Programmering i naturfag

Et observasjonsstudium av læreres undervisning

Tor Eivind Folden

Masteroppgave
Lektorprogrammet
30 studiepoeng

Institutt for lærerutdanning og skoleforskning
Det utdanningsvitenskapelige fakultet

UNIVERSITETET I OSLO

Juni, 2023

Programmering i naturfag

- *Et observasjonsstudium av læreres undervisning*

2023

Programmering i naturfag Tor Eivind Folden

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

Trykk: Reprosentralen, Universitetet i Oslo

Sammendrag

Hensikten med denne oppgaven har vært å finne ut av hvordan programmering i naturfagundervisningen kan foregå og hvilke nye utfordringer som skapes i møte med disse fagfeltene. Jeg har filmet to VG1-klasser gjennomføre et opplegg om populasjonsutvikling hos moskuser og bakterier i en dobbeltime med programmering i naturfag, og deretter analysert interaksjonene mellom deltagerne. I tillegg er det gjennomført intervjuer med lærerne både før og etter undervisningen for å få et bedre bilde av hva lærerne har tenkt underveis i opplegget. Jeg har også brukt en anonymisert undersøkelse lærerne gjennomførte i timene.

Filmene er transkribert og analysert etter TPACK-modellen og en modell for algoritmisk tenkning. En interaksjonsanalyse er brukt for å komme mer i dybden på datamaterialet. TPACK-modellen handler om hvilke deler av undervisningen som er knyttet til programmering, pedagogikk og naturfaglige elementer og modellen for algoritmisk tenkning er brukt for å identifisere hvilke TPACK-elementer som kan knyttes til algoritmisk tenkning. Interaksjonsanalysen er brukt for å få innblikk i hvordan elevene interagerer med hverandre.

Viktige funn er blant annet at bare en femtedel av timen koder for det naturfaglige elementet, men at denne kan økes ved bruk av refleksjonsoppgaver i plenumsundervisning, og antagelig blir større når elevene kan mer programmering. Dette er særlig viktig fordi det er det naturfaglige elementet som er nærmest knyttet til algoritmisk tenkning, og det er derfor en veldig viktig del av undervisningen.
Analysene viser også at det er en konflikt mellom hvordan elevene ser på kopiering av tekst innenfor programmering og i andre fag og at temaet variabelkontroll er et viktig element for både programmering og i naturfag generelt og derfor kan bli trukket fram i større grad. Til slutt kan det trekkes fram at selv om ingen av lærerne hadde undervist med programmering i naturfag og hadde blandede forventninger ble begge lærerne fornøyde med undervisningen, fordi elevene fikk jobbet effektivt og utnyttet tiden godt.

Siden dette er et kvalitativt studium vil det ikke være representativt for alle klasser, men vil gi et innblikk i utfordringer knyttet til programmering i naturfagundervisningen.

Forord

Noe av bakgrunnen for at jeg skriver denne masteroppgaven er fordi vi har lært noe om programmering i kjemi og biologi gjennom studiet på lektorprogrammet, men vi har fått veldig begrenset erfaring med hvordan vi faktisk underviser dette i klasserommet. Ettersom jeg er interessert i programmering ble dette en god måte å få lært meg selv mer om temaet og forhåpentligvis dermed være bedre rustet til å jobbe med programmering som lærer.

Masteroppgaven hadde likevel ikke blitt til uten hjelp fra folk rundt meg. Først og fremst vil jeg takke Erik Knain, hovedveilederen min, for at han har vært tydelig på områder hvor jeg har trengt støtte og på det akademiske, men samtidig har gitt meg frihet til å utforske det jeg selv har syntes var mest interessant uten for sterke føringer. Jeg vil også takke Nani Teig som har vært biveilederen min, blant annet fordi hun har pusha meg når jeg trengte det og har vært veldig behjelpelig med å finne referanser til oppgava. Jeg er veldig takknemlig for at dere begge har vært veilederne mine og ikke minst for at dere har vært så interesserte i temaet, selv om ingen av dere egentlig har programmering som fagfelt.

Jeg vil også takke «Ida», «Ola» og lærerne på skolen jeg gjennomførte prosjektet på. Takk for at dere hadde troa på prosjektet og fordi dere turte å si ja til å delta, selv om dere selv var i oppstartsfasen. Det har vært inspirerende å samarbeide med dere og jeg har følt meg veldig velkommen.

På universitetet vil jeg takke IT-folka på Nils Henrik Abels hus for at dere har vært tålmodige med meg, og aller mest for at dere ikke har fått meg til å føle meg dum når jeg spør om hjelp - dette er inspirerende for en kommende lærer. Jeg vil også takke CCSE – Center for Computing in Science Education for flere nyttige seminarer over det siste året. Disse har vært en inngangsport for meg for å forstå problemstillinger og muligheter tilknyttet programmering i naturfagundervisningen og har vært en del av prosessen min med å skrive denne oppgava.

Sist, men ikke minst vil jeg takke familie og venner for middager, turer og samtaler om alt annet enn programmering, både under skrivingen av masteroppgaven, men også tidligere under pandemien. Det har vært noen utfordrende år og da ser man viktigheten av å ha folk rundt seg.

Innholdsfortegnelse

1 Innledning

Hensikten med denne masteroppgaven er å gi et innblikk både i hvordan programmering konkret kan undervises i naturfagundervisningen i VG1, hvilke utfordringer som oppstår i møtet med programmering, men også hvilke implikasjoner dette har for undervisning med programmering. I oppgaven vil det også komme fram hvordan lærerne reflekterer over rollen til programmering i naturfag.

Jeg har valgt problemstillingen «hvilke fagdidaktiske utfordringer kan identifiseres i gjennomføringen av et undervisningsopplegg med programmering i naturfag vg1 og hvilke implikasjoner har dette for undervisning?».

For å løse denne problemstillingen har jeg beskrevet et undervisningsopplegg som ble gjennomført i to klasser med naturfag i vg1. Undervisningen fra dette opplegget ble filmet med fokus på interaksjoner mellom læreren og elevene og senere analysert for å beskrive hva som skjedde. Det ble holdt intervjuer med lærerne både før og etter undervisningen for å få fram hvordan lærerne har tenkt om undervisningsopplegget og hvordan de reflekterer over undervisningen deres.

Undervisningen ble analysert med TPACK-modellen for å skille mellom ulike deler av undervisningen og deretter koblet sammen med en modell for algoritmisk tenkning. Deler av undervisningen er nærmere beskrevet og analysert med en interaksjonsanalyse. I tillegg er data fra en spørreundersøkelse lærerne brukte i timen brukt for å utfylle dataene fra de andre metodene.

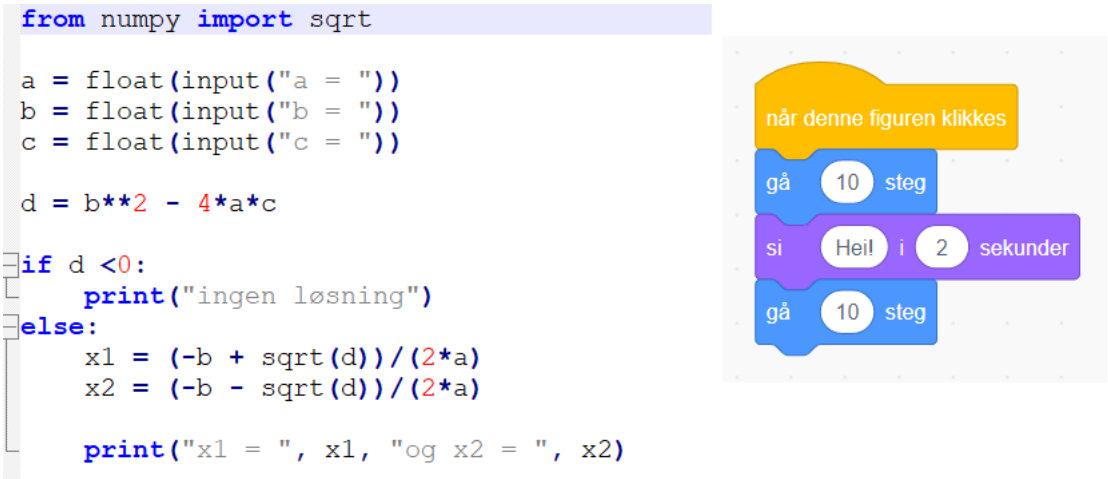
Dette er en viktig problemstilling fordi programmering ble innført i naturfaget gjennom Fagfornyelsen i mye større grad enn tidligere, men det er få studier som beskriver hvordan programmering benyttes i naturfagundervisningen og hvilke utfordringer som oppstår ved å bruke det, i hvert fall ikke i norsk kontekst. Norge har innført programmeringsopplæring som en del av matematikkundervisningen hvor det skal brukes i andre fag. Dette gjør at naturfagundervisningen blir avhengig av hvordan programmering opplæres i matematikkundervisningen, men også hvordan det legges opp til at elevene faktisk skal benytte programmeringen i naturfagundervisningen. I mange andre land er programmering et eget fag i seg selv og dette medfører at det vil bli undervist litt annerledes fordi det er de programmeringsfaglige hensynene som står i fokus, ikke de matematikkfaglige hensynene.

Dette vil gjøre at man jobber med andre typer oppgaver og i større grad vektlegger elementer som ikke nødvendigvis er like nyttige rent mattefaglig.

Å integrere programmeringen i matematikkundervisningen vil i større grad koble sammen matematikken og programmeringen og det er rimelig at vi kan får andre typer konflikter mellom fagene og det vil derfor ikke være helt sammenlignbart. Dette vil begrense hvor stor sammenligningsverdi andre studier har til den norske skolen.

Det er noen få studier på hvordan programmering benyttes innenfor spesifikke realfag som fysikk, men også her er det forskjeller, for eksempel fordi fysikk og biologi er valgfag på videregående skole, mens naturfag er et fellesfag som skal passe for alle elevene. Programmering er ikke like integrert i kjemifagene og disse er derfor ikke like relevante å sammenligne med. Selv om det er få studier som allerede er gjennomført har OsloMet et prosjekt kalt MASCOT, hvor de vil undersøke programmering i undervisningen nærmere. Dette vil være nyttig å følge mer med på for å få et bedre innblikk.

Tilsvarende vil det også være enkelte studier på lavere trinn i naturfag, men disse er heller ikke helt sammenlignbare, fordi programmeringen i norsk skole er gjerne delt inn i blokkbasert programmering på de lavere trinnene, og tekstbasert programmering innføres gjerne i varierende grad fra skole til skole mellom ungdomstrinnet og videregående (Haraldsrud et al., 2020).



Figur 1: Eksempel på tekstbasert og blokkbasert koding

Figur 1 viser til venstre et eksempel på tekstbasert koding i Python benyttet i videregående skole og til høyre er et eksempel på blokkbasert koding i Scratch.

En vesentlig forskjell mellom disse typene programmeringsspråk er mengden feil man får. For eksempel er det veldig vanlig å få ulike feilmeldinger i Python fordi man for eksempel har skrevet noe feil, eller at man har brukt en funksjon på en måte som datamaskinen ikke forstår. Dette vil ikke skje like ofte i blokkbaserte programmer fordi man ikke har like mange muligheter til å skrive nye ting, og de ulike variablene og funksjonene har ulike former som gjør at de bare passer på noen få måter. På den andre siden gjør dette også at man kan gjøre mye mer med et tekstbasert programmeringsspråk, siden dette vil gjøre det lettere å bruke mange forskjellige kommandoer og ikke bare kommandoer som er laget på forhånd.

Programmering i naturfag vg1 er derfor ikke helt sammenlignbart med alle andre fag, selv om det også vil være likheter og noe overføringsverdi.

Som nevnt tidligere er det i fagfornyelsen lagt opp til at programmering skal læres i matematikk, dvs. blant annet matematikk 1P eller P-matte og i matematikk 1T, eller T-matte. Programmering er nevnt konkret i et kompetansemål i T-matte, og i begge læreplanene er det nevnt konkret under digitale ferdigheter under grunnleggende ferdigheter (Utdanningsdirektoratet, 2019a) (Utdanningsdirektoratet, 2019b). Programmering skal dermed være integrert i begge disse fagene, men man kan tenke at det vil vektlegges noe tyngre i T-matte.

Noe av min personlige motivasjon for å velge nettopp dette temaet er fordi jeg gjennom lektorprogrammet har lært meg å programmere, men ikke selv undervist med det, men i mange tilfeller er jeg usikker på hvor mye det har tilført undervisningen sammenlignet med om vi hadde gått for en enklere løsning ved å bruke for eksempel Excel eller lignende. I tillegg har vi ikke gjennom studiet lært så mye om hvordan vi skal undervise programmering, eller programmeringsdidaktikk i naturfag om man vil, og for å få bedre kompetanse på dette feltet vil jeg lære mer om det. I tillegg til dette har jeg også en egeninteresse for programmering og synes det er viktig å forstå teknologien rundt oss, spesielt i lys av utviklingen med kunstig intelligens og hvilke konsekvenser dette kan ha for samfunnet vi er en del av.

Et spørsmål jeg har stilt meg selv i arbeidet med masteroppgaven handler om hvordan undervisningen kan benytte programmering for å gi større utbytte av naturfagundervisningen, heller enn at undervisningen bare handler om at programmering skal læres som en verdi i seg selv. Dette kommer jeg nærmere inn på i diskusjonsdelen av oppgaven, men det er begrenset

hvor mye en enkel kvalitativ masteroppgave kan svare på dette og dette vil derfor være et område for mer forskning.

1.1 Gjennomføring av prosjektet

I denne oppgaven har jeg undersøkt hvordan man underviser med programmering i naturfag, med utgangspunkt i to klasser i VG1, studiespesialiserende fra den samme skolen i Osloområdet i skoleåret 2022/2023. Skolen ble valgt ut fordi det er en samarbeidsskole med Universitetet i Oslo. Jeg har valgt å kalle lærerne jeg har samarbeidet med «Ida» og «Ola». Både Ida og Ola har erfaring med programmering, og må kunne beskrives som over gjennomsnittlig kompetente i programmering.

I prosessen med å rekruttere lærere tok jeg kontakt med mange av samarbeidsskolene til universitetet. Jeg fikk få svar og en skole forklarte at jeg ikke kunne observere fordi de fremdeles var i stadiet hvor lærerne fikk opplæring i programmering selv, og derfor ikke hadde kommet i gang med å undervise programmering. Jeg anser det som sannsynlig at dette også gjelder flere av skolene hvor jeg ikke fikk svar. Skolen hvor jeg gjennomførte prosjektet hadde heller ikke kommet i gang med undervisning av programmering i naturfag, men de hadde jobbet med programmering i matematikkundervisningen og var villige til å prøve en oppstart med programmering i naturfag. Elevene hadde altså ikke erfaring med programmering innenfor naturfag, men hadde rundt 7 klassetimer erfaring med programmering i matteundervisningen. Dette gjaldt både elevene med p- og t-matte. Selv om skolen var i startgropa når det gjelder programmering, er det greit å presisere at lærerne antagelig hadde mer kompetanse enn det som er vanlig i skolen.

Prosjektet besto av et førintervju med lærerne før undervisningsøkta, observasjon med videokamera hvor læreren hadde en «mygg»-mikrofon, og intervju i etterkant av undervisningen. I etterkant har jeg også fått anonymiserte data fra en evaluering elevene gjennomførte i timen. Jeg har også kontaktet Ola i etterkant av intervjuet for klargjøring av en av situasjonene i transkripsjonen.

Videofilmen ble analysert ut fra TPACK-modellen, som handler om at undervisning i programmering krever en kombinasjon av teknisk kunnskap, pedagogisk kunnskap og faglig kunnskap, i mitt tilfelle naturfaglig og matematisk kunnskap. I analysen har jeg også brukt et rammeverk for algoritmisk tenkning for å identifisere sekvenser som bygger algoritmisk

tenkning for elevene. Jeg har også analysert noen enkeltsekvenser med en interaksjonsanalyse. Dette er en tredelt sekvens hvor elevene får hjelp til et problem på ulike tidspunkter gjennom skoletimen til Ola, og to episoder hvor jeg har lagt mer vekt på elev-elev-interaksjoner, for å få mer innblikk i hva elevene forstår og ikke forstår.

Jeg har brukt de to intervjuene før undervisningen for å ha et innblikk i hvilke forventninger lærerne hadde før de startet undervisningsøkta, og jeg har brukt intervjuet i etterkant av undervisningsøkta for å vite hvordan lærerne reflekterte underveis og hvorfor de gjorde som de gjorde, men også for å sammenligne erfaringene lærerne fikk i øktene de underviste og reflektere mer generelt rundt temaet programmering i naturfagundervisningen. Jeg har også sett etter om det er samsvar mellom det lærerne sier de gjør og det de faktisk gjør, som en triangulering av dataene.

Jeg har ikke lagt stor vekt på egenevalueringen elevene gjennomførte, fordi den er selvrapportert og ettersom lærerne vet hvem som har svart hva kan elevene i noen grad ha blitt påvirket av dette. Jeg har likevel sammenlignet dataene med annet datamateriale i noen tilfeller for å se om det er samsvar mellom det inntrykket jeg har fra andre kilder og det elevene selv sier.

Lærerne valgte selv ut et opplegg fra Aschehoug univers om populasjonsdynamikk hos moskusokser (Aschehoug univers, 2023). Dette opplegget er beskrevet nærmere i oppgaven under punkt 3.2. Under punkt 3.3. er det en beskrivelse av de ulike klassene som deltok i studiet. Dette er gjort for å gi kontekst til klassemiljøet, men også for å i best mulig grad beskrive hvordan den enkelte læreren har tenkt rundt undervisningsopplegget.

2 Teori

2.1 Algoritmisk tenkning

Algoritmisk tenkning (Engelsk: Computational thinking) er et sentralt begrep innenfor programmering, med mange definisjoner. Begrepet er definert som «tankeprosessen involvert i å formulere problemer og løsninger, slik at løsningene er i en form hvor de effektivt kan bli gjennomført av en informasjonssøkende agent» (Wing, 2011, min oversettelse). Begrepet «informasjonssøkende agent» vil i denne oppgaven bli vurdert som en datamaskin, ettersom det er denne informasjonssøkende agenten man benytter i praksis. I denne definisjonen er algoritmisk tenkning tett knyttet til programmering.

Utdanningsdirektoratet bruker derimot en annen definisjon:

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

Jeg vil i denne oppgaven holde meg til Utdanningsdirektoratets definisjon. Algoritmisk tenkning handler altså om hvordan man kan formulere problemer og løsninger, slik at for eksempel en datamaskin vil kunne finne en løsning på disse. I Utdanningsdirektoratets definisjon skal ikke alle problemer løses gjennom programmering, men kunnskap om programmering og algoritmisk tenkning er likevel viktig i vurderingen av hvorvidt et problem kan løses effektivt ved hjelp av programmering eller om det bør løses på andre måter og vil derfor også falle innenfor definisjonen av den digitale, grunnleggende ferdigheten, ved at elevene «i økende grad skal benytte selvstendighet og dømmekraft i bruk av digitale verktøy» (Utdanningsdirektoratet, 2019c).

2.2 Programmering og algoritmisk tenkning i læreplanen

I den nye læreplanen i naturfag er programmering nevnt på forskjellige måter. For alle årstrinn er programmering og modellering nevnt eksplisitt som en del av den grunnleggende

ferdigheten digitale ferdigheter. I tillegg er programmering nevnt eksplisitt i ulike kompetansemål. For VG1 studieforberedende er kompetansemålet «eleven skal kunne vurdere og lage programmer som modellerer naturfaglige fenomener» nevnt eksplisitt. For 10. trinn skal elevene «kunne bruke programmering til å utforske naturfaglige fenomener», i tillegg til å «utforske, forstå og lage teknologiske systemer som består av en sender og en mottaker». Også i 7. trinn skal elevene kunne «utforske, lage og programmere teknologiske systemer som består av deler som virker sammen» (Utdanningsdirektoratet, 2019c). Dette viser at det skal benyttes programmering i naturfag. Ettersom dette er nytt, er det ikke dokumentert mye erfaring med programmering i naturfagundervisning i norsk kontekst. I mange andre land er programmering innlemmet som et eget fag i motsetning til i norsk kontekst hvor opplæring av programmering er satt til matematikkundervisningen for deretter å benyttes i andre fag, og situasjonen i andre land er derfor ikke uten videre sammenlignbar med norske forhold. For eksempel vil temaer som cybersikkerhet, databaser og hvordan nettlesere fungerer være mye mindre relevant i matematikkundervisningen enn i et eget programmeringsfag. Dette er temaer som er representerte i programmeringspensum i for eksempel England og deler av den engelsk-språklige verden (Falkner et al., 2019).

Utover kompetansemål i læreplanen er algoritmisk tenkning nevnt i Kelentrić et al. sitt rammeverk for lærerens profesjonsfaglige digitale kompetanse (PfDK) for Utdanningsdirektoratet. Her er det nevnt at læreren skal kunne forstå grunnleggende prinsipper for algoritmisk tenkning og hvilke konsekvenser det har i samfunnet, men også kunne bruke digital teknologi for å skape rammene for elevers utvikling av bl.a. algoritmisk tenkning. Programmering er her ikke nevnt spesifikt, men vil være en del av den digitale teknologien som blir brukt. (Kelentrić et al., 2017)

Selv om algoritmisk tenkning ikke er nevnt eksplisitt i læreplanen er det likevel helt sentralt når man jobber med programmering, fordi det kreves for at man skal kunne forstå hvordan datamaskiner «tenker» og for at man skal kunne lage et program som datamaskinen forstår. Det vil ikke være mulig å bruke programmering uten at de lærer en viss grad av algoritmisk tenkning, og for at elevene skal bli bedre til å programmere og å vurdere programmer vil derfor algoritmisk tenkning være helt nødvendig.

2.3 Programmeringens rolle i naturfag

Programmering i naturfag er nytt, og det finnes derfor ulike perspektiver på i hvilken grad programmering er nødvendig for at elevene skal kunne lære naturfag. En del naturfag trenger man ikke å kunne programmere for å lære i det hele tatt, og både prosessen med å lære programmering og prosessen med å bruke programmering kan ta en del tid. Programmering er likevel relevant for naturfag av flere grunner, både som et nyttig verktøy man kan bruke til å gjøre ting man ikke har kunnet tidligere, men også fordi verden vi lever i stadig blir mer og mer digital og det vil bli vanskelig å forstå verden uten å også ha en viss forståelse av algoritmisk tenkning og programmering. Dette gjelder også dersom man vil følge ny naturvitenskapelig forskning. På denne måten er programmeringens rolle i skolen knyttet til nytteargumentet og demokratiargumentet presentert i Sjøberg (2009) for hvorfor vi skal ha naturfag i skolen i utgangspunktet. Nytteargumentet går ut på at det for elevene er nyttig å kunne naturfag fordi de får bruk for det de lærer også utenfor skolen, mens demokratiargumentet går ut på at elevene trenger kunnskap om naturfag for å kunne forstå og diskutere naturfaglige problemstillinger som er viktige på samfunnsplan. Dette kan for eksempel være tilknyttet genteknologi eller global oppvarming.

En av de viktigste grunnene til å ha programmering i skolen er nytteargumentet. Programmering er et verktøy som gjør det mulig å utforske naturfaglige fenomener på andre måter enn tidligere, blant annet fordi man kan unngå avanserte matematiske metoder ved for eksempel å jobbe numerisk. Programmering gir også nye perspektiver på oppgaver man også tidligere har kunnet gjennomføre, som kan illustrere nye sider ved gamle problemer (Haraldsrud et al., 2020). Programmering kan dermed ha en rolle ved at det er en metode for å visualisere og utforske naturfaglige fenomener som tidligere har vært vanskelige å utforske. Dette kan være fordi fenomenet ikke er tilgjengelig i klasserommet, som for eksempel hvis man vil sammenligne det å kaste en ball på jorda og på månen med fysikkens lover, eller fordi fenomenet krever mye datamateriale for å utforske og det er ikke praktisk gjennomførbart å gjøre dette med analoge verktøy. Dette kan for eksempel være å sammenligne karaktertrekk innad og mellom arter ved å plotte disse i ulike dimensjoner, eller å analysere hvordan en populasjon med bakterier utvikler seg over tid.

Nytteargumentet er viktig også fordi man bruker programmering i forskning innenfor naturfag, og en viss kjennskap til programmering gjør det lettere å forestille seg hvordan forskere har jobbet for å finne ut av ulike naturfaglige fenomener, i tillegg til at det vil være

nyttig for fremtidige forskere å ha en grunnleggende forståelse av verktøy de vil måtte bruke i fremtiden.

Demokratiargumentet er relevant fordi en forståelse av programmering og algoritmisk tenkning er viktig for å forstå samfunnet vi lever i. ChatGPT og kunstig intelligens er trukket fram som nye utfordringer og verktøy for skolen, men for at elevene skal kunne forstå hvordan disse fungerer og vurdere informasjonen disse gir trenger elevene en viss forståelse for hva kunstig intelligens er, hva algoritmer er og hvordan datamaskiner fungerer. Et konkret eksempel på kunstig intelligens som brukes aktivt i skolen er appen «Artsorakel» som bruker bildegjenkjenning for å foreslå hvilke planter, dyr eller andre organismer som er avbildet på bilder. Denne appen brukes flere steder enten som et tillegg til floraer, eller i stedet for floraer, for at elevene skal få kjennskap til artsmangfoldet rundt dem. Dette er en kunstig intelligens som har øvd seg på å kjenne igjen en art basert på bilder den har fått tilgang til, men for å få en større forståelse av hvordan appen fungerer er det for eksempel relevant å vite at den har blitt opplært med bilder av norske, ville arter og derfor ikke vil kunne gjenkjenne utenlandske arter eller hageplanter i like stor grad, og man vil oftere få dårligere treff på artene hvis de er med i det hele tatt. Et annet eksempel på det samme er når elevene bruker kunstig intelligens for å få svar på andre naturfaglige problemstillinger. Elevene trenger ikke å kunne lage kunstig intelligens selv, men de må kunne forholde seg kritisk til den, og til dette trenger de verktøy som algoritmisk tenkning.

Som vist i figur 1 i innledningen kan programmering være blokkbasert eller tekstbasert. Blokkbasert programmering består av at man setter sammen blokker med mer eller mindre ferdig kode sammen og lager et program, mens tekstbasert programmering krever at elevene skriver inn kommandoer o.l. selv. Dette har styrker og begrensninger og gjør blant annet at elevene i større grad får syntaks-feil, men det gir flere muligheter for hvilke program man vil lage og det kan bli lettere å redigere programmet når man først kan det. Gjeldende praksis er at barneskolen i stor grad bruker blokkbasert programmering som Scratch og at videregående i stor grad bruker tekstbasert programmering som Python. I ungdomsskolene vil bruken variere i større grad fra skole til skole og det vil skje en overgang fra blokkbasert programmering til tekstbasert. Ettersom elevene lærer programmering gjennom matematikkundervisningen vil det være naturlig at man i naturfagundervisningen bruker det samme programmeringsspråket som er brukt i matematikkundervisningen som utgangspunkt.

Programmering kan gi noen fordeler, men har også egne utfordringer. En av de største utfordringene er at det er et nytt tema som de fleste elevene ikke har drevet med tidligere. Dette er både fordi det i langt større grad enn tidligere krever at elevene forstår algoritmisk tenkning, men også fordi det er store forskjeller i hvor mye opplæring både elever og lærere har i programmering. Gjennom arbeidet med masteroppgaven min har jeg allerede funnet at det er skoler i Osloområdet som ikke hadde begynt med programmering høsten 2022, mens andre var kommet litt i gang. Forskjeller i programmeringskompetanse i ungdomsskolen, både individuelt mellom elever og mellom skoler vil føre til store forskjeller i kompetansen til elevene i videregående skole de nærmeste årene. Dette vil også kunne påvirke på tilsvarende måte i starten av ungdomsskolen. Dette gjør at læreren må differensiere i større grad enn ellers i naturfag, fordi det kan være både elever som driver med programmering på fritida og elever som aldri har vært borti programmering før i samme klasse. Det kan også være en utfordring for læreren at de flinkeste elevene kan mer enn læreren (Haraldsrud et al., 2020).

2.4 Dybdelæring

Dybdelæring er et av de viktige begrepene som ble introdusert med Fagfornyelsen og det ble kuttet i antall kompetansemål blant annet for at man skal kunne fordype seg i fagstoffet i deler av fagstoffet i større grad enn før. Dybdelæring kan også knyttes til opplæringen av programmering. Man kan tenke at elevene bare skal kunne bruke programmering og ikke forstå hvordan datamaskinen tenker, men dette vil gjøre det vanskelig å faktisk benytte programmering ettersom det er et verktøy, ikke bare kunnskap. Samtidig er poenget med programmeringen i naturfag ikke programmeringen i seg selv, men at de skal kunne bruke programmering for å gjennomføre noe innen naturfag. Det er derfor viktig at de lærer programmeringen som er relevant for naturfag og at man begrenser fokuset på selve programmeringen. Dette er relevant for kompetansemålet om blant annet å vurdere programmer, for man kan vurdere programmer både med hensyn på naturfaglige aspekter og med hensyn på programmeringsfaglige aspekter. Som et eksempel er det en forskjell på spørsmålet «gir dette programmet svar på det naturfaglige spørsmålet jeg stiller?» og «jobber dette programmet effektivt eller bør man skrive det på en annen måte, slik at det fungerer bedre?».

Ut ifra læreplanen i naturfag vil det være naturlig at man vurderer i første rekke med hensyn på det naturfaglige, og at det programmeringsfaglige først blir relevant dersom programmet

ikke fungerer eller bruker så lang tid at det ikke bruker for lang tid. Dette virker også som et godt prinsipp med tanke på dybdelæring. Elevene må kunne programmering i den grad at de kan forstå, bruke og endre programmer for naturfagets del i tillegg til å lage enkle programmer selv, men de trenger ikke en dybdeforståelse for hvorfor datamaskinen kan løse oppgaver mye mer effektivt på noen måter enn andre og pedagogiske hensyn som gjør programmer mer forståelige vil kunne være mer viktig enn at ting gjøres på den mest effektive måten rent programmeringsfaglig.

Dette kan underbygges med nytteargumentet og demokratiargumentet fra tidligere. Programmering er relevant for naturfag kun så lenge den er nyttig ut fra nytteargumentet og/eller demokratiargumentet. Dersom programmeringen ikke lenger er viktig for det naturfaglige vil det ikke være fornuftig å bruke mye tid på dybdelæring i det fra et naturfaglig perspektiv.

2.5 Utforskende arbeidsmåter i naturfag og støttestrukturer i undervisningen

Bruk av utforskende arbeidsmåter har hatt en sterk tradisjon i naturfag i Norge, og dette brukes i mange ulike former som for eksempel tradisjonelle eksperimenter i laboratoriet og å finne kilder til en argumenterende tekst og mye annet. Arbeidsmåtene kjennetegnes ved at de kobler sammen tekst og empiri og ved at elevene får større muligheter til å lage egne problemstillinger, får større rom for å velge metoder selv og/eller ved at elevene må velge mellom ulike svar. Denne friheten gjør at elevene i større grad kan finne ut av noe de selv synes er meningsfylt og stimulerer til kreativitet blant annet ettersom det ikke nødvendigvis finnes et fasitsvar. I tillegg vil dette gi innsikt i naturfagets egenart og naturfaglig metode (Knain & Kolstø, 2019).

En utfordring med utforskende arbeidsmåter er at elever som ikke er flinke til å strukturere sitt eget arbeid vil få problemer. For å løse dette brukes andre former for rammer og støttestrukturer for at elevene skal få strukturert arbeidet sitt. Knain, Bjønness og Kolstø (2019) definerer at rammer angir området det skal arbeides i og omfatter et tema, mens støttestrukturer er definert som redskaper elevene får tilgjengelig for å ta seg fram gjennom rammen slik at arbeidet får god kvalitet. Støttestrukturer er også omtalt andre steder som «stillas» og er også relevant i mer tradisjonelle læringssituasjoner.

I utforskende arbeidsmetoder vil læreren få en rolle mer som veileder enn tradisjonell lærer, fordi elevene har forskjellige løp og forskjellig progresjon. Læreren kan også få en rolle å motivere elevene, gi råd til elevene eller gjennomføre underveisvurdering av elevene. Undervisningen vil gjerne veksle mellom perioder med struktur og at elevene får spillerom, slik at elevene får veiledning, men samtidig får frihet til å løse oppgaven slik de ønsker. På denne måten veksler undervisningen mellom å være elevstyrt når elevene jobber selv og at undervisningen er lærerstyrt. De lærerstyrte periodene kan være både i små grupper og i helklasse (Bjønness, Johansen, & Byhring, 2019).

2.6 Hvordan undervise programmering

Programmering vil spille en rolle i naturfagundervisningen, men hvordan lærer man bort programmering i naturfag?

Det er vist av Kwon et al. (2021) at et introduksjonskurs med blokkbasert koding kan bidra til å tette gapet mellom de som har god kontroll på algoritmisk tenkning og de som ikke har det. I en systematisk gjennomgang av flere studier av Ogegbo & Ramnarain (2022) vises det til at det brukes flere metoder og at man kan bruke programmering til både problemløsning og for å lage modeller av naturfaglige temaer. Undervisningen er gjerne utforskende og konseptet algoritmisk tenkning er ofte sentralt, selv om dette er et begrep med mange ulike definisjoner. Fordeler knyttet til bruken av algoritmisk tenkning er blant annet at elevene får en bedre forståelse for hvordan forskning foregår og hvordan man kan designe programmer for å løse problemstillinger.

En kjent praksis i programmeringssammenheng er live-koding, eller en prosess hvor en person koder i plenum, gjerne med innspill fra andre deltagere. Prosessen «programmeringsdiktat» beskrevet av Haraldsrud et al. (2020) minner noe om dette. Selv om live-programmering er en anbefalt metode for å undervise programmering mangler det empirisk data som underbygger dette (Selvaraj et al., 2021). Samtidig viser forskning at når flere samarbeider om programmer blir resultatet mer presise programmer og samarbeid kan være viktig for at elevene skal kunne hjelpe hverandre og få muligheter til å reflektere over programmene (Haraldsrud et al., 2020).

I oppgaven min har jeg tatt utgangspunkt i TPACK-modellen, for som nevnt i punkt 2.4 påvirker både naturfaglige hensyn, programmeringsfaglige hensyn og pedagogiske hensyn

hvordan man underviser programmering i naturfag. Dette er fordi hensikten med å undervise om programmering i naturfag er at det tilfører noe til naturfag, uavhengig av hvilken verdi det å lære programmering har i seg selv. Alle disse hensynene påvirker undervisningen og en analyse bør derfor omfatte alle disse hensynene for å se hvordan de påvirker hverandre. Denne påvirkningen kan være både positiv og negativ. På ulike områder kan de ulike hensynene ha synergier med hverandre eller være i konflikt med hverandre. På noen områder vil prinsipper man lærer i programmering også være nyttig for det naturfaglige, men i andre situasjoner vil dette være negativt.

Dersom en læringssituasjon knyttet til programmering også har en kobling til naturfag vil dette kunne bli et «teachable moment» i naturfag eller en potensiell læringssituasjon (min oversettelse) hvor refleksjon vil kunne gi eleven(e) mulighet til å lære noe nytt om naturfaget og (i denne situasjonen) trekke koblinger til programmering. Disse potensielle læringssituasjonene er nærmere beskrevet av Haug (2014) og for at man som lærer skal kunne utnytte situasjonene er det viktig å planlegge for dette i forkant, ellers risikerer man at den potensielle læringssituasjonen går tapt.

I motsatt fall kan det være at det oppstår konflikter mellom naturfaget og programmeringen. Dette kan forårsake feiloppfatninger eller dårlige læringsstrategier og for å unngå dette er det viktig at læreren vet hva slags konflikter som kan oppstå, slik at man enten kan unngå at de oppstår, eller slik at man kan tydeliggjøre hvorfor ting gjøres annerledes i de ulike fagtradisjonene. Feiloppfatninger innenfor naturfagene er beskrevet mange steder, blant annet i kjemikontekst av Ringnes og Hannisdal (2014).

For å se hvordan dimensjonene henger sammen har jeg i denne oppgaven lagt en del vekt på en metode som sammenligner disse hensynene eller kunnskapene kalt TPACK-modellen, i samspill med andre metoder. Mer om de konkrete metodene står i metodekapittelet.

2.7 Validitet og reliabilitet

Validitet og reliabilitet er sentrale begreper for å vurdere kvalitet av empiriske forsøk. Validitet handler om at man måler det man faktisk sier at man skal måle, mens reliabilitet handler om at dersom analysen ble gjort flere ganger ville man kommet fram til samme resultat. En åpenbar utfordring med tanke på reliabilitet er at dette er et kvantitativt studium med utgangspunkt i to klasser og det vil derfor ikke være representativt for skolene i Norge. I

tillegg er videofiler og lydfiler fra intervjuene slettet av hensyn til deltagernes personvern, og man vil derfor ikke kunne gjennomføre hele prosessen på nytt.

Ved bruk av kvalitative metoder, som i denne oppgaven, er det flere faktorer som bygger opp under en studies validitet, reliabilitet og troverdighet som beskrevet av Patton (1999) og Creswell & Miller (2000). Blant annet er refleksivitet, triangulering og fyldige beskrivelser viktige elementer. Refleksivitet vil si at det skal være så enkelt som mulig å forstå hvordan jeg har gjennomført studiet og hvilken relasjon jeg har hatt til deltagerne i studiet (Gleiss, 2021). For å være så transparent som mulig legger denne oppgaven derfor en del vekt på metodedelen og analysedelen. I tillegg er det viktig å påpeke at jeg ikke har hatt noen personlig tilknytning til andre deltagere i studiet. Dette er påpekt under punktet om forskningsetikk. Triangulering vil for eksempel si at en enten bruker flere metoder for å observere et fenomen, at man bruker ulike datakilder for å beskrive et fenomen eller at det er flere mennesker som er med å analyserer dataene (Patton, 1999). I denne oppgaven er det hovedsakelig metodetriangulering og datatriangulering som er benyttet, ved at observasjonsdataene er analysert med flere metoder som utfyller hverandre og ved at intervjudataene utfyller observasjonsdataene. I tillegg ble noen enkeltsekvenser av transkripsjonen analysert sammen med en veileder med tanke på TPACK-modellen. Jeg har forsøkt å gjengi fyldige detaljer fra datamaterialet, både gjennom eksempler i analysen med TPACK-modellen, med deler av transkripsjonen i interaksjonsanalysen og ved å gjengi enkelte beskrivende sitater fra intervjuet med lærerne. I tillegg er selve undervisningsopplegget beskrevet, og store deler av det er tilgjengelig digitalt fra Aschehoug univers med lisens.

3 Metode

3.1 Forskningsetikk

Prosjektet er gjennomført som beskrevet i søknad godkjent av NSD, med noen merknader:

I Olas klasse var det en elev som kom inn i klasserommet uten å ha gitt samtykke. Videoen ble stoppet før hen kom med i skjermbildet og videoen ble satt på pause. Dette medførte noen minutters pause i filmingen.

I starten av opplegget i Idas klasse var det en elev som trakk samtykket sitt. Delene av videoen hvor denne eleven deltok enten i bildet eller med lyd ble slettet i løpet av to dager og alle videofiler ble lagret trygt etter Universitetet i Oslos regler for oppbevaring av fortrolige data i denne perioden (Universitetet i Oslo, 2022).

I resten av perioden er filmene jobbet med på fortrolig eller rødt nivå, da de inneholder noe sensitivt materiale. Dette er i tråd med søknaden godkjent av NSD.

Jeg vil også påpeke at jeg ikke har noen tilknytning til Aschehoug Univers eller Aschehoug, og kjente heller ingen av elevene eller lærerne fra før.

3.2 Metoder for datainnsamling

3.2.1 Filmopptak

Som metode for datainnsamling var video et lett valg fordi det er mye som skjer i et klasserom og med en ganske åpen problemstilling var det helt nødvendig å kunne se nærmere gjennom det som skjedde i timen flere ganger og transkribere i sakte tempo. En problemstilling med videofilming er at man kun får med seg det som skjer innenfor skjermen og man er derfor avhengig av at man har filmet de rette stedene, slik at filmen blir representativ for klasserommet (Blikstad-Balas, 2017). Ettersom jeg ville fokusere på interaksjoner mellom læreren og elevene var ikke dette et stort problem. Læreren snakker stort sett med elevene rett ved siden av seg med mindre det er plenumsundervisning og det er derfor lett å få med de aller fleste elevene rundt læreren. I plenumsundervisningen er det

derimot noen steder jeg ikke har klart å identifisere elevene ettersom stemmen kommer fra et sted utenfor skjermen.

Jeg valgte å bruke et kamera på stativ med mikrofon, i tillegg til at læreren fikk en myggmikrofon. Dette gjorde at jeg fikk med god lyd på læreren og de rett ved siden av læreren og det er hovedsakelig denne lyden som har vært nyttig i transkripsjonen. I tillegg har jeg fått med litt av lyden som var nærmest kameraet, som har vært litt nyttig i plenumsundervisningen for å få med meg det elevene sier. Fokuset på lyden rundt læreren har begrenset mengden av elev-elev-interaksjoner i utvalget og fremtidige studier som ønsker å fokusere på elev-elev-interaksjoner bør vurdere å ha flere mikrofoner i klasserommet, for eksempel på hvert gruppebord.

For å få et bilde av det andre som skjedde i klasserommet og få med meg elev-elevinteraksjoner har jeg også av og til filmet det andre som skjedde i klasserommet. Dette gjelder blant annet situasjonen beskrevet i figur 15 i resultatdelen av oppgaven. En utfordring med dette er nettopp det at jeg kun hadde to mikrofoner i klasserommet og dersom elevene ikke satt i nærheten av læreren eller veldig nær kameraet fikk jeg ikke med tilstrekkelig lyd til å kunne forstå hva som skjedde. Elever oppfører seg av og til annerledes om læreren er i nærheten enn hvis læreren er et helt annet sted og dette vil påvirke interaksjonene.

Med filming er det også alltid et spørsmål om de som filmes påvirkes av selve filmingen, dette kalles refleksivitet. I starten på timen til Ida var det for eksempel en elev som byttet plass for å få en annen vinkel til kameraet og var tydelig påvirket i starten av timen. Det var også en elev som lagde grimase til kameraet. Etter introduksjonen av timen var ikke denne effekten merkbar lenger og det var flere situasjoner i begge klasserommene hvor elevene åpenbart ikke brydde seg om kameraet. Dette stemmer overens med forskning på bruk av video i forskning – effekten er størst når kameraet kommer inn i klasserommet, men blir raskt ubetydelig (Blikstad-Balas, 2017). Lærerne var antagelig mer bevisste på kameraet siden de hadde myggmikrofoner og hadde snakka mer med meg i forkant, men de oppførte seg normalt og jeg tror ikke denne effekten har vært spesielt stor. Det har hatt noe påvirkning blant annet fordi Ida mistet mikrofonen en gang og dette førte til at et 15-sekunderssegment i undervisningen handlet om at Ida hadde en mikrofon i stedet for faglig innhold.

3.2.2 Intervju

I tillegg til videoen valgte jeg å benytte intervju som metode for å støtte videofilmen. Dette er hovedsakelig fordi filmen ikke direkte kan vise hvorfor lærerne gjorde som de gjorde eller hvordan de har tenkt, men dette kan komme tydelig fram av et intervju, og når en del av problemstillingen min i tillegg er å få fram hvordan lærerne reflekterer rundt opplegget var intervju et godt valg. Det er ikke alle som er enige i at intervju kan få fram hvordan det faktisk ble tenkt i en gitt situasjon (Svenkerud, 2021), og for å få et best mulig bilde av hvordan lærerne faktisk har tenkt delte jeg intervjuet i et førintervju og et hovedintervju, i tillegg til å sammenligne det lærerne sier med observasjonsdata. Førintervjuet var et kort intervju på 5-10 minutter hvor jeg snakka med lærerne individuelt før undervisningen deres om hvilke forventninger lærerne hadde til elevene og til opplegget. Av praktiske årsaker fikk jeg ikke gjennomført intervjuet med Ola muntlig og dette ble gjennomført skriftlig i stedet. Hovedintervjuet var et intervju med både Ola og Ida sammen på ca. 45 minutter om hvilken erfaring elevene deres hadde med programmering, hvordan lærerne opplevde undervisningsøkta de selv underviste, men også mer om hvordan de reflekterte rundt rollen til programmering i naturfagundervisningen. Jeg hadde i utgangspunktet egentlig tenkt å holde intervjuet individuelt med hver enkelt lærer, men ettersom lærerne hadde en god dynamikk til hverandre og samarbeidet godt valgte jeg med samtykke fra lærerne å ha et felles intervju i stedet. Lærerne ble oppfordret til å påpeke det dersom de var uenige med hverandre, siden en risiko med gruppeintervju er at uenigheter ikke kommer fram. Siden Ola og Ida hadde undervist ulike klasser var det noen forskjeller i hvordan undervisningsøktene hadde vært og disse forskjellene kom fram i intervjuet. Inntrykket mitt er at intervjuet fungerte godt og at dynamikken mellom lærerne gjorde at de bygget på svarene til hverandre og jeg fikk et rikere intervju enn om jeg hadde gjennomført dem individuelt. Intervjuene ble gjort med taleopptak.

Intervjuet hadde primært en støttende rolle i datainnsamlingen og en del av datamaterialet er benyttet blant annet for å beskrive klassene og konteksten rundt undervisningen, men i tillegg til dette har jeg et punkt knyttet til lærernes refleksjon rundt egen undervisning. Intervjuet er først og fremst brukt i diskusjonen der det enten støtter eller motsier andre deler av datamaterialet, gjerne fordi det belyser hvordan lærerne har tenkt og dermed trianguleres observasjonsdataene.

3.3 Analysemetoder

For å tolke dataene har jeg brukt flere ulike metoder for å få fram ulike aspekter. Jeg har brukt en TPACK-modell for å gi en oversikt over undervisningen utenom innledningen og avslutningen. TPACK-modellen har fordelene at den er godt kjent innen fagfeltet og at den er anvendbar og kan brukes på mange forskjellige måter (Pamuk et al., 2015). Noen svakheter med TPACK er at det kan være vanskelig å skille mellom noen av kategoriene og at den kanskje kan være litt for generell. For å unngå disse svakhetene har jeg fokusert på skillet mellom de ulike typene kunnskap som pedagogisk kunnskap (PK) og teknisk kunnskap (TK) og sekundært sett på kombinasjonene av disse som for eksempel programmeringsdidaktisk kunnskap (TPK). De ulike kunnskapene og kombinasjonene er beskrevet under kapittel 3.3.1. For at ikke modellen skal bli for generell og for at jeg skal kunne koble modellen til begrepet algoritmisk tenkning har jeg også brukt et CT-rammeverk. CT (Computational thinking) er engelsk for algoritmisk tenkning. Her har det vært en utfordring at timene som ble observert er helt i oppstarten av programmeringsundervisningen i naturfag og fokuset for timen var mer å lære å bruke verktøyet programmering heller enn å bruke det til å jobbe med algoritmisk tenkning i størst mulig grad.

I tillegg til disse metodene har jeg benyttet et interaksjonsstudium for å gå mer i dybden på enkelte situasjoner i undervisningen. Disse situasjonene er valgt ut for å gi et rikere bilde av hvordan undervisningssituasjonene faktisk foregikk, men også fordi dette var situasjoner jeg mener kan ha implikasjoner for hvordan man bør undervise programmering i naturfagundervisningen. I et tilfelle har jeg fulgt opp de samme elevene over hele undervisningsøkta for å se elevenes utvikling over en kort tidsperiode. I de fleste andre utdragene består situasjonene av elev-elev-interaksjoner for å få et annet innblikk i hvilken forståelse elevene har når læreren ikke deltar like aktivt i samtalen. Jeg har også prøvd å få fram eksempler på hvilke forestillinger elevene har som kan krasje med undervisningen med programmering og hvilke potensielle læringssituasjoner som kan oppstå i undervisningen.

Til slutt har jeg en selvevaluering elevene fylte ut som jeg fikk tilsendt av lærerne som holdt undervisningen. Jeg kommer ikke til å fokusere på denne i oppgaven, men jeg har sett på hvordan denne undersøkelsen støtter eller motsier dataene fra de andre metodene.

3.3.1 TPACK-modellen

I denne oppgaven har jeg benyttet meg av modellen TPACK. TPACK står for teknisk kunnskap (TK), pedagogisk kunnskap (PK) og «content» kunnskap (CK) og er en teori som

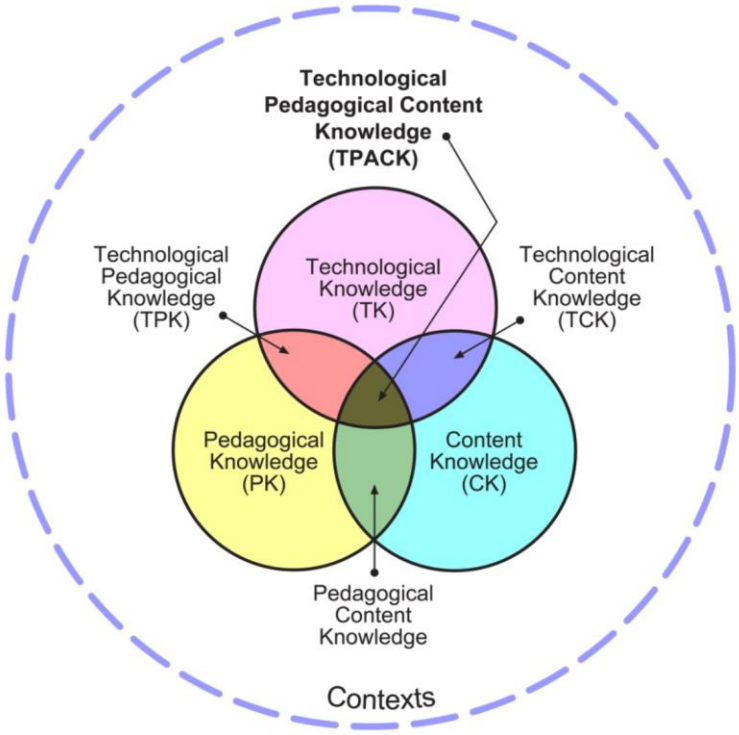
handler om at disse formene for kunnskap må kombineres for å undervise med tekniske hjelpemidler (Koehler & Mishra, 2009).

«Content» kunnskap, CK, er kunnskap som handler om selve temaet for undervisningen, men også metoder som er brukt for å komme fram til denne kunnskapen og vil i denne oppgaven primært handle om naturfaglig kunnskap (NK) og matematisk kunnskap (MK). Naturfaglig kunnskap er kunnskap knyttet til begreper i naturfag, fagstoff som vekst i naturfag og vil også inkludere temaer tilknyttet den naturvitenskaplige metode i seg selv. Matematisk kunnskap er blant annet kunnskap tilknyttet det å lese av en graf, og det som handler om det rent matematiske. Dette er viktig for å forstå temaet og er dermed en del av CK, men denne delen er likevel såpass annerledes fra begrepskunnskapen i naturfag og den naturvitenskapelige metode at jeg vil forsøke å skille mellom disse.

Pedagogisk kunnskap, PK, er lærerens pedagogiske kunnskap og handler om hvilken praksis og metoder læreren bruker i undervisningen. Dette inkluderer både overordnede strukturer, som hvilke verdier og mål som settes i undervisningen og hva læreren fokuserer på, men også mer konkrete ting som hvordan læreren svarer på spørsmålene til elevene og hvordan læreren prøver å forstå hva elevene kan. PK omfatter også hvilke grep læreren tar for at hele klassen som helhet skal ha læringsutbytte og tilpasninger som gjøres underveis, enten kollektive eller individuelle. Det er vanskelig å se for seg undervisning uten et element av pedagogisk kunnskap, men det vil likevel være mulig å se om enkelte handlinger fokuserer på PK eller ikke.

Teknisk kunnskap, TK, er litt vanskelig å definere, men vil i denne oppgaven handle om programmering generelt og mer spesifikke utfordringer tilknyttet Python som er programmeringsspråket som brukes. TK vil også omfatte det å vite hvordan datamaskinen fungerer, og hvordan denne tolker koden som blir laget. TK vil også omfatte det å tolke feilmeldinger. Teknisk kunnskap kan også omfatte analoge verktøy, men dette vil ikke bli lagt vekt på i denne oppgaven.

Som figur 2 nedenfor av Koehler & Mishra (2009) antyder, vil det være områder hvor disse kunnskapene overlapper.



Figur 2: Illustrasjon av TPACK-modellen (Koehler & Mishra, 2009)

Teknisk «content» knowledge, TCK, handler om hvordan teknologi og temaet påvirker og begrenser hverandre. Et typisk eksempel i denne sammenhengen på dette er at teknologi i stor grad gjør simulasjoner mulig, og man kan bruke dette for å utforske et naturfaglig fenomen. Samtidig vil tekniske begrensninger i hvordan programmet er bygd opp og hvordan datamaskiner fungerer gjøre at deler av simulasjonen ikke vil fungere likt som i virkeligheten, for eksempel fordi man har glemt å ta med luftmotstand i koden, eller fordi det er små regnefeil som vanligvis ikke spiller noen rolle, som likevel påvirker i noen tilfeller.

Teknisk pedagogisk kunnskap, TPK, handler om hvordan teknologi og pedagogikk påvirker og begrenser hverandre. Disse kan i noen tilfeller være i konflikt med hverandre. For eksempel er det vanlig praksis i programmeringsmiljøer å importere kodepakken numpy ved å bruke kommandoen

```
import numpy as np
```

og deretter bruke «np.» som et prefiks foran kommandoene som importeres (f.eks. np.linspace). I pedagogisk sammenheng, hvor man vil forenkle koden for elevene er det vanlig å i stedet bruke kommandoen

```
from numpy import *
```

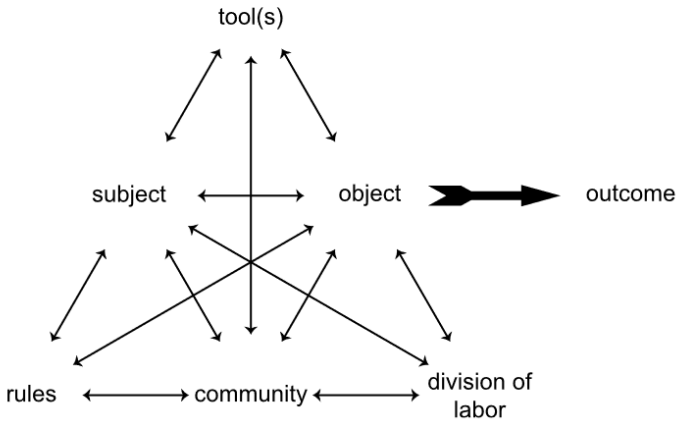
slik at man slipper å bruke prefikset, og dermed forenkler koden for elevene (f.eks. linspace). Dette har noen begrensninger på koden, men er likevel vurdert positivt av pedagogiske hensyn (Haraldsrud et al., 2020).
TPK kan også handle om for eksempel å bruke metaforer for å forklare elevene hvordan kode fungerer.

Pedagogisk «content» kunnskap, PCK, handler om hvordan temaet og pedagogikk henger sammen. Eksempler på dette fra timen kan være å finne analogier når begrepet «bæreevne» skal forklares av læreren. Det kan også handle om å sette kunnskapen i kontekst av annen kunnskap eleven har. Et eksempel på dette i timen var når læreren knyttet eksponentiell vekst fra grafen som ble laget til noe de hadde gjort i mattetimen.

Teorien om TPACK har fått noe kritikk for at det kan være vanskelig å definere hva som er forskjellen mellom for eksempel PCK og PK, og det er noe glidende overganger. Jeg har derfor tatt utgangspunkt i begrepene TK, CK og PK og heller brukt TPK, TCK og PCK for å utfylle disse, for å få mer tydelig definerte områder.

3.3.2 CT-rammeverket

I oppgaven kommer jeg også til å bruke et rammeverk for algoritmisk tenkning i undervisningen. Dette rammeverket bygger på aktivitetsteori, som igjen bygger på en sosiokonstruktivistisk forståelse av læring og handler om at læring foregår i et sosialt miljø hvor elevene påvirker og påvirkes av hverandre, læreren, konkrete og immaterielle verktøy som f.eks. en simulering, både formelle regler og normer for klasserommet og undervisningen og hvordan gruppene er organisert (Hurt et al., 2023). Hvordan disse variablene henger sammen med hverandre er vist i modellen i figur 3.



Figur 3: Aktivitetssystem som viser hvordan ulike variabler påvirker undervisningen.

I CT-rammeverket er algoritmisk tenkning tenkt som et produkt av undervisningen i aktivitetssystemet. Når jeg bruker denne modellen i klassene jeg observerte blir dette litt misvisende, fordi hensikten med undervisningsøktene i utgangspunktet var å bli kjent med verktøyet i undervisningen – altså programmering i Python. Målet med timen var dermed ikke det samme og dette vil påvirke hvor mye algoritmisk tenkning vi kan forvente å observere, som vist av modellen.

Aktivitetsteori er et nyttig redskap for å forstå bedre hva som skjer i en sosial undervisningskontekst, og jeg har brukt disse kategoriene for å vurdere de ulike klassene i forsøket.

Tabell 1: Oversikt over kategorier i CT-modellen

CT-S		Cognitive Processes		
		Reflective Use	Design	Evaluation
		of a computational tool for		
Science Activity	Data Collection	How can I use the time-lapse photography feature on my phone to collect data on how fast my plants grow?	How could I change my robot's code so that it would randomly sample the quadrats for my ecological study?	If this pseudocode for getting the temperature sensor readings was running on my computer, under which conditions would it work well, and when might it fail?
	Data Processing	Which features in the spreadsheet software can help me identify the relationship that exists between force, mass, and acceleration?	I have images that show the sun's height above the horizon from every day of the year from a given location taken at solar noon; what would I need software to do to help me find patterns related to seasonal changes?	I am using software to create a data visualization of how the air quality in my city changed due to forest fires; what are the software's affordances and limitations for helping me communicate those data?
	Modeling	How can I use an orrery to predict the next transit of Venus?	What rules would I need to include in a plate tectonics simulation to accurately model different types of plate boundaries?	Which aspects of this digital model accurately reflect natural selection and which do not?
	Problem-Solving	I want to share my experimental procedure for measuring the speed of sound with a lab partner who is blind; how can I use their screen-reading software to make sure the procedure will be communicated correctly?	My classmates gave different arguments about climate change; how could I create an algorithm to evaluate the claims and reasoning of their respective arguments?	I need to test a phone app that was designed to call an emergency contact when it detects a car crash; how should I test it and how will I know I have sufficiently tested it?

Rammeverket vist i tabell 1 deler CT inn i 12 forskjellige bokser. Tabellen er delt inn i tre kolonner. En kolonne handler om reflekterende bruk av programmering, en kolonne handler om design i programmering og en kolonne handler om evaluerende bruk av programmering. Reflekterende bruk av programmering vil være hvordan man overordnet planlegger et program, slik at det løser et problem. Design i programmering vil typisk handle om hvordan man endrer konkret kode slik at det løser et problem og evaluerende bruk av programmering vil handle om at man har et program og skal vurdere hvor godt det virker.

Disse kolonnene møter 4 forskjellige rekker som beskriver ulike typer naturvitenskapelige aktiviteter. Her kan det enten handle om hvordan data samles inn, hvordan dataene prosesseres, hvordan et program modellerer et fenomen eller hvordan man kan løse et nytt problem. Kolonnene og radene krysser hverandre, og vi får til sammen $4 * 3 = 12$ kategorier for CT (se tabell 1).

Hensikten min med å benytte dette rammeverket er både at jeg får sett hvor stor del av undervisningen som handler om algoritmisk tenkning, men også at jeg ville prøve å koble

modellen til TPACK-modellen og dermed se om det var noen kategorier i TPACK-modellen som bidrar mer til algoritmisk tenkning.

3.3.3 Interaksjonsanalyse

Interaksjonsanalyser er en type analyse som legger vekt på tale og samhandlinger med objekter og artefakter mellom flere parter, for å tolke sosiale handlinger. Interaksjonsanalyser kombinerer små utdrag fra samtalen med etnografisk data for å tolke sosiale handlinger. Hver ytring er tolket i konteksten av den pågående samtalen som foregår og det som tidligere er sagt og analysene kan også brukes over et lengre tidsperspektiv, for eksempel over flere interaksjoner i en klassetime (Knain et al., 2017). Ettersom man må kunne observere detaljer i interaksjonen mellom partene egner interaksjonsanalyser seg godt i videoanalyser.

Det er ulike fremgangsmåter knyttet til hvordan analysen utføres, men det er vanlig at man først grovtranskriberer data for eksempel fra et gruppearbeid og deretter velger seg ut potensielt interessante videosekvenser for finere transkripsjon. Interaksjonsanalyser er generelt induktive, det vil si at man lager kategorier etter som man transkriberer heller enn å ta utgangspunkt i allerede definerte kategorier. Etter å ha fintranskribert og tolket situasjonene, ser man gjerne gjennom videoklippene flere ganger for å se om det man har tolket faktisk har røtter i virkeligheten, og man sammenligner situasjoner fra ulike deler av videofilen for å se om man kan finne sammenhenger på tvers av dem. Dette gjøres ofte i grupper for å redusere bias (Jordan & Henderson, 1995).

Transkripsjonen gjennomføres gjerne iterativt, det vil si at man starter med en grovere transkripsjon og legger til flere detaljer ettersom man finner ut hva som er viktig å få med i transkripsjonen av et bestemt klipp. Dette gjør at man får med de detaljene som er viktige for en enkelt studie, men man er kanskje i større grad avhengig av å gjennomføre transkripsjonen selv, ettersom en annen forsker vil vektlegge andre detaljer i transkripsjonen. Det vil alltid være viktig å få med tale, men avhengig av den enkelte studien vil ulike variabler være viktige å få med.

3.3.4 Spørreundersøkelse

Spørreundersøkelsen som er benyttet er en selvevaluering av elevenes innsats i løpet av timen, med en kombinasjon av kvantitative data med skalaer fra 1 til 10 og tekstsvar med

tilbakemeldinger til læreren. Spørreundersøkelsen var primært en del av lærernes metode for å få tilbakemeldinger fra elevene ettersom det var den første økten med programmering i naturfag for lærerne og de var derfor interesserte i å få tilbakemeldinger. I etterkant har lærerne anonymisert dataene og sendt dem til meg som utfyllende data.

En av styrkene til spørreundersøkelser er at man får data fra mange respondenter på ganske kort tid. Dette gjør at man kan benytte dette i undervisningen uten at det går for mye på bekostning av tid til undervisning. I timene ble alle elevene i klassen spurt og de aller fleste har levert svar til læreren, og den vil derfor også være representativ for klassen (Frønes & Pettersen, 2021).

Utfordringer med spørreundersøkelser er blant annet at man som regel ikke får fulgt opp svarene man får. Dette gjelder ikke denne undersøkelsen, ettersom den ikke opprinnelig var anonym og lærerne derfor kan følge opp med tilleggsspørsmål dersom de ønsker det. At undersøkelsen ikke var anonym vil på den andre siden skape nye problemstillinger. En annen utfordring er at man vil til syvende og sist være avhengig av at elevene forstår spørsmålene, men også at de er ærlige om svarene sine. Ikke alle elever har like stor selvinnsikt, men VG1-elever vil antageligvis ha mer enn yngre elever. Ettersom elevene ikke var anonyme, er det derimot et mer åpent spørsmål om de var ærlige i besvarelsen av undersøkelsen. På spørsmålet om hvor godt de jobba i dag vil det for eksempel være fristende å si at man har jobba bedre enn man faktisk har jobba for å gi et bedre inntrykk til læreren, som tross alt har vurderingsansvar for elevene.

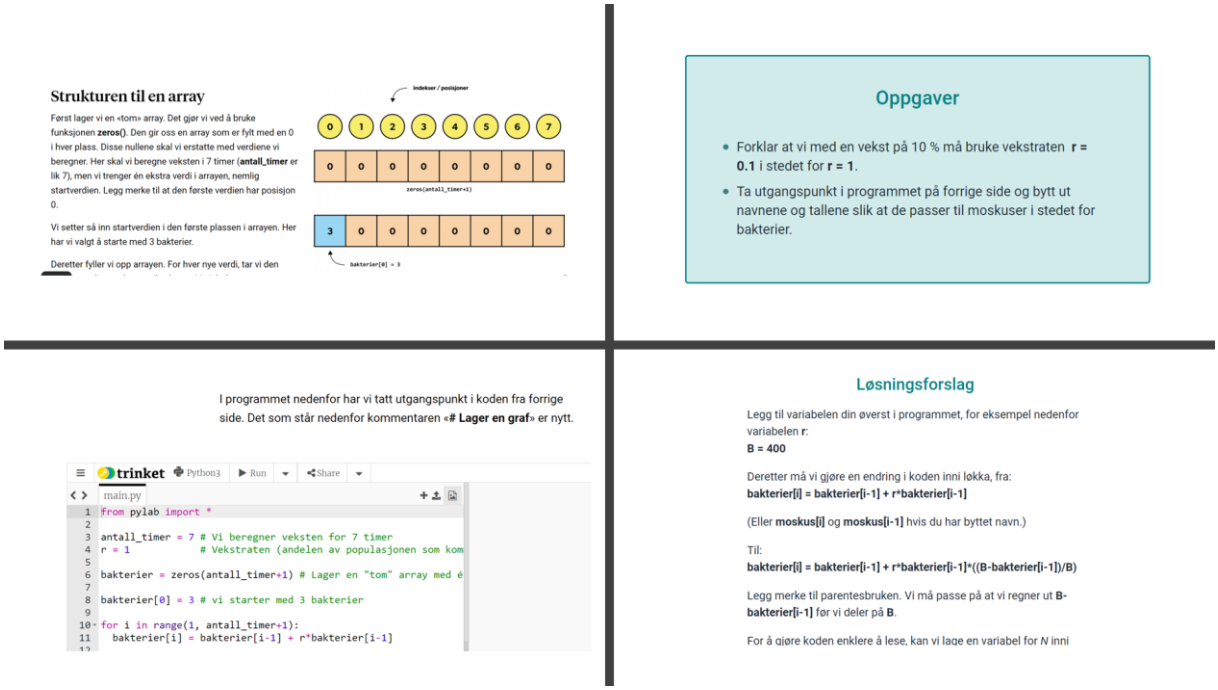
Slik jeg ser det er spørreundersøkelsen relativt godt egnet for det som var hovedhensikten, det vil si å gi lærerne tilbakemelding fra timen. Problemene knyttet til anonymitet og at undersøkelsen ikke er utarbeidet for å svare på problemstillingen begrenser likevel verdien dens for denne oppgaven. Jeg har likevel valgt å ta den med fordi jeg ellers ikke har fokusert på elevenes perspektiv i resten av oppgaven, og spørreundersøkelsen gir derfor indikasjoner på hvorvidt elevenes refleksjoner sammenfaller eller motsier resultatene fra de andre analysene.

3.4 Beskrivelse av undervisningsopplegget

Undervisningsopplegget var delt i tre deler. Først startet lærerne med en introduksjon hvor de avklarte mål for timen, gjenfortalte hva ChatGPT hadde fortalt dem da de spurte hvorfor man

må lære programmering i naturfag og sist, men ikke minst introduserte 4 egenskaper de ville at elevene skulle vise i løpet av timen: Samarbeid, tålmodighet, selvstendighet og utholdenhet.

I hoveddelen gjennomførte elevene et undervisningsopplegg fra Aschehoug univers/aunivers (2023) om populasjonsdynamikk og vekst i både en bakteriepopulasjon og moskuspopulasjonen på Dovrefjell. Opplegget er tilgjengelig fra https://aunivers.lokus.no/fagpakker/realfag/naturfag-sf/innhold/programmering/moskuspopulasjonen-paa-dovrefjell (krever lisens). Opplegget består av fagtekster om både programmering og naturfaglige fenomener, oppgaver, og områder hvor man kan endre på allerede eksisterende kode eller skrive ny kode. Opplegget refererer også til reelle populasjonsdata fra miljødirektoratet, men disse ble ikke brukt i undervisningen.



Figur 4: Skjermdumper fra undervisningsopplegget til Aschehoug.

Figur 4 viser skjermdumper fra undervisningsopplegget med eksempler på fagstoff, oppgaver, programmeringsfelter og løsningsforslag til oppgavene. Man kan kort oppsummere opplegget med at elevene først kjører og tilpasser en modell for populasjonsvekst hos bakterier. Denne modellen blir siden tilpasset slik at den passer for en moskusbestand som har mye tregere vekst. Til slutt blir bæreevnen til moskusene tatt med i modellen, slik at populasjonen stabiliserer seg rundt bæreevnen.

Elevene kan ikke velge oppgavene selv, men de kan velge hvordan de vil løse oppgavene og det vil finnes flere koder som gir riktig svar på spørsmålene. Dette kommer tydelig fram vet at det er løsningsforslag, ikke løsninger, dermed er flere ulike løsninger gyldige. På grunn av dette kan oppgaven betraktes som en relativt lukket utforskende arbeidsmetode fordi elevene har noe frihet i hvordan de løser oppgavene.

Noe lærerne trakk fram i intervjuet er at undervisningsopplegget ikke krever at elevene kan så mye programmering på forhånd, men at det blir fokusert på det naturfaglige aspektet i undervisningen. Man kan gjennomføre undervisningsopplegget uten å kunne så mye programmering og dette gjør det lettere å beholde fokuset på naturfag og ikke selve programmeringen.

Til avslutningen ville lærerne at elevene skulle sende en evaluering av timen og deres egen innsats, for at lærerne skulle få tilbakemeldinger fra timen. Denne evalueringen er siden blitt anonymisert og er med i datamaterialet til denne oppgaven for å få en antydning av hvor godt elevene selv sier de har lært. Tallverdier er presentert i figur 10, og enkelte sitater fra tekstsvar er gjengitt i diskusjonsdelen av denne oppgaven. Ettersom undersøkelsen i utgangspunktet ikke var anonym, kan dette ha påvirket tilbakemeldingene i noen grad og den er derfor ikke tillagt stor vekt.

3.5 Beskrivelse av klassene

Selv om Ola og Ida utarbeidet opplegget sammen og undervisningsopplegget var helt likt, var det likevel noen forskjeller både i elevgruppene og med hensyn til hvilke grep lærerne gjorde underveis. De viktigste forskjellene her er læreren og elevgruppene, altså subjektet i undervisningen og det sosiale samspillet dem imellom ut fra aktivitetssystemet i figur 3. I tillegg var arbeidsfordelingen litt ulik ettersom elevene ble delt inn i grupper på litt ulike måter. I Idas klasserom var det også oppfordret sterkere til at elevene skulle snakke med hverandre, så reglene for undervisningssituasjonen var også litt ulike.

3.5.1 Undervisningen i Olas klasse

I Olas klasse var det flere elever som valgte å gjennomføre oppgaven i et grupperom i stedet, og dermed ikke deltok i filmingen. Det var ca. 15 elever som ble igjen og dermed deltok i studiet. Dette gjorde bl.a. at støynivået var lavere og læreren brukte en noe mindre andel av

tiden på å gå rundt til elevene. En av elevene skrev i egenevalueringen at hen «(…) skjønte det mer fordi det var få i klassen», altså var det lettere å lære fordi det var færre elever enn vanlig. Ut ifra observasjoner og selvevalueringen virker det som at Ola har tid til å hjelpe litt mer i dybden enn Ida. Dette regner jeg med at henger sammen med antall deltagere i klasserommet.

Kjønnsmessig var fordelingen jevn, men omtrent to tredeler av deltagerne hadde T-matte, som lærerne i intervjuet etter undervisningen vurderte at kanskje kan ha en sammenheng med interessen elevene har for realfag og programmering generelt. Elevene satt i par og fikk beskjed om å samarbeide underveis.

Underveis i timen tok Ola opp flere problemer og spørsmål i plenum. Han tok opp spørsmål som har med forståelsen av hvordan tall skrives i Python med utgangspunkt i at elevene skulle forstå tallet 1.024 E+3 og spørsmål om hvilke sider ved modellen som er realistiske og hvilke sider som ikke er realistiske. Klassen som helhet fikk ikke begynt skikkelig på siste del av opplegget med bæreevnen, men det kan være enkeltelever som gjennomførte hele opplegget. I deler av plenumsundervisningen foregikk det enkel live-koding, og i en situasjon hvor en elev fikk laget en kode for bæreevnen til moskuspopulasjonen ble det forsøkt å reprodusere denne til plenumsundervisningen, men dette fungerte ikke fordi kun halvparten av koden ble med når denne ble oversendt.

Det siste av Olas spørsmål i plenum var et spørsmål som ikke var like konkretisert i undervisningsopplegget. Ved å konkretisere dette nærmere og ta det opp i plenum påvirker Ola rammene for timen ved at noe blir vektlagt i større grad enn andre ting. Han krever også at alle elevene følger med samtidig, selv om de er på ulike oppgaver. Å gi konkrete rammer for timen er en viktig del av det å være lærer.

Ola har en bakgrunn med mye programmering, og ifølge intervjuet samarbeider han med Ida ved at de snakker sammen om hvordan de vil lage opplegget, men det er Ola som gjennomfører selve kodingen, fordi det er mest effektivt.

Ola er opptatt av at det er viktig at elevene får en skikkelig forståelse av programmeringen for at de skal kunne bruke det og forstå det. Dette legges vekt på i intervjuet med lærerne hvor han snakker om at det er viktig å starte med det som er kjedelig, som hva variabler er og hvordan likhetstegnet fungerer i programmering, siden det ikke er det samme som i matematikk. Ola underviser nemlig en del av elevene i T-matte og har undervist noe

programmering til dem allerede. Han hadde i utgangspunktet ganske lave forventninger til undervisningsopplegget, men ble positivt overraska over hvor godt det fungerte.

3.5.2 Undervisningen i Idas klasse

I Idas klasse deltok nesten alle, og de var ca. 25 elever. I denne klassen var det en liten overvekt med jenter. Elevene samarbeidet to og to på større bord med ca. 6 elever per bord. Elevene interagerer ofte på tvers av parene og det var noe støy fra diskusjoner rundt om i klassen omtrent hele tiden, med mindre Ida ga en beskjed i plenum. Det er derfor rimelig å anta at dette gjør at parene lettere kunne samarbeide med hverandre, ikke bare innad i paret.

I løpet av undervisningen peker Ida flere ganger på de fire stikkordene for timen: samarbeid, tålmodighet, utholdenhet og selvstendighet og flere ganger i løpet av økta oppfordret hun elevene til å snakke sammen – uansett om de forsto oppgaven eller ikke forsto. Dette kan også sees ut fra søylediagrammene i for eksempel figur 5 i resultatdelen, ved at Idas klasse har en mye større andel av timen som koder for pedagogikk (PK) som i en del tilfeller handler om nettopp dette. Dette og organiseringen av klasserommet bidro antagelig til at det var betydelig mer snakking i dette klasserommet enn i Olas klasserom. Noe av dette er fordi det var flere elever, men det var også mer snakking per elev.

Ida hadde opprinnelig tenkt å ta opp felles problemstillinger i plenum, men valgte ifølge intervjuet å la være fordi det var så god flyt i jobbinga. Dette gjør at undervisningen holder seg mer elevsentrert og Ida må dermed gi struktur til undervisningen på andre måter, i dette tilfellet ved at hun fortsetter å gå rundt og veilede elevene i grupper.

I Idas klasse kom flere elever til slutten av opplegget til aunivers, men det virket ikke som at noen elever ble helt ferdig før helt på slutten av timen. Når Ida gikk rundt i klasserommet var det flere elever som rakk opp hånda samtidig, og det tok av og til litt tid før elevene fikk hjelp av Ida, fordi hun var opptatt med å hjelpe andre elever. Siden det var flere elever var det flere elever som ikke fikk hjelp før det hadde gått en del tid og en av elevene skrev i evalueringen at Ida måtte gi «kjappere hjelp, fikk det først etter lang tid».

Ida har en bakgrunn med introduksjonsemne i programmering og har derfor mindre erfaring med programmering enn Ola. Hun har likevel kontroll på spørsmålene elevene gir henne i timen. Ida underviser i 1P-matte, og har derfor kjennskap til en del elever som sliter med matematikken. Fra intervjuet kom det fram at Ida prioriterer at elevene lærer det naturfaglige i

oppgaven heller enn programmeringsdelen i noe større grad enn Ola, og har blant annet en lav terskel for å anbefale at elevene ser på løsningsforslag.

Ida hadde mer positive forventninger til opplegget enn Ola, men også hun ble fornøyd med resultatet. Dette kommer blant annet fram av en kommentar mot slutten av intervjuet – «Vi skal gjøre programmering mer fordi … dette gikk jo så bra!»

3.6 Intervjudata

Intervjuene var delt inn i et kort førintervju med hver av lærerne og et hovedintervju med begge to. I førintervjuene spurte jeg kort om hvilket inntrykk lærerne hadde rundt undervisningsopplegget og hva lærerne trodde elevene ville få ut av timen, og om dette ville ha en relevans for det andre de har lært i naturfag. I tillegg spurte jeg om en del faktainformasjon som er gjengitt i beskrivelsene av klassene i stedet.

Hovedintervjuet var delt inn i en del generelt om programmering og programmeringsbakgrunnen til læreren og elevene, en del om selve undervisningsøkta og en del om refleksjon rundt programmering i naturfagundervisningen. For å presentere dette mer oversiktlig og tematisk har jeg delt disse inn i en generell del om programmering, en del spesielt knyttet til utfordringer med undervisningsøkta og en del om lærernes generelle opplevelse av undervisningsøkta. Som tidligere har jeg latt være å gjengi faktaopplysninger som er beskrevet under beskrivelsene av timene og undervisningsopplegget, med mindre intervjuet legger til et viktig perspektiv.

3.6.1 Førintervju - Ida

Som utfordringer trakk Ida fram hvordan variable defineres, konseptet med en array og for eksempel hvorvidt elevene forstod hva som var kommentarer i programmet. Samtidig oppfattet Ida undervisningsopplegget slik at man ikke nødvendigvis trengte å forstå alt for å få mestringsfølelse i opplegget, og dette ble trukket fram som en styrke med opplegget.

Ida trakk fram var at læreplanen regner med at elevene kan mye mer programmering enn de faktisk kan. Læreplanen forutsetter at elevene har programmert siden 5. klasse, men det har de ikke, for de har fulgt en eldre læreplan uten programmering. Man må derfor ta læreplanen med en klype salt, fordi det er begrenset hvor mye man får lært elevene på et år. I tillegg er

det utfordrende at man som lærer plutselig skal kunne undervise med programmering bare ved å lære programmering selv, uten at man har fått opplæring i selve prosessen med å undervise elever i programmering. Dette nevnes også av Ola i hovedintervjuet.

Ida tenker at elevene får bruk for programmering i arbeidslivet, blant annet fordi de har hatt folk inne i klassen som har snakket om viktigheten av nettopp dette. Hun er likevel usikker på hvor nyttig det blir for selve naturfaget. Hun trekker fram r-tallet i forbindelse med koronapandemien og et opplegg med befolkningsvekst i mattetimene som temaer som var litt relevante og kanskje ville være nyttige for elevene, ellers håper hun at opplegget vil gjøre det litt lettere å fortsette å jobbe med programmering i framtiden.

3.6.2 Førintervju – Ola

Olas førintervju ble gjennomført skriftlig av praktiske årsaker.

Ola trekker blant annet fram at opplegget går litt raskt framover og skulle gjerne hatt mulighet til å tilpasse det i større grad, men dette var vanskelig på grunn av brukergrensesnittet hos Aunivers. Det er vanskelig å nå kompetansemålet om å simulere naturfaglige fenomener og samtidig gi elevene mulighet til å arbeide kreativt med programmering, som Ola mener er en veldig viktig faktor for at elevene skal bli interesserte i programmering.

I timen delte Ola opp et spørsmål om modellen for vekst av moskuspopulasjonen var realistisk, men det hadde vært fint om man kunne tilpasset dette i nettleseren også.

Ola tror elevene vil få utforsket litt og gjort noen oppdagelser og at de dermed får øvd litt på den naturvitenskapelige metode. Han tror opplegget vil gi elevene mulighet til å bruke sine logiske kapasitet, uavhengig om de har høy eller lav karakter i naturfag og at dette vil gi mestringsfølelse. Samtidig tror han det blir veldig vanskelig å få elevene interessert i programmering.

3.6.3 Hovedintervju

Generelt om programmering

Både Ida og Ola har fått noen slags tilbud om etterutdanning i programmering, men ingen av dem følte at opplegget klaffa helt. Ida hadde mest om oppvarming til

programmeringsaktiviteter, mens Ola fikk tilbud om et kurs for viderekomne der de fremdeles lærte om grunnleggende ting og ikke hadde programmeringsdidaktikk. Ettersom Ola allerede har lært grunnleggende programmering og det ikke var noe programmeringsdidaktikk var ikke kurset spesielt interessant.

På spørsmål på hvordan de samarbeider sammen og med resten av lærerne på skolen trekker Ola fram det å samarbeide i utarbeidelsen av opplegg, for å finne relevante opplegg, fordi det ifølge Ola er «veldig mange ting man kan gjøre i programmering som er veldig uinteressant, som er bortkasta tid». Han sier at det å finne relevante oppgaver å bruke programmering til er det vanskeligste med programmering. Dette støttes av Ida: «det er ganske lett å bruke ferdiglagde undervisningsopplegg, men det å lage et selv er dødsvanskelig».

Det kommer fram at det er en fordel å ha jobba med programmering i matematikken, bl.a. fordi det gjør at man vet hva elevene sliter med og kan ta utgangspunkt i dette.

Av oppgavene som gis får elevene stort sett programmer som skal brukes eller som skal redigeres. Poenget er ikke at elevene skal lære seg å kode – det kan de gjøre i valgfag programmering. Etter en liten tid kommer Ola på at det kanskje sto noe om å lage programmer i kompetansemålet likevel, men der er ikke elevene helt enda. Dette henger ifølge han sammen med at elevenes kompetanse henger noen år igjen bak læreplanen.

I intervjuet kom det fram et behov om at man i kompetansemålene presiserer om man skal vurdere det naturfaglige eller programmeringen i programmene, fordi kompetansemålet er tvetydig.

Angående rollen til programmering i naturfaget kommer det klart fram at det benyttes som et verktøy. I tillegg sier Ola at det gir det en bedre forståelse for hvordan dataprogrammene rundt oss fungerer – at det faktisk er kode – og dermed gir det en bedre forståelse av hvordan verden rundt oss fungerer. Ida legger til at siden vår verden er så digitalisert gir det mening at programmering er en del av naturfag, fordi argumentet for naturfag er at det beskriver alt vi ser og kan ta på i verden. Hun legger til at når vi bruker programmering, som er ekstremt teoretisk og teknisk, må vi bruke det til å visualisere noe naturfaglig.

Utfordringer i undervisningsopplegget

For at elevene skal kunne løse en del problemer selv har Ola og Ida lært elevene hvordan de kan lese feilmeldinger. I tillegg opplever de at det er manglende selvstendighet, tålmodighet, samarbeid og utholdenhet som oftest gjør at elevene gir opp, og derfor har de fokus på dette. Ola sier at når han må hjelpe elevene er det ofte fordi de har gitt opp, heller enn at de bare ikke har skjønt det og ikke klarer å komme videre.

Ida nevner at når elevene skulle skrive inn en formel i koden om bæreevne mot slutten av undervisningsopplegget, stoppa mange opp. Dette kan henge sammen med at dette var et punkt hvor oppgaven endra karakter, så det ble en annen oppgave enn tidligere, ifølge Ola. I Olas klasse var det også en del av elevene som stoppet opp på den siste oppgaven, for der var det en helt tom koderute og elevene måtte lese oppgaveteksten for å finne ut hva de skulle gjøre. Dette var ikke et like stort problem i Idas klasse, for der var det en av elevene som spurte «må vi lese teksten» veldig tidlig og derfor var det flere av elevene som kom seg forbi det. Hun nevnte også at hun prøver å få elevene til å lese teksten en gang til dersom de ikke forsto den etter første gjennomlesning.

Ida måtte gå gjennom dobling med noen av elevene for å knytte programmet til grafen, fordi en elev lurte på om dette var riktig. Samtidig var det ikke det som var viktig i oppgaven. Ola påpeker at dette er litt nytt, for vanligvis når de jobber med tall kan de sjekke om det ble riktig. Til dette repliserer Ida at man kan gjøre dette i grafen også, men det er vanskelig å sjekke om det ble riktig når man ikke vet hva programmering gjør.

Ida refererte også til situasjonen beskrevet i figur 16 om gruppa som fikk feilmeldinger hele tiden fordi da de skulle endre r-tallet fra 1 til 0.1 endret de alle de andre ettallene i samme slengen, inkludert der de skulle lage en array med (antall_år + 1) plasser.

Ola opplevde at måten de brukte linspace i dette undervisningsopplegget var utfordrende, fordi man ikke kunne tilpasse det for å få en graf med flere datapunkter, som elevene hadde gjort med et lignende opplegg i matematikkundervisningen. I dette opplegget førte endringer til feilmeldinger fordi det var forskjellig antall punkter på x- og y-aksen i grafen. Dette ble spesielt utfordrende fordi hele førstesida handler om array og dette er en vanlig måte å jobbe med array på.

På spørsmål om hvordan lærerne hjalp elevene sa begge at de tror de av og til bare sier hva elevene skal gjøre, men de prøver å stille spørsmål slik at elevene kan finne det ut selv. Det er

flere grupper som ikke spør etter hjelp, men Ola går bort til dem likevel og spør hva de holder på med, for de spør egentlig ikke etter hjelp på egen hånd.

Et tema i intervjuet var om lærerne gjorde endringer underveis i undervisningsøkta. Ola gjennomførte økta omtrent som planlagt, mens Ida egentlig hadde tenkt å ta ting felles hvis det var flere som satt fast på det samme. Hun valgte å ikke gjøre dette fordi hun var innom hver gruppe så ofte at hun følte at hun hadde god oversikt over hvordan de lå an. De jobba bra, og derfor ville hun ikke avbryte med å ta opp ting i plenum.

Ida tilføyde at tematisk sett passet opplegget veldig godt med at de hadde jobba med befolkningsvekst og eksponentiell vekst i P-matte i forkant av økta.

Opplevelsen av økta

I starten av opplegget var Ola litt skeptisk til hvordan økta kom til å bli, men etter opplegget var begge mer positive. Ida forklarte at konseptet programmering er litt avskrekkende, men etter opplegget var faktisk det noen som sa «Dette er gøy – jeg følte jeg skjønte en del». Ola mente at opplegget var godt fordi elevene kunne jobbe i sitt eget tempo: «Man kan fremdeles få til ting uten å forstå – det gir mestring.»

Ola syntes egentlig programmering er litt lite tidseffektivt – fordi det bringer så mye som ikke er nevnt i læreplanen, og læreplanen er allerede full fra før. Samtidig gir programmering et nytt perspektiv på naturfaglige fenomener ved at de kan simuleres og man får med en del på kjøpet. Han tilføyde at på den andre siden jobba elevene hele tida og på den måten var det effektivt. Arbeidsinnsatsen var høy, og de lærte mye. Elevene snakka om det Ola egentlig vil kalle naturfaglig metode, fordi de diskuterte hvordan de ulike variablene påvirka simuleringen.

Det kom fram av intervjuet at begge lærerne egentlig ønsker å bruke mer reelle data, men elevene må få en del øvelse i å lage programmer før de kan lage noe komplekst på egen hånd. Det kan også være vanskelig å lage opplegg til datasett.

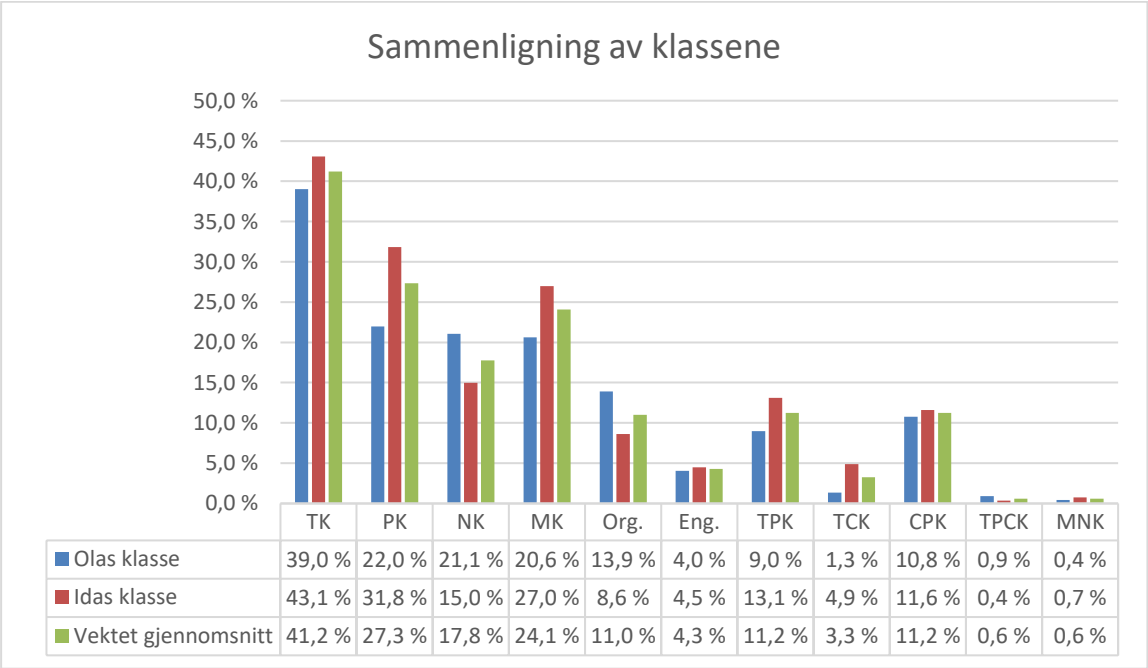
Helhetsinntrykket var likevel godt. Som Ida sa: «Vi skal gjøre programmering mer fordi … dette gikk jo så bra!»

4 Resultater

4.1 Resultater - TPACK

Figur 5 som er vist under viser en oversikt over hvordan segmenter i Olas og Idas klasse koder etter TPACK-modellen. De ulike bokstavene er utdypet mer i analysedelen av oppgaven og står henholdsvis for teknisk kunnskap (TK), pedagogisk kunnskap (PK), naturfaglig kunnskap (NK) og matematisk kunnskap (MK). TPK, TCK, CPK og TPCK er overlappen mellom de ulike kunnskapstypene i TPACK-modellen hvor CK er en kombinasjon av NK og/eller MK. MNK er de få gangene hvor segmenter koder for både matematisk og naturfaglig kunnskap, og er stort sett brukt av statistiske hensyn. De øvrige segmentene er delt inn i org. eller eng. Søylene merket org. er interaksjoner tilknyttet organisering og klasseledelse i klassen mens interaksjoner tilknyttet det å engasjere elevene i klassen er merket eng.

Ettersom et segment kan kode for flere bokstaver vil ikke søylene til sammen gi 100%, men dersom man legger sammen TK, PK, NK, MK, org., eng og søylen TPCK og deretter trekker fra søylene TCK, TPK, CPK og MNK skal dette til sammen gi 100% av segmentene om man tar hensyn til avrundingsfeil.



	TK	PK	NK	MK	Org.	Eng.	TPK	TCK	CPK	TPCK	MNK
Olas klasse	39,0 %	22,0 %	21,1 %	20,6 %	13,9 %	4,0 %	9,0 %	1,3 %	10,8 %	0,9 %	0,4 %
Idas klasse	43,1 %	31,8 %	15,0 %	27,0 %	8,6 %	4,5 %	13,1 %	4,9 %	11,6 %	0,4 %	0,7 %
Vektet gjennomsnitt	41,2 %	27,3 %	17,8 %	24,1 %	11,0 %	4,3 %	11,2 %	3,3 %	11,2 %	0,6 %	0,6 %

Figur 5: TPACK - Sammenligning av Olas og Idas klasse.

Til sammen er det snakk om 267 segmenter i Idas klasse og 223 segmenter i Olas klasse. Av Olas segmenter var 146 segmenter med gruppearbeid og 77 segmenter i plenum. Dette tilsvarer henholdsvis 66,75 minutter og 55,75 minutter i Idas og Olas klasser.

Siden undervisningen ikke inneholder like mange segmenter er det den prosentvise andelen og ikke antall segmenter som er brukt i diagrammene. Det kan være at dette påvirker datasettet i noen grad, men ettersom antall segmenter totalt sett er ganske likt har jeg gått ut ifra at påvirkningen vil være liten og sannsynligvis mindre enn andre tilfeldige forskjeller mellom timene. For å lage et gjennomsnitt på tvers av klassene har jeg tatt hensyn til hvor mange segmenter det var i de ulike klassene. Dette vil i praksis bli påvirket mer av Idas klasser, spesielt når gruppeundervisningen sammenlignes under i figur 6, ettersom det er flere segmenter i Idas klasse.

Olas undervisning består av 223 15-sekunderssegmenter, mens Idas undervisning består av 267 slike segmenter. Dette er hovedsakelig fordi det er noen deler av Olas undervisning hvor han ikke interagerer med noen, men for eksempel prøver å organisere kode for å vise i plenum. Det er også noen korte deler fra begges timer hvor filmen er kuttet. Dette er beskrevet nærmere under forskningsetikk.

Som man ser i figur 5 er det en del likheter mellom klassene. Omtrent 41% av undervisningen koder for TK, 27% for PK, 18% for NK og 24% for MK. Rundt 15% koder for eng. eller org. og koder ikke for de andre kategoriene. Naturfaglig kunnskap er dermed enkeltkategorien i min utgave av TPACK-modellen det kodes minst for dersom man ser bort fra de kombinerte kategoriene og eng./org. Av disse kategoriene er det PK som varierer mest mellom klassene: i Idas klasse koder nesten 10 prosentpoeng flere av segmentene for PK. Dette handler i stor grad om at Ida i mye større grad enn Ola refererer til de 4 målene for timen.

Dersom man slår sammen kategoriene NK og MK og derfor også trekker fra MNK ender man opp med at rundt 41% koder for CK, «content» kunnskap, altså er det fremdeles en betydelig del av timen som er relevant med tanke på fag, men er stor del av dette er knyttet til matematikk, ikke det rent naturfaglige.

Dersom vi ser nærmere på naturfaglig kunnskap er denne andelen høyere hos Ola enn hos Ida, dette vil jeg diskutere nærmere i figur 7. Andelen matematisk kunnskap er høyere hos Ida enn hos Ola. Dette kan ha en sammenheng med at Ida har noen av elevene i p-matte og kjenner til hvilke av elevene som sliter med den grunnleggende matematikken. Det kan også ha noe å

gjøre med at Ola forklarer noe av matematikken i plenum og derfor får forklart alle elevene hva «1.024 E+3» betyr samtidig ($1{,}024*10^3$, dersom noen lurer). Ida bruker for eksempel flere minutter på å forklare en elev hva en eksponentiell funksjon er og hvordan man kan kontrollere at datamaskinen regner riktig, noe det vil være mindre behov for i en klasse hvor de fleste er over gjennomsnittlig gode i matematikk, og flertallet i Olas klasse hadde T-matte.

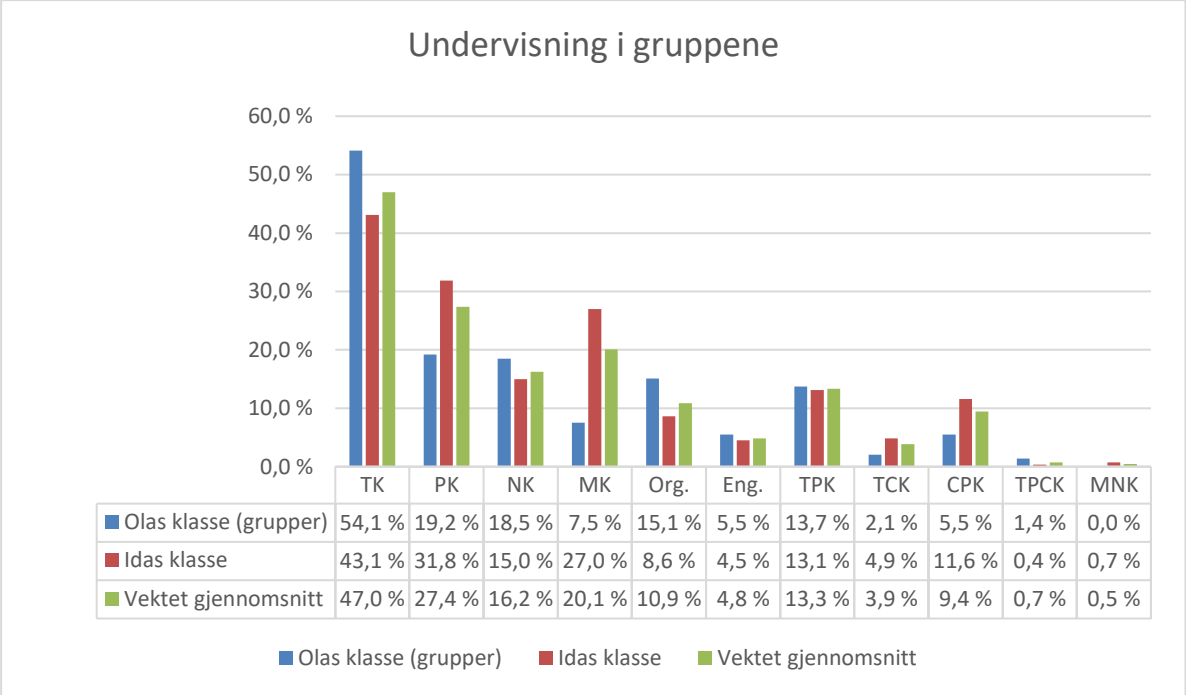
I sammenligningen av de sammensatte kategoriene TPK, TCK og CPK har jeg vurdert om fordelingen er høyere eller lavere enn det den ville vært dersom kategoriene var tilfeldig distribuert. Dette vil kunne si noe om det er en sammenheng mellom de ulike typene kunnskap. Dersom kategoriene er tilfeldig distribuert, vil frekvensen av for eksempel TPK tilsvare frekvensen av TK multiplisert med frekvensen av PK når man tar hensyn til tilfeldige variasjoner. Mer matematisk kan dette skrives som $frekvens\ (a \cap b) = frekvens\ (a) * frekvens\ (b)$, hvor a og b er ulike typer kunnskap. Beregnede frekvenser av disse kategoriene er vist og sammenlignet med faktiske verdier i tabell 2.

Tabell 2: Beregnede verdier for kombinerte kategorier.

	TCK	CPK	TPK
Ola (beregnet verdi)	16,1 %	9,1 %	8,6 %
Ola (faktisk verdi)	1,3 %	10,8 %	9,0 %
Ida (beregnet verdi)	17,8 %	13,1 %	13,7 %
Ida (faktisk verdi)	4,9 %	11,6 %	13,1 %

Som man kan se av tabellen skiller TCK seg klart ut fra de andre kategoriene ved å ha en mye lavere frekvens i virkeligheten enn i de beregnede dataene skulle tilsi dersom kategoriene var tilfeldig distribuert. TCK skiller seg fra kategoriene CPK og TPK hvor frekvensen er omtrentlig den samme som den beregnede frekvensen. Dette tyder på at den tekniske kunnskapen og den naturfaglige kunnskapen ble brukt adskilt fra hverandre, mens den pedagogiske kunnskapen i større grad ble brukt sammen med de andre kategoriene. Dette er ganske rimelig ettersom den pedagogiske kunnskapen handler om hvordan undervisningen ble gjennomført i større grad enn hva som var temaet for undervisningen. En lav frekvens av TCK kan likevel gjøre at den programmeringsfaglige delen av undervisningen og den naturfaglige delen av undervisningen blir opplevd som adskilte deler av undervisningen som ikke nødvendigvis har en klar sammenheng. Dersom man ser nærmere på de enkelte

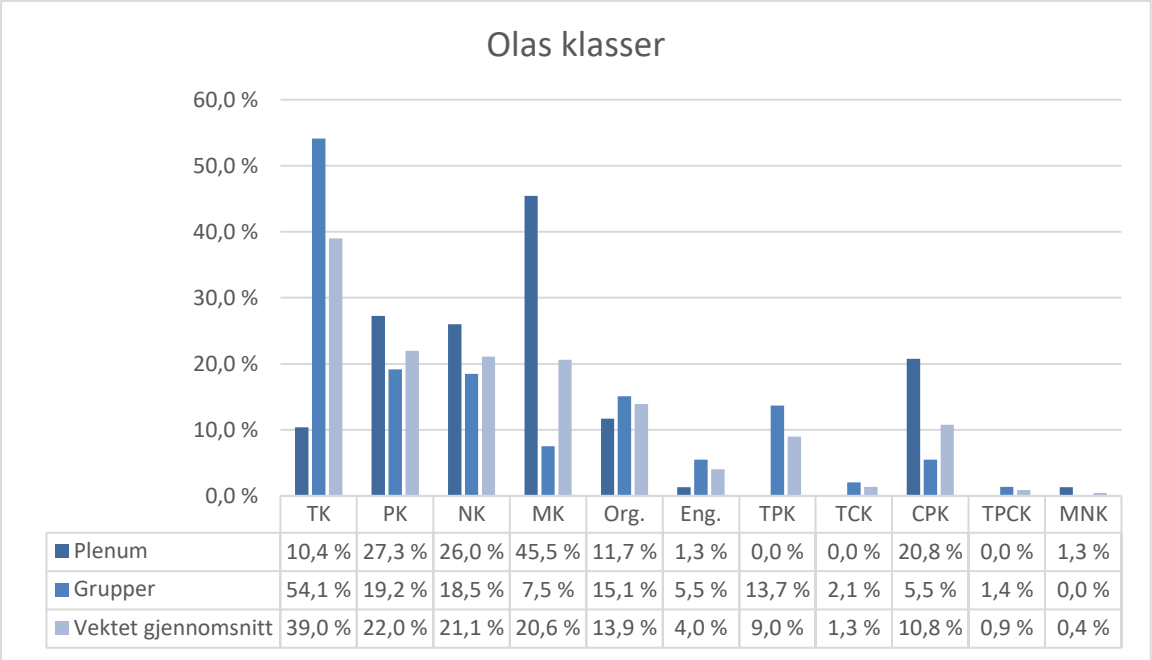
segmentene som koder for TCK er det dessuten en stor andel som koder for MK og TK, altså er kombinasjonen av naturfaglig kunnskap og teknisk kunnskap kun en del av denne prosentandelen.



	TK	PK	NK	MK	Org.	Eng.	TPK	TCK	CPK	TPCK	MNK
Olas klasse (grupper)	54,1 %	19,2 %	18,5 %	7,5 %	15,1 %	5,5 %	13,7 %	2,1 %	5,5 %	1,4 %	0,0 %
Idas klasse	43,1 %	31,8 %	15,0 %	27,0 %	8,6 %	4,5 %	13,1 %	4,9 %	11,6 %	0,4 %	0,7 %
Vektet gjennomsnitt	47,0 %	27,4 %	16,2 %	20,1 %	10,9 %	4,8 %	13,3 %	3,9 %	9,4 %	0,7 %	0,5 %

Figur 6: Sammenligning av gruppeundervisning i de to klassene.

I figur 6 vises en oversikt over fordelingen av koder i gruppeundervisningen til Ola og Ida. Dette representerer 267 segmenter i Idas klasse og 146 segmenter i Olas klasse. Antallet i Olas klasse er mye lavere ettersom han også brukte en del av tiden på plenumsundervisning. Det mest oppsiktsvekkende fra denne figuren er at Ola kun bruker 7,5% av tiden til å hjelpe elevene med ren matematikk. Dette kan henge sammen med at han bruker en betydelig del av tiden i plenum på å snakke om matematikk, men også at han har elever som har høyere matematikkompetanse.



	TK	PK	NK	MK	Org.	Eng.	TPK	TCK	CPK	TPCK	MNK
Plenum	10,4 %	27,3 %	26,0 %	45,5 %	11,7 %	1,3 %	0,0 %	0,0 %	20,8 %	0,0 %	1,3 %
Grupper	54,1 %	19,2 %	18,5 %	7,5 %	15,1 %	5,5 %	13,7 %	2,1 %	5,5 %	1,4 %	0,0 %
Vektet gjennomsnitt	39,0 %	22,0 %	21,1 %	20,6 %	13,9 %	4,0 %	9,0 %	1,3 %	10,8 %	0,9 %	0,4 %

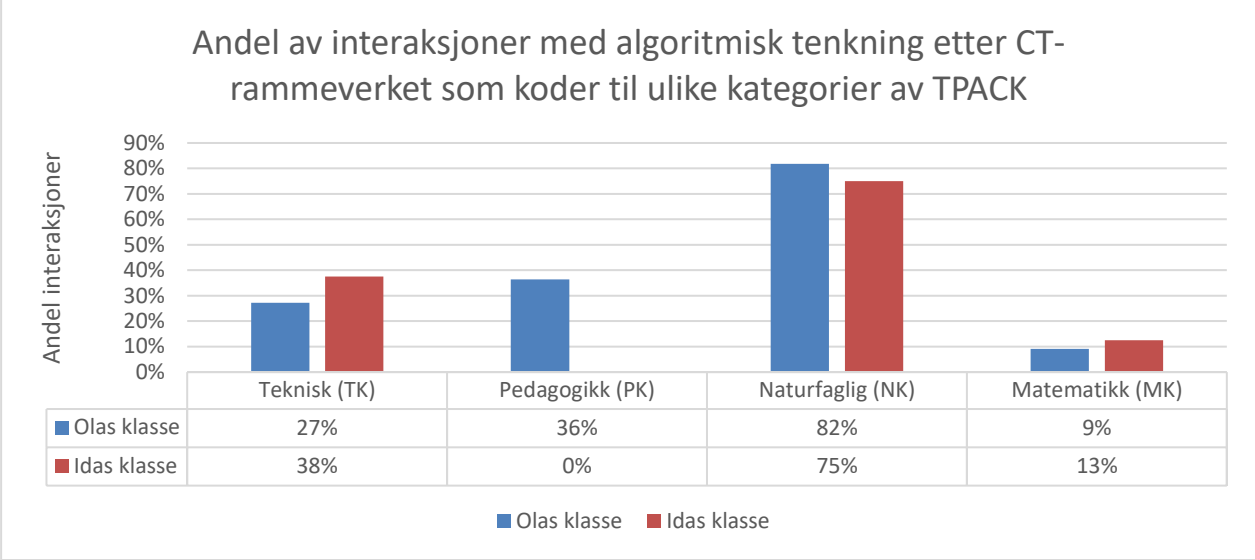
Figur 7: Sammenligning av plenumsundervisning og gruppeundervisning i Olas klasse.

Figur 7 viser en sammenligning av tiden Ola bruker i plenum og tiden han bruker med de enkelte gruppene. Søylene er her farget med samme fargenyanser for å tydeliggjøre at det her er snakk om samme klasse. Vi ser at det er en stor forskjell i kategoriene disse koder til. Noe av dette er antageligvis stokastiske eller tilfeldige forskjeller fordi plenumsundervisningen bare består av 77 15-sekunderssegmenter mot gruppeundervisningens 146. Det er likevel så store forskjeller at det er verdt å kommentere. Blant annet står det tekniske (TK) for henholdsvis 54% av gruppeundervisningen mot bare 10% av plenumsundervisningen, og matematikkandelen står for henholdsvis 45% av plenumsundervisningen mot 8% av gruppeundervisningen. Vi ser også en større andel av pedagogikk (PK) og naturfag (NK) i plenumsundervisningen og andelen som kombinerer pedagogikk og naturfag eller matematikk (CPK) står for 21% av plenumsundervisningen, mot 6% av gruppeundervisningen. Kategoriene TPK og TCK er fraværende i plenumsundervisningen.

Det er helt tydelig at Ola vektlegger ulike temaer i plenumsundervisningen og i gruppeundervisningen. I plenumsundervisningen får han gått gjennom hvordan man kan forstå hvordan Python skriver tall og hvor realistisk bakteriemodellen er og i hvilke sammenhenger modellen vil passe. Dette gjør at han får løftet fram temaer han ønsker at elevene skal ha i bakhodet mens de jobber med programmene, eller han får oppklart vanlige misforståelser til alle elevene samtidig. Samtidig kan dette forstyrre elever som har kommet lengre eller ikke har kommet langt nok til å henge med. Ut fra figur 7 ser vi at

plenumsundervisning kan være et viktig verktøy og bruken av plenumsundervisning vil bli diskutert nærmere i diskusjonsdelen av denne oppgaven.

4.2 Resultater - CT-rammeverk



	Teknisk (TK)	Pedagogikk (PK)	Naturfaglig (NK)	Matematikk (MK)
Olas klasse	27%	36%	82%	9%
Idas klasse	38%	0%	75%	13%

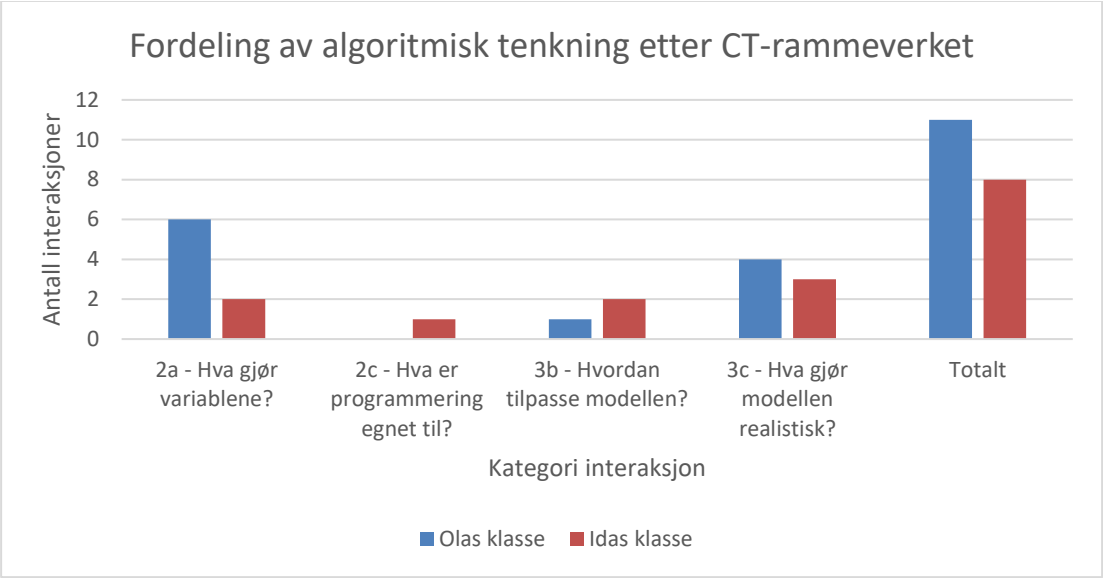
Figur 8:Sammenligning av Olas og Idas klasser ut fra CT-rammeverket.

Figur 8 sammenligner hvordan interaksjonene med algoritmisk tenkning ut fra CT-rammeverket koder med hensyn på TPACK-modellen. Ettersom hver av interaksjonene kan kode etter flere bokstaver i TPACK-modellen vil ikke dette utgjøre 100% til sammen.

Vi kan se at en veldig stor andel av interaksjonene som koder for kategorier i CT-rammeverket også koder for naturfaglig kunnskap. Dette er påfallende når man legger til at bare rundt 17% av undervisningen kodet for naturfaglig kunnskap. Dette henger sannsynligvis sammen med at flere av kategoriene for CT-rammeverket sammenfaller med kategorier for naturfaglig kunnskap – for eksempel er vurderingen av naturfaget i oppgaven knyttet til begge kategoriene, og den naturfaglige tenkemåten som henger sammen med algoritmisk tenkning vil også i mange tilfeller havne i begge kategoriene. Ut ifra CT-rammeverket ser vi også at det er en stor forskjell på matematisk kunnskap og naturfaglig kunnskap. Matematisk kunnskap var vanligere enn naturfaglig kunnskap i TPACK-modellen, men matematisk kunnskap står bare for omtrent 10% av interaksjonene i CT-rammeverket.

Jeg har valgt å presentere dataene i figur 8 med andelen interaksjoner fordi det er andelen interaksjoner som er mest relevant for å knytte sammen TPACK-modellen og CT-modellen og dette tydeliggjør dette aspektet ved dataene. Her må det likevel presiseres at interaksjonene

var av ulik lengde og derfor ikke nødvendigvis korresponderer fullstendig til tiden som ble brukt i de ulike interaksjonene, og derfor kan virke litt misvisende.



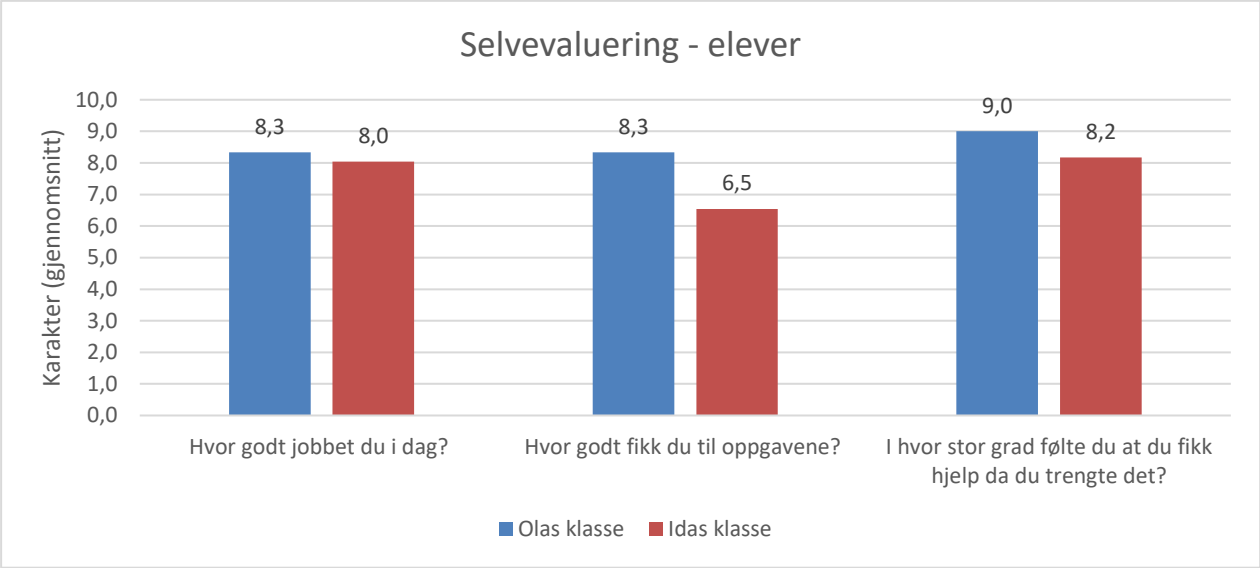
Figur 9: Antall interaksjoner med algoritmisk tenkning i de ulike klassene.

Figur 9 presiserer hvilken kategori av CT-rammeverket de ulike interaksjonene koder til. Her er det antall interaksjoner som er vist i y-aksen, ikke andelen. Søylediagrammet bruker kun korte former av beskrivelsene av kategoriene av plasshensyn.

I figur 9 har jeg valgt å presentere antall interaksjoner i stedet fordi andelen fordi jeg her er mest interessert i hvilke kategorier som er vanligst generelt, og med antall interaksjoner kan man lett legge sammen interaksjonene i begge klassene. Kombinasjonen av disse framstillingene gjør at det blir mer tydelig at det var relativt få interaksjoner som ble direkte knyttet til CT-rammeverket. Disse er tross alt samlet gjennom to dobbelttimer i hver klasse.

Dersom man går nærmere inn på de ulike interaksjonene i Olas klasse ser man at 5 av disse skjer i plenumsundervisningen til tross for at plenumsundervisningen bare var en tredel av hele undervisningen. Dette kan tyde på at en større andel av plenumsundervisningen var relevant for algoritmisk tenkning, men her er det så lite datagrunnlag at det er vanskelig å si noe sikkert.

4.3 Resultater - Selvevaluering



Figur 10:Sammenligning av selvevalueringen i Olas og Idas klasser.

Figur 10 viser gjennomsnittskarakteren som ble gitt på de ulike spørsmålene i selvevalueringen. Elevene skulle gi en karakter mellom 1 og 10. Søylediagrammet fanger ikke opp forskjellen i svarene mellom elevene. Dette er blant annet fordi Idas klasse hadde betydelig flere respondenter enn Olas klasse og dette ville påvirket for eksempel et standardavvik i stor grad.

Elevene i Olas klasse har gitt en noe høyere skår på at de fikk godt til oppgavene og at de fikk hjelp når de trengte det. Dette kan ha en sammenheng med at Olas elever ikke kom like langt i oppgavene og derfor ikke kom til oppgavene som var vanskelig å forstå, eller det kan ha en sammenheng med at Ola gjennomgikk oppgaver i plenum og dermed gjorde at det ble lettere å forstå oppgavene. Det er også rimelig at Olas elever i større grad fikk hjelp raskt, ettersom det var få elever i klasserommet og dermed lettere å hjelpe alle elevene.

4.4 Resultater – interaksjonsanalyse

I interaksjonsstudiet har jeg valgt med 6 ulike utdrag. Utdragene i figur 11 og 15 er interaksjoner hvor det er mer tydelig hvordan elevene interagerer med hverandre, mens utdrag 12, 13 og 14 følger to elever i Olas klasse fra starten av timen til slutten av timen. Utdraget i figur 16 er i stor grad en lærer-elev-interaksjon med to elever. Utdragene ble valgt fordi delvis for å belyse elev-elev-interaksjoner som ikke er belyst i like stor grad som lærer-elev-interaksjonene, men også fordi de inneholder poenger som kan være nyttig å være bevisste på i et lærerperspektiv. I etterkant ser jeg at alle utdragene handler om den tekniske dimensjonen

til en viss grad. Dette var ikke bevisst fra min side, men det viser seg at mange av utfordringene i undervisningen på en eller annen måte hang sammen med programmeringen. Dette er kanskje ikke så rart når elevene har lite erfaring med dette, en viktig del av hensikten med timen var å lære selve programmeringen og rundt 40% av undervisningen fokuserer på teknisk kunnskap.

I transkripsjonene under i fig. 11-16 er kontekst og handlinger markert med doble parenteser rundt teksten. Tegnet «[» markerer at personen avbryter de andre. = betyr at teksten er en fortsettelse av tidligere tale, (.) er en kort pause og … er en lengre pause. Enkle parenteser gir tilleggsinformasjon om talen.

Fig. 11 og 16 er tatt fra Idas klasse mens fig. 12-15 er tatt fra Olas klasse. Fig. 11 og 12 handler i større grad om elev-elev-interaksjoner, mens fig. 13-15 beskriver en situasjon rundt de samme elevene og fig. 16 beskriver situasjonen rundt et elevpar og læreren.

((Elevene jobber med å tilpasse programmet slik at det fungerer på moskus i stedet for bakterier. Ida bøyer seg ned slik at hodet hennes er på samme nivå som elevene.))
f3 (til f4): Når det er timer det skal for hvert år #00:36:19-0#
Ida (til d3): Se moskus #00:36:22-9#
d3: Jeg forstår ikke hvor. #00:36:24-9#
d2: Så hvis vi starter med en moskus, så etter 20 år blir det 7. Er det det det står? #00:36:31-7#
d3: Ser du, ((gestikulerer med hendene)) det var det jeg sa: Den funker ikke! #00:36:33-4#
Ida: Hva står det, hva står det, hva står det? #00:36:34-8#
d3: Det er no feil her. Name År is not defined #00:36:38-3#
Ukjent elev: Det gir egentlig ikke mening #00:36:41-5#
Ida: Name år - nei, og det er fordi den heter "antall år". #00:36:48-1#
d3: Skal den bare hete år? Antall år der også. #00:36:53-9#
d2: Skal vi bare kopiere vår kode helt akkurat det samme på hennes PC også? Sånn at... #00:36:58-3#
Ida: Har du trykka run? #00:36:59-8#
d3: Ja #00:36:59-8#
d2: Se det funker, det er samme kode. #00:37:02-1#
d3: Det funka til han også. #00:37:04-6#
d4: Vi har da helt lik kode #00:37:04-8#
Ida: Det her minner meg om da jeg studerte og droppa ut av et kurs (elever ler) fordi jeg ikke skjønte hvorfor mitt program ikke funka og de andre sitt funka. I feel your pain. #00:37:14-6#
d2: Skal vi prøve å skrive akkurat det samme? #00:37:14-8#
d3: Det går greit #00:37:15-9#
Ida: Nei det ville ikke vært så lurt. Ja, tålmodighet. Bra. Jeg hadde ikke så mye av det #00:37:21-4#
d3: Å, vent da. #00:37:24-5#
Ida: Ååh... #00:37:27-4#
d4: Det funker #00:37:29-4#
Ida: Nydelig, d4! #00:37:29-8#
d2: Hva var feilen? #00:37:32-0#
d3: Jeg skrev måneder #00:37:34-2#
Ida: [Fantastisk! #00:37:35-1#
d3 (til d4): Hvordan så du? #00:37:36-6#

Figur 11:Transkripsjon fra Idas klasse

Utdraget i figur 11 handler om en elev som ikke skjønner hvorfor programmet gir en feilmelding og får hjelp av både Ida og medelever. Elev d3 er tydelig oppgitt fordi programmet hens ikke vil fungere og skjønner ikke hvorfor. I filmen vises dette tydelig med gesten til eleven og med at stemmen hens endrer litt stemmeleie, som er typisk når man er oppgitt. Feilmeldingen sier at «name år is not defined». Medeleven d2 forsøker å hjelpe ved å tilby seg å la d3 kopiere sin kode, mens elev d4 (som sitter ved siden av d3) sammenligner koden med sin egen kode. Ida gjør eleven oppmerksom på feilmeldingen ved å spørre hva som står i den og sier hvorfor programmet ikke virker.

Ida identifiserer problemet ut fra feilmeldingen, i tillegg til å lytte til frustrasjonen til eleven og vise gjennom et eksempel fra sin egen studietid at hun har sympati og kjenner seg igjen i elevens situasjon. I tillegg knytter hun situasjonen opp til at programmering krever tålmodighet og at det å utvikle dette er et av målene for timen.

Elev d4 finner til slutt den siste delen av problemet – at elev d3 har brukt feil variabelnavn i en av linjene. Dette blir rost av Ida.

Hovedtemaet i denne samtalen er feilmeldinger og dette utdraget koder hovedsakelig til teknisk kunnskap.

((Ola sitter på huk mellom elev p1 og elev p2. Det er stille i klasserommet. Eleven har blitt bedt om å endre variablene i en modell med bakterievekst og se hvordan dette påvirker programmet. De har akkurat snakka litt om hvordan Python bruker punktum som desimaltegn. De ser på skjermen til p1.))
Ola: #00:15:12-4#Ehm... Nå endra jo du to tall...
p1: mhm
Ola: =Hvordan vet du... eller hva vet du hva de forskjellige to talla endra på? #00:15:23-0#
p1: Eeh... At de ((peker på skjermen))... bakterier... De starta med 3 bakterier og. Eh(.) Jeg husker... Jeg tok fort liksom hvor masse liksom bakterier er sånn #00:15:37-0#
Ola:[Ja #00:15:37-0#
p1:det var omkring 2 tror jeg #00:15:39-8#og så bytta jeg til 6. #00:15:40-7#
Ola:[Ja ja #00:15:41-8#
p1: og da kom det 6 ((trekker tilbake hånda)) #00:15:43-7#
Ola: [Ja. Og du har 2, så der står det 2. Okay, skjønner#00:15:48-6#
p1: Også... ((peker med hånda og trekker den inn igjen)) #00:15:48-6#
Ola: [Tallet 4, da? #00:15:51-2#
p1: eeh.. jeg vet ikke. #00:15:54-2#

Figur 12:Transkripsjon fra Olas klasse -1/3

I figur 12, 13 og 14 følger vi elevene p1 og p2 gjennom hele undervisningsøkta, med fokus på hvordan de forstår to linjer med kode som skal lage aksetitler. I figur 12 endrer elevene på to variabler for å se hva som endrer seg når de gjør dette, slik at de skal forstå hva variablene

gjør. I figur 12 er det særlig interessant at elevene prøver å endre begge variablene samtidig. Som Ola sier er ikke dette en god strategi, for da vil ikke elevene vite hvilken variabel som gjør hva. Variabelkontroll er et viktig tema for flere deler av oppgaven og vil bli diskutert nærmere under diskusjonspunktet.

((Ola snakker foran klassen i plenum. De diskuterer hvor realistisk modellen for bakterievekst er og har brukt melk som står framme på kjøkkenbordet som eksempel. På skjermen på tavla vises Python-programmet, inkludert et plott av bakterieveksten over tid. Linje 17 og 18 i Python-programmet skal lage aksetitler i plottet))
Ola: Og det er bakterier da (.) som gjør det. Så det kommer an på hvilke for-hold det er egentlig. Takk. Hva snakka dere om? ((Ola går nærmere elevene og henvender seg til p1 og p2)) Hva er det ja? #00:59:51-3#
p1: Eeh, prøvde å forandre på 18 og 17 og legge inn tall. Men det forandret ikke seg. #00:59:57-6#
Ola: Nei. #00:59:57-1#
p1: Og det ble... så den er realistisk nok til at (.) den er ikke så realistisk? ((Ola snur og går fram plottet)) til Vet ikke? #01:00:03-8#
Ola: Nei, fordi dere endra tittelen på aksene ((peker på skjermen)) her og så skjedde det ingenting (ja) Skjønner. Takk. #01:00:12-2#

Figur 13:Transkripsjon fra Olas klasse 2/3

I figur 12, 13 og 14 følger vi elevene p1 og p2 gjennom hele undervisningsøkta, med fokus på hvordan de forstår to linjer med kode som skal lage aksetitler. I figur 13 er det ved slutten av en plenumssamtale, hvor Ola snakker om hva to linjer gjør i programmet. Her kommer det fram at elevene ikke har skjønt hva disse linjene gjør. En nærmere beskrivelse følger etter figur 14.

Ola: Jeg tenkte bare å vise dere en ting. ((Setter seg på huk bak elevene))
Eeh, ja hvordan den ser ut. Okay. For jeg bare tenkte å vise en ting. Her på populasjonsvekst så hørte jeg du sa, p1 at disse tallene ikke gjorde noen ting. ((peker på skjermen)) Men det er de tallene som er der. #01:12:39-6#
p1: Som? #01:12:39-6#
Ola: Der står tallet 9, og der står tallet 8. #01:12:42-9#
p1: Ååh #01:12:42-9#
Ola: Så det er x-aksen #01:12:46-2#
Annen elev:[Ola, skal vi ha mer programmering?
Ola (til annen elev): ((trekker tilbake hånda)) Vi skal ha ja... Både i naturfag og i matte. #01:12:52-3#
Ola: =((peker på skjermen, pekefingeren følger x-aksen og y-aksen)) X-aksen går bortover der og y-aksen oppover der. Og du ser x-label står bortover der og y-label oppover ((reiser seg opp)). #01:12:58-9#
p1: Ååh, tro'kke vi merka det. #01:13:00-3#
Ola: Nei, nei, jeg skjønner det, men nå - jeg ville bare at du skulle se det nå. ((Går bort fra eleven)) #01:13:03-8#

Figur 14:Transkripsjon fra Olas klasse 3/3

I figur 12, 13 og 14 følger vi elevene p1 og p2 gjennom hele undervisningsøkta, med fokus på hvordan de forstår to linjer med kode som skal lage aksetitler. Til slutt i figur 14 på slutten av timen går Ola tilbake til elev p1 og p2 for å oppklare akkurat hva disse linjene gjør og følge opp det som skjedde i figur 13. Etter TPACK-modellen vil jeg kategorisere denne samtalen som hovedsakelig teknisk kunnskap, men det er også noen elementer av naturfaglig kunnskap i figur 12, ettersom dette også handler om den naturvitenskapelige tenkemåte.

Figur 13 og 14 må ses i sammenheng med hverandre. I figur 13 kommer det fram at elevene ikke fikk med seg at det skjedde noe når linje 17 og 18 ble endret, mens i figur 14 kommer det fram at grunnen er at elevene antageligvis bare ikke merka at det skjedde en endring i plottet. Uansett: Ola legger merke til dette og følger det opp ca. 12-13 minutter senere, helt på slutten av timen. Ved å følge opp dette bare med gruppen det gjelder slipper Ola at elevene føler seg irettesatt og han kan bruke mer tid på å følge dem opp uten at det har negative konsekvenser for relasjonen hans med elevene. Dette er selvfølgelig relevant for all undervisning, men kanskje ekstra relevant i denne sammenhengen fordi det er store forskjeller i hvor mye elevene kan om programmering og det gjør at de flinkeste elevene kan mye mer enn de svakeste i programmering. De flinkeste elevene vil ikke ha nytte av å få påpekt noe de allerede har kontroll på, og ved å påpeke dette foran klassen ville man bare svekket de

svakeste elevenes faglige selvvurdering og mestringsforventning på grunn av sammenligning (Skaalvik & Skaalvik, 2015).

Disse utsnittene viser også at det kan være vanskelig å se hva elevene lærer underveis i en økt med programmering, og man må som lærer gå tilbake for å følge opp det man har forklart, for å se hva elevene faktisk har lært og klarne opp i misforståelser. I denne masteroppgaven er det lagt stor vekt på det elevene sier, men det er ikke nok at elevene klarer å gjennomføre oppgavene eller si de rette tingene – elevene må også få en forståelse for hvordan programmering fungerer slik at de kan bruke dette til naturfag.

((Elevsamtale i bakgrunnen. Elevene skal kopiere programmet inn i et nytt programmeringsfelt for å gjøre noen flere endringer. Elev s1 har snudd seg i stolen og ser mot elev t1. Ola hjelper elev r1 som sitter i nærheten. Det er litt dårlig lyd på videoen fordi ingen av mikrofonene er helt i nærheten av det som skjer))
s1 til t1: (Utydelig) ((t1 løfter opp begge hendene i været og tar dem ned igjen)) Okay, så få se hva du har nå.. ((Prøver å ta PCen til t1)). #00:56:05-4#
t1: ((Tar tak i PCen for å holde igjen)) Nei. #00:56:05-1#
s1: Pikk ((Lukker igjen PC-en til t1)) #00:56:05-5#
t2: ((gjør kastebevegelse)) (Uklart) #00:56:07-1#
((t1 åpner PC-en igjen)) #00:56:08-4#
Ukjent: Ja han har målt... #00:56:10-1#
((Ola går forbi, bak elevene)) #00:56:11-1#
t1: Jeg sa aldri vært låsete(? uklart) #00:56:11-1#
s2: Ja, akkurat ((Håndbevegelse)), det var jo det jeg sa #00:56:13-5#
s1: Ja, akkurat, jeg trodde det var det du sa. #00:56:14-8#
t1: jeg trodde du sa at det siste var (uklart) #00:56:16-8#
((s1 prøver å snu PC-en til t1)). #00:56:16-8#
t1: ((holder igjen)) Nei, hvorfor... #00:56:18-4#
Ola (til t1): Kan jeg hjelpe deg? ((vendt mot s1)) Hæ, hva da? #00:56:22-6#
t1: ((peker. men vender ansiktet sitt til Ola)) Hun lurte på om jeg kopierte eller om jeg skrev for hånd, men jeg har ikke skrevet så mye på 3 sekunder. #00:56:27-0#
Ola: Neinei, det er veldig lurt å kopiere. #00:56:28-6#
t1: Hørte du det, s1? #00:56:29-9#
t2: Skal jeg kopiere alt fra forrige? #00:56:30-3#
t1: Det er lurt å kopiere #00:56:30-8#
Ola: Ja, gjør det #00:56:32-2#
s1: Det er lurt å kopiere, men hvis du skal huske det så må du gjøre det for hånd. #00:56:36-2#
r2: Nei, Nei, du skriver koden for hånd (uklart) #00:56:38-0#
t1: Men der .... #00:56:38-4#
(litt uro i klassen) #00:56:46-8#

Figur 15: Transkripsjon - konflikt om kopiering

Utdraget i figur 15 handler om en konflikt hvor elevene har en uenighet om det er greit å kopiere kode eller ikke. Dette kommer først fram på slutten av samtalen, antageligvis fordi det dessverre er ganske dårlig lyd i denne samtalen og derfor litt vanskelig å forstå hva elevene

sier. Ola snakker med elev r1 som sitter i nærheten og det virker som at han derfor ikke følger helt med på hva de andre elevene gjør.

Utdraget starter med at elev s1 prøver å få tak i PC-en til elev t1 for å se hvordan hen har løst oppgaven. Elev t1 holder igjen PC-en og s1 svarer med å bruke et skjellsord mot t1. På dette tidspunktet er Ola ferdig med samtalen han har hatt med elev r1. Når han går forbi elev t1 skifter konflikten litt karakter, det blir i hvert fall ikke brukt skjellsord lengre. Elev s1 prøver igjen å få snudd PC-en til t1, men t1 forhindrer dette og sier nei. Ola kommer bort til t1 og jeg tolker det slik at når Ola spør om han kan hjelpe til med noe er dette fordi Ola har forstått at det skjer noe mellom s1 og t1 og han vil bidra med å løse konflikten.

Elev t1 spør Ola om det er greit å kopiere, og når Ola svarer ja bruker t1 dette som et poeng i konflikten med elev s1. Elev s1 svarer med at det er lurt å skrive ned og ikke bare kopiere, fordi dette gjør at man husker bedre. Elev r2 blander seg inn i diskusjonen og påpeker at man aller helst bør skrive ned for hånd for at man skal huske det.

Ettersom dette er en litt sensitiv situasjon tok jeg kontakt med Ola i etterkant av analysen og fikk bekreftet at han ikke fikk med seg at elevene prøvde å ta PC-ene til hverandre eller at elev s1 brukte et skjellsord, men at han oppfattet at det var en slags konflikt og derfor tok kontakt med elevene. Dette er rimelig ettersom Ola var opptatt med å prate med en annen elev. I samtalen med Ola kom det også fram at de ikke har snakket om det å kopiere i naturfag, så dette har sannsynligvis kommet fra et annet sted.

Denne samtalen er vanskelig å kategorisere ut fra operasjonaliseringene i analysekapittelet alene. For det første er det et spørsmål om det er snakk om en interaksjon, ettersom læreren ikke er til stede før mot slutten. For det andre passer ikke operasjonaliseringene helt med samtalen. Jeg har valgt å ta med hele interaksjonen fordi læreren blir innblandet og hele interaksjonen er nødvendig for å kunne tolke situasjonen. Samtalen er i stor grad kategorisert som teknisk fordi den er knyttet til hvordan maskinen leser kode ved at kopiering forhindrer maskinen i å lese feil. Ettersom dette er en situasjon som kan tolkes på ulike måter har jeg i denne situasjonen også tatt i bruk definisjonen av TPK fra metodekapittelet som viser til at dette handler om blant annet når det tekniske og det pedagogiske er i konflikt med hverandre. Dette er tydelig i akkurat denne situasjonen og vil derfor være et argument for at det er en situasjon knyttet til teknisk kunnskap.

((Ida har akkurat gått til en ny elev. Elev f1 og f2 har kommet til delen av oppgven hvor de skal bytte ut bakterier med moskuser for å få modell som gir mening for moskuser i stedet. Ida står ved siden av f1 og bøyer seg ned slik at hodene deres er på samme nivå. De ser på skjermen til eleven. Elev f2 følger samtalen og har en stor rød feilmelding på skjermen.))
f1: Trenger hjelp #00:45:47-9#
Ida: Ooh, fin feilmelding. Så gøy. #00:45:49-2#
f1: Så bytta vi ut alle bakterier til moskuser #00:45:53-0#
Ida: [Mhm #00:45:53-6#
f1: =og så, siden det er med 10% og ikke ett prosent og så bytta vi til 0.1 #00:46:01-7#
Ida: [mhm #00:46:02-9#
f1: =Ja, for det skal være 10 % og så fjerna vi åene, for det kom feilmelding. #00:46:13-9#
Ida: [Det går nok bra med "å", men (.) for dere har åer her også #00:46:18-0#
f1: Åja, men det sto her at timer skulle... vi gjorde feil og så... #00:46:20-8#
Ida: mhm
f1: =Så sjekka vi løsningsforslaget for å sjekke hvor vi gjorde feil og så sto det at nei er alt det jeg skrev borte? #00:46:30-7#
Ida: Nei, nei. Du bare gikk en tilbake, håper jeg. Åja. Jo, det kan være du bytta ... nei det var her, var det ikke det? #00:46:37-6#
f1: Jo, det er det.) #00:46:40-9#
f2: Bare bytt form (utydelig) #00:46:48-1#
Ida: Da må du bytte den også, for den hører til den #00:46:51-2#
f1: (utydelig) #00:46:51-8#
Ida: mhm. Jeg tror ikke det er år som er problemet. Det er i linje 6, står det. Moskuser = zeros(antall_år + 0.1). Hvorfor står det +0.1? #00:47:06-6#
f1: Det sto allerede +1, og siden vi bytta... #00:47:11-1#
Ida: Å. Men den bruker bare r der det står r. Grunnen til at vi bruker en variabel ((Ida ser på elevene i stedet for skjermen)) er at du bare skal trenge å endre det akkurat der, så ((gestikulerer med hånda)) der hvor det sto +1... #00:47:27-7#
f1: ...skal det fortsatt stå +1 #00:47:27-2#
Ida: Ja. Så her så lager den arrayen(.) hvor er vi ((ser på skjermen og peker)). Den lager en array. Så den sier: I denne tomme notatblokken min med antall punkter ((bruker hånden for å markere punktene i en tenkt notatblokk)) på en måte så skal jeg ha 7 plasser ((peker på skjermen)) og en plass til. Og den klarer ikke å lage 0,1 plass så den må ha hele plasser å skrive data på #00:47:52-5#
f1: Det gir jo mening fordi at den i steden for å skrive 0.1, så skriver de bare r nå.
Ida: Mhm ((trekker til seg hånda))
f1: =så alle steder det egentlig sto 1 skal det fortsatt stå 1.
Ida: ja
f1: Ah. Det var derfor det gikk frem til vi gjorde det, men vi trodde at vi skulle bytte det på alle stedene. #00:48:06-1#
Ida: Mhm, skjønner. Det er derfor vi bruker r, for å slippe å bytte på alle stedene. For da får vi så mye sånn feil, eller at vi glemte det et sted. Det er derfor vi bruker variabler. #00:48:16-0#
f1: Vi tenkte ikke så langt. #00:48:17-8#
Ida: Det skjønner jeg godt. Det har dere ikke nok erfaring med programmering til å gjøre. #00:48:22-1#
f1: (til f2) er det riktig? #00:48:24-2#
f2: Ja #00:48:24-2#
Ida: Åh, bra! #00:48:26-6#
f1. Da er det ingen andre feilmeldinger - håper jeg. Ja #00:48:28-7#
Ida: Okay, fint. Bra. #00:48:32-8#

Figur 16: Transkripsjon – «Array»

Transkripsjonsutdraget presentert i figur 16 handler om en situasjon hvor elevene f1 og f2 får feilmeldinger de ikke skjønner hvordan de skal løse. Ida kommer bort til dem og starter samtalen med å si at det var en fin feilmelding. F1 fortsetter med å forklare at de er i den delen av oppgaven hvor de skal bytte ut vekstraten fra 1 til 0.1, for å tilpasse modellen til moskusene i stedet for bakterier. Siden de fikk feilmeldinger fjerna de alle å-ene, fordi de trodde at dette kunne skape feilmeldingene. De sjekka også løsningsforslaget for å finne ut hva de hadde gjort feil. Etter hvert ser Ida feilmeldingen, leser den for elevene og stiller dem spørsmål for å finne ut hvorfor de har endra programmet. Dette leder til at Ida forklarer elevene både hva en array er og hva variabler er og de får fiksa programmet

Ida starter samtalen med å si at det var en fin feilmelding. Dette kan være fint, for når man får røde feilmeldinger er det lett å bli skuffa, spesielt når man ikke skjønner hva som er feilen. Etter dette kommer vi til problemene elevene har i denne interaksjonen:

For det første forstår ikke elevene feilmeldingen. Dette kan være fordi de ikke leser den eller fordi de ikke skjønner hva den sier. For å løse feilen velger de i stedet å vurdere hva som kan være feil i programmet og bestemmer seg for at bokstaven «å» kan være en feil, siden den ikke finnes i det engelske alfabetet og derfor ikke alltid er gyldig som en bokstav i dataprogrammer.

For det andre har ikke elevene forstått hva koden gjør, hva arrays er og hvordan variabler fungerer. Dette er årsaken til at Ida begynner å forklare de ulike tingene.

I TPACK-modellen har jeg vurdert at dette er en interaksjon med fokus på det tekniske og det pedagogiske. Selve problemet er at elevene ikke forstår hvordan datamaskinen leser koden og løsningen er at Ida forklarer ved hjelp av metaforer hva de forskjellige funksjonene gjør.

5 Analyse

I oppgaven min vil jeg analysere interaksjonene mellom elever og lærere transkripsjonene både med hensyn på TPACK, men også ut fra CT-rammeverket. I tillegg har jeg brukt en interaksjonsanalyse for å gå i dybden på enkelte interaksjoner og jeg har sammenlignet resultatene fra selvevalueringen til elevene.

Det er flere begrensninger med datagrunnlaget. For det første er det et ganske lite datagrunnlag med observasjon fra bare to klasser. Dette gjør at funnene fra studien i liten grad vil være generaliserbare, men kan likevel gi et innblikk i hva det kan si å gjennomføre programmering i naturfagundervisningen i VG1. For det andre er det hovedsakelig interaksjonene mellom læreren og elevene som ble analysert av tekniske årsaker. Dette vil si at analysen ikke fanger opp all læringen som har skjedd i klasserommet, men fokuserer på det læreren har best grunnlag for å gjøre noe med, altså lærerens interaksjoner og interaksjoner som skjer rundt læreren. Det er også noen bruddstykker av elev-elevinteraksjoner som ble fanget opp hvor læreren har en birolle, men dette er unntak fra regelen.

I gruppearbeidet har jeg valgt å definere start og slutt av interaksjonene ved at læreren går til en ny gruppe med elever, og henvender seg til denne med kroppsspråk og tale. Dette vil markere en slutt av elev-elev-interaksjonene som skjedde før læreren kom, men også starten på en ny interaksjon. I plenumsituasjoner er start og slutt av interaksjoner definert ved at læreren stiller et spørsmål og en elev(gruppe) får ordet av læreren. Disse interaksjonstypene er noe ulike og vil derfor skilles fra hverandre. I analysen med TPACK-modellen har jeg telt antall sekvenser som koder for ulike kategorier, mens i CT-analysen har jeg kun telt antall interaksjoner. Dette er beskrevet nærmere under de ulike analysene.

5.1 TPACK-modellen

Analysen med TPACK-modellen ble gjennomført ved at jeg undervisningen først ble delt inn i 15-sekunders segmenter. Av disse segmentene har jeg vurdert hvilke som inkluderer en interaksjon mellom læreren og elevene og vurdert om hvert av disse segmentene koder for teknisk kunnskap (TK), pedagogisk kunnskap (PK), «Content» kunnskap (CK) som igjen er splittet i naturfaglig kunnskap (NK) og matematisk kunnskap (MK). Interaksjonene som ikke kodet for noen av disse er splittet i kategorien «org.» som handler om organisering og

klasseledelse og kategorien «eng.» som handler om å engasjere elevene. En interaksjon er definert litt ulikt i arbeid i gruppene og i plenum. I gruppearbeidet er starten på en interaksjon definert ved at læreren går til en ny gruppe eller henvender seg til en ny gruppe gjennom kroppsspråk og snakker med denne gruppen. Slutten på en interaksjon er definert ved at læreren går bort fra gruppa eller tydelig henvender seg til en ny gruppe. I plenum er en interaksjon startet ved at læreren stiller et spørsmål eller gir informasjon og henvender seg til en elev. Interaksjonen slutter ved at eleven gir et svar på spørsmålet og læreren med kroppsspråk ikke lenger henvender seg til eleven.

Dette betyr i hovedsak at jeg har utelatt segmenter hvor for eksempel læreren setter seg ved kateteret for eksempel for å se nærmere på noe kode. I tillegg har jeg selvfølgelig utelatt segmenter hvor jeg måtte slutte å filme av hensyn til forskningsetikk.

Jeg har dessuten utelatt oppstarten av timen og avslutningen av timen, fordi disse tydelig bærer helt andre preg enn hoveddelen av timen, blant annet med å introdusere oppgaven, at elevene finner fram til oppgaven o.l. eller at elevene fyller ut selvevalueringen og er derfor ikke helt sammenlignbare.

Jeg hadde opprinnelig tenkt å analysere hver interaksjon for seg selv, men fant raskt ut at det var stor forskjell på lengden av interaksjonene og at dette derfor ble lite representativt for dataene. Et annet problem var at det ble vanskelig å bestemme hvilke temaer som var viktigst i en samtale som først handlet om bare programmering et par minutter og deretter gikk over til å bare handle om matematikk, samtidig som krysningspunktet mellom disse, TCK, kun var delvis til stede. Jeg valgte derfor å dele inn undervisningen i segmenter på 15 sekunder fordi dette er en kort nok periode til at samtalen stort sett handler om det samme temaet, samtidig som det er en lang nok periode til at elevene og læreren for eksempel kan ha en kort pause i samtalen mens de venter på at datamaskinen skal jobbe, uten at transkripsjonen gir inntrykk av at det ikke har skjedd noe på en stund. Dersom perioden gjøres lengre vil det sannsynligvis bli flere segmenter som koder til flere kategorier og dette er viktig å være obs på dersom andre forskere gjentar studiet.

I TPACK-modellen har jeg tatt utgangspunkt i de deduktive kategoriene TK, PK og CK og splittet opp CK til NK og MK. Det kan være utfordrende å kategorisere innenfor disse typene kunnskap. Dette gjelder spesielt for den pedagogiske kategorien, ettersom pedagogikk er noe som forhåpentligvis gjennomsyrer det meste som skjer i klasserommet, uten at det

nødvendigvis er så lett å definere hvor det pedagogiske er et hovedtema. Av denne grunnen har jeg definert noen operasjonaliseringer som gjør det lettere å definere disse temaene.

I utarbeidelsen av disse har jeg tatt utgangspunkt i beskrivelsene gjengitt i metodedelen og deretter lagd kategorier avhengig av hva jeg forventer kan være relevant for de ulike kategoriene. Jeg gjennomførte filmingen før jeg hadde fullført å lage disse kategoriene og inntrykket fra filmingen har derfor i noen grad påvirket hva jeg forventet kunne være relevant for de ulike kategoriene.

Operasjonaliseringene er presentert i tabellene under. Jeg har koblet dette sammen med eksempler fra transkripsjonen for å tydeliggjøre hvordan jeg har kategorisert segmenter. Flere eksempler er vist i tabell 8, til sist i dette delkapittelet.

Teknisk kunnskap (TK) involverer minst ett av følgende elementer fra tabell 3:

Tabell 3: Operasjonalisering av teknisk kunnskap (TK)

Operasjonalisering	Eksempler
Programmeringselementer	Hva er en array? Hvordan fungerer en løkke? Hva vil det si å importere kommandoer? Hva er en variabel i programmeringssammenheng? Hvordan fungerer ulike funksjoner i Python?
Feilmeldinger	Elevene leser en feilmelding eller får hjelp til å lese en feilmelding for å se hva som er problemet i programmet.
Ulike tegn i programmeringsspråket	Hvordan skriver man et regnestykke i Python? Problemer knyttet til at man bruker punktum som desimaltegn i Python og ikke komma. Forvirring over at man ikke kan bruke kolon som deletegn i Python.
Spørsmål knyttet til hvordan datamaskinen leser koden	Problem med at variabler ikke er definert før de brukes i programmet. Diskusjoner om når i programmet ulike ting bør skje.

Naturfaglig kunnskap involverer minst ett av følgende elementer fra tabell 4:

Tabell 4: Operasjonaliseringer av naturfaglig kunnskap (NK)

Operasjonalisering	Eksempler
Naturfaglige begreper og modeller	Hva er en bæreevne? Hva er en r- og en K-selektert art? Hva er et økosystem? Hva er en populasjon? Hva er abiotiske og biotiske faktorer?
Vurdering av naturfaget i oppgaven	Er det realistisk at det er flere titusen moskuser på Dovre? I hvilke sammenhenger er eksponentiell vekst for bakterier realistisk?
Spørsmål tilknyttet populasjonsvekst i naturfag. Disse vil i noen tilfeller også falle inn under matematikk	Hvordan bruker vi r-tallet for å kalkulere vekst og hva betyr det i praksis? Hvordan fungerer formelen for bæreevnen i praksis? Hvordan vil veksten i en bakteriekultur påvirkes ved at man endrer miljøet?
Variabelkontroll	Hva er en avhengig variabel og hva er en uavhengig variabel? Hva kan være problematisk med å variere flere variabler samtidig?

Matematisk kunnskap involverer minst ett av følgende elementer fra tabell 5:

Tabell 5: Operasjonalisering av matematisk kunnskap

Operasjonalisering	Eksempler
Spørsmål tilknyttet eksponentielle funksjoner og å lese disse	Hva slags funksjon er dette? Hvilke egenskaper har slike funksjoner? Hvordan leser jeg av en graf?
Lesing av tall i Python og forklaringer av standardform, regnerekkefølge, nøyaktighet i et tall.	Hva betyr 3.4e-7? Hva betyr 3.4*10^-7? Hvordan regner man regnestykket 3.4*10^-7? Hvordan viser antallet desimaler nøyaktigheten av et tall?
Generell matematikk hvor det er tydelig at de matematiske ferdighetene til elevene er en del av problemet.	Hvordan dobler jeg et tall? Hjelp til utregning av regnestykker.
Kontrollering av matematikken til datamaskinen (også tilknyttet det tekniske temaet)	Hvordan kan jeg lese av grafen for å se at maskinen har regnet riktig?

Pedagogisk kunnskap regnes hvis man kan svare «i stor grad» på et av følgende spørsmål fra tabell 6:

Tabell 6: Operasjonalisering av pedagogisk kunnskap

Operasjonalisering	Eksempler
Bruker læreren metaforer og sammenligninger?	En array er som en skriveblokk der du lager plasser til å skrive noe. Du må ha med første linje (from numpy import *) fordi dette er verktøykassa
Svarer læreren med spørsmål som krever at elevene må konstruere sin egen kunnskap?	Hva tror du er oppover her (y-aksen) og bortover her (x-aksen)? (også MK) Hvorfor det? Har dere noen forklaring på det?
Omformulerer læreren spørsmålene dersom de ikke blir forstått?	Gjenformuleringer av andre spørsmål for å forklare hva som menes.
Bevisstgjør læreren elevene i stor grad på målene for timen?	Nå var dere flinke på tålmodighet! Når programmerere snakker med hverandre, så øker dere forståelsen i hele gruppa og for dere selv (samarbeid).
Knytter læreren ny kunnskap til tidligere kunnskap?	"Husker du at vi så på populasjonsvekst i en kommune i går i matten?"

Resterende segmenter er enten klassifisert som «org.», dersom det handler om klasseledelse og å gi informasjon til elevene, eller som «eng.» dersom segmentet handler om å engasjere elevene, for eksempel ved å fortelle historier fra lærerens egen studietid.

Kategoriene org. og eng. med eksempler er vist i tabell 7.

Tabell 7: Kategorisering av org. og eng.

Kategori	Eksempel
Org.	Kanskje ikke potetgull med PC-er? Kan dere legge det vekk? Se på dette tastaturet da!

	Jeg ser at du mangler strøm – har du en lader tilgjengelig?
Eng.	Se på dette her, Ola! «Dette minner meg om en gang på universitetet da …»

Det er viktig å presisere at dette også er viktige deler av undervisningen, selv om dette ikke er spesielt interessant for denne oppgaven.

For å gi fyldige eksempler på de ulike kategoriene har jeg presentert noen flere eksempler på de ulike kategoriene unntatt kategoriene org. og eng. i tabell 8 nedenfor. I tillegg er de ulike kategoriene kunnskap vurdert i utdragene fra interaksjonsanalysen.

Tabell 8: Flere eksempler på TPACK-kategoriene

Teknisk kunnskap (TK)	Pedagogisk kunnskap (PK)	Matematisk kunnskap (MK)	Naturfaglig kunnskap (NK)
Tips om at dersom man dobbeltklikker på en variabel i Python får du opp andre steder det er skrevet likt	«Nå var du god på tålmodighet og utholdenhet»	Eksponentialfunksjoner ser helt like ut dersom man ikke leser verdiene på x- og y-aksen	"Grafen beskriver ikke hvor mange som dør. Så når antallet går opp med moskuser, så blir flere født."
Hva er forskjellen på integer, float og andre tallformater i Python?	«Husker du at vi så på populasjonsvekst i går i matten?»	Hvordan kan jeg kontrollere at programmet regner riktig? (også TK)	Har bæreevne med sykdommer å gjøre? (tilknyttet bærere av gensykdommer)
Tips om å kopiere så man slipper å skrive feil (fra Ida)	Forklaring av pylab slik at det er forståelig	«Hvis vi tenker matematisk, så vil stigningstallet være veksten.»	Diskusjon om "mating season" knyttet til om grafen var realistisk.
«Name 'år' is not defined – hva betyr	Hva betyr det egentlig at r er 10?	Hva betyr 10E6?	Kan moskusene vokse uten begrensninger til

dette?			evig tid?

5.2 CT-rammeverket

I timene som ble observert er det spesielt dataprosessering og modellering som er sentrale punkter, da oppgaven fokuserte på disse områdene. Elevene fikk for eksempel ikke tilgang på reelle datamateriale over populasjonsdynamikken hos moskus eller bakterier, og dimensjonen «datainnsamling» ble derfor ikke relevant. «Problemløsning» ble heller ikke særlig relevant, ettersom dette kan kreve mer kunnskap om hvordan modelleringen og dataprosesseringen gjennomføres, og dette blir mer relevant når elevene selv finner et problem som skal løses.

I CT-modellen har jeg brukt deduktive kategorier, men har forenklet kategoriene noe fordi elevene i denne oppgaven har lite erfaring med programmering og nærmer seg kategoriene på et nybegynnernivå. Jeg hadde i utgangspunktet planlagt å telle segmenter som kodet for hver av de enkelte kategoriene på samme måte som i TPACK-modellen, men jeg fant raskt ut at det var få interaksjoner hvor algoritmisk tenkning ble observert og det var vanskelig å avgrense når de ikke skjedde lengre, fordi samtalen gjerne gikk over i et relatert tema som ikke i seg selv nødvendigvis passet inn i kategorien, men som i kontekst av resten av samtalen kanskje likevel passet inn i kategorien. Ved å kun telle antall interaksjoner forenklet dette analysen betraktelig uten å miste for mye av nytteverdien med antall segmenter.

Under CT-rammeverket har jeg kategorisert interaksjoner avhengig av hva slags aspekt av algoritmisk tenkning som er relevant for interaksjonen. De ulike kategoriene er vist i tabell 9. Ettersom opplegget ikke inneholdt reelle data og elevene ikke skulle løse et spesifikt problem på egenhånd ble særlig kategoriene datainnsamling og problemløsning lite relevante og disse er derfor utelatt fra tabellen. Jeg hadde opprinnelig tenkt å vurdere hvert eneste segment, men ettersom det var få relevante segmenter valgte jeg i stedet bare å telle opp antall interaksjoner som passer til kategoriene.

Tabell 9: Kategorier etter CT-rammeverket

	Reflekterende (a)	Design (b)	Evaluerende (c)
Bearbeiding av data (2)	2a) Hvordan kan jeg finne ut hva ulike	2b) Hvordan kan jeg få datamaskinen til å	2c) Hva er begrensninger og

	variabler gjør i programmet?	tolke datamaterialet mitt?	styrker med programmering for å bearbeide data?
Modellering (3)	3a) Hvordan kan jeg bruke en modell av en vekstrate for å forstå virkeligheten?	3b) Hva må jeg ta med for at modellen av vekstraten faktisk skal stemme med virkeligheten?	3c) Hvilke aspekter av modellen stemmer overens med virkeligheten og hvilke gjør det ikke?

I søylediagrammet i resultatdelen er kategori 2a forkortet til «Hva gjør variablene?», kategori 2c er forkortet til «Hva er programmering egnet til?», kategori 3b er forkortet til «Hvordan tilpasse modellen?» og kategori 3c er forkortet til «Hva gjør modellen realistisk?» for å passe til søylediagrammet. De andre kategoriene ble ikke observert.

5.3 Interaksjonsanalyse

For mer generelle beskrivelser av hvordan interaksjonsanalyser foregår, se metodekapittelet.

I mitt studium startet jeg analysen med å grovtranskribere begge videoene med vekt på tale for å identifisere sekvenser som kunne være interessante å analysere nærmere. Disse sekvensene har jeg deretter fintranskribert med vekt på kontekst, pauser og avbrytelser og hvordan deltagerne bruker skjermene for å kommunisere med hverandre, blant annet med vekt på peking og hvorvidt deltagerne ser på hverandre eller på skjermen, men også om de for eksempel viser skjermen sin til hverandre for å hjelpe hverandre. I tilfeller hvor deltagerne har brukt gestikulering for å legge vekt på poenger har også dette blitt tatt med. I studiet ville det vært gunstig å få med skjermbilder hos elevene i større grad, men dette ble vanskelig å få med på filmkameraet og har derfor bare blitt med i begrenset grad.

I valget av sekvenser har jeg prøvd å få med flere sekvenser hvor det skjer elev-elev-interaksjoner for å få disse representert i større grad enn i selve videoen av undervisningen. Dette gjør at klippene representerer en litt annerledes side ved gruppeundervisningen, ettersom de andre analysene i stor grad handler om interaksjoner med læreren. Jeg har prøvd å få med situasjoner som sier noe om konflikter og potensielle læringsøyeblikk i undervisningen fordi dette er deler av undervisningen det er nyttig for lærere å kjenne til. I et

tilfelle har jeg også sett hvordan to elever relaterte seg til det samme problemet gjennom hele økta for å komme mer i dybden på hvordan disse elevene har utviklet kunnskapen sin over tid. I tillegg har jeg sett etter om det finnes parallelle episoder i de ulike klassene, for dette vil være en indikasjon på om dette kan være vanlige fenomener som i større grad er representative for andre klasserom enn bare disse.

5.4 Spørreundersøkelse

Tabell 10 viser spørsmålene som ble brukt i spørreundersøkelsen.

Tabell 10: Oversikt over spørsmål i spørreundersøkelsen

Spørsmål	Hvor godt jobbet du i dag?	Hvor godt fikk du til oppgaven?	I hvor stor grad følte du at du fikk hjelp da du trengte det?	Skriv noe du lærte i timen i dag	Har du tips til neste økt med programmering?
Svartype	Skala, 1-10	Skala, 1-10	Skala, 1-10	Tekstsvar	Tekstsvar

I analysen av responsene har jeg funnet gjennomsnittsverdiene fra de første spørsmålene. Dette lar seg gjøre fordi spørsmålene har intervallnivå som målenivå, det vil si at det er like stor avstand mellom 9 og 10 som mellom 7 og 8, i hvert fall i prinsippet, og det gir derfor mening å finne gjennomsnittsverdier for alle responsene. Her har jeg vurdert å ta med flere måltall som median og kvartiler, men jeg har valgt å droppe dette fordi utvalgene er små og det er flere usikkerhetsmomenter knyttet til selve målene bl.a. fordi undersøkelsen ikke var anonym, og ved å ta med for mye statistikk kan jeg gi inntrykk av at tallene er sikrere enn de faktisk er. Jeg har også bevisst unngått standardavvik, fordi det er en stor forskjell i antall respondenter i de to klassene og jeg antar at dette vil påvirke standardavviket i Olas klasse mye mer enn i Idas klasse og dermed vil man tro at det er større forskjeller i svarene til Olas klasse, selv om dette i stor grad vil være fordi det er færre elever.

Ettersom jeg ikke lagde spørreundersøkelsen selv, vil jeg i denne delen også legge til ting som kunne forbedret spørreundersøkelsen dersom jeg hadde tenkt å bruke den kun til oppgaven. Spørreundersøkelsen bruker en kombinasjon av skalasvar og tekstsvar, og dette vil få fram

ulik informasjon. Tekstsvarene gir kvalitative svar mens skalaene gir kvantitative data, og jeg har tenkt at tekstsvarene kan brukes for å tolke svarene i skalaene.

I utformingen av skalaer er det flere spørsmål som må besvares. Et viktig spørsmål er hvor mange svaralternativer man bør ha. For mange alternativer kan gjøre det vanskelig å skille mellom ulike nivåer. I en skala fra 1-10 kan det for eksempel være vanskelig å skille mellom 7 og 8, og dersom jeg skulle gjennomført undersøkelsen selv ville jeg antagelig ha redusert antall svaralternativer, for eksempel til 5. Dette gir en grovere skala, men det vil være lettere å besvare riktig fra en skala på 1 til 5 enn fra en skala mellom 1 og 10, og man kan i større grad forsikre seg om at elevene tolker verdiene likt. Dette vil derfor styrke analysens reliabilitet på bekostning av noe nyanse. Reliabilitet vil blant annet si at analysen gir det samme resultatet dersom den gjentas. Et annet viktig spørsmål er hva slags variabler det er snakk om, fordi dette kan sette begrensninger for hvordan dataene bør behandles. Variablene som ble benyttet med skalaer er variabler på intervallnivå, det vil si at det er like stor avstand mellom de ulike tallene og resultatene kan derfor presenteres med gjennomsnittsverdier (Frønes & Pettersen, 2021).

I tillegg til dette ville det vært mer nyttig med en anonym spørreundersøkelse fordi dette ville økt sannsynligheten for at elevene gir ærlige svar. Det kunne også for eksempel vært nyttig å dele opp spørsmålene i flere spørsmål hvor noen av spørsmålene var motsatt formulert for å oppdage det dersom elevene ga høy karakter til alt uten å lese spørsmålene. Dette ville dermed vært relevant for analysens validitet, eller at undersøkelsen faktisk måler det den sier at den måler (Frønes & Pettersen, 2021). Et problem med dette ville derimot vært at det krever mer tid både å lage en slik undersøkelse, men også å svare på en slik undersøkelse, og dette ville stått i konflikt med tidsaspektet i undersøkelsen, ettersom hensikten til undersøkelsen var å gi rask tilbakemelding til læreren om hvordan elevene opplevde undervisningen.

6 Diskusjon

Problemstillinga for denne oppgava var «hvilke fagdidaktiske utfordringer kan identifiseres i gjennomføringen av et undervisningsopplegg med programmering i naturfag vg1 og hvilke implikasjoner har dette for undervisning?» Diskusjonsdelen er derfor oppdelt tematisk etter hvilke fagdidaktiske utfordringer som har vist seg å være spesielt relevante gjennom resten av oppgava.

I tillegg til disse utfordringene er det et behov for mer forskning på det meste innenfor temaet. For eksempel vil det være interessant å gjenta dette undervisningsopplegget på andre klasser for at det i større grad skal representere norsk skole. Det kan gjøres kvantitative studier for å se hvor langt skolene faktisk er kommet med innføringen av programmering i naturfag, og man kan jobbe mer med undervisningsopplegg. Innenfor temaet vil det også bli interessant å følge MASCOT-prosjektet ved OsloMet som handler om dette temaet.

6.1 Feilmeldinger og motivasjon

Figur 11 viser en situasjon hvor eleven får en feilmelding og holder på å gi opp. Feilmeldinger er også et sentralt tema i figur 16. I tillegg var det mange andre ganger at elevene fikk feilmeldinger. Noen ganger klarte elevene å løse dem på egenhånd, men mange trengte hjelp. Det var for eksempel flere elever som lurte på om Python ikke skjønte bokstaven «å» fordi feilmeldingen for eksempel sa at «år» ikke var akseptert, men her var det at variabelen heter «antall år» i stedet.

Disse eksemplene viser at det er viktig at elevene får opplæring i hvordan de kan lese feilmeldinger. I intervjuet blir det presisert av lærerne at problemene ofte er at elevene mangler selvstendighet, tålmodighet, samarbeid og utholdenhet, og at når Ola må hjelpe elevene er det fordi de har gitt opp, heller enn at de bare ikke har skjønt det og ikke klarer å komme seg videre.

Fordi problemet ofte er at elevene har gitt opp er det viktig med tålmodighet hos læreren og at læreren viser medfølelse for elevene. Samtidig er det viktig å lære elevene å ikke gi seg så lett. Dette kommer også fram i intervjuet når Ida sier at hun prøver å få elevene til å lese oppgaven igjen dersom de ikke forsto den første gangen.

Et av poengene til Ola i intervjuet var at en fordel med dette undervisningsopplegget er at man ikke var avhengig av å få til programmeringen for å få noe ut av det. Dette gjør at man kan få noe mestringsfølelse selv om man ikke får til all programmeringa, og dette vil være viktig for at elevene ikke skal miste motet, spesielt i en overgangsfase hvor programmering er nytt og ikke alltid gir helt mening.

6.2 Kopieringens rolle i programmering

Situasjonen i figur 15 handler i hvert fall til dels om rollen til kopiering i programmering kontra i andre fag. Svaret fra elev s1 og r2 om at man husker ting bedre når man skriver det selv tror jeg handler om en konflikt mellom rollen til kopiering i naturfag og andre fag generelt og rollen til kopiering i programmering. I pedagogikk generelt er det en vanlig oppfatning at man lærer bedre det man har skrevet ned selv – altså ikke kopiert. I programmering er det derimot helt vanlig å kopiere, og mange programmer bruker kode som andre har laget, for eksempel fra kataloger som numpy som brukt tidligere i denne oppgaven. Dette skaper en konflikt mellom programmeringsdidaktikken og den generelle pedagogikken elevene har brukt tidligere og i dette eksempelet vises dette gjennom at det oppstår en liten konflikt i klasserommet. Jeg vil anta at det kan være andre grunner for konflikten også, men den utløsende faktoren er i hvert fall kopiering.

I programmering er det viktig å kopiere, og kopiering er relevant flere steder i undervisningsopplegget. I en annen situasjon hvor en elev har fått feilmeldinger om syntakstfeil fordi ord er skrevet ulikt påpeker Ida til en elev at ved at man kopierer koden og navnene på variablene slipper man syntaksfeil som at man har skrevet «år» i stedet for antall år eller lignende. I tillegg kan eleven dobbeltklikke på ord i Python-programmet for å se hvor det er skrevet likt i resten av programmet, som kan være relevant for å finne ut hvor det er skrevet feil.

Det var også flere elever som ikke forsto hva de skulle gjøre i den samme oppgaven som i figur 15 og det var tydelig at de ikke skjønte at de skulle kopiere programmet fra forrige side. For at man skal unngå slike konflikter vil det være viktig å påpeke at man godt kan kopiere sitt eget program uten at dette er et problem når man programmer, både for lærere og andre som lager undervisningsopplegg – men selvfølgelig er det ikke innafor å kopiere hele programmer som andre har laget og late som at man har skrevet det helt selv.

6.3 Variabelkontroll i programmeringen

I figur 12 kommer det fram at variabelkontroll er et relevant tema ved programmering i naturfagundervisningen. I naturfag er variabelkontroll et sentralt element innenfor den naturvitenskapelige tenkemåte, det vil si at man passer på at man ikke endrer flere variabler om gangen og at man kontrollerer de andre variablene i for eksempel et forsøk, slik at man kan kontrollere for feilkilder. I intervjuet med lærerne ble det diskutert om det var noen elementer i faget hvor naturfag og programmering hang sammen mer enn at programmering kan brukes som et verktøy for å jobbe med store datasett og modeller, og da ble det oppdaget at variabelkontroll er et slikt element. Dette er spesielt nyttig å trekke fram fordi figur 5 i analysen med TPACK-modellen viser at andelen av 15-sekunderssegmenter som koder for både programmering/teknisk og naturfag/matematikk (TCK) er betydelig lavere enn CPK og TPK. I undervisningen er det vanlige at programmering og naturfag læres hver for seg og deretter i noen eksempler kombineres. Jeg vil også anta at denne effekten er større enn det som vises i søylediagrammet, ettersom i hvert fall halvparten av disse tilfellene er en kombinasjon av det matematiske og det tekniske. Det er rimelig å anta at dersom man får til å kombinere naturfaglige elementer med programmeringen vil det være lettere for elevene å forstå hvorfor de skal bruke programmering i naturfagundervisningen. Programmering kan brukes for å vise viktigheten av variabelkontroll og som en øvelse for elevene i å se sammenhenger mellom uavhengige variabler, det vil si variabelen som endres på, og avhengige variabler, det vil si effekten av variabelen på en annen variabel.

6.4 Grunnleggende programmering

I flere av interaksjonene kommer det fram at elevene mangler den grunnleggende forståelsen av programmering. Dette er ikke rart, siden de akkurat har starta med det, men dette er likevel en kilde til utfordringer. Interaksjonen i figur 16 viser at manglende forståelse for variabler og arrays gjør det krevende å endre på enkle programmer, og elevene trenger en god del støtte for å løse det rent programmeringsfaglige. Dette er også noe det ble snakka om fra både Ida og Ola i både førintervjuet til Ida og hovedintervjuet.

Læreplanen forventer at elevene allerede har lært mye programmering før de kommer til VG1 og at de dermed har gode forutsetninger for eksempel for å kunne lage programmer selv og vurdere disse. Så langt har vi ikke kommet enda i dagens skole, men det kan hende vi når

fram dit om en tid. Dette gjør at man som lærer må bruke mer tid på å gjennomgå det grunnleggende og kanskje senke ambisjonene litt, for å sikre at elevene faktisk følger med.

Til tross for at elevene ikke har så mye programmeringskompetanse enda oppgir mange av elevene at de har jobba godt i selvevalueringen og de viser at de har lært en god del. Begge lærerne var fornøyde med undervisningsøktene, selv om Ola hadde vært ganske skeptisk i forkant. En av Idas kommentarer i intervjuet var at dette måtte de gjøre mer av, og når den grunnleggende forståelsen er mer på plass er det realistisk at elevene får brukt programmeringen i naturfaget konkret og ikke bare bruke tid på å lære programmeringsbiten.

En annen side ved denne utfordringen er at lærere ikke nødvendigvis har programmeringskompetanse. Både Ida og Ola har lært noe i løpet av studiene og det vil være andre skoler hvor ingen eller kanskje bare en enkelt lærer har noe kompetanse med programmering. I denne situasjonen vil ferdigsnekra undervisningsopplegg som det som er brukt i denne oppgaven kunne være til hjelp. På denne måten slipper man utfordringen med å finne temaer som passer spesielt til programmering og man slipper å lage et helt undervisningsopplegg fra bunnen av, som både Ola og Ida mener er noe av det vanskeligste med programmering i naturfag. Det vil likevel være viktig at læreren selv har prøvd ut opplegget og forstår hvordan det fungerer, for å kunne forklare det videre til elever.

6.5 Å tydeliggjøre naturfaget i undervisningen

Fra analysen med TPACK-modellen, figur 5, ser vi at andelen av undervisningen som direkte henger sammen med det naturfaglige er lav. Denne andelen vil antagelig øke når elevene blir flinkere til å programmere, men det er viktig at man som naturfaglærer tydeliggjør hva som er naturfaget i oppgavene, slik at elevene ikke bare gjør oppgaver for å bli ferdig med dem, men at de også ser hvordan dette er knyttet til naturfag og får jobbet med algoritmisk tenkning. Fra CT-analysen (fig. 8) ser vi at rundt 80% av tilfellene hvor algoritmisk tenkning blir observert skjer under segmenter som koder for det naturfaglige og dette tyder ikke overraskende på at det vil være gunstig å fokusere mer på det naturfaglige aspektet i undervisningen for at elevene skal utvikle algoritmisk tenkning.

Dette er i tråd med prinsippet om dybdelæring. Innenfor naturfagundervisningen er det naturlig at man går mest i dybden på det naturfaglige, men det er nødvendig med en viss forståelse også innenfor programmering, slik at man kan se om det for eksempel er

begrensninger i det programmeringsfaglige som også setter begrensninger for det naturfaglige, eller for at elevene skal kunne forstå når det er nyttig å bruke programmering innen naturfag.

6.5.1 Plenumsundervisning

En av de store forskjellene i undervisningen til Ola og Ida var at Ola hadde plenumsundervisning i større deler av timen, ikke bare i innledningen. I plenum tar Ola opp både hvordan tall skrives i Python, men også en samtale om i hvilke situasjoner modellene er realistiske og knytter dette til hvilket miljø bakteriene lever i. Fra TPACK-modellen (fig. 7) ser vi at Ola hadde en mye større andel segmenter som koder for naturfaglig kunnskap i plenum enn i gruppeundervisningen, og resultatene fra CT-modellen kan tyde på at Ola også fikk løftet den algoritmiske tenkingen i plenumsundervisningen. I plenumsundervisningen er det tatt med litt enkel live-programmering for å demonstrere det som blir sagt slik at Ola har noe å peke på, men også for å modellere hvordan elevene kan kode. Dette er likevel ikke et bærende element i undervisningen.

Jeg tolker det slik at Ola bruker plenumsundervisningen som et verktøy for å styre målet med timen mer mot det naturfaglige aspektet i oppgaven ved at elevene faktisk må reflektere over hva modellene deres betyr og ikke bare løse oppgavene. Dette vil i så fall veie positivt for at elevene læres opp i algoritmisk tenkning og knytter programmet til naturfaget. Dette vil også gjøre at Ola får kontrollert at alle elevene faktisk reflekterer over dette og det vil bli lettere å knytte dette inn i undervisningen ved en senere anledning. Plenumsgjennomgangen gjør at alle elevene får med seg de naturfaglige hovedpoengene uavhengig av hvordan selve programmeringen gikk.

Det er i hvert fall en potensiell utfordring med å bruke plenumsundervisning i dette undervisningsopplegget. Elevene på ulike oppgaver og noen av elevene er allerede ferdige med oppgaven det er snakk om, mens andre kanskje bare så vidt har begynt. For at plenumsundervisningen skal være relevant for alle elevene må den enten handle om noe alle elevene lurer på, eller så må den handle om noe som er relevant for alle oppgavene, uansett om du har gjort en konkret oppgave eller ikke. I dette tilfellet vil jeg si at Ola lykkes med dette, fordi alle elevene trenger å ha kontroll på hvorfor Python skriver tall som det gjør og dette hadde ikke de fleste kontroll på før gjennomgangen. Med det naturfaglige temaet får han knyttet programmet til virkelige situasjoner og dette bidrar til å gjøre programmet mer

relevant for naturfaget. Det blir ikke lenger bare en teoretisk modell, men den sier også noe om virkeligheten.

Å bruke plenumsundervisning er likevel en balansegang, fordi det endrer oppgaven fra å være elevstyrt til å i større grad bli mer lærerstyrt. Dette kan fungere i en viss grad innenfor utforskende arbeidsmåter, men dersom det fortsetter for lenge vil ikke elevene få nok spillerom til å gjøre oppgaven til sin egen og man vil miste fordelene med å ha en utforskende oppgave.

Ida hadde også opprinnelig tenkt å ha plenumsundervisning underveis i økta, men ombestemte seg fordi elevene var på så forskjellige steder i oppgaven, og det ble vanskelig å si noe som faktisk var relevant for alle elevene. Vi vet fra tidligere at det er stor spredning i nivået elevene er på innenfor programmering (Haraldsrud et al., 2020) og dette kan gjøre det vanskelig å gjennomføre en plenumssamtale med alle elevene. Dersom man finner samtaleemner som handler om refleksjon over hvordan programmet knytter seg til naturfaget vil dette likevel være nyttig for elevene. Elevene som jobber raskt blir tvunget til å reflektere over det de allerede har gjort, og elevene som jobber tregt får løftet blikket sånn at de ser sammenhenger de kanskje ikke ville sett ellers hvis de bare hadde jobbet på egenhånd. Dette vil også gjøre at alle elevene får med seg det naturfaglige med oppgaven, selv om de kanskje har varierende resultater med selve programmeringen.

Det er rimelig å anta at det finnes andre måter læreren kan sikre seg at elevene får reflektert over oppgavene, men på grunn av den muntlige og visuelle metodikken i denne oppgaven er det naturlig at det er muntlige måter som blir identifisert. I et større kildemateriale vil det også være flere ting som kan oppdages.

7 Oppsummering

I denne masteroppgaven har jeg beskrevet et opplegg jeg har gjennomført i programmering i naturfag i to VG1-klasser i Osloskolen. Undervisningen er analysert, og jeg har identifisert flere utfordringer som oppsto i undervisningen. I andre undervisningssituasjoner kan det være at andre utfordringer oppstår, og det er derfor et behov for flere studier på emnet for å få mer informasjon om hvordan undervisning foregår og flere utfordringer man bør være bevisst på som lærer.

Som nevnt tidligere er dette et lite datamateriale og vil ikke være representativt for alle VG1-klasser, men det vil likevel gi noen pekepinner på hva som er nyttig å tenke på både for lærere som skal bruke programmering i undervisningen i naturfag, men også for forskere og personer som lager undervisningsopplegg. Det vil også være rimelig å anta at det har en viss overføringsverdi til programfagene innenfor naturfagene som biologi og fysikk.

Et viktig funn i denne oppgaven er det er en lav andel av undervisningen som fokuserer på det naturfaglige, men blant annet plenumssamtaler hvor man reflekterer over programmet som lages kan være et viktig verktøy for å øke denne andelen og forsikre seg om at elevene forstår poenget med programmeringen og kan bruke det videre i undervisningen. Dette fungerer selv om elevene er på forskjellige steder i oppgaven, som er en styrke i programmeringsøkter hvor elevene ofte har ulik progresjon. Funnet er viktig fordi det virker til å være en sammenheng mellom andelen av undervisningen som koder for det naturfaglige og antall sekvenser som koder for algoritmisk tenkning og det vil derfor være gunstig at større deler av undervisningen koder for naturfaglig kunnskap.

Et annet funn er at en gjentagende utfordring med programmering generelt er feilkoder og motivasjon. Elevene må bli lært opp til hvordan de leser og tolker feilkoder, slik at de i større grad klarer å løse dem på egen hånd. Dette kan innebære også at man jobber med samarbeid, tålmodighet, selvstendighet og utholdenhet, slik at elevene ikke gir opp så fort. Av denne grunn vil det også være viktig å tenke på elevenes mestringsfølelse i undervisningsopplegget, fordi programmering er nytt for mange elever og kan føles fremmed..

Et uventet funn i oppgaven er at det er en konflikt mellom rollen til kopiering i den generelle pedagogikken og i programmeringsdidaktikk. I den generelle pedagogikken kan det kanskje være mer nyttig å skrive ned hånd i stedet for å kopiere, men innenfor programmering er

kopiering av tekst et verktøy for å forhindre at man skriver feil og får mange syntaksfeil som ofte vil gi feilmeldinger, men i verste fall kan føre til at programmet virker på en annen måte enn det skal. Dette vil det være viktig å følge med på for både lærere og de som lager undervisningsopplegg, slik at dette blir forstått av elevene.

Et fjerde funn viser at det er en tydelig sammenheng mellom programmering og naturfag i temaet variabelkontroll og den naturfaglige tenkemåten, som gjerne kan brukes i større grad. Dette er ekstra viktig fordi andelen av segmenter som koder for både programmering og naturfag er lav og en lav andel kan potensielt føre til at elevene ikke skjønner hvorfor programmering er nyttig i naturfag.

Til slutt er det også tydelig gjennom oppgaven at det samme undervisningsopplegget kan benyttes på ganske forskjellige måter og jeg vil derfor påpeke at kompetansen til lærerne også er veldig viktig når man jobber med programmering. Dette gjelder programmeringsfaglig, altså at lærerne både kan programmere selv, men også at de har kompetanse til å undervise programmering i en naturfaglig kontekst. Her kan man antagelig slippe litt billigere unna ved å benytte et ferdiglagd undervisningsopplegg, men man må likevel kunne bruke opplegget og forklare det til elevene. Hvis man har denne kompetansen kan det være at man får det samme inntrykket som Ida: at man må gjøre det mer, fordi det gikk så bra.

Litteraturliste

Aschehoug univers. (2023, april 2). *Moskuspopulasjonen på Dovrefjell*. Moskuspopulasjonen på Dovrefjell - Aschehoug univers. https://aunivers.no/fagpakker/realfag/naturfag-sf/innhold/programmering/moskuspopulasjonen-paa-dovrefjell

Bjønness, B., Johansen, J. & Byhring, A. K. (2019). Lærerens tilrettelegging av utforskende arbeidsmetoder. I Erik Knain & Stein D. Kolstø (Red.), *Elever som forskere i naturfag* (2. utg.). Universitetsforlaget.

Blikstad-Balas, M. (2017). Key challenges of using video when investigating social practices in education: Contextualization, magnification, and representation. *International Journal of Research & Method in Education*, *40*(5), 511–523. https://doi.org/10.1080/1743727X.2016.1181162

Creswell, J. W., & Miller, D. L. (2000). Determining Validity in Qualitative Inquiry. *Theory Into Practice*, *39*(3), 124–130. https://doi.org/10.1207/s15430421tip3903_2

Falkner, K., Sentance, S., Vivian, R., Barksdale, S., Busuttil, L., Cole, E., Liebe, C., Maiorana, F., McGill, M. M., & Quille, K. (2019). An International Comparison of K-12 Computer Science Education Intended and Enacted Curricula. *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, 1–10. https://doi.org/10.1145/3364510.3364517

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

Gleiss, M. S. (2021). *Forskningsmetode for lærerstudenter: Å utvikle ny kunnskap i forskning og praksis* (1. utgave.). Cappelen Damm akademisk.

Haraldsrud, A. D., Sveinsson, H. A., & Løvold, H. H. (2020). *Programmering i skolen*. Universitetsforlaget.

Haug, B. S. (2014). Inquiry-Based Science: Turning Teachable Moments into Learnable Moments. *Journal of science teacher education*, *1*, 79–96. https://doi.org/10.1007/s10972-013-9375-7

Hurt, T., Greenwald, E., Allan, S., Cannady, M. A., Krakowski, A., Brodsky, L., Collins, M. A., Montgomery, R., & Dorph, R. (2023). The computational thinking for science (CT-S) framework: Operationalizing CT-S for K–12 science education researchers

and educators. *International Journal of STEM Education*, *10*(1), 1. https://doi.org/10.1186/s40594-022-00391-7

Jordan, B., & Henderson, A. (1995). Interaction Analysis: Foundations and Practice. *Journal of the Learning Sciences*, *4*(1), 39–103. https://doi.org/10.1207/s15327809jls0401_2

Kelentrić, M., Helland, Karianne, & Arstorp, Ann-Thérèse. (2017). *Rammeverk for lærerens profesjonsfaglige digitale kompetanse (PfDK)*.

Knain, E. & Kolstø, S. D. (2019). Utforskende arbeidsmåter—En oversikt. I Erik Knain & Stein D. Kolstø (Red.), *Elever som forskere i naturfag* (2. utg.). Universitetsforlaget.

Knain, E., Bjønness, B. & Kolstø, S. D. (2019). Rammer og støttestrukturer i utforskende arbeidsmetoder. I Erik Knain & Stein D. Kolstø (Red.), *Elever som forskere i naturfag* (2. utg.).

Knain, E., Fredlund, T., Furberg, A., Mathiassen, K., Remmen, K. B., & Ødegaard, M. (2017). Representing to learn in science education: Theoretical framework and analytical approaches. *Acta Didactica Norge*, *11*(3), 11. https://doi.org/10.5617/adno.4722

Koehler, M. J., & Mishra, P. (2009). *What Is Technological Pedagogical Content Knowledge? 9*(1).

Kwon, K., Ottenbreit-Leftwich, A. T., Brush, T. A., Jeon, M., & Yan, G. (2021). Integration of problem-based learning in elementary computer science education: Effects on computational thinking and attitudes. *Educational technology research and development*, *69*(5), 2761–2787. https://doi.org/10.1007/s11423-021-10034-3

Ogegbo, A. A., & Ramnarain, U. (2022). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, *58*(2), 203–230. https://doi.org/10.1080/03057267.2021.1963580

Pamuk, S., Ergun, M., Cakir, R., Yilmaz, H. B., & Ayas, C. (2015). Exploring relationships among TPACK components and development of the TPACK instrument. *Education and Information Technologies*, *20*(2), 241–263. https://doi.org/10.1007/s10639-013-9278-4

Patton, M. Q. (1999). Enhancing the quality and credibility of qualitative analysis. *Health Services Research*, *5 pt 2*, 1189–1208.

Ringnes, V. (2014). *Kjemi fagdidaktikk: Kjemi i skolen* (3. utg.). Cappelen Damm akademisk.

Selvaraj, A., Zhang, E., Porter, L., & Soosai Raj, A. G. (2021). Live Coding: A Review of the Literature. *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*, 164–170. https://doi.org/10.1145/3430665.3456382

Sjøberg, S. (2009). *Naturfag som allmenndannelse: En kritisk fagdidaktikk* (3. utg.). Gyldendal akademisk.

Skaalvik, E. M., & Skaalvik, S. (2015). *Motivasjon for læring: Teori og praksis*. Universitetsforl.

Svenkerud, S. W. (2021). Intervjuer i klasseromsforskning. I E. Anderson-Bakken & C. P. Dalland (Red.), *Metoder i klasseromsforskning*. Universitetsforlaget.

Universitetet i Oslo. (2022). *Lagringsguiden*. Universitetet i Oslo. https://www.uio.no/tjenester/it/sikkerhet/lsis/tillegg/lagringsguide.html

Utdanningsdirektoratet. (2019a). *Læreplan i matematikk fellesfag vg1 praktisk (matematikk P) (MAT08-01)*. Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2019. https://www.udir.no/lk20/mat08-01/kompetansemaal-og-vurdering/kv31?lang=nob

Utdanningsdirektoratet. (2019b). *Læreplan i matematikk fellesfag vg1 teoretisk (matematikk T) (MAT09-01)*. Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2019. https://www.udir.no/lk20/mat09-01/kompetansemaal-og-vurdering/kv42?lang=nob

Utdanningsdirektoratet. (2019c). *Læreplan i naturfag (NAT01-04)*. Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2019. https://www.udir.no/lk20/nat01-04/kompetansemaal-og-vurdering/kv77

Utdanningsdirektoratet. (2019d, mars 27). *Algoritmisk tenkning*. https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/

Wing, J. M. (2011). *Research Notebook: Computational Thinking—What and Why?*