Høgskolen i Østfold

MASTEROPPGAVE

Tittel

Potensiale for dybdelæring i algebra gjennom Python-programmering i ungdomsskolen

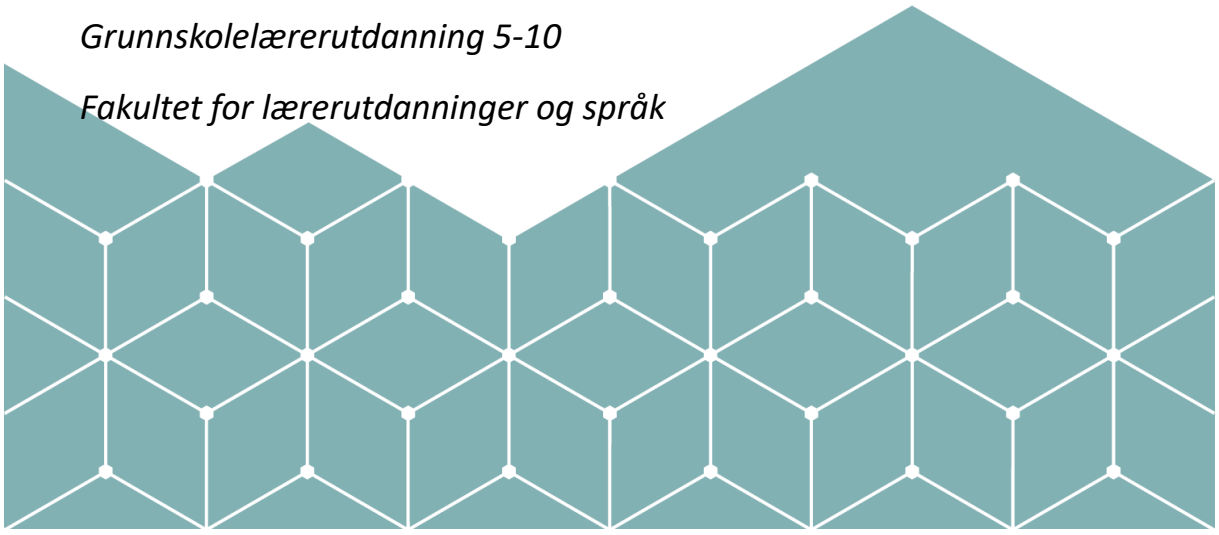
Christer R. Stenersrød

Dato

15.05.24

Grunnskolelærerutdanning 5-10

Fakultet for lærerutdanninger og språk



Sammendrag

Denne masteroppgaven undersøker potensialet for dybdelæring gjennom integrering av Python-programmering i algebraundervisningen for ungdomsskoleelever. Studien adresserer to sentrale spørsmål:

1. Hvilket potensial for dybdelæring i algebra kan identifiseres i elevers arbeid med programmering i Python?
2. På hvilken måte kan hindringer gi grobunn for dybdelæring?

Masteroppgaven bygger på teorier om dybdelæring og programmeringens rolle i matematikkundervisningen, inkludert Schoenfelds konsept om robust læring og ulike strategier for generalisering innen algebra. Teorien understreker betydningen av at elever kan anvende og utforske algebraiske konsepter gjennom problem-baserte utfordringer.

For å besvare spørsmålene har jeg valgt en kvalitativ forskningsmetodikk. Innsamlingen av data ble gjennomført ved observasjon av elevers arbeid i grupper med programmeringsoppgaver. Analysen fokuserer på hvordan elever anvender programmering til å utvikle en dypere forståelse av matematiske konsepter.

Funnene omfatter elevers arbeidsprosesser med programmeringsoppgaver og interaksjoner i klasserommet, samt teoretisk støttelitteratur som diskuterer programmeringens plass i matematikkundervisningen.

Sentrale funn fra observasjoner og intervjuer viser at elever gjennom programmering kan utvikle en dypere forståelse for algebraiske konsepter, men at teknologiske barrierer kan hindre optimal implementering. Videre indikerer funnene at ved å overkomme disse barrierene gjennom gode planlagte undervisningsopplegg som tilrettelegger for dialog, kan dybdelæring fremmes effektivt.

Masteroppgaven understreker nødvendigheten av å ruste elever med ferdighetene som kreves i et digitalisert samfunn. Resultatene har relevans for både lærerens praksis og utdanningspolitikk.

Abstract

This master's thesis investigates the potential for deep learning through the integration of Python programming in algebra teaching for middle school students. The study addresses two key questions:

1. What potential for deep learning in algebra can be identified in students' work with programming in Python?
2. In what way can obstacles provide a foundation for deep learning?

The thesis is based on theories of deep learning and the role of programming in mathematics education, including Schoenfeld's concept of robust learning and various strategies for generalization within algebra. The theory emphasizes the importance of students being able to apply and explore algebraic concepts through problem-based challenges.

To answer these questions, I have chosen a qualitative research methodology. Data collection was conducted by observing students' work in groups with programming tasks. The analysis focuses on how students use programming to develop a deeper understanding of mathematical concepts.

The findings include students' work processes with programming tasks and interactions in the classroom, as well as theoretical support literature discussing the role of programming in mathematics education.

Key findings from observations and interviews show that through programming, students can develop a deeper understanding of algebraic concepts, but technological barriers can hinder optimal implementation. Furthermore, the findings indicate that by overcoming these barriers through well-planned teaching arrangements that facilitate dialogue, deep learning can be effectively promoted.

The thesis underscores the necessity of equipping students with the skills required in a digitalized society. The results have relevance for both teachers' practice and educational policy (Oversatt av Ai).

Forord

Disse fem årene ved Høgskolen i Østfold har virkelig flydd av sted. Fra starten av lærerutdanningen med innføringen av LK20, til overgangen til hjemmeundervisning på grunn av Covid-19, avslutter jeg nå min grunnskolelærerutdanning med en fullført masteroppgave.

Dette studiet har vært intens og krevende, fylt med utfordringer og betydelige mestringserfaringer. Å utforske potensialet for dybdelæring i programmering innenfor algebra har vært både lærerikt og spennende.

Først vil jeg takke informantene mine og matematikklæreren som har tatt seg tid til å organisere slik at jeg skulle få gjennomføre undersøkelsen. Så vil jeg gi en takk til min veileder Toril Eskeland Rangnes for god hjelp og konstruktive tilbakemeldinger. Takk til svigermor som har hjulpet meg med språklige utfordringer. Takk til medstudenter for mange gode samtaler og diskusjoner i løpet av studiet.

Jeg vil også rette en spesiell takk til min kone som har vært tålmodig. Som har hjulpet meg å holde struktur i hverdagen, slik at denne masteroppgaven har latt seg gjennomføre. Nå ser jeg frem til sommerferie og mer tid med familien min, før jeg tar sats inn i arbeidet som nyutdannet lærer på en ungdomsskole.

Halden, 15.mai 2023

Christer R. Stenersrød

Innhold

Figur-oversikt

Tabell-oversikt

1.0 Innledning

Gjennom dette masterprosjektet tar jeg sikte på å undersøke hvordan Python-programmering kan virke inn på dybdelæringen i algebra for elever i ungdomsskolen. Det er et økt fokus på teknologi i utdanningssystemet, blant annet spesifisert i matematikkfaget (LK20). Betydningen av datakompetanse har ført til behov for å undersøke hvordan programmeringsspråk, som for eksempel Python, kan integreres i matematikkundervisningen.

En av grunnene til at Python programmering i emnet algebra motiverer meg, er observasjon under praksis hvor jeg så at mange elever hadde vanskeligheter med å forstå algebra. Videre ønsker jeg mer kunnskap om hvilket potensiale programmering i matematikk kan ha for læring og forståelse av matematiske konsepter som algebra.

I dette prosjektet vil jeg som nevnt tilegne meg mer innsikt i hvordan programmering kan bidra til dybdelæringen i algebra for elever i ungdomsskolen. De resultatene jeg finner gjennom denne studien, vil ha relevans for min lærerrolle og for hvordan elever kan få en dypere forståelse av algebra i arbeid med programmering. Studien kan også være et bidrag for lærere og andre som jobber i utdanningssektoren.

Programmering, som et relativt nytt tilskudd i skolematematikken, antas å ha en unik rolle i å fremme dybdelæring ifølge Ludvigsen utvalget (NOU 2015:8, 2015). De mener det er viktig å bygge bro mellom teori og praksis, programmering gir derfor muligheter for å knytte teori opp mot en praktisk kontekst. Dette valget er også personlig motivert fordi mitt første møte med tekstbasert programmeringsspråk, spesielt Python, skjedde ved ett av emnene i matematikk ved Høgskolen Østfold. Dette emnet åpnet øynene mine for potensialet programmering kan ha i å styrke matematisk forståelse og dybdelæring i dagens og fremtidens skole.

1.1 Bakgrunn for mastergradsprosjektet

Ifølge læreplanene i matematikk i skolen utgjør dybdelæring et sentralt konsept, hvor gjennom dyp forståelse og anvendelse av matematiske prinsipper står sentralt. Ifølge LK20, er dybdelæring definert som en progressiv utvikling av kunnskap og varig forståelse for faglige begreper, metoder og sammenhenger, både innenfor og på tvers av fagområder. Denne prosessen inkluderer refleksjon over egen læring og anvendelse av lært kunnskap i nye og kjente kontekster, individuelt eller i samarbeid med andre. Denne tilnærmingen understreker

viktigheten av dybdelæring i utviklingen av matematisk forståelse og ferdigheter, i tråd med de mål og intensjoner som er lagt frem i Kunnskapsløftet 2020. Med innføringen av Kunnskapsløftet 2020 (LK20) har begrepet dybdelæring, i kombinasjon med kritisk tenkning og problemløsning, fått mye fokus i matematikkfaget.

Læreplanen for matematikk legger dessuten stor vekt på utforskning i matematikk og kommunikasjon om matematikk. Den er utformet slik at elevene skal kunne koble matematikk til utfordringer i et stadig skiftende samfunn og fremtidens arbeidsliv. Algoritmisk tenkning og programmering blir også fremhevet i læreplanen som viktige strategier for problemløsning og utvikling av matematisk forståelse (Utdanningsdirektoratet, 2020).

Algoritmisk tenking, den norske oversettelsen av "computational thinking," er et nøkkelbegrep i begrunnelsen for innføringen av programmering i skolen, ifølge Utdanningsdirektoratet. De ser ikke på undervisning i programmering bare som en måte å forstå datamaskiner og digitale verktøy på, men også som en måte å ruste elever med effektive verktøy for problemløsning. Utdanningsdirektoratet definerer "algoritmisk tenking" som en problemløsningsprosess som innebærer en systematisk tilnærming til store og komplekse problemer, og inkluderer organisering og analyse av data på en logisk måte, utvikling av algoritmer, bruk av abstraksjoner og modeller, samt generalisering av løsninger (Utdanningsdirektoratet, 2018).

Den "algoritmiske tenkeren" er definert som en elev som kan bruke algoritmisk tenkning i problemløsningssituasjoner. Dette omfatter evnen til å vurdere hvordan man nærmer seg et problem, formulere det på en måte som gjør det mulig for datamaskiner å løse det helt eller delvis, og implementere relevante strategier for å løse problemet (Utdanningsdirektoratet, 2019a, 2019b).

Med LK20 har integreringen av programmering i skolehverdagen blitt forsterket. Programmering blir fremhevet som et verktøy i matematikk, hvor man lærer å anvende matematiske konsepter i programmeringskontekster. Til tross for at LK20 har vært i bruk i tre år, har jeg erfart at mange elever og lærere fortsatt strever med, eller unngår, å benytte programmering i matematikk.

I denne masteroppgaven undersøker jeg samspillet mellom programmering og dybdelæring i matematikk. Jeg undersøker hvordan programmering kan fungere som et verktøy for å fremme dybdeforståelse av matematiske konsepter.

1.2 Problemstilling

Implementeringen av LK20-læreplanen i matematikk har introdusert et skifte til trinnbaserte kompetansemål. Dette skiftet legger ifølge Utdanningsdirektoratet til rette for økt dybdelæring. Kunnskapsdepartementet (2020a) definerer dybdelæring som en gradvis utvikling av kunnskap og forståelse av begreper, metoder og sammenhenger. På 8. trinn fokuserer seks av elleve kompetansemål på algebra, variabler, likninger og funksjoner, noe som antyder en bevisst satsing på dybdelæring i algebra i løpet av det første året i ungdomsskolen (Kunnskapsdepartementet, 2020b). Dette kan ha sin bakgrunn i at internasjonale studier som TIMSS, PISA, og TIMSS Advanced siden 1995 har vist at det norske utdanningssystemet har hatt utfordringer med å forbedre resultater i algebra. Norske myndigheter har blitt kritisert for å ikke ha gjort tilstrekkelig for å adressere disse problemene. Dette blir spesielt fremhevet gitt at forskjellen mellom totalscoren og scoren i algebra i TIMSS 2015 var det største avviket i noe fagområde for noe land (Nilsen et al. 2022).

Schwab (2017) påpeker at samfunnet nå befinner seg i kjernen av den fjerde industrielle revolusjonen. Denne revolusjonen utmerker seg ikke kun ved utviklingen av nye maskiner og systemer, men også gjennom økende integrasjon av ulike teknologier. Schwab understreker at denne digitale revolusjonen berører områder som genetisk sekvensering, nanoteknologi og kvanteberegning, og fokuserer på de nye mulighetene som oppstår når disse teknologiene kombineres. I Norge ser vi også denne revolusjonen ta form, gjennom et stadig økende behov for digital ferdighet hos både voksne og barn (Schwab, 2017).

I den oppdaterte matematikklæreplanen for 1.-10. trinn, utgitt av Utdanningsdirektoratet (2020c), er programmering nå integrert som en nøkkeldel av digitale ferdigheter. Denne tilnærmingen innebærer at programmering benyttes for å prosessere informasjon, samt for utforskning og løsning av matematiske utfordringer (Utdanningsdirektoratet, 2020a). Det legges vekt på at elever skal anvende digitale verktøy effektivt for å forbedre sin digitale kompetanse. Dette inkluderer en gradvis innføring og anvendelse av teknologiske hjelpemidler for å utforske, løse, og presentere matematiske problemstillinger (Utdanningsdirektoratet, 2020a).

For å undersøke hvilke muligheter som ligger i programmering i matematikkundervisning har jeg laget to forskningsspørsmål:

Hvilket potensial for dybdelæring i algebra kan identifiseres i elevers arbeid med programmering i Python?

På hvilken måte kan hindringer gi grobunn for dybdelæring?

I dette prosjektet søker jeg innsikt i Schoenfeld (2018) og hans teori om robust learning Schoenfeld (2018). Teorien legger vekt på dyp forståelse og problemløsning i matematikk. Ved å anvende Schoenfelds perspektiv på dyp læring, blir det tydeligere for meg hvordan programmering kan bidra til å forbedre forståelsen av algebra blant elever. Dette oppnås ved å oppmuntre til aktiv deltakelse, stimulere til kritisk tenkning og fremme praktisk bruk av matematiske konsepter. Munthe (2022) støtter seg til programmering som et verktøy for å fremme matematikkundervisning. Hans forskning indikerer at bruk av programmeringsspråk som Python i matematikkfaget muliggjør at elever kan utforske algebraiske konsepter på en interaktiv måte og kan potensielt fremme dyp læring. Munthe tar også for seg den utforskende samtalen, og hvordan programmeringsoppgaver bør legge til rette for disse samtalene. Disse perspektivene vil danne grunnlaget for min tilnærming til bruk av programmering i matematikkundervisningen i min oppgave.

1.3 Oppgavens struktur

I kapittel 2.0 Teori presenterer jeg relevant teori for oppgaven. Her legger jeg frem teori som tar for seg dybdelæring, algebra og programmering.

Kapittel 3.0 Metode handler om prosjektets forskningsdesign og metodologi. I dette kapittelet diskuterer jeg utvelgelsen av informanter og detaljer rundt hvordan datainnsamlingen ble utført. Videre fremlegger jeg prosessen for analysen av de innsamlede dataene. Avslutningsvis evaluerer jeg studiens kvalitet og vurderer de etiske aspektene knyttet til forskning på elevers oppgaveløsning med video- og lydopptak.

I kapittel 4.0 Resultat presenterer jeg fire nøkkelfunn som er avdekket gjennom analyseringen av data.

Dybdeforståelse av algebra: Analyser viser at programmering kan hjelpe elevene til å utvikle en dypere forståelse for algebraiske konsepter ved å anvende dem i løsning av praktiske problemer.

Overvinning av teknologiske barrierer: Funnene indikerer at selv om teknologiske barrierer opprinnelig kan hindre læring, kan overvinning av disse barrierene faktisk styrke forståelsen og anvendelsen av matematikk.

Fremme av matematisk samtale: Programmeringsaktivitetene har tilrettelagt for å fremme en utforskende matematisk samtale blant elever, som har vist seg å være kritisk for deres forståelsesprosess.

Agens og eierskap i læreprosessen: Elever viser en økt følelse av agens og eierskap over sin læreprosess, noe som kan motivere og engasjere dem i større grad. Disse funnene støttes med konkrete eksempler hentet direkte fra det innsamlede datamaterialet.

I kapittel 5 Diskusjon diskuterer jeg funnene som ble presentert i kapittel 4, med fokus på å diskutere potensielle årsaker og konsekvenser. Dette vil inkludere en integrasjon av relevant teori fra kapittel 2. Videre vil jeg diskutere hvordan disse funnene relaterer seg til og belyser forskningens problemstilling.

I det avsluttende kapittelet 6.0 Konklusjonen oppsummerer jeg svar på forskningsspørsmålene behandlet i studien. Jeg drøfter der også de teoretiske og praktiske implikasjonene av studiens funn. Avslutningsvis diskuterer jeg mulige fremtidige forskningsretninger og foreslå områder for videre undersøkelser innen dette feltet.

2.0 Teori og tidligere forskning

Med bakgrunn i at jeg skal se på hvilket potensial for dybdelæring i algebra vi kan identifisere i elevers arbeid med programmering, har jeg valgt å strukturere kapittelet i fire deler. Den første delen vil omhandle teorier og forskning relatert til dybdelæring. Siden temaet elevene skal arbeide med er algebra fokuserer jeg på teorier og forskning knyttet til algebra og algebraens rolle i skolen i andre delen. Den tredje delen handler om programmering, så vil jeg ta for meg det teoretiske rammeverket som vil være grunnlaget for denne masteroppgaven. Til slutt vil jeg diskutere utvalgte artikler som utforsker både programmering og dybdelæring, hvorav to av disse er basert på tidligere forskning.

Teorien og tidligere forskning i dette kapittelet har to hensikter. Det danner bakgrunn for valg og utvikling av oppgaver som jeg skal gi til elevene når jeg innhenter data, og det gir en teoretisk bakgrunn for analyse av data og for diskusjon av funn.

2.1 Dybdelæring

Begrepet "dybdelæring" ble først innført for over 40 år siden i sammenheng med en studie utført av Ference Marton og Roger Säljö (1976). De introduserte begrepene "overflatelæring" og "dybdelæring". Studien avslørte at elever som praktiserte overflatelæringsstrategier i hovedsak fokuserte på å memorere regler og fakta, uten å anerkjenne deres større sammenhenger. Motsatt viste elever som engasjerte seg i dybdelæring en dypere interesse for å forstå og sette sammen ulike elementer i sin læring. Denne gruppen elever viste også en indre motivasjon for å oppnå en dypere forståelse, som gikk utover enkle akademiske prestasjoner.

Som påpekt av Sawyer (2006), har samfunnet gjennomgått betydelige forandringer på grunn av den teknologiske revolusjonen. Disse transformasjonene har ført til økt kompleksitet i samfunnet, og som et resultat kreves det ny kunnskap. Som for eksempel å ha evnen til å forstå og bruke digitale verktøy og plattformer effektivt er essensielt i et samfunn der teknologi spiller en stadig større rolle. Siden kritisk tenkning og problemløsning er nødvendig i en verden med komplekse og raskt skiftende problemer, er det å kunne tenke kritisk og løse problemer på kreative måter gode verktøy å ha med seg. Med kontinuerlig utvikling i teknologi og endring av arbeidsmarkedsbehov, er evnen til å lære og tilpasse seg nye ferdigheter gjennom hele livet avgjørende. I denne konteksten blir utdanningssektorens rolle avgjørende, da den er ansvarlig for å utdanne fremtidige arbeidstakere og samfunnsborgere til

det moderne, komplekse samfunnet. Dette understreker behovet for at elever utvikler dybdelæringsferdigheter.

2.1.1 Dybdelæring i matematikk

Ifølge Brekke og Gjone (2001), er assosiasjonen med ordet "matematikk" ofte begrenset til begreper som regning, symboler og formler, mens evnen til å oppdage mønstre og forstå sammenhenger ofte glemmes. Dette kan delvis tilskrives den konvensjonelle tilnærmingen til matematikkundervisning i skoler, der elevene typisk arbeider med symboler etter gitte regler og formler uten nødvendigvis å reflektere over symbolenes betydning.

Læring i matematikk fokuserer på konstruksjonen av et nettverk av forbindelser som integrerer matematiske konsepter, problemer, og personlige erfaringer poengterer Noss og Hoyles (1996, s. 105). Denne tilnærmingen er i harmoni med Brousseaus (1997) ideer om adidaktiske situasjoner, hvor læring skjer gjennom selvoppdagelse og utforskning, uten direkte instruksjon fra læreren. Dette kommer jeg tilbake til senere.

Munthe (2022) kommer opp med tre funn i sin forskning på programmering i matematikk. I min masteroppgave vil jeg bruke disse funnene til å designe oppgaver/undervisningsopplegg som legger til rette for matematiske samtaler, noe jeg utdyper i metode kapittelet. Munthes funn var:

1. Når programmering implementeres i matematikklasserom, kan det legge til rette for matematisk utforskende samtale.
2. Programmering er best implementert for å legge til rette for dyp læring av allerede kjente matematiske konsepter, da dette initierer utforskende samtale; imidlertid er forsiktighet nødvendig når man bruker programmering for å lære nye matematiske konsepter.
3. Motgang er både viktig og utfordrende når man implementerer programmering i matematikklasserom. Det er viktig i den forstand at det kan legge til rette for matematiske adidaktiske situasjoner, men utfordrende fordi programmering legger til et ekstra lag av kompleksitet (Munthe, 2022, s.141).

Munthe (2022) sin forskning baserer seg mye på Brousseau sin ¨Teori om Didaktiske situasjoner¨ (TDS). I matematikkdidaktikk er det et viktig begrep som kalles den didaktiske kontrakten. Dette er avtalen mellom læreren og elevene i klasserommet. Den innebærer visse forventninger om hva elevene skal lære og hvordan de skal lære det. Elevene forventer å lære matematikk ved å jobbe med oppgavene som læreren gir dem, mens læreren forventer at elevene vil følge instruksjonene og fullføre oppgavene. Når elevene jobber med oppgavene, skjer det en interaksjon mellom dem og læringsmiljøet. Denne perioden kalles den adidaktiske situasjonen, der elevene tar ansvar for sin egen læring. Før den adidaktiske situasjonen, kan det være en didaktisk situasjon der læreren eller oppgaven presenterer et problem som skal løses. Etter den adidaktiske situasjonen kommer en annen didaktisk situasjon der læreren eller oppgaven knytter den oppnådde kunnskapen til målet med oppgaven. Brousseau (2008) argumenterer for at både den foregående og påfølgende didaktiske situasjonen kan være en del av oppgavedesignet. Dette gjelder spesielt for den påfølgende situasjonen der oppgaven kan hjelpe til med å diskutere løsningen i en større sammenheng. Lærerens evne til å forstå hva elevene gjør og tilpasse undervisningen er viktig, og dette kalles institusjonalisering av kunnskapen elevene skal tilegne seg (Brousseau, 2008). Institusjonalisering av kunnskap refereres til lærerens evne til å forstå og integrere elevenes handlinger og læring i undervisningen. Dette innebærer at læreren kan gjenkjenne hva elevene har lært og hjelpe dem med å knytte denne kunnskapen til videre læring eller større konsepter innenfor matematikk. Institusjonalisering av kunnskap er viktig for å sikre at elevene ikke bare løser oppgaver, men også forstår konseptene bak dem og kan anvende denne kunnskapen i ulike sammenhenger. Det bidrar til en dypere og mer varig læring.

Utfra Munthes sitt resultat og teorien fra Brousseau ser det ut som at gjennom programmering i Python, kan det legges til rette for å engasjere elevene i algebra på en praktisk og interaktiv måte. Dette kan også styrke elevenes evne til kritisk tenkning og problemløsning. Bruken av utforskende arbeidsmetoder og dialog innen programmering gir rom for elevdrevet læring, hvor de kan utforske, eksperimentere og lære av hverandre. Denne tilnærmingen kan potensielt transformere måten algebra blir oppfattet og lært på, og jeg er spesielt interessert i å utforske hvordan denne metoden kan bidra til en dypere forståelse og anvendelse av algebraiske konsepter i den virkelige verden.

2.2 Algebra

Algebra, en fundamental gren av matematikk, har utviklet seg betydelig gjennom historien. Fra å være en metode for å løse ligninger ved hjelp av ord, utviklet det seg til dagens form, preget av bokstaver og symboler, er algebraens formål og anvendelse transformert. Denne utviklingen, beskrevet av Aubert (2009) og Carraher & Schliemann (2018), omfatter overgangen fra en retorisk til en symbolsk fase.

Algebraens historiske utvikling kan deles inn i tre faser: retorisk, synoptisk, og symbolsk, som beskrevet av Carraher og Schliemann (2018). Den symbolske fasen, popularisert av René Descartes, representerer en avgjørende overgang til bruk av bokstaver for å representere kjente og ukjente størrelser.

Elevers forståelse av algebra er kompleks, som vist i Küchemanns (1978) studie. Elever har ulike oppfatninger av bokstaver, fra kjente verdier til variabler. Usiskin (1988) påpeker at variabelbegrepet ofte misforstås, og understreker behovet for en dypere forståelse av variabler i skolematematikken.

Küchemanns forskning indikerer at elevers forståelse av bokstaver som variabler er dårlig. Ofte forstår elevene bokstavene som konkrete, uforanderlige objekter eller spesifikke tall, snarere enn som symboler som representerer varierte og potensielt ubestemte verdier. Denne misforståelsen kan hindre deres evne til å manipulere og løse algebraiske ligninger. Dette antyder til et behov for en pedagogisk tilnærming som fokuserer mer på metaforståelse av variabler. Usiskin (1988) foreslår å introdusere funksjoner tidligere i undervisningen for å fremme en bedre forståelse av avhengige og uavhengige variabler. Dette har stor betydning for mitt prosjekt da det peker på en viktig utfordring innen matematikkundervisning knyttet til algebra. Küchemanns forskning indikerer at elevers forståelse av variabler er mangelfull, og at det er behov for en pedagogisk tilnærming som fokuserer på metaforståelse av variabler. Dette understreker viktigheten av å utvikle undervisningsmetoder som hjelper elever med å forstå konseptet med variabler på en dypere og mer meningsfull måte.

Blanton og Kaput (2005) poengterer at algebra er en aktivitet hvor man bruker bokstaver som står for tall for å generalisere matematiske ideer og representere funksjonelle relasjoner. Denne forståelsen av algebra som en studie av funksjoner, relasjoner og avhengig variasjon er spesielt relevant i programmering. I programmering brukes variabler for å håndtere data som kan endre seg over tid eller mellom kjøringer av programmet, akkurat som variabler i algebra

brukes til å representere generelle eller ukjente verdier i en ligning. Dette tillater programmerere å skrive fleksibel og gjenbrukbar kode som kan håndtere ulike inngangsdata.

Programmering gir en unik mulighet til å anvende og utforske algebraiske konsepter i en praktisk sammenheng. Det tradisjonelle bildet av algebra, som sentrerer rundt å løse likninger og ulikheter gjennom symbolmanipulasjon, som beskrevet av Watanabe (2011), kan utvides i en programmeringskontekst. Her kan elever anvende og eksperimentere med algebraiske ideer i sanntid, se konsekvensene av deres handlinger umiddelbart, og dermed kunne utvikle en dypere forståelse for de underliggende prinsippene.

Kaput og Blanton (2017) beskriver algebra som et system bestående av fem tråder, hvorav en viktig tråd er generalisering fra numeriske og geometriske mønstre. De fem trådene er:

- **Symboler og Uttrykk:** Denne tråden fokuserer på forståelse og bruk av algebraiske symboler og uttrykk. Elever lærer hvordan å tolke og manipulere disse symbolene for å representere matematiske forhold og ideer.
- **Ligninger, Ulikheter og Funksjoner:** Her lærer elever om ligninger og ulikheter, og hvordan disse kan brukes til å modellere og løse problemer. De lærer også om funksjoner som en fundamental del av algebraisk tenkning.
- **Generalisering fra Numeriske og Geometriske Mønstre:** Dette området handler om å gjenkjenne, forstå, og bruke mønstre for å lage generaliseringer. Elever utforsker hvordan numeriske og geometriske mønstre kan representere komplekse ideer og relasjoner.
- **Rekursive Relasjoner:** Denne tråden fokuserer på forståelsen av rekursive prosesser, hvor et uttrykk er definert i forhold til seg selv. Dette er viktig i forståelsen av sekvenser og serier i matematikk.
- **Strukturert Tenkning og Argumentasjon:** Den siste tråden legger vekt på utviklingen av logisk og strukturert tenkning. Det inkluderer å utvikle argumenter og bevis, samt å forstå og bruke matematisk logikk og resonnement. (Kaput og Blanton, 2017, s.5-15, min oversetting).

Disse trådene fremmer en dypere forståelse av algebra som et dynamisk og anvendelig verktøy i matematikk. I programmering kan elever utforske algebra ved å skape og manipulere figurmønstre og observere mønstre i data. Dette kan hjelpe elever til å forstå rekursive relasjoner og hvordan de kan representere slike mønstre i kode. Rekursive relasjoner er prosesser eller sekvenser der etterfølgende ledd defineres ut fra de foregående. Denne

tråden i algebraen fokuserer på å forstå og anvende slike sekvenser, noe som er viktig i mange matematiske og realfaglige sammenhenger. Denne forståelsen er spesielt relevant i informatikk, der rekursive algoritmer ofte brukes til å løse komplekse problemer (Kaput og Blanton, 2011).

Slik integrerer programmering de grunnleggende kjerneelementene som abstraksjon og generalisering til praktisk anvendelse av matematiske funksjoner og mønstre (LK20). Gjennom denne tilnærmingen kan elever oppnå en dypere forståelse av algebra, ikke bare som en teoretisk studie, men som et anvendelig verktøy.

2.2.1 Figurmønstre

I denne masteroppgaven har jeg valgt å se på figurmønstre som en tilnærming til algebra. Figurmønster er et av flere temaer innenfor algebra, som reflekteres i uttrykket "Patterns are the heart and soul of mathematics" (Zazkis & Liljedahl, 2002, s. 379). Disse mønstrene gir en kontekst hvor elever blir bedt om å formulere regler som kan brukes til å identifisere andre tilfeller av det samme mønsteret (Lannin, 2005).

I slike oppgaver brukes begrepet "figurtall" for å beskrive elementer i figurer eller diagrammer. Et viktig aspekt ved figurmønsteroppgaver er inkluderingen av en variabel som endres, noe som gir elever muligheten til å observere, verbalisere og representere generaliseringene symbolsk (Zazkis & Liljedahl, 2002). Dette oppmuntrer elevene til å utvikle algebraiske ferdigheter ved å finne generelle regler som kan beskrive hele mønsteret.

Cooper og Warren (2008) argumenterer for verdien av å arbeide med figurmønster, da det ikke bare hjelper elever med å identifisere relasjoner og mønstre, men også oppfordrer dem til å formulere generelle regler som kan beskrive hele følgen. Lannin (2005) fremhever visuelle elementer ved figurmønsteroppgaver, som hjelper elever med å beskrive og forklare regelmessigheter som gjelder for alle figurer i mønsteret. Dette visuelle elementet kan være en verdifull ressurs for elever når de skal forstå hvordan komponentene i et algebraisk uttrykk representerer elementene i et figurmønster.

I følge Lannin (2005) er figurtall en effektiv måte å introdusere algebra på fordi de gjør det mulig for elever å visualisere og utforske matematiske mønstre og relasjoner på en konkret måte. Ved å arbeide med figurtall, blir elevene oppfordret til å identifisere mønstre og utvikle generaliseringer om hvordan tallene er organisert og vokser. Dette er en kjernekomponent i

algebraisk tenking – evnen til å se utover de spesifikke tallene og forstå de underliggende strukturene og relasjonene som definerer dem. Gjennom mønsteraktiviteter som involverer figurtall, lærer elevene å formulere uttrykk som beskriver disse mønstrene, noe som kan føre til en dypere forståelse av variabler, uttrykk, og ligninger – nøkkelelementer i algebra. Disse aktivitetene bygger bro mellom aritmetisk forståelse og algebraiske konsepter ved å gjøre overgangen fra konkret, visuell manipulering av tall til mer abstrakt symbolsk representasjon.

2.3 Programmering

Mange land har inkorporert programmering i sine nasjonale læreplaner, og ofte blir programmering knyttet sammen med matematikk (Bocconi et al., 2018). Ideen om å inkludere programmering i matematikkundervisningen er ikke ny. Allerede på 1980-tallet argumenterte Papert (1980) for bruk av programmering som et verktøy for å lette læringen hos elever. Til tross for dette mislyktes implementeringen, delvis på grunn av begrensninger i å utnytte programmering som et læringsverktøy i matematikk (Misfeldt and Ejsing-Duun, 2015). Det er nå en økende interesse for forskning som utforsker hvordan programmering kan integreres i matematikkundervisningen på skoler, og det er mange aspekter av denne endringen som er under nærmere undersøkelse (Benton et al., 2016). En viktig del av implementeringen av programmering i matematikkundervisningen er designet av oppgaver. Denne masteroppgaven ser på hvordan oppgavene generer til samtale og om det kan identifiseres en dypere forståelse av den matematiske kunnskapen i samtalene.

I de nordiske landene er det nå en økende tendens til å inkludere programmering i matematikkundervisningen (Bocconi et al., 2018). Denne sammenkoblingen av matematikk og programmering åpner opp en rekke muligheter og utfordringer for læring. Noen av de positive aspektene inkluderer bruken av numeriske metoder og simuleringer, som gir elever nye verktøy for å utforske og forstå matematiske konsepter på en dypere måte. I tillegg gir denne tilnærmingen muligheten til å transformere undervisningen ved å introdusere innovative metoder for å engasjere og inspirere elevene til å utforske matematikk på en mer praktisk måte.

Seymour Papert (1980) skriver i *MINDSTORMS: Children, Computers, and Powerful Ideas* at programmering er prosessen med å skrive instruksjoner som en datamaskin kan forstå og utføre. Disse instruksjonene kalles kode, og de skrives vanligvis i et programmeringsspråk som er utviklet for å kommunisere med datamaskiner. Programmering kan brukes til å lage alt

fra enkle programmer til komplekse applikasjoner og systemer. Å lære å programmere kan gi en dypere forståelse av datamaskiner og teknologi. Det kan også hjelpe med å utvikle ferdigheter som problemløsning, logisk tenkning og kreativitet.

2.3.1 Programmering i skolen

Grunnen til at jeg vil se på tidligere forskning i klasserommet er for å få et innblikk i implementeringsstrategien og forstå hvordan programmering har blitt formidlet til elever tidligere. Dette inkluderer å utforske Teorien om Didaktiske Situasjoner (TDS) og didaktiske undervisningsopplegg, spesielt med tanke på bruk av Python i matematikkundervisning. Det er viktig å forstå både de pedagogiske mulighetene og utfordringene knyttet til å integrere programmering i utdanningssektoren, og hvordan dette kan påvirke elevers læring og forståelse av matematiske konsepter.

Kaufmanns og Stenseth sin artikkel fra (2020) fremhever viktigheten av programmering i dagens samfunn, og hvordan dette gjenspeiles i læreplaner, særlig i Europa. Målet er å styrke elevers algoritmiske tenkning, og programmering blir ofte integrert med andre fag, spesielt matematikk. Resultatene viser at programmering kan forbedre elevers matematikk-kunnskaper, øke deres motivasjon, og styrke deres problemløsningsferdigheter. Imidlertid påpeker artikkelen også utfordringer ved implementeringen av programmering i skolefag og understreker behovet for å forstå hvordan elever anvender programmering for å sikre at de tilpasses effektivt til disse endringene. Disse endringene og tilpasningene i utdanningssystemet speiler en anerkjennelse av teknologiens rolle i samfunnet og nødvendigheten av å utstyre kommende generasjoner med de ferdighetene de trenger for å navigere og lykkes i et digitalisert arbeidsmiljø og samfunn.

Selv om det har vært betydelige fremskritt innen programmeringsverktøy og en økende forståelse for programmering, eksisterer det fortsatt en utfordring med å forene denne tekniske kunnskapen med pedagogisk kompetanse innen programmering. Både lærere og elever kan ha en begrenset oppfatning av programmering, ofte begrenset til å følge trinnvise instruksjoner i tilgjengelige læringsverktøy (Benton et al., 2016). For å forbedre kvaliteten på undervisning som involverer programmering, er det nødvendig å identifisere og implementere pedagogiske verktøy. Disse verktøyene vil støtte lærere i å designe og tilpasse sine egne undervisningsopplegg til forskjellige temaer innen matematikk.. Benton et al. (2016) understreker viktigheten av at lærere forstår det sentrale målet med programmering, nemlig

kraften i algebraisk tenkning, og hvordan dette kan integreres i matematikkundervisningen. Clement (1999) sin forskning viser at programmering i skolen har potensiale til å hjelpe elever med å utvikle avansert matematisk tenkning, inkludert abstraksjon, algebraisk tenkning og problemløsning.

Ifølge Brousseau (1997) oppstår verdifull matematisk læring ofte når elevene engasjerer seg i å løse problemer. I teorien om didaktiske situasjoner (TDS) betraktes kunnskap som en egenskap i et system som består av både en aktør (eleven) og et miljø (inkludert problemet, programmeringsspråket og gruppen med elever). Dette miljøet skaper rammer der interaksjonen mellom elevene og kunnskapen forekommer. Ved å utforske potensialet til algebraisk dyp læring gjennom Python-programmering, skapes et læringsmiljø der interaksjoner mellom elever og matematiske konsepter blir tydeliggjort og forbedret. Python-programmering fungerer som en katalysator for denne interaksjonen, og transformerer tradisjonell matematikklæring til en dialog mellom elever og det digitale miljøet. Elevene engasjerer seg ikke bare i å løse matematiske problemer, men kan også utvikle forståelse av grunnleggende algebraiske prinsipper gjennom koding. Denne tilnærmingen stemmer overens med Brousseaus (1997) utdanningsteori, som understreker viktigheten av aktive læringsroller. Elevene blir ikke bare mottakere av kunnskap, men også aktive deltakere i å bruke programmering til å utforske, eksperimentere og oppdage matematiske konsepter. I prosessen får elevene umiddelbar tilbakemelding fra programmeringsmiljøet, og skaper en dynamisk og interaktiv læringsopplevelse som fører til en dypere forståelse og mestring av algebra.

2.3.3 Algoritmisk tenking

Dette kapittelet undersøker hvordan elever kan bruke programmering som et verktøy for å løse matematiske problemer, noe som strekker seg utover det tradisjonelle rammeverket for matematisk problemløsning. Et sentralt konsept i denne sammenhengen er "computational thinking," eller "algoritmisk tenkning" som det er kjent som i Norge.

Wing (2006) satte et sterkt fokus på "computational thinking," og argumenterte for at forståelsen av programmering kunne utstyre mennesker med ferdigheter til å forbedre problemløsningsevner og hjelpe med å forstå og løse problemer i den virkelige verden. Wing beskrev dette som en måte mennesker løser problemer på, ikke et forsøk på å få mennesker til å tenke som datamaskiner. I 2017 redefinerte Wing begrepet computational thinking som "tankeprosesser involvert i å formulere et problem og uttrykke løsninger på en måte som gjør

at en datamaskin, enten den er menneskelig eller mekanisk, kan utføre den effektivt" (Wing, 2017).

Csizmadia et al. (2015) har imidlertid en annen tilnærming til definisjonen av "computational thinking." De utviklet et rammeverk for dette i det engelske skolesystemet, beskrevet som kognitive prosesser som hjelper elever med å løse problemer, forstå prosedyrer og utvikle systemforståelse. Dette inkluderer konsepter som logisk tenkning, algoritmisk tenkning, problemdelingsferdigheter, generalisering, mønsteridentifisering, abstraksjon, valg av passende representasjoner og evnen til å evaluere. Dette rammeverket sammenfaller mye med Utdanningsdirektoratets modell for "computational thinking" i norsk skole (Utdanningsdirektoratet, 2020b).

I følge Kaufmann et al. (2018) handler algoritmisk tenking om å kunne tenke systematisk på algoritmer og hvordan datamaskiner kan hjelpe til med å løse komplekse problemer. Gjennom programmering lærer elevene seg å tenke algoritmisk, det vil si å bryte ned problemer i mindre biter, utvikle løsningsstrategier og implementere disse strategiene ved hjelp av koding. Algoritmisk tenking bidrar til å utvikle elevenes evne til å analysere, forstå og løse problemer på en strukturert måte, samt å kunne abstrahere og generalisere løsninger for å kunne løse lignende problemer i fremtiden.

Mange skoleelever begynner med begrenset programmeringskunnskap og ofte tar de en "linje-for-linje" tilnærming i stedet for å kunne bygge meningsfulle programmer (Lahtinen et al., 2005). Linje-for-Linje tilnærming refererer til en metode hvor elever fokuserer på å skrive og forstå kode på et veldig grunnleggende, linje-for-linje-nivå. De konsentrerer seg ofte om hva hver enkelt linje med kode gjør, i stedet for å forstå den bredere konteksten eller den overordnede strukturen og logikken i programmet. Med denne tilnærmingen kan elevene miste det overordnede bildet av programmering. Programmering handler ikke bare om å få hver linje med kode riktig; det handler om hvordan disse linjene fungerer sammen for å løse et problem eller utføre en oppgave. En linje-for-linje-tilnærming kan føre til en fragmentert forståelse hvor det overordnede formålet og funksjonaliteten til programmet ikke er fullt ut forstått.

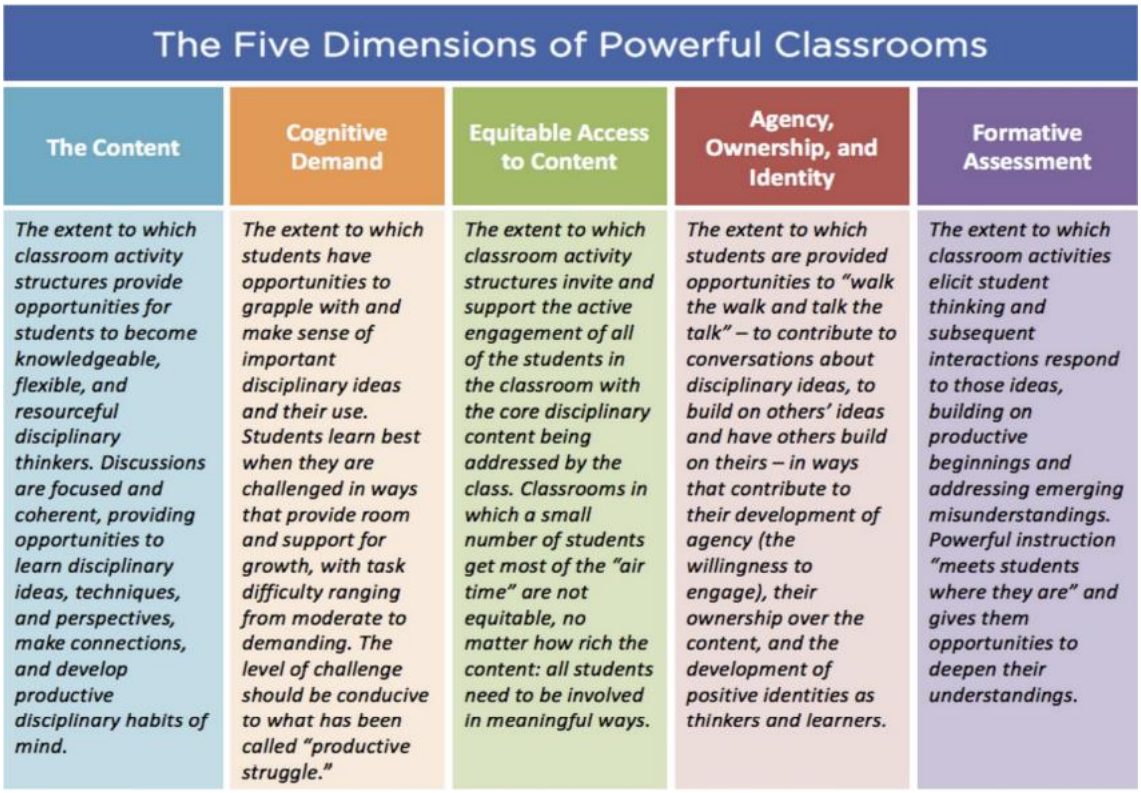
2.4 Teoretisk rammeverk

Målet for å inkludere dette teoretiske rammeverket i min oppgave er å legge grunnlaget for en dypere forståelse av hvordan elevers arbeid med Python-programmering i algebra kan bidra til dybdelæring. Ved å integrere både Schoenfelds konsept om robust læring og ideer om dybdelæring, kan jeg undersøke og identifisere det potensialet som ligger i krysningen mellom programmering og matematikkundervisning. Schoenfelds rammeverk om robust læring, med fokus på elevenes agens, eierskap, og identitet, sammen med strategier for å fremme kritisk tenkning og problemløsning, tjener som en veiledning for å identifisere hvordan elever engasjerer seg med og mestrer matematiske konsepter gjennom programmering. Samtidig, ved å anerkjenne og anvende generaliseringsstrategier i mønsteroppgaver (Lannin, 2005), legges det opp til en undervisningsmetode som understøtter elevers utvikling av algebraisk tenkning, noe som er essensielt for dybdelæring i matematikk. Dette delkapittelet vil også adressere barrierer og utfordringer elever kan møte i programmeringsprosessen, og hvordan disse kan overkommes for å støtte læring. Ved å kombinere disse teoriene, står vi bedre rustet til å svare på forskningsspørsmålet: "Hvilket potensiale for dybdelæring kan identifiseres i elevers arbeid med Python-programmering i algebra?"

2.4.1 Robust learning (agens, eierskap og identitet)

Innen utdanningsforskning er både Schoenfelds teori om "robust læring" og begrepet "dybdelæring" viktige tilnærminger som tar sikte på å fremme dyp forståelse og kompetanse hos elever (Schoenfeld, 2018). Selv om de deler visse likheter, er det også betydelige forskjeller som skiller dem fra hverandre. Schoenfelds robuste læring er spesifikt tilpasset matematikkundervisning og tar for seg detaljerte aspekter av hvordan matematikk bør undervises for å fremme forståelse og læring (Schoenfeld, 2018). Dybdelæring er derimot en bredere tilnærming som kan anvendes på tvers av forskjellige fagområder og ikke er bundet til et spesifikt fag.

Hovedtrekkene i Schoenfeld sin teori inkluderer solid forståelse av matematiske konsepter, utvikling av sterke problemløsnings- og resonneringsferdigheter, oppfordring til aktiv deltakelse i reelle problemer, og fleksibel anvendelse av kunnskap i ulike sammenhenger (Schoenfeld, 2018).



Figur 1 The five dimensions of robust classrooms—the teaching for robust understanding (TRU) framework (Schoenfeld, 2018, s.3)

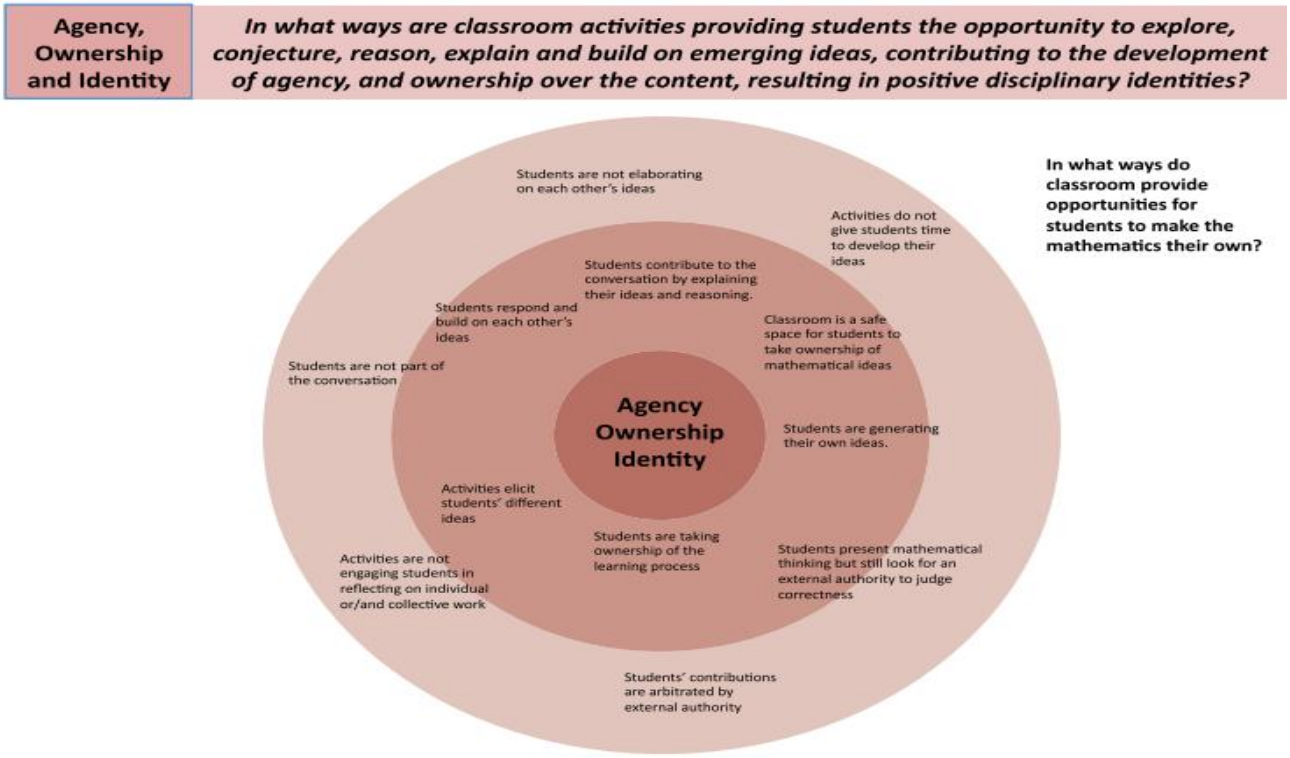
- **Innholdet:** Vektlegger muligheter for at elever forstår og tenker kritisk innen fagområdet. Det fokuserer på å fremme kunnskapsrike, fleksible og ressurssterke faglige tenkere.
- **Kognitiv etterspørsel:** Oppmuntrer til oppgaver som fremmer dyp forståelse og aktivt engasjement med innhold, som støtter elevenes vekst gjennom "produktiv kamp".
- **Likeverdig tilgang til innhold:** Sikrer at alle elever har meningsfulle muligheter til å engasjere seg med faglige konsepter og praksiser, og adresserer mangfoldet av elever.
- **Agens, Eierskap og Identitet:** Fokuserer på muligheter for at elever kan uttrykke sine tanker og bygge sin identitet i faget, og fremmer engasjement og eierskap over innholdet.
- **Formativ vurdering:** Innebærer å bruke klasseromsaktiviteter for å forstå elevenes tenkning og læringsprogresjon, som muliggjør undervisning som møter deres behov og fordyper forståelse (Schoenfeld, 2018, s.3, min oversetting).

Mange av prinsippene som er nevnt her, legger til rette for dybdelæring, spesielt fokuset på å forstå sammenhenger, oppfordring til elever til å forklare og begrunne sine tanker,

oppmuntring til samarbeid og lærerens rolle i å bygge videre på elevenes ideer. Ved å oppmuntre til kritisk tenkning, oppfordres elever til å utforske og reflektere over de underliggende prinsippene og konseptene som de lærer. Dette kan styrke deres evne til å se sammenhenger mellom forskjellige ideer og anvende disse forståelsene i nye sammenhenger. For dybdelæring betyr dette at elever ikke bare lærer isolerte fakta, men kan utvikle en forståelse av hvordan faglige konsepter er sammenvevd og kan anvendes praktisk. TRU-rammeverket inneholder fem viktige dimensjoner som bidrar til å gi elevene muligheten til å oppnå en solid forståelse. Spesielt vil jeg fremheve dimensjonen kalt "agens, eierskap og identitet."

Ifølge Schoenfeld (2018) er det viktig at læringsmiljøer som tar sikte på å fremme dyp forståelse blant elevene, gir rom for:

> Agens: Elevenes aktive deltakelse og engasjement i læringsprosessen.
> Eierskap: Elevenes følelse av eierskap og ansvar for sin egen læringsstrategi.
> Identitet: Oppmuntring av elevenes selvbilde som kompetente matematikklærende (Schoenfeld, 2018, s.3, min oversetting).



Figur 2 A ¨bull's-eye¨ representation of the richness of various activities. (Schoenfeld, 2018, s.14)

I Schoenfeld (2018) sin studie, kommer han frem til en figur som han kaller ¨A bull's-eye representation of the richness of various activity structures¨. En "bull's-eye" representasjon kan brukes til å visualisere ulike nivåer av kompleksitet, dybde eller rikdom i ulike aktivitetsstrukturer. I undervisningssammenheng kan denne typen figur bidra til å vise hvordan ulike undervisningsmetoder eller aktiviteter påvirker elevenes evne til å ta eierskap til sin læring (ownership), utvikle sin identitet som lærende (identity), og ta ansvar for sin egen læringsprosess (agency). Ved å plassere ulike aktivitetsstrukturer i ulike "ringer" av bull's-eye'en, kan man illustrere hvordan noen aktiviteter kan være mer effektive enn andre når det gjelder å fremme disse aspektene av læring. Aktiviteter som ligger utover ytterkanten av bull's-eye'en kan indikere en lavere grad av studentengasjement og eierskap, mens aktiviteter som ligger nærmere sentrum kan symbolisere mer dyptgående og meningsfulle læringsopplevelser.

Ved å analysere og reflektere over denne typen visualisering, kan man få innsikt i hvordan de ulike aktivitetene brukt i undervisningen kan påvirke elevenes opplevelse av matematikk og deres evne til å ta ansvar for sin egen læring. Ved å ta i utgangspunkt av bull's eye, kan dette bidra til å tilpasse undervisningen for å fremme en dypere forståelse og engasjement blant elevene.

I min forskning på hvilket potensial for dybdelæring i algebra kan identifiseres i elevers arbeid med programmering i Python blir det tydelig at både sosiale og emosjonelle aspekter ved læring er viktige. Min interesse ligger i hvordan engasjement, holdninger, og troen på egen mestring, i kombinasjon med matematiske samtaler og følelsen av eierskap, gir muligheter for at elevene kan få en dypere forståelse. Schoenfelds (2018) vektlegging av elevenes deltakelse i diskusjoner om matematiske ideer resonnerer med mitt syn på læring som en aktiv, deltakende prosess.

Imidlertid definerer Schoenfeld robust læring som en prosess hvor elever anvender matematikk i nye situasjoner og utvikler en dypere forståelse. Han legger vekt på kognitiv utfordring og produktiv kamp, noe som innebærer at elevene utforsker og engasjerer seg i matematiske problemer på en meningsfull måte (Schoenfeld, 2018). Schoenfelds vektlegging av diskurs i læreprosessen resonnerer med Noss og Hoyles' (1996) syn på kommunikasjon og samarbeid i matematikklæring.

I lys av min problemstilling om integrering av programmering i matematikkundervisning, fremstår tilnærmingene til Noss, Hoyles og Schoenfeld som relevante. Disse pedagogiske perspektivene legger vekt på viktigheten av å utvikle en dyp forståelse og et meningsfullt engasjement i matematikklæring. Programmering kan bidra til kognitive utfordringer i matematikkundervisningen slik at en intellektuelt utfordrere elevene slik Noss og Hoyles' (1996) samt Schoenfelds (2018) setter fokus på. Ved å anvende programmering i matematikk, stimuleres elever til å tenke kritisk og løse problemer på nye måter. Integreringen av programmering kan også fremme en læringskultur som er preget av respekt, åpenhet og aktiv deltakelse. I programmeringsprosessen oppfordres elever til å samarbeide og dele ideer, noe som kan styrke denne kulturen og fremmer et mer inkluderende og interaktivt læringsmiljø. I tillegg gir programmering en mulighet til å koble matematisk teori med praktisk anvendelse. Elever får muligheten til å anvende og se relevansen av matematiske konsepter i programmeringsoppgaver, noe som kan øke deres forståelse og evne til å anvende matematikk i praktiske situasjoner. Programmering tillater elever å integrere sine personlige erfaringer og interesser i læringen. Dette kan gjøre matematikk mer engasjerende og relevant for elever, noe som er viktig for å oppnå en dypere og mer meningsfull læring.

Sammenfattet ut fra hva Noss & Hoyles (1996) og Schoenfelds (2016) kan integreringen av programmering i matematikkundervisning bidra til å oppnå pedagogiske mål om dybde forståelse og meningsfylt engasjement.

2.4.2 Generaliseringsstrategier

I denne oppgaven undersøker jeg betydningen av generaliseringsstrategier i matematikkundervisningen, med fokus på hvordan disse strategiene støtter elevers utvikling av algebraisk tenkning og dybdelæring. Generaliseringsstrategier er avgjørende for å hjelpe elever å forstå og anvende matematiske mønstre og strukturer, og spiller en rolle i overgangen fra konkret tallbehandling til abstrakt algebraisk resonnement.

Ved å fremme kritisk tenkning og problemløsning, legger generaliseringsstrategier til rette for at elever kan engasjere seg i utforskninger og diskurser rundt matematiske konsepter. Dette kan bidra til en dypere forståelse av variabler og algebraiske uttrykk, som er essensielt for både matematisk og programmerings kompetanse. Videre kan disse strategiene støtte utviklingen av elevers evne til å kommunisere matematisk og det kan forsterke deres forståelse av hvordan teoretiske konsepter anvendes i praktiske situasjoner.

I Lannins studie (2005), ble generalisering blant sjetteklassinger undersøkt. Han analyserte elevenes resonnementer mens de arbeidet med "patterning activities," som er oppgaver som involverer mønsterdannelse. I disse oppgavene skulle elevene identifisere den avhengige variabelen i forhold til endringer i den uavhengige variabelen, for eksempel antall fyrstikker som trengs for å lage én rute, to ruter, tre ruter, og så videre. Til slutt skulle elevene formulere en generell formel som beskrev mønsteret og gi en begrunnelse for formelen.

Lannin (2005) henviste til retningslinjer fra USA, Storbritannia og Australia, som anbefalte bruken av mønsteroppgaver som en måte å introdusere algebraiske konsepter på. Han argumenterte for at denne tilnærmingen gir en kontekst som støtter elevenes forståelse av symbolske representasjoner og samtidig knytter seg til deres tidligere kunnskap innen aritmetikk. I Lannin (2005) sin studie ble også ulike strategier for generalisering utforsket. Han identifiserte fem forskjellige generaliseringsstrategier og undersøkte elevenes forklaringer og begrunnelser for dem. Artikkelen presenteres ulike generaliseringsstrategier som elever kan bruke i mønsteroppgaver. Den første strategien er å telle, der elevene teller elementene i mønsteret for å finne et mønster eller en regel. Den andre strategien er rekursiv, der elevene bygger på den forrige figuren eller figurene i mønsteret for å bestemme påfølgende figur. Videre er den tredje strategien helobjekt, der elevene bruker en del som enhet for å konstruere en større enhet ved å multiplisere. Den fjerde strategien er gjette og sjekke, der elevene gjetter en regel uten hensyn til hvorfor denne regelen kan fungere, og deretter eksperimenterer med ulike operasjoner og tall gitt i problemet. Og den femte strategien er kontekstuell, der elevene konstruerer en regel basert på informasjonen som er gitt i situasjonen og knytter regelen til en telle-teknikk. Disse strategiene gir elevene ulike tilnærminger til å generalisere mønstre og utvikle algebraisk resonnering. Det er viktig å merke seg at strategiene for generalisering kan variere avhengig av typen oppgaver som elevene blir presentert for. Telling og den rekursive strategien betraktes som ikke-eksplisitte tilnærminger. Strategiene kan ikke brukes direkte til å beregne den avhengige variabelen, da de avhenger av informasjon fra tidligere trinn i sekvensen. Dette gjør dem utfordrende å anvende når man beveger seg lengre inn i sekvensen. Et eksempel på den rekursive strategien er å legge til tre fyrstikker for å danne en ny rute i en rekke med ruter. Selv om den rekursive strategien kanskje ikke egner seg for beregninger, gir den likevel viktig innsikt i mønsteret og hvordan størrelsene utvikler seg over tid.

På den annen side anses Helobjekt, gjette og sjekke, og den kontekstuelle strategien som eksplisitte tilnærminger. Dette betyr at de kan brukes til å beregne den avhengige variabelen

for en vilkårlig verdi av den uavhengige variabelen (Lannin, 2005). Gjette og sjekke-strategien innebærer vanligvis å bekrefte at en regel gjelder for noen få tilfeller uten å gi en begrunnelse for hvorfor den gjelder for alle tilfeller. Denne tilnærmingen fører ikke til algebra, da den ikke fanger opp den generelle strukturen i oppgaven (Radford, 2010). Helobjekt-strategien kan knyttes til oppgavekonteksten, men Stacey (1989), som også har studert bruken av denne strategien blant elever, påpeker at den ofte brukes feil. For eksempel innebærer Helobjekt-strategien å doble antallet synlige sider når antallet terninger som plasseres oppå hverandre dobles. Imidlertid krever det riktig forståelse av hvordan man justerer svaret for å passe til situasjonen, da en synlig side blir skjult når nye terninger legges på toppen av de eksisterende. Derfor må man trekke fra en side etter å ha doblet antallet synlige sider for å få riktig svar. Blant disse strategiene er det den kontekstuelle strategien som alltid gir en sammenheng med oppgavesituasjonen. Den kontekstuelle strategien går ut på at eleven utvikler en regel ved å analysere informasjonen som er tilgjengelig i situasjonen der eleven oppdager en sammenheng mellom mønsteret i figurene og det algebraiske uttrykket.

En overbevisende argumentasjon for generalisering bør kunne beskrive eller forklare en universell sammenheng som gjelder for alle tilfeller i den gitte oppgavesituasjonen. Når generalisering brukes som en innføring til algebra, blir det viktig at elevene klarer å uttrykke den kontekstuelle sammenhengen ved hjelp av symbolske representasjoner. Ifølge Radford (2010) er det bare når det dannes en generell regel som kan representere ethvert tilfelle i mønsteret, at vi kan snakke om generalisering. Elevenes oppgave er derfor å gå fra å oppdage det generelle mønsteret, til å uttrykke dette mønsteret symbolsk på en måte som er gyldig for alle tilfeller i mønsteret. Denne symbolske representasjonen blir deretter et objekt som kan manipuleres videre. Det er viktig å forstå at symbolske representasjoner ikke bare handler om å uttrykke det samme på en annen måte; det handler om å få tilgang til en dypere forståelse av mønsteret og sammenhengen mellom elementene, noe som er relevant for denne masteroppgaven studien.

2.4.3 Barrierer og hindringer

Munthe (2022) beskriver de utfordringene elevene står overfor når de jobber med programmeringsoppgaver i matematikk, og anvender adidaktiske situasjoner fra TDS for å belyse dette. En adidaktisk situasjon oppstår når elevene interagerer med miljøet, som består av deres jevnaldrende, verktøyene og utformingen av problemene, og tar initiativet og har ansvar for resultatet. Undersøkelser har grundig vurdert utfordringer som oppstår når man

lærer programmering (Medeiros et al., 2018; Piteira & Costa, 2013). Disse fokuserer på å forstå programmeringssyntaks og hvordan man håndterer variabler, løkker og betingelser, spesielt i en skolekontekst, som ofte utgjør en betydelig utfordring for elever (Bosse & Gerosa, 2017).

Ko et al. (2004) har identifisert seks barrierer i programmeringsprosessen, hvorav tre er spesielt relevante for skoleprogrammering. Den første barrieren er knyttet til valg av programmeringsverktøy og kunnskapen om hvilke verktøy og kommandoer som skal brukes for å bygge et ønsket resultat. Den andre barrieren omhandler forståelse av hvordan man skal håndtere feil og uventet oppførsel i programmet, en velkjent utfordring innenfor programmeringsundervisning (Lahtinen et al., 2005). Den tredje barrieren innebærer situasjoner der programmet ikke gir forventede resultater eller ikke oppfører seg i tråd med hypotesene. Elevers evne til å evaluere og diskutere matematiske problemer spiller en sentral rolle i undervisningen. Ko et al. (2004) har også delt disse barrierene inn i underkategorier som "overkommelige barrierer," som elevene kan overvinne, og "uoverkommelige barrierer," som oppstår når kompleksiteten eller antallet barrierer blir for overveldende for elevene, og de står overfor en uoverkommelig hindring. Overkommelige barrierer likner på epistemologiske hindringer, mens uoverkommelige barrierer likner på ontologiske hindringer (Brousseau, 1997).

Å tillate elever å tilpasse sine strategier for å oppnå ønsket kunnskap er en utfordrende oppgave. Ifølge Brousseau (1997) er det nødvendig å lette denne tilpasningen gjennom oppgavene for at elevene skal lykkes. Når elever møter utfordringer og overvinner dem mens de arbeider med oppgaver, kan tilpasning oppstå. Brousseau (1997) beskriver en epistemologisk hindring som en type kunnskap som tidligere har vært nyttig og vellykket i spesifikke sammenhenger, inkludert skolesammenhenger, men som av og til blir feil eller utilstrekkelig. Hindringene kan også ha forskjellige karakterer: ontogene hindringer knytter seg til elevers begrensninger og manglende tidligere læring, mens didaktiske hindringer er relatert til presentasjonen av emnet og resultatet av begrenset eller feilaktig instruksjon (Harel og Sowder, 2005, s. 34). Ut fra dette er ontogene og didaktiske hindringer som begrenser læring noe som bør unngås, mens epistemologiske hindringer kan fremme læring.

Balacheff (1990, s.264) forklarer en epistemologisk hindring som følger:

> *All kunnskap må bygge på elevenes tidligere kunnskap. Men denne eksisterende kunnskapen kan noen ganger hemme dannelsen av nye begreper, selv om den er*

> *nødvendig som grunnlag. Imidlertid er det ofte slik at overvinningen av denne hindringen er en del av prosessen med å konstruere mening for det nye innholdet. (s.264, oversatt av meg)*

Dette indikerer viktigheten av at oppgavedesign tar hensyn til tidligere kunnskap for å lette dannelsen av nye begreper og fremme konstruksjonen av ny forståelse gjennom design av adidaktiske situasjoner.

I løpet av arbeidet med oppgaver vil elever oppleve perioder hvor de vet hvordan de skal utføre nødvendige handlinger, samt perioder hvor de møter hindringer. Adidaktiske situasjoner gir elevene muligheten til å revurdere sine strategier, utvikle nye tilnærminger, delta i diskusjoner med jevnaldrende, gjøre antagelser og eksperimentere. Alt dette er knyttet til den tiltenkte læringsprosessen (Leung og Baccaglini-Frank, 2016). Som et resultat fremmer adidaktiske situasjoner matematisk tenking og resonnement, samt utvikling av konseptuell kunnskap (Hiebert, 2013; Skemp, 1976).

Selv om elevene har en viss forståelse av både programmeringssyntaks og semantikk, kan de likevel ha vanskeligheter med å kombinere disse elementene for å skape gyldige og effektive programmer (Winslow, 1996). Til tross for flere forsøk på å utvikle strategier som støtter nybegynnere i programmering, for eksempel samarbeid i team, veiledning fra medelever, workshops og online-forum, forblir det betydelige utfordringer når elever introduseres for konsepter som algoritmisk tenkning, logikk og problemløsning (Stephens & Kadijevich, 2020). Kombinasjonen av kompleksiteten i algoritmisk tenkning og den utfordrende oppgaven med å mestre syntaksen kan ofte føre til frustrasjon, avvisning av emnet og en generelt negativ opplevelse for elevene (Buitrago Flórez et al., 2017). Dette betyr ikke nødvendigvis at læring av programmering er en håpløs oppgave, men det understreker behovet for effektive undervisningsmetoder som kan lette disse utfordringene. Frustrasjon og utfordringer kan ses på som en naturlig del av læringsprosessen, spesielt i et fagområde så komplekst som programmering.

2.5 Sammendrag av teorien

I dette teorikapittelet presenterer jeg potensialet for dybdelæring gjennom elevers arbeid med programmering, fra teori og tidligere forskning fordelt over fire hoveddeler. Den innledende delen setter fokus på dybdelæringsteorier, algebraens rolle i skolen, programmering og til slutt det teoretiske rammeverket som legger grunnlaget for denne masteroppgaven.

Viktige bidrag fra forskningen til Lannin og Munthe, samt Schoenfelds teori om robust læring, blir brukt i denne masteroppgaven. Lannins arbeid med generaliseringsstrategier gir innsikt i hvordan elever kan utvikle en dypere forståelse av matematiske mønstre gjennom programmering. Munthes forskning fremhever hvordan programmering kan bidra til matematiske utforskende dialoger og fordype forståelsen av velkjente konsepter. Han trekker også frem utfordringene knyttet til å tilegne seg nye matematiske konsepter gjennom programmering. Schoenfelds rammeverk om robust læring tar frem viktigheten av innhold, kognitiv etterspørsel, likeverdig tilgang til læring, samt elevers agens, eierskap, og identitet i læringsprosessen.

Ved å kombinere disse tilnærmingene i min forskning, vil jeg se på hvordan elever anvender generaliseringsstrategier i programmeringsoppgaver og hvordan deres agens, i form av eierskap, engasjement og identitetsutvikling, påvirker deres læring og anvendelse av algebraiske konsepter. Denne tilnærmingen vil hjelpe meg å identifisere potensialet for dybdelæring i programmeringsbasert matematikkundervisning og utvikling av strategier for å identifisere og støtte denne typen læring i skolen.

3.0 Metode

I denne studien er fokuset å se på hvilket potensial for dybdeæring som kan identifiseres i elevers arbeid med programmering i algebra. For å svare på problemstillingen min har jeg valgt en kvalitativ tilnærming. Først vil jeg argumentere for hvorfor en casestudie passer til mine forskningsspørsmål. Deretter vil jeg beskrive undervisningsoppleggene og oppgavene som ble brukt i studien der disse forankres i teori. Videre redegjør jeg for hvordan utvalget av elever ble gjort. Avslutningsvis beskriver jeg bruken av observasjon som metode for datainnsamling, og hvordan jeg har analysert det innsamlede datamaterialet.

3.1 Et kvalitativt forskningsdesign

Jeg har valgt en instrumentell casestudie som forskningsdesign. I min masteroppgave kan en instrumentell casestudie brukes til å undersøke hvordan spesifikke eksempler på bruk av Python-programmering i algebraundervisning for ungdomsskoleelever kan belyse pedagogiske spørsmål om dybdelæring og integrering av programmering i matematikkundervisningen. Ved å studere denne spesifikke situasjonen, kan jeg trekke ut innsikter som er relevante for å forstå hvordan programmering potensielt kan transformere matematikkundervisning mer generelt, og identifisere faktorer som bidrar til eller hindrer denne prosessen.

Dette valget av forskningsdesign understrekes av Postholm (2018) for dets fleksibilitet og åpenhet, noe som gjør det spesielt velegnet til å utforske dybden av komplekse fenomener gjennom å konsentrere seg om et begrenset antall caser, i dette tilfellet elevers erfaringer med Python i algebra (Postholm et al., 2018, s. 137; Thagaard, 2018, s. 55; Creswell, 2013, s. 247).

Den instrumentelle casestudien, som Creswell (2013, s. 98) definerer, fokuserer mindre på de individuelle deltakerne og mer på fenomenet som undersøkes. Det innebærer at selv om oppmerksomheten for eksempel er rettet mot en utvalgt gruppe på sju elever, er det deres erfaringer og interaksjoner med Python i algebra som står i sentrum for analysen. Målet er å utvikle en dypere forståelse av hvordan arbeid med programmeringsspråket Python kan fremme eller hindre prosesser av dybdelæring innenfor algebra, gjennom å utforske elevers samtaler og valget deres av generaliseringsstrategier.

Ved å anvende en instrumentell casestudie, er hensikten å belyse det empiriske grunnlaget for å forstå potensialet for dybdelæring i elevers arbeid med Python i algebra. Dette designet

tillater en grundig undersøkelse av det spesifikke fenomenet, hvor elevers strategivalg og agens identifiseres som en indikator på dybdelæring.

Denne tilnærmingen er ideell for å svare på den sentrale problemstillingen, ettersom den gir mulighet for å observere og analysere hvordan elever anvender og forstår algebraiske konsepter gjennom programmering. Ved å fokusere på fenomenet fremfor deltakerne, understreker studien viktigheten av å utforske de underliggende prosessene som bidrar til dybdelæring. Denne tilnærmingen tillater også identifisering av hindringer elever møter og hvordan disse kan overvinnes, noe som er avgjørende for å forstå hvordan programmering med Python kan optimere læringen av algebra.

Denne studien, basert på teoretiske rammeverk fra Postholm et al. (2018), Thagaard (2013), og Creswell (2013), tilbyr innsikt i hvilke undervisningsmetoder som mest effektivt støtter dybdelæring gjennom integrasjon av programmering i matematikkundervisningen. Denne studien bidrar dermed til forskning som søker å forstå og forbedre matematikkundervisning gjennom teknologisk integrasjon, med spesifikt fokus på programmeringsspråket Python.

3.2 Deltakere

Denne våren har jeg jobbet som vikar på en ungdomsskole, noe som gjorde det praktisk å samle inn data der. Valget av 10. klasse som fokus for studien var strategisk. På dette trinnet står elevene overfor kompetansemål som oppfordrer til utforsking av matematiske egenskaper og sammenhenger ved hjelp av programmering, et mål som direkte samsvarer med studiens formål.

Jeg ønsket en klasse som jeg har hatt lite kontakt med som vikar, og der eventuell tidligere kontakt har vært i andre fag enn matematikk som for eksempel kroppsøving eller naturfag. Jeg hadde derfor en samtale med en lærer, som jeg visste underviste en klasse i matematikk, for å diskutere hvordan datainnsamlingen kunne organiseres. Jeg gjorde læreren oppmerksom på at jeg hadde søkt om godkjenning fra SIKT og trengte underskrifter fra deltakerne, og understreket at deltakelse selvsagt var frivillig. Videre ba jeg læreren om å informere elevene selv, slik at jeg minst mulig skulle påvirke deres beslutning om å delta i studien. Jeg uttrykte også et ønske om å organisere elever i grupper på tre, basert på forskningen til Berry og Sahlberg (2006) om at dette formatet fremmer høykvalitetsinteraksjon.

En slik interaksjon er kjennetegnet ved at deltakerne aktivt engasjerer seg i diskusjoner, utforsker ulike løsningsstrategier, stiller spørsmål, utfordrer hverandre og reflekterer sammen over problemstillinger. Slik interaksjon kan skape et læringsmiljø som fremmer dyp forståelse, kritisk tenkning og samarbeid, og støtter utviklingen av matematiske ferdigheter og problemløsningsstrategier. Dette er viktig i denne studien da den fokuserer på samtalene, og hvordan elevene interagerer med hverandre. Videre viser også en studie av Lou et al. (2001) at elever som jobbet i små grupper og elever som arbeidet individuelt med teknologi, viste ulike atferdsmønstre, hvor gruppearbeid fremmet mer positiv samhandling og bruk av effektive læringsstrategier. For å sikre at gruppene hadde best mulig dynamikk, fikk jeg hjelp av læreren som kjente godt elevene og visste hvem som kunne arbeide sammen i gruppe.

Siden programmering er en relativ ny del av matematikkundervisningen kan kompetanse og trygghet på å programmere, variere fra klasse til klasse. For å tilrettelegge mest mulig for datainnsamlingen valgte jeg bevisst informanter som hadde noe erfaring med programmering, for å minimere potensielle "programmeringsbarrierer", som beskrevet av Munthe (2022). Disse barrierene inkluderer utfordringer elever møter når de arbeider med programmering, og jeg var spesielt opptatt av å unngå det Munthe kaller "selection barriers". Det omhandler forståelsen av brukergrensesnittet i programmeringsverktøy og kjennskapen til kommandoer som brukes i programmeringsarbeidet. Ut fra disse kriteriene ble mitt utvalg karakterisert som et ikke-sannsynlighetsutvalg eller kriteriebasert utvalg, der deltakere ble valgt basert på forhåndsbestemte kriterier, som beskrevet av Gleiss og Sæther (2021, s. 39).

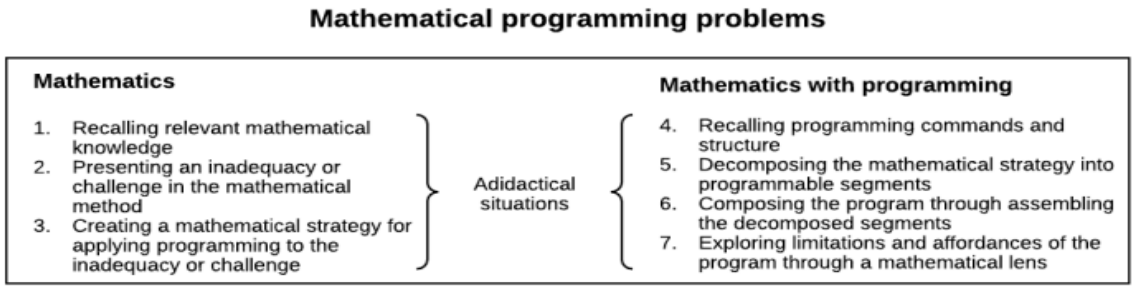
Forsøket var å legge til rette så mye som mulig for å få grupper på tre. Med et begrenset antall deltakere som ville være med i studien endte det opp med totalt tre grupper. To grupper med to deltakere og en gruppe med tre deltakere. Den ene gruppen med to deltakere besto av to jenter og den andre gruppen med to deltakere besto av to gutter. Den tredje gruppen med tre deltakere besto av tre jenter.

3.3 Utforming av oppgaver

I min masteroppgave har jeg valgt å bruke generalisering av figurtall som et tema i undervisningsopplegget som elevene skal få løse. I Lannin (2005) sin forskning er hovedfokuset på hvordan elever kan overføre og anvende deres eksisterende kunnskap om aritmetikk til å forstå og anvende algebraiske prinsipper. Gjennom å engasjere seg i mønsterbaserte aktiviteter, blir elevene utfordret til å tenke abstrakt og generalisere

matematiske prinsipper. Dette er kritiske ferdigheter i algebraisk tenking. Denne prosessen krever også at elevene rettferdiggjør sine resonnementer og løsninger, en praksis som fremmer dypere forståelse og kritisk tenking. For å utforme undervisningsopplegget rundt oppgavene har jeg brukt Munthe (2022) sin artikkel ¨Designing mathematical programming problems¨ som inspirasjon til strukturen på undervisningsopplegget. Design ideene i artikkelen bli kalt for matematiske programmeringsproblemer heretter forkortet til (MPP), hvor målet er å fremme dybdelæring ved bruk av programmering i læringsprosessen. MPP-ene består av syv faser som legger tilrette for elevenes samhandling med miljøet, fra å trekke frem tidligere kunnskap til utviklingen av nye strategier.

Dette er en representasjon av Munthe sin design ide for MPP.



Figur 3 (Munthe, 2022, s. 40)

For å forklare dette designet av et undervisningsopplegg så starter man med å:

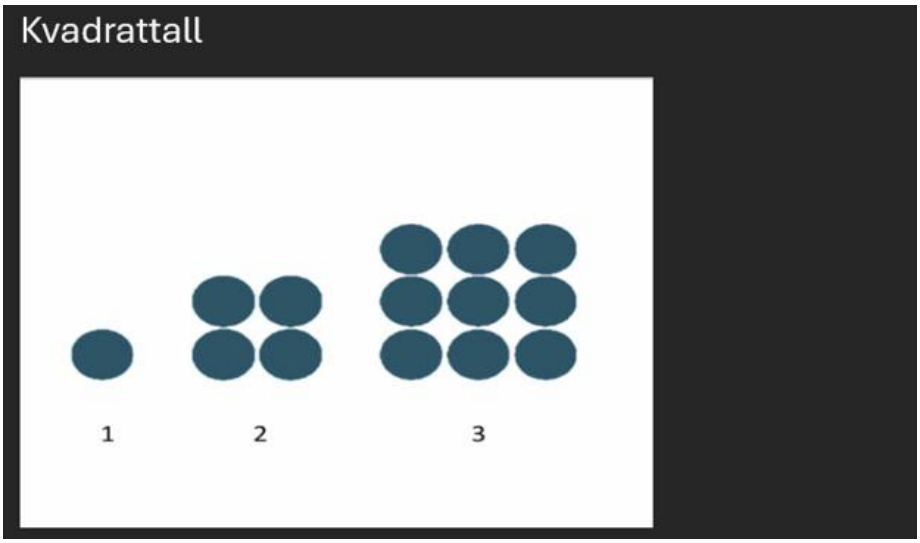
1. Gjenkalle relevant matematisk kunnskap
2. Presentasjon av en utilstrekkelighet eller utfordring i den matematiske metoden.
3. Utvikling av en matematisk strategi for å anvende programmering på utilstrekkeligheten eller utfordringen.
4. Gjenkalling av programmeringskommandoer og strukturer.
5. Dekomponering av den matematiske strategien i programmerte segmenter.
6. Sammensetting av programmet ved å sette sammen dekomponerte segmenter.
7. Utforsking av begrensninger og muligheter i programmet med et matematisk perspektiv (Munthe, 2022, s.40, min oversetting).

Når elever jobber med MPP-er vil elevene oppleve perioder der de vet hva de skal gjøre, og perioder hvor de møter hindringer som kan legge til rette for matematiske læringsmuligheter. Hindringer, når de blir epistomologiske som beskrevet av Brousseau (1997) er en type kunnskap som tidligere har vært nyttig og vellykket i noen sammenhenger, men som andre ganger blir feil eller utilstrekkelig. Disse epistemologiske hindringene kan bidra til å fremme

læring, fordi det gir muligheter for elevene til å tenke nytt om strategien sin, utvikle nye tilnærminger, diskutere med medelever, komme med hypoteser og eksperimentere. Ifølge Munthe (2022) har MPP potensiale til å fremme matematisk tenking, resonnering og utvikle konseptuell kunnskap, og slik kan disse føre til dybdelæring av matematisk innhold som er det jeg skal undersøke.

3.3.1 Figurtallsoppgaven

Hovedmålet med designet av undersøkelsen var å få fram data som kunne besvare problemstillingen. Gjennomføringen av oppgaven gikk ut på at elevene først skulle «Gjenkalle relevant matematisk kunnskap», og ble innledet med at de fikk en visuell presentasjon av de tre første figurene i ett kvadrattallsmønster. Deretter skulle de finne den neste figuren og gjenkjenne hvor mange prikker det var i mønsteret, videre finne antall prikker og mønster for figurer som gjør det vanskelig å bruke enkle generaliseringsmetoder. I en typisk oppgave innen figurtallmønstre presenteres elever med en figur eller et mønster som har en spesifikk vekst. Oppgaven deres er å identifisere denne sammenhengen og formulere en regel som kan brukes til å beregne hvilket som helst trinn i mønsteret. Dette brukes ofte som en introduksjon til algebra, der elevene oppfordres til å uttrykke regelen som en matematisk formel med en variabel. Dette gir elevene muligheten til å forstå at matematiske uttrykk og variabler har en viktig funksjon og viser en matematisk sammenheng (Lannin,2005). Mine informanter fikk følgende oppgave:



Figur 4 Første oppgave for å gjenkalle matematisk kunnskap

1. Tegn kvadrattall nr.4. Hvor mange prikker er det i kvadrattall nummer 4?
2. Diskuter i gruppen hvordan figuren utvikler seg?
3. Hva med nummer 7 eller 12?
4. Lag en generell formel for n? Forklar hvordan figurene utvikler seg.

Elevene skulle svare på de fire spørsmålene, og deretter diskutere hvor figuren utvikler seg visuelt. Målet var å legge til rette elevenes engasjement og deltakelse i diskusjonen, sånn at deres interesse for å identifisere mønstre skulle vekkes.

I oppgaven om kvadrattall skal elevene utvikle en generell formel for figurtallene se Figur 4, hvor de nok en gang må anvende strategier for generalisering. Fullstendig oversikt over de ulike delen blir beskrevet i kapittel 4.1. Lannin (2005) understreker at det er å foretrekke at elevene bruker en kontekstuell strategi i generaliseringsprosessen ved å støtte seg på de matematiske sammenhengene i mønsteret.

Oppgavedesignet er med på å få frem viktige aspekter ved læring, og kan være sentrale når jeg skal identifisere hvilket potensiale elevene har for dybdelæring gjennom programmering. Først og fremst legger designet jeg har utarbeidet opp til aktivt engasjement og gjenkalling av kunnskap, ved å introdusere elevene for visuelle presentasjoner av matematiske mønstre.

Oppgaven stimulerer elevene til å utforske og observere. De må identifisere mønstre og sammenhenger, noe som krever en dypere forståelse fremfor kun overfladisk memorering. Denne prosessen, hvor elevene observerer, utforsker og identifiserer mønstre, fremmer en utforskende tilnærming til læring. Her må elevene selv arbeide seg frem til løsningene, noe som kan observeres og bidra til innsikt i potensialet for dybdelæring.

Diskusjon og refleksjon utgjør også et viktig element i oppgavedesignet mitt. Når elevene blir engasjerte i samtaler om hvordan figuren utvikler seg visuelt, oppmuntres de til å tenke kritisk og dele sine tanker og ideer. Dette kan tilrettelegge for en dypere forståelse og et personlig eierskap til læringen. Ved å engasjere seg i diskusjoner, kan elevene utvide sin egen kunnskap gjennom å ta del i hverandres ideer. Dette bidrar til at de utvikler evnen til å argumentere for sine synspunkter og forstå ulike perspektiver. En slik samarbeidsprosess er essensiell for å utvikle kritisk tenkning og problemløsningsferdigheter, som er sentrale elementer i dybdelæring (Schoenfeld, 2018).

Til slutt vektlegges det i oppgavedesignet hvor viktig det er å kunne generalisere og anvende matematiske konsepter. Ved å utfordre elevene til å formulere en generell formel, blir de

introdusert for temaet algebra på en måte som krever abstrakt tenkning. Dette steget er avgjørende for å gå fra konkret forståelse til abstrakte konsepter, en prosess som er sentral i matematisk forståelse og dybdelæring. Å kunne generalisere og anvende lærte konsepter i nye sammenhenger viser en dypere forståelse av materialet, noe som er målet med dybdelæring (Lannin, 2005).

Oppgavedesignet er utformet for at det skal være mulig å identifisere noen sammenhenger mellom programmering og dybdelæring i algebra gjennom at elevene i læringssituasjonen utfordres til å diskutere, generalisere, tenke kritisk og ta agens.

3.3.2 Programmeringsoppgaven og valget av verktøy

På skolen der dataene ble samlet inn, er det Python som benyttes som programmeringsverktøy i matematikkundervisningen. Tekstbasert programmering, som innebærer å formulere instruksjoner i tekstformat som kode for en datamaskin, tillater utvikling av mer komplekse og avanserte programmer sammenlignet med blokkbasert programmering (Kaufmann et al., 2022, s. 593). Haraldsrud et al. (2022, s. 13-14) fremhever tekstbasert programmering som den mest effektive metoden for å lære programmering på ungdomsskole- og videregående nivå, med argumenter om dens relevans og anvendelighet i fremtidig yrkesliv. Python, anerkjent for sin enkelhet og effektivitet spesielt innen matematiske applikasjoner, er fremhevet som et ideelt verktøy for denne formen for programmering. Dets popularitet i næringslivet og tilgjengeligheten av et omfattende bibliotek med forhåndsskrevne koder understreker dets anvendelighet (Haraldsrud et al., 2022, s. 14). Under datainnsamlingen ble MU editor benyttet som programmeringsmiljø.

Programmeringsoppgavene bygger på de samme fundamentale prinsippene som anvendes i oppgaver relatert til figurtall, men med en viktig forskjell: i stedet for å støtte seg på visuelle figurer, måtte elevene her anvende generelle formler for kvadrattall, trekanttall, og potensielt femkanttall. Fra oppgave 1 til 3 følges denne tilnærmingen (vedlegg 2), mens i den fjerde oppgaven utfordres elevene til å modifisere koden for å visualisere de forespurte figurene, et skritt som integrerer programmering med matematisk forståelse.

3.4 Metode for datainnsamlingen

I forbindelse med min undersøkelse av potensiale for dybdelæring i algebra i elevers arbeid med programmering, valgte jeg å bruke lydopptak i datainnsamling. Samtidig ønsket jeg å fange opp hva elevene gjorde på datamaskinene sine mens de arbeidet med programmeringsoppgaver. Det var essensielt for meg at lydopptakene og skjermopptakene skulle være synkroniserte, slik at jeg kunne observere hva som ble sagt samtidig som elevene skrev kode. Jeg løste dette ved å benytte Windows sin egen skjermopptaks app Xbox gamebar som ga meg muligheten til å generere én fil med synkroniserte skjerm- og lydopptak.

Som Postholm og Jacobsen (2018) påpeker, kan lyd- og videoopptak være svært nyttige verktøy for forskere ved observasjon. Det er imidlertid viktig for forskeren å være kjent med det nødvendige utstyret (Postholm & Jacobsen, 2018). Derfor testet jeg nøye opptaksfunksjonen til Windows i samarbeid med en faglærer for å sikre at alt fungerte som det skulle i forkant.

For å gjennomføre datainnsamlinga, ble oppgavene introdusert som en undervisningsøkt. Jeg startet opp med å forklare hvordan økten var strukturert og hva slags type oppgaver de kom til å møte, og at det viktigste med denne oppgaven var gruppesamarbeid og at de måtte prate og ha en tydelig dialog. Ved oppstart gav jeg også beskjed om at de ikke kunne gå ut av programmeringsprogrammet da skjermopptakeren stopper ved å gå ut av programmet. Så ved start gikk jeg rundt til de forskjellige gruppene for å sette opp mikrofon, skjermopptaker og programmeringsprogram.

3.4.1 Min rolle som observatør og deltaker

I forberedelsen og gjennomføringen av datainnsamlingen var det essensielt å vurdere forskerrollen nøye, med spesielt fokus på valg av felt og graden av åpenhet, som Grønmo (2016, s. 156) fremhever som viktige overveielser. Under datainnsamlingen var min tilstedeværelse i klasserommet både i rollen som deltaker og observatør, en strategi som tillot direkte interaksjon med elevgruppene og observasjon av deres aktiviteter (Grønmo, 2016, s. 155). Valget av klasserommet ble gjort med tanke på å sikre nok plass for at gruppene kunne arbeide separat samtidig som det var mulig for meg å observere alle uten hinder. Åpenhet rundt observasjonsprosessen og studiens hensikter ble prioritert, i tråd med Grønmo (2016, s. 158) sine anbefalinger om å være transparent om studiens natur overfor elevene.

Gleiss og Sæther (2021, s. 111) understreker utfordringene med forskerens påvirkning i datainnsamlingssituasjoner, og peker på at det er viktig å reflektere over hvordan denne påvirkningen håndteres. Ifølge Grønmo (2016, s. 158) er etableringen av aksept, tillit, og feltrelasjoner mellom forsker og deltakere kritisk for en vellykket datainnsamling. En god relasjon til klassen ble etablert, delvis takket være en tidlig introduksjon og muligheten til å bli kjent med elevene før datainnsamlingen startet. Datainnsamlingen ble integrert i en undervisningsøkt om et emne elevene allerede var engasjert i, og informasjon om at innholdet ville være relevant for senere vurderinger bidro til å skifte fokuset fra forskningsaktiviteten til skolerelaterte læringsmål. Dette kan ha bidratt til at elevene engasjerte seg mer og at kommunikasjonen mellom elevene ble hyppigere slik at jeg fikk nok data. Som deltaker i situasjonen hadde jeg mulighet til å få tilgang til mer detaljerte data. Gjennom direkte deltakelse kunne jeg oppleve og forstå konteksten på en mer grundig måte, og dermed samle inn mer rike og nyanserte data. Min deltakelse i situasjonen virket som den bygget tillit, slik at det førte til mer åpne og ærlige diskusjoner, samt et bedre samarbeid.

3.5 Validitet og reliabilitet

I forskning er validitet essensielt for å sikre at funn og konklusjoner nøyaktig reflekterer potensialet for dybdelæring i arbeidet med programmeringsspråket Python innen algebraundervisning, som understreket av Creswell (2013). Dette prinsippet understreker viktigheten av å anvende grundige metoder for datainnsamling og -analyse for å oppnå pålitelige resultater som kan bidra til en dypere forståelse av hvordan slik undervisning kan påvirke elevers læring og engasjement.

For å styrke forskningens validitet har jeg inkludert bruk av detaljerte og rike beskrivelser av hvordan undervisningen utføres og hvordan elevene interagerer. En triangulering gjennom innsamling av data fra flere kilder (skjermopptak og lydopptak), og en refleksjon over forskerens egne meninger (Creswell, 2013).

Gitt studiens mål om å utforske dybdelæring gjennom bruk av Python-programmering i algebra, har det vært avgjørende å analysere hvordan denne metoden påvirker elevers forståelse og engasjement i matematikk. Dette har involvert en grundig undersøkelse av hvordan elevenes interaksjoner med programmeringsoppgaver fremmer en dypere forståelse av algebraiske konsepter. Med tanke på dybdelæringens potensial i elevers arbeid med Python

i algebra, har det vært nødvendig å reflektere over min egen forforståelse og min relasjon til elevene, da forskerens rolle og perspektiv er kritiske i kvalitativ forskning (Creswell, 2013).

Selv om det kvalitative forskningsdesignet ikke sikter mot å generalisere funnene, er målet å belyse potensielle muligheter og utfordringer basert på de observerte casene. Det er anerkjent at elevgrupper og undervisningssituasjoner varierer, og at resultatene er påvirket av mange faktorer. Videre er påliteligheten av transkripsjonene fra datainnsamlingen viktig for å sikre at de representerer virkeligheten så nøyaktig som mulig, gitt at transkripsjoner er fortolkninger og bare gir et begrenset innblikk i konteksten (Cohen et al., 2007, s. 367).

Datainnsamlingen, som inkluderte lyd- og skjermopptak av elever i arbeid, gav et rikt og detaljert grunnlag for transkripsjoner og analyser. Denne tilnærmingen muliggjorde en helhetlig beskrivelse av situasjonene, inkludert observasjoner av elevers interaksjoner, verbale uttrykk, og digitale handlinger. Dette inkluderer også, når det var tydelig, stemning mellom elever eller holdninger (for eksempel irritasjon, sinne og latter). Hendelser dokumentert gjennom skjermbilder er også integrert i transkripsjonene for å gi en forståelse av situasjonene, noe som ytterligere styrker studiens validitet og troverdighet.

I kvalitativ forskning refererer indre reliabilitet til konsistensen i anvendelsen av metoder og tilnærminger innad i en studie, sikrende at forskningsprosessen er gjennomført på en pålitelig måte som kan gjøres igjen. Ytre reliabilitet fokuserer derimot på forskningsresultatenes generaliserbarhet til andre sammenhenger. Det undersøker om funnene fra en studie kan overføres eller gjenskapes i lignende studier utført av andre forskere. Gitt kvalitativ forsknings metodologiske mangfold og dens tilpasningsdyktige natur, presenterer oppnåelsen av både indre og ytre reliabilitet seg som en utfordring. Dette skyldes at nøyaktig replikasjon av studiekontekster og deltakerdynamikk sjelden er mulig, noe som krever en nøye vurdering av metodenes pålitelighet og overførbarhet (Jacobsen, 2021, s. 222-223)..

I kvalitativ forskning refererer validitet til hvor nøyaktige og relevante forskningsresultatene er i forhold til det studerte fenomenet. Det innebærer å vurdere om funnene gir et korrekt bilde av fenomenet. Validitet kan påvirkes av flere faktorer, inkludert tolkningene til forskeren, hvordan deltakerne blir valgt og samhandler, samt innsamling og analyse av data. For å oppnå høy validitet, er det avgjørende å sikre at dataene er representative for fenomenet som studeres, og at forskerens tolkninger blir tydelig diskutert og reflektert i forskningsrapporten (Jacobsen, 2021, s. 229).

3.6 Etiske hensyn

Det ble lagt stor vekt på at alle deltakerne i studie skulle være fullstendig informert om formålet og deres rettigheter. I løpet av datainnsamlingsperioden ble både skjerm- og lydopptak benyttet. Når man arbeider med slike opptak, er det nødvendig å vurdere konsekvensene publiseringen av disse funnene kan få for deltakerne. Etter utførelsen av oppgavesekvensen ble alle opptak overført til datamaskinen og deretter slettet fra elevenes datamaskiner. Hensikten å benytte transkripsjonene sammen med skjermbilder fra programmering som forskningsfunn i vår rapport; hverken lydklipp eller skjermopptak vil være tilgjengelige for andre enn deltakerne i prosjektet.

Før prosjektet startet, ble samtykkeskjemaer gitt til de deltakerne som ønsket delta, som inneholdt grundig informasjon om prosjektets hensikt, prosjektledere, frivillighet for deltakelse, databehandling og deltakernes rettigheter. Samtlige elever som deltok i studien, fremla samtykkeerklæringer.

Det er en betydelig grad av ansvar knyttet til å bruke elever, spesielt barn, som informanter i forskning. Å innhente samtykke fra elevene har vært en prioritert og selvsagt nødvendighet i denne studien (Tangen, 2010). Før innsamlingen av data begynte, sendtes det en melding til SIKT (SIKT- Kunnskapssektorens tjenesteleverandør), inkludert informasjonsskriv til elever, det er ikke behov for foresattes underskrift i dette prosjektet fordi elevene er over 15år (Vedlegg 1). Denne søknaden ble registrert hos SIKT og overholder reglene for datalagring og forskningsetikk som er fastsatt av HIØ, før oppstart av datainnsamlingen. I disse skrivene ble det tydelig fremhevet at all innsamlet data ville bli behandlet med konfidensialitet og anonymitet som prioritet. Eventuelle skjerm- og lydopptak som ikke enkelt kunne anonymiseres, ble lagret på en ekstern harddisk og behandlet uten tilgang til internett for å opprettholde personvernet i henhold til HiØs etiske retningslinjer.

I forbindelse med transkripsjonsprosessen vurderte jeg å bruke kunstig intelligens (AI), men etter grundig vurdering lot jeg dette være da den eksisterende kontrakten med elevene som var at kun jeg og min veileder har tilgang til dataene. AI har i denne oppgaven blitt anvendt som et verktøy for oversettelse, både til oversetting av sammendraget og flere doktoravhandlinger som var relevante for min forskning. Det ble oversatt ved hjelp av AI for å forbedre min forståelse av innholdet. Denne strategien sikret integriteten av den sensitive dataen samtidig som den effektiviserte arbeidsflyten ved språklige utfordringer.

3.7 Analysemetode

I denne masteroppgaven har jeg i analyse anvendt Lannins generaliseringsstrategier sammen med Schoenfelds teori om agens (eierskap, engasjement og identitet). Gjennom dette rammeverket kunne jeg identifisere hvilke typer generaliseringsstrategier elevene brukte, samtidig som jeg kunne se på elevenes agens, eierskap og identitet i sine valg av strategier. Dette rammeverket fokuserer på elevers anvendelse av matematiske konsepter gjennom programmering og innvirkningen av deres agens på læringsprosessen. Analyseprosessen omfatter flere faser. I dette kapittelet er analysen delt inn i to underkapitler. Først undersøker jeg generaliseringsstrategier i 3.7.1, og deretter, i 3.7.2, analyserer jeg agens ved hjelp av Schoenfelds teori. I 3.7.3 undersøker jeg forbindelsen mellom agens og generaliseringsstrategier, og videre i 3.7.4 anvender jeg Brousseaus teori for å identifisere hindringer.

3.7.1 Identifisering av generaliseringsstrategier

I den innledende fasen identifiserte jeg hvilke av Lannins generaliseringsstrategier som ble benyttet av elever i deres programmeringsarbeid innen algebra. Dette inkluderer:

Telling: Hvor jeg vil se etter elevers telling av objekter eller mønstre direkte i koden for å finne løsninger.

Rekursiv: Her vil jeg se etter indikasjoner på at elever anvender tidligere resultater for å definere nye løsninger i programmeringen.

Helobjekt: Identifisering av tilfeller hvor elever anser hele problemet som en enhet og anvender dette perspektivet i koden for å generere løsninger.

Gjette og Sjekke: Evaluering av om elever tester ulike løsningsforslag i koden før de finner en fungerende løsning.

Kontekstuell: Observasjon av om elever anvender informasjon fra problemstillingen for å utvikle kode som reflekterer deres forståelse av problemet.

I denne prosessen var det utfordrende å skille mellom noen av strategiene, spesielt mellom rekursiv og kontekstuell strategi, da elever ofte brukte elementer fra begge i samme løsningsforsøk. Jeg tok beslutningen om å fokusere på de fremtredende strategiene i hver elevs arbeid for å klarlegge mønstre.

3.7.2 Analyse av elevers agens

I analyse av elevers agens, skal jeg se på hvordan Schoenfelds dimensjoner av agens vises i elevers programmeringsarbeid:

Eierskap: Her vil jeg se etter bevis på hvordan elever tar ansvar for sin egen læring gjennom valg av problemløsningstilnærminger og tilpasning eller modifikasjon av deres kode basert på forståelsen av matematiske konsepter.

Engasjement: Vurdering av elevers aktive deltakelse og utholdenhet i programmeringsoppgaver, inkludert deres villighet til å eksperimentere og utforske forskjellige løsningsmetoder.

Identitet: Observasjon av hvordan elever identifiserer seg selv som matematikere og programmerere gjennom deres interaksjon med oppgaven og refleksjoner om egen læring.

Rangnes og Herheim (2019) understreker viktigheten av at lærere oppmuntrer elever til å argumentere, hører på deres perspektiver, og tillater dem å påvirke undervisningen. Agens manifesteres gjennom hvordan elevenes handlinger og bidrag resulterer i respons fra både medelever og lærere. Det er også sentralt med delt agens, hvor flere aktører påvirker undervisningsprosessen. Ved å støtte elevers agens, bidrar lærere til en mer engasjerende og demokratisk læringsprosess i matematikkundervisningen. Denne tilnærmingen oppfordrer elever til å ta ansvar for egne ideer og synspunkter, og krever at de underbygger sine argumenter med bevis og utdyper sin tenkning. Dette fremmer en dypere forståelse og et sterkere engasjement i læringsprosessen (Schoenfeld, 2018, s.12).

Under denne prosessen oppdaget jeg at det kunne være utfordrende å skille mellom eierskap og engasjement, da begge dimensjoner ofte er synlige i samme atferd, spesielt når elever justerer sin tilnærming til en oppgave. Jeg tok en praktisk tilnærming ved å fokusere på de mest tydelige funnene av hver dimensjon og noterte hvor det forekom. Identifisering av agens i programmeringskonteksten viste seg også å være en kilde til innsikt, men krevde tolkning av både observerbare handlinger og elevenes egne refleksjoner.

3.7.3 Sammenkobling av strategier og agens

I denne fasen fokuseres det på forholdet mellom bruk av generaliseringsstrategier og manifestasjon av agens. Det undersøkes hvordan anvendelse av spesifikke generaliseringsstrategier reflekterer eller påvirker elevers eierskap, engasjement og identitet i læringsprosessen. Eksempelvis, hvordan en elev bruker en kontekstuell strategi for generalisering som kan indikere en dyptgående forståelse av matematiske konsepter, som igjen kan lede til en forsterket følelse av eierskap over problemet som løses.

Figur 5 Elevers agens til valg av generaliseringsstrategi

Gjennom denne analytiske prosessen blir det mulig å utforske hvordan integrering av programmering kan bidra til dybdelæring innen algebra, spesielt gjennom elevers anvendelse av matematiske konsepter og deres utvikling av agens.

3.7.4 Identifisering av hindringer og barrierer

Videre vil jeg på epistemologiske hindringer og ontologiske hindringer som Brousseau skriver om, og se på i hvilken grad elvene møter disse hindringene og hvordan elevene håndterer hindringene.

Epistemologiske hindringer: Brousseau (1997) skriver om "overkommelige hindringer", som kan ses i sammenheng med epistemologiske hindringer. Epistemologiske hindringer relaterer seg til kunnskapen som tidligere har vært nyttig og vellykket i bestemte kontekster, men som i noen tilfeller kan vise seg å være feilaktig eller utilstrekkelig. Disse hindringene er knyttet til elevers eksisterende forståelse og kan ofte oppstå fra tidligere læringsopplevelser. De kan både begrense og fremme læring, avhengig av hvordan de identifiseres og adresseres i undervisningsprosessen. Overvinning av epistemologiske hindringer er sett på som en del av

læringsprosessen, der elever må utfordre og revidere sin eksisterende forståelse for å utvikle en dypere innsikt.

Ontologiske Hindringer: Ontologiske hindringer beskriver Brousseau (1997) som "uoverkommelige barrierer" oppstår når kompleksiteten eller antallet utfordringer blir så overveldende at det virker uoverkommelig for elevene. Dette relaterer til elevenes oppfatning av læremateriale og kan omfatte alt fra konseptuell forståelse til følelsen av å være overveldet av oppgavens omfang. Disse hindringene er fundamentalt knyttet til elevens oppfatning av kunnskap og læring, og kan være spesielt fremtredende i utforskende og selvstyrt læring, som i programmeringsoppgaver. Ontologiske hindringer krever ofte en omstrukturering av elevens tilnærming til læring og en dyptgående støtte fra lærerens side for å overkomme.

Ved å se på hvordan elevene jobber videre, og hva slags strategier elevene bruker i møte med disse kan jeg avgjøre om hindringene er epistemologiske eller ontologiske, som hjelpe meg å kunne svare på, på hvilken måte hindringer kan gi grobunn for dybdelæring.

4.0 Resultater

I dette kapittelet vil jeg legge frem funnene som skal besvare problemstillingene: hvilke potensiale for dybdelæring kan identifiseres i elevers arbeid med programmering i algebra? Og på hvilken måte kan hindringer gi grobunn for dybdelæring?

For å identifisere potensialet har jeg brukt Schoenfeld beskrivelse av agens gjennom TRU-rammeverket, Lannin sine generaliseringsstrategier og Brousseau sin beskrivelse av barrierer og hindringer som potensiale for læring. Gjennom transkripsjon av tale og analyse av programmeringskoder i skjermopptak, har jeg sett på sammenkoblingen mellom elevenes eierskap, identitet og valg av strategier de bruker for å løse oppgavene.

De tre elevgruppene som deltok i undersøkelsen besto av Alfred og Brage i gruppe 1, Celine, Daniella og Erna i gruppe 2 og Frida og Gina i gruppe 3.

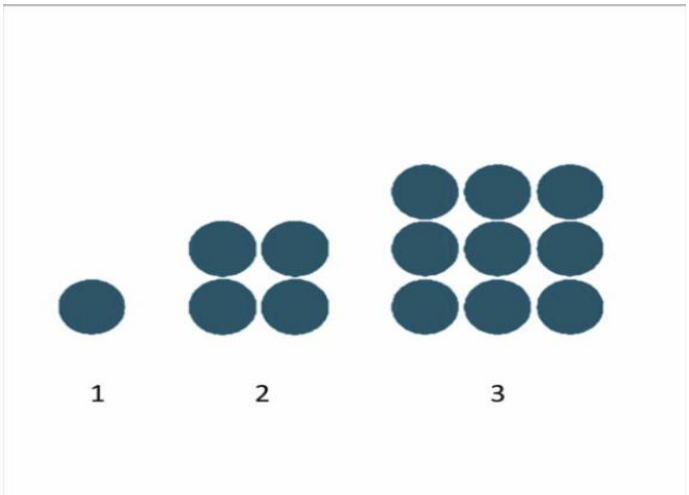
4.1 Gjennomføring av datainnsamling

I de første oppgavene i oppgavesettet var ikke hensikten at elevene skulle bruke programmering, men at de skulle gjenkalle tidligere kunnskap. Jeg ønsket imidlertid å ha med dialogen mellom elevene også i denne delen av oppgaveløsningen, selv om de ikke brukte programmering, startet opptakene allerede fra begynnelsen av undervisningsøkten.

Da oppgavene ble utleverte, begynte elevene straks å bruke ulike strategier for å løse oppgavene. Gruppe 2, Celine, Daniella og Erna gikk umiddelbart i gang med å diskutere hvordan de skulle løse oppgaven, mens gruppe 1, Alfred og Brage satt lenge stille før de foreslo forskjellige strategier. De brukte også lenger tid enn de andre gruppene før de begynte å skrive ned noe. Gruppe 3, Frida og Gina, lyttet til hverandre, de stilte spørsmål til hverandres utsagn, og de så ut til å bygge videre på det den andre sa. De la også tidlig fram mange ulike strategier for hverandre.

Den første oppgaven så slik ut:

Kvadrattall



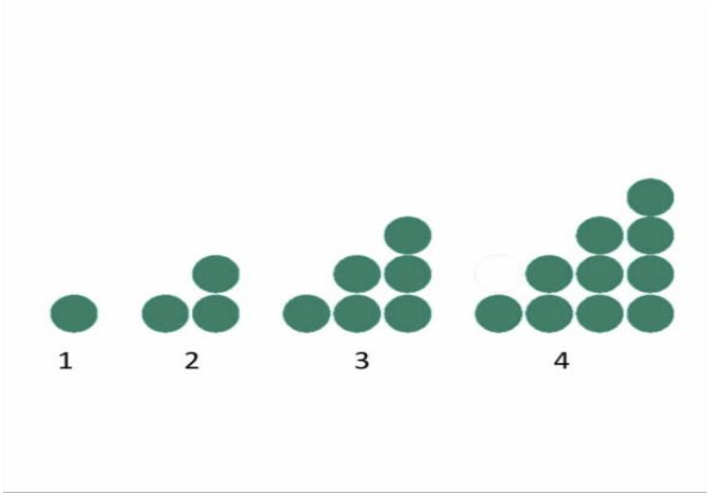
Figur 6 Kvadrattall, Lekolar Skole, Hentet fra https://lekolar-skole.no/figurtall/

1. Tegn kvadrattall nr. 4. Hvor mange prikker er det i kvadrattall nummer 4?
2. Diskuter i gruppen: hvordan utvikler figuren seg?
3. Lag en generell formel for n?
4. Hva med nummer 7 eller 12?

Den første oppgaven var i hovedsak ment for å gjenkalle kunnskap, og alle gruppene begynte med å se på mønsteret og diskutere ulike generaliserte formler. Etter relativt kort tid endte alle gruppene med $n * n$ som formel og brukte ikke formen n^2.

Den andre oppgaven elevene fikk, så slik ut:

Trekanttall



Figur 7 Trekanttall, Lekolar Skole, Hentet fra https://lekolar-skole.no/figurtall/

1. Tegn trekanttall nr. 5. Hvor mange prikker er det i trekanttall nummer 5?
2. Diskuter i gruppen hvordan utvikler figuren seg?
3. Lag en generell formel for n?
4. Hva med nummer 9 eller 19?

Også i denne oppgaven begynte gruppene å diskutere ulike strategier, de brukte telle strategier, tegne strategier, og de prøvet og feilet. Det utviklet seg en ganske høylytt diskusjon i begge jentegruppene, mens i guttegruppen satt den ene eleven for seg selv og jobbet på ark, mens den andre jobbet med å teste forskjellige fremgangsmåter i Python. Det viste seg at denne oppgaven bød på større utfordringer for alle tre gruppene, og oppgaven utviklet seg til å bli en problemløsningsoppgave som ingen av gruppene fikk helt til. Videre i oppgaven skulle de bruke programmering, og oppgaven lød som følger:

Python – Programmering (MU – Editor)

Oppgave 1

Bruk den generelle formelen for kvadrattall og lag ett program som beregner hvor mange prikker det er i kvadratene.

Kan du finne ut hvor mange prikker det er i figur 15, 27 og 88?

Oppgave 2

Bruk den generelle formelen for trekanttall og lag et program som beregner hvor mange prikker det er i trekantene.

Kan du finne ut hvor mange prikker det er i figur 13, 39 og 51?

Oppgave 3

Hvor mange prikker er det i femkant tall?
Diskuter hvordan figurene ser ut og utvikler seg.
Kan du finne ut hvor mange prikker det er i figur 3, 5 og 9?

Oppgave 4

Se om du kan justere koden under ¨turtle¨ til å tegne *figur 7* både som trekant og kvadrattall?

```
import turtle

def tegn_punkt(x, y):
    turtle.penup()
    turtle.goto(x, y)
    turtle.pendown()
    turtle.begin_fill()
    turtle.circle(5)  # Juster størrelsen på punktene etter behov
    turtle.end_fill()

def tegn_trekanttall(n):
    punkter = 0
    x, y = -100, 100  # Startposisjonen for tegningen
    avstand = 30  # Avstand mellom punktene

    for i in range(1, n + 1):
        for j in range(i):
            tegn_punkt(x + j * avstand, y)
            punkter += 1
        y -= avstand

    turtle.done()

# Tegn et trekanttall, for eksempel trekanttall for 5
tegn_trekanttall(5)
```

Figur 8 Programmeringsoppgave med å finne feil i koden

Gruppe 1, Alfred og Brage kom ganske raskt i gang, og de begynte med programmeringen for å teste de tidligere formlene de hadde jobbet med, men de kom ikke helt i mål med trekanttall oppgaven. De delte opp formelen i mindre deler og satt delene mer og mer sammen, til de fikk en kode som kunne gi trekanttall, og på denne måten kan en si at de brukte algoritmisk tenking.

Gruppe 2 Celine, Daniella og Erna, og gruppe 3, Frida og Gina, prøvde med flere forskjellige formler og brukte en helt annen tilnærming enn den gruppe 1 hadde brukt. Gruppene 2 og 3 prøvde mange ulike metoder for å løse oppgaven, de prøvde blant annet å bruke oppgave 4, for å se om de kunne få noen tips derfra. Når de jobbet med koden ble det observert at de strevde med det mest grunnleggende i kodingen, som for eksempel hvor de skulle sette inn tallet for det figurtallet de skulle undersøke. Dermed satte de tallet for figurtallet rett inn i koden, og ikke i tekstboksen i bunnen. De brukte feilsøkingen i programmet for å komme seg fremover i oppgaven, og de kom frem til en løsning til slutt.

Når gruppene kom til oppgaven som ba dem om å se på femkanttall, ble gruppe 1 ganske stille i en lengre periode. Som tidligere jobbet Brage med å generalisere en formel på ark,

mens Alfred prøvde ut flere forskjellige strategier med koding. Gruppe 2 brukte tid i starten der de så ut til at de ikke hadde framgang, de kom seg ikke videre. De hoppet etter en liten stund direkte til oppgave 4. I den oppgaven skulle de lage ett program som skulle visuelt fremstille kvadrattall og trekanttall. Løsningen var mer eller mindre avskrift av koden som sto på arket, med noen få justeringer som de selv måtte legge inn, for å få programmet til å bli riktig.

Etter cirka 40 minutter gikk jeg rundt i klasserommet og la ett løsningsforslag på bordet til de tre gruppene. Dette gjorde jeg for at den eller de gruppene som møtte på programmerings-barrierer eller matematiske barrierer kunne få hjelp til å komme seg videre, eller at de gruppene som hadde klart å komme frem til en kode kunne få se på hvordan jeg ville gjort det. Sekvenser fra denne delen bli analysert som hindringer i delkapittel 4.3.

Videre i oppgaven tar jeg for meg arbeidet til to av gruppene, gruppe 1 og deretter gruppe 2. Jeg valgte å kun ta med de to gruppene da jeg fant mye av de samme funnene fra den siste gruppen og derfor valgte jeg å ikke presentere data fra denne gruppen.. I 4.2 er fokuset i analysen på generaliseringsstrategier sammen med eierskap, engasjement og identitet. For så å svare på hvilke potensiale for dybdelæring som kan identifiseres i elevers arbeid med programmering i algebra. I 4.3 analyserer jeg elevenes møte med programmering og hvilke hindringer jeg observerte at de møtte på i arbeid med programmeringsoppgaver med utgangspunkt i kritisk tenking, engasjement og refleksjon.

4.2 Generaliseringsstrategier, eierskap, engasjement og identitet

I dette delkapittelet presenteres analyse av arbeidet til gruppe 1, Alfred og Brage. I oppgave 2 der Alfred og Brage jobbet med figurtalloppgavene, skulle de diskutere hvor mange prikker det er i trekanttall og hvordan figuren utviklet seg, for så å konstruere en generell formel for n. Først beskriver jeg hvordan de jobbet med å løse oppgaven, der jeg ser etter hvilke generaliseringsstrategier de brukte, og hvilke tegn på agens som kunne identifiseres i dialogen. Datagrunnlaget kom fram i både samtale og skjermbilde fra programmeringskodene de skrev underveis. Utdragene er valgt fordi de demonstrerer flere former for generaliseringsstrategier, og de viser elevenes ulike former for engasjement som kan være uttrykk for eierskap og agens.

Alfred og Brage løste oppgave 2 der de skulle bruke den generelle formelen for trekanttall og lage et program som beregner hvor mange prikker det var i trekantene. I utdrag 1 arbeidet de med å finne hvor mange prikker det er i figur 13, 39 og 51.

I denne oppgaven brukte de den generaliserte formelen de kom frem til, og så hvor mange prikker figur 13, 39 og 51 hadde. I forkant hadde de jobbet med kvadrattall og funnet en generalisert formel for kvadrattall, og hadde nå begynt på programmeringsbiten av trekanttall.

Utdrag 1:

Tabell 1 1. Utdrag fra analyse fra gruppe 1

A: Jeg fant den
 	
B: en halv N, jeg var inn på det da
 	
A: Ja, vil du vite hvor det sto heller? Hvis vi går tilbake på den siden du var.
 	
B: faen a, det sto der ja, en halv N. hmm..
 	
A: sjekk om den formelen kan stemme?
 	
B: Det er basicly det jeg har skrevet, bare at det aaa , ja ehm. Det funka litt men ikke helt.
 	
Ok, halvparten av.. okei, funker på første, okei funker der å, neste, funker der å. ehm 12, funker der å, skal vi prøve på 4 å, 2 ehm ja det funker.
	Koden for trekanttall
	`Trekanttall = input("skriv et tall:")`
	`Trekanttall = int(Trekanttall)`
	`Svar1 = Trekanttall * Svar1`
	`Svar2 = Svar1 + 1`
	`Svar3 = Svar1 * Svar2`
	`print(Svar3)`
	
	Løsning for 5
	`Skriv et tall5`
	`15.0`
	`>>>`
	
	Løsning for 19
	`Skriv et tall19`
	`190.0`
	`>>>`
	
	Løsning for 13
	`Skriv et tall13`
	`91.0`
	`>>>`

A: ja, da sto formelen rett der du har hatt pc'en de siste 20minuttene
 	
B: Jeg lagde noe som ligna da
 	
A: Hva var det du lagde da?
 	
B: Var ikke helt der, men det funka på noen av greiene
	Løsning for 51
	`Skriv et tall51`
	`1326.0`
	`>>>`

Elevene kommuniserte med hverandre underveis når de løste oppgaven, og de byttet på å kode. I den første delen var det Brage som skrev.

I denne analysen har jeg identifisert hvordan elevene Alfred og Brage engasjerte seg i oppgaver med algebra som tema, ved å bruke programmering som verktøy. Gjennom dialogen avdekket jeg deres anvendelse av generaliseringsstrategier. Dette inkluderte hvordan de tilnærmet seg problemene og utviklet løsninger ved å utforske ulike mønstre og sammenhenger.

Videre viste analysen av samme dialog hvordan Alfred og Brage manifesterte agens i sitt arbeid. Dette ble spesielt tydelig i måten de tok eierskap til læringsprosessen, viste utholdenhet i utforsking av programmeringsutfordringer, og reflekterte over sin rolle og identitet som matematikere og programmerere. Gjennom dialog mellom Alfred og Brage, testet Brage formelen "en halv N", og justerte denne basert på resultatene. Denne arbeidsmetoden reflekterer en strategi kjent som *Gjette og Sjekke*, og viser en tilnærming preget av utforsking og tilpasning.

Når det gjelder agens, det vil si eierskap, engasjement, og identitet, avdekkes det i dialogen flere viktige aspekter. Når Brage anerkjente behovet for å justere den opprinnelige formelen for optimal funksjonalitet, kan dette indikere at han har *eierskap* til læreprosessen. *Engasjement* kom til uttrykk gjennom den aktive dialogen mellom Alfred og Brage der de gjentok tester av løsninger, og viste en vilje til å revurdere og validere informasjon. Dette vistes også igjennom banning, som kan være tegn på at Brage følelsesmessig engasjerte seg. Ved hans vilje til å endre og hans følelsesutbrudd, kan dette ses som et tegn på vilje til å

arbeide videre for å klare oppgaven. Når han sier «Det er basicly det jeg har skrevet», sammenlignet han det han hadde gjort med formelen for trekanttall, som er algoritmisk tilnærming hvor han delte opp formelen i mindre deler. Løsningsforslaget så slik ut: n*(n+1)//2, så han manglet bare å dele på to for å finne svaret. Brages kommentar om å ha "lagd noe som ligna", en begynnende identifikasjon med rollen som problemløser *(identitet).* Ved å aktivt sammenligne sitt arbeid med et standardresultat og uttrykke en forståelse av likheten, demonstrerer Brage agens – hans kapasitet til å handle, tilpasse, og ta initiativ i sin læringsprosess. Dette er en indikator på at han begynner å se seg selv som en kompetent aktør som kan manipulere og utforske matematiske ideer uavhengig.

Et overraskende funn var at elevene ikke umiddelbart setter inn en generalisert formel direkte i programmet. I stedet bryter de ned problemet i mindre deler og generaliserer det steg for steg. Denne tilnærmingen er et tydelig eksempel på algoritmisk tenkning i praksis, hvor elevene anvender en metodisk prosess for å løse problemer.

Algoritmisk tenkning, definert som prosessen med å dele opp problemer i mindre, håndterbare enheter, utvikle løsningsstrategier og deretter implementere disse strategiene gjennom koding, blir her demonstrert på en måte som er tilgjengelig selv for dem uten erfaring med Python eller annen programmering. For eksempel, i stedet for å bruke den kjente formelen for trekanttall, $\frac{n(n+1)}{2}$,direkte i en enkel linje kode, tar elevene følgende skritt: Elevene starter med å identifisere og isolere deler av problemet. For trekanttall kunne dette innebære å først beregne produktet n×(n+1). Deretter beveger de seg gjennom en sekvens av koder, hvor de progressivt bygger opp til den fullstendige løsningen. De kan for eksempel skrive et lite program som først beregner n×(n+1), lagrer dette resultatet, og deretter deler det på 2 i en etterfølgende operasjon. I stedet for å skrive result = n * (n + 1) // 2 direkte.

Videre jobbet Alfred og Brage med neste oppgave som var å ta for seg femkanttall, hvor Alfred arbeidet med å løse denne med programmering og Brage arbeidet med å løse den på papir.

Oppgave 3
Hvor mange prikker er det i femkant tall?
Diskuter hvordan figurene ser ut og utvikler seg.
Kan du finne ut hvor mange prikker det er i figur 3, 5 og 9?

Dette var en av de siste oppgavene de skulle jobbe med, og timen gikk mot slutten. Elevene fikk utdelt ett løsningsforslag for å få litt tips til veien videre, eller eventuelt til å se hva de gjorde forskjellig fra løsningsforslaget. De testet flere strategier underveis, både ved *telling* og *Gjette og Sjekke* - strategien.

I *Gjette og Sjekke* strategien innebærer det at elevene prøver seg frem med forskjellige løsninger for å se hva som fungerer. Brages kommentar om at løsningen "funka på noen av greiene" viser til denne tilnærmingen. Elevene tester ulike løsninger og observerer resultatene. Når Brage bemerker at noe fungerer, viser det at de er engasjert i en iterativ prosess, hvor de justerer og prøver igjen basert på tilbakemeldingene de får fra deres forsøk. Dette er ikke bare en mekanisk prosess; det krever at elevene tenker kritisk om hvorfor visse løsninger fungerer og andre ikke, og tilpasser deres tilnærming deretter.

Når Brage sier at løsningen "funka på noen av greiene", indikerer dette også en kontekstuell forståelse. Elevene anvender ikke bare en formel mekanisk, men de reflekterer over og justerer deres tilnærming basert på den spesifikke konteksten av problemet de står overfor. De bruker deres forståelse av problemet til å veilede deres problemløsning, og de tilpasser seg når de ser at deres første tilnærming ikke løser alle aspekter av problemet. Dette viser en adaptiv og kontekstbasert anvendelse av matematiske konsepter, som er avgjørende for dybdelæring.

Sammenhengen mellom de anvendte strategiene *Gjette og Sjekke* og *Kontekstuell,* og manifestasjonen av agens ble identifisert i deres problemløsning. Brages erkjennelse av at deres løsning "funka på noen av greiene", og tilpasningen basert på det, reflekterer en prosess av dypere forståelse og personlig tilpasning. Det understreker hvordan anvendelsen av spesifikke generaliseringsstrategier ikke bare er teknikker for problemløsning, men også kan være en bidragsyter for å skape en dypere forståelse.

Utdrag 2

Tabell 2 2. Utdrag fra analyse gruppe 1

A: vi har fasiten jo
 	
B: men er det ikke morsommere å gjøre det sjøl?
 	
A: Ja, vi har gjort det på to helt forskjellige måter uansett
 	
B: er lik femkanttall delt på 2, også femkanttall.
 	
Svar 1 ganger er lik 3 ganger femkanttall, minus 1
 	
Sånn
 	
Hvis vi prøver på ett av de vi vet da
 	
4
 	
22.
 	
Ja, funker
 	
A: 5
 	
35
 	
B: Ja, vi fikk det til
	Kode for femkanttall
	
	`Femkanttall = input("Skriv et tall: ")`
	`Femkanttall = int(Femkanttall)`
	`Svar1 = Femkanttall/2(3 * Femkanttall - 1)`
	
	`print(Svar)`
	
	Løsning for 5
	
	`Skriv et tall: 5`
	`35.0`
	`>>>`
	
	`Femkanttall = input("Skriv et tall: ")`
	`Femkanttall = int(Femkanttall)`
	`Svar1 = Femkanttall/2`
	`Svar2 = 3 * Femkanttall - 1`
	`Svar3 = Svar1 * Svar2`
	`print(Svar3)`
	
	`Running: afewvfawveafew.py`
	`Skriv et tall: 6`
	`51.0`
	`>>>`

Liten klapp på slutten.	

I denne dialogen mellom Alfred og Brage identifiseres et matematisk problem gjennom samarbeid og individuell tenking, der de engasjerte seg i en læringsprosess ved å utforske femkanttall. Alfred påpekte at de allerede hadde fasiten, mens Brage uttrykte at det var mer givende å løse problemet selv. Det viser et ønske om dybdelæring og forståelse, framfor å bare kjenne svaret.

De hadde begge tilnærmet seg problemet på ulike måter, og det illustrerer mangfoldet i matematisk tenkning og problemløsning. Brage introduserte en formel for å beregne femkanttall, som de deretter testet med konkrete eksempler. Det demonstrerer en anvendelse av *Gjette og Sjekke*-strategien, der de eksperimenterte med tall for å se om de passet med den foreslåtte formelen.

Når de brukte spesifikke talleksempler, som for eksempel 4 og 5, og fant de tilsvarende femkanttallene, bekreftet elevene at tilnærmingen deres fungerte i praksis. Denne prosessen engasjerte dem ikke bare i aktiv problemløsning, men hadde også potensiale til å fremme en dypere forståelse av matematiske konsepter ved å anvende dem direkte.

Avslutningen med et "liten klapp på skulderen" symboliserte ikke bare en felles anerkjennelse av innsats og suksess, men var også et tegn på den indre motivasjonen som drev dem. Dette øyeblikket reflekterer viktigheten av motivasjon og positiv forsterkning i læringsprosesser, som er essensielt for å opprettholde engasjementet og fremme en følelse av prestasjon. Interessant nok søkte de ikke ekstern anerkjennelse fra verken lærer eller medelever; deres egen lille feiring var tilstrekkelig. Det viser at de har tatt eierskap til både prosessen og resultatet av sitt arbeid. Umiddelbart etter dette gikk de videre til neste problem og begynte å kode, noe som ytterligere understreker deres selvstendighet og engasjement i læringsprosessen.

En annen observasjon av denne gruppens arbeid, var at det stadig kom korte perioder hvor det var helt stille, og det skjedde når de møtte på problemer. Brage begynte umiddelbart å jobbe på papir, og så ut til å tenke veldig, ved at han fokuserte på arket når han arbeidet. Alfred derimot, testet i programmet for å komme frem med en løsning. I disse situasjonene kunne det være perioder opp mot 8-10 minutter hvor det ikke var noe form for kommunikasjon mellom dem. Så selv om denne dialogen samlet sett viste hvordan de arbeidet utforskende og

samarbeidet, noe som kan fremme dybdelæring og forståelse, viste arbeidet deres samtidig tegn til selvstendig tenkning og problemløsning, der de i etterkant diskuterte og kom fram til felles løsning.

4.3 Hindringer som grobunn for dybdelæring og kritisk tenking

I dette delkapittelet er data som presenteres fra gruppe 2, Celine, Daniella og Erna. Denne gruppen startet med at de viste mye engasjement i form av lydnivå, bevegelse og energi. De gikk rett på for å løse oppgavene. Her vil jeg gi en gjennomgang av hvordan de arbeidet med oppgavene, så vil jeg analysere utsnitt av programmeringen i sammenheng med transkripsjonen, der har jeg trukket ut de situasjonene hvor gruppen møter på hindringer. Jeg undersøker hvordan de løser disse hindringene og ser dette i sammenheng med identifisering av engasjement og eierskap.

Denne gruppen viste engasjement i form av høyt lydnivå, bevegelse, de var målrettet og de begynte umiddelbart å løse oppgavene. De samarbeidet bra, og lot alle komme til ordet og alle skulle få gjennomslag for å prøve sin ide. De byttet på å jobbe direkte med programmeringen og hvem som snakket. Det som preget denne gruppen, var at de viste manglende programmeringsferdigheter. I tillegg strevde de når det var meningen at de skulle gjenkalle kunnskap. De kom frem til en formel for kvadrattall, men viste at de strevde med å få programmet til å fungere riktig. Først etter 17 minutters arbeid med programmeringsoppgavene, begynte programmeringen å gå lettere. Før dette skrev de inn de tallene som de skulle regne på rett inn i koden manuelt. De jobbet også med forskjellige versjoner av den generaliserte formelen de hadde kommet frem til, se Tabell 3.

Elevene engasjerte seg i diskusjon om formelen for kvadrattall og trekanttall, og hvordan disse formlene kunne implementeres i Python, noe utdraget under demonstrerer.

Her møter gruppen syntax hindringer, og vet ikke helt hva de gjør galt.

Utdrag fra data:

Tabell 3 Utdrag fra analyse gruppe 2

C: Skal vi sjekke nå om hvis jeg gjør sånn her. Bare har den sånn.	

D: Se om det kommer noe der.

D: Da ble det noe helt annet.

D: Nei.

E: Det er sikkert det som er riktig.

D: Nei, det var det ikke.

D: Nei, hallo, vi gidder ikke.

E: Nei, nei, nei, nei, nei.

D: Vi gidder ikke å gjøre det på nytt.

D: Vi skulle slå oss til.

C: Da har vi gjort det feil,

C: da har vi gjort det feil.

D: Sånn er det vel.

E: Jeg vet ikke,

D: vi har lært av feilen vår.

```python
import math
def antall_prikker_kvadrat(n):
    return n*88*2
print("kvadrattall for figur 88:",antall_prikker_kvadrat(88))
```

```
Running: matte.py
kvadrattall for figur 88: 15488
>>>
```

Her har de oppdaget feilen og endre koden slik at den fungerer.

```python
import math
def antall_prikker_trekanttall(n):
    return n**2
print("trekanttall for figur 51:",antall_prikker_trekanttall(51))
```

```
Running: matte.py
trekanttall for figur 51: 2601
>>>
```

C: Det er sånn her vi skal gjøre det på trekant. D: Hva var det jeg sa? D: Vi skal bruke den formelen. E: Åja. C: Haha.	

Under programmeringsøkten observerte jeg at elever støtte på en spesifikk "Error"-melding da de forsøkte å kjøre sitt Python-program. Meldingen indikerte en syntaksfeil. Elevene reagerte først med forvirring, men snart begynte de å identifisere og korrigere feilen. Det ledet til en gruppediskusjon hvor de vurderte ulike måter variablene kunne være feilaktig på.

Ved å observere variablene og diskutere mulige feil, begynte elevene å teste ut nye tilnærminger. Celine sa: "Skal vi sjekke nå om hvis jeg gjør sånn her?" etterfulgt av å endre variabelnavnet. De kjørte koden igjen for å se om endringen løste problemet. Denne tilnærmingen av å stille hypoteser og umiddelbart teste dem viser en forståelsesprosess og en tilnærming til problemløsning.

Gjennom denne episoden demonstrerte elevene evnen til å tenke kritisk om deres eget arbeid ved å analysere feilmeldingen, diskutere potensielle årsaker, og strategisk modifisere koden. Denne interaksjonen reflekterer en kritisk tankeprosess der de ikke bare identifiserer feil, men også vurderer og implementerer løsninger, og reflekterer over effekten av disse løsningene.

Denne hendelsen understreker hvordan programmering kan støtte utviklingen av kritisk tenkning og problemløsningsevner hos elever. Det viser hvordan de anvender matematiske og logiske konsepter praktisk, og hvordan de engasjerer seg i dybdelæring ved å aktivt utforske, anvende og reflektere over deres læring i reelle problemløsningssituasjoner. Disse ferdighetene er avgjørende for forståelse, analyse og anvendelse av kunnskap, noe som er sentralt i prosessen med dybdelæring.

De møtte på barrierer og hindringer i form av at de bygget programmer og oversatte matematiske konsepter til kode, som kan være en solid ressurs for å fremme dybdelæring. Gjennom at elevene møtte, og overvant disse syntax hindringene, engasjerte de seg i problemløsning og refleksjon. Det kan bidra til å utvikle deres kritiske tenkning og evne til å se sammenhenger mellom matematiske konsepter og programmering. Når elevene konfronterte og arbeidet seg gjennom disse barrierene, kan de oppnå en dypere forståelse av både matematikk og programmering.

Elevenes arbeid med programmeringsoppgaver involverer en prosess der de først står overfor problemer, som når programmet gir en "Error" som tilbakemelding. En slik type utfordring fører de inn i en dialog om årsaken til problemet og potensielle løsninger. For eksempel, diskusjonene de fører om å skrive om variablene eller teste nye tilnærminger – "Skal vi sjekke nå om hvis jeg gjør sånn her?" illustrerer en aktiv utforskning av muligheter og tilnærminger. Gjennom denne prosessen arbeider elevene ikke bare med å identifisere og forstå hva som er galt, men også med å tenke kritisk om hvordan de kan løse problemet. De tester sine hypoteser og vurderer resultatene, noe som krever at de analyserer og reflekterer over både problemet og løsningene de foreslår. Det viser hvorfor kritisk tenkning er viktig og kan være med på å skape dybdelæring.

Til slutt kom elevenes evne til refleksjon fram i prosessen med feilsøking og vurdering av egne løsningsforslag, som for eksempel når de diskuterte potensielle feil og justeringer i koden. Det kommer blant annet til uttrykk i utsagnet: " vi har lært av feilen vår". Elevene viste dermed en metakognitiv tilnærming til læring, i og med at de tenkte over sin egen tenkning og læringsprosess, noe som også er en viktig del av dybdelæring.

Disse eksemplene viser hvordan integrering av programmering i matematikkundervisningen kan fremme dybdelæring ved å engasjere elever i komplekse problemløsningsprosesser, anvende og utvide faglig kunnskap, samt stimulere til refleksjon over egen læring.

Gjennom analysen av utdraga, kommer det fram hvordan integrering av programmering i matematikkundervisningen har potensiale til å fremme dybdelæring. Det kan skje ved at elevene møter hindringer. Som igjen kan føre til frustrasjon «vi gidder ikke gjøre det på nytt», men også nye muligheter for å tenke kritisk på hva en har gjort og motivasjon til å prøve ut nye tilnærminger. Hindringer kan bidra til å stimulere elevens refleksjon over egne læring «Vi har lært av feilen vår» som er viktig del av dybdelæring.

5.0 Diskusjon

Denne studien har undersøkt potensialet for dybdelæring i algebra gjennom elevers arbeid med Python-programmering. Ved en kvalitativ tilnærming, i analyse av observasjon av skjermopptak og lydopptak, har jeg observert hvordan elever engasjerer seg med og at det kan oppnås dybdelæring når de anvender programmering. Dette arbeidet er forankret i en pedagogisk kontekst som verdsetter dybdelæring, som definert av Kunnskapsløftet 2020, samt Schoenfelds teori om robust læring.

Først i dette kapittelet vil jeg ta for meg funnene presentert i resultatkapittelet ved å knytte dem sammen med teorien introdusert i teorikapittelet. Det gjøres med det overordnede målet om å utforske følgende forskningsspørsmål; hvilket potensiale for dybdelæring kan identifiseres i elevers arbeid med programmering i algebra? For det andre, ser jeg på hvilken måte kan hindringer gi grobunn for dybdelæring.

Videre har jeg delt opp kapittelet i fire deler, hvor jeg ser på hva masteroppgaven har bidratt til av ny forståelse, hvilken påvirkning masteroppgaven har gjort på min lærerrolle, hvordan masteroppgaven kan bidra til forskningsfeltet og til slutt videre forskning.

5.1 Hvilket potensiale for dybdelæring kan identifiseres i elevers arbeid med programmering i algebra?

I dialogen mellom Alfred og Brage, anvender og justerer de generaliseringsstrategier som ved "en halv N" for trekanttall, det illustrerer et engasjement med matematiske konsepter og en prosess som kan vise potensiale for dybdelæring. Dybdelæring, karakterisert ved en søken etter å forstå materiale på et dypere nivå fremfor overflatisk memorering, er avgjørende i møte med samfunnets kompleksitet og teknologisk utvikling (Marton & Säljö, 1976; Sawyer, 2006). Alfred og Brages eksperimentelle tilnærming reflekterer denne formen for læring, hvor aktiv utforskning og praktisk anvendelse av teori kan bidra til utvikling av både algebraiske ferdigheter og kritisk tenkning.

Schoenfeld (2018) og Munthe (2022) understreker viktigheten av robust læring, hvor praktisk anvendelse av matematiske konsepter, som programmering, fremmer dypere forståelse og refleksjon over løsningsmetoder. Gjennom dialog og følelsesmessige reaksjoner som frustrasjon viser elever som Alfred og Brage en involvering i læringsprosessen, en prosess som indikerer både agens og eierskap over egen læring.

Deres samarbeid og selvstendige problemløsning fremhever samspillet mellom kollektiv og individuell innsats i læring. Schoenfeld (2018) poengterer at samarbeidsprosesser er avgjørende for utvikling av kritisk tenkning og problemløsningsferdigheter, som er essensielle aspekter ved dybdelæring. Videre fremheves motivasjon og engasjement ved at elevene viser en indre drivkraft og feirer sine suksesser, dette støtter opp under en positiv læringsatmosfære som kan fremme dybdelæring og takling av komplekse problemstillinger.

Alfred og Brage viser ansvar for sin læring og viser en vilje til å forbedre sin forståelse ved å utforske, reflektere, og revurdere informasjon. Det reflekterer kjerneprinsippene i teorien om dybdelæring og understreker betydningen av både samarbeid og selvstendig tenkning for å utvikle en dypere forståelse og ferdigheter i matematikk og programmering.

Kaufmann et al.'s (2018) beskriver algoritmisk tenkning som å utvikle evnen til å analysere, forstå, og løse problemer på en strukturert måte, dette reflekteres også i elevenes arbeid. Ved å ikke umiddelbart sette inn en generalisert formel, men heller utforske problemet gjennom programmering, benytter elevene seg av en læringsprosess som kan skape dybdelæring. Det viser hvordan programmering kan tjene som et verktøy for læring, ved å tillate elever å eksperimentere, feile, og til slutt forstå den underliggende logikken og strukturen i matematiske problemstillinger.

Til slutt, ved å bryte fri fra den vanlige "linje-for-linje" tilnærmingen, som ofte kan resultere i en oppstykket forståelse av programmering og matematiske konsepter, har jeg observert at elevene viser evne til å se sammenhenger. De anvender programmering ikke bare som en teknikk for å utføre løsninger, men som et verktøy for å utdype forståelsen av de utfordringene de møter. Denne innsikten understreker betydningen av algoritmisk tenking for å stimulere kreativitet, kritisk refleksjon og læring i det 21. århundre, slik Lahtinen et al. (2005) fremhever.

5.2 På hvilken måte kan hindringer gi grobunn for dybdelæring?

Kritisk tenkning og engasjement er fundamentale i utviklingen av dybdelæring, spesielt i konteksten av programmering og matematikkundervisning. Når elever konfronterer programmeringsfeil og logiske utfordringer, illustrerer de anvendelsen av kritisk tenkning ved å analysere, evaluere og løse komplekse problemer på en reflektert måte. Dette engasjementet i feilsøking og problemløsning reflekterer ikke bare deres evne til å anvende kritisk tenkning,

men også en involvering i læringsprosessen som er avgjørende for å oppnå dybdelæring (Munthe, 2022; Schoenfeld, 2018).

Robust læring, som understreker viktigheten av problemkonfrontasjon og -løsning, sammen med teorier om agens, eierskap og identitet, fremmer elevens evne til selvrefleksjon og selvregulert læring. Dette bidrar til utviklingen av elevens kritiske tenkning og motivasjon, som er nøkkelen til dybdelæring. Aktivt engasjement i problemløsning fører ikke bare til en sterkere motivasjon og interesse for læring, men styrker også elevens evne til å takle komplekse utfordringer (Schoenfeld, 2018).

Videre, når elevene reflekterer over og evaluerer egne løsningsstrategier, demonstrerer de en metakognitiv tilnærming som er essensiell for dybdelæring i matematikk. Det omfatter kontinuerlig feilsøking og vurdering av løsninger, noe som fremmer en dypere forståelse av matematiske konsepter og forbedrer problemløsningsferdigheter (Schoenfeld, 2018; Munthe, 2022).

Integrasjonen av programmering i matematikkundervisningen tilbyr en unik mulighet for elever til å utforske matematiske konsepter gjennom praktisk anvendelse, som støttes av forskning fra Munthe (2022), Kaufmann og Stenseth (2020) og Benton (2016). Denne tilnærmingen bidrar til å overvinne læringens barrierer og fremmer en helhetlig forståelse av både matematikk og programmering. Schoenfeld (2018) peker på potensielle utfordringer knyttet til hvordan programmeringsintegrasjon kan føre til en overfladisk forståelse hvis den ikke støttes med passende pedagogiske strategier og ressurser. Han argumenterer for at uten grundige forberedelser og kontinuerlig støtte, kan integreringen av programmering i matematikk skape ytterligere forvirring hos elever som allerede sliter med faget, og kan dermed forsterke eksisterende hindringer.

Det som er interessant er at gjennom prosessen med feilsøking, tilpasning av kode og et tydelig engasjement i læringsprosessen, blir elevenes møte med og overvinning av hindringer og barrierer en viktig ressurs i å fremme dybdelæring. Dette engasjementet i problemløsning og refleksjon er nøkkelen til å utvikle kritisk tenkning og evnen til å se sammenhenger mellom matematikk og programmering, hvilket kan føre til en dypere forståelse av begge felter.

Ved å analysere disse resultatene i lys av de ontologiske hindringene beskrevet i kapittel 2.4, kan vi se en direkte kobling mellom de teoretiske konseptene og elevers praktiske erfaringer. Ontologiske hindringer, som oppstår når kompleksiteten eller antallet barrierer blir så

overveldende at det virker uoverkommelig, synes å være en del av læringsprosessen i programmering. Denne utfordringen blir imidlertid møtt med en tilnærming der elever aktivt søker løsninger, tilpasser sine strategier og engasjerer seg i metakognitiv refleksjon – alle aspekter som er kritiske for dybdelæring.

Det interessante med elevenes tilnærming til disse utfordringene er hvordan de navigerer mellom "overkommelige" og "uoverkommelige" barrierer, som Ko et al. (2004) diskuterer. Funnene indikerer at ved møte på en barriere eller eventuelt hindringer, var det forskjellige hindringer som genererte til videre engasjement og motivasjon til å løse oppgaven. Når elevene møtte på barrierer var det av tekniske barrierer, ved manglende kunnskap om programmering eller ved for vanskelige matematiske oppgaver. Dette løste elevene ofte med å gå videre til neste oppgave. Ved aktivt å engasjere seg i feilsøking og refleksjon over egne løsningsforslag, demonstrerer elevene en prosess for å overvinne de overkommelige barrierene. Denne prosessen, hvor de lærer av sine feil og justerer sine tilnærminger, er i tråd med hva Brousseau (1997) beskriver som nødvendig for tilpasningene gjennom oppgavene.

Mine funn underbygger tydelig det Buitrago Flórez et al. (2017) har påpekt at utfordringer og frustrasjoner utgjør en essensiell del av læringsprosessen i programmering. Det bidrar ikke kun til en dypere forståelse, men også til utviklingen av kritisk tenkning, problemløsningsevner og en positiv holdning til læring. Munthe fremhever at selv om programmering kan styrke matematisk forståelse og problemløsning, kan det oppstå utfordringer knyttet til at elever ofte kan finne programmeringsspråk og konsepter abstrakte. Dermed også vanskelige å relatere til konkrete matematiske problemstillinger. Det kan føre til økt frustrasjon og hindre læring hvis ikke undervisningsmetodene tilpasses for å møte disse utfordringene.

5.3 Ny forståelse

Resultatet fra denne masteroppgaven gir en forståelse av potensialet for dybdelæring gjennom elevers arbeid med programmering, spesielt med fokus på algebra. Gjennom bruk av teorier om dybdelæring, algebra og programmering, samt en analyse av tidligere forskning innen feltet sammen med egen empiri og analyse, bidrar masteroppgaven til å gi innsikt i hvordan elever kan utvikle en dypere forståelse av matematiske konsepter gjennom programmering. Ved å kombinere konsepter som generaliseringsstrategier og elevers agens i læreprosessen, viser masteroppgaven hvordan programmering kan være en metode for å fremme dybdelæring og styrke elevenes matematiske forståelse. Den nye forståelsen viser potensialet for dybdelæring i matematikkundervisningen gjennom integrering av programmering, og gir innsikt i hvordan elever kan engasjeres i problemløsningsprosesser og refleksjon over egen læring for å oppnå en dypere forståelse av algebra og matematikk generelt.

Det er imidlertid også klart fra denne studien at selv om programmering byr på betydelige pedagogiske muligheter, møter elever en rekke utfordringer som kan dempe denne læringsfordelen. Teknologiske barrierer, manglende læreropplæring og elevers variable tilgang til digitale ressurser har vist seg å begrense mulighetene for effektiv implementering av programmering i matematikkundervisningen. Disse utfordringene er støttet av lignende funn i studier utført av forskere som Benton et al. (2016), som påpeker at manglende infrastruktur og ressurser kan undergrave læringsutbyttet betydelig. Misfeldt og Ejsing-Duun (2015) identifiserer et fenomen de kaller "the planning paradox," som oppstår når læreres ønske om å nå matematiske mål eller deres fokus på matematikk resulterer i overdreven detaljert planlegging. Det kan paradoksalt nok virke mot sin hensikt ved å redusere elevenes oppfatning av programmering som et nyttig verktøy. Det reiser spørsmålet om hvorvidt de strenge rammene i oppgavedesignet faktisk kan være begrensende for noen elever.

I likhet med Munthe sin avhandling (2022) blir det tydelig hvilke utfordringer elever står overfor når de navigerer i programmeringsoppgaver, og hvordan det å overvinne disse hindringene kan styrke kritiske tenkeferdigheter og problemløsningsevner. Munthe understreker at motgang ikke bare er en uunngåelig del av læringsprosessen, men også en essensiell faktor for å fremme dybdelæring. Mine observasjoner og analyser i denne masteroppgaven korresponderer med Munthes (2022) fremheving av generaliseringsstrategier som et middel for å oppnå dybdelæring. Det underbygger argumentet om hvordan programmering kan være et effektivt verktøy for å fremme en aktiv og engasjerende læringsprosess. Gjennom mitt arbeid blir det tydelig at anvendelsen av programmering i

utdanning ikke bare er et pedagogisk valg, men en strategi som kan forsterke læringsdybden og engasjementet hos elevene, et synspunkt som også Munthe fremhever.

5.4 Styrker, svakheter og videre forskning

I fremtidig forskning vil det være behov for mer forskning på ulike typer programmeringsoppgaver spesifikt kan designes for andre temaer i matematikk, som kan være med på å skape dybdelæring. Det er også viktig å adressere de utfordringene elever møter, og utvikle strategier for å støtte elever gjennom disse utfordringene for å sikre at alle elever kan dra nytte av potensialet som ligger i å integrere programmering i matematikkundervisningen.

Masteroppgaven kombinerer programmering med algebraundervisning, noe som er relevant i dagens digitale samfunn. Det tilrettelegger for dybdelæring ved å koble teoretisk kunnskap med praktisk anvendelse. Den inneholder konkrete eksempler på hvordan Python-programmering kan integreres i matematikkundervisningen, noe som kan være til stor hjelp for lærere som vil implementere lignende tiltak.

Masteroppgaven identifiserer teknologiske barrierer og utforming av relevante oppgaver som utfordringer. Det kunne vært nyttig med en dypere utforskning av hvordan disse barrierene kan overkommes praktisk i skolekontekst. Selv om elevstemmen kommer fram i elevers diskusjoner underveis i oppgaveløsningen, kunne masteroppgaven ha tjent på mer direkte innsikt fra elevperspektivet, for eksempel gjennom elevintervjuer eller spørreundersøkelser, for å få en dypere forståelse av hvordan de opplever denne formen for læring. Masteroppgaven kunne også ha dratt nytte av å diskutere hvordan nyere utviklinger innen teknologi og pedagogikk, som kunstig intelligens i utdanningen, kan påvirke fremtidens programmeringsundervisning.

6.0 Konklusjon

Denne masteroppgaven har undersøkt potensialet for dybdelæring gjennom bruk av Python-programmering i algebraundervisningen for ungdomsskoleelever. Resultatene viser at integrering av programmering ikke bare kan styrke elevers forståelse av matematiske konsepter, men også fremme deres problemløsende ferdigheter og kritiske tenkning. Disse funnene adresserer problemstillingen ved å bekrefte at både dybdelæring og overvinning av hindringer er mulig gjennom målrettet pedagogisk bruk av programmering i matematikken.

Studien tilbyr forslag til strategier for hvordan lærere kan implementere programmering i matematikkundervisningen, noe som kan overvinne pedagogiske barrierer og styrke læringsmiljøer. Ved å identifisere utfordringer og tilby løsninger, gir oppgaven praktiske verktøy for lærere som søker å integrere teknologi i undervisningen.

Som lærer har jeg gjennom masteroppgaven erfart at det er mulig å integrere programmering, spesielt Python, i matematikkundervisningen for å engasjere elevene på en interaktiv måte og fremme dybdelæring i algebra. Jeg har lært at jeg som lærer kan veilede elevene i å bruke generaliseringsstrategier som en metode for å utvikle en dypere forståelse av matematiske konsepter. Underveis har jeg blitt mer bevisst på å fokusere på elevers agens i læreprosessen, der jeg kan oppmuntre elevene til å ta eierskap over sitt eget læringsarbeid og velge strategier som passer deres individuelle behov og læringsstil. Gjennom disse tiltakene har jeg lært at jeg kan jeg bidra til å skape et læringsmiljø som fremmer dybdelæring, kritisk tenkning og praktisk anvendelse av matematiske konsepter blant elevene.

I konklusjonen viser denne masteroppgaven hvordan programmering i matematikkundervisning har potensiale til å være et verktøy for å fremme dybdelæring i matematikk, spesielt innenfor området algebra. Ved å integrere teoretiske rammeverk gir oppgaven innsikt i hvordan lærere kan implementere programmering i undervisningen for å styrke elevers forståelse og utvikle deres kritiske tenkning. Videre belyser oppgaven praktiske tilnærminger og utfordringer ved slik integrasjon, og presenterer strategier for å overvinne disse barrierene. Gjennom disse funnene bidrar masteroppgaven med konkrete verktøy og perspektiver som kan anvendes av lærere og skoleledere for å utforske og forbedre undervisningspraksiser.

7.0 Referanser

Aubert, K. E. (2009). Algebra. Hentet Januar 10, 2022, fra https://snl.no/algebra

Balacheff, N. (1990). Towards a problématique for research on mathematics teaching. *Journal for research in Mathematics Education*, 21(4), 258-272.

Blanton, M. L., & Kaput, J. J. (2005). Characterizing a classroom practice that promotes algebraic reasoning. *Journal for research in mathematics education*, 36(5), 412-446.

Blanton, M. L., & Kaput, J. J. (2011). Functional thinking as a route into algebra in the elementary grades. *In Early algebraization: A global dialogue from multiple perspectives* (s. 5-23). Berlin, Heidelberg: Springer Berlin Heidelberg.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2016, February). Building mathematical knowledge with programming: insights from the ScratchMaths project. Suksapattana Foundation.

Berry, J., & Sahlberg, P. (2006). Accountability affects the use of small group learning in school mathematics. *Nordic Studies in Mathematics Education*, 11(1), 5-31.

Brekke, G., & Gjone, G. (2001). Matematikk. I S. Sjøberg (Red.), *Fagdebatikk: Fagdidaktisk innføring i sentrale skolefag* (s. 215-265). Oslo: Gyldendal Akademisk.

Bocconi, S., Chioccariello, A., & Earp, J. (2018). The Nordic approach to introducing Computational Thinking and programming in compulsory education. *Report prepared for the Nordic@ BETT2018 Steering Group*, 42.

Bosse, Y., & Gerosa, M. A. (2017). Why is programming so difficult to learn? Patterns of difficulties related to programming learning mid-stage. *ACM SIGSOFT Software Engineering Notes*, 41(6), 1–6.

Brousseau, G. (1997). *Theory of didactical situations in mathematics: Didactique des mathématiques, 1970–1990* (M. Cooper, N. Balacheff, R. Sutherland, & V. Warfield, Trans.). Kluwer Academic Publishers.

Brousseau, G. (2008). Research in mathematics education. *In ICME-10 Proceedings* (s. 244-254). Roskilde University

Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a generation's way of thinking: Teaching computational thinking through programming. *Review of Educational Research*, 87(4), 834-860.

Carraher, D. W., & Schliemann, A. D. (2018). Cultivating early algebraic thinking. *Teaching and Learning Algebraic Thinking with 5-to 12-Year-Olds: The Global Evolution of an Emerging Field of Research and Practice*, 107-138.

Creswell, J.W. (2013) *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. 4th Edition, SAGE Publications, Inc., London.

Cohen, L., Manion, L., & Morrison, K. (2007). *Research methods in education* (6. edition). Abingdon: Routledge.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T. W., & Selby, C. (2015). *Computing at school: Computational thinking*. London Knowledge Lab.

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

Grønmo, S. (2016). *Samfunnsvitenskapelige metoder* (2. utg.). Fagbokforlaget.

Haraldsrud, A. D., Sveinsson, H. A., & Løvold, H. H. (2022). *Programmering i skolen*. Universitetsforlaget.

Harel, G., & Sowder, L. (2013). Advanced mathematical-thinking at any age: Its nature and its development. In Advanced Mathematical Thinking (s. 27-50). Routledge.

Hiebert, J., & Lefevre, P. (2013). Conceptual and procedural knowledge in mathematics: An introductory analysis. In Conceptual and procedural knowledge (s. 1-27). Routledge.

Kaput, J. J., Carraher, D. W., & Blanton, M. L. (Eds.). (2017). *Algebra in the early grades*. Routledge.

Kaufmann, O. T., Stenseth, B., & Holone, H. (2018). Programmering i matematikkundervisningen. I A. Norstein & FO Haara (Red.). *Matematikkundervisning i en digital verden*, 73-95.

Kaufmann, O. T., & Stenseth, B. (2020). Programming in mathematics education. *International Journal of Mathematical Education in Science and Technology.* https://doi.org/10.1080/0020739X.2020.1736349

Kaufmann, O. T. Stenseth, B. og Forsström (2022). Programmering I matematikkundervisningen. Gustavsen, T. S., Rinvold, R. A., Hinna, K., & Sundtjønn, T. (Red.) *QED 1-7: matematikk for grunnskolelærerutdanningen: Bind 1: Vol. Bind 1* (2. utgave.). Cappelen Damm akademisk.

Ko, A. J., Myers, B. A., & Aung, H. H. (2004). Six learning barriers in end-user programming systems. *In Proceedings of the 2004 IEEE Symposium on Visual Languages-Human Centric Computing.*

Kunnskapsdepartementet. (2020a, 13. mars). Dybdelæring. Hentet 31. desember 2023, fra https://www.udir.no/laring-og-trivsel/dybdelaring/

Kunnskapsdepartementet. (2020b). Matematikk 1–10 (MAT01 05). Hentet 31. desember 2023, fra https://www.udir.no/lk20/mat01-05/kompetansemaal-og-vurdering/kv16/

Küchemann, D. (1978). Children's understanding of numerical variables. *Mathematics in School,* 7(4), 23-26. Hentet fra http://www.jstor.org/stable/30213397

Lannin, J. K. (2005). Generalization and justification: The challenge of introducing algebraic reasoning through patterning activities. *Mathematical Thinking and Learning*, 7(3), 231-258. https://doi.org/10.1207/s15327833mtl0703_3

Lahtinen, E., Ala-Mutka, K., & Järvinen, H.-M. (2005). A study of the difficulties of novice programmers. *Acm Sigcse Bulletin*, 37(3), 14–18.

Leung, A., & Baccaglini-Frank, A. (2016). *Digital Technologies in Designing Mathematics Education Tasks: Potential and Pitfalls* (Vol. 8). Springer.

Marton, F. & Säljö, R. (1976). On qualitative differences in learning: Outcome and process. *British journal of educational psychology*, 46(1), 4-11.

Medeiros, R. P., Ramalho, G. L., & Falcão, T. P. (2018). A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education,* 62(2), 77–90.

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

Munthe, M. (2022a). Programmering i matematikklasserommet – Utfordringer elevene møter. *Acta Didactica Norden*, 16(4). https://doi.org/10.5617/adno.9173

Nilsen, T., Kaarstein, H., & Lehre, A. C. (2022). Trend analyses of TIMSS 2015 and 2019: School factors related to declining performance in mathematics. *Large-scale Assess Educ*, 10, 15. https://doi.org/10.1186/s40536-022-00134-8

Noss, R., & Hoyles, C. (1996*). Windows on mathematical meanings: Learning cultures and computers* (Vol. 17). Springer Science & Business Media.

NOU 2015:8 (2015). Fremtidens skole — Fornyelse av fag og kompetanser. Utredning fra et utvalg oppnevnt ved kongelig resolusjon 21. juni 2013. Avgitt til Kunnskapsdepartementet 15. juni 2015. Oslo: Kunnskapsdepartementet. Hentet fra http://www.regjeringen.no

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc.

Piteira, M., & Costa, C. (2013). Learning computer programming: Study of difficulties in learning programming. *In Proceedings of the 2013 International Conference on Information Systems and Design of Communication* (ss. 75–80). New York, NY, USA: ACM.

Postholm, M. B. & Jacobsen, D. I. (2018*). Forskningsmetode for masterstudenter i lærerutdanning.* Cappelen Damm.

Radford, L. (2010). Layers of generality and types of generalization in pattern activities. *PNA*, 4(2), 37- 62.

Rangnes, T. E., & Herheim, R. (2019). Lærers tilrettelegging for argument og agens. I T. E. Rangnes & R. Herheim (Red.), *Demokratisk danning i skolen: Tverrfaglige empiriske studier* (ss. 168-186). Oslo: Universitetsforlaget.

Sawyer, R. K. (2006). *The Cambridge handbook of the learning sciences.* Cambridge University Press.

Schwab, K. (2017). *The fourth industrial revolution*. Crown Business.

Schoenfeld, A. H. (2016). Dynamic assessment: A Bay Area approach. Hentet fra ERIC database (ED424933).

Schoenfeld, A. H. (2018). Video analyses for research and professional development: The teaching for robust understanding (TRU) framework. *Educational Psychologist*, 53(1), 31-45.

Skemp, R. R. (1976). Relational understanding and instrumental understanding. *Mathematics Teaching,* 77(1), 20–26.

Stacey, K. (1989). Finding and using patterns in linear generalising problems. *Educational Studies in Mathematics*, 20(2), 147-164. https://doi.org/10.1007/bf00579460

Stephens, M., & Kadijevich, D. M. (2020). Computational/algorithmic thinking. *In encyclopedia of mathematics education* (s. 117–123).

Tangen, R. (2010). "Beretninger om beskyttelse»; etiske dilemmaer i forskning med sårbare grupper - barn og ungdom. *Norsk Pedagogisk tidsskrift*, 94(4), 318-329.

Thagaard, T. (2018). Systematikk og innlevelse: *En innføring i kvaliatitve metoder* (5.utg.). Fagbokforlaget.

Usiskin, Z. (1988). Conceptions of school algebra and uses of variables. I B. Moses (Red.),

Algebraic Thinking, Grades K–12: Readings from NCTM's School-Based journals and other publications (1999) (s. 7-13). Reston, Va.: National Council of Teachers of Mathematics.

Utdanningsdirektoratet. (2018). Programmering i skolen - begrunnelser.

https://www.udir.no/laring-og-trivsel/prosjekter/programmering/programmering-i-skolen/begrunnelser/

Utdanningsdirektoratet. (2019a). Algoritmisk tenking i skolen - en veiledning for skolen.

https://www.udir.no/laring-og-trivsel/prosjekter/programmering/programmering-i-skolen/veiledning/algoritmisk-tenking-i-skolen-en-veiledning-for-skolen/

Utdanningsdirektoratet. (2019b). Kompetansepakker - programmering.

https://www.udir.no/laring-og-trivsel/prosjekter/programmering/programmering-i-skolen/kompetansepakker/

Utdanningsdirektoratet. (2020). Hva er nytt i matematikk? Elevene skal bli gode

problemløsere og forstå hvordan matematikk henger tett sammen med andre fag. https://www.udir.no/lk20/mat1-03/

Utdanningsdirektoratet. (2020c). Læreplan i matematikk (LK20). Hentet fra

https://www.udir.no/lk20/mat1-03/

Utdanningsdirektoratet. (2020a). Grunnleggende ferdigheter: Matematikk 1-10 (MAT01-05)

https://www.udir.no/lk20/mat01-05/om-faget/grunnleggende-ferdigheter?lang=nob

Utdanningsdirektoratet. (2020c). Læreplan i matematikk 1.-10. trinn (MAT01-05).

https://www.udir.no/lk20/mat01-05

Warren, E., & Cooper, T. (2008). Generalising the pattern rule for visual growth patterns:

Actions that support 8 year olds' thinking. *Educational Studies in mathematics*, 67, 171-185.

Watanabe, T. (2011). Shiki: A critical foundation for school algebra in Japanese elementary school mathematics. I J. Cai & E. Knuth (Red.), *Early Algebraization* (s. 109–124). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-17735-4_7

Winslow, L. E. (1996). Programming pedagogy—a psychological overview. *Acm Sigcse Bulletin*, 28(3), 17–22.

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

Wing, J. M. (2017). Computational thinking benefits society. *Social Issues in Computing*, 27-35.

Zazkis, R., & Liljedahl, P. (2002). Generalization of patterns: The tension between algebraic ‘thinking and algebraic notation. *Educational Studies in Mathematics*, 49(3), 379–402. https://doi.org/10.1023/A:1020291317178

8.0 Vedlegg

Vedlegg 1 – Samtykkeskjema

Vil du delta i forskningsprosjektet

Potensiale for dybdelæring gjennom Python-programmering i ungdomsskolen.

Formålet med prosjektet

Dette er et spørsmål til deg om du vil delta i et forskningsprosjekt hvor formålet er å

- undersøke hvordan Python-programmering kan virke inn på dybdelæringen i algebraundervisningen for elever i ungdomsskolen. Fokuset på teknologi i utdanningssystemet og betydningen av datakompetanse i samfunnet gjør at det er behov for å undersøke hvordan programmering kan være en del av matematikkundervisningen.
- Dette er en del av en masteroppgave som skal bidra til dypere forståelse av hvordan programmering kan føre til læring av algebra for elever på ungdomstrinnet. Studien er relevant for lærerstudenter, lærere og utdanningssektoren.

Hvorfor får du spørsmål om å delta?

Du får denne forespørselen fordi du er elev i 10.klasse og har matematikk, hvor noen av matematikkemnene skal inneholde programmering og algebra.

Hvem er ansvarlig for forskningsprosjektet?

Høgskolen i Østfold er ansvarlig for personopplysningene som behandles i prosjektet. I tillegg vil min veileder Toril Eskeland Rangnes og jeg vil ha tilgang til dataene som blir samlet inn.

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg.

Hva innebærer det for deg å delta?

For å samle inn data vil det bli tatt opp lyd og skjermopptak fra elevene sine Pc'er. Dette vil bli gjort i en undervisningssøkt på inntil to timer. Alle data som blir samlet inn i prosjektet vil bli slettet etter prosjektet avsluttes.

Kort om personvern

Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler personopplysningene konfidensielt og i samsvar med personvernregelverket. Du kan lese mer om personvern på neste side.

Med vennlig hilsen

Christer Stenersrød og veileder Toril Eskeland Rangnes

Utdypende om personvern – hvordan vi oppbevarer og bruker dine opplysninger
Det er kun min veileder Toril Eskeland Rangnes og jeg som vil ha tilgang til datainnsamlingen. Det vil bli erstattet med fiktive navn/koder for å anonymisere elevene, samt opptaksdataene vil bli lagret på et passord beskyttet lagringsenhet. Prosjektet vil også være sikret med en to faktorisering.

Hva gir oss rett til å behandle personopplysninger om deg?
Vi behandler opplysninger om deg basert på ditt samtykke.

På oppdrag fra Høgskolen i Østfold har personverntjenestene ved Sikt – Kunnskapssektorens tjenesteleverandør, vurdert at behandlingen av personopplysninger i dette prosjektet er i samsvar med personvernregelverket.

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

- å be om innsyn i hvilke opplysninger vi behandler om deg, og få utlevert en kopi av opplysningene
- å få rettet opplysninger om deg som er feil eller misvisende
- å få slettet personopplysninger om deg
- å sende klage til Datatilsynet om behandlingen av dine personopplysninger.

Vi vil gi deg en begrunnelse hvis vi mener at du ikke kan identifiseres, eller at rettighetene ikke kan utøves.

Hva skjer med personopplysningene dine når forskningsprosjektet avsluttes?
Prosjektet vil etter planen avsluttes 15. august 2024, og opplysningene vil da slettes.

Spørsmål
Hvis du har spørsmål eller vil utøve dine rettigheter, ta kontakt med:

- Christer Stenersrød på e-post: chrisrs@hiof.no eller på telefon: XX XX XX XX
- Toril Eskeland Rangnes på e-post toril.e.rangnes@hiof.no eller på telefon: XX XX XX XX
- Vårt personvernombud hos Høgskolen i Østfold, e-post: personvernombud@hiof.no eller på telefon: 40 28 15 58

Hvis du har spørsmål knyttet til Sikts vurdering av prosjektet, kan du ta kontakt på e-post: personverntjenester@sikt.no, eller på telefon: 73 98 40 40.

Jeg har mottatt og forstått informasjon om prosjektet potensiale for dybdelæring i Python-programmering i ungdomsskolen, og har fått anledning til å stille spørsmål. Jeg samtykker til:

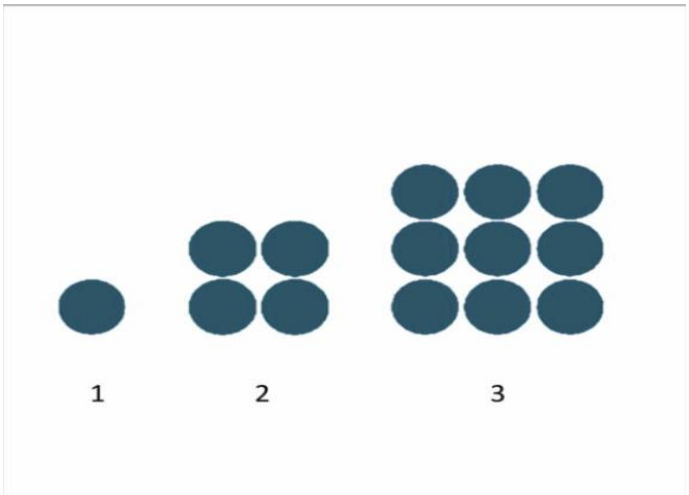
☐ å delta i observasjonsstudiet

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

---
(Signert av prosjektdeltakers foresatte)

Oppgaver om algebra med Python som hjelpemiddel

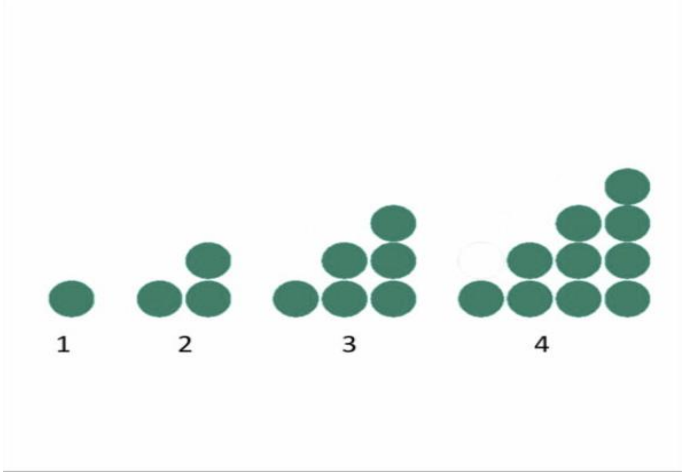
Kvadrattall



Figur 9 Kvadrattall, Lekolar Skole, Hentet fra https://lekolar-skole.no/figurtall/

5. Tegn kvadrattall nr. 4. Hvor mange prikker er det i kvadrattall nummer 4?
6. Diskuter i gruppen hvordan utvikler figuren seg?
7. Lag en generell formel for n?
8. Hva med nummer 7 eller 12?

Trekanttall



Figur 2 Trekanttall, Lekolar Skole, Hentet fra https://lekolar-skole.no/figurtall/

5. Tegn trekanttall nr. 5. Hvor mange prikker er det i kvadrattall nummer 5?
6. Diskuter i gruppen hvordan utvikler figuren seg?
7. Lag en generell formel for n?
8. Hva med nummer 9 eller 19?

Python – Programmering (MU – Editor)

Oppgave 1

Bruk den generelle formelen for kvadrattall og lag ett program som beregner hvor mange prikker det er i kvadratene.

Kan du finne ut hvor mange prikker det er i figur 15, 27 og 88?

Oppgave 2

Bruk den generelle formelen for trekanttall og lag et program som beregner hvor mange prikker det er i trekantene.

Kan du finne ut hvor mange prikker det er i figur 13, 39 og 51?

Oppgave 3

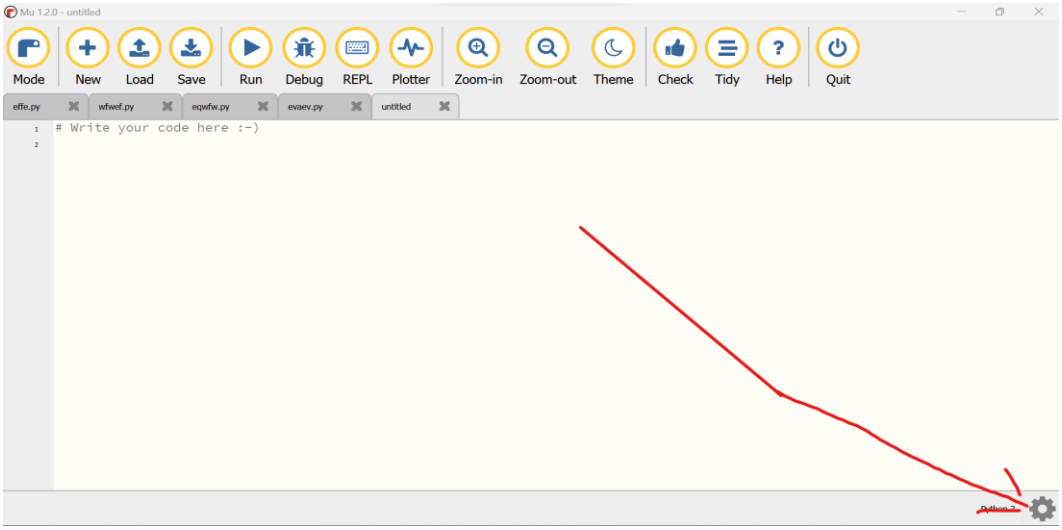
Hvor mange prikker er det i femkant tall?
Diskuter hvordan figurene ser ut og utvikler seg.
Kan du finne ut hvor mange prikker det er i figur 3, 5 og 9?

Oppgave 4

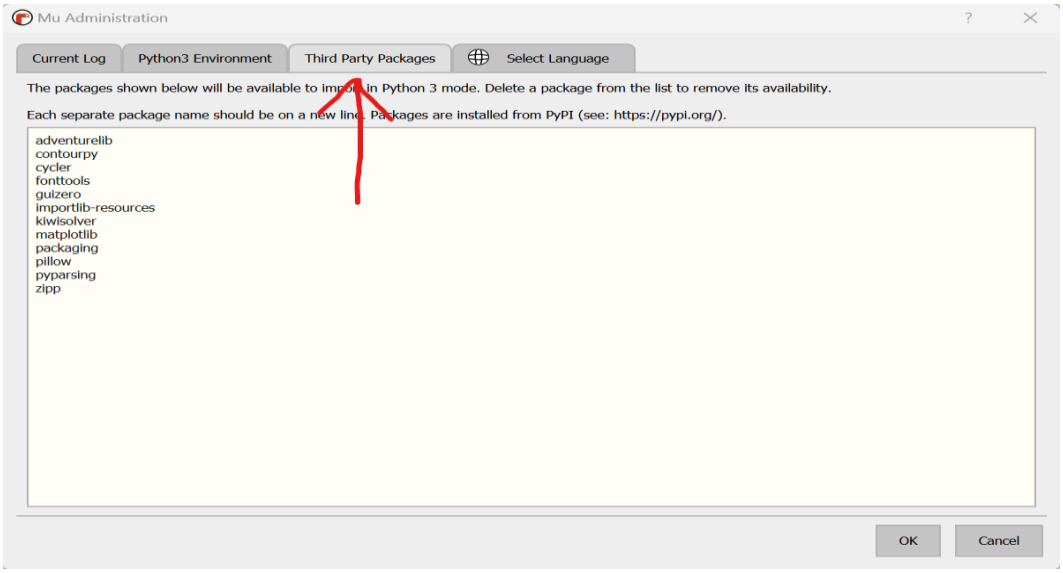
Se om du kan justere koden under ¨turtle¨ til å tegne *figur 7* både som trekant og kvadrattall?

```
import turtle

def tegn_punkt(x, y):
    turtle.penup()
    turtle.goto(x, y)
    turtle.pendown()
    turtle.begin_fill()
    turtle.circle(5)  # Juster størrelsen på punktene etter behov
    turtle.end_fill()

def tegn_trekanttall(n):
    punkter = 0
    x, y = -100, 100  # Startposisjonen for tegningen
    avstand = 30  # Avstand mellom punktene

    for i in range(1, n + 1):
        for j in range(i):
            tegn_punkt(x + j * avstand, y)
            punkter += 1
        y -= avstand

    turtle.done()

# Tegn et trekanttall, for eksempel trekanttall for 5
tegn_trekanttall(5)
```

Hvis dere ikke får koden til å fungere, må dere antakelig legge til Third Party Package. Dette gjøre ved å trykke på tannhjulet ned til høyre.



Så får dere opp dette vinduet her.



Her skriver dere inn: turtle og trykker så ¨ok¨, så vil den installere turtle tegneprogrammet automatisk. Så er det bare å teste koden.

Syntakser:

def antall_prikker_figur(n):

return

print

Vedlegg 3 – Utdrag transkripsjon

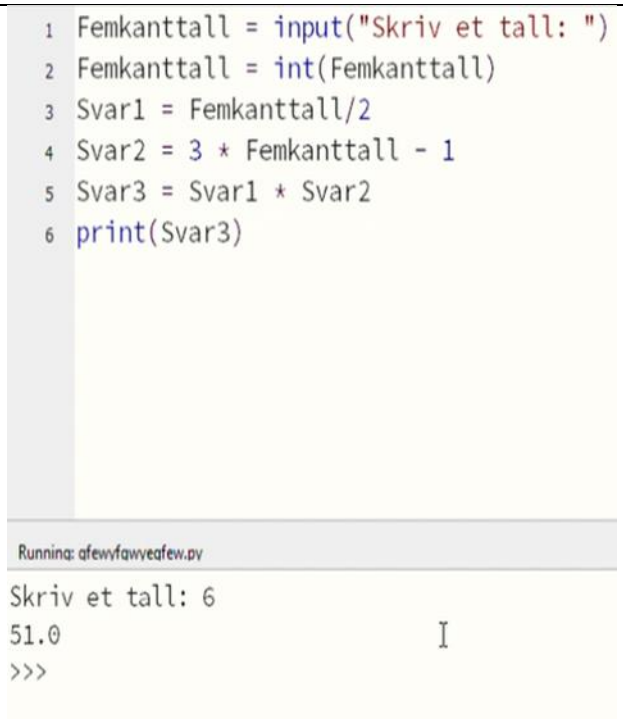
Utdrag 1:

A: Jeg fant den
 	
B: en halv N, jeg var inn på det da
 	
A: Ja, vil du vite hvor det sto, eller? Hvis vi går tilbake på den siden du var.
 	
B: faen a, det sto der ja, en halv N. hmm..
 	
A: sjekk om den formelen kan stemme?
 	
B: Det er basicly det jeg har skrevet, bare at det aaa…., ja ehm. Det funka litt, men ikke helt.
 	
Ok, halvparten av.. okei, funker på første, okei funker der å, neste, funker der å. ehm 12, funker der å, skal vi prøve på 4 å, 2 ehm ja det funker.
 	
A: ja, da sto formelen rett der du har hatt pc'en de siste 20minuttene
 	
B: Jeg lagde noe som ligna da
 	
A: Hva var det du lagde da?
	Koden for trekanttall
	`Trekanttall = input("skriv et tall:")`
	`Trekanttall = int(Trekanttall)`
	`Svar1 = Trekanttall * Svar1`
	`Svar2 = Svar1 + 1`
	`Svar3 = Svar1 * Svar2`
	`print(Svar3)`
	
	Løsning for 5
	`Skriv et tall5`
	`15.0`
	`>>>`
	
	Løsning for 19
	`Skriv et tall19`
	`190.0`
	`>>>`
	
	Løsning for 13
	`Skriv et tall13`
	`91.0`
	`>>>`
	
	Løsning for 51
	`Skriv et tall51`
	`1326.0`
	`>>>`

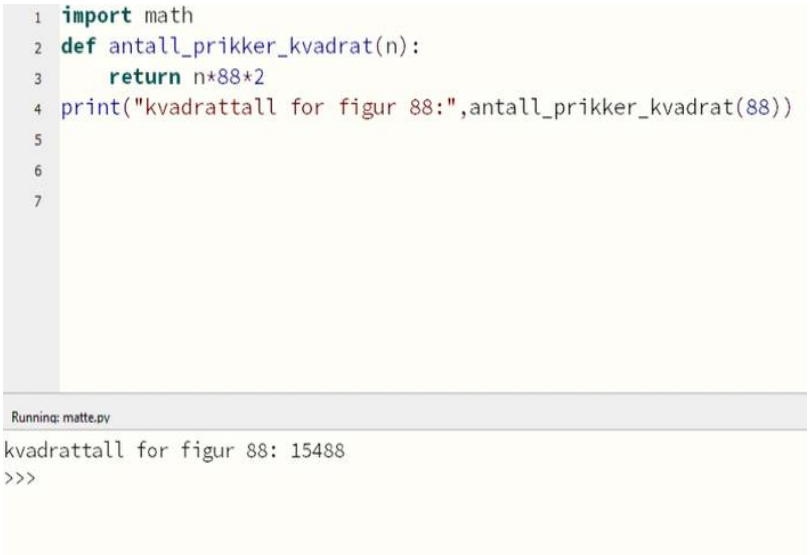
B: Var ikke helt der, men det funka på noen av greiene	

Utdrag 2

A: vi har fasiten jo
 	
B: men er det ikke morsommere å gjøre det sjøl?
 	
A: Ja, vi har gjort det på to helt forskjellige måter uansett
 	
B: er lik femkanttall delt på 2, også femkanttall.
 	
Svar 1 ganger er lik 3 ganger femkanttall, minus 1
 	
Sånn
 	
Hvis vi prøver på ett av de vi vet da
 	
4
 	
22.
 	
Ja, funker
 	
A: 5
	Kode for femkanttall
	`Femkanttall = input("Skriv et tall: ")`
	`Femkanttall = int(Femkanttall)`
	`Svar1 = Femkanttall/2(3 * Femkanttall - 1)`
	
	`print(Svar)`
	
	Løsning for 5
	`Skriv et tall: 5`
	`35.0`
	`>>>`

35 B: Ja, vi fikk det til Liten klapp på slutten.	

Utdrag 3

C: Skal vi sjekke nå om hvis jeg gjør sånn her. Bare har den sånn. D: Se om det kommer noe der. D: Da ble det noe helt annet. D: Nei. E: Det er sikkert det som er riktig. D: Nei, det var det ikke. D: Nei, hallo, vi gidder ikke. E: Nei, nei, nei, nei, nei. D: Vi gidder ikke å gjøre det på nytt. D: Vi skulle slå oss til.	Hvor de tidlig møter syntax hindringer, og ikke vet helt hva de gjør galt. 

C: Da har vi gjort det feil,

C: da har vi gjort det feil.

D: Sånn er det vel.

E: Jeg vet ikke,

D: vi har lært av feilen vår.

C: Det er sånn her vi skal gjøre det på trekant.

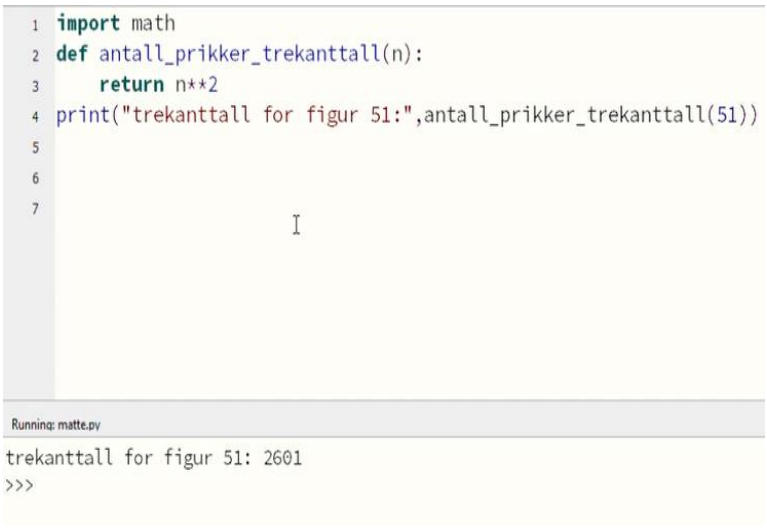
D: Hva var det jeg sa?

D: Vi skal bruke den formelen.

E: Åja.

C: Haha.

Her har de oppdaget feilen og endre koden slik at den fungerer.



Vedlegg 4-

Løsningsforslag datainnsamlingsoppgaver

Oppgave 1: Kvadrattall

```python
def antall_prikker_kvadrat(n):
    return n**2

# Test for figur 15, 27 og 88
print("Kvadrattall for figur 15:", antall_prikker_kvadrat(15))
print("Kvadrattall for figur 27:", antall_prikker_kvadrat(27))
print("Kvadrattall for figur 88:", antall_prikker_kvadrat(88))
```

Oppgave 2: Trekanttall

```python
def antall_prikker_trekant(n):
    return n*(n + 1)//2

# Test for figur 13, 39 og 51
print("Trekanttall for figur 13:", antall_prikker_trekant(13))
print("Trekanttall for figur 39:", antall_prikker_trekant(39))
print("Trekanttall for figur 51:", antall_prikker_trekant(51))
```

Oppgave 3: Femkanttall

```python
def antall_prikker_femkant(n):
    return (3*n**2 - n)//2

# Test for figur 3, 5 og 9
print("Femkanttall for figur 3:", antall_prikker_femkant(3))
print("Femkanttall for figur 5:", antall_prikker_femkant(5))
print("Femkanttall for figur 9:", antall_prikker_femkant(9))
```