardavviket er 1,39. Samanlikna med resten av verdiane i tabell 4.13 er dette låge verdiar, noko som gjer uttrykk for at elevane generelt sett er delvis ueinig i at programmeringsarbeidet var lærerikt. Nokre av tilbakemeldingane frå spørjeundersøking 2 støtter opp om dette. Blant anna står det i beskriving 5 i delkapittel 4.3.2.7 at denne eleven ikkje skjønte noko av programmeringa. I beskriving 6 i det same delkapittelet har ein elev gitt uttrykk for at hen ikkje skjønte så mykje av programmeringa. Beskriving 1, 8 og 9 ser behovet for meir repetisjon og/eller fleire læringsressursar som elevane kan lene seg på, noko som kan tyde på at elevane meiner at deira forståing kunne ha vore betre om dei fekk desse behova oppfylt. I delkapittel 4.3.2.8 står det i tilbakemelding 4 at eleven hadde vanskar med å forstå det meste, noko som også støtter opp om den relativt låge middelverdien til påstanden om lærerikt arbeid.

Frå intervjua kan ein også hente ut element som kan knytast opp mot den prosedyremessige forståinga av programmering. Elev 1 seier følgande om programmeringsarbeidet:

> *«(...) Eg syns det er litt vanskelig å forstå seg på (...)»*

Eleven verkar å ha vanskar med å forstå programmering. Delar av forklaringa bak eleven sine forståingsvanskar kan ein finne ved å sjå på eleven sitt fornuftsgrunnlag. Elev 1 sine tidlegare dårlege erfaringar med bruk av digitale hjelpemiddel i matematikken kan tenkast å minke eleven si interesse for programmering, eleven si oppfatning av nyttigheita av programmering (noko som andre utsegner frå intervju 1 heilt klart også gir evidens for), eleven si oppleving av glede frå programmering og eleven si nysgjerrigheit knytt til programmering. Alle desse svekka komponentane vil kunne føre til at eleven får eit instrumentelt fornuftsgrunnlag for læring av programmering. Det vert endå tydelegare at elev 1 lener mot eit instrumentelt fornuftsgrunnlag for læring når ein også tek omsyn til eleven si utsegn om at motivasjonen for å arbeide med matematikk hentar eleven frå det å unngå og få ein strykkarakter i faget. Eit instrumentelt

fornuftsgrunnlag kan føre til ei ufullstendig forståing av lærestoffet, og kan også føre til at læringa til eleven stoppar heilt opp (Mellin-Olsen og Hoel, 1984, side 43-44).

Elev 2 viser i intervjuet sitt symptom for eit sosialt fornuftsgrunnlag for læring av programmering. Når programmeringsoppgåvene vert diskuterte seier eleven at hen bruker ekstra tid til å danne seg ei forståing av programmet som er vorte konstruert for å løyse ei oppgåve. Dette vitnar om at eleven syns arbeidet er såpass viktig at hen har eit ønske om å opparbeide seg ei strukturoppfatning av programmeringsreglane nytta i oppgåvene. Dette vil kunne føre til ei djupare forståing av programmeringa. Elev 2 uttaler også at lærestoffet var meir lærerikt enn hen fyrst hadde trudd det skulle vere.

5.3 Elevane sitt programmeringssyn

Det tredje forskingsspørsmålet tek føre seg elevane sitt syn på programmering, og dette vil bli diskutert i dette kapittelet. For å kartlegge elevane sitt syn på programmering vil eg ta utgangspunkt i dei tre dimensjonane som Sthephani et al. (2019) beskriv i teksten introdusert i kapittel 2.3. Ved å forsøke å kartlegge dimensjonane (kognitiv, emosjonell og motivasjon) basert på det tilgjengelege datamaterialet vil ein kunne danne seg eit bilete av elevane sine syn på programmeringa som har vorte nytta i matematikkundervisninga.

5.3.1 Den kognitive dimensjonen

Den kognitive dimensjonen omhandlar elevane si sjølvsikkerheit, vurdering av eigen kompetanse og vanskar innanfor programmering.

Sjølvsikkerheit vart i kapittel 2.3 definert som samlinga av følelsar som relaterer til trua på seg sjølv, og eigen effektivitet når det kjem til å prestere i ein sosial omgjevnad (Sthephani et al., 2019, side 2). Sjølv om programmering fyrst vart introdusert for elevane etter at spørjeundersøking 1 vart gjennomført kan ein plukke opp enkelte interessante moment frå denne undersøkinga som relaterer seg til elevane si tru på seg sjølv når det kjem til å lære seg programmering. På spørsmålet om elevane trur at programmering vil vere nyttig i

matematikkundervisninga (delkapittel 4.3.1.7) er det to elevar utan tidlegare programmeringserfaring som gir svar som tyder på at dei ikkje trur dei kjem til å meistre programmeringa. Desse svara gir uttrykk for lite tru på eigne ferdigheiter og eigne evner til å lære seg å programmere. Elev 2 gir også uttrykk for opphaveleg låg sjølvtillit når det kjem til å lære seg programmering, då eleven i intervju uttalar følgande:

> *«(…) eg trodde egentlig fra begynnelsen at eg ikkje kom til å lære noe, og berre bli frustrert (…)»*

Elev 2 sin låge sjølvtillit til å lære seg programmering kan også delvis stamme frå det at eleven trudde programmering skulle vere mykje vanskelegare enn det den viste seg å vere.

I løpet av observasjonen vart det observert at mange elevar søkte hjelp frå lærar med ein gong noko gjekk feil i programmet som dei prøvde å konstruere. Samtidig som dette gir uttrykk for ei låg forståing så gir det også uttrykk for låg sjølvsikkerheit knytt til programmering. Elevane har kanskje ikkje trua på at dei sjølv klarer å rette opp i feila sine, derav den kjappe søken etter assistanse. Likevel har vi datamateriale som tyder på at søkinga etter hjelp ikkje nødvendigvis stammar frå låg grad av sjølvsikkerheit, men avgrensa erfaring med programmering. Dette kan ein seie sidan det vart observert ein markant minke i elevane si søken etter hjelp etter at dei hadde blitt undervist i korleis dei skulle tolke feila dei gjer i kodeprogramma, og rette opp i desse.

Vanskar innanfor programmering og vurdering av eigen kompetanse er kopla tett saman. Dette ser ein blant anna på definisjonen av vanskar i kapittel 2.3. Vanskar innanfor programmering relaterer seg til trua rundt eigen kompetanse og suksess. Kompetanse blir her innlemma i definisjonen av vanskar innanfor programmering, og eg vel derfor å sjå på desse to komponentane samla.

I både spørjeundersøking 2 og intervjua kjem det fram at det som fyrst og fremst er ei overordna vanske for elevane er å lage program som fungerer som dei ønsker, utan å bryte syntaksen som ligg til grunn for at koden dei skriv skal kunne fungere. Dette er som sagt ein overordna vanske, som stammar frå mindre vanskar som elevane har når dei skriv programma sine. Vanskane er tett knytt til elevane si forståing, både konseptuelt og prosedyremessig, så mykje av det datamaterialet som gjorde seg gjeldande i kapittel 5.2 vil også vere aktuelt å studere her. Fyrst og fremst vil dei meir spesifikke vanskane til elevane komme klarare til syne ved å nok ein gong studere observasjonen som vart gjort i undervisningstimane.

Under observasjonen blir det observert at elevane har fleire vanskar når det kjem til å programmere. Det å tolke forskjellen på ulike funksjonar var noko som elevane hadde vanskar med, som diskutert kapittel 5.2. Elevane hadde også vanskar med å organisere rekkefølga til kodelinjene sine for å danne eit program som fungerte som dei ønska. I dei fyrste øktene vart det tydeleg at elevane hadde svært låg tru på eigen kompetanse innanfor programmering, då standardhandlinga til elevar som støytte på feil eller andre uføresette hindringar i programmeringsarbeidet var å be om hjelp frå lærar. Som diskutert tidlegare er dette også handlingar som kan tolkast som manglande sjølvtillit, men mest sannsynleg så stammar denne handlinga frå fleire ulike faktorar, der både elevane sin sjølvtillit og deira vurdering av eigen kompetanse er faktorar som spelar inn.

Elevane si vurdering av eigne kompetansar vil nok variere basert på kva utgangspunkt elevane har. Det er naturleg å tru at dei 2 elevane i gruppe 2 som svarer at dei har tidlegare programmeringserfaring stiller med eit lite fortrinn over dei elevane som svarer at dei ikkje har tidlegare erfaring med å programmere. Spørsmålet i denne samanhengen er om vurderinga av elevane sin eigen kompetanse innan programmering vil synast å stige i tråd med aukande programmeringserfaring.

Elev 3, som verker til å vere den eleven med mest tidlegare programmeringserfaring, har vurdert det slik at det vil vere rimeleg å arbeide med matematikkoppgåver og programmeringsoppgåver i like tidsmengder. Dette er uavhengig om oppgåvene er typisk små deloppgåver eller om oppgåvene er vanskelege. Frå intervjuet med denne eleven vert det avdekka at denne vurderinga er tatt basert på elevens eigne erfaringar med dei to ulike oppgåvetypane. Frå samanlikning av tidsspørsmåla på dei to spørjeundersøkingane ser ein at den generelle trenden frå elevgruppa er at dei estimerte tidsmengdene auka når programmering vert inkludert som eit verktøy for å løyse oppgåvene. Det at elev 3 ikkje har gjort dette kan vitne om at eleven vurderer sin kompetanse innafor programmering som høgare enn elevane som har mindre eller ingen tidlegare programmeringserfaring. Dette kan sjåast i direkte samanheng med vanskane som eleven har med å programmere. Elev 3 uttaler at alt som vart gjennomgått i løpet av undervisningssekvensane var kjend stoff for hen. Dette betyr at eleven i mindre grad støytte på vanskar i arbeidet med programmering, noko som igjen kan ha gitt eleven ei forsikring om at hen sine programmeringskompetansar er høge.

Elev 1 er den andre eleven som har oppgitt tidlegare programmeringserfaring. Til tross for dette ser ein ikkje like tendensar hos denne eleven som ein ser hos elev 3. Elev 1 har nemleg dobla

den tida hen meiner det bør ta å løyse ei programmeringsoppgåve samanlikna med ei matematikkoppgåve. Dette tyder ikkje på at denne eleven vurderer sin eigen kompetanse i programmering like høgt som elev 3, iallfall ikkje samanlikna med matematikk utan programmeringsverktøy. I intervjuet med elev 1 kjem det også fram at eleven har møtt på fleire vanskar i løpet av programmeringsarbeidet som hen føler har innverknad på forståinga i emnet. No skal det seiast at sjølv om både elev 1 og elev 3 har tidlegare programmeringserfaring så verker det som om erfaringa til elev 3 var mykje meir aktuell for denne undervisninga, då erfaringa delvis inkluderte programmering i same programmeringsspråk som vert nytta i undervisninga på VGS; Python. Erfaringa til elev 3 gjekk også over lengre tid, og det er kortare tid sidan elev 3 programmerte i forhold til elev 1. Dette vil høgst sannsynleg ha innverknad på elevane sine vanskar i arbeidet og vurderinga av deira programmeringskompetansar.

5.3.2 Den emosjonelle dimensjonen

Denne dimensjonen relaterer seg til elevane si fornøyelse av å jobbe med programmering i matematikkfaget. I delkapittel 2.3 vart fornøyelse definert som ein følelsesmessig respons som omfattar positive eller negative følelsar (Sthephani et al., 2019, side 2). Det vil her vere aktuelt å sjå på datamaterialet som fortel noko om elevane sine følelsar knytt til programmering. I spørjeundersøking 2 vert det gjort eit forsøk på å kartlegge i kva grad elevane syns arbeidet med programmering var frustrerande og moro, og desse to følelsane vart det også fokusert på i intervjua med utvalet av elevar. Desse følelsane vil vere hovudfokuset i dette delkapittelet, og vil kunne gi eit bilete av elevane sine emosjonelle reaksjonar knytt til arbeidet med programmering.

Middelverdien på påstanden om at arbeid med programmering var frustrerande (delkapittel 4.3.2.6) er 3,86. Dette er ein verdi som ligg over midtverdien på skalaen, noko som vil seie at dei fleste elevane har svart at dei er heilt eller delvis einige i at arbeidet med programmering var frustrerande. Standardavviket til denne påstanden er svært høg, noko som indikerer at elevane er delt i sine meiningar kring denne påstanden.

For å undersøke nærmare kva elevane spesifikt syns kan vere frustrerande i arbeidet med programmering går vi litt vidare i spørjeundersøking 2. I delkapittel 4.3.2.8 ser vi at tilbakemeldingane 1 og 2 fortel om frustrasjonen knytt til programmering. Tilbakemelding 1 høyrer til elev 2, og inneheld blant anna denne teksten:

> *«(…) det var frustrerende hvor lite som måtte til for å få feil.»*

Det kjem fram i intervjuet med elev 2 at hen føretrekk å arbeide med penn og papir over PC, fordi eleven liker å løyse oppgåver på sin eigen måte og utan reglar. På ein PC må ein føre alt korrekt for å få produsert eit svar, og det er dette elementet ved programmering som eleven syns er frustrerande. Det at elev 2 syns det er frustrerande å få feil når det blir arbeida i kodeprogram strid imot Sfard og Leron (1996) si vurdering av feil i arbeid med programmering. Dei argumenterer for at feil som oppstår i kodeprogram må sjåast på som del av arbeidet, og at dei er viktige og nødvendige for å få produsert eit fullgodt resultat (Sfard og Leron, 1996, side 192). I denne konteksten kan feil i programmeringsarbeidet verke motiverande, men det kan verke som om elev 2 ikkje er einig i framstillinga som Sfard og Leron gir.

I tilbakemelding 2 i delkapittel 4.3.2.8 står det følgande:

> *«Det var bare vanskelig å lage programmet riktig og legge til ting det var litt frustrerende fordi eg fikk det ikke til.»*

Denne tilbakemeldinga er gitt av elev 4, som i sitt intervju utdjupar at det å køyre eit program om att og om att fleire gonger utan å klare å rette opp i alle feila i programmet var det som sto for frustrasjonen i arbeidet. Dette er nok eit eksempel på ein elev som synast å ikkje sjå på feil og vidareutviklinga av program som noko motiverande, men heller frustrerande. Som eit resultat av at elev 3 er erfaren innanfor programmering så opplevde ikkje denne eleven noko frustrasjon i timane på VGS, men eleven gav uttrykk for at då hen fyrst vart introdusert for programmering på ungdomsskulen så opplevst det som eit frustrerande arbeid. Denne frustrasjonen beskriv eleven som å stamme frå «floker» i programma som gjorde at dei ikkje fungerte. I denne samanhengen kan nok desse flokene tolkast som ei samling av feil som gjer at programma ikkje køyrer som forventa.

Ein kan stilla seg spørsmålet om kvifor det verkar som om elevane sine erfaringar med programmering i så stor grad ser ut til å gå imot det som Sfard og Leron beskriv som ein av dei mest framståande fordelane ved programmering. På dette spørsmålet ligg det nok mange ulike svar, og eitt av desse kan vere at miljøet der datamaterialet er samla inn på i Sfard og Leron sin tekst skil seg frå miljøet der datamaterialet i denne studien vert samla inn. Som nemnd i delkapittel 1.2.2 så er forklaringane i teksten til Sfard og Leron gitt frå ein undervisar sitt ståpunkt, ikkje ved hjelp av innsamla datamateriale frå studentane. Dette svekker reliabilitet til argumenta i teksten.

Ein annan forskjell som kan nemnast er at teksten til Sfard og Leron tek føre seg to distinkte fag; matematikk og programmering. I denne studien går derimot alt programmeringsarbeidet føre seg i matematikkfaget, noko som kan gjere at elevane forventar at oppgåvene som dei blir gitt innanfor programmering skal kunne løysast i same tids- og arbeidsrammer som matematikkoppgåvene dei er vane med. Middelverdien til tidsmengda som elevane tenker er rimeleg å nytte på ei typisk matematikkoppgåve er 2 minutt og 49 sekund, noko som er 49 sekund høgare enn middelverdien i Alan Schoenfeld (1989a, side 345) sin studie, der han har stilt same spørsmål til ei større gruppe elevar i den vidaregåande skulen. Alan Schoenfeld meiner at denne korte tida kjem som eit resultat av elevane sine erfaringar med korte og konsise oppgåver i matematikkfaget, og det er naturleg å tenke at dette også er grunnen til at elevane i denne studien har gitt eit såpass kort tidsrom på ei typisk matematikkoppgåve. Ein kan vidare tenke seg at elevane førestiller seg at programmeringsoppgåvene skal kunne løysast innanfor det same tidsrommet, i og med at det er oppgåver gitt i matematikkfaget. Frå dette kan det verke plausibelt at hovudgrunnen til elevane sin frustrasjon i programmeringsarbeidet kjem som ein reaksjon på at det ikkje verkar mogeleg å løyse programmeringsoppgåvene i eit liknande tidsrom som det dei bruker på matematikkoppgåvene utan programmering.

Dette tidsaspektet ved løysing av oppgåver kan tenkast å vere relatert til dei sosiale og sosiomatematiske normene som er vorte etablert i denne klassen og i Sfard og Leron sine klasserom. Om det i 1P-klasserommet er vorte etablert ei norm som tilseier at det er forventa at oppgåvene som vert arbeida med skal kunne løysast raskt, og dette ikkje skjer, så kan brytinga av denne norma vere ei forklaring på kvifor frustrasjon oppstår i dette klasserommet. Om det i programmeringsklasserommet i Sfard og Leron sin tekst derimot har blitt etablert ei norm som etablerer ei forventing om at oppgåvene skal ta tid, og skal løysast som eit resultat av vidarebygging av feila ein støyter på undervegs, så er det mogeleg at feila som oppstår i arbeidet i større grad verkar motiverande for elevane.

Middelverdien til påstanden om at arbeid med programmering var moro er 2,71, den lågaste verdien på rangeringsspørsmåla frå spørjeundersøking 2. Denne låge verdien stadfestar at ein stor del av elevane, i større eller mindre grad, er ueinige i at arbeidet med programmering i matematikkfaget var moro. Om ein ser denne verdien i samanheng med verdien til påstanden om at arbeidet med programmering var frustrerande så gir det intuitiv meining at ein høg verdi på påstanden om frustrerande arbeid kan kunne gi ein tilsvarande låg verdi på påstanden om moro arbeid. Dette vil sjølvsagt ikkje følgje direkte, då ein kan tenke seg elevar som synst at arbeidet var både frustrerande og moro på same tid.

Sidan målinga av om elevane syntest programmering var moro eller ikkje er målt av eit rangeringsspørsmål er det berre eit fåtal av elevane som har kommentert noko om dette på nokon av fritekstspørsmåla. Ein av kommentarane gitt på spørsmålet om undervisningsforventingar i delkapittel 4.3.2.1 ser slik ut:

> *«Nei, eg trodde det skulle være bare kjedelig programmering, men det var faktisk gøy»*

Eleven som har gitt denne tilbakemeldinga har altså forventa at programmeringa ikkje skulle vere moro, men har tydelegvis vorte positivt overraska av å finne ut at hen faktisk syns det var moro. Ein annan tilbakemelding som kan knytast til i kva grad elevane syntest undervisninga var moro er denne, som er gitt i delkapittel 4.3.2.7:

> *«Hvis oppgavene hadde vært gøyere»*

Denne tilbakemeldinga er gitt som forslag til korleis undervisninga kunne ha vore lagt opp annleis for å styrke eleven sitt faglege utbytte, og omhandlar spesifikt oppgåvene som elevane arbeidde med.

Elev 2 samanliknar programmering med andre ting som er vorte gjennomgått i matematikkundervisninga tidlegare, og kommenterer i den forbindelse at det finst andre ting i matematikken som eleven syns er meir moro. Resten av elevane som er vorte intervjua føyer ikkje til noko av relevans som kan bidra til å belyse dette betre.

5.3.3 Dimensjonen knytt til motivasjon

I dette delkapittelet er motivasjon hovudtema, og det vil her bli diskutert i kva grad elevane synast å gi uttrykk for motivasjon i arbeidet med programmering.

Eickelmann (2019, side 54) lister opp tre ulike framgangsmåtar som ein lærar kan nytte når det kjem til å motivere elevane til å arbeide med algoritmisk tenking. Av desse tre framgangsmåtane nyttar læraren som er vorte observert i denne klassen til dels den fleirfaglege framgangsmåten og til dels informatikkframgangsmåten. Når læraren introduserer omgrepet algoritme så vert det presisert at dette omgrepet ikkje berre gjer seg relevant i matematikkfaget. Sjølv om eksempla som læraren gir er henta frå matematikkfaget, så blir ein algoritme beskrive som ei detaljert oppskrift for å løyse ei kva som helst oppgåve. Her har læraren nytta den

fleirfaglege framgangsmåten. Den fleirfaglege framgangsmåten blir også nytta i det fyrste oppgåvearbeidet, der elevane skal programmere enkle program som ikkje inneheld tal, men strengar. Læraren nemner ikkje direkte at den algoritmiske tenkinga er essensiell innanfor fagfeltet informatikk, men ein ser likevel at informatikkframgangsmåten vert nytta i måten læraren arbeider med den algoritmiske tenkinga i klasserommet. Programmering er verktøyet som står i fokus i dette arbeidet, og det blir då indirekte presisert for elevane kor viktig den algoritmiske tenkinga er innanfor informatikk.

Om elevane vert motivert av måten læraren presenterer den algoritmiske tenkinga på er eit anna spørsmål. Motivasjon var eitt av hovudtema som eg ønska å kartlegge i intervjua med elevane, så for å kartlegge elevane sin motivasjon knytt til programmering er det naturleg å sjå på datamaterialet frå intervjua.

Sidan programmeringa vert introdusert i matematikkundervisninga var det aktuelt å fyrst kartlegge kva som i utgangspunktet motiverer elevane til å arbeide i matematikkfaget. Elev 1 har det som kan kjennast igjen som ein kontrollert ytre motivasjon i matematikken. Eleven uttalar at det som motiverer hen til å arbeide med matematikk er å ikkje stryke i faget. Elev 1 uttalar at digitale verktøy er noko som hen ikkje er liker å bruke, og argumenterer for dette med dårlege erfaringar frå ungdomsskulen. Eleven seier også at motivasjonen i matematikk er høgare når eleven får arbeide med penn og papir samanlikna med programmering, fordi penn og papir er kjend for eleven. Programmering vart også oppfatta som veldig vanskeleg av denne eleven, og dette gjer også at motivasjonen minkar. Alt i alt kan ein seie at elev 1 sin motivasjon til å arbeide med programmering er svært låg, tatt i betraktning at elevens motivasjon i matematikkfaget er ytre kontrollert i utgangspunktet, og arbeid med eit nytt vanskeleg verktøy bidreg til å svekke denne motivasjonen ytterlegare.

Motivasjonen til elev 2 kan gjenskjennast som delvis ytre autonom og delvis ytre kontrollert. Motivasjonen er klassifisert som ytre fordi denne eleven arbeider for å oppnå gode resultat, men denne ytre motivasjonen er delvis autonom då resultata vert arbeida mot delvis for å oppfylle eleven sine eigne krav til seg sjølv og delvis kontrollert då dei vert arbeida mot for å oppfylle forventningane til elevens foreldre. Elev 2 likar å nytte penn og papir for å arbeide med matematikk, sidan dette er ei arbeidsform der eleven føler ein slags fridom til å løyse oppgåver på sin måte. Når elev 2 nytter programmering så må visse reglar oppfyllast for å få eit program til å fungere, og som ein konsekvens av dette så kan ikkje eleven jobbe like fritt som hen skulle ha likt. Dette gjer at når eleven programmerer hentar hen motivasjon frå det å få produsert eit

fungerande program, i motsetnad til når eleven nyttar penn og papir, der eleven i større grad vert motivert av sjølve arbeidsprosessen mot å få produsert eit riktig svar. Eleven uttaler også i intervju at hen vert motivert av å jobbe med programmering fordi eleven er klar over at dette er eit fagfelt som vert meir og meir relevant. Denne ytre motivasjonen er vanskeleg å kategorisere då den på den eine sida er kontrollert (digitaliseringa av samfunnet sett press på eleven), men også autonom (eleven arbeider for å betre kunne forstå og ta del i framtidas digitale løysingar).

Elev 3 har ein indre motivasjon knytt til arbeid i matematikkfaget, då hen viser til å verte motivert av å lære seg nye ting i faget. Eleven syns at arbeid med programmering i matematikkfaget verkar motiverande fordi PCen gir ein respons til eleven når programmering vert nytta vert nytta. Denne «samtalen» som oppstår mellom elev og PC er noko som Sfard og Leron (1996, side 192) nyttar for å fremme programmering sin plass i matematikklasserommet.

Elev 4 sin motivasjon i matematikkfaget vert henta frå følelsen av meistring eleven får i arbeidet med faget. Denne motivasjonen er indre retta, då eleven arbeider med matematikk fordi arbeidet i seg sjølv er fornøyeleg. Denne motivasjonen kan verke til å bli svekka når eleven arbeider med programmering samanlikna med penn og papir, då eleven meiner det er enklare å arbeide med penn og papir. Eleven syns likevel at programmeringa er motiverande når eleven oppnår forståing, og klarer å løyse programmeringsoppgåver. Altså er det meistringa i arbeidet som står for motivasjonen i matematikkfaget også når programmeringsverktøy vert tatt i bruk.

Innsatsen ein legg ned i arbeidet med matematikk er innlemma i denne dimensjonen av matematikksynet. Innsats vart i delkapittel 2.3 definert som intensiteten av eleven sin motivasjon (Sthephani et al., 2019, side 2), og i denne studien kan ein få ein indikasjon på elevane sin innsats ved å studere tidsmengdene som elevane har gitt på spørsmåla knytt til arbeid med programmering i faget. Sjølv om desse spørsmåla ikkje direkte knyt seg til elevane sin eigen tidsbruk så kan svara gi eit bilete på innsatsen som elevane føler det er rimeleg å legge ned når ein arbeider med matematikk med og utan programmeringsverktøy. Frå intervjua ser ein også at alle elevane i intervjuutvalet relaterer spørsmåla om tidsbruk til sitt eige arbeid og sine eigne erfaringar i klasserommet, noko som aukar sensibiliteten av å nytte elevane sine svar om tidsbruk som ein indikasjon på deira individuelle innsatsar i programmering.

Ein ser tendensar til at når det både er snakk om typiske og vanskelege oppgåver så aukar elevane tidsmengdene når oppgåvene inkluderer programmering. Dette kan indikere at elevane føler dei treng å legge inn ein større innsats når det kjem oppgåver som inkluderer

programmering, men det treng ikkje nødvendigvis å bety dette. Sidan intervjua tydar på at elevane i stor grad kjem fram til svara på desse spørsmåla ved å bruke sine eigne erfaringar kan auken i tidsmengdene rett og slett stamme frå avgrensa erfaring ved bruk av programmering. Elevar som har lite eller inga erfaring med programmering kan setje av lengre tid på spørsmåla som inkluderer programmering fordi dei sjølv føler at dei treng lengre tid for å forstå oppgåva. Denne påstanden vert støtta opp under i intervju med elev 1, elev 2 og elev 4, som alle uttaler at dei sett av ekstra tid til programmeringsoppgåver enten fordi det er eit nytt verktøy eller fordi dei syns programmeringa var utfordrande. Elev 3 har tidlegare programmeringserfaring, og lærte ikkje noko nytt i timane med programmering på VGS. Denne eleven svarer like tidsmengder på spørsmålet om ei typisk oppgåve med og utan programmering (5 minutt) og på spørsmålet om ei vanskeleg oppgåve med og utan programmering (10 minutt). Dette kan tyde på at forskjellen i tid på dei to spørjeundersøkingane kjem som ein konsekvens av generell manglande erfaring i elevmassen, i staden for ein auka innsatsvilje. Likevel uttalar elev 2 at hen arbeider lenger med programmeringsoppgåvene fordi eleven ønsker å få til oppgåvene, og skjønne korleis dei fungerer. Denne utsegna vitnar om ein auka grad av innsats hos eleven.

5.4 Vurdering av prosjektet

Ein viktig del av alle typar prosjekt er å vurdere det i ettertid. Dette masterprosjektet er no gjennomført, og eg ønsker i dette kapittelet å sjå tilbake på arbeidet som er lagt ned for å blant anna vurdere styrkar og svakheiter ved prosjektet, og sjå på kva som kunne ha vore gjort annleis for å hatt mogelegheita til å oppdrive eit meir nøyaktig og/eller aktuelt datamaterialet. Kapittelet er delt inn i 4 delkapittel. I 5.4.1 vil eg komme med nokon avklaringar rundt prosjektet som tidlegare ikkje er vorte nemnd, i 5.4.2 vil eg sjå litt på feilkjeldene som kan ha påverka resultata i prosjektet, i 5.4.3 vil eg vurdere dette prosjektet sine styrkar og svakheiter, og i 5.4.4 vil eg snakke litt om kva eg ville ha gjort annleis om eg fekk sjansen til å gjennomføre dette prosjektet på nytt.

5.4.1 Avklaringar

I dette delkapittelet vil eg avklare nokon element ved studien som ved fyrste augekast kan ha verka forvirrande for lesaren.

Om ein ser på spørjeundersøking 2 i vedlegg 4 så ser ein at det står eit spørsmål som relaterer seg til plasseringa av algoritmisk tenking innunder eit eller fleire av kjerneelementa i matematikk 1P. Datamateriale vart henta inn frå dette spørsmålet, men er ikkje vorte nytta vidare i oppgåva. Dette er på grunn av at spørjeundersøkinga vart konstruert før problemstillinga og forskingsspørsmåla tok endeleg form. Hovudmomenta i problemstillinga (erfaringar, forståing og syn) har alltid vore til stades, men i byrjinga av prosjektet var problemstillinga ein del vidare enn slik den har enda opp med å bli. Spørjeundersøking 1 måtte gjennomførast rimeleg tidleg i skuleåret for at læraren sine planar for klassen ikkje skulle skli ut, noko som gjorde at ei spørjeundersøking måtte konstruerast før problemstillinga var fullstendig justert til det den har blitt til i dag. Dette gjer at dette spørsmålet for den noverande problemstillinga er uaktuell, og datamaterialet frå dette spørsmålet er ikkje vorte nytta vidare i arbeidet.

På førehand syns eg det var vanskeleg å lage spørsmål som i ettertid kan nyttast for å konkludere noko kring problemstillinga i prosjektet. Dette gjorde at eg nok laga ein del fleire spørsmål enn det eg hadde trengt (både på spørjeundersøkingane og i intervjua), og at noko av datamaterialet diverre ikkje er så aktuelt. Meir om dette i delkapittel 5.4.3 og 5.4.4.

Som ein ser er denne oppgåva skrive på nynorsk, mens spørsmåla gitt i spørjeundersøkingane er skrive på bokmål. Dette kjem av at elevane i klassen der desse undersøkingane er gjennomført primært nyttar bokmål som skriftspråk, og eg valte av den grunn å gi dei spørsmål på det skriftspråket dei var mest van med å bruke. Dette for å unngå unødige forvirringar frå eit skriftspråk som elevane ikkje var van med å lese.

5.4.2 Feilkjelder

I dette delkapittelet vil eg diskutere enkelte av dei feilkjeldene som eg føler kan ha vore med å påverke resultata i prosjektet.

For at studieprogresjonen min og undervisningsprogresjonen til læraren i klassen skulle gå som planlagt vart det nødvendig å gjennomføre dei to spørjeundersøkingane og dei fire intervjua i timar som gjekk på ulike tidspunkt på dagen. Dette kan ha påverka resultata, då elevane verka mindre fokusert og konsentrert i timar som gjekk føre seg seint på dagen, i forhold til timar som låg plassert tidlegare på dagen. Ideelt sett skulle spørjeundersøkingane og intervjua ha blitt gjennomførte i timar som låg på same tidspunkt på dagen, for å minke påverknaden av elevane sin varierande motivasjon, konsentrasjon og iver.

Ei feilkjelde som kan ha bidratt til ei feil framstilling kring elevane si konseptuelle forståing av omgrepet algoritme er at spørsmåla som tok føre seg beskrivinga av dette omgrepet ikkje hadde eit svaralternativ som elevar som ikkje visste kva ein algoritme var kan velje. Elevane blei «tvungne» til å velje eit svaralternativ på det spørsmålet, og når ein elev som ikkje visste kva ein algoritme var har valt eit av svaralternativa så vart det tatt for gitt at den eleven trudde ein algoritme var det som han/ho hadde valt. På dette spørsmålet vart det vurdert å inkludere eit svaralternativ som elevar som ikkje visste kva ein algoritme var kunne krysse av på, men dette gjorde eg ikkje av frykt for at alle elvane som var litt usikre på kva ein algoritme var skulle velje dette alternativet. Det er tross alt det elevane meiner ein algoritme er som skal målast på desse spørsmåla, men ein må ta høgde for at ein eller fleire elevar har tippa svaralternativ på desse to spørsmåla.

Begge spørjeundersøkingane vart gjennomførte på elevane sine datamaskiner, noko som opnar mogelegheita for elevane at elevane har søkt opp svara på spørsmåla dei vart gitt. Sjølv om dei vart bedt om å ikkje gjere dette så er det mogeleg at elevar har søkt opp svar på til dømes definisjonen på ein algoritme og/eller algoritmisk tenking. Elevane i denne klassen var gruppert i mindre grupper i klasserommet, og sidan dei satt såpass tett så kan ein også tenke seg at fleire av elevane samarbeida, og i staden for å svare det dei tenkte/trudde/meinte så har dei svart det som fleirtalet i gruppa dei sitt i tenkte/trudde/meinte.

5.4.3 Styrker og svakheiter i prosjektet

Det vil alltid finnast styrkar og svakheiter i prosjekt som dette. I dette delkapittelet vil eg diskutere nokon av styrkane og svakheitene til dette prosjektet.

Prosjektet samlar inn store mengder datamateriale, og tek i bruk fleire ulike metodar for å hente inn dette materialet. Det å ha store mengder med datamateriale gjer at ein kan studere fleire aspekt, og sjå desse aspekta i samanheng med andre deler av datamaterialet. Dette kan gjere at ein ser klarare tendensar. Ein ser til dømes i dette diskusjonskapittelet at det veldig ofte vert dratt linjer mellom det datamaterialet som er samla inn frå observasjonen, spørjeundersøkingane og forskingsintervjua. Det at både kvantitative og kvalitative metodar er vorte nytta gjer at ein kan kombinere og samanlikne det kvantitative og det kvalitative datamaterialet for å endå klarare sjå dei tendensane ein ønsker å studere. Dette er ein av grunnane til at bruk av mixed methods kan bidra til å auke validiteten i eit prosjekt (Cohen et al., 2011, side 26).

Ein annan styrke som dette prosjektet har er at problemstillinga som vert tatt utgangspunkt i er svært aktuell. Prosjektet har gått føre seg i byrjinga av det fyrste skuleåret der algoritmisk tenking har vore ein del av læreplanane som lærarane i den norske skulen bruker. Sjølv om dette prosjektet tek utgangspunkt i ein 1P-klasse så kan ein tenke seg at funna frå denne studien kan gjere seg gjeldande for fleire klassetrinn der programmering og algoritmisk tenking er eit tema.

Denne studien hentar inn datamateriale frå 19 elevar, der 14 elevar sine svar kan nyttast i samanlikninga av to sett med datamateriale frå spørjeundersøkingane. I tillegg deltek 4 av desse 14 elevane i forskingsintervju. Dette er eit lite tal på elevar, noko som gjer at graden av intern og ekstern validitet i prosjektet er låg. Dette gjer at funna frå studien i liten grad kan generaliserast, og det er ei svakheit ved prosjektet.

Det er vanskeleg å sjå føre seg oppførselen til personar, noko som gjer at det å lage spørsmål til spørjeundersøkingar og intervju for å kartlegge spesifikke aspekt er vanskeleg. Elevar kan tolke spørsmåla på ulike måtar, og nokon vil som ein konsekvens av dette svare på ein annan ting enn det som spørsmåla var meint å kartlegge. Eg ser diverre ein tendens til at spørsmåla i denne studien i for liten grad er spesifisert til det som eg har hatt som mål å kartlegge, og eg tek sjølvkritikk for denne svakheita ved prosjektet. Nokon av momenta som var meint å kartleggast føler eg at eg har litt for lite datamateriale til å seie mykje om, fordi spørsmåla som er vorte brukt i kartlegginga omfattar for mykje.

5.4.4 Kva kunne vore gjort annleis?

Når eg ser tilbake på det arbeidet som er vorte lagt ned under datainnsamlinga så ser eg ting som eg skulle ønske eg hadde gjort annleis. Desse endringane knyt seg i all hovudsak til det som har blitt vurdert som prosjektet sine svakheiter.

Det fyrste som kunne ha vore gjort annleis er utforminga av spørsmåla som vert nytta for å hente inn datamaterialet i prosjektet. Som sagt i delkapittel 5.4.3 så føler eg at enkelte av spørsmåla er litt for vage til å effektivt kartlegge det som dei har som hensikt å kartlegge. Spørsmåla kunne i større grad ha vorte prøvd knytt mot forståinga av programmering og synet på programmering.

Eg skulle også ønske at eg hadde lagt ned ein større innsats i å formulere ei endeleg og konsis problemstilling tidlegare enn det eg gjorde. Dette kunne ha bidratt til at eg kunne ha utforma spørsmåla på ein meir konsis måte, som diskutert i avsnittet over.

6 AVSLUTNING

Prosjektet er gjennomført, resultata er presenterte og funna er blitt diskutert. I dette avsluttande kapittelet av masteroppgåva vil dei lause trådane bli forsøkt samla, med den hensikta å utforme ein endeleg konklusjon på problemstillinga i oppgåva. Mine tankar om vegen vidare frå det som denne oppgåva har produsert vil også bli presentert.

6.1 Konklusjon

Problemstillinga spør om elevane i ein matematikk 1P-klasse si erfaring, forståing og syn knytt til programmering som verktøy i matematikkundervisninga. Desse aspekta er vorte diskutert i kapittel 5, og i dette delkapittelet vil hovudfunna frå kapittel 5 bli summert opp og nytta til å gi ein konklusjon på problemstillinga.

Ein ser i kapittel 5.1 at det er usikkerheit knytt til elevane si eiga oppfatning av programmeringserfaring. Minst ein elev har kryssa av for ingen tidlegare erfaring med programmering, sjølv om eleven i realiteten har programmert tidlegare. Frå elevutvalet som er valt ut til intervju ser ein likevel at dei fleste av desse elevane har gjort riktige vurderingar knytt til sine eigne erfaringar med programmering. Om ein tek for gitt at elevane er ærlege på undersøkingane kan ein konkludere at av dei elevane som deltok i prosjektet så har 21%, altså om lag ein femtedel, av elevane tidlegare erfaring frå det som dei huskar og/eller oppfattar som programmering. Elevane sine tidlegare programmeringserfaringar varierer med omsyn på kor omfattande dei er. To av elevane har grunngitt kvifor dei har kryssa av for tidlegare programmeringserfaring, og ein av desse elevane har erfaring frå eit programmeringskurs som eleven deltok på som born, mens den andre eleven har sin erfaring frå eit toårig programmeringsfag på ungdomsskulen som fokuserte på programmering i Scratch og Python.

Læraren har valt å legge opp eit pensum som nyttar programmeringa som eit verktøy for at elevane betre skal skjønne fagstoffet rundt bruk av formlar i matematikken. Dette gjer at nytten av programmeringa får eit avgrensa fokus i undervisninga, noko som ein stor del av elevgruppa verkar å meine er ugunstig. Dette synest å ha påverka elevane sitt engasjement og interesse for fagstoffet i ei negativ retning, noko som kan ha gitt eit tilsvarande negativt utslag på deira

forståing. Elevane arbeida med ein god del problemløysingsoppgåver der den algoritmiske tenkinga sto sentralt. Enkelte elevar har gitt tilbakemeldingar som kommenterer at oppgåvene i større grad kunne ha vore engasjerande, moro og verkelegheitsnære. Alt i alt ser ein tendensar til at undervisninga og oppgåvearbeidet har bidratt til at elevane har opparbeida seg ei middels god konseptuell forståing av programmering, mens den prosedyremessige forståinga av programmering ikkje ser ut til å vere spesielt god hos elevane. Generelt sett syns elevane at arbeidet med programmering var noko lite lærerikt, og datamaterialet som kan bidra til å gi eit inntrykk av elevane si forståing av programmering etter undervisninga viser at forståinga til elevgruppa har vorte betre, men i noko liten grad.

Ein kan frå datamaterialet sjå at enkelte av elevane har låg sjølvtillit knytt til programmering før undervisninga tek til. Også i løpet av undervisninga kan ein sjå tendensar på låg sjølvtillit hos elevane, men det vert også observert element som tyder på at elevane si sjølvtillit stig noko i tråd med deira aukande erfaring med å programmere. Sjølv om det finst unntak, så synast elevane å vurdere sine eigne kompetansar innanfor programmering som låge, kanskje som eit resultat av dei mange vanskane dei støyter på i løpet av programmeringa. Desse vanskane verkar å gjere elevane frustrerte, og elevane er også generelt sett einige om at arbeidet med programmering i liten grad er moro. Ein ser evidens for både indre og ytre motivasjon for arbeid med matematikk hos elevane, men generelt sett minkar graden av motivasjon hos elevane når matematikkarbeidet inkluderer programmering. Oppsummert gir datamaterialet evidens for at elevgruppa i 1P synast å ha eit syn på bruk av programmeringsverktøy i matematikkundervisninga som gir uttrykk for låg grad av fornøyelse og minkande grad av motivasjon i arbeidet, samt ein låg, men noko stigande, sjølvtillit etter kvart som arbeidet går framover.

6.2 Vegen vidare

I vidare forsking kan det vere aktuelt å forsøke og kartlegge elevane si forståing og syn på programmering etter at elevane har arbeida med verktøyet over lengre tid, til dømes eit heilt skuleår. Datamaterialet i dette prosjektet er prega av aspekt som i mange tilfelle kan forklarast utifrå elevane si manglande erfaring, så det kan tenkast at ein ville ha sett klarare tendensar hos elevane om ein hadde gjennomført same studie over eit heilt skuleår. Etter kvart vil også

ungdomsskuleelevar som har mykje programmeringserfaring frå ungdomsskulen byrje på VGS, så liknande studiar på desse elevane kunne også ha vore aktuelt og interessant.

Det vil også vere aktuelt å forsøke og kartlegge dei same momenta frå denne studien i andre klassetrinn der programmering er pensum. Når fagfornyinga er fullstendig innført i den norske skulen kan det vere interessant å kartlegge elevane sine erfaringar, forståing og syn knytt til programmering på alle klassetrinna der det skal takast i bruk.

REFERANSELISTE

2018a. *Logic* [Internett]. Computer Hope. Tilgjengeleg frå: https://www.computerhope.com/jargon/l/logic.htm [Henta 16. november 2020].

2018b. *Syntax* [Internett]. Computer Hope. Tilgjengeleg frå: https://www.computerhope.com/jargon/s/syntax.htm [Henta 16. november 2020].

Alves, P. F. (2014) Vygotsky and Piaget: Scientific concepts. *Psychology in Russia : state of the art,* 7 (3)**,** s. 24-34.

Bakhurst, David og Shanker, Stuart (2001) *Jerome Bruner : language, culture, self,* London ; Thousand Oaks, Calif., SAGE Publications.

Barr, Valerie og Stephenson, Chris (2011) Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community. *ACM Inroads,* 2 (1)**,** s. 48-54.

Breiteig, Trygve (2008) Problemløsning som inngangsport til matematikk. *Tangenten,* 1**,** s. 35-40.

Çakır, Aslı og Akkoç, Hatice (2020) Examining socio-mathematical norms related to problem posing: a case of a gifted and talented mathematics classroom. *Educational studies in mathematics,* 105 (1)**,** s. 19-34.

Clements, Douglas H. og Battista, Michael T. (1989) Learning of Geometric Concepts in a Logo Environment. *Journal for Research in Mathematics Education,* 20 (5)**,** s. 450-467.

Cohen, Louis, Manion, Lawrence og Morrison, Keith (2011) *Research methods in education,* London ; New York, Routledge.

Dalland, Olav (2017) *Metode og oppgaveskriving,* Oslo, Gyldendal Akademisk.

Danielsen, Anne Grete (2013) Kunnskapsbygging i skolen via kvantitative verktøy : statistikk og spørreskjema. Oslo, Universitetsforl., cop. 2013, s. 138-154.

Devore, Jay L. og Berk, Kenneth N. (2012) Modern Mathematical Statistics with Applications. 2nd ed. utg. New York, NY, Springer New York : Imprint: Springer.

Dubinsky, Ed og Tall, David (1991) Advanced Mathematical Thinking and the Computer. *Advanced Mathematical Thinking.* Hingham, Kluwer Academic Publishers, s. 231-248.

Eickelmann, Birgit (2019) Measuring Secondary School Students' Competence in Computational Thinking in ICILS 2018 – Challenges, Concepts, and Potential Implications for School Systems Around the World IKong, Siu-Cheung og Abelson, Harold (red.) *Computational Thinking Education.* Springer, s. 53-64.

Evenshaug, Oddbjørn og Hallen, Dag (2000) *Barne- og ungdomspsykologi,* Oslo, Gyldendal akademisk.

Forsström, Sanna Erika og Kaufmann, Odd Tore (2018) A Literature Review Exploring the use of Programming in Mathematics Education. *International Journal of Learning, Teaching and Educational Research,* 17**,** s. 18-32.

Fraillon, Julian, et al. (2018) IEA International Computer and Information Literacy Study 2018 - Assessment Framework.

Garland, Ron (1991) The Mid-Point on a Rating Scale: Is it Desirable? *Marketin Bulletin,* 2**,** s. 66-70.

Germeten, Sidsel og Bakke, Jarle (2013) Observasjon: Å innta klasserommet med egne sanser. *Læreren som forsker***,** s. 109-123.

Herzog, A. Regula og Bachman, Jerald G. (1981) Effects of Questionnaire Length on Response Quality. *Public opinion quarterly,* 45 (4)**,** s. 549-559.

Hyland, Ken (2009) *Teaching and researching writing,* Harlow, Longman.

John Paul, Mueller (2014) *Beginning programming with Python for dummies,* Somerset, Somerset: Wiley.

Kong, Siu Cheung og Abelson, Harold (2019) *Computational thinking education,* Singapore, Springer Open.

Kunnskapsdepartementet (2013) Stortingsmelding 20: På rett vei - kvalitet og mangfold i skolen.

Kunnskapsdepartementet (2016) Stortingsmelding 28: Fag - Fordypning - Forståelse - En fornyelse av kunnskapsløftet.

Kvale, Steinar, et al. (2015) *Det kvalitative forskningsintervju,* Oslo, Gyldendal akademisk.

Leron, Uri og Dubinsky, Ed (1995) An Abstract Algebra Story. *The American Mathematical Monthly,* 102 (3)**,** s. 227-242.

Lie, Johan, Hauge, Inge Olav og Meaney, Tamsin Jillian (2017) Computer programming in the lower secondary classroom: learning mathematics. *Italian journal of educational technology,* 25 (2)**,** s. 27-35.

Mellin-Olsen, Stieg og Hoel, Sverre (1984) *Eleven, matematikken og samfunnet : en undervisninglære,* Bekkestua, NKI-forl.

Myers, Christopher R. og Sethna, James P. (2007) Python for Education: Computational Methods for Nonlinear Systems. *Computing in science & engineering,* 9 (3)**,** s. 75-79.

National Council of Teachers of Mathematics (2014) *Principles to actions : ensuring mathematical success for all,* Reston, VA, NCTM National Council of Teachers of Mathematics.

National Research Council (U.S.). Committee for the Workshops on Computational Thinking. (2010) *Report of a workshop on the scope and nature of computational thinking,* Washington, D.C., National Academies Press.

Newton, Douglas P. (2012) *Teaching for understanding : what it is and how to do it*, London, Routledge.

Niss, Mogens og Højgaard, Tomas (2019) Mathematical Competencies Revisited. *Educational Studies in Mathematics*, 102**,** s. 9-28.

Olson, David R. (2001) Education: The Bridge From Culture to Mind. I Bakhurst, David og Shanker, Stuart (red.) *Jerome Bruner: Language, Culture and Self.* Sage Publications, s. 104-115.

Oppenheim, A. N. (1992) *Questionnaire design, interviewing, and attitude measurement*, London ; New York, Continuum.

Rossen, Eirik. 2018. *Informatikk* [Internett]. Store Norske Leksikon. Tilgjengeleg frå: https://snl.no/informatikk [Henta 14. oktober 2020].

Ryan, Richard M. og Deci, Edward L. (2000) Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. *Contemp Educ Psychol*, 25 (1)**,** s. 54-67.

Santos-Trigo, Manuel (2019) Mathematical Problem Solving and the Use of Digital Technologies. Cham, Cham: Springer International Publishing.

Schoenfeld, Alan (1989a) Explorations of Students' Mathematical Beliefs and Behavior. *Journal for research in mathematics education*, 20 (4)**,** s. 338-355.

Schoenfeld, Alan (1989b) Teaching Mathematical Thinking and Problem Solving. I Resnick, Lauren B. og Klopfer, Leopold E. (red.) *Toward the Thinking Curriculum: Current Cognitive Research.* ASCD, s. 83-103.

Sfard, Anna og Leron, Uri (1996) Just Give me a Computer and I will Move the Earth: Programming as a Catalyst of a Cultural Revolution in the Mathematics Classroom. *International Journal of Computers for Mathematical Learning*, 1 (2)**,** s. 189-195.

Sierpinska, Anna (1994) Understanding in mathematics. London ;,Washington, D.C., Falmer.

Skaalvik, Einar M. og Skaalvik, Sidsel (2013) *Skolen som læringsarena : selvoppfatning, motivasjon og læring*, Oslo, Universitetsforl.

Skovsmose, Ole (1998) Undersøgelseslandskaber. I Dalvang, Tone og Rohde, Vetle (red.) *Matematikk for alle.* Trondheim, Landslaget for matematikk i skolen (LAMIS), s. 24-37.

Solvang, Ragnar (1986) *Matematikk-didaktikk*, Rud, NKI-forlaget.

Sthephani, A., et al. (2019) Students' view of mathematics in general high school and vocational high school. *Journal of physics. Conference series*, 1321**,** s. 32079.

Tangen, Reidun (2014) Balancing Ethics and Quality in Educational Research-the Ethical Matrix Method. *Scandinavian journal of educational research*, 58 (6)**,** s. 678-694.

Utdanningsdirektoratet (2013) Læreplan i matematikk fellesfag.

Utdanningsdirektoratet (2019) Algoritmisk tenkning.

Utdanningsdirektoratet (2020a) Læreplan i matematikk fellesfag Vg1 praktisk (matematikk P).

Utdanningsdirektoratet (2020b) Læreplan i matematikk fellesfag Vg1 teoretisk (matematikk T).

Weese, Joshua Levi (2017) Bringing Computational Thinking to K-12 and Higher Education. ProQuest Dissertations Publishing.

Wing, Jeannette (2006) Computational thinking. *Communications of the ACM,* 49 (3)**,** s. 33-35.

Wiske, Martha Stone, Franz, Kristi Rennebohm og Breit, Lisa (2005) Teaching for Understanding with technology. 1st ed. utg. San Francisco, CA, Jossey-Bass.

VEDLEGG

Vedlegg 1: Prosjektgodkjenning frå Norsk Senter for Forskningsdata (NSD)

04.09.2020 - 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 den 04.09.2020 med vedlegg, samt i meldingsdialogen mellom innmelder og NSD. 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 01.01.2021.

LOVLIG GRUNNLAG

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

PERSONVERNPRINSIPPER

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

- lovlighet, rettferdighet og åpenhet (art. 5.1 a), ved at de registrerte får tilfredsstillende informasjon om og samtykker til behandlingen

- formålsbegrensning (art. 5.1 b), ved at personopplysninger samles inn for spesifikke, uttrykkelig angitte og berettigede formål, og ikke viderebehandles til nye uforenlige formål

- dataminimering (art. 5.1 c), ved at det kun behandles opplysninger som er adekvate, relevante og nødvendige for formålet med prosjektet

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

DE REGISTRERTES RETTIGHETER

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 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 eventuelt 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: Mirza Hodzic

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

Vedlegg 2: Informasjonsskriv og samtykkeskjema utdelt til elevane

Førespurnad om å delta i forskingsprosjekt

Mitt namn er Stian Hirth, og eg skriv dette semesteret ei masteroppgåve i matematikkdidaktikk som del av den integrerte lektorutdanninga ved Universitetet i Bergen.

Dette er ein førespurnad til deg om å delta i eit forskingsprosjekt som omhandlar algoritmisk tenking og bruk av programmering i matematikk 1P.

Algoritmisk tenking som del av læreplanen 2020

Hausten 2020 vart dei nye læreplanane i matematikk 1P tatt i bruk. Under "Utforsking og problemløysing" i kjerneelementa finn ein følgande sitat: *Algoritmisk tenking er viktig i prosessen med å utvikle strategiar og framgangsmåtar for å løyse problem og inneber å bryte ned eit problem i delproblem som kan løysast systematisk. Vidare inneber det å vurdere om delproblema best kan løysast med eller utan digitale verktøy.*

I masteroppgåva mi ønsker eg å utforske elevars syn og tankegangar på implementeringa av arbeid knytt til algoritmisk tenking i matematikkpensumet. Ein stor del av denne kartlegginga vil bli fokusert rundt arbeidet med programmeringsverktøy i klasserommet. Her ønsker eg blant anna å undersøke korleis arbeid med slike verktøy endrar arbeidsrutinane i matematikkfaget, og elevane sine haldningar til desse endringane.

Deltaking i forskingsprosjekt

Innsamling av datamateriale vil gå føre seg ved bruk av to spørjeundersøkingar gitt ved ulike tidspunkt. På bakgrunn av svara i undersøkingane vil eit utval elevar bli plukka ut til å gjennomføre eit intervju der ein vil utforske enkelte aspekt i undersøkinga nærmare. I intervjua vil lydopptakar verte nytta. Spørjeundersøkingane vil bli gjennomført i klasserommet, og intervjua vil også finne stad i skuletida. Nærmare informasjon kring tid og stad for gjennomføringa vil bli gitt ved deltaking.

Det er heilt frivillig å delta i prosjektet, og du vil til ei kvar tid ha mogelegheit til å trekkje deg undervegs, utan å måtte grunngje dette valet nærare. Dersom du skulle ønske å trekkja deg vil all innsamla data som omhandlar deg verte sletta. Opplysningane vil bli anonymisert, og vil bli behandla konfidensielt. Lydopptaka vil bli sletta seinast 1. januar 2021, når oppgåva er ferdig.

For å sikre anonymitet vil kvar prosjektdeltakar få tildelt ein deltakarkode som vert brukt for å kople saman dei to spørjeundersøkingane. Namn blir altså ikkje brukt verken i spørjeundersøkingane eller intervjua. Filene med datamaterialet frå spørjeundersøkingane og intervjua, samt lydopptaka frå intervjua, vert oppbevart som krypterte filer på ein PC som kun er tilgjengeleg for personane involverte i prosjektet.

Dine rettigheiter

Så lenge du kan identifiserast i datamaterialet har du rett til:

- innsyn i kva for personopplysningar som er registrert om deg.
- å få retta personopplysningar om deg.
- å få sletta personopplysningar om deg.
- å få utlevert ein kopi av dine personopplysningar.
- å sende klage til personvernombodet eller datatilsynet om behandlinga av dine personopplysningar.

Kva gir meg rett til å behandle personopplysningar om deg?

Eg vil behandle opplysningar om deg basert på ditt samtykke.

Matematisk institutt ved Universitetet i Bergen er behandlingsansvarleg.

Dersom du kunne tenkje deg å delta på to spørjeundersøkingar og eit intervju er det fint om du kan skrive under på den vedlagte samtykkeerklæringa.

Kontaktopplysningar

Dersom du har spørsmål til prosjektet kan du ta kontakt med:

- Stian Hirth, student – e-post: Stian.Hirth@student.uib.no eller telefon: 90 09 14 66
- Johan Lie, rettleiar – e-post: Johan.Lie@uib.no
- Janecke Helene Veim, personvernombod ved Universitetet i Bergen – e-post: personvernombud@uib.no
- Norsk senter for forskningsdata AS (NSD) – e-post: personverntjenester@nsd.no eller telefon: 55 58 21 17

Med vennleg helsing

Student	Prosjektansvarleg
	(Rettleiar)
Stian Hirth	Johan Lie

Samtykkeerklæring

Eg har motteke og forstått informasjonen om prosjektet, og har fått anledning til å stille spørsmål. Eg samtykker til:

- ☐ å bli observert i skuletimar
- ☐ å delta i to spørjeundersøkingar
- ☐ å delta i intervju
- ☐ at intervjuet kan takast opp med lydopptakar

Eg samtykker til at mine opplysningar vert behandla fram til prosjektet er avslutta, 1. januar 2021.

---------- ----------

Dato Signatur til prosjektdeltakar

Signatur til foreldre/føresette (om prosjektdeltakar er under 16 år)

Vedlegg 3: Spørjeundersøking 1

Skriv inn din deltagerkode

_____

Om du forstår det underliggende materialet, hvor lang tid tenker du at det skal ta å løse en typisk matematikkoppgave? For eksempel en deloppgave der en ligning skal løses

_____

Hva tenker du er en rimelig tidsmengde å bruke på en matematikkoppgave før en gir opp, og innser at oppgaven er for vanskelig til å løses?

_____

Har du tidligere erfaring med programmering?

(1) ❑ Ja
(2) ❑ Nei

Hvilke(t) programmeringsspråk har du tidligere erfaring med?

(1) ❑ Python
(2) ❑ Java
(3) ❑ C
(4) ❑ Scratch
(5) ❑ Andre
(6) ❑ Vet ikke / husker ikke

Hvilken av definisjonene under føler du gir den beste beskrivelsen av ordet "algoritme"?

(1) ❑ En gjentakende prosess som alltid gir samme resultat

(2) ❑ Samlingen av kodegrunnlaget til et dataprogram eller en app

(3) ❑ En oppskrift på hvordan et problem løses mest mulig nøyaktig

(4) ❑ En fullstendig beskrivelse av fremgangsmåten for å løse en oppgave

(5) ❑ Et hjelpemiddel for å vurdere om en gitt framgangsmåte gir et ønsket resultat

Hva tror du ligger i begrepet "algoritmisk tenkning"?

_____

_____

_____

_____

_____

_____

Ranger følgende utsagn på en skala fra 1 til 6, der 1 er helt uenig og 6 er helt enig

	1	2	3	4	5	6	Vet ikke
Jeg kommer aldri til å få bruk for matematikken jeg lærer på skolen senere i livet	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(6) ❑	(11) ❑
Matematikk er lite relevant for meg	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(6) ❑	(11) ❑
Det var et stort fokus på bruk av digitale hjelpemidler i matematikkfaget på ungdomsskolen	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(6) ❑	(11) ❑
Jeg håper på et stort fokus på bruk av digitale hjelpemidler i matematikkfaget på VGS	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(6) ❑	(11) ❑

Tror du fokus på bruk av programmering i matematikkfaget vil være nyttig? Hvorfor/hvorfor ikke?

_____
_____
_____
_____
_____
_____

Under er kjerneelementene i den nye læreplanen til matematikk 1P listet. Under hvilke(t) kjerneelement tror du algoritmisk tenkning hører til?

(1) ❑ Utforskning og problemløsning
(2) ❑ Modellering og anvendelser
(3) ❑ Resonnering og argumentasjon
(4) ❑ Representasjon og kommunikasjon
(5) ❑ Abstraksjon og generalisering
(6) ❑ Matematiske kunnskapsområder

Vedlegg 4: Spørjeundersøking 2

Skriv inn din deltagerkode

_____

Foregikk undervisningen i programmering slik du hadde sett for deg? Utdyp gjerne svaret ditt

_____
_____
_____
_____
_____
_____

Om du skjønner det underliggende materialet, hvor lang tid tenker du det skal ta å løse en typisk programmeringsoppgave? For eksempel å lage et program som regner ut arealet av et rektangel

_____

Hva tenker du er en rimelig tidsmengde å bruke på en programmeringsoppgave før en gir opp, og innser at oppgaven er for vanskelig til å løses?

_____

Hvilken av definisjonene under føler du gir den beste beskrivelsen av ordet "algoritme"?

(1) ❑ En gjentakende prosess som alltid gir samme resultat
(2) ❑ Samlingen av kodegrunnlaget til et dataprogram eller en app
(3) ❑ En oppskrift på hvordan et problem løses mest mulig nøyaktig
(4) ❑ En fullstendig beskrivelse av fremgangsmåten for å løse en oppgave
(5) ❑ Et hjelpemiddel for å vurdere om en gitt framgangsmåte gir et ønsket resultat

Hvordan ville du definert "algoritmisk tenkning"?

_____
_____
_____
_____
_____
_____

Ranger følgende utsagn på en skala fra 1 til 6, der 1 er helt uenig og 6 er helt enig

	1	2	3	4	5	6	Vet ikke
Jeg kommer aldri til å få bruk for programmeringen jeg har lært på skolen	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(7) ❑	(6) ❑
Programmering virker som et nyttig verktøy	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(7) ❑	(6) ❑
Arbeid med programmering var frustrerende	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(7) ❑	(6) ❑
Arbeid med programmering var lærerikt	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(7) ❑	(6) ❑
Arbeid med programmering var moro	(1) ❑	(2) ❑	(3) ❑	(4) ❑	(5) ❑	(7) ❑	(6) ❑

Kunne undervisningen i programmering blitt lagt opp på en annen måte for at du skulle hatt mer nytte av den?

_____
_____
_____
_____
_____
_____

Har du andre erfaringer fra undervisningen i algoritmisk tenkning som du ønsker å dele? For eksempel noe som overrasket deg, noe som frustrerte deg eller lignende

_____
_____
_____
_____
_____
_____

Vedlegg 5: Intervjuguide

Intervjuguide

Bakgrunnsinformasjon

- Hadde du høyrt om programmering tidlegare?
- Kor lang erfaring har du med å programmere?
- Kva programmeringsspråk har du nytta tidlegare?
- Korleis er digitale hjelpemiddel vorte nytta i matematikkundervisninga som du har deltatt i tidlegare?
- Kva motiverer deg for å arbeide med matematikk?

Utforsking av svara i spørjeundersøkinga

- Korleis har du vurdert når du har oppgitt eit svar på kor lang tid du tenker det skal ta å løyse ei matematikkoppgåve?
- Korleis har du vurdert når du har oppgitt eit svar på kor lang tid du tenker det skal ta å løyse ei programmeringsoppgåve?
- Kvifor/kvifor ikkje er dei to tidsromma ulike?
- Om du ikkje klarer å løyse ei matematikkoppgåve, kor lenge pleier du å arbeide med denne før du gir opp?
    - Er denne tida annleis om du arbeider med ei oppgåve som inneheld programmering? Kvifor/kvifor ikkje?

- Kvifor har du valt den definisjonen av algoritme som du har gjort i spørjeskjemaet?
- Hadde du høyrt ordet algoritme tidlegare?
- Kvifor/kvifor ikkje har du endra definisjonen av algoritme mellom dei to spørjeundersøkingane?
- Korleis har dine tankar rundt omgrepet «algoritmisk tenking» endra seg i løpet av tida mellom dei to spørjeundersøkingane?

Programmeringsundervisning

- Korleis syns du undervisninga i programmering har vore?
- Hjelper programmering deg å bli motivert for arbeid i matematikk?
- Merker du nokon forskjell i måten du blir motivert på når du arbeider med programmering i forhold til når du jobbar med penn og papir?
- Ser du for deg at du kjem til å bruke programmering vidare?

Programmering i matematikkundervisninga

- Kva er dine synspunkt knytt til bruk av digitale hjelpemiddel i matematikktimane?
- I matematikk T er det direkte fokus på programmering. Kvifor trur du det er eit større direkte fokus på programmering i T i forhold til P?
- Føler du at du har hatt nytte av å arbeide med programmering i matematikken?
- Korleis vil du sei at arbeid med programmering skil seg frå matematikkundervisninga du har hatt tidlegare?
- Vil du seie at algoritmisk tenking er meir eller mindre relevant for deg om temaet blir samanlikna med resten av pensumet i matematikk 1P? Kvifor?