Høgskulen på Vestlandet

Matematikk 3, emne 4 - Masteroppgave

MGUMA550-O-2023-VÅR2-FLOWassign

Predefinert informasjon

Startdato:	02-05-2023 09:00 CEST	**Termin:**	2023 VÅR2
Sluttdato:	15-05-2023 14:00 CEST	**Vurderingsform:**	Norsk 6-trinns skala (A-F)
Eksamensform:	Masteroppgave - Bergen		
Flowkode:	203 MGUMA550 1 O 2023 VÅR2		
Intern sensor:	(Anonymisert)		

Deltaker

Kandidatnr.:	234

Informasjon fra deltaker

Antall ord *:	29310

Egenerklæring *: Ja

Jeg bekrefter at jeg har registrert oppgavetittelen på norsk og engelsk i StudentWeb og vet at denne vil stå på vitnemålet mitt *: Ja

Jeg godkjenner avtalen om publisering av masteroppgaven min *

Ja

Er masteroppgaven skrevet som del av et større forskningsprosjekt ved HVL? *

Nei

Er masteroppgaven skrevet ved bedrift/virksomhet i næringsliv eller offentlig sektor? *

Nei

Høgskulen på Vestlandet

MASTEROPPGAVE

Kvaliteter ved programmeringsoppgaver i matematikk

Analyse av programmeringsoppgaver fra to matematikk læreverk på 8. trinn

Qualities of Programming Tasks in Mathematics

Analysis of Programming Tasks from two Mathematics Textbooks on 8th Grade

Magnus Botnen

Masteroppgave i matematikk i Grunnskolelærerutdanning 5-10

Fakultet for lærerutdanning, kultur og idrett

Institutt for språk, litteratur, matematikk og tolking

Veileder: Elena Severina

15. mai 2023

Jeg bekrefter at arbeidet er selvstendig utarbeidet, og at referanser/kildehenvisninger til alle kilder som er brukt i arbeidet er oppgitt, *jf. Forskrift om studium og eksamen ved Høgskulen på Vestlandet, § 12-1.*

Sammendrag

Programmering i matematikk har fått et mye større fokus etter iverksettelsen av LK20. For både erfarne og nyutdannede matematikklærere kan det være krevende å integrere programmering inn i undervisningen på en måte som fremmer læring i matematikk. Læreres begrenset kunnskap om temaet og begrenset forskning på feltet gjør at det er behov for mer forskning om programmering i matematikk. Jeg forsøker igjennom dette studiet å bidra til forskningen ved å studere kvaliteter ved programmeringsoppgaver i matematikk som tilbys av to læreverk på 8. trinn. Kvalitetene som undersøkes er oppgavenes *matematiske temaer, kognitive krav* (Smith & Stein, 1998) og komponenter fra *den algoritmiske tenkeren* modellen (Csizmadia et al., 2015). Ved å være bevist over disse kvaliteter og hvordan de kommer til syne, vil jeg kunne vurdere hvordan ulike programmeringsoppgaver kan brukes i matematikk.

I dette studiet undersøkes *Matemagisk 8* og *Campus Matte 8* for programmeringsoppgaver i matematikk. Utvalget er basert på svarene fra en undersøkelse med 152 svar fra matematikklærere og 40 svar fra ungdomsskoler om hvilket læreverk de bruker til å finne programmeringsoppgaver i matematikk. Med en kvalitativ innholdsanalyse med kvantitative preg og en todelt analyse, inspirert av Charalambous et al. (2010), undersøkes og tolkes læreverkenes bakgrunnsinformasjon og struktur horisontalt, mens matematisk innhold av leksjoner og oppgaver undersøkes vertikalt. Resultatene kvantifiseres og tolkes deretter opp mot teori, tidligere forskning og hver andre.

Studiet viser at 131 programmeringsoppgaver i matematikk fra de to læreverkene kan plasseres under de matematiske temaene *brøk og desimaltall* (21), *algebraiske uttrykk og formler* (66)*, potenser, kvadratrøtter og regnerekkefølge* (4), *parenteser og likninger* (30) og *funksjoner* (10). Deres fordeling av kognitive krav består av flest lave kognitive krav (72 %) som består av *memorering* (31 %) og *prosedyrer uten sammenheng* (41 %). De høye kognitive kravene (28 %) består av *prosedyrer med sammenheng* (25 %) og *gjøre matematikk* (3 %). Nøkkelbegreper fra *den algoritmiske tenkeren* som *logikk*, *mønster* og *algoritmer* identifiseres i 76 til 98 % av alle oppgavene, mens *dekomposisjon*, *abstraksjon* og *evaluering* i 3 til 26 % av dem.

Abstract

Programming in mathematics has been given a more prominent role after the implementation of the new national curriculum in Norway (LK20). For both skilled and newly educated mathematics teachers it can be challenging to integrate programming in the subject which also strengthens learning in mathematics. Teachers limited knowledge about programming and the lack of studies on the topic creates a demand for more studies on programming in mathematics. Through this study I try to supply this demand by studying what different qualities programming tasks in mathematics from 8th grade textbooks have. The qualities that I am searching for are *math subjects*, *cognitive demands* (Smith & Stein, 1998) and traits from *The Computational Thinker* (Csizmadia et al., 2015). By being aware of these qualities and how they work I can evaluate how different programming tasks can be used in mathematics.

In this study programming tasks from *Matemagisk 8* and *Campus Matte 8* are analyzed. The selection of textbooks is based on 152 teacher and 40 school answers from a survey about which textbooks they use to pick out programming tasks in mathematics. With a qualitative content analysis with quantitative characteristics and a two-part analysis, inspired by Charalambous et al. (2010), the textbooks are examined and interpreted for their background information and structure horizontally, while mathematical content of lessons and tasks are examined vertically. The results are quantified before they are interpreted across theory, previous research and each other.

The study shows that the 131 programming tasks in mathematics from the two textbooks can be placed underneath the subject's *fraction and decimal* (21), *algebraic expressions and formulas* (66), *exponents, square roots and order of operations* (4), *parentheses and equations* (30) and *functions* (10). Their distribution of cognitive demands consists mostly of lower-level demands (72 %), which is divided into *memorization* (31 %) and *procedures without connections* (41 %). The higher-level demands (28 %) divides into *procedures with connections* (25 %) and *doing mathematics* (3 %). *Concepts* from *The Computational Thinker* that are often used, from 76 to 98 %, are *logic*, *algorithms* and *patterns*. Lesser used *concepts*, from 3 to 26 %, are *decomposition*, *abstraction* and *evaluation*.

Forord

Innleveringen av denne masteroppgaven symboliserer avslutningen av fem år som lærerstudent på HVL. Disse årene har gitt meg gode minner, mye kunnskap og et utvidet og kritisk syn på forskning innen undervisning. Denne oppgaven har gitt meg en gylden mulighet til å forske på noen av mine interesseområder innenfor undervisning: matematikk og programmering. Det har vært en frustrerende og utrulig lærerik prosess som har gitt meg et nytt syn på hvordan programmeringsoppgaver i matematikk kan brukes for å styrke læringspotensial. Jeg ser fram til å dele min kunnskap med kollegaer og å bruke det i praksis.

Jeg ønsker først og fremst takke min veileder for all hjelpen jeg har fått. Du har gitt meg gode tilbakemeldinger, vist interesse og støttet meg. Videre ønsker jeg å takke mamma. Du har også støttet meg, vert en god kilde til å diskutere dilemmaer jeg har møtt på veien og gitt meg motivasjon til å fortsette. Til slutt en stor takk til min samboer som har satt livet på vent det siste året for meg. Du har ordnet og fikset det meste for at jeg skulle få tid til å arbeide med denne oppgaven. Du har også motivert meg med gode ord og dratt meg med på lufteturer når det har vært kaos i hodet mitt. Tusen takk, jeg er takknemlig og verdsetter deres innsats!

Innholdsfortegnelse

Tabeller

Figurer

1 Introduksjon

Matematikkfaget har fått hovedansvaret for opplæring i programmering og det skaper en utfordring for lærere skriver Johansen (2020). Hun presiserer videre at deres manglende kunnskap om programmering gjør det ambisiøst å integrere det inn i matematikkundervisningen. Odd Tore Kaufmann (u.å., sitert i Johansen, 2020) legger også til at det er få studier som ser på hvilken effekt denne integreringen gir. Børre Stenseth (u.å., sitert i Johansen, 2020) formidler at det er viktig at elevene, spesielt gjennom arbeid med problemløsning, ser nytten i programmering som et verktøy for å forstå matematikk. Dette gjør at koblingen mellom programmering og matematikk i undervisning og dens effekt for matematiskutvikling er et viktig felt som har et behov for å forskes mer på.

1.1 Programmeringens plass i matematikk

Kunnskapsløftet 2020, den nye fagfornyelsen forkortet til LK20, trådde i kraft høsten 2020. Denne innføringen forårsaket noen endringer i grunnskolen og videregående opplæring. Time- og fagfordelingen ble uendret, men alle læreplaner for fag ble fornyet (Utdanningsdirektoratet, 2022). Kunnskapsdepartementet (2016) har gjennom LK20 presisert at skoler må henge med på den teknologiske utviklingen i samfunnet. Det betyr at dersom elevene skal være mest mulig klar for å møte den teknologiske utviklingen i arbeidslivet, må de få tilstrekkelig med digitalkompetanse allerede i grunnskolen. For å øke elevenes digitalkompetanse, ble programmering innført som en del av undervisningen.

Gjennom hele opplæringsløpet skal skoler legge til rette for de grunnleggende ferdighetene i læreplanverket som er *lesing*, *skriving*, *regning*, *muntlige ferdigheter* og *digitale ferdigheter* (Kunnskapsdepartementet, 2020, s. 4-5). I matematikkfaget, omhandler digitale ferdigheter å kunne bruke verktøy for graftegning, regneark, CAS og programmering til å løse og utforske matematiske problemer. Det har blitt mer fokus på programmering når man ser for seg framtidens arbeidsmarked (Sevik et al., 2016, s. 11). Sevik et al. (2016, s. 11) fra *Senteret for IKT i utdanning* beskriver at teknologisk kompetanse er nødvendig i arbeidslivet alle vil møte i fremtiden. Selv om programmering ofte knyttes opp mot utvikling av ny teknologi, så mener de at det er like viktig for å kunne forstå hvordan teknologien fungerer, bruke det til å løse utfordringer og ikke minst for å bruke teknologisk utstyr. I sammenheng med dette ser

man på programmering som en av kompetansene for det 21. århundre (Sevik et al., 2016, s. 10). Programmering er derfor ikke bare en egenskap som kan være nyttig å lære seg i arbeid med matematikk, men også for å være rustet for hva framtiden vil bringe.

Aktiviteten *programmering* innebærer å dele opp komplekse problemer i mindre deler og å gi presis forklaring av løsningens fremgangsmåte (Statped, 2021). Sande (u.å., s. 3) ser på aktiviteten som å planlegge og designe, skrive, teste, feilsøke og forbedre kildekode til et dataprogram. *Programmering* er ikke nytt innenfor matematikken. I 1980 ble det engelske begrepet *computational thinking* (CT) introdusert av Saymour Papert. Han utviklet det første programmeringsspråket for barn kalt *LOGO* (Papert, 1980). Innvirkningen av ideen til Papert varte ikke lenge, Kotsopoulos et al. (2017, s. 156) forklarer at den digitale utvikling i verden rundt oss var så stor at Papert sin ide ble skyggelagt av den enorme utviklingen. Flere år senere i 2006 blir begrepet CT reintrodusert av Wing (2006). Et annet ord som ofte brukes i sammenheng med *programmering*, og kan ikke brukes som et synonym, er *koding*.

Koding er en del av *programmering* der instruksjoner gis til en datamaskin som utfører en handling (Statped, 2021). Selve prosessen i *koding* utføres når man skriver programmet gjennom programmeringsaktiviteten å *skrive*. Det handler ikke om aktiviteter fra programmering som å planlegge og designe, teste, feilsøke og forbedre fra Sande (u.å., s. 3). Å kode et program kan gjøres enten via tekst eller blokker (Statped, 2021). Å kode med tekst omtales som tekstprogrammering. Her skrives det linjer med koder basert på bokstaver, tall og symboler. *Python* er et eksempel på et programmeringsspråk i *tekstprogrammering*. Ulike programmeringsspråk skrives med egne sett av språkregler og bruksområder som egner seg til ulike formål som å lage nettsider, dataprogram eller dataspill i forhold til andre. Å kode via blokker omtales som *blokkprogrammering*. Gjennom denne metoden kan man lage program ved å plassere ulike blokker med forhandsproduserte koder oppå hverandre i logisk rekkefølge, som leses fra toppen av og ned (Statped, 2021).

Implementering av LK20 medførte nye kompetansemål i læreplan for matematikk fra 8. til 10. trinn. Tre av de nye kompetansemålene i MAT01-05 for 8.-10. handler eksplisitt om programmering i matematikk:

Kompetansemål etter 8. trinn:

- utforske hvordan algoritmer kan skapes, testes og forbedres ved hjelp av programmering.

Kompetansemål etter 9. trinn:

- simulere utfall i tilfeldige forsøk og beregne sannsynligheten for at noe skal inntreffe, ved å bruke programmering.

Kompetansemål etter 10. trinn:

- utforske matematiske egenskaper og sammenhenger ved å bruke programmering (Kunnskapsdepartementet, 2020).

Her ser vi at kompetansemålene for 8. og 10. trinn er knyttet opp mot utforsking av algoritmer og matematiske egenskaper gjennom programmering. For 9. trinn derimot, er målet mer spesifikt rettet mot matematikken bak sannsynlighet. Likevel vil det å simulere utfall i sannsynlighetsregning gjennom bruk av programmering kreve kompetanse om hvordan man bruker programmeringsprogram og være med på å utforske tilfeldige forsøk. Fra et progresjonsperspektiv, viser kompetansemålene at det er et behov å arbeide med programmering som utforsker matematiske sammenhenger gjennom hele ungdomstrinnet.

Det som utgir selve kjernen av opplæringen elevene skal gjennom i matematikkfaget på grunnskolen, er de seks kjerneelementene: *utforsking og problemløsing*, *modellering og anvendelser*, *resonnering og argumentasjon*, *representasjon og kommunikasjon*, *abstraksjon og generalisering* og *matematiske kunnskapsområder*. Utdanningsdirektoratet (2019b) påpeker at elevene må læres i de gitte kjerneelementene for å kunne bruke og mestre faget slik det er lagt opp i læreplanen i dag. Disse elementene består av flere sentrale begreper, tenkemåter, metoder, uttrykksformer og kunnskapsområder og påvirker progresjon og innhold i læreplanen. Med denne oppbyggingen av læreplanen ønsker Utdanningsdirektoratet (2019b) at elever får fordypet kunnskap av innhold og sammenhenger i faget over tid. Kompetansemålene som er rettet direkte mot programmering i matematikk, vist tidligere, fremhever rollen av utforsking og er tett knyttet til problemløsning. Dette gjør at en av kvalitetene ved programmeringsoppgaver i matematikk handler om hvor godt en oppgave er forankret i kjerneelementet *utforsking og problemløsning.*

Kjerneelementet *utforsking og problemløsning* er sammensatt av to deler. Kunnskapsdepartementet (2020, s. 2) beskriver utforsking som elevenes arbeid i å finne sammenhenger og mønstre i matematikk, og å skape en felles forståelse gjennom diskusjon. Noe av det som ønskes her, er at elevene heller oppsøker forståelse for fremgangsmetoden og strategier i staden for å søke etter løsningen. Problemløsning handler om å skape en løsningsmetode for et problem de ikke kjenner fra før gjennom analyse, omforming av ukjente og kjente problem. Selve løsningen og vurdering av dens gyldighet er også en del av problemløsning (Kunnskapsdepartementet, 2020, s. 2). Det innebærer også å utvikle strategier og framgangsmåter og å løse problemer systematisk ved å dele de opp i delproblemer gjennom algoritmisk tenkning.

LK20 er ny og implementeringen støttes gjennom fornyelse av læreverk. Selv om programmering på skolen ikke er en helt ny ide, er den ny i denne fagfornyelsen. Lærere med mindre erfaring i programmering, inkludert de nyutdannede lærerne, kommer til å støtte seg på tilgjengelige ressurser og læreverk.

1.2 Forskning på læreverk

Matematikkundervisningen blir ofte strukturert basert på læreverks innhold og er derfor et viktig felt å forske på (Jones & Tarr, 2007, s. 6). Charalambous et al. (2010, s. 118) presiserer at forskere har ulike synspunkt når det gjelder hvilken effekt slike studier har. Blant annet at det kan forklare forskjellen mellom elevers resultater i faget på tvers av land og bidra til å utvikle læreres egen kompetanse. Det har blitt gjennomført flere studier som ser på innhold og oppgaver fra hele læreverk eller spesifikke matematiske temaer fra dem. Blant disse er det få som fokuserer på programmeringsoppgaver i matematikk. Dette kan skyldes at igangsettelsen av LK20 i 2020 har ført til at nye læreverk ikke har vert publisert lenge nok.

Charalambous et al. (2010) gjennomførte selv et forskningsprosjekt hvor de undersøkte brøkoppgavers kognitive krav i matematikklæreverk brukt i Kypros, Irland og Taiwan for 1. – 6. trinn. Undersøkelsen ble gjennomført i fem læreverk hvor to av dem var fra Irland, to fra Taiwan og en fra Kypros. I alle læreverkene fra Irland og Kypros, ble over 85 % av oppgavene klassifisert som lavt kognitivt krevende, de taiwanske læreverkene inneholdt derimot rundt

71 % oppgaver av høye kognitive krav (Charalambous et al., 2010, s. 138-139). Resultatene av kognitive krav gir en beskrivelse av hvilke oppgavetyper læreverk fra ulike land legger vekt på. Deres resultater av studiet forteller at læreverk fra Irland og Kypros tilbyr oppgaver som legger mest vekt på pugging og bruk av algoritmer, mens i Taiwan er det mer vektlegging av utforsking av matematikk og strategier.

Det har også blitt skrevet masteroppgaver med lignende forskning. Blant annet gjennomførte Heimstad og Strand (2018) en studie om kognitive utfordringer ved oppgaver i læreverkene *Faktor* og *Maximum* for ungdomstrinnet. Deres resultater viser til at det var en fordeling på 73,7 % lavt og 26,3 % høyt kognitivt krevende oppgaver i begge læreverkene uavhengig av matematisk tema (Heimstad & Strand, 2018, s. 95). Hofstad og Liland (2021) gjennomførte en studie hvor de undersøkte algebraoppgaver i to læreverk for 8. trinn. En del av deres undersøkelse var også å se etter oppgavenes kognitive krav. Funnene deres om fordelingen av kognitive krav i læreverkene var nokså ulike for læreverkene. I *Matemagisk 8* inneholdt 55 % av algebraoppgavene lave kognitive krav, mens i *Matematikk 8* inneholdt slike oppgaver om lag 79 % lave kognitive krav (Hofstad & Liland, 2021, s. 58-59).

De fleste av studiene jeg finner bruker Smith og Stein (1998) til å beskrive matematikkoppgavers kognitive krav hvor datamaterialet består av oppgaver fra ulike matematiske temaer. I dette studiet prøves det å måle kognitive krav av programmeringsoppgaver i matematikk. Dette krevde noen tilpassinger og justeringer for å tilfredsstille matematiskaktivitet gjennom programmering. Studiet bidrar derfor med å utforske flere bruksområder av rammeverket til Smith og Stein (1998). Heimstad og Strand (2018) bruker, i likhet med mitt studie og flere andre, et analytisk rammeverk inspirert av Charalambous et al. (2010). Dette viser at deres rammeverk som i utgangspunktet var utviklet for å analysere brøkoppgaver i læreverk, kan brukes til å undersøke oppgaver i andre matematiske temaer.

1.3 Problemstilling

Hvordan er programmeringsoppgaver i matematikk satt sammen? Hvilke av deres kvaliteter vil kunne tilføre mer utforsking av matematisk innhold? Å ha denne kunnskapen kan tenkes

å være viktig for å kunne være kritisk og bevist på valg av hvilke programmeringsoppgaver elever skal arbeide med som også støtter læring som fremmer matematikk.

Gjennom mitt studium vil jeg bidra til bedre kjennskap med hva de fornyede læreverkene har å tilby for arbeid med programmering i matematikkfaget: hva programmeringsoppgaver i matematikk inneholder, hvilken matematikk som er berørt og hvordan det legges til rette for utforsking av matematikk. Å studere programmeringsoppgavene som er gitt i matematikklæreverk, fornyet til LK20, på denne måten bidrar til forskningsfeltet ved å belyse hvilke muligheter programmering gir for utvikling av forståelse i matematikk og utvide vår forståelse om sammenhengen mellom programmering og matematikk. Masterstudiets rammer begrenser mengden av oppgaver som kan studeres. Studiet har ikke formål om å sammenligne læreverk, men å se på kvaliteter ved oppgaver som er tilgjengelige. Det velges derfor to ulike læreverk hvor oppgavene studeres som et samlet utvalg. Med en slik begrensning, stilles problemstillingen slik:

Hvilke kvaliteter har programmeringsoppgaver i matematikk på tvers av to læreverk?

Programmeringsoppgaver i matematikk blir videre henvist til med forkortelsen POiM. Slike oppgaver innebærer at både matematikk og programmering blir ivaretatt. Programmeringsoppgaver som ikke innebærer matematikk identifiseres derfor ikke som POiM. Krav som må utfylles for at en programmeringsoppgave indentifiseres som en POiM utdypes i kapittel 3.3.1.1. Begrepet *kvaliteter* i problemstillingen refererer til ulike egenskaper POiM kan ha, mens i dette studiet fokuserer jeg kun på tre av disse, nemlig: *matematiske temaer*, *kognitive krav* og *algoritmisk tenkning*. Kvaliteten *matematiske temaer* knyttes opp mot hvilket tema i matematikken som POiM er plassert under. Dette vil kunne si noe om hvor programmering i forbindelse med matematikk er plassert. Begrepet *kognitive krav* forstås i tråd med Smith og Stein (1998, s. 345) beskrivelse av hvilke type tankevirksomhet som kreves av elevene for å løse en oppgave i matematikk. Ved å se på hvilke kognitive krav POiM krever, vil man kunne se om læreverk vektlegger repetisjon og trening på løsningsstrategier eller utforsking av matematiske sammenhenger og strategier. Begrepet, sammen med ulike typer av kognitive krav, er utdypet i kapittel 2.2. *Algoritmisk tenkning* blir i dette studiet forstått med utgangspunkt i (Utdanningsdirektoratet, 2019a, s.

1) definisjon, hvor begrepet sees på som en problemløsningsmetode som bruker teknologisk kompetanse til å løse deler eller hele problemet. I kapittel 2.5 vises hvilke elementer som kan gjenkjennes i POiM fra *den algoritmiske tenkeren* og knyttes opp mot hvilke steg av *problemløsningsprosessen* som brukes.

I mitt studium valgte jeg å prioritere tre av flere mulige kvaliteter ved POiM, og dette resulterte i formulering av følgende forskningsspørsmålene:

1. Hvilke matematiske temaer er POiM plassert under?
2. Hvilke kognitive krav stiller formulering av POiM til elever?
3. Hvilke nøkkelbegrep for algoritmisk tenkning kan man gjenkjenne gjennom arbeid med POiM?

Begrepet *læreverk* i dette studiet omtaler alle ressurser som er knyttet opp mot hverandre i et fag som tilbys av et forlag. Et eksempel er det matematiske læreverket *Matemagisk 8* fra forlaget *Aschehoug*. Det inneholder en grunnbok, parallellbok, elevhåndbok, digitale ressurser og lærerveiledning for 8. trinn som til sammen går under felles navnet *Matemagisk 8*. For å beholde studiet innenfor masteroppgavens omfang, måtte jeg begrense meg til analyse av to læreverk. Hvis studiet handlet om to læreverk på forskjellige trinn av samme forlag, skulle resultatene være noe snevret da de viste kun ett av perspektivene som lærere blir presentert med. Jeg ville få forståelse for kvaliteter ved POiM uavhengig av utgiver/forlag, derfor valgte jeg å velge to læreverk som er mest brukt av lærere på 8. trinn (se kapittel 3.1 for mer informasjon). Det eksisterer både analoge og digitale læreverk i matematikk på dette trinnet, derfor forventet jeg behov for at studiet måtte ta høyde for begge former.

For å besvare problemstillingen legges det fram teori rundt både utforsking og problemløsning i matematikk sammen med teori for kognitive krav og algoritmisk tenkning, knytt til rammeverket i neste kapittel. I Kapittel 3 belyses metodiske valg og det analytiske rammeverket tilpasses studiet ved hjelp av den relevante teorien. Her synliggjøres også utvalget av læreverk til studiet. Resultatene fra analysen presenteres gjennom kapittel 4 før

de diskuteres opp mot hverandre, teori og tidligere forskning i kapittel 5. Avslutningsvis trekker jeg frem problemstillingen med forskningsspørsmål igjen og gir en konklusjon.

2 Teori

Gjennom dette kapittelet presenteres først teori som viser til betydningen av å arbeide med utforskendeaktivitet i matematikk. Ulike kognitive krav som matematikkoppgaver har, blir utdypet og gitt eksempler til. Deretter bygges det opp med teori for problemløsning og problemløsningsprosessen for å utdype algoritmisk tenkning.

2.1 Utforsking i matematikk

Nøkkelen til å utvikle bredere kompetanse i matematikk etter dagens samfunn er utforsking (Opheim & Simensen, 2020, s. 102). Dersom elevene er aktive i problemløsningsprosessen og må argumentere og begrunne sine valg og løsninger er det mer enn regneferdigheter som utvikles. Arbeid med utforsking vil utvikle rike erfaringer og begreper i matematikk. Kieran (2013) argumenterer for viktigheten av å balansere utforsking og prosedyreorientert tilnærming, som inneholder struktur og system, på hver sin side. Ved å se på disse sidene hver for seg og legge til rette for dem, vil elevene kunne utvikle dybdekunnskap i matematikk fordi tilnærmingene understøtter og komplementerer hverandre. Opheim og Simensen (2020, s. 107) poengterer at tilrettelegging av disse tilnærmingene skaper rom for utvikling av matematisk kunnskap og at man ikke kan trene på utforsking uten å trene prosedyreorienterte ferdigheter.

2.2 Kognitive krav for matematikkoppgaver

Begrepet *kognitiv* omfatter aktiviteter som innebærer å gjenkalle og huske kunnskap, tenke, skape og problemløse (Bloom et al., 1956, s. 2). I kontekst med matematikkoppgaver så beskriver *kognitive krav* hvilke prosesser elever bruker i arbeidet. Silver og Stein (1996) gjennomførte en undersøkelse, *The QUASAR Project,* for å finne ut av hva som gjorde en forskjell for matematikken elever lærer i et klasserom. Etter fem år med datainnsamling fant de noe som kunne vesentlig prege matematisk forståelse. Det var at elever som fikk høyest poeng ut fra en kartleggingsprøve i matematikk, var de som arbeidet med matematikk gjennom oppgaver med høye kognitive krav (Silver & Stein, 1996, s. 501-504). I sammenheng med dette funnet lagde Smith og Stein (1998) et rammeverk, kalt for *The task-analysis guide* på engelsk, for å beskrive nivået av kognitive krav en matematikkoppgave kan kreve for å gjennomføres. Rammeverket består av fire ulike typer kognitive krav som deles inn i lave og

høye kognitive krav. De blir presentert på norsk med min oversettelse og engelsk navn i parentes. Den første kategorien er lave kognitive krav som består av *memorering* (memorization) og *prosedyrer uten sammenhenger* (procedures without connections to concepts or meaning). Høye kognitive krav består av *prosedyrer med sammenhenger* (procedures with connections to concepts or meaning) og *gjøre matematikk* (doing mathematics) (Smith & Stein, 1998). Med utgangspunktet i Smith og Stein (1998) sine kognitive krav, har Valenta (2016) for hver av dem beskrevet kjennetegn og gitt eksempler.

2.2.1 Memorering

Memorering består av oppgaver av det laveste kognitive kravet av Smith og Stein (1998) og kjennetegnes sammen med Valenta (2016) som:

- Oppgaver som bruker fakta, formler, regler eller definisjoner man allerede har lært for å løses.
- Oppgaver som ikke kan løsest med strategier, fordi det enten ikke eksisterer en strategi for slike oppgaver eller oppgaven kan løsest raskere uten å en.
- Oppgaver som krever reproduksjon av tidligere materiale eller oppgaver man har jobbet med.
- Oppgaver som ikke skaper noe sammenheng mellom det matematiske konseptet eller ideer gitt i formler, regler eller definisjoner (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 3).

«Hvor mange cm er det i en meter?» (Valenta, 2016, s. 3) blir gitt som et eksempel for en *memorerings*-oppgave. Oppgaven handler om å svare på et spørsmål ved bruk av fakta. Hensikten er ofte å svare basert på memorering og reproduksjon av informasjon (Valenta, 2016, s. 3). Det er kun et svar uten forklaring som kreves. Målet er å memorere hvor mange cm det er i en meter. Det kreves ikke en strategi eller skapes ikke sammenheng mellom hvorfor det er 100 cm i en meter. Denne oppgaven kan dermed kjennetegnes etter kravene som en *memorerings*-oppgaver.

2.2.2 Prosedyrer uten sammenhenger

Prosedyrer uten sammenhenger, er den andre kategorien fra Smith og Stein (1998) med lavt kognitivt nivå. Smith og Stein (1998) og Valenta (2016) kjennetegner slike oppgaver som:

- Algoritmiske og viser enten eksplisitt at det kreves en strategi eller implisitt med bakgrunn i tidligere erfaringer.
- Oppgaver hvor det kreves begrenset kognitiv aktivitet for å finne svaret, men kan i noen tilfeller kreve litt mer aktivitet.
- Oppgaver som ikke skaper sammenheng mellom matematiske konsepter eller meninger, men setter søkelys på å finne det korrekte svaret istedenfor matematisk forståelse (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 4).

«Arne har 4 klistremerker, så får hun 3 til fra sin bestemor. Hvor mange har hun nå?» (Valenta, 2016, s. 4) er et eksempel på en oppgave med det kognitive kravet *prosedyrer uten sammenhenger*, fordi denne oppgaven krever at elevene har en løsningsstrategi basert på lignende oppgaver, som å finne tallene og nøkkelord som sier hvilke regneoperasjon som skal gjøres (Valenta, 2016, s. 4). I forhold til kjennetegnene fra forrige avsnitt, kreves det implisitt en strategi, elevene utfordrer ikke matematisk forståelse rundt hvorfor strategien er gyldig. Oppgaven kan være mer krevende for yngre elever som ikke er kjent med slike oppgaver. Den kan derfor i noen tilfeller klassifiseres som en oppgave av et høyere *kognitivt krav* dersom det er deres første møte med slike oppgaver (Valenta, 2016, s. 4).

2.2.3 Prosedyrer med sammenhenger

Den første kategorien med et høyere nivå av kognitivt krav er *prosedyrer med sammenhenger.* Slike oppgaver beskrives av Smith og Stein (1998) og Valenta (2016) som:

- Oppgaver der elevers oppmerksomhet blir flyttet fra å finne løsningen til å utvikle en dypere forståelse av matematiske konsepter og ideer.
- Hinter enten implisitt eller eksplisitt om hvilken strategi man bør bruke. Målet er mer fokusert på å få elever til å utforske underliggende matematiske konsepter istedenfor å bruke algoritmer.

- Ofte presenteres med ulike representasjoner, slik som grafiske fremstilling av diagram, tabeller, symboler og situasjoner.
- Oppgaver som kan kreve både høy og i noen grad kognitivt nivå. Den kan løsest delvis ved å følge en generell strategi, men vil deretter kreve at eleven utforsker de underliggende matematiske ideen fra strategien for å finne løsningen (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 5).

«Hvis vi tenker på $3 \cdot 17$ som 3 bunker med 17 klosser i hver bunke, hvordan kan vi da forklare at $3 \cdot 17 = 3 \cdot 10 + 3 \cdot 7$?» (Valenta, 2016, s. 5) eksemplifiserer *prosedyrer med sammenhenger*-oppgaver. I denne oppgaven får vi representert multiplikasjon gjennom symbol og regnefortelling. En strategi av multiplikasjon blir fremhevet, men oppgaven forsøker å skape utvikling for forståelse av strategien gjennom resonnering (Valenta, 2016, s. 5). De tidligere listet opp kjennetegnene gjenkjennes i oppgaven ved at den presenteres gjennom flere representasjoner, hinter til multiplikasjon som strategi og forsøker å utforske underliggende matematiske konsepter og ideer.

2.2.4 Gjøre matematikk

Gjøre matematikk er Smith og Stein (1998) sin siste kategori og omfavner de oppgavene med det høyeste kognitive nivået. Smith og Stein (1998) og Valenta (2016) kjennetegner dette kognitive kravet som:

- Oppgaver som krever at eleven forstår og utforsker matematiske fenomen, strategier og sammenhenger på et komplekst nivå.
- Ingen introduksjon eller retning til løsningen blir eksplisitt gitt og elevene kan ikke bruke algoritmisk tilnærming som den eneste prosessen, men de må utforske og bruke elementer fra algoritmene til å skape en løsning.
- Oppgaver som vil kreve at elevene må tenke over egen utvikling gjennom metakognisjon.
- Elevene må bruke tid på å analysere oppgaven for å finne hva som er relevant informasjon og for å begrense mulige løsningsstrategier.

- Elevene må bruke tidligere kunnskap og erfaringer sammen med ny kunnskap for å skape mening mellom matematiske ideer og konsepter gjennom arbeidet med oppgaven (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 7).

Oppgaven «hvis alle skal klemme hverandre i vår klasse, hvor mange klemmer blir det totalt?» (Valenta, 2016, s. 7) er et eksempel på en oppgave fra det kognitive kravet *gjøre matematikk*. Gjennom denne oppgaven må elevene selv definere rammer, finne ut av relevant informasjon og lage strategier (Valenta, 2016, s. 7). I forhold til kjennetegnene fra forrige avsnitt må elevene lage egne strategier og utforske dem, samtidig som de må hente inn informasjon som trengs for å løse oppgaven og bruke dette til å skape mening mellom løsningsstrategi og problemet.

2.3 Problemløsning i matematikk

Det er flere som ser på problemløsning som kjernen i matematikk (Hana, 2014, s. 205). Her er det fundamentalt å kunne bruke ulike elementer fra matematikken til å skape mening til og løse nye problemer. Noe av det som kan skape forvirring i diskusjonen rundt problemløsning, er hvordan man skal skille *problem* og *oppgaver* fra hverandre. Hana (2014, s. 205) skiller ordene ved å definere *problem* som en oppgave der den som løser problemet må bruke mer enn eksisterende kunnskap for å finne en løsning. *Oppgaver* derimot defineres som *rutineoppgaver*, altså en oppgave der personen må gjøre noen rutiner for å finne løsningen til oppgaven. Memorerings-oppgaver kan settes opp mot Hana (2014) sin definisjon av *rutineoppgaver*, fordi man må bruke rutiner basert på det man allerede vet for å finne løsningen. Det største skillet her er at det er avhengig av hvilke eksisterende kunnskaper problemløseren har fra før, som vil kategorisere om det er et problem eller oppgave personen har framfor seg. Torkildsen (2017, s. 2) poengterer at arbeid med problemløsningsaktiviteter vil kunne bidra med å fremme elevenes forståelse og læring i matematikk. Gjennom denne aktiviteten får elevene utviklet helhetlig matematisk kompetanse som bidrar til økt dybdelæring.

Johnson et al. (2018) definerer problemløsning som å vite hva man skal gjøre når man ikke vet hva man skal gjøre. Dette alene kan høres veldig svevende ut, men det blir utdypet at

problemløseren må først forsøke å forstå og skape mening til problemet før man prøver noe nytt får å finne en løsning. I flere av tilfellene vil man feile, men man kan lære fra resultatene av dette og forsøke på nytt (Johnson et al., 2018). De to definisjonene for *problem* blir fremstilt ulikt, men det som er felles for dem er at det må skje en prosess hvor problemløseren selv må forsøke å finne ut hva løsningen er. Torkildsen (2017) gir et eksempel på hva som kan være en problemløsningsoppgave for noen og rutineoppgaver for andre. «Etter at Mattias hadde tatt ¼ av kakene i boksen, var det 18 kaker igjen. Hvor mange kaker hadde det vært i boksen?» (Torkildsen, 2017, s. 3). Dette er et eksempel hvor yngre elever muligens må tenke mye over problemet for å komme fram til en løsning. En slik oppgave for disse elevene ville kunne kategorisert som en problemløsningsoppgave, dersom de måtte finne løsningen via strategier som ikke innebærer å regne det ut ved hjelp av brøk rutiner. For eldre elever, som antakeligvis har en bredere forståelse av brøk, vil en slik oppgave i flere tilfeller klassifiseres som en rutineoppgave.

Definisjonene av hva som er problemløsning gjør det vanskelig å kunne stemple en oppgave som en problemløsningsoppgave. De kognitive aktivitetene som kreves for å løse slike oppgaver vil kunne samsvare med Smith og Stein (1998) sine krav for *gjøre matematikk*. Det betyr ikke at alle oppgaver med dette kravet kan klassifiseres som problemløsningsoppgaver, men at problemløsningsoppgaver kan plasseres under *gjøre matematikk*. For at elevene skal lettere kunne arbeide med slike oppgaver, vil det være viktig å trene på de ulike prosessene som problemløsning innebærer.

2.4 Problemløsningsprosessen

Det finnes flere tolkninger av prosessen for problemløsning. Woods (2000, s. 443) skriver at det finnes mer en 150 problemløsningsmodeller på tvers av fagområder. I matematikk er flere av dem ulike variasjoner av Polya (2014) sin modell (Woods, 2000, s. 443). I dette kapittelet presenteres Polya (2014) sin problemløsningsmodell. Jeg har også valgt Schoenfeld (2014) som en tilfeldig alternativ modell av problemløsning for å vise sammenheng mellom dem.

2.4.1 Polya (2014) sin problemløsningsmodell

Polya (2014) var en ungarsk matematiker som gjennom boken «*How to solve it*» fra 1945, lagde en problemløsningsmodell bestående av fire faser. Disse viser oss hans tolkning på hvordan man finner løsningen av et problem. Prosessen består av *å forstå problemet*, *lage en plan*, *gjennomføre planen* og *å se tilbake*. De fire trinnene beskriver de ulike synspunktene vi har på et problem som forandrer seg gjennom løsningsprosessen.

Å forstå problemet er det første trinnet man går igjennom i møtet med problemet. For at elevene skal forstå problemet, må de analysere problemstillingen og finne de ulike elementene problemet gir dem (Polya, 2014). Noe av det Polya (2014) mener de kan se etter, er problemets ukjente, hvilket datamateriale som er tilgjengelig og hvilket krav problemstillinger stiller. Noe han fremmer som viktig er at problemet bør være godt formulert dersom elevene skal kunne analysere hvilken informasjon problemstillinger har tilgjengelig. Dersom oppgavens formulering er god og elevene klarer å analysere informasjonen og skille dem fra hverandre, vil de kunne danne et skille mellom hva som er relevante og ikke relevante opplysninger for problemet (Polya, 2014). Når elevene har forstått problemet beveger de seg inn i trinn to, *å lage en plan*. Planen kan både være fullstendig og en ide av hvilke utregninger som er nødvendig for å kunne finne en løsning (Polya, 2014). Ofte i denne fasen er det elevens tidligere erfaringer og matematikk kunnskap som danner planen. Elevene vil prøve å kjenne igjen problemet gjennom tidligere problem de har løst og bruke elementer derifra (Polya, 2014). Polya (2014) mener at elevene kan med hjelp av dette forsøke å se sammenhengen mellom problemstillingen og tidligere erfaringer for å danne en plan. Trinn tre i prosessen er *å gjennomføre planen*. Her er det opp til elevene å gjøre de matematiske utregningene korrekt og sørge for at de er utført matematisk korrekt. Det siste trinnet er *å se tilbake*. For at elevene skal få mest læring gjennom problemløsning er det viktig at de ser tilbake på problemet og hele prosessen for å vurdere validiteten til strategien, argumentasjonen, matematikken de har brukt og løsningen (Polya, 2014). Gjennom refleksjon av alle trinnene vil de kunne videre utvikle sin matematiske forståelse og muligens se andre løsningsstrategier (Polya, 2014).

2.4.2 Schoenfeld (2016) sin problemløsningsmodell

Schoenfeld (2016) utviklet en ny problemløsningsmodell med seks faser inspirert av Polya (2014). Denne ble utviklet ved å observere matematikere og studenter gjennomføre problemløsningsoppgaver. Ved hjelp av observasjonene, noterte han ned hvor mye tid de brukte i ulike faser gjennom løsningsprosessen. De ulike fasene fra observasjonen danner modellen hans: *å lese*, *analysere*, *utforske*, *planlegge*, *iverksette* og *å verifisere*. Schoenfeld grunngir ikke definisjonen av hver fase, men siden den er inspirert av Polya (2014) kan den sammenlignes med hans modell. Hovedforskjellen mellom de to modellene er at Schoenfeld har to flere faser slik at det kunne måle tiden av flere elementer av løsningsprosessen. Beskrivelsen av Polya (2014) første trinn var *å forstå problemet* innebærer Schoenfeld (2016) sine tre første faser; *å lese*, *analysere* og *utforske*. Videre kan *å lage en plan* og *planlegge*, *å gjennomføre planen* og *iverksette* og *å se tilbake* og *verifisere* være identiske. En forskjell er at Schoenfeld sin modell ikke var bundet til å være stegvis, her kan man bevege seg mellom stegene gjennom hele prosessen (Schoenfeld, 2016). Etter undersøkelsen ser Schoenfeld (2016) at det er viktig for problemløseren å forstå hvorfor man gjør slike oppgaver. Studenter som ikke var trent til denne type oppgaver leste oppgaveteksten og valgte en strategi for å forsøke å løse den uten mål og mening. Matematikere brukte mesteparten av tiden til å analyser og utforske problemstillingen for å gi den mening før de begynte på løsningen (Schoenfeld, 2016). Etter å ha trent på å gi mening til problemstillingen gjennom metakognitive aktiviteter, vil man kunne utvikle seg til å takle slike problemoppgaver bedre (Schoenfeld, 2016). Som et resultat av å trene elever eksplisitt i metakognisjon, viser elevene en mye mer analytisk og dynamisk tilnærming gjennom løsningsprosessen i forhold til elevene uten opplæring (Schoenfeld, 2016).

De to problemløsningsmodellene viser at selv om de forskjellige modeller har ulike faser, er prosessen nokså universal for problemløsning. Man må først og fremst forstå problemet for å kunne lage en plan og utforske den. Når planen er strukturert kan man gjennomføre den og så evaluere om den er gyldig eller om det kreves noen forbedringer av planen. Noe som er ulikt, etter mine tolkninger av problemløsningsmodellene, er at Polya (2014) følger stegene i logisk rekkefølge med mulighet til å gå tilbake i siste steg. Schoenfeld (2014) derimot, tydeliggjør at man kan bevege seg mellom de ulike stegene gjennom hele arbeidet med oppgaven. Dette studiet bruker Polya (2014) som utgangspunkt for

problemløsningsprosessen. Schoenfeld (2016) sin modell ble brukt til å kartlegge hvor i løsningsprosessen forskningsobjektet befant seg til ulike tider gjennom observasjon. I dette studiet er det de gitte oppgavene som må tilrettelegge for løsningsprosessen, derfor er det mer realistisk å oppdage Polya (2014) sine steg da jeg ikke kan observere hvordan elever løser oppgavene i min forskningsmodell.

2.5 Den algoritmiske tenkeren

Både kjerneelementet *utforsking og problemløsning* og forskning om programmering i matematikk trekker *algoritmisk tekning* opp mot problemløsning. Stenseth et al. (2019, s. 8) beskriver begrepet som en metode for å lage og beskrive nye metoder som både mennesker og maskiner kan lese. *Algoritmisk tenkning* er en fundamental ferdighet som involverer tankeprosessene som formes når et problem skal formuleres og løses gjennom digitale hjelpemidler (Wing, 2006, s. 33; 2010, s. 1). Utdanningsdirektoratet forenkler begrepet som «å tenke som en informatiker» (Utdanningsdirektoratet, 2019a, s. 1). Her mener de at elever bruker *algoritmisk tenkning* når de gjennomgår en problemløsningsprosess hvor de bruker sin teknologiske kompetanse som et hjelpemiddel til å løse hele eller deler av problemet. Definisjonene av begrepet uttrykker at det kreves en prosess for å handtere og arbeide med et problem, disse er beskrevet gjennom en modell av Csizmadia et al. (2015). Dette studiet bruker denne modellen sammen med en oversettelse til norsk av Utdanningsdirektoratet (2019a). Modellen er navngitt *den algoritmiske tenkeren* og viser seks *nøkkelbegrep* og 5 *arbeidsmåter* som kan brukes til å løse et problem ved hjelp av digitale hjelpemidler. Figur 1 viser de seks nøkkelbegrepene; *logikk*, *algoritmer*, *dekomposisjon*, *mønstre*, *abstraksjon* og *evaluering*. Den viser også fem arbeidsmåter; *fikle*, *skape*, *feilsøke*, *holde ut* og *samarbeide*. *Nøkkelbegrep* og *arbeidsmåter* blir utdypet etter figuren.

Figur 1: Den algoritmiske tenkeren.

2.5.1 Nøkkelbegrep for den algoritmiske tenkeren

Det første nøkkelbegrepet *logikk*, handler om at eleven skaper mening til problemet ved å analysere det og finner relevant informasjon for å forutse hva som kommer til å skje (Csizmadia et al., 2015). Csizmadia et al. (2015) mener at logisk resonnering får elevene til å bruke tidligere kunnskap til å forsøke å bruke det i en ny kontekst. *Logikk* er noe som også blir mye brukt i andre nøkkelbegrep fra modellen.

Algoritmer er det andre nøkkelbegrepet i modellen. Det handler om å komme fram til en løsning ved å lage klare definisjoner av stegene man må gjennomgå. Prinsippet er å vite steg-for-steg hvordan man skal løse problemet og å gjennomføre det (Csizmadia et al., 2015). Csizmadia et al. (2015) forteller at hver enkelt algoritme ikke trenger å modelleres for hvert problem man møter, men at man lager seg forskjellige algoritmer for å håndtere ulike hinder som man kan bruke når det trengs. Et eksempel på en algoritme er hvordan man skal regne ut multiplikasjon eller divisjon. Når man har lært seg å gjøre disse utregningene, bruker man denne algoritmen hver gang man møter en slike oppgave uavhengig av hvilke tall som blir oppgitt (Csizmadia et al., 2015).

Dekomposisjon er en metode å dele problemet inn mindre deler. Ved å se på separate deler av et problem, vil det kunne skape en bredere forståelse av hver del individuelt og hjelpe til med å løse hele problemet (Csizmadia et al., 2015). I tillegg til å få mer forståelse for ulike deler av et problem, vil *dekomposisjon* bidra til å skape mulighet for forbedring av til dømes et program. Ved å dele opp et program i ulike seksjoner vil man kunne se og endre hvordan programmet håndterer de ulike elementene hver for seg (Csizmadia et al., 2015).

Mønstre, eller generalisering slik som Csizmadia et al. (2015) navngir det, er det fjerde nøkkelbegrepet i modellen. Det handler om å indentifisere mønstre, koblinger og likheter for å finne løsningen til problemet (Csizmadia et al., 2015). Dette innebærer å bruke tidligere løsninger av andre problem til å sette elementer fra dette inn i en ny løsning. Denne prosessen kan ifølge Csizmadia et al. (2015) startes ved å stille seg spørsmål som «har jeg løst et lignende problem før?» og «hvor forskjellig er det problemet i forhold til det nye?». I flere tilfeller vil man oppdage at hele eller deler av tidligere problem kan bli brukt til å finne løsningen på et nytt problem.

Det femte nøkkelpunktet *abstraksjon* handler om å gjøre problemet mer forståelig. Dette gjøres ved å skille ut unødvendig informasjon, slik at man står igjen med det som er nødvendig for problemet (Csizmadia et al., 2015). Csizmadia et al. (2015) bruker kart som et eksempel for *abstraksjon* til å skille nødvendig informasjon. Når man skal navigere seg ved hjelp av et kart, vil man sette søkelys på nåværende posisjon og ulike kjennemerker for å komme fram til sin destinasjon. Ved å fjerne unødvendig informasjon vil man lettere kunne finne løsningen til et problem (Csizmadia et al., 2015).

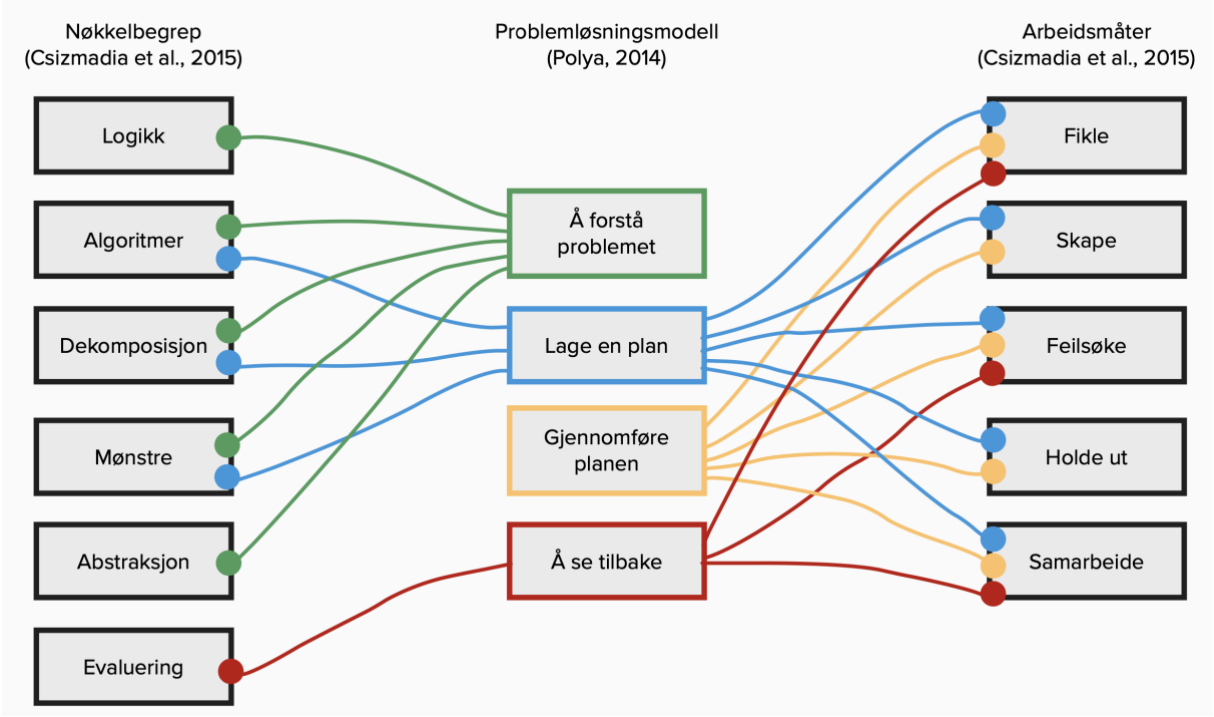
Det siste nøkkelbegrepet i *den algoritmiske tenkeren* er *evaluering*. Evalueringen skjer ved å se og tenke over løsningen eller deler av den, for å se om den er god nok (Csizmadia et al., 2015). Dette er et trinn der man kan gjøre endringer i løsningen for å forbedre den. Når man lager dataprogram vil *evaluering* kunne bidra til å gjøre programmet mindre resurskrevende, mer brukervennlig og eventuelt mer presist (Csizmadia et al., 2015).

2.5.2 Arbeidsmåter for den algoritmiske tenkeren

Modellen til Csizmadia et al. (2015) inkluderer også fem arbeidsmåter. *Fikle* handler om det å utforske og eksperimentere med problemet. Man kan designe og lage løsninger gjennom arbeidsmåten *å skape*. *Feilsøke* eller det engelske ordet *debugging* mye brukt i programmering og benyttes ved å oppdage og rette opp feil i koder (Csizmadia et al., 2015). *Holde ut* går ut på å ikke gi seg om man feiler, her må man fortsette og prøve igjen. Den siste arbeidsmetoden er å *samarbeide*. Dette kan gjøres gjennom samtale, deling og samarbeid med medelever.

2.5.3 Koblinger mellom den algoritmiske tenkeren og problemløsningsmodellen til Polya

For å belyse koblingene mellom nøkkelbegrep og arbeidsmåter i den algoritmiske tenkeren (Csizmadia et al., 2015) og problemløsningsmodellen til Polya (2014), har jeg konstruert Figur 2. Her knyttes elementene fra den algoritmiske tenkeren (til venstre og høyre) med de fire trinne i problemløsningsmodellen (i midten) via linjer.



Figur 2: Sammenligning av den algoritmiske tenkeren og problemløsningsmodellen til Polya (2014).

De fem første nøkkelbegrepene er punkter som kan sammenlignes med Polya (2014) sitt første trinn *å forstå problemet* fra kapittel 2.4.1. Ved å bruke logisk resonnering til å gi

mening til problemet setter man i gang det første steget i Polya (2014) problemløsningsmodell. Forskjellen her er at *logikk* kan brukes til mer enn å forstå problemet, det kan også brukes til å forutse hvordan ulike endringer påvirker løsningen (Csizmadia et al., 2015). De fire begrepene *algoritmer*, *dekomposisjon*, *mønstre* og *abstraksjon* vil også fungere som strategier til *å forstå problemet* slik Polya (2014) første trinn beskriver å skape mening til problemet. *Algoritmer*, *dekomposisjon* og *mønstre* er nøkkelbegrep som også kan sees i sammenheng med Polya (2014) andre trinn, *å lage en plan*. Forståelse av problemet som kan skapes gjennom disse tre nøkkelbegrepene vil kunne være med på å identifisere elementer som må inkluderes i planen for å finne løsningen. Det siste nøkkelbegrepet fra Csizmadia et al. (2015) sin modell, *evaluering*, sees i sammenheng med Polya (2014) siste trinn *å se tilbake*. Ved å validere løsningen til problemet, vil man kunne utvikle egen kompetanse og skape nye og bedre løsninger (Csizmadia et al., 2015; Polya, 2014).

Om man skal plassere arbeidsmåter inn i problemløsningsmodellen til Polya (2014) vil alle være hvordan man *lager en plan* og *gjennomføre planen*, altså trinn 2 og 3. Arbeidsmåtene går ut på hvordan man velger å finne ut av løsningen, i dette tilfellet hvilken metode man planlegger å bruke og hvordan man gjennomfører planen. Å *fikle*, *feilsøke* og *samarbeide* vil i noen tilfeller også fungere som arbeidsmåter innenfor det siste steget *å se tilbake* på problemet av Polya (2014), fordi man kan redigere eller diskutere løsningen for å forbedre den.

3 Metode

Gjennom dette kapittelet utdypes forskningsprosessen som kreves for å besvare problemstillingen: *hvilke kvaliteter har programmeringsoppgaver i matematikk på tvers av to læreverk?* Det redegjøres for valget av læreverk, hvordan det analytiske rammeverket er sammensatt og løser utfordringer som krever tilpassinger og utvidelser og hvordan det brukes i studiet. Til slutt belyses studiens validitet, relabilitet og forskningsetikk. Hensikten er å tydeliggjøre hvordan studiet besvarer forskningsspørsmålene til problemstillingen og studiens truverdighet.

3.1 Valg av læreverk

Valget av læreverk er gjort gjennom to undersøkelser. Hovedkravet jeg satte til valget, var at de skulle være læreverk for ungdomstrinnet (8. – 10. trinn) som baserte seg på de nye kompetansemålene og kjerneelementene gitt av LK20. Læreverk kunne være både tradisjonelle og digitale læreverk så lenge de fyllet kravet over. Undersøkelsen for valg av læreverk ble utført gjennom et samarbeid med en medstudent der vi delte resultatene med hverandre. Jeg og medstudenten, Robert Pepaj, valgte begge å gjennomføre analyse av POiM fra læreverk på ungdomstrinnet. For å danne et større perspektiv på valget av læreverk vi hadde foran oss, valgte vi å samarbeide på innhenting av data til utvalget. Pepaj gjennomførte en digital undersøkelse på en lukket lærerkanal på det sosiale mediet *Facebook*. Kanalen heter *Matematikkdidaktikk* og medlemmene består av matematikklærere eller matematikklærerstudenter. I denne kanalen kan man diskutere og stille spørsmål til medlemmene om relevante temaer i matematikkfaget. For å få tilgang må man sende inn en søknad som viser til formålet ditt for hvorfor man ønsker tilgang. Dersom formålet ikke godkjennes av medlemmer i denne kanalen, vil man nektes tilgang. Samtidig, tok jeg kontakt med ungdomsskoler via e-post. Vi begge sendte også e-post til ulike forlag for å spørre om oversikt over hvor mange som brukte deres læreverk.

Pepaj la ut en spørreundersøkelse på lærerkanalen, med spørsmål om hvilket læreverk de brukte for å velge ut programmeringsoppgaver til matematikkundervisningen. Det ble gitt alternativer å velge mellom, og mulighet til å legge til egne alternativ dersom deres læreverk ikke var til stede. I min undersøkelse, hvor jeg sendte e-post til 140 ungdomsskoler,

konstruerte jeg en felles e-post som jeg sendte til skoler. E-posten stilte skolene spørsmål om hvilket læreverk de brukte til å undervise programmering i matematikk. En av utfordringene med å få kontakt med ulike skoler via e-post, var å få meldingen til å ikke bli slettet i deres kommunale filter. Jeg oppdaget raskt at jeg ikke fikk tillatelse til å sende e-post til alle skolene, da min egen e-postadresse ikke var registrert som godkjent innenfor deres kommune sin infrastruktur eller vart markert som søppel-e-post. I ti av tilfellene der jeg opplevde dette, fikk jeg et generert e-post svar fra mottakeren som forklarte årsaken. Jeg valgte derfor å sende ut e-poster individuelt til hver enkelt skole for å minke sannsynligheten til å bli markert som søppel-e-post. 130 av 140 e-poster gikk igjennom uten den generert beskjeden som fortalte hvorfor e-post filteret slettet meldingen. Over en lenger periode, fikk jeg tilbakemelding fra 26 skoler hvor de nevnte en eller flere læreverk de brukte for å undervise programmering i matematikk. Samtidig sendte jeg e-post til fem kommuner fra forskjellige fylker hvor jeg stilte spørsmål om de hadde oversikt over hvilket læreverk skoler i deres kommune brukte, men fikk ingen respons. Resultatene fra begge undersøkelsene presenteres i Tabell 1.

Tabell 1: Resultat fra Pepaj og min undersøkelse av læreverk brukt av lærere og i skolen.

Medstudent sin undersøkelse.

Læreverk	Matemagisk	Kidsakoder.no	Skolen.cdu.no	Kikora.no	Campus Inkrement	Aunivers	Salaby	Skolestudio (Multi smart)	Annet	**Totalt**
Antall stemmer	19	35	21	21	17	5	14	1	19	**152**

Min undersøkelse.

Læreverk	Matemagisk	Kidsakoder.no	Skolen.cdu.no	Kikora.no	Campus Inkrement	Maximum	Matematikk (programmerings-heftet)	Skolestudio.no	Annet	**Totalt**
Antall stemmer	8	2	6	4	9	3	2	3	3	**40**

Deltakerne kunne svare på flere alternativer i Pepaj sin undersøkelse fargekodet blått i Tabell 1. Det betyr at totalt antall stemmer ikke korresponderer med antall deltakere, men det gir likevel en oversikt hvilket læreverk denne gruppen bruker mest til å hente POiM til matematikkundervisningen deres. Resultatet viser at i denne gruppen stemte flest deltakere på at de hentet POiM fra nettressursen *Kidsakoder.no* (35). Både *Skolen.cdu.no* (21) og *Kikora.no* (21) fikk like mange stemmer og er nest mest brukt i denne gruppen. Like under er *Matemagisk* (19) og *Campus Inkrement* (17) som stiller seg nokså likt som de andre. De resterende resultatene er *Salaby* (14), *Aunivers* (5) og til slutt *Skolestudio* sin *Multi Smart*

Øving (1). Annet (19) er en sammenslåing av resultater som viser til udefinerte kilder, som egne eller kollegaers oppgaver, og kan derfor ikke defineres i likhet med de andre alternativene.

Nederst i Tabell 1 viser resultatene av responsen jeg fikk fra 26 ungdomsskoler i Norge, fargekodet med rødt. De fleste av ungdomsskolene som gav respons, brukte læreverket *Campus Inkrement* (9 stykk) og *Matemagisk* (8 stykk) til å undervise programmering i matematikk. De andre resultatene, *Kikora.no* (4), *Maximum* (3), *Skolestudio.no* (3) og annet (3), var nokså jevnt stilt. Til slutt kom programmeringsheftet til *Matematikk* (2) og *Kidsakoder.no* (2). I responsene nevnte ti skoler at de også brukte det fysiske læreverket til *Matematikk* fra *Cappelen Damm* til undervisning, men ikke til programmering. Bare to av alle svarene fortalte de at de brukte tilleggsressursen programmeringsheftet fra *Matematikk* til å undervise programmering. Det fysiske læreverket inneholder ingen oppgaver om programmering, alt om programmering er flyttet til tilleggsressursen. Jeg valgte derfor å utelate disse ti fra besvarelsen og bare oppgi det de brukte som læreverk for programmering, fordi det ellers ville påvirke resultatet drastisk.

I resultatene er det et alternativ som ikke passer inn i studiens rammer. Det er derfor nødvendig å eliminere dette alternativet. *Kidsakoder.no* er en gratis læringsressurs på internett åpen for alle. Formålet deres er å hjelpe unge å lære programmering, bruke teknologi og bli kjent med informatikk (Kidsakoder, u.å.). De kan ikke defineres som et læreverk for matematikk fordi *Kidsakoder.no* er en frivillig organisasjon som produserer programmeringsoppgaver og ulike undervisningsopplegg uavhengig av LK20. De andre kandidatene følger kompetansemål knytt til matematikk og kjerneelement fra LK20 og er bygd opp med progresjon ut ifra dem. Jeg kan derfor ikke inkludere denne i utvalget fordi det ikke er et læreverk, selv om det er en ressurs som brukes av mange lærere.

For å få et helhetlig bilde av resultatene, har jeg slått sammen begge resultatene fra begge undersøkelsene (se Tabell 2) og så valgt to læreverk å fokusere på.

Tabell 2: Sammenlagte resultat fra Pepaj og min undersøkelse av læreverk brukt av lærere og i skolen.

Læreverk	Matemagisk	skolen.cdu.no	Kikora.no	Campus Inkrement	Aunivers	Salaby	Multi Smart	Maximum	Matematikk (programmerings-heftet)	Skolestudio.no	Annet	**Totalt**
Stemmer	27	27	25	26	5	14	1	3	2	3	22	**155**

Tabell 2 viser at Matemagisk og Skolen.cdu.no har begge fått 27 av 155 stemmer hver. Med en stemme mindre har vi *Campus Inkrement*, *Kikora.no* fikk en stemme mindre enn dette igjen. De resterende resultatene har fått mellom 1 og 14 stemmer hver, om man ser bort fra *annet* som ikke kan defineres. På grunn av differansen mellom stemmene til de fire læreverkene med flest stemmer og de resterende, utelukkes alternativer med under 20 stemmer fra dette studiet. Valget stod derfor mellom *Skolen.cdu.no*, *Kikora.no* og *Campus inkrement* som digitale læreverk og *Matemagisk* som et analogt læreverk.

Matemagisk er forlaget *Aschehoug* sitt læreverk for matematikk på ungdomstrinnet og *Matemagisk 8* er utgitt 24.04.2020. Forfatterne er Anne Karin Wallace og Asbjørn Lerø Kongsnes. Læreverkets innhold beskrives som å ha stor variasjon av oppgaver, aktiviteter og spill som ønsker å engasjere og skape meningsfull matematikkundervisning for elever og lærere (Aschehoug, u.å.). Oppgavene er bygget opp av ulike vanskelighetsgrader, slik at elever kan selv differensiere oppgaver passet til deres nivå. *Aschehoug* har designet flere aktiviteter som de har definert som *Snakke matte*. Med slike aktiviteter ønsker de å oppfordre elever til å argumentere med egne ord i diskusjon med medelever og utvikle kritisk tenkning (Aschehoug, u.å.). De fremhever også at elever kan ved hjelp av grunnboken deres lære programmering og utvikle algoritmisk tenkning i takt med matematikkfagets premisser. Forlaget har tilleggsressurser til Matemagisk-serien. Blant annet en elevhåndbok for elever som gir dem nyttige tips og eksempler. En parallellbok som erstatter grunnboken for elever som ifølge forlaget vil kunne hjelpe de som strever mest med matematikk (Aschehoug, u.å.). Andre tilleggsressurser er digitale utgaver av grunnbøkene og lærerveiledning. Pepaj tok kontakt med *Aschehoug* for å høre om de hadde talldata på hvor mange ungdomsskoler som brukte deres læreverk. De fortalte at de ikke kunne oppgi salgstall, men formidlet at de er mye brukt og har en betydelig markedsandel.

Campus Inkrement er et digitalt læremiddel for matematikk og naturfag. Deres læreverk for matematikk heter *Campus Matte*. Dette læreverket legger vekt på dybdelæring og tilpasset opplæring og er utviklet utfra Kunnskapsløftet 2020 (Inkrement, u.å.). De bygger på prinsippet omvendt undervisning, hvor elever ser på teoretiske undervisningsvideoer hjemme som forberedning til skolen. I matematikkundervisningen bruker de tid på å diskutere og løser oppgaver sammen med lærer. I likhet med flere av de andre læreverkene, leverer *Campus Matte* nivådifferensierte oppgaver og klasseromsaktiviteter for å skape motivasjon og følelse av mestring for alle elever (Inkrement, u.å.). *Campus Matte følger* et «læringssti»-prinsipp som inneholder teori, oppgaver og videoforklaringer. Her får elevene også mulighet til å se om svarene deres er korrekt eller feil utført. Noe som dette læringsmidelet har, er egenvurdering av leksjoner. Her ønsker *Campus Matte* å få elevene til å bli bevisst over egen læring (Inkrement, u.å.). Jeg kontaktet *Campus Inkrement* for å høre om talldata på hvor mange som brukte deres læremiddel for matematikk på ungdomstrinnet. De ønskte heller ikke å oppgi salgstall, men de kunne si at det er totalt over 1000 grunnskoler som benytter *Campus Matte*. Ettersom dette er et digitalt læreverk, krever det innlogging med lisens for å få tilgang. Gjennom e-post fikk jeg en egen innlogging med tilgang til læreverket.

Skolen.cdu.no, eller forkortet *Skolen*, er en digital nettressurs av *Cappelen Damm* hvor man kan få lisens til å arbeide med en digitalutgave av læreverkene deres. I tillegg til å kunne lese igjennom læreverkene digitalt, er det designet egne læringsstier som gjør at man kan jobbe med fagene i egent tempo. Læreverket ønsker å gi elevene, gjennom ressursen sin, engasjerende klasseaktivitet, dybdelæring, aktuelle oppgaver og progresjon (Cappelen Damm, u.å.). Jeg tok kontakt med *Cappelen Damm* for å høre om de hadde talldata på hvor mange ungdomsskoler som brukte deres nettressurs. Svaret var at de ikke oppgav salgstall på produkter, men kunne si at det er svært mye brukt og rundt annenhver elev bruker eller har tilgang til *Skolen* på ungdomstrinnet. Dette inkluderer alle læreverk fra forlaget, ikke bare for matematikk.

Kikora.no (*Kikora*) er et heldigitalt læreverk for matematikk (Kikora, u.å.). Det inneholder matematikkbøker for hele grunnskolen og videregåendeskole. De reklamerer med å ha utforskende læringspakker hvor elever blir aktiviserte i oppgavene og får umiddelbart

tilbakemelding på utregninger (Kikora, u.å.). *Kikora* har læringsstier som er nivådifferensierte for å sikre at alle elever får mulighet til å kunne kjenne på mestring uavhengig av nivå. Jeg tok kontakt med *Kikora* for å høre om de hadde talldata på hvor mange ungdomsskoler som bruker deres læreverk. De forteller at det er omtrent 370 ungdomsskoler i Norge som har tilgang til *Kikora* som læreverk.

Avgjørelsen for valget av læreverk til dette studiet falt på *Matemagisk* og *Campus Matte*. Disse var læreverk som ifølge undersøkelsene er mye brukt av lærere og det gav studiet et utvalg av både et analogt og digitalt læreverk. Gjennom bibliotek og kontakt med utgiver fikk jeg tilgang til disse læreverkene. Begge læreverkene har nokså lik struktur med tanke på oppgaver av ulike nivåer og diskusjonsoppgaver. De er også veldig ulike i hvordan de er konstruert til å brukes. Ved å analysere programmeringsoppgaver fra disse læreverkene fikk studiet en variert samling av empiri.

3.2 Innholdsanalyse

For å besvare problemstillingen trenger jeg å identifisere kvaliteter gjennom analyse av læreverks innhold som POiM og struktur av temaer. Jeg valgte derfor å gjennomføre en innholdsanalyse i dette studiet. Metoden ble valgt fordi den blir sett på som en fleksibel metode for å analysere data (Hsieh & Shannon, 2005, s. 1277). Denne fleksibiliteten bidrar til at studiet kan åpnes opp for en kvalitativ tilnærmelse med kvantitative preg. Den kvalitative innholdsanalysen er gjennomført med en systematisk gjennomgang av læreverket hvor det skjer en kategorisering av innholdet (Grønmo, 2016, s. 175). Dette medfølger at jeg kan se på hvordan læreverket har strukturert innholdet med et kvalitativt blikk og kategorisere deres matematiske temaer, formidlingsmetoder og læringsmål. Ved å supplere dette med en kvantitativ innholdsanalyse, kan jeg se på relevante deler av innholdet som blir systematisert og registrert slik at det kan brukes som studiens datagrunnlag (Grønmo, 2016, s. 213). Dette skaper muligheten for å kode relevant innhold, som er programmeringsoppgaver i dette studiet, basert på tidligere forskning og teori. Med denne sammensettingen kan jeg kvantifisere innholdet i læreverkene basert på en kvalitativ tolkningsprosess for å besvare studiens problemstilling.

3.3 Analytisk rammeverket

Rammeverket er inspirert av Charalambous et al. (2010) som gjennomførte en internasjonal studie der de sammenlignet hvordan addisjon og subtraksjon av brøk ble behandlet av ulike lærebøker for matematikk. For å sikre at analysen av læreverk tok hensyn til hvordan de ulike forlagene presenterte matematiskinnhold og forventninger til matematikkoppgavene, så valgte Charalambous et al. (2010) å se på innholdet gjennom *horisontal* og *vertikal* analyse. Jeg brukte samme tilnærming, men har utvidet og tilpasset den for å undersøke POiM i analoge og digitale læreverk.

Den horisontale analysen sitt formål er å se på presentasjonen læreverket gir av seg selv, strukturen og det matematiske innholdet (Charalambous et al., 2010). Dette innebærer å se på blant annet forfatterne, antall sider og temaer i boken. Charalambous et al. (2010) valgte å dele horisontal analyse i to underkategorier for informasjon; *bakgrunnsinformasjon* og *struktur*. Den vertikale analysen gir en mer beskrivende og dypere analyse av matematikkinnhold i læreverkene og består av tre underkategorier: *formidling til elever*, *krav til elever* og *sammenhenger (Charalambous et al., 2010)*.

Tabell 3: Horisontal analyse av læreverk, min oversettelse.

Horisontal analyse av læreverk

Bakgrunnsinformasjon	Struktur
o Tittel.	o Antall leksjoner og sider per lekskjon.
o Antall bøker.	o Struktur av leksjoner.
o Antall sider og tetthet.	o Læringsmål dekt.
o Forfatter status og rådgiving.	o Rekkefølge av tema.
o Utgiver og publikasjonsdato.	
o Tillegsmaterialet.	

Tabell 3 viser innholdet til underkategoriene til den horisontale analysen av Charalambous et al. (2010). *Bakgrunnsinformasjon* ser på ulike elementer som kan gi et beskrivende overblikk over læreverket og hvem som har produsert materialet. Her ser vi på hvor mange bøker læreverket består av, deres titler og hvem som er forfattere og utgiver til bøkene, samt på

antall sider og hvor mye hver side inneholder, når læreverket er publisert og om det eventuelt finnes tilleggsmaterialet. (Charalambous et al., 2010).

Struktur ser på temaer og læringsmål sammen med hvordan de har valgt å organisere dette (Charalambous et al., 2010). Her er søkelyset på selve strukturen av leksjonene, hvor mange leksjoner og antall sider per leksjon. Det blir også sett på hvilket læringsmål leksjonene ønsker fokusere på og hvordan rekkefølgen av temaene er organisert. Den horisontale analysen beskriver ikke direkte matematikkoppgavene som skal undersøkes i dette studiet, men den vil gi et overblikk over hvordan læreverkene har valgt å strukturere innholdet som kan påvirke hvordan oppgaver blir gitt. Analysen av strukturen vil kunne gi svar på hvilke forkunnskaper elevene har fått fra læreverket før oppgaver blir gitt. Dette vil være med på å avgjøre hvilke kvaliteter POiM har i den vertikale analysen.

Tabell 4: Vertikal analyse av læreverk, min oversettelse.

Vertikal analyse av læreverk

Formidling til elever	Krav til elever	Sammenhenger
Matematisk Innhold ○ Tema-spesifisert kontstruksjon og struktur. ○ Definisjoner, regler og konvensjoner. ○ Representasjoner gjennom illustrasjon.	Kognitive krav: - Memorering - Prosedyrer uten sammenheng. - Prosedyrer med sammenheng. - Gjøre matematikk.	● Sammenhenger innenfor og mellom tråder. ● Sammenhenger mellom klasseromsinstruksjon og læreverk. ● Sammenheng mellom situasjonen utenfor skolen.
Matematisk praksis ○ Bearbeidet eksempler. ○ Modellere tekning.	Type svar: - Kun svar. - Svar med matematisk setning. - Forklaring. - Vise gyldighet.	
Holdning ○ Syn på matematikk.		

Tabell 4 viser innholdet til underkategoriene av vertikal analyse. *Formidling til elever* inkluderer *matematisk innhold*, *matematisk praksis* og *holdning*. *Matematisk innhold* ser på læreverkets temaer og definisjoner, regler og representasjoner som elever kan støtte seg til fra boken når de arbeidet med matematikk. *Matematisk praksis* ser på hvordan læreverket bruker deres eksempler til å formidle noe og modellere tenkning, mens *holdning* ser på hvordan læreverkets syn på matematikk påvirker innholdet (Charalambous et al., 2010).

Krav til elever av Charalambous et al. (2010) definerer hvilke *potensielle kognitive krav* og *type svar* en oppgave har. For å klassifisere *potensielle kognitive krav* brukes Smith og Stein (1998) sin fordeling av fire kognitive krav fra deres task-analysis guide, som var tidligere utdypet i 2.2. *Type svar* beskriver hvilken form for svar en oppgave kan kreve. Alternativene som blir brukt her er *kun svar*, *svar med matematisk setning*, *forklaring* eller *vise gyldighet*. De defineres av Charalambous et al. (2010) som:

- *Kun svar* – et enkelt numerisk svar eller uttrykk.
- *Svar med matematisk setning* – gir *kun svar* sammen med en setning som forklarer det.
- *Forklaring* – forklarer løsningen eller prosessen de brukte.
- *Vise gyldighet* – beviser svaret deres ved å begrunne tilnærmingen av løsningen og rasjonaliteten av det (Charalambous et al., 2010, s. 129, min oversettelse).

I *sammenhenger*, undersøkes det om det kan trekkes tråder fra sammenhenger mellom innholdet i læreverket og virkeligheten. Charalambous et al. (2010) ser etter sammenhenger innenfor og mellom tråder fra boken, læreverk og klasseromsundervisning og til slutt mellom situasjoner utenfor skolen. Formålet er å se om oppgaver kan knyttes opp mot andre temaer for å skape sammenheng, brukes gjennom klasseromsaktivitet eller relateres til virkeligheten (Charalambous et al., 2010).

3.3.1 Adapsjon av rammeverket til POiM

Rammeverket til Charalambous et al. (2010) var utarbeidet for å analysere oppgaver innen et matematisk tema, brøk. Matematiske temaer er et av elementene som læreverk bruker for å strukturere innholdet. I motsetning til dette, kan POiM høre til forskjellige matematiske temaer og kan plasseres for eksempel under brøk eller funksjoner. Jeg har derfor to utfordringer foran meg: skape tydelige kriterier for å identifisere POiM og å bestemme hvilke matematiske temaer de tilhører. Det er også nødvendig å definere hva som er elevers teoretiske grunnlag når de skal løse POiM.

3.3.1.1 Kriterier for å identifisere POiM

Ved å sette klare krav for hvordan jeg skal identifisere POiM, kan dette gjøres gjennom den vertikale analysen. Ved å se om oppgaven kan kategoriseres under en av Csizmadia et al. (2015) *arbeidsmåter* fra *den algoritmiske tenkeren*, vil den identifiseres som en programmeringsoppgave*:*

- *Skape* – oppgaven ber eleven om å lage et program for å finne løsningen.
- *Fikle* – oppgaven ber eleven om å gjøre endringer i et gitt eller tidligere program for å utforske eller eksperimentere.
- *Feilsøke* – oppgaven ber eleven om å undersøke et program for å finne noe som er galt og eventuelt rette opp i det.
- *Holde ut* – elever velger å ikke gi seg og fortsetter å prøve når de ikke klarer å løse en oppgave.
- *Samarbeide* – oppgaven ber om at elever diskuterer eller deler tanker med andre rundt et program basert på oppgavens kontekst (Csizmadia et al., 2015).

Å *holde ut* er en arbeidsmåte elevene selv kan velge å bruke dersom de sitter fast ved en oppgave. De andre arbeidsmåtene identifiseres ved at oppgaven ber om at elevene gjør dem. Siden studiet undersøker POiM som læreverk legger til rette for, vil jeg ikke kunne samle inn data som tilsier at elevene bruker *holde ut* i arbeidet med oppgavene. Jeg valgte derfor å ikke telle å *holde ut* som en av arbeidsmåtene jeg ser etter. Dersom en oppgave går under en av arbeidsmåtene og defineres som en programmeringsoppgave, så må den også tilfredsstille neste krav. Det er om programmeringsoppgaven kan knyttes til en av læringsmålene for programmering i matematikk fra den horisontale analysen. Om en oppgave utfyller begge kravene, telles den som en POiM og undersøkes videre gjennom den vertikale analysen. I tilfeller hvor oppgaven tilfredsstiller bare en av kravene eller deler av læringsmålet som ikke inneholder matematikk, ekskluderes oppgaven fra studiet. I kapittel 4.2.4 vises det et eksempel hvor en oppgave går under en av arbeidsmåtene for programmeringsoppgaver, men tilfredsstiller ikke den matematiske delen av læringsmålet.

3.3.1.2 Kriterier for å identifisere matematiske temaer til POiM

Ved å forberede krav som må oppnås for å kunne identifisere hvilket matematisk tema en POiM har, kan den andre utfordringen løses ved hjelp av resultatene fra den horisontale analysen. For å klassifisere POiM sitt matematiske tema, må den være tilknyttet et kapittel med et matematisk tema og inneholde læringsmål for programmering. I tilfeller når POiM er tilknytt et læringsmål med programmering, men er ikke under kapittel for et spesifikt matematisk tema, må jeg gjennom den vertikale analysen undersøke hva oppgaveteksten spør om og hvilken matematikk som kreves for å løse den. Dette danner et grunnlag for å plassere oppgaven under et matematisk tema.

Ettersom de utvalgte læreverkene for studien er oppbygde etter LK20, og inndelingen er basert på kompetansemål etter 8. trinn i LK20, vil matematiske temaer læreverk jobber med være like. Dette bidrar til å skape resultater som kan sees i samsvar på tvers av hverandre. Det dannes derimot en utfordring gjennom læreverkenes struktur. *Matemagisk 8* har valgt å plassere POiM under spesifikke kapittel knytt til ulike matematiske temaer, mens *Campus Matte 8* har plassert dem under et egent kapittel for programmering. I forhold til kravene i avsnittet over, er det naturlig å gjennomføre analysen av *Matemagisk 8* først fordi POiM sine matematiske temaer kan identifiseres med deres kapitler. Om det hadde blitt gjort motsatt ville det ikke påvirket resultatene, men det ville skapt en situasjon hvor to like temaer hadde fått to ulike navn, for eksempel temaet «algebra» i *Campus Matte 8* og «algebraiske uttrykk og formler» i *Matemagisk 8*. Det er derfor mer logisk for utvalget i dette studiet å plassere «algebra» inn under «algebraiske uttrykk og formler», da *Matemagisk 8* har allerede navngitt temaet gjennom kapittelet. Ved en tilnærming hvor *Matemagisk 8* analyseres først kan matematiske temaer for POiM i *Campus Matte 8* analyseres for å se om de kan plasseres under temaene som er allerede på listen eller om det behøves nye.

3.3.1.3 Ståsted for hva som elevers teoretiske grunnlag og ønskelig løsningsformat

Mitt ståsted for progresjon i læreverkene vil påvirke analysen når den kommer til hvilke forkunnskaper elever har i møtet med POiM, derfor måtte den defineres. I skolehverdagen er det opp til læreren å avgjøre hvordan et læreverk blir brukt, men siden dette studiet undersøker innholdet i læreverkene er det ikke mulig å ta høyde for dette. Jeg har derfor tatt ståstedet som ser på progresjon i læreverket som lineært fra første til siste side. Det betyr at

oppgavene gitt i læreverkene forsøker å vise fremgang i tråd med læreverkets oppbygning, derfor antar jeg at elevene også jobber lineært. En slik tilnærming blir ikke helt realistisk fordi det ikke tar høyde for at elevene selv kan velge mellom oppgaver som er nivådifferensierte. For innholdsanalysen av POiM betyr det i praksis at elevene har kun fått leksjoner, videoleksjoner og sider med eksempler, regler og dømer som er gitt før oppgavene som deres teoretiske grunnlag. Når elevene går videre til neste oppgave, tar de med seg dette grunnlaget inkludert løsningen av forrige oppgave og eventuell ny kunnskap gitt i en form for leksjon mellom oppgavene. For *Campus Matte 8*, hvor POiM er gitt i det siste kapittelet, ansees det at alle matematiske temaer som er relevante for oppgaven er gjennomgått og alle POiM er gjennomgått lineært.

POiM som ber elevene lage et program eksisterer. Det blir ikke alltid spesifikt oppgitt i hvilket format programmet skal skives med. I praksis ville man kunne lage programmet gjennom flere ulike programmeringsspråk med en fungerende løsning. I sammenheng med avsnittet over, ønsker jeg å følge læreverkene lineært. Da vil jeg også bruke programmeringsspråket som de selv introduserer og bruker i det aktuelle læreverket. Dette fører til at elevenes teoretiske grunnlag for programmeringsferdigheter baserer seg på innholdet de arbeider med fra læreverket. Dersom en oppgave indikerer at den kan løses med andre koder eller programmeringsspråk, så tas ikke de alternative løsningene med i evalueringen for kvaliteter. Dette er fordi de løsningene kan ikke knyttes opp mot læreverkets teoretiske innhold. I *Campus Matte 8*, det digitale læreverket, får elevene en tilbakemelding på om svaret deres er korrekt eller ikke etter de har gitt svaret deres. Dette gjelder og for programmeringsoppgavene deres. Oppgavene vil derfor analyseres med utgangspunkt i at løsningen gis i det formatet læreverket kan gi tilbakemelding på.

3.3.2 Tilpassing og utvidelse av rammeverket

Rammeverket til Charalambous et al. (2010) var opprinnelig ment for å undersøke temaet brøk. Det oppstår derfor noen utfordringer ved å bruke det for POiM uten noen tilpassinger. For å besvare forskningsspørsmål 3, krevde det også en utvidelse av rammeverket. Jeg skal nå tydeliggjøre hvilke deler av Charalambous et al. (2010) sitt rammeverk som brukes, ekskluderes og tilpasses i dette studiet før utvidelsen beskrives til slutt.

3.3.2.1 Tilpassing av horisontal analyse

Den horisontale analysen sin rolle var å skape et overblikk over læreverkene. *Bakgrunnsinformasjon* bidrar ikke med besvarelsen av problemstillingen, men den gir oversikt til hvem som har produsert læreverket og om det er tilleggsmaterialet som er aktuelt å inkludere i studiet. Jeg har valgt å beholde *tittel*, *antall sider*, *utgiver og publikasjonsdato* og *tilleggsmaterialet* fra rammeverket. *Side tetthet* og *forfatter status og rådgiving* er ekskludert, da denne informasjonen ikke er nødvendig for min studie.

Alle elementer fra *struktur* er bevart fra rammeverket. Selv om *struktur* ikke ser direkte på POiM, vil den kunne si noe om hvordan forlagene har strukturert deres oppbygging av oppgaver og hvilken type oppgaver som gis. Det blir derfor sett på som nødvendig å undersøke dette for å skape en helhet av læreverket som bidrar til å klassifisere element i innholdsanalysen. *Antall leksjoner og sider per leksjon* og *struktur av leksjoner* gir en oversikt over hvor leksjoner befinner seg i læreverket. Dette vil gjøre det lettere å finne innhold som representerer elevenes teoretiske grunnlag og kartlegge hvilke kategorier av oppgaver de har. *Læringsmål dekt* og *rekkefølge av tema*, gir oversikt over hvilke læringsmål som inneholder programmering og for hvilke tema de tilhører. Metode for å identifisere matematisk tema POiM hører til var beskrevet i kapittel 3.3.1.2. Mulige matematiske temaer til POiM blir identifisert her, mens endelige temaer er verifisert gjennom den vertikale analysen.

3.3.2.2 Tilpassing av vertikal analyse

Den vertikale analysen sin rolle er å analysere matematiskinnhold i leksjoner, regler, eksempler, representasjoner og oppgaver. Jeg utdyper hvilke deler som inkluderes, ekskluderes eller tilpasses for hver av de tre kategoriene fra vertikal analyse i egne avsnitt under.

Fra *formidling til elever* beholdes *matematisk innhold* og *matematisk praksis* fra rammeverket. Hva læreverket ønsker å formidle med innholdet deres vil beskrive elevenes teoretiske grunnlag før de skal gjennomføre oppgaver. Dette vil kunne prege hvilke kvaliteter en oppgave har senere i analysen. Dersom elever kan se tilbake på leksjoner, eksempler, regler eller tidligere løsninger av oppgaver for å løse en ny oppgave, vil dette føre

til reproduksjon av innhold. *Holdning* ekskluderes fordi læreverkets *syn på matematikk* ikke bidrar til å besvare studiets problemstilling. Det er POiM som skal undersøkes i problemstillingen, ikke læreverkets holdning til programmering i matematikk.

I *krav til elever* utgir *potensielle kognitive krav* store deler av datamaterialet i dette studiet fordi det gir datagrunnlag for å kunne besvare det andre forskningsspørsmålet. Jeg bygger på Smith og Stein (1998) og Valenta (2016) sin forskning (se kapittel 2.2) og viser hvordan beskrivelser av kognitive krav kan tolkes for POiM i Tabell 5.

Tabell 5: Sammenheng mellom matematikkoppgavers kognitive krav og POiM.

	Definisjon av kognitive krav (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 3-7)	**Kognetive kravs tilknytting til programmering**
Memorering	Det laveste kognitive kravet hvor man reproduserer tidligere innhold og det skapes ikke sammenheng mellom løsningen og regler, formler eller definisjoner. Det brukes ikke strategier for å løse oppgaven, fordi den ikke eksisterer eller er nødvendig å bruke.	Svaret formes ved å gjøre ingen eller små endringer, som å bytte tallverdier på program som er reproduserte fra leksjoner, eksempler eller tidligere oppgaver. Det skapes ikke sammenheng mellom kodene til programmet og matematikk.
Prosedyrer uten sammenhenger	Lavt kognitivt krevende algoritmiske oppgaver hvor man bruker en strategi man kjenner til og som ikke skaper sammenheng mellom løsningen og matematiske konsepter eller ideer. Målet er derfor å finne den korrekte løsningen.	Elevene oppfordres til å bruke tidligere lærte koder for å produsere et svar gjennom programmering. Det er mer fokus på hvordan koden kan brukes enn memorering av den.
Prosedyrer med sammenhenger	Høyt kognitivt krevende oppgaver som forsøker å utvikle dypere forståelse av matematiske konsepter eller ideer gjennom bruk av strategier. Ofte kan løsningen delvis besvares ved å følge strategier, men så vil den kreve at selve strategien utforskes i kontekst med problemet som skal løses. De presenteres ofte gjennom ulike representasjoner.	Elevene oppfordres til å knytte matematiske sammenhenger opp mot tidligere lærte koder og selve programmet.
Gjøre matematikk	Det høyeste kognitive kravet for oppgaver som krever utforsking og forståelse av matematiske strategier, fenomen og sammenhenger. Løsningen kan ikke skapes gjennom kun bruk av tidligere strategier, men ved å knytte sammen tidligere kunnskap og erfaringen med ny kunnskap fra utforskingen, skapes det mening mellom matematiske ideer og konsepter som kan bidra til løsningen. Det vil være nødvendig å begrense mulige løsningsstrategier ved å finne relevant informasjon fra problemet.	Elevene oppfordres til å utforske matematiske sammenhenger med lærte og eventuelt nye koder.

Hvilken *type svar* elevene må oppgi, vil kunne bidra til å definere ulike kvaliteter POiM har. Dette kan begrunnes med å se sammenheng mellom svaret de må levere og Polya (2014) sitt siste trinn i problemløsningsprosessen, gitt i kapittel 2.4.1, å *se tilbake*. Avhengig av hvilket svar oppgaven krever, må eleven muligens reflektere over strategier, argumenter og matematiske elementer de har brukt for å komme frem til løsningen og grunngi dette. For å vinkle det inn mot POiM, kreves det noen tilpassinger av *type svar*. Fra rammeverket

beholdes *kun svar* og *vise gyldighet*, mens *svar med matematisk setning* og *forklaring* settes sammen til *forklaring*. Charalambous et al. (2010) *type svar* er bearbeidet og presenteres under med definisjoner som passer for POiM:

- *Kun svar* – et enkelt svar gjennom produksjon av et program. Svaret kan også gis i form av å skrive eller kryss av et alternativ fra en liste.
- *Forklaring* – forklarer svaret eller prosessen de brukte. Oppgaven kan også be om *kun svar* sammen med forklaringen.
- *Vise gyldighet* – reflektere over eller begrunne løsningen av oppgaven og dens rasjonalitet (Charalambous et al., 2010).

Fra *sammenhenger* ekskluderes *sammenhenger mellom klasseromsinstruksjon og læreverk* og *sammenhenger mellom situasjoner utenfor skolen*. Siden det er kvaliteter ved POiM som undersøkes i dette studiet, er det ikke relevant å se på disse to sammenhengene. Det er derimot interessant å se på *sammenhenger innenfor matematiske tråder*. *Matemagisk 8*, som er en av læreverkene som skal undersøkes, har valgt å ikke plassere programmering i et egent kapittel, slik som *Campus Matte 8*. Dette betyr at man vil finne POiM gjennom ulike delkapittel i *Matemagisk 8*. Ved å se på sammenhenger mellom det matematiske temaet og POiM i hvert kapittel, kan man se hvilke forutsetninger de får fra det matematiske temaet i møte med oppgaven. For *Campus Matte 8* må POiM sees i sammenheng med instruksjon og innhold fra kapittel med sammenhengende matematiske temaer. Dette vil kunne bidra til identifiseringen av krav fra *krav til elever*.

3.3.2.3 Utvidelse av vertikal analyse

I det opprinnelige rammeverket skapes det ikke datagrunnlag til forskningsspørsmål 3, jeg var derfor nødt til å utvide den vertikale analysen. Ved å se etter Csizmadia et al. (2015) seks *nøkkelbegrep* (*logikk*, *algoritmer*, *dekomposisjon*, *mønstre*, *abstraksjon* og *evaluering)*, kan jeg identifisere hvilke kvaliteter av algoritmisk tenkning POiM oppfordrer elever til. Nøkkelbegrepene er definerte i kapittel 2.5.1 av Csizmadia et al. (2015) med hensikt til POiM kan oppsummeres slik:

- *Logikk* – elevene trenger/bes om å bruke tidligere kunnskap for å skape mening til problemet og se for seg hvordan det kan skape en løsning.
- *Algoritmer* – elevene trenger/bes om å lage en plan som går igjennom steg for steg for hvordan oppgave kan løses, planen inneholder hva programmet skal gjøre og hvordan det håndterer matematiske operasjoner.
- *Dekomposisjon* – elevene trenger/bes om å dele opp problemet i ulike deler for å forstå helheten av det.
- *Mønster* – elevene trenger/bes om å se tilbake til tidligere løsninger av andre oppgaver eller teoretiske elementer for å indentifisere likheter eller mønster som kan bidra til finne løsningen.
- *Abstraksjon* – elevene trenger/bes om å plukke ut nødvendig informasjon fra oppgaven, gjennom tekst eller andre representasjon, for å gjøre problemet mer forståelig. Dersom unødvendig informasjon fjernes, vil løsningen enklere kunne oppdages.
- *Evaluering* – elevene trenger/bes om å reflektere over løsningen for å se om den er god nok eller kan forbedres (Csizmadia et al., 2015).

For å bestemme om en oppgaver er POiM var *arbeidsmåter* av Csizmadia et al. (2015) brukt i analysen (se kapittel 3.3.1.1). Både *arbeidsmåter* og *nøkkelbegrep* viser hvordan oppgaven utfordrer elever i algoritmisk tenkning og dermed viser hvilke krav som settes til elevene og plasseres derfor under *krav til elever* i vertikal analyse.

3.3.3 Oversikt over det utvidete og tilpassete rammeverket

Gjennom kapittel 3.3 har rammeverket blitt introdusert før det ble tilpasset og utvidet til dette studiet. Hensikten med dette kapitlet er å skape oversikt over det tilpassete rammeverket (se Tabell 6).

Tabell 6: Tilpasset og utvidet rammeverk for horisontal og vertikal analyse.

Horisontal analyse av læreverk

Bakgrunnsinformasjon	Struktur
○ Tittel.	○ Antall leksjoner og sider per lekskjon.
○ Antall sider.	○ Struktur av leksjoner.
○ Utgiver og publikasjonsdato.	○ Læringsmål dekt.
○ Tillegsmaterialet.	○ Rekkefølge av tema.

Vertikal analyse av læreverk

Formidling til elever	Krav til elever	Sammenhenger
Matematisk Innhold ○ Tema-spesifisert kontstruksjon og struktur. ○ Definisjoner, regler og konvensjoner. ○ Representasjoner gjennom illustrasjon. *Matematisk praksis* ○ Bearbeidet eksempler. ○ Modellere tekning.	Kognitive krav: - Memorering - Prosedyrer uten sammenheng. - Prosedyrer med sammenheng. - Gjøre matematikk. Type svar: - Kun svar. - Forklaring. - Vise gyldighet. Nøkkelbegrep: - Logikk. - Algoritmer. - Dekomposisjon. - Mønster. - Abstraksjon. - Evaluering. Arbeidsmåter: - Skape. - Fikle. - Feilsøke. - Samarbeide.	● Sammenhenger innenfor og mellom tråder.

Gjennom et tilpasset rammeverk for Charalambous et al. (2010) horisontale analyse av læreverkene, blir *Matemagisk 8* og *Campus Matte 8* undersøkt for *bakgrunnsinformasjon* og *struktur*. Forskningsspørsmål 1 besvares delvis ved å se på sammenhenger mellom temaer og læringsmål som dekker programmering. Andre elementer i tabellen fra den horisontale analysen er med på å skape en helhetlig oversikt over innholdet og strukturen av læreverket. Dette bidrar til å kartlegge hvor POiM for den vertikale analysen befinner seg i læreverkene. I den vertikale analysen undersøkes leksjoner, regler, eksempler og andre representasjoner for matematisk innhold og praksis fra *formidling til elever*. Dette fremstiller hva elevenes teoretiske grunnlag er før møtet med oppgaver, som også har en betydning for hvilke kvaliteter POiM identifiseres med gjennom *krav til elever*. Kravene som undersøkes er hvilke

kognitive krav fra Smith og Stein (1998), *type svar* inspirert av Charalambous et al. (2010) og *nøkkelbegrep* og *arbeidsmåter* fra Csizmadia et al. (2015). Til slutt undersøkes det for *sammenhenger innenfor og mellom tråder*. Både *formidling til elever* og *sammenhenger* brukes til å tolke innholdet i læreverkene for å klassifisere *krav til elever* som danner studiets grunnlag for besvarelsen av forskningsspørsmål 2 og 3.

3.3.4 Analyseguide

For å vise hvordan rammeverket brukes når jeg analyserer en oppgave, har jeg satt sammen en analyseguide (se Figur 3). Den er todelt og ser først på hvordan oppgaven kan knyttes til funn fra den horisontale analysen, deretter undersøkes oppgavens innhold og knyttes opp mot funn fra den vertikale analysen for å identifisere kvaliteter.

Analyseguiden:

1. Horisontal analyse:
    a. Innholdsanalyse av læreverkets bakgrunnsinformasjon.
    b. Innholdsanalyse av læreverkets struktur:
        i. I hvilken oppgavekategori er oppgaven under?
        ii. I hvilket kapittel og delkapittel er oppgaven under?
        iii. Har delkapittelet et læringsmål for programmering og knyttes oppgaven opp mot dette læringsmålet for programmering i matematikk?
            1. *Ja:* Oppgaven klassifiseres som en programmeringsoppgave inntil videre.
            2. *Nei:* Analysen stopper, dette er ikke en programmeringsoppgave eller har ikke et læringsmål for programmering i matematikk.
        iv. Hvilke leksjoner, videoleksjoner, dømer, eksempler, regler og tidligere oppgaver har læreverket gitt oss før oppgaven?
2. Vertikal analyse:
    a. Innholdsanalyse av oppgaven:
        i. Hva er det oppgaven spør om?
        ii. Kan oppgaven plasseres inn under en *arbeidsmåte*?
            1. *Ja:* Oppgaven klassifiserer som en programmeringsoppgave.
            2. *Nei:* Analysen stopper, dette er ikke en programmeringsoppgave.
        iii. Hvilke matematiske tema går oppgaven under?
        iv. Hva er det oppgaver gir oss av informasjon?
        v. Hvilken *type svar* krever programmeringsoppgaven?
        vi. Hvilke løsninger er mulige?
        vii. Har elevene gjort eller fått leksjon av en lignende oppgave tidligere?
        viii. Hvilke *nøkkelbegreper* kan gjenkjennes i programmeringsoppgaven?
        ix. Hvilke *kognitivt krav* har programmeringsoppgaven?

Figur 3: Analyseguide.

Analyseguiden viser at jeg starter med å gå igjennom funn fra den horisontale analysen (1). Her blir hvilket læreverk oppgaven er fra (1.a) og dens plass i læreverket først definert (1.b.i og 1.b.ii). Videre ser jeg etter læringsmål for programmering og om oppgaven kan knyttes opp mot dette (1.b.iii). Dersom oppgaven kan knyttes opp mot et eller flere av læringsmålene som er funnet, klassifiseres oppgaven midlertidig som en programmeringsoppgave. Den er bare godkjent midlertidig fordi det er to krav som må oppfylles for å avgjøre om det er POiM (se kapittel 3.3.1.1). Hvilke leksjoner eller andre representasjoner av matematisk innhold (1.b.iv) som er vist i læreverket før oppgaven som analyseres, brukes for å kartlegge innhold som er relevant å undersøke for å se hvor mye det støtter løsningen av oppgaven.

Den andre delen av analyseguiden, vertikal analyse (2), klassifiserer kvaliteter som oppgave inneholder (2.a). Først ser jeg etter hva det er oppgaven spør om gjennom oppgaveteksten (2.a.i). Her ser jeg på hva oppgaven ønsker at elevene skal gjøre, for eksempel om elevene skal lage eller gjøre endringer på et program eller i en kode. Informasjonen brukes til å se om oppgaven kan plasseres under en av *arbeidsmåtene* (2.a.ii) til *den algoritmiske tenkeren* av Csizmadia et al. (2015). Blir det ikke mulig, avsluttes innholdsanalysen av denne oppgaven, ellers klassifiseres oppgaven endelig som en POiM. Deretter undersøkes oppgaven for i hvilket matematiske tema (2.a.iii) oppgaven kan plasseres under basert på resultater fra den horisontale analysen. Videre kartlegges alle elementer av informasjon som oppgaven gir oss (2.a.iv). Informasjonen brukes som argumenter for hvilke kvaliteter jeg identifiserer i POiM. Hvilken *type svar* (2.a.v) som kreves blir identifisert etter Charalambous et al. (2010) sin modell tilpasset dette studiet. Det blir så undersøkt om det eksisterer flere mulige løsninger (2.a.vi) og om elevene har løst eller fått leksjoner av lignende oppgaver tidligere (2.a.vii). Denne informasjonen brukes som argumenter for begrunnelsen av POiM kvaliteter. Til slutt identifiseres både hvilke *nøkkelbegreper* (2.a.viii) fra Csizmadia et al. (2015) og hvilket *kognitivt krav* (2.a.ix) fra Smith og Stein (1998) denne POiM har ved hjelp av all informasjon funnet gjennom analyseguiden.

3.4 Koding av oppgaver

Sammen med valget av innholdsanalyse som metode i starten av kapittel 3.2, ble det klargjort at studiet bruker en kvalitativ tilnærmelse med kvantitative preg. I følge Grønmo (2016, s. 176) kategoriseres relevant innhold i gjennomførelsen av datainnsamlingen under kvalitativ innholdsanalyse. I dette studiet gjelder dette for kategoriseringen av matematiske temaer med programmerings læringsmål i den horisontale analysen, og for hvilke typer oppgavekategorier oppgavene befinner seg i. Siden disse ikke kan bestemmes på forhånd, blir matematiske temaer kodet i datamaterialet med første bokstaver i hvert ord som ikke er en konjunksjon og med store bokstaver. Oppgavekategorier skrives med navnet som defineres gjennom den horisontale analysen. For kvantitativ innholdsanalyse, sier Grønmo (2016, s. 214) derimot at kategoriseringen skjer i forkant av datainnsamlingen. Det vil si at elementene som POiM undersøkes for i dette studiet, er kategoriserte før analysen gjennomføres. Kvantitative analysen i denne studien skjer i den vertikale analysen, derfor er det nødvendig å lage koder for elementene som undersøkes her. I Tabell 7 er koder for Csizmadia et al. (2015) *arbeidsmåter* og *nøkkelbegrep*, Charalambous et al. (2010) *type svar* og Smith og Stein (1998) *kognitive krav* definert.

Tabell 7: Koder for den vertikale analysen.

Arbeidsmåter	**Koder**
Fikle	FIKLE
Skape	SKAPE
Feilsøke	FS
Samarbeide	TW
Nøkkelbegrep	**Koder**
Logikk	lo
Algoritmer	al
Dekomposisjon	de
Mønstre	mø
Abstraksjon	ab
Evaluering	ev
Type svar	**Koder**
Kun svar	S
Forklaring	F
Vise gyldighet	V
Kognitive krav	**Koder**
Memorering	M
Prosedyrer uten sammenhenger	PUS
Prosedyrer med sammenhenger	PMS
Gjøre matematikk	GM

Tabell 7 viser kodene som er fastsatt før den vertikale analysen. Kodene er fargekodet slik at de er lettere å lokalisere i skjemaet. Kodene brukt for *arbeidsmåter* vises i tabellen med blå bakgrunn. De skrives med store bokstaver, dersom de inneholder mer enn to stavelser skrives de med forkortelser. Kodene for *nøkkelbegrep* er farget oransje i tabellen og skrives kun med de to første bokstavene sammen med små bokstaver for å lettere skille dem fra de andre kodene. *Type svar* har fått fargen grønn og er kodet til kun første bokstav og store bokstaver. Til slutt er kodene for *kognitive krav* farget gul og sammensatt av første bokstav i hvert ord med store bokstaver. De fleste av kodene brukes kun i datamaterialet, mens *kognitive krav* brukes også når de er referert til videre i studiet.

Kodene belyst over, var plottet inn etter hvert som POiM vart analysert i *Microsoft Excel* (se Tabell 8). Innholdet i dette regnearket brukes senere i kapittel 4.3 for å vise resultatene fra studiets vertikale analyse etter opptelling.

Tabell 8: Utdrag fra den vertikale analysen hvor koder har blitt plassert i Excel.

	A	B	C	D	E	F	G	H	I	J	K	L
26	Læreverk	Matemagisk 8										
27	Kapittel	Algebraiske uttrykk og formler										
28	Delkapittel	Programmering med løkker										
29	Læringsmål	* Utforske og lage program i Python som gjenntar ein kodesekvens flere ganger.										
30		* Vurdere i hvilken situasjon det er lurt å bruke for-løkker, og når det er lurt å bruke while-løkker.										
31												
32	Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
33	Oppgaver	3.27b	AUF	FIKLE	lo						S	M
34	Oppgaver	3.28b	AUF	FIKLE	lo						S	M
35	Oppgaver	3.29b	AUF	FIKLE	lo						S	M
36	Oppgaver	3.29c	AUF	FIKLE	lo						S	M
37	Oppgaver	3.30b	AUF	FIKLE	lo						S	M
38	Oppgaver	3.30c	AUF	FIKLE	lo						S	M
39	Oppgaver	3.31b	AUF	FIKLE	lo						S	M
40	Oppgaver	3.31c	AUF	FIKLE	lo						S	M
41	Diskusjon	s. 112 a	AUF	TW	lo		de	mø	ab		F	PUS
42	Diskusjon	s. 112 b	AUF	TW	lo		de	mø	ab		F	PUS
43	Oppgaver	3.32b	AUF	FIKLE	lo						F	PUS
44	Oppgaver	3.33b	AUF	FIKLE	lo						F	PUS
45	Oppgaver	3.33c	AUF	FIKLE	lo						F	PUS
46	Diskusjon	s.112a	AUF	TW	lo		de	mø			F	PUS
47	Diskusjon	s.112b	AUF	TW	lo		de	mø			F	PUS
48	Diskusjon	s.112c	AUF	TW	lo		de	mø			F	PUS
49	Diskusjon	s.112d	AUF	TW	lo		de	mø			F	PUS
50	Diskusjon	s.114	AUF	TW				mø	ab		F	PUS
51	Oppgaver	3.34b	AUF	SKAPE	lo		de	mø			F	PUS
52	Nivå 1	3.35'	AUF	SKAPE	lo	al		mø			S	M
53	Nivå 1	3.36'	AUF	SKAPE	lo	al		mø			S	M
54	Nivå 2	3.37'	AUF	SKAPE	lo	al		mø			S	PMS
55	Nivå 2	3.38a	AUF	SKAPE	lo	al		mø			F	PMS
56	Nivå 2	3.38b	AUF	SKAPE	lo	al		mø			S	PUS
57	Nivå 2	3.38c	AUF	FIKLE	lo	al	de	mø			S	PUS
58	Nivå 2	3.38d	AUF	FIKLE	lo	al		mø			S	PUS

Tabell 8 viser to rammer som er omringet av kantlinjer som er separert med et mellomrom (31) og knyttes sammen med tilhørende informasjon. Øverst ramme består av rad 26 til 30 i tabellen. De viser generell informasjon hentet fra den horisontale analysen. Deres funksjon i dette regnearket er å vise i hvilke læreverk (26), kapittel (27) og delkapittel (28) POiM er plassert under. Det viser også delkapittelets læringsmål for programmering (29-30). Under denne rammen, rad 32 til 58, vises POiM som har blitt analysert gjennom den vertikale analysen. I de tre kolonnene uten farge til venstre (A - C) vises innholdet som ikke er forhands kodet. Fra venstre er hvilken oppgavekategori (A) POiM befinner seg under, så vises læreverkets oppgavenummer (B). Til slutt vises det matematiske temaet (C) POiM er plassert under. I dette utdraget er temaet kodet til *AUF* som er forkortelsen for kapittelet

gitt i rad 27. Til høgre for dette, vises *arbeidsmåter* (D) kodet med blått, *nøkkelbegrep* (E – J) i oransje, *type svar* (K) i grønt og *kognitive krav* (L) i gult.

I fire tilfeller gjennom den vertikale analysen vises det oppgavenummer med en apostrofe etter tallet, utdraget over i kolonne B og rad 52 til 54 inneholder slike situasjoner. Apostrofen indikerer at POiM består av mer enn en deloppgave, men er slått sammen til å telles som totalt en. Dette er fordi disse fire POiM sine deloppgaver baserer seg på å gjøre minimale endringer av løsningen de produserte i første deloppgave, som for eksempel å bytte om to tall. Kvalitetene til deloppgavene telles sammen og summeres til en POiM. Ved å telle alle deloppgavene til hver av disse tre oppgavene som en, vises de summerte resultatene fra den vertikale analysen mer presist. Dersom dette ikke gjøres i studiet, vil sluttresultatet inneholde repeterende POiM av lave kvaliteter som kan slå negativt ut på resultatene. Senere, i kapittel 4.2.3, vises det et eksempel hvor en oppgave med apostrofe i datamaterialet analyseres og som begrunner hvorfor den telles som bare en oppgave.

3.5 Validitet og reliabilitet

Postholm et al. (2018) skriver at studiers validitet og reliabilitet vil kunne være med på å styrke troverdigheten til forskningen som blir utført. I dette kapittelet drøfter jeg studiets validitet og reliabilitet.

3.5.1 Validitet

Begrepet validitet blir brukt til å belyse hvor relevant datamaterialet er for å besvare problemstillingen (Grønmo, 2016, s. 251; Postholm et al., 2018, s. 43-44). Postholm et al. (2018, s. 223) deler validitet inn i to typer, indre og ytre validitet. Indre validitet omhandler i hvor stor grad mine konklusjoner er gyldige i forhold til det som er studer. Ytre validitet omhandler derimot i hvor stor grad forskningen er generaliserbar. (Postholm et al., 2018, s. 223)

Den indre validiteten ser på om analyse, teori og begreper samsvarer med virkeligheten som studeres (Postholm et al., 2018, s. 229). Gjennom studiet blir teoretiske begreper og rammeverk fra tidligere forskning benyttet. *Kognitive krav* fra Smith og Stein (1998) task-

analysis guide og *nøkkelbegreper* fra Csizmadia et al. (2015) i *den algoritmiske tenkeren* inneholder begreper som problemstillingen spør etter i form av kvaliteter. Sist nevnte settes inn i Charalambous et al. (2010) horisontale og vertikale analyse, hvor *kognitive krav* eksisterer fra før, og til sammen danner et tilpasset og utvidet rammeverk. Her følger studiet en tilpasset analysemodell av den horisontale og vertikale analysen. Bakgrunnsinformasjon og struktur som blir undersøkt gjennom den horisontale delen gir svar på forskningsspørsmålet om matematiske temaer. Kvaliteter som *kognitive krav* og *nøkkelbegreper* undersøkes gjennom den vertikale delen og gir svar på forskningsspørsmålene om kvaliteter fra *kognitive krav* og *den algoritmiske tenkeren*. Til sammen besvares problemstillingen gjennom datamaterialet samlet fra rammeverket. Det er noen elementer i rammeverket som ikke direkte besvarer problemstillingen, men gir støtte til å definere hvilke kvaliteter som blir gitt. Smith og Stein (1998) *type svar* bidrar med å se etter *kognitive krav*. Charalambous et al. (2010) horisontale analyse innebærer å lokalisere leksjoner, dømer, regler og formler som i den vertikale analysen gir grunnlag for hvilke *kognitive krav* POiM har. Csizmadia et al. (2015) *arbeidsmåter* sammen med den horisontale analysen sin undersøking om læringsmål, bidrar til å klassifisere om en oppgave er en POiM. Studiets teoretiske begreper brukes derfor i stor grad til å besvare problemstillingen direkte eller indirekte gjennom støttende begrunnelser.

Studiets ytre validitet, som nevnt tidligere, omhandler hvor stor grad forskningen er generaliserbar. Det er to læreverk som er analysert i dette studiet, *Matemagisk 8* og *Campus Matte 8*. Utvalget er basert på mest brukte læreverk for programmering i matematikk besvart av 152 stemmer fra lærere og 26 ungdomsskoler. Utvalget i dette studiet kan derfor ikke generaliseres på landsbasis, men kan gi en pekepinn på hvilke læreverk som brukes mest. Læreverkene som undersøkes er kun for 8. trinn. Kvalitetene fra POiM i dette studiet kan derfor heller ikke generaliseres for alle POiM. Selv om resultatene ikke representerer alle læreverk og alle trinn, vil drøfting mellom forskningsspørsmålene kunne si noe om kvaliteter ved POiM. For eksempel, fordelingen av *kognitive krav* og *nøkkelbegrep* for hver av de matematiske temaene. Jeg ser på alle POiM fra disse læreverkene som representanter av POiM på 8. trinn. Til slutt, vil undersøkelsen av læreverk ikke kunne si noe om hvordan det blir brukt i praksis. Her kan lærere velge å bruke oppgavene på andre måter enn slik læreverkene har tilrettelagt dem. Analysen går lineært med læreverkenes oppbygging, altså

fra første til siste side. I en undervisningssituasjon vil lærere kunne endre i hvilken rekkefølge temaer og oppgaver blir gitt. Kvalitetene funnet i dette studiet, er derfor basert på hvordan læreverkene selv har valgt å strukturere innholdet deres.

3.5.2 Reliabilitet

Grønmo (2016, s. 242), Johannessen et al. (2021, s. 256) og Postholm et al. (2018, s. 222-223) omtaler reliabilitet som hvor pålitelig datamaterialet til studiet er. Ifølge dem, vil datamaterialet være pålitelig dersom det kan produseres like resultater om noen reproduserer studiets metode og data. Dette kan deretter måles ved hjelp av ulike tester for å definere studiets reliabilitet. Når det gjelder kvalitative studier, vil slike tester ikke være mulig å utføre da de ofte baserer seg på tolkninger (Grønmo, 2016, s. 248; Johannessen et al., 2021, s. 256). Dette er et kvalitativt studium med kvantitative preg. Her undersøkes POiM for ulike forhåndsdefinerte kvaliteter som identifiseres gjennom tolkning før de telles opp. Siden kvalitetene identifiseres gjennom tolkninger, vil det ikke være mulig å gjennomføre slike tester for å vurdere studiets pålitelighet. For å kunne øke påliteligheten i min studie, vil det derfor være viktig å reflektere over min egen påvirkning og synliggjøre forskningsprosessen slik at lesere kan reflektere over mine valg (Grønmo, 2016, s. 249; Postholm et al., 2018, s. 224).

Gjennom studiet, har jeg forholdt meg til å unngå å ta egne beslutninger uten å ha begrunnelser for det. Blant annet er datamaterialet basert på undersøkelser hvor skoler og lærere besvarer hvilke læreverk de bruker for å hente ut POiM. Dette gir studiet mer relevant datamaterialet for hva som faktisk brukes i skolene. Undersøkelsen baserer seg ikke på landsbasis, men på et utvalg av norske skoler og lærere. Den vil derfor ikke beskrive hvilke læreverk som bruker mest i hele Norge. Likevel, så dannes det et grunnlag for hvilke læreverk jeg ønsker å undersøke gjennom studiet. For kvalitetene som identifiseres gjennom analysen, er det egne tolkninger som produserer resultatene. Min tolkning av hver av kvalitetene kan påvirke om en oppgave blir klassifisert med en kvalitet eller en annen. Ved å definere og gi kriterier for ulike kvaliteter gjennom metoden, forhindrer jeg at valgene er gjort uten bestemte rammer og begrunnelser.

For å synliggjøre forskningsprosessen, har jeg gjennom metoden valgt å definere og vise kriterier til ulike elementer en oppgave i læreverket skal analyseres for. Dette ble til slutt satt sammen til i en analyseguide som brukes gjennom studiet. I kapittel 4.2 viser jeg fire ulike eksempler fra analysen, hvor jeg viser begrunnelser og argumenter for hvilke kvaliteter en POiM har ut ifra analyseguiden. Gjennom eksemplene blir det synliggjort hvordan forskningsprosessen foregår, og mine tolkninger er begrunnet basert på rammene satt fra metoden. Det siste eksempelet går igjennom en situasjon hvor en oppgave som innebærer å lage et program ikke klassifiseres som en POiM i dette studiet. Alle resultater fra analysen blir gitt som vedlegg. Dette gir andre en mulighet til å reflektere overfor valgene som er gjort i studiet i forhold til hvordan det er definert gjennom metoden.

3.5.3 Forskningsetikk

Postholm et al. (2018, s. 45) definerer forskningsetikk som etisk spørsmål og dilemmaer man kan møte på gjennom forskning. Disse omgår blant annet forskningsetiske retningslinjer, prinsipper og hva som gjelder som personopplysninger. Den nasjonale forskningsetiske komité for samfunnsvitenskap og humaniora (NESH) gir forskningsetiske retningslinjer. Disse kan oppsummeres som tre typer hensyn: selvbestemmelse og autonomi for informanten, respektere privatliv og unngå skader (Postholm et al., 2018, s. 45).

På grunn av at studiet baserer seg på å analysere læreverk, blir ingen enkeltpersoner berørt. Studiet inneholder derimot noen skjermklipp eller bilder av innhold fra læreverkene. For å bruke disse, har jeg sendt e-post til forlagene og fått tillatelse til å bruke bilder av innhold som er produsert av forfatterne. Dersom bildene skulle inneholde illustrasjoner eller bilder, som kunst, måtte jeg ha tillatelse fra rettighetshaver. I mitt tilfelle, inneholder ikke disse utklippene slike illustrasjoner eller bilder. I andre tilfeller, som bilder av modeller fra teori og tidligere forskning, har jeg produsert egne modeller for å presentere innholdet.

Gjennom studiet har jeg stilt meg nøytral til læreverkene. Jeg har ikke hatt noen formeninger eller fremstilt andre meninger underveis. Jeg ønsker ikke å drive markedsføring eller stille noe i et negativt lys. Funnene fremstilles derfor fra et nøytralt perspektiv. I tillegg så sammenligner ikke studiet læreverk opp mot hverandre, de brukes til å innhente data for

kvaliteter av POiM. Alle kilder som er brukt i teksten refereres til gjennom referansestilen APA 7.

4 Analysen

I dette kapittelet presenteres resultater fra den horisontale og vertikale analysen. Læreverkene presenteres hver for seg i den horisontale analysen for å systematisk belyse deres bakgrunnsinformasjon og struktur. Dette gir grunnlag for å besvare forskningsspørsmålet: «Hvilke matematiske temaer er POiM plassert under?». Resultatene fra den vertikale analysen av POiM settes sammen for å se på kvalitetene på tvers av læreverkene og gir grunnlag til besvarelsen av forskningsspørsmålene: «Hvilke kognitive krav stiller formulering av POiM til elever?» og «Hvilke nøkkelbegrep for algoritmisk tenkning kan man gjenkjenne gjennom arbeid med POiM?».

4.1 Resultater fra horisontal analyse

4.1.1 Bakgrunnsinformasjon

Matemagisk 8 er et læreverk for matematikk på 8. trinn av *Aschehoug*. Læreverket består av fysiske bøker sammen med noen nettressurser. *Aschehoug* ønsker med *Matemagisk* å legge til rette for at elever kan være aktive, oppdage og utforske sammenhenger gjennom matematikk (Kongsnes & Wallace, 2020). Dette studiet tar for seg læreverkets grunnbok *Matemagisk 8*. Læreverket har noen tilleggsressurser, blant annet en alternativ parallellbok med lettere innhold, arbeidsbok for hele ungdomstrinnet og en digital lærerveiledning. Studiet vil inkludere lærerveiledningen, fordi den inneholder en del undervisningsvideoer som læreverket anbefaler å bruke i undervisningen. Parallellboken blir ikke tatt med fordi den ikke er publisert før studiets start og arbeidsboken ekskluderes fordi det er en samling av fagstoffet fra grunnbøkene.

Campus Matte 8 er et digitalt læreverk for 8. trinn produsert av *Campus Inkrement*. Det er en heldigital nettressurs, hvor hele læreverket befinner seg på internett. Selve læreverket er utviklet i forhold til Kunnskapsløftet 2020 og vektlegger tilpasset opplæring og dybdelæring (Thue et al., 2022). Det baserer seg på omvendt undervisning, hvor elevene selv ser videoleksjoner før matematikk undervisning på skolen. Læreverket legger det opp på denne måten, slik at lærere og elever får mer tid i undervisningen til å diskutere og utforske matematikkoppgaver eller spørsmål. *Campus Matte 8* har ett kompendium som tilleggsmaterialet og kan åpnes gjennom læreverkets nettside. Siden kompendiet er ment

som en oppslagsbok elevene kan bruke som hjelp til å løse oppgaver er det inkludert i analysen. Å ta med kompendiet vil skape et helhetlig bilde av det totale læreverket. *Campus Matte 8* har ikke et egent dokument for lærerveileder, men de har et lærerpanel som bare lærere kan se. Her kan lærere se hva elevene har arbeidet med, hvor langt de har kommet og deres egenvurderinger. Man får tilgang til ulike kapittelprøver, diskusjonsoppgaver, undervisningsplanlegger og rapporter som viser statistikk for hele klassen. Diskusjonsoppgavene er oppgaver som læreren starter i fellesskap med klassen, hvor elevene går inn på en egen side i læreverket og deltar på diskusjonen digitalt. Disse oppgavene tas med i analysen selv om det bare er læreren som kan starte dem. Lærerveileden kan ikke skilles fra læreverket, det er derfor slått sammen med læreverket videre i den horisontale analysen. Tabell 9 viser informasjon som tittel, forfattere, utgiver, publikasjons år og antall sider for både grunnboken og lærerveiledningen til *Matemagisk 8* og nettressursen og kompendiet til *Campus Matte 8*. Betydningen av innholdet til radene forklares i kolonnen i midten markert med mørk grå. Innholdet under *Campus Inkrement sin* første kolonne kan derfor leses slik: *Campus Matte* 8 er tittelen til deres grunnbok og er skrevet av Bjørn Ove Thue, Rolf-Anders Moldeklev og Ole Thomas Røyland. Den er utgitt av *Inkrement AS* i 2020 og er en nettressurs som ikke kan identifiseres med sidetall. De fire objektene fra *Matemagisk 8* og *Campus Matte 8* analyseres videre for struktur gjennom den horisontale analysen.

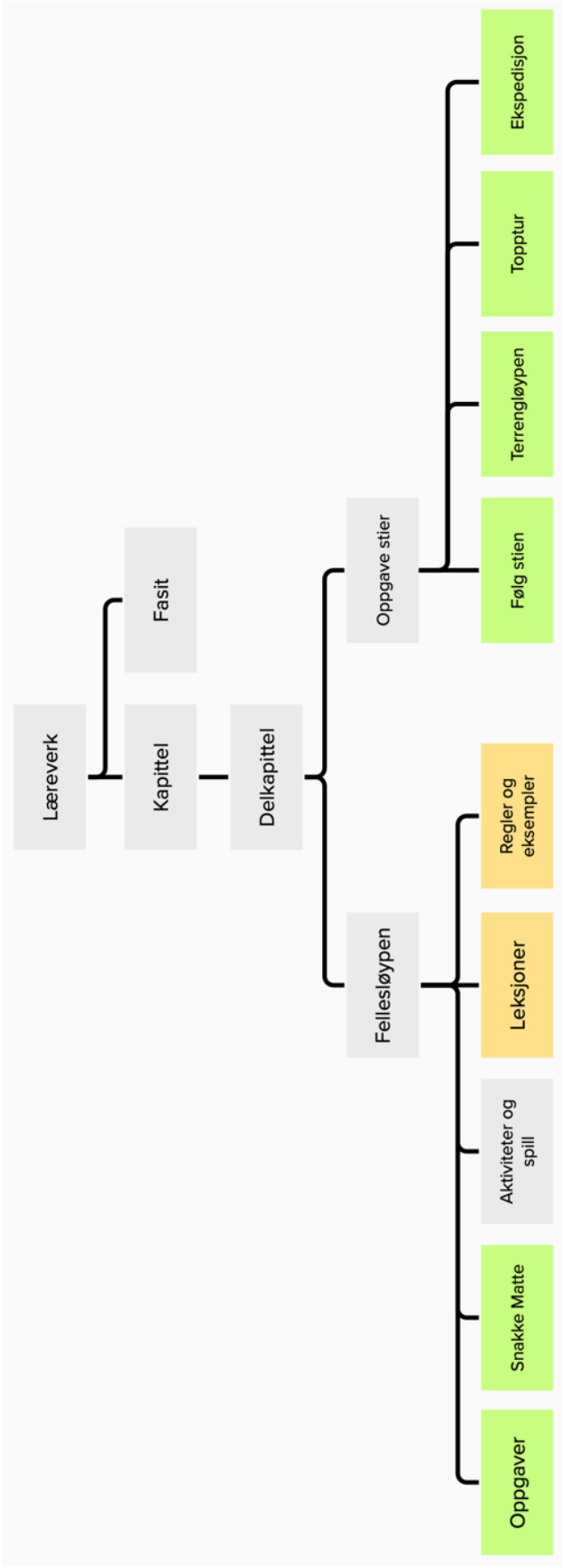
Tabell 9: Bakgrunnsinformasjon til Matemagisk 8 og Campus Matte 8.

Aschehoug			**Campus Inkrement**	
Matemagisk 8 (grunnbok)	Matemagisk 8-10 Digital lærerveiledning til bøkene	**Tittel**	Campus Matte 8 (grunnbok)	Campus Matte 8 Kompendium
Asbjørn Lerø Kongsnes Anne Karin Wallace	Aunivers	**Forfattere**	Bjørn Ove Thue Rolf-Anders Moldeklev Ole Thomas Røyland	Bjørn Ove Thue Rolf-Anders Moldeklev Ole Thomas Røyland
Aschehoug Undervisning	Aschehoug Undervisning	**Utgiver**	Inkrement AS	Inkrement AS
2020	2021	**Publikasjonsdato**	2020	2022
304 sider	Nettressurs	**Antall sider**	Nettressurs	90 sider

4.1.2 Struktur

Grunnstruktur av *Matemagisk 8* og *Campus Matte 8* presenteres hver for seg via slektstre. De begynner på toppen av figurene og går nedover via linjer. Dersom en seksjon (en rute i slektstreet) ikke har videre linjer som går nedover, eksisterer det ikke videre seksjonering. I tillegg er seksjonene fargekodet. En grå bakgrunn har to betydninger. Den første betydningen er at denne seksjonen kan føres videre til en eller flere underseksjoner. Den andre betydninger er dersom det ikke finnes videre seksjonering, så er denne seksjonen ikke interessant for innholdsanalysen i dette studiet. Det er to farge alternativer for endeseksjoneringer, grønn eller oransje. Dersom en seksjon har oransje bakgrunnsfarge, er dette en seksjon som undersøkes i den horisontale analysen. Om seksjonen har grønn bakgrunnsfarge, tilhører seksjonen den vertikale analysen. Jeg går deretter igjennom felles trekk fra grunnstrukturen som danner grunnlaget for hvilke oppgaver som skal analyseres gjennom den vertikale analysen. Til slutt viser jeg inndelingen av strukturen til kapitler og leksjoner for begge læreverkene fra grunnstrukturen.

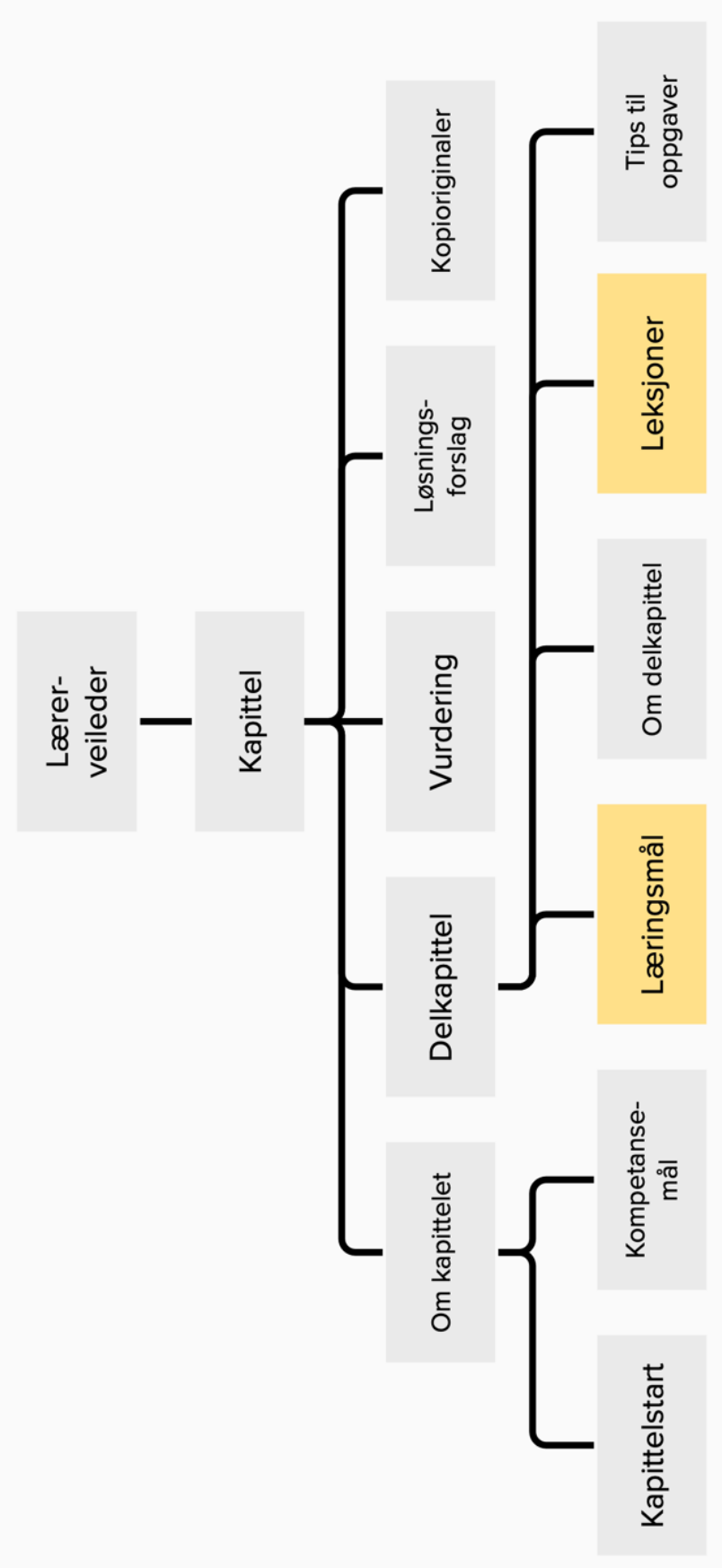
4.1.2.1 Grunnstruktur Matemagisk 8 – Grunnbok.



Figur 4: Grunnstrukturen til læreverket Matemagisk 8.

Figur 4 viser at læreverket *Matemagisk 8* deler seg opp i kapiteler og en felles fasit bakerst i boken. Hvert kapittel deler seg opp i delkapitler. Inni hvert delkapittel, finnes det en *fellesløype* og opp til fire oppgave stier. *Fellesløypen* inneholder ulike *oppgaver*, diskusjonsoppgavene *Snakke Matte*, *aktiviteter og spill*, *leksjoner* og *regler og eksempler*. *Fellesløypen* er utviklet for arbeid i fellesskap med hele klassen (Kongsnes & Wallace, 2020). *Oppgaver* og *Snakke Matte* under *fellesløypen* er fargekodet med grønn og er oppgaver som er analysert i den vertikale analysen. *Leksjoner* og *regler og eksempler* under *fellesløypen* er fargekodet oransje og undersøkes i den horisontale analysen. Det eksisterte ingen aktiviteter og spill for programmering i læreverket og den er derfor kodet grå. Under delkapittel, splittes det ut til oppgave stier. De fire oppgavestiene er *følg stien*, *terrengløypen*, *topptur* og til slutt *ekspedisjon*. Dette er læreverkets nivådifferensierte oppgaver hvor elevene kan selv velge hvor de skal arbeide etter vanskelighetsgrad. Disse oppgavene utgir en stor del av alle oppgavene i læreverket som skal analyseres i den vertikale analysen og er derfor fargekodet grønn. *Følg stien* og *terrengløypen* eksisterer bakerst i hvert delkapittel og *topptur* i siste delkapittel i hvert kapittel. *Ekspedisjon* finnes bare i fire delkapittel.

4.1.2.2 Grunnstruktur Matemagisk 8 - lærerveileder

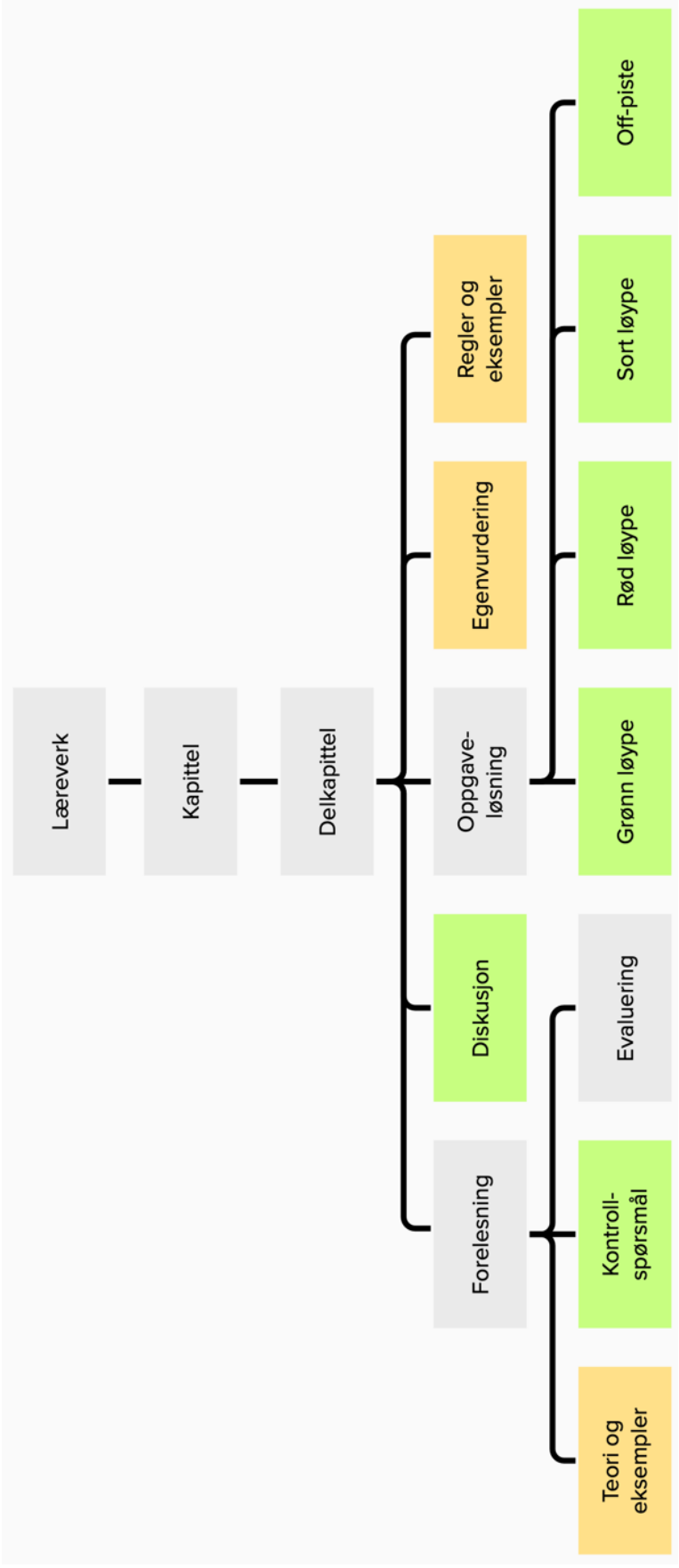


Figur 5: Grunnstruktur til Matemagisk 8-10 sin lærerveiledning.

Figur 5 viser grunnstrukturen til den digitale lærerveiledningen for *Matemagisk 8*. For hvert kapittel i grunnboken, finnes det et kapittel i lærerveilederen. Her får læreren tilgang til informasjon *om kapittelet*, *delkapitler*, *vurdering*, *løsningsforslag* og *kopioriginaler*. Under *om kapittelet*, får man se informasjon som formålet, tips til gjennomføring og hva man skal være oppmerksom på for hvert kapittel i boken. Her får vi også se ulike kompetansemål som blir dekt. Under *delkapitler*, ser vi læringsmålene deres, informasjon om delkapittelet, leksjoner og tips til oppgaver. *Læringsmål* er kodet til oransje og utdypes i den horisontale analysen. I *om delkapittel* får vi en beskrivelse over hva elvene skal jobbe med og hvilket kopioriginaler som er knyttet til dette delkapittelet. I *leksjoner* under delkapitler, er det flere videoleksjoner som er ment som støtte til undervisning. Disse videoleksjonene spiller en viktig rolle i analysen til studiet og er derfor kodet til oransje. Til slutt under delkapitler, gis tips til oppgaver, informasjon som formål, hva man skal være oppmerksom på og tips til gjennomføringen av oppgaver. Tilbake i kapittel så splitter det seg til *vurdering*, her finner man ulike kapittelprøver og fasit til dem. *Løsningsforslag* gir et detaljert forslag til løsninger av oppgavene i hvert kapittel. *Kopioriginaler* inneholder ulike dokument som kan skrives ut og deles ut til elever under aktiviteter eller spill.

I lærerveilederen til *Matemagisk 8*, forteller de litt om hvorfor de bruker programmeringsspråket *Python* i deres læreverk. Blant annet nevner de at dette er antakeligvis elvenes første møte med tekstprogrammering med *Python* i skolen (Aunivers, 2021). Læreverket ønsker derfor å bruke tid på å lære seg grunnleggende koder for tekstprogrammeringsprogrammet. Dette medfører at noen av programmeringsoppgavene som elevene møter, kan ha et større fokus på å lære seg programmering overfor matematikk. Læreverket argumenterer for hvorfor de gjør dette med at de ønsker å forberede elevene til programmeringen de kan møte når de begynner på videregåendeskole (Aunivers, 2021). *Matemagisk 8* sin beslutning om å introdusere tekstbasertprogrammering, overfor blokkprogrammering som elevene kan ha kjennskap til fra tidligere skoleår, vil kunne prege kvaliteten på nivået av *kognitive krav* til POiM. Denne oppgavestrukturen fører til at POiM som er skapt for å introdusere eller utforske programmeringsstrategier vil kunne ha mindre fokus på å utforske matematikk gjennom programmering. Det vil derfor kunne oppstå flere oppgaver av lavere kognitive krav.

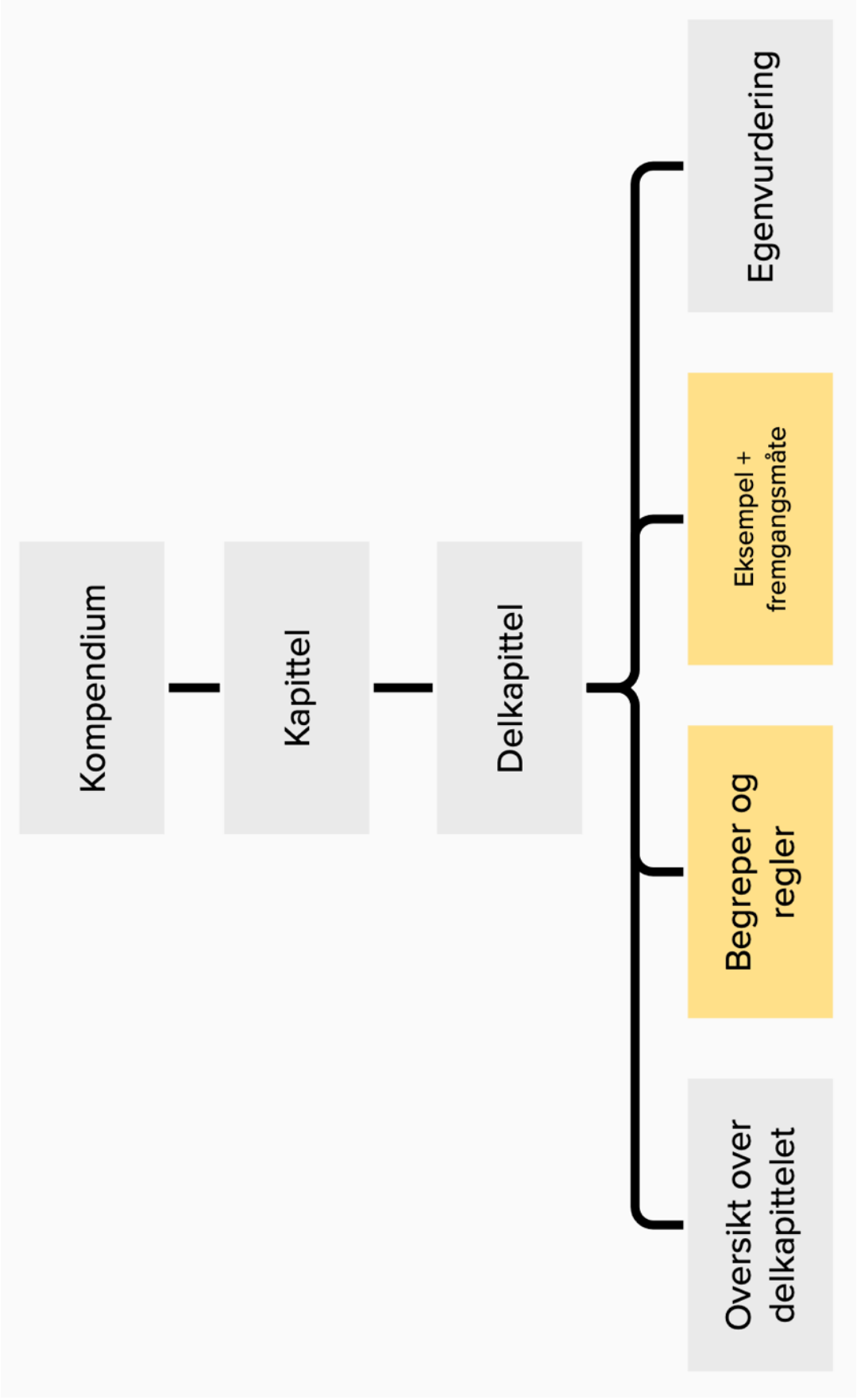
4.1.2.3 Grunnstruktur Campus Matte 8 - Nettressurs



Figur 6: Grunnstrukturen til læreverket Campus Matte 8.

Figur 6 viser hvordan læreverket *Campus Matte 8* på toppen deler seg inn i ulike seksjoner nedover. I læreverket finner man *kapittel* som går til *delkapittel*. Læreverket har valgt å kalle delkapitlene for leksjoner. For å forhindre at disse blandes med forelesningsleksjoner eller videoleksjoner videre i studiet, har jeg valgt å referere til dem som *delkapittel* istedenfor leksjoner. Hvert *delkapittel* deler seg opp i flere grener. *Forelesning* som igjen deler seg opp i *teori og eksempler*, *kontrollspørsmål* og *evaluering*. *Teori og eksempler* er videoleksjoner laget av læreverket og skal sees av elevene før undervisning. Disse leksjonene danner en del av grunnlaget for hva elevene har lært før de løser oppgaver til hvert delkapittel og er kodet til oransje. *Kontrollspørsmål* er en type oppgave som elevene får etter en videoleksjon, slik får elevene teste ut om de har forstått videoleksjonen. *Kontrollspørsmålene* blir analysert i den vertikale analysen til studiet og er derfor kodet til grønn. Til slutt i forelesingen får elevene en evalueringsseksjon. Her skal de rangere hvor forståelig leksjonen var og skrive ned eventuelle spørsmål de har til læreren. Spørsmålene vil da kunne være med på å lage en diskusjon rundt et tema i matematikkundervisningen på skolen. Den andre grenen *delkapittel* deler seg opp i er *diskusjon*. Dette er diskusjonsoppgavene som er tilgjengelig for læreren å starte fra lærerpanelet og er kodet grønn fordi de analyseres i den vertikale analysen. Tredje grenen til *delkapittel*, er *oppgaveløsning*. Her deler det seg igjen opp til fire ulike løyper som er *Campus Matte 8* sine nivådifferensierte oppgaver. Oppgavene fra disse løypene inneholder mesteparten av oppgavene som skal inkluderes i den vertikale analysen og er derfor kodet grønn. *Delkapittel* deler seg også opp til *egenvurdering*, hvor elevene selv vurderer hvor bra de utfører læreverkets læringsmål. Dette gjør de ved å rangere etter farger hvor rødt er dårlig, gult er middels og grønt er bra. *Campus Matte 8* oppgir ikke spesifikt hva deres læringsmål er, men *egenvurderingene* vil i dette studiet telles som læringsmål. Dette er fordi de formuleres som: «Jeg kan (gjøre noe matematisk)» og gir en pekepinn for hva de skal lære i hvert delkapittel og er derfor kodet oransje. Til slutt splittes *delkapittel* ut til *regler og eksempler* som også er kodet oransje. Her får elevene en kort oppsummering av innholdet til tilleggsmaterialet *kompendiet*.

4.1.2.4 Grunnstruktur Campus Matte 8 - Kompendium



Figur 7: Grunnstruktur til Campus Matte 8 sitt kompendium.

Figur 7 viser grunnstrukturen til *Campus Matte 8* sitt kompendium. *Kompendiet* deler seg opp i *kapittel* og så *delkapittel*. Under hvert *delkapittel*, er det en oversikt over delkapittelet som kort forteller hva elevene skal lære i dette delkapittelet. Her finner vi også *begreper og regler* sammen med *eksempler* som viser fremgangsmåten. Disse seksjonene utgir deler av det som er elevenes teoretiske grunnlag før de møter oppgaver i læreverkene og er derfor kodet oransje. Til slutt viser også *kompendiet* de ulike punktene for *egenvurdering*.

4.1.2.5 Oppgavekategorier

Gjennom presentasjonene av grunnstrukturen til *Matemagisk 8* sin grunnbok og lærerveileder og *Campus matte 8* sin nettressurs og kompendiet, vises det blant annet hvor oppgaver er fordelt i læreverkene. For å belyse dette mer, skal jeg utdype seksjonene som inneholder oppgaver og presentere hvordan læreverkene omtaler dem og deres hensikt. Til slutt samles likheter ved oppgavenes strukturelle trekk og danner grunnlaget for hvilke oppgaver som undersøkes i den vertikale analysen.

Aschehoug som har produsert *Matemagisk 8*, har i de første sidene valgt å introdusere boken med velkomstord og inkludert noen korte forklaringer som viser ulike oppgaver elevene vil møte på i boken. Kongsnes og Wallace (2020) skriver at deres læreverk legger til rette for at elevene skal få kunne være aktive, utforske og oppdage sammenhenger i matematikk. Elevene skal få snakke matematikk, bli gode problemløsere og utvikle forståelse. Til slutt nevner de at boken gir varierte oppgaver som er relevante for dagens samfunn rundt oss (Kongsnes & Wallace, 2020, s. 4). Kongsnes og Wallace (2020, s. 4 - 5) beskriver *Matemagisk 8* sin struktur for oppgaver via løyper:

- *Fellesløype* – en løype som omfatter teori, eksempler, matematiske samtaler, oppgaver, utforskende oppgaver, programmering, aktiviteter og spill som blir gjort i fellesskap.
- *Snakke Matte* – oppgaver hvor elever skal diskutere og snakke matematikk sammen. Her trener elevene på å forklare hvordan de tenker. Oppgavene finnes i *fellesløypen*.

- *Følge stien* – oppgaver som egner seg for trening og repetisjon av det som blir gått igjennom i felleskap fra *fellesløypen*. Her trener elevene på en ting av gangen.
- *Terrengløypen* – oppgaver som er skapt for å bygge videre på innholdet fra *fellesløypen*. Her trener elevene på en eller flere ting av gangen.
- *Topptur* – inneholder oppgaver som er svært utfordrende og er over forventningsnivået til deres trinn. *Topptur* er en løype for de som mestrer *terrengløypa* godt.
- *Ekspedisjon* – oppgaver som gir svært god trening i abstraksjon, generalisering og avansert problemløsning. Oppgavene er langt over forventningsnivået til deres trinn og eksisterer bare i fire kapittel (Kongsnes & Wallace, 2020, s. 4 - 5).

Løypene er tilsynelatende inspirert av navn som man forbinder med det å gå på tur, med unntak av *Snakke Matte* som eksisterer i *fellesløypen*. De fire siste løypene inneholder oppgaver av ulik vanskelighetsgrad og brukes som læreverkets nivådifferensierte oppgaver. I likhet med vanskelighetsgraden av ulike turer, er oppgavene gitt i de ulike løypene vanskeligere dess vanskelig turen til løypen er.

Det andre læreverket i dette studiet er *Campus Matte 8*. Når man går inn på læreverket får man en kort beskrivelse av det. Thue et al. (2022) forteller at læreverket er utviklet i henhold til LK20 og vektlegger tilpasset opplæring og dybdelæring. De beskriver også at det enkelt gir alle elever et løft i matematikk. Inne på læreverket, får man opp ulike seksjoner som inneholder diverse matematikk relaterte objekter. Thue et al. (2022) beskriver *Campus Matte 8* sine seksjoner for oppgaver som:

- *Forelesning* – her kan man se korte videosnutter som omhandler oftest teori og eksempler som en gjennomgang av det aktuelle matematikktemaet. Det er også kontrollspørsmål som tester om elevene har forstått videoforelesningene.
- *Diskusjon* – en klasseromsaktivitet som lærer setter i gang en diskusjon via det digitale læreverket som elevene blir med på ved hjelp av deres egne digitale verktøy. Her ser elevene på ulike matematikkproblem og velger et svaralternativ

fra en liste eller skriver inn svaret som de trur er korrekt. Svarene blir videre diskutert i klasserommet.

- *Grønn løype* – inneholder enklere oppgaver som er rettet for elever som synes matematikktemaet er vanskelig.
- *Rød løype* – oppgaver av middels vanskelighets grad som er ment for elever som ønsker litt mer utfordring i forhold til *grønn løype*.
- *Svart løype* – oppgaver som er utfordrende og er for elever som mestrer det aktuelle matematikktemaet.
- *Off-piste* (*gul løype*) – oppgaver som er langt over forventningsnivået til læreverkets trinn og som ikke nødvendigvis har blitt tilrettelagt nødvendig teori for å kunne løses (Thue et al., 2022).

Forelesning er en stor del av læreverkets omvendt undervisning. Læreverket gir grunnlag for at elevene skal gjennomføre denne seksjonen som hjemmelekse, slik at elever og lærere kan bruke tiden på skolen til samtale, oppgaveløsning og diskutere det som er vanskelig i matematikken (Thue et al., 2022). Elevene blir ofte oppfordret til å løse de ved hjelp av blyant og papir før de kan skrive inn svaret deres i læreverket for å få kontrollert om det er korrekt eller ikke (Thue et al., 2022). De fire siste seksjonene har blitt fargekodet som viser vanskelighetsgraden til oppgaven. Selve fargekodesystemet kan minne om ulike vanskelighetsgrader for skiløyper. Vanskeligere skiløype i form av fargekode er mer utfordrende, slik er det og for de ulike løypene til læreverket. Dette er *Campus Matte 8* sin måte å vise nivådifferensiering i oppgavene.

Både *Matemagisk 8* (Kongsnes & Wallace, 2020) og *Campus Matte 8* (Thue et al., 2022) har en del likheter ved deres generelle struktur. Begge læreverkene har tydelig markert nivådifferensiering av oppgaver ved hjelp av en form for tur representasjon. Selve kategoriene til oppgavene kan sammenlignes. De enklere oppgavene, som er mer en form for repetisjon av det som har blitt gått igjennom via forelesing eller video, ligger i *følg stien* hos *Matemagisk 8* og *grønn løype* hos *Campus Matte 8*. Oppgavene som er både repeterende og tar et steg videre vil man finne i *terrengløypen* og *rød løype*. Oppgaver som er utfordrende, krever at elevene mestrer de nedre nivåene og kan være over forventningsnivået til deres trinn, ligger i *topptur* og *svart løype*. Til slutt, det som ligger i

ekspedisjon og *off-piste* (*gul løype*) er oppgaver av den høyeste vanskelighetsgrad. Disse oppgavene er langt over forventningsnivået og er utfordrende for elevene på det aktuelle trinnet. I tillegg inneholder begge læreverkene oppgaver som ligger i en diskusjons kategori. *Matemagisk 8* har sine *Snakke Matte* oppgaver hvor elevene deltar i klasseaktivitet og diskuterer matematikk. *Campus Matte 8* har også en lignende aktivitet på sitt digitale læreverk, som de henviser til som *diskusjon*.

Læreverkene har ulik tilnærming til den teoretiske og gjennomgangs-delen av matematikkinnholdet. *Matemagisk 8* minner muligens mer om en tradisjonell klasseromsundervisning, hvor man forklarer matematikken og viser eksempler. *Campus Matte 8* derimot, som er bygd på omvendt undervisnings-prinsippet, etterlater denne delen til elevene selv. Allikevel danner *Campus Matte 8* sin *forelesning* og *Matemagisk 8* sin *fellesløype* grunnlaget for matematikken elevene skal kunne. I disse delene av læreverkene finnes det også oppgaver for elevene. Siden slike oppgaver ikke er farge kodet eller spesifikt kategorisert til et nivå av vanskelighetsgrad, må disse oppgavene kategoriseres for seg selv.

Tabell 10: Felles benevnelser for oppgaver gitt i Matemagisk 8 og Campus Matte 8.

Matemagisk 8		**Campus Matte 8**
Fellesløypen	**Oppgaver**	Forelesning
Snakke Matte	**Diskusjon**	Diskusjon
Følg stien	**Nivå 1**	Grønn løype
Terrengløype	**Nivå 2**	Rød løype
Topptur	**Nivå 3**	Svart løype
Ekspedisjon	**Nivå 4**	Off-Piste

Tabell 10 viser at fellesbenevnelsen for oppgaver i fellesløypen og forelesning er satt til *oppgaver*. Snakke matte og diskusjon nevnes fra nå som *diskusjon*. Følge stien og grønn løype utgir oppgaver av *nivå 1*. Terrengsløype og rød løype er oppgaver av *nivå 2*. Topptur og svart løype er oppgavekategorier som inneholder oppgaver av *nivå 3*. Til slutt er oppgaver under ekspedisjon og off-piste fra nå av nevnt *som nivå 4*. Dette er oppgavekategorier som inneholder de oppgavene som analyseres gjennom den vertikale analysen til studiet.

4.1.2.6 Struktur av kapitler og læringsmål

Matemagisk 8 består av 10 kapittel med opp til 4 delkapitler hver. I Tabell 11 er hvert kapittel med deres delkapittel presentert i rekkefølge. I tillegg er antall læringsmål oppgitt sammen med, om de eksisterer, hvor mange som inneholder programmering i parentes.

Tabell 11: Kapittel, delkapittel og antall læringsmål i Matemagisk 8.

Kapittel	**Delkapittel**	**Læringsmål (programmering)**
1 Hele tall	1A Regnestrategier	4
	1B Variabler og egenskaper ved multiplikasjon	2
	1C Primtall og faktorisering	3
	1D Negative tall	2
2 Brøk og desimaltall	2A Brøk	4
	2B Desimaltall	3
	2C Måleenhet	1
	2D Programmering i Python	2 (2)
3 Algebraiske uttrykk og formler	3A Verdien av algebraiske uttrykk	1
	3B Praktiske situasjoner	3
	3C Programmering med løkker	2 (2)
	3D Figurtall	3 (1)
4 Potenser, kvadratrøtter og regnerekkefølge	4A Potenser og kvadratrøtter	4 (1)
	4B Regnerekkefølge	2
5 Algebra og likninger	5A Algebra og likninger	2
	5B Algebraisk løsningsmetode for likninger	2
	5C Likninger i praktiske situasjoner	2
6 Parenteser og likninger	6A Parenteser i algebraiske uttrykk	2
	6B Likninger med brøker og parenteser	1
	6C Å løse likninger med programmering	1 (1)
7 Hva er en funksjon?	7A Funksjonsmaskiner	4 (1)
8 Grafen til en funksjon	8A Koordinatsystem	1
	8B Å tegne grafen til en funksjon	4
9 Lineære funksjoner	9A Lineære funksjoner i praktiske situasjoner	4
	9B Å utforske grafen til lineære funksjoner	2
10 Sammensatte målenheter	10A forholdstrekanten	3
	10B gjennomsnittsfart	1
Totalt	**27**	**65 (8)**

Tabell 11 leses langs rader og betydningen står i den øverste raden. Kapittel 4 *Potenser, kvadratrøtter og regnerekkefølge* leses dermed som å inneholde delkapittel 4A *Potenser og*

kvadratrøtter med fire læringsmål hvor et er for programmering. Det inneholder også delkapittel 4B *Regnerekkefølge* med to læringsmål hvor ingen er for programmering. Totalt er det 10 kapittel med 27 delkapitler med til sammen 65 læringsmål hvor 8 av dem inneholder programmering. Det er 6 delkapittel som inneholder læringsmål i programmering hvor 3 av dem er eksplisitt knyttet til programmering gjennom deres tittel (delkapittel 2D, 3C og 6C).

Funnene viser at kapittel *Brøk og desimaltall*, *Algebraiske uttrykk og formler*, *Potenser, kvadratrøtter og regnerekkefølge*, *Parenteser og likninger* og *Hva er en funksjon?* inneholder læringsmål om programmering i *Matemagisk 8*. Videre skal kapittel 7 *Hva er en funksjon?* refereres til som det matematiske temaet *funksjoner*.

I Tabell 12 vises sammenheng mellom kapitler og tilknyttet læringsmål for programmering. For eksempel, det matematiske temaet *funksjoner* er et tema som *Matemagisk 8* har læringsmålet å «lage et Python-program som fungerer som en funksjonsmaskin» (Kongsnes & Wallace, 2020, s. 202). Dette temaet, sammen med de fire andre fra tabellen, inkluderes i listen av *matematiske temaer* som POiM kan plasseres under og delvis besvarer det første forskningsspørsmålet.

Tabell 12: Kapittel med læringsmål i programmering fra Matemagisk 8.

Matematisk tema/ Kapittel	**Læringsmål i programmering**
Brøk og desimaltall	● Utforske og lage en algortime for et program som utfører en regneoperasjon. ● Bruke Python og lage et kalkulatorprogram som henter verdier fra brukeren, utfører en regneoperasjon og skriv svaret til skjermen.
Algebraiske uttrykk og formler	● Utforske og lage program i Python som gjenntar ein kodesekvens flere ganger. ● Vurdere i hvilken situasjon det er lurt å bruke for-løkker, og når det er lurt å bruke while-løkker. ● Bruke programmering og regneark til å beskrive og skrive ut tallmønster.
Potenser, kvadratrøtter og regnerekkefølge	● Bruke programmering og regneark til å skildre og skrive ut tallmønster.
Parenteser og likninger	● Lage, forbedre og programmere algoritmer som løser ligninger.
Funksjoner	● Lage et Python-program som fungerer som en funksjonsmaskin.

Campus Matte 8 består av 8 kapittel med opptil 10 delkapittel hver. I Tabell 13 er deres kapitler satt opp i rekkefølge sammen med antall delkapittel og læringsmål. I tillegg er læringsmålene som inneholder programmering oppgitt i parenteser etter antall læringsmål. Jeg har valgt å ikke oppgi navnet på alle delkapitlene da det er totalt 60 stykk, med unntak av de som inneholder programmering.

Tabell 13: Kapittel, delkapittel og antall læringsmål i Campus Matte 8.

Kapittel	**Delkapittel**	**Læringsmål (programmering)**
1 Tall	1.1 – 1.8	30
2 Regning	2.1 – 2.10	34
3 Brøk	3.1 – 3.9	27
4 Potenser	4.1 – 4.9	31
5 Likninger og algebra	5.1 – 5.9	24
6 Funksjoner	6.1 – 6.5	15
7 Mål og enheter	7.1 – 7.8	27
8 Programmering	8.9 Kalkulatorer	3 (3)
	8.10 Problemløsning	3 (3)
Totalt	**60**	**194 (6)**

Tabell 13 viser at hvert kapittel har mellom 2 og 10 delkapitler med til sammen 60 delkapittel på 8 kapittel. Når det gjelder læringsmål er det totalt 194 stykker hvor 6 av dem inneholder programmering. I læreverket så er alt programmerings relatert blitt plassert i et eget kapittel. Dette kapittelet ser vi i tabellen som kapittel 8 *Programmering* og der er det bare to delkapittel som tilhører 8 trinn. I selve læreverket består programmerings kapittelet av 10 delkapittel som går fra 5. til 8. trinn. Det blir ikke oppgitt informasjon om hvorfor læreverket har valgt å inkludere delkapitlene av programmering for lavere trinn. En mulig årsak til dette kan tenkes at er overgangen til LK20. Med mer vektlegging av programmering i denne overgangen, kan det hende at elever ikke har nådd tidligere trinns kompetansemålene i programmering fordi overgangen skjedde etter de har fullført disse års trinnene. Som et resultat av dette kan det hende at *Campus Matte 8* valgte å inkludere delkapittel i programmering for lavere trinn, slik at elevene har muligheten til å hente seg inn igjen. Basert på dette og oppgavens problemstilling og avgrensning, oppgis bare de to

delkapitlene for programmering på 8. trinn da det er programmeringsoppgavene fra dette trinnet som skal undersøkes.

Oppgavene i programmeringskapittelet til *Campus Matte 8* kan ikke knyttes til et spesifikt matematisk tema gjennom navnet til kapittelet, slik som det gjøres hos *Matemagisk 8*, derfor må oppgavene i disse delkapitlene analyseres for å se om de kan plasseres i et eksisterende tema eller under et nytt (se kapittel 3.3.1.2). Eksisterende matematiske temaer sammenlignes med kapitler og delkapitler i *Campus Matte 8*. Tabell 14 viser mulige temaer en POiM kan plasseres under fra begge læreverk. Dersom et kapittel inneholder delkapitler som kan plasseres i flere temaer, henviser parentesene til det aktuelle delkapittelet. For eksempel eventuelle POiM under *Brøk og desimaltall* fra *Matemagisk 8* og *Brøk* og *Desimaltall og tallinjen* delkapittelet fra *Tall* kapittelet i *Campus Matte 8* kan plasseres under temaet *brøk og desimaltall.*

Tabell 14: Alle mulige matematiske temaer i Matemagisk 8 og Campus Matte 8.

Tema	Matemagisk 8	Campus Matte 8
Brøk og desimaltall	Brøk og desimaltall	Brøk
		Tall (Desimaltall og tallinjen)
Algebraiske uttrykk og formler	Algebraiske uttrykk og formler	Likninger og algebra
Potenser, kvadratrøtter og regnerekkefølge	Potenser, kvadratrøtter og regnerekkefølge	Potenser
		Tall (Regnerekkefølge)
Parenteser og likninger	Parenteser og likninger	Likninger og algebra
		Tall (Parenteser)
Funksjoner	Funksjoner	Funksjoner
Tall		Tall
Regning		Regning
Mål og enheter		Mål og enheter

POiM fra programmeringskapittelet i *Campus Matte 8* kan plasseres under de eksisterende matematiske temaene *brøk og desimaltall*, *algebraiske uttrykk og formler, potenser, kvadratrøtter og regnerekkefølge, parenteser* og *likninger* og *funksjoner* eller de nye temaene *tall, regning* og *mål og enheter*. Hvilke matematiske temaer som POiM fra

læreverkene er identifisert med avgjøres gjennom den vertikale analysen og prosessen vises gjennom eksempelet i kapittel 4.2.1. Resultatene for endelige temaer vises i kapittel 4.3.

4.1.2.7 Struktur av leksjoner og sider i læreverkene

For å få en oversikt over programmerings relatert innhold i læreverkene som gir elevene støtte og et teoretisk grunnlag til å løse POiM de møter, skal jeg nå presentere funnene som viser leksjoner og antall sider sammen med videoleksjoner og deres varighet under. Disse har en viktig rolle i identifiseringen av kvalitetene til POiM. Først presenteres funnene fra *Matemagisk 8* og deretter funnene fra *Campus Matte 8*.

Matemagisk 8 har leksjoner spredt gjennom deres *fellesløype* i hvert delkapittel. De har også leksjoner i lærerveilederen under hvert delkapittel. I Tabell 15 er antall leksjoner i grunnboken, videoleksjoner i lærerveilederen og antall sider per kapittel presentert. Antall sider representerer antallet for *fellesløype*. Ved å ekskludere alle andre sider fra opptellingen, skapes et mer presist overblikk over antall sider som inneholder eksempler, regler og dømer. Tall oppgitt i parenteser er hvor mange leksjoner eller sider som inneholder programmering. Under videoleksjoner i lærerveileder presenteres antall timer, minutter og sekunder videoene utgir til sammen inni hakeparenteser. De blir fremstilt i formatet t:mm:ss som beskriver timer først, minutter i midten og sekunder til slutt. Tall oppgitt i parenteser sier hvor mange leksjoner, videoleksjoner eller sider i kompendiet som inneholder programmering. For eksempel, kapittel 6 *Parenteser og likninger* inneholder en leksjon uten programmering og en videoleksjon i veilederen på 1 minutt og 44 sekund som omhandler programmering. Av 13 sider i dette kapittelet er det 2 sider som inneholder programmering.

Tabell 15: Antall leksjoner, videoleksjoner og sider for læreverket Matemagisk 8.

Kapittel	Leksjoner	Videoleksjoner i lærerveileder	Antall sider
1 Hele tall	3	4 [0:08:22]	22
2 Brøk og desimaltall	6 (2)	10 [0:22:21] (5 [0:10:20])	29 (7)
3 Algebraiske uttrykk og formler	1	8 [0:18:13] (4 [0:08:39])	15 (4)
4 Potenser, kvadratrøtter og regnerekkefølge	1	0	12 (1)
5 Algebra og likninger	1	0	14
6 Parenteser og likninger	1	1 [0:01:44] (1 [0:01:44])	13 (2)
7 Hva er en funksjon?	0	0	10 (1)
8 Grafen til en funksjon	1	0	11
9 Lineære funksjoner	0	0	12
10 Sammensatte målenheter	2	0	10
Totalt	**16 (2)**	**23 [0:50:40] (10 [0:20:43])**	**148 (15)**

Som det kommer frem i Tabell 15, er det totalt 148 sider med 16 leksjoner i fellesløype og 23 videoleksjoner i lærerveilederen med totalt 50 minutter og 40 sekunder med video. For programmering er det totalt 15 sider med 2 leksjoner og 10 videoleksjoner på 20 minutter og 43 sekunder som danner grunnlaget for hva elevene lærer om programmering utenfor selve oppgavene. Videoleksjoner for programmering i lærerveilederen er kun knytt til kapitelene *Brøk og desimaltall, Algebraiske uttrykk og formler* og *Parenteser og likninger*. Det skapes dermed en forventning om at det eksisterer en større mengde POiM i disse matematiske temaene i forhold til de andre. Denne forventningen støttest også med at det er flest antall sider om programmering i disse temaene og totalt to programmeringsleksjoner i læreverket som er under *brøk og desimaltall*. Det er derfor viktig at disse resultatene tas i betraktning når kvaliteter til hver enkelt programmeringsoppgave i den vertikale analysen skal identifiseres, fordi det gis mer teoretisk grunnlag til programmering i disse temaene.

Campus Matte 8 har plassert sine videoleksjoner i seksjonen for forelesning som er i begynnelsen av hvert delkapittel. I Tabell 16 er antall videoleksjoner sammen med deres varighet og sider i kompendiet oppgitt for hvert kapittel.

Tabell 16: Antall videoleksjoner og sider i kompendiet for læreverket Campus Matte 8.

Kapittel	Videoleksjoner	Sider i kompendiet
1 Tall	48 [1:51:19]	9
2 Regning	48 [1:39:40]	11
3 Brøk	41 [1:27:46]	9
4 Potenser	42 [1:18:44]	9
5 Likninger og algebra	58 [1:03:03]	9
6 Funksjoner	23 [0:54:28]	6
7 Mål og enheter	41 [0:59:26]	10
8.9 Kalkulatorer	6 [0:10:36] (6 [0:10:36])	1 (1)
8.10 Problemløsning	6 [0:13:22] (6 [0:13:22])	1 (1)
Totalt	**313 [9:38:24] (12 [0:23:58])**	**65 (2)**

Tabell 16 viser at det er til sammen 313 videoleksjoner, summert til 9 timer, 38 minutter og 24 sekunder med video i *Campus Matte 8* og 65 sider i deres kompendium som er tilegnet de aktuelle kapitelene. For programmering er det 23 minutter og 58 sekunder med video fordelt utover 12 videoleksjoner og 2 sider i kompendiet. Det vises videoleksjoner i alle delkapitler i dette læreverket, det er derfor mulig å se om det gis kortere eller lengre leksjoner for programmering i forhold til andre temaer. Ved å gjøre om den totale lengden av alle videoleksjoner til sekund og dele på antall videoleksjoner totalt, så får vi gjennomsnittslengden til en videoleksjon på 110,9 sekunder. Samme prosedyre for programmering gir en gjennomsnittslengde for en programmeringsleksjon på 119,9 sekunder. For antall sider i kompendiet er gjennomsnittet, basert på antall sider delt på 60 delkapittel (se Tabell 13), 1,1 side per delkapittel og 1 side per programmeringsdelkapittel (2 sider delt på 2 delkapittel). Programmerings relaterte leksjoner har derfor i snitt 9 sekunder lenger videoleksjoner og 0,1 side mindre i kompendiet i forhold til andre temaer. Læreverket vektlegger derfor delkapittel om programmering i omtrent like stor grad som andre matematiske temaer, selv om det er færre delkapittel for programmering. Resultatene for videoleksjoner og sider i kompendiet for programmering utgir grunnlaget for kunnskapen elevene har om programmering når de skal utføre POiM fra *Campus Matte 8*.

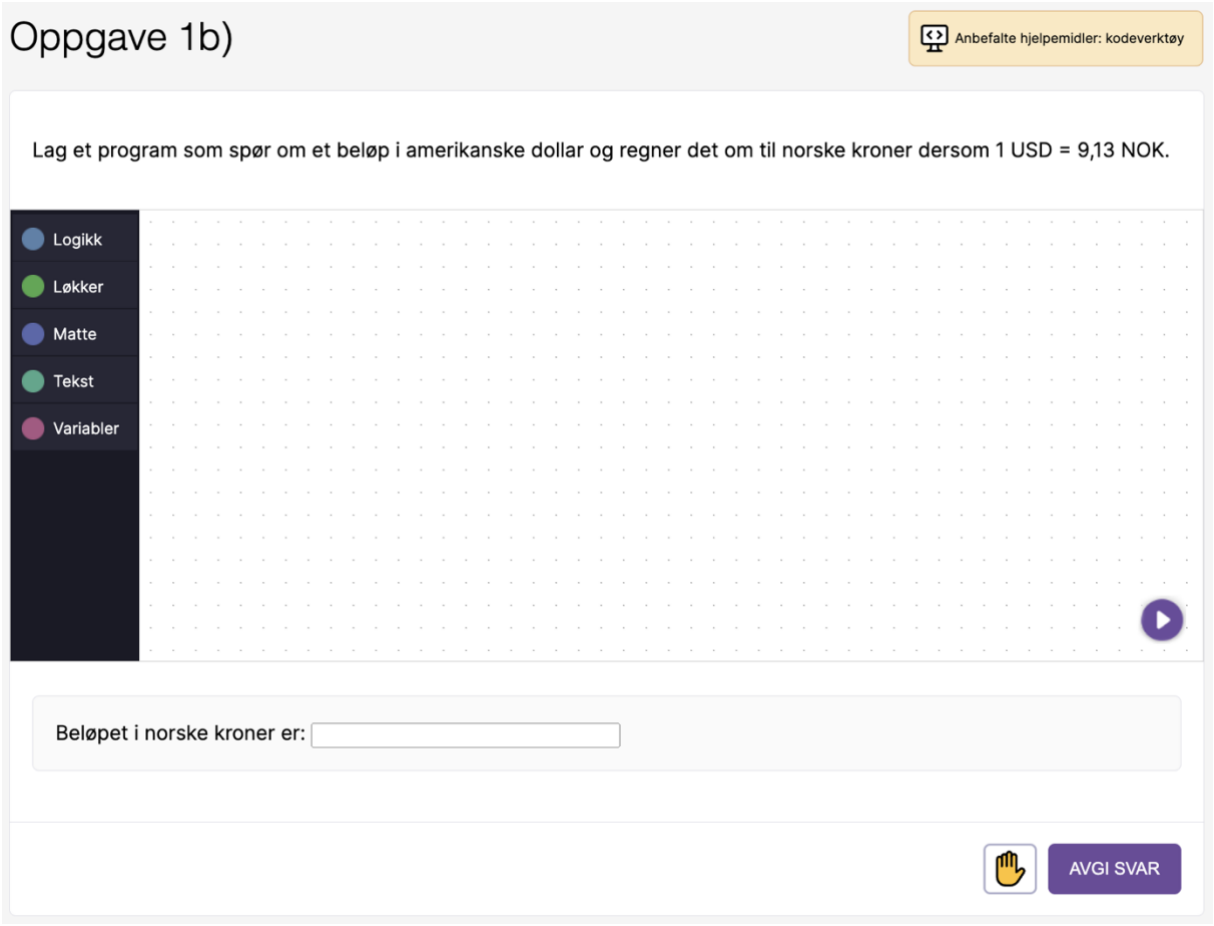
Oppsummert fra den horisontale analysen av strukturen til *Matemagisk 8* og *Campus Matte 8*, er det identifisert åtte matematiske temaer som POiM kan bli plassert under: *brøk og desimaltall*, *algebraiske uttrykk og formler*, *potenser, kvadratrøtter og regnerekkefølge*, *parenteser og likninger*, *funksjoner, tall, regning* og *mål og enheter*. I *Matemagisk 8* er det 15 sider med eksempler, regler, dømer og 2 leksjoner i grunnboken sammen med 20 minutter og 43 sekunder med videoleksjoner fra lærerveilederen knytt til programmering. I *Campus Matte 8* er det 2 sider med eksempler, regler og dømer i kompendiet og 23 minutter og 58 sekunder med videoleksjoner for programmering i læreverket. Disse leksjonene utgir deler av det teoretiske grunnlaget for hva elevene har gjennomgått av programmering før de gjennomfører en POiM. Andre leksjoner knytt til matematikk uten programmering gir teoretisk grunnlag for matematisk kompetanse i de tilhørende læreverkene. Hvilke leksjoner elevene har gjennomgått før oppgaven, er basert på en lineær progresjon som følger rekkefølgen til læreverkene (se kapittel 3.3.1.3). Leksjonene spiller en viktig rolle for hvilke kvaliteter som identifiserer i POiM i den vertikale analysen som besvarer forskningsspørsmål 2 og 3.

4.2 Eksempler fra analysen

Før jeg presenterer resultater fra den vertikale analysen, vil jeg vise noen eksempler. Jeg har valgt ut fire oppgaver fra studiet og viser hvordan analysen brukte logikken fra analyseguiden presentert i kapittel 3.3.4. De tre første eksemplene er av POiM med ulike kvaliteter for kognitive krav. Den siste er en av oppgave som ikke klassifiseres som en POiM. Det første eksempelet forklares i litt mer detalj enn de andre, for å vise hvor deler av datamaterialet er funnet i innholdsanalysen.

4.2.1 Eksempel 1 – memorering

Oppgave 1b) fra *Campus Matte 8* (Thue et al., 2022), vist i Figur 8, brukes til å vise hvordan analyse prosessen foregikk.



Figur 8: Kalkulatorer oppgave 1b fra Campus Matte 8.

Figur 8 viser hvordan elevene møter oppgave 1b) i *Campus Matte 8*. Gjennom funnene fra den horisontale analysen i kapittel 4.1.2, kartlegger jeg at oppgavekategori den befinner seg i er *nivå 1* og ligger i delkapittel *Kalkulatorer* til *Programmering*. Delkapittelet har tre læringsmål for programmering. «Jeg kan lese og forstå programmer som utfører matematiske beregninger. Jeg kan skrive programmer som utfører matematiske beregninger. Jeg kan vurdere om programmer som utfører matematiske beregninger fungerer feilfritt.» (Thue et al., 2022, s. 89). Læringsmålene for programmering i dette delkapittelet baserer seg altså på at elevene skal kunne lese, forstå, skrive og vurdere program som utfører en utregning. Oppgave 1b) knyttes til det andre læringsmålet, å skrive et slikt program, den klassifiseres derfor midlertidig som en programmeringsoppgave. Det teoretiske grunnlaget elevene har fått før oppgaven i dette kapittelet er 10 minutter og 36 sekunder med videoleksjoner og en side i kompendiet med formler, regler og eksempler for programmering. De har tidligere gjennomført tre kontrollspørsmål etter videoleksjonene, to *diskusjon* og en *nivå 1* oppgaver.

Den vertikale analysen av oppgaven krever at elevene skal lage et program som konverterer valutaen amerikanske dollar (USD) til norske kroner (NOK) med en bestemt verdi. Her må altså elevene ved hjelp av programmering lage et program som fungerer som en kalkulator og gjør det som er beskrevet i oppgaveteksten. Matematisk krever oppgaven at elevene kan bruke valuta informasjonen til å utføre en konvertering mellom dem ved hjelp av multiplikasjon. Det å lage et slikt program, kan plasseres under Csizmadia et al. (2015) sin arbeidsmåte å *skape*, fordi oppgaven krever at elevene skaper et program. Oppgaven har tidligere blitt knyttet opp mot et læringsmål for programmering og har blitt plassert under en av *arbeidsmåtene*, derfor klassifiseres den endeling som en POiM i dette studiet og analyseres videre. I den horisontale analysen i kapittel 4.1.2.6, ble de matematiske temaene *brøk og desimaltall*, *algebraiske uttrykk og formler*, *potenser, kvadratrøtter og regnerekkefølge*, *parenteser og likninger, funksjoner, tall, regning* og *mål og enheter* mulige temaer som POiM kunne identifiseres med. Ettersom at oppgave 1b krever at elevene setter sammen et uttrykk som regner om valutaer, i form av USD = NOK * x i programmet og bruker dette som en formel i kalkulatorprogrammet. Så er den satt til å tilhøre *algebraiske uttrykk og formler*. Videre gir oppgaven oss en del informasjon fra ulike representasjoner. Oppgaveteksten oppgir at kursen er fastsatt til at 1 USD er verdt 9,13 NOK. Øverst blir det fortalt at kodeverktøy er det anbefalte hjelpemiddelet og det befinner seg under oppgaveteksten. I kodeverktøyet gis elevene tilgang til listene logikk, løkker, matte, tekst og variabler. Hver av disse listene inneholder et utvalg av kodeblokker som skal plasseres ut i programmeringsvinduet. Dette indikerer at læreverket ønsker at elevene lager programmet ved hjelp av blokkprogrammering. Til slutt gis det et vindu nederst hvor utskrift av hva programmet har regnet ut blir vist, som i dette tilfellet er hvor mange NOK et antall USD er verdt. Denne POiM krever ikke at elevene skal forklare noe eller vise gyldighet til løsningen. Den kan derfor kategoriseres som Charalambous et al. (2010) *kun svar*, fordi det ikke kreves mer enn å svare med programmet. Videre undersøkes hvilke løsninger som er mulige. Ifølge informasjonen oppgaven gav oss, var det anbefalt å bruke kodeverktøyet som hjelpemiddelet. Slik som *Campus Matte 8* er strukturert, vil det gi elevene en indikasjon om svaret til en oppgave er korrekt eller ikke. Dette er bare mulig, dersom løsningen blir gitt via bruk av kodeverktøyet til læreverket. På grunn av at hjelpemiddelet er bare anbefalt, betyr det at oppgaven kan løses på andre måter. Lærer og elev kan derfor gjøre en avtale på at

løsningen kan bli gitt i en annen form enn hva læreverket kan kontrollere. I dette studiet tar jeg utgangspunkt i å følge læreverkets strukturering og for at svaret skal bli kontrollert av læreverket, må løsningen av oppgaven løses ved hjelp kodeverktøyet de tilbyr.

Det neste steget i den vertikale analysen av oppgaven er å kartlegge om leksjonene og oppgavene de tidligere har utført gir støtte til løsningen av oppgave 1b). Ved å se igjennom oppgavens tidligere teoretiske grunnlag funnet i den horisontale analysen, er det to elementer som gir støtte til løsningen av oppgaven. De to elementene er oppgave 1a) og videoleksjonene funnet i delkapittel *Kalkulatorer* i *Programmering* fra *Campus Matte 8* (Thue et al., 2022). Under vises oppgave 1a) og et skjermbilde av videoleksjonen tatt på slutten.

Figur 9: Oppgave 1a) fra delkapittel Kalkulator i Campus Matte 8.

Figur 9 viser oppgave 1a) fra *Campus Matte 8*, som er en lignende oppgave gitt med en annen fremstilling. I denne oppgaven får elevene se et ferdig laget program som utfører en konvertering av valuta, slik som oppgave 1b) ønsker elevene skal lage. Forskjellen mellom disse to oppgavene, er at i oppgave 1a) skal elevene regne ut hva programmet ville vist som utskrift dersom 200 norske kroner skulle konverteres til danske kroner med kursen 1 norsk er lik 1,35 danske.

Figur 10: Skjermbilde av videoleksjon fra delkapittel Kalkulator i Campus Matte 8.

Figur 10 viser videoleksjonen som gir steg for steg, med forklaring via lyd, hvordan man lager et program som regner om euro til norske kroner med den bestemte kursen 1 euro er lik 9,87 norske (Thue et al., 2022). Sammen gir videoleksjonen og oppgave 1a) en tydelig løsning av hvordan programmet til oppgave 1b skal skrives, med unntak av at navn og verdier på variabler må endres for å besvare oppgaven.

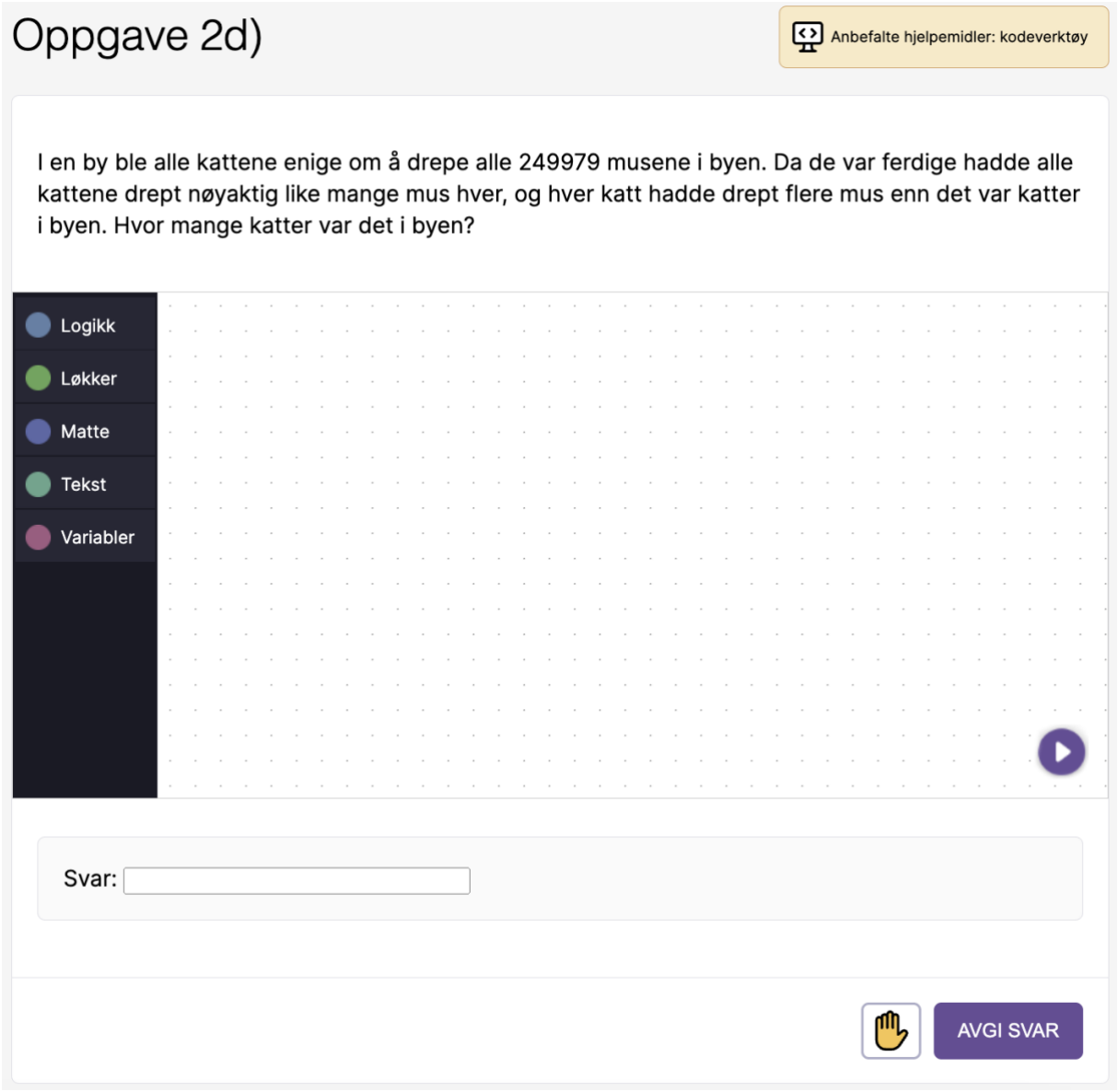
De to siste stegene av analysen er å bruke alle funnene til å indentifisere ulike *nøkkelbegreper* av Csizmadia et al. (2015) og til slutt definere oppgavens *kognitive krav* etter Smith og Stein (1998) sin task-analysis guide. Det første nøkkelbegrepet jeg identifiserer er bruken av *logikk*. I denne POiM må elevene identifisere relevant informasjon gjennom analysering av oppgaveteksten og se for seg hvordan dette kan brukes. I dette tilfelle, se for seg hvordan informasjonen om valuta verdien kan brukes i programmet for å skape løsningen. Den andre kvaliteten som jeg identifiserer, er *algoritmer*. For at oppgaven skal løses, må elevene sette sammen hvilke steg og i hvilken rekkefølge stegene skal utføres. Her må altså elevene først be brukeren oppgi antall valuta som skal omgjøres og la programmet lagre dette i en variabel. Deretter skal programmet utføre omregningen av valutaene. Til slutt skal programmet skrive ut summen av den nye valutaen. Den siste kvaliteten av nøkkelbegrep som jeg identifiserte, er *mønster*. Ved at elevene oppdager likheter med oppgaven i forhold til oppgave 1a og videoleksjonene, vil de se at store deler av løsningen er allerede gitt. De resterende kvalitetene *dekomposisjon*, *abstraksjon* og *evaluering*

identifiseres ikke i denne POiM. Den krever ikke at elevene trenger å separere deler av problemet, skille bort unødvendig informasjon eller vurdere om endringen i programmet vil gjøre løsningen bedre for å løse oppgaven. Denne POiM sitt *kognitive krav* blir i dette tilfellet satt til *memorering*. Dette begrunnes med at oppgavens programmeringskrav kan enkelt løses ved å repetere innholdet fra videoleksjoner og forrige oppgave. Her er alle nødvendige blokker allerede satt opp korrekt, og eleven trenger bare å kopiere løsningen. Matematisk trenger elevene bare å se hva verdien til valutaen er og endre dette i programmet som er kopiert.

Oppgave 1b i delkapittel *Kalkulatorer* i *Programmering* fra *Campus Matte 8* (Thue et al., 2022), har blitt identifisert som en POiM gitt i oppgavekategorien *nivå 1* med arbeidsmåten å *skape* og type svar *kun svar*. Den har blitt klassifisert med nøkkelbegrepene *logikk*, *algoritmer* og *mønster* og det kognitive kravet *memorering*.

4.2.2 Eksempel 2 – prosedyrer uten sammenheng

I dette eksempelet brukes oppgave 2d) fra delkapittelet *Problemløsning* i kapittelet *Programmering* fra *Campus Matte 8* (Thue et al., 2022) (se Figur 11).

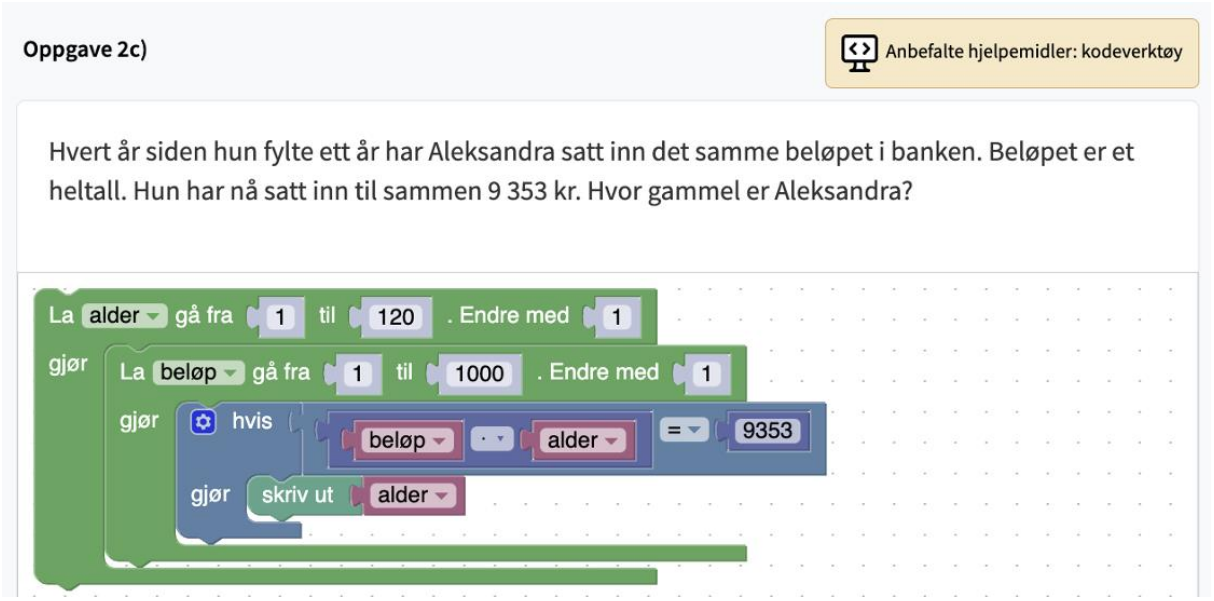


Figur 11: Problemløsning oppgave 2d) fra Campus Matte 8.

Figur 11 viser oppgave 2d) som ligger i oppgavekategorien *nivå 1* i kapittelet *Programmering* og delkapittel *Problemløsning* fra *Campus Matte 8* (Thue et al., 2022). Dette delkapittelet har 3 læringsmål for programmering. «Jeg kan lese og forstå et program som benytter «gjett-og-sjekk»-metoden. Jeg kan lage dataprogrammer som løser problemer ved hjelp av «gjett-og-sjekk» metoden. Jeg kan forbedre mine egne algoritmer slik at programmene mine finner løsningen raskere.» (Thue et al., 2022, s. 90). Læringsmålene handler om å lese, forstå og lage program med problemløsningsmetoden «gjett-og-sjekk» og forbedre egne algoritmer. Denne oppgaven knyttes opp mot det andre læringsmålet, å lage et program som bruker «gjett-og-sjekk»-metoden og klassifiseres dermed midlertidig som en

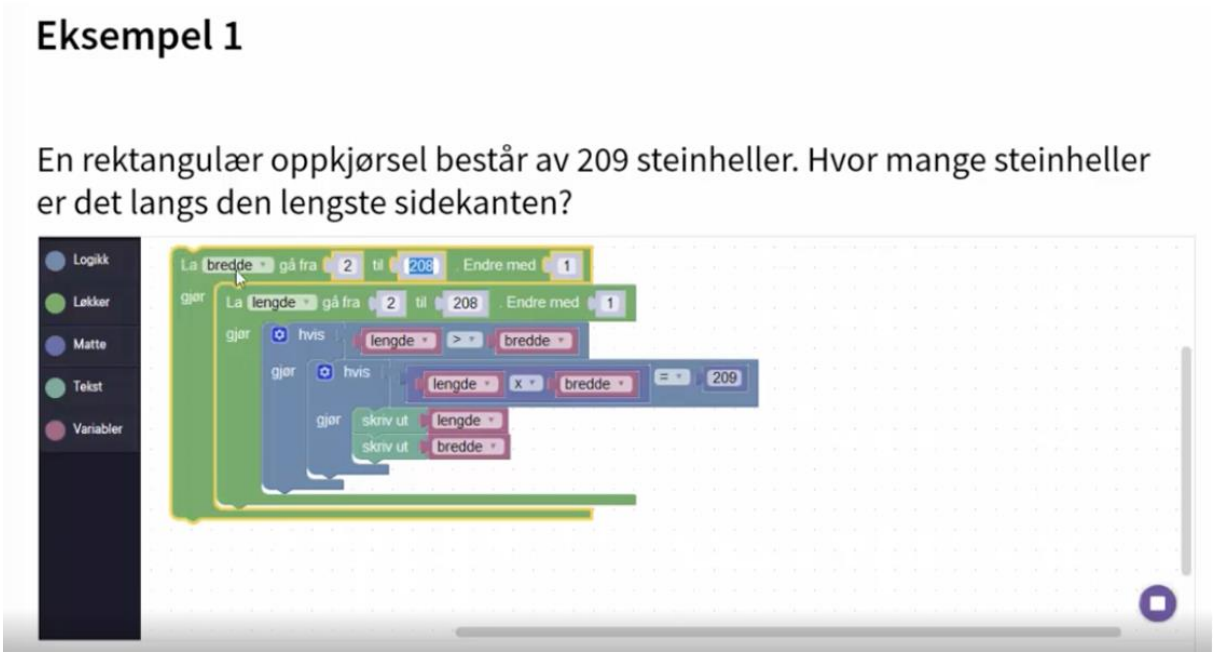
programmeringsoppgave. Basert på forrige og nåværende delkapittel har elevene teoretisk grunnlag fra videoleksjoner på 23 minutt og 58 sekund sammen med 2 sider i kompendiet. I tillegg til 44 oppgaver, hvor 37 er klassifisert som POiM.

Oppgaveteksten forklarer en situasjon hvor katter skal til sammen drepe et stort antall mus. Kattene drepte like mange mus hver og det er flere mus enn antall katter. Oppgaven spør deretter om å finne ut antallet katter. For at elevene skal løse oppgaven, må de programmere et program som bruker «gjett-og-sjekk»-metoden. Matematisk må de konstruere en likning som brukes i programmet som sjekker om tallene det gjetter er korrekt. Ettersom at oppgaven krever at elevene lager et program, kan det kategoriseres under Csizmadia et al. (2015) arbeidsmåte å *skape* og den defineres endelig som en POiM. Siden det kreves å lage en likning som programmet skal kontrollere om er korrekt, plasseres oppgaven under det matematiske temaet *parenteser og likninger* funnet i den horisontale analysen. Oppgaveteksten definerer en del krav som elevene må bruke i løsningen. Antall mus er satt til 249979, kattene skal drepe like mange mus hver og det er flere mus en katter. For å besvare oppgaven, kreves det ingen *forklaring* eller å *vise gyldighet*. Den klassifiseres derfor som Charalambous et al. (2010) *kun svar*. I likhet med oppgave 1b) i det første eksempelet, så ønsker læreverket å kontrollere om løsningen er korrekt og gi tilbakemelding på dette. Samtidig fastslår læringsmålene at løsningen skal løses ved hjelp av «gjett-og-sjekk»-metoden. Det er derfor begrenset hvor mange løsninger som er mulige i denne oppgaven, da læreverket er ute etter en løsning.



Figur 12: Oppgave 2c) gitt i delkapittel problemløsning fra Campus Matte 8.

Figur 12 viser oppgave 2c) som er gitt før oppgaven i fokus (Thue et al., 2022). Den har en oppgavetekst som er svært ulik oppgave 2d), men framgangsmåten til løsninger er nokså lik. Løsningen baserer seg på å lage et program som bruker «gjett-og-sjekk»-metoden for å gjette på tall og sjekke om tallene får en likning til å gå opp.



Figur 13: Skjermbilde av videoleksjonen «eksempel 1» gitt i delkapittel problemløsning fra Campus Matte 8.

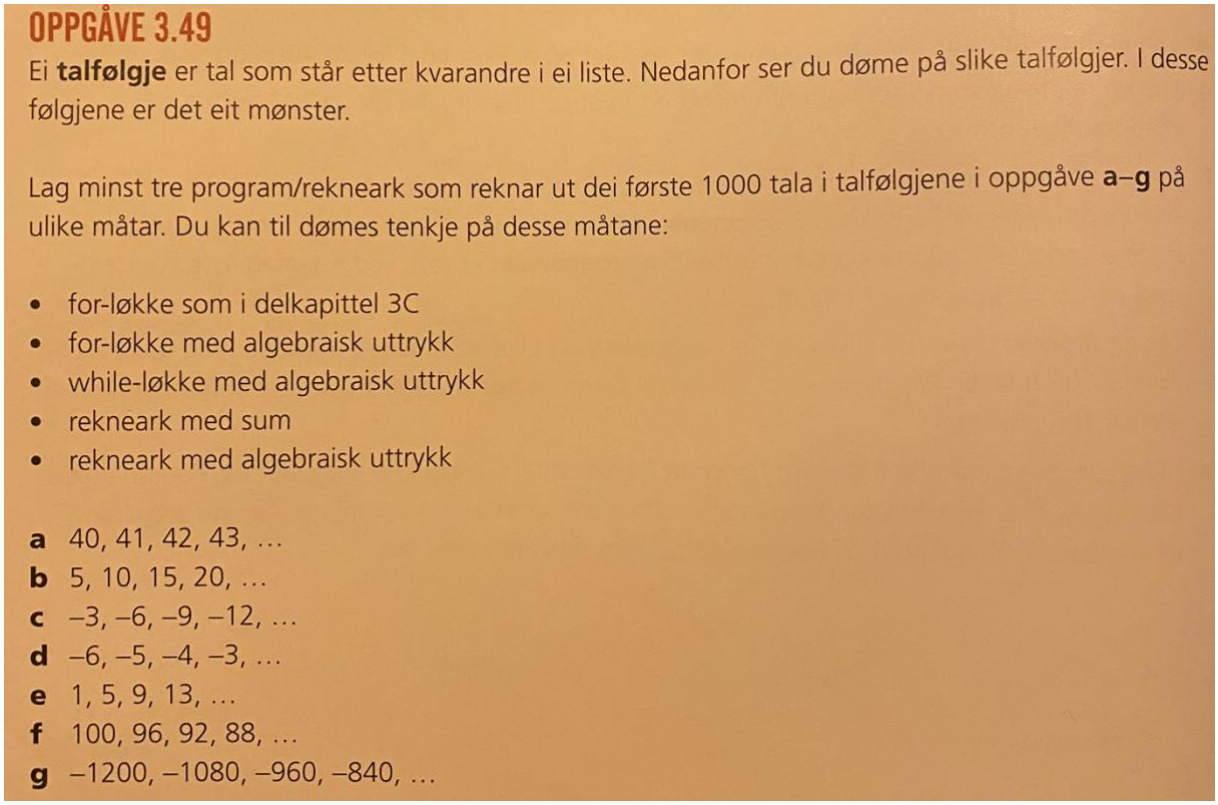
Figur 13 viser skjermbilde av en videoleksjon som går igjennom steg for steg hvordan man kan bruke denne «gjett-og-sjekk»-metoden til å løse ulike problem (Thue et al., 2022). Her får elevene forklart hvordan man setter sammen et program som bruker denne metoden og hvilken funksjon de forskjellige verdiene i koden har. Både videoleksjonen og løsningen av oppgave 2c) gir støtte til hvordan elevene kan løse oppgave 2d).

Den første nøkkelbegrep kvaliteten av Csizmadia et al. (2015) som blir identifisert i denne POiM er *logikk*. Elevene må identifisere relevant informasjon og forutse hvordan den kan brukes til å skape en løsning. *Algoritmer* er identifisert, basert på at oppgavens ulike krav må tas i betraktning og det må tydelig komme fram hvordan hvert steg bidrar til løsningen. For eksempel, må elevene tydeliggjøre at et antall katter ikke skal gjettes på dersom antall mus er mindre eller lik katter. Dersom dette ikke gjøres, vil løsningen gi svar som ikke følger kravene til oppgaven. *Dekomposisjon* er identifisert, dersom elevene deler opp problemet i to deler. Ved å først skape en løsning som regner ut antall katter og antall mus drept per katt det trengs får å drepe 249979 mus totalt, noen som er kjent fra deres teoretiske grunnlag. Så kan de deretter ta inn kravet om at det er flere mus enn katter ved å gjøre små endringer i programmet for å finne løsningen. Kvaliteten *mønster* er gjenkjent ved å bruke likheter fra innholdet i fra videoleksjonen og oppgave 2c) til å konstruere programmet og finne løsningen. Den siste identifiserte kvaliteten av nøkkelbegrep i denne POiM er *abstraksjon*. Oppgaveteksten er konstruert på en måte som krever at elevene må analyserer informasjonen for hva som er relevant fordi den fremføres som en kort fortelling. Ulike krav må skilles ut fra teksten for å lettere vise viktig informasjon, slik som at det er flere mus enn katter. Annen informasjon, som at kattene befinner seg i en by, er ikke relevant for løsningen. Nøkkelbegrepet *evaluering* ble ikke identifisert fordi oppgaveteksten ikke oppfordrer elevene til å se eller tenke over løsningen for å forbedre den. POiM kategoriseres under *prosedyrer uten sammenhenger* av Smith og Stein (1998). Selv om at mye av løsningen kan hentes via elevenes teoretiske grunnlag, blir den ikke klassifisert som *memorering*. Oppgaven kan ikke løses direkte ved å kopiere fra deres teoretiske grunnlag og endre på noen få tall verdier. Fokuset er mer vendt mot hvordan løsningsmetoden kan brukes i ulike tilfeller.

Oppgave 2d) fra delkapittelet *Problemløsning* i kapittelet *Programmering* fra *Campus Matte 8* (Thue et al., 2022) er identifisert som en *nivå 1* POiM med arbeidsmåten å *skape* og krever svaret *kun svar*. Kvalitetene som er klassifisert i oppgaven er nøkkelbegrepene *logikk*, *algoritmer*, *dekomposisjon*, *mønster* og *abstraksjon* og det kognitive kravet PUS.

4.2.3 Eksempel 3 – prosedyrer med sammenheng

Oppgave 3.49) i delkapittelet *Figurtall* i kapittelet *Algebraiske uttrykk og formler* fra *Matemagisk 8* (Kongsnes & Wallace, 2020, s. 124) brukes i dette eksempelet og vises i Figur 14.



OPPGÅVE 3.49

Ei **talfølgje** er tal som står etter kvarandre i ei liste. Nedanfor ser du døme på slike talfølgjer. I desse følgjene er det eit mønster.

Lag minst tre program/rekneark som reknar ut dei første 1000 tala i talfølgjene i oppgåve **a**–**g** på ulike måtar. Du kan til dømes tenkje på desse måtane:

- for-løkke som i delkapittel 3C
- for-løkke med algebraisk uttrykk
- while-løkke med algebraisk uttrykk
- rekneark med sum
- rekneark med algebraisk uttrykk

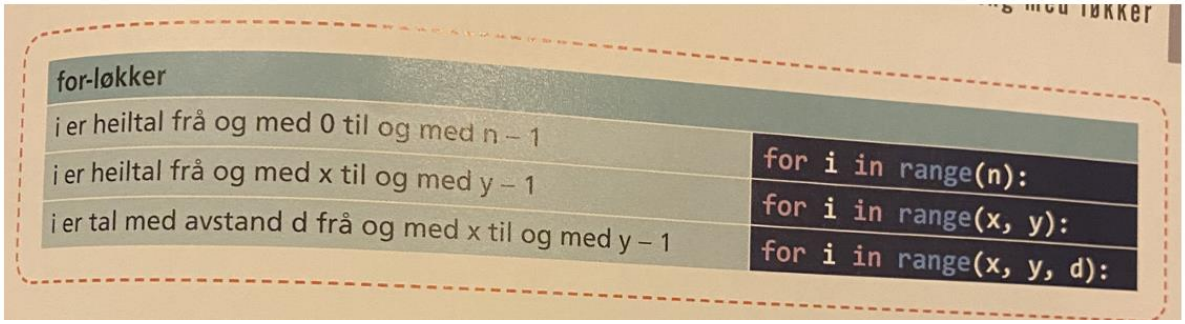
a 40, 41, 42, 43, …
b 5, 10, 15, 20, …
c –3, –6, –9, –12, …
d –6, –5, –4, –3, …
e 1, 5, 9, 13, …
f 100, 96, 92, 88, …
g –1200, –1080, –960, –840, …

Figur 14: Figurtall oppgave 3.49) fra Matemagisk 8.

Oppgave 3.49) er en av POiM i dette studiet som er blitt markert med en apostrofe i datamaterialet, diskutert i tidligere i kapittel 3.4. Dette er fordi oppgaven ber elevene selv velge hvor mange deloppgaver som skal gjøres og at deler av oppgaven kan fullføres ved å gjøre små endringer i programmet. Oppgaven er plassert under oppgavekategorien *nivå 2*. Den befinner seg i kapittelet *Algebraiske uttrykk og formler* og delkapittel *Figurtall* i

Matemagisk 8. Delkapittelet har et læringsmål for programmering. «Bruke programmering og regneark til å beskrive og skrive ut tallmønster.» (Kongsnes & Wallace, 2020, s. 116). Det omhandler å finne og skrive ut tallmønster ved hjelp av programmering, oppgaven knyttes opp mot dette læringsmålet. Elevenes teoretiske grunnlag baseres på 2 leksjoner om programmering, 18 minutter og 59 sekunder med videoleksjon fra lærerveilederen og 11 sider med programmerings relatert innhold med oppgaver, eksempler og regler.

Oppgaven ber elevene lage minst tre program som regner ut de første 1000 tallene til ulike tallrekker. Det stilles ikke krav til hvilke metoder som må brukes, men hinter til noen. For programmering må altså elevene lage ulike program som bruker for-løkker, while-løkker eller andre løsninger. Matematisk må elevene resonnere seg fram til ulike figurtall fra listen med tallrekker. På grunn av at oppgaven krever at elevene lager program, identifiserer den som å *skape* fra Csizmadia et al. (2015) *arbeidsmåter*. Den defineres derfor som en POiM. Figurtallene må skrives som et algebraisk uttrykk eller en formel i programmet og kan derfor plasseres under det matematiske temaet *algebraiske uttrykk og formler*. Informasjon som at elevene skal lage minst tre program og at de velger selv hvordan de løser oppgaven, indikerer at oppgaven kan gjøres på flere måter. Annen informasjon er at de fire første tallene i alle tallrekkene er gitt i oppgaveteksten. Det blir ikke oppgitt at det krever en *forklaring* eller å *vise gyldighet* av svaret. Den kategoriseres derfor som Charalambous et al. (2010) *kun svar*. Som nevnt i sted, forteller oppgaven oss at det er opp til elevene selv hvilken løsning de ønsker å gi. Det er derfor mange mulige løsninger.



Figur 15: Forklaring av for-løkker gitt i Matemagisk 8.

Figur 15 viser en forklaring av ulike for-løkker i *Matemagisk 8* av Kongsnes og Wallace (2020, s. 113). Her blir forklaringen og koden gitt i form av en tabell. Dette gir elevene teoretisk grunnlag for hvordan de kan formatere for-løkker på ulikemåter ved hjelp av tekstprogrammering.

```
1  tall = 0
2  while tall < 50:
3    print(tall)
4    tall = tall + 1
```

Følgende skrives til skjermen når programmet kjøres:

0	5	10	15	20	25	30	35	40	45
1	6	11	16	21	26	31	36	41	46
2	7	12	17	22	27	32	37	42	47
3	8	13	18	23	28	33	38	43	48
4	9	14	19	24	29	34	39	44	49

Figur 16: Videoleksjon om while-løkker fra lærerveilederen til Matemagisk 8.

Figur 16 viser et skjermbilde av slutten til en videoleksjon gitt delkapittel *Programmering med løkker* i lærerveilederen til Matemagisk 8 (Aunivers, 2021). Leksjonen forklarer funksjonen til hver av de fire linjene i programmet som utfører en while-løkker for 50 tall. Resultatet fra programmet skrives ut og vises nederst i figuren. Sammen med Figur 15, utgir forklaringen og videoleksjonen teoretisk grunnlag for hvordan for- og while-løkker fungerer og skrives.

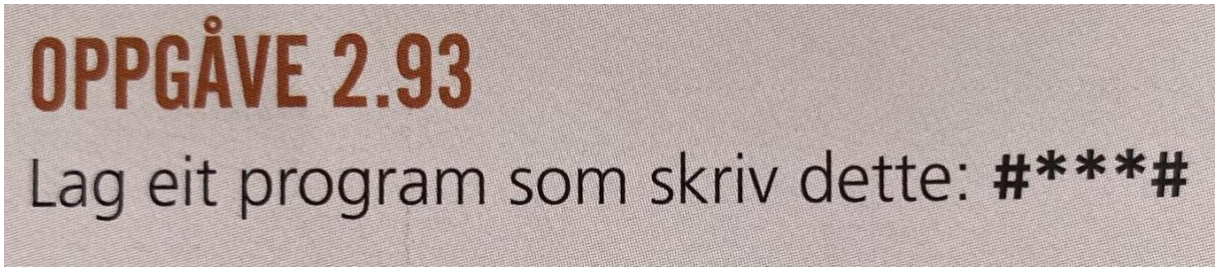
Nøkkelbegrep kvaliteter fra den algoritmiske tenkeren av Csizmadia et al. (2015) som identifiseres i denne POiM er først *logikk*. Ved å analysere tallrekkene for å finne figurtallet og forutse hvordan dette kan brukes til å skape en løsning brukes denne kvaliteten. *Algoritmer* er en kvalitet som identifiseres. Elevene må bruke den informasjonen de har samlet inn og deres teoretiske grunnlag til å lage en steg-for-steg-plan for å lage

programmet. Stegene vil være ulike for hver av løsningene de skaper. Den tredje kvaliteten er *mønster*. Her må elevene finne figurtallet til tallrekken og se etter likheter mellom den og metoden de bruker. Til slutt identifiseres kvaliteten *abstraksjon*. Elevene må skille ut den nødvendige informasjon for hver av tallrekken for å identifisere differansen mellom tallene. Alle tallrekkene endres lineært og kan derfor enkelt løses ved å se differansen mellom de to første tallene i tallrekkene. Det ble ikke identifisert *dekomposisjon* og *evaluering*. Denne POiM kategoriseres som en av det kognitive kravet *prosedyrer med sammenhenger* av Smith og Stein (1998). Begrunnelsen for dette valget er basert på at elevene må se matematiske sammenhenger mellom figurtall og hvordan figurtall kan brukes i programmering. Hver tallrekke løses på minst 3 ulike tilnærminger. Dersom elev velger å lage programmet ved hjelp av for- og while-løkker, kan det skapes en dypere forståelse av sammenhengen mellom den matematiske løsningen og programmet.

Oppgave 3.49) identifiseres derfor som en *nivå 2* oppgave, som krever at elevene *skaper* et program og svarer med *kun svar*. Nøkkelbegreper som *logikk*, *algoritmer*, *mønster* og *abstraksjon* sammen med det kognitive kravet *prosedyrer med sammenhenger* er klassifiserte kvaliteter ved oppgaven.

4.2.4 Eksempel 4 – ikke POiM

For å vise et tilfelle hvor en oppgave som i utgangspunktet er en programmeringsoppgave, men ikke en POiM, brukes oppgave 2.93) fra Matemagisk 8 (Kongsnes & Wallace, 2020, s. 92) vist i Figur 17.



Figur 17: Programmering i Python oppgave 2.93) fra Matemagisk 8.

Oppgaven 2.93) er plassert i oppgavekategorien nivå 1, kapittel *Brøk og desimaltall* og delkapittel *Programmering i Python* (Kongsnes & Wallace, 2020, s. 92). Delkapittelet inneholder to læringsmål for programmering: «Utforske og lage en algoritme for et program

som utfører en regneoperasjon. Bruke Python og lage et kalkulatorprogram som henter verdier fra brukeren, utfører en regneoperasjon og skriv svaret til skjermen.» (Kongsnes & Wallace, 2020, s. 86). Læringsmålene for programmering omhandler altså å lage og utforske algoritmer og skrive program som utfører regneoperasjoner. Denne oppgaven krever at elevene lager et program som skal skrive ut bestemte symboler. Det kreves ingen regneoperasjoner for å løse oppgaven, fordi elevene trenger kun å skrive en linje med kode som skriver ut symbolene. Dette blir eksemplifisert som et døme, hvor oppgaven løses ved å skrive inn koden print() med symbolene inni parentesene og anførselstegn på side 86 i *Matemagisk 8* (Kongsnes & Wallace, 2020). På grunn av at løsningen ikke krever en regneoperasjon, knyttes den ikke til læringsmålene for programmering. Den klassifiserer derfor ikke som en POiM i dette studiet selv om den løses ved hjelp av programmering. Kriteriene satt i dette studiet krever at en POiM skal knyttes opp et læringsmål for programmering og at den bruker en av arbeidsmåtene fra Csizmadia et al. (2015) (se kapittel 3.3.1.1). Analysen avsluttes derfor for denne oppgaven og den ekskluderes fra datamaterialet.

4.3 Resultater fra vertikal analyse

Etter at den horisontale analysen kartla hvilke *matematiske temaer* i læreverkene som inneholdt læringsmål i programmering, er oppgavene som undersøkes i dette studiet snevret inn. Dersom en programmeringsoppgave kan knyttes til et læringsmål for programmering, er det første kriteriet godkjent. Det andre kriteriet ble gjort igjennom den vertikale analysen, hvor oppgaven undersøkes om den kan plasseres inn i en av kategoriene til Csizmadia et al. (2015) *arbeidsmåter* i analysens rammeverk. Dersom en av oppgavene fra første kriteriet kunne plasseres under en av arbeidsmåtene, ble oppgaven definert som POiM og ble analysert for resten av rammeverket fra den vertikale analysen. Dette resulterte i at *Matemagisk 8* har 82 og *Campus Matte 8* har 49 POiM. Funnene blir utdypet i Tabell 17 som først viser antall POiM under *matematiske temaer* og deretter antall POiM etter *arbeidsmåter* og *type svar* for begge læreverkene. Her ser vi, for eksempel, at *brøk og desimaltall*, består av POiM fra *Matemagisk 8* (16) og *Campus Matte 8* (5). Disse bruker arbeidsmåtene å *skape* (15), *fikle* (2), *feilsøke* (3) og *samarbeide* (1) og de krever *kun svar* (14), *forklaring* (6) og *vise gyldighet* (1). Dette tilsvarer 21 POiM av 131.

Tabell 17: Antall programmeringsoppgaver under matematiske temaer og arbeidsmåter for begge læreverkene.

Læreverk	Brøk og desimaltall	Algebraiske uttrykk og formler	Potenser, kvadratrøtter og regnerekkefølge	Parenteser og likninger	Funksjoner	Totalt:
Matemagisk 8	16	43	4	9	10	**82**
Campus Matte 8	5	23	-	21	-	**49**
Arbeidsmåter						
Skape	15	44	3	23	8	**93**
Fikle	2	15	1	5	2	**25**
Feilsøke	3	-	-	-	-	**3**
Samarbeide	1	7	-	2	-	**10**
Type svar						
Kun svar	14	54	3	28	8	**107**
Forklaring	6	12	1	2	2	**23**
Vise gyldighet	1	-	-	-	-	**1**
Antall oppgaver:	**21**	**66**	**4**	**30**	**10**	**131**

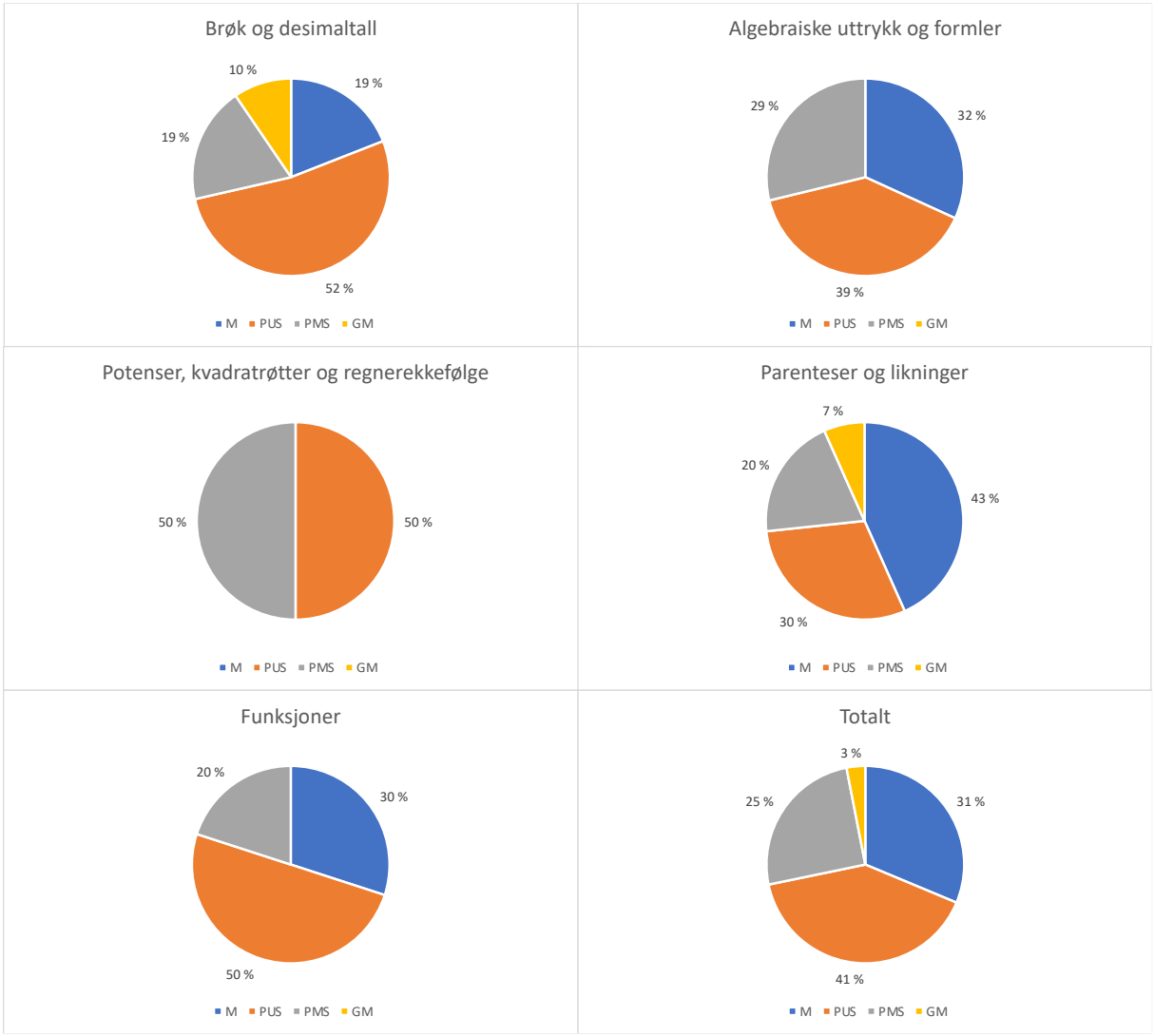
Resultatene viser at alle POiM fra *Campus Matte 8* kunne plasseres i de eksisterende temaene *brøk og desimaltall*, *algebraiske uttrykk og formler* og *parenteser og likninger*. POiM fra *Matemagisk 8* kunne også plasseres under disse temaene i tillegg til *potenser, kvadratrøtter og regnerekkefølge* og *funksjoner*. Resultatene under *arbeidsmåter* og *type svar* brukes til videre tolkninger i de neste underkapitlene hvor det er relevant.

4.3.1 Kognitive krav

Gjennom den vertikale analysen ble hver av de totalt 131 POiM analysert og undersøkt for kognitive krav etter Smith og Stein (1998) task-analysis guide. Hvilke kognitivt krav POiM ble identifisert med er basert på innholdsanalyse av selve oppgaven etter rammeverket fra den vertikale analysen. Resultatene er basert på innholdet i leksjoner, videoleksjoner, dømer, eksempler og regler funnet i den horisontale analysen, i kapittel 4.1.2.6, og løsningen av tidligere gitte oppgaver. Resultatene for *kognitive krav* identifisert ved POiM er først presentert i Tabell 18, deretter vises fordelingen av resultatene per *matematiske tema* i sektordiagram. Diagrammer viser prosent som er rundet av til nærmeste heltall for å vise resultatene mer ryddig og tydelig. Dette medfører at prosentene ikke viser helt nøyaktige resultat, men de representerer resultatene mer en godt nok til å skape en oversiktlig fordeling.

Tabell 18: Programmeringsoppgavenes kognitive krav fordelt ut over matematiske temaer.

	Brøk og desimaltall	Algebraiske uttrykk og formler	Potenser, kvadratrøtter og regnerekkefølge	Parenteser og likninger	Funksjoner	Totalt:
M	4	21	-	13	3	**41**
PUS	11	26	2	9	5	**53**
PMS	4	19	2	6	2	**33**
GM	2	-	-	2	-	**4**
Totalt:	**21**	**66**	**4**	**30**	**10**	**131**



Figur 18: Fordelingen av programmeringsoppgavers kognitive krav for alle matematiske temaer.

Tabell 18 viser antall POiM av ulike *kognitive krav* fordelt utover de matematiske temaene funnet i den horisontale analysen. Resultatene kan leses på to måter. Først kan man velge et *matematisk tema* fra den øverste raden og lese nedover i kolonnen for å se hvilke kognitive krav POiM har i valgte tema. For eksempel temaet *brøk og desimaltall* inneholder 4 M, 11 PUS, 4 PMS og 2 GM som til sammen er 21 programmeringsoppgaver. Den andre måten å lese resultatene på, er å velge et *kognitivt krav* fra kolonnen til venstre og lese av antall POiM for hvert *matematiske tema* langs raden. Det *kognitive kravet* M inneholder 4 POiM i *temaet brøk og desimaltall*, 21 i *algebraiske uttrykk og formler*, ingen i *potenser, kvadratrøtter og regnerekkefølge*, 13 i *parenteser og likninger* og 3 i *funksjoner*. Sammenlagt er det 41 M-POiM.

Resultatene viser at det er store sprik mellom fordelingen av POiM i de matematiske temaene. *Algebraiske uttrykk og formler* inneholder omtrent halvparten (66 av 131) av POiM i studiet. Temaene *funksjoner* og *potenser, kvadratrøtter og regnerekkefølge* inneholder færrest (10 og 4).

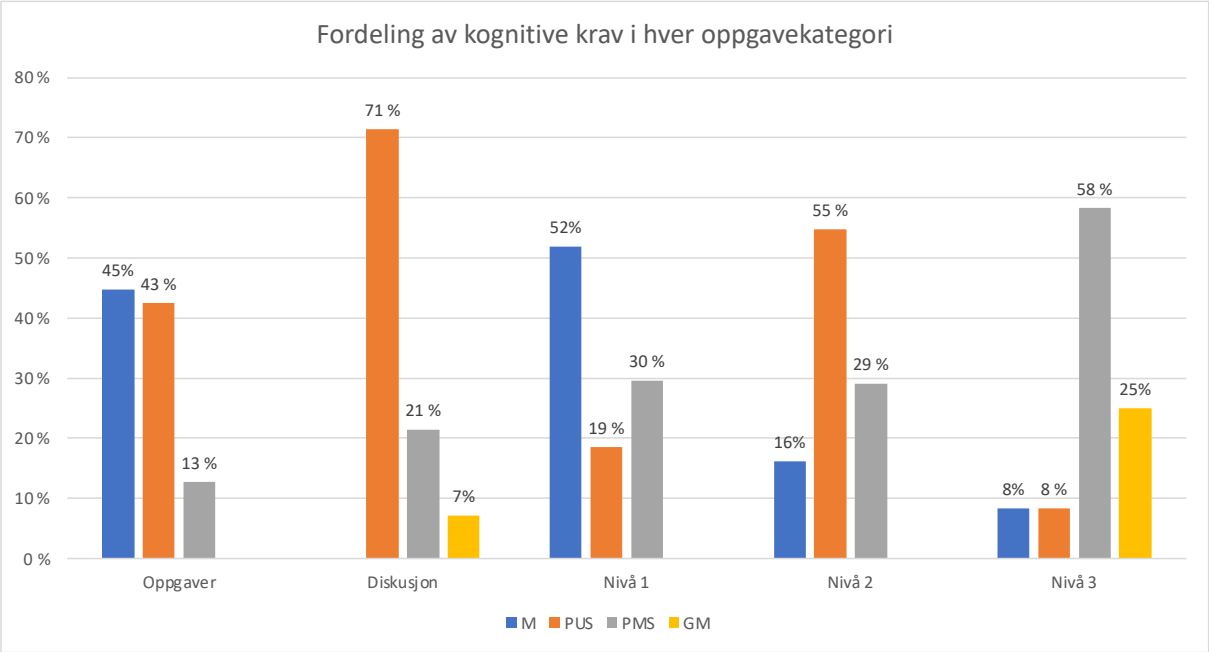
Første fem diagram på Figur 18 viser fordelingen av POiM sine *kognitive krav* for en av de fem matematiske temaene, mens den siste, Totalen, viser fordelingen uavhengig av tema. Alle diagram har samme fargekode: blå representerer det kognitive kravet M, oransje PUS, grå PMS og gul GM. For eksempel, sektordiagrammet til *parenteser og likninger* inneholder 43 % M-, 30 % PUS-, 20 % PMS- og 7 % GM-oppgaver. Så lenge temaene *funksjoner og potenser*, *kvadratrøtter og regnerekkefølge* har få POiM, hvor en POiM utgir 10 % eller mer av resultatene, er deres sektordiagram lite informative. Ved å se på de tre andre temaene, har fordelingen av lavt og høyt kognitivt krevende POiM en differanse på 2 %. Resultatene viser også at *algebraiske uttrykk og formler*, med flest POiM, ikke inneholder noen GM-oppgaver.

Funnene av *kognitive krav* kan også settes opp mot hvilken *oppgavekategori* de tilhører. Oversikten presenteres først i Tabell 19 som viser hver *oppgavekategori* øverst og antall POiM med de ulike *kognitive kravene* for kategorien under. Deretter presenteres resultatene fra tabellen i Figur 19. Fordelingen vises i prosent som regnes ut ved å dele antallet for de

ulike kognitive kravene i en *oppgavekategori* på antall programmeringsoppgaver i *oppgavekategorien*. Figuren vil gjøre det lettere å se om det eksisterer trender i fordelingen.

Tabell 19: Programmeringsoppgavenes kognitive krav fordelt ut over oppgavekategorier.

	Oppgaver	Diskusjon	Nivå 1	Nivå 2	Nivå 3	Nivå 4	Totalt:
M	21	-	14	5	1	-	**41**
PUS	20	10	5	17	1	-	**53**
PMS	6	3	8	9	7	-	**33**
GM	-	1	-	-	3	-	**4**
Totalt:	**47**	**14**	**27**	**31**	**12**	**0**	**131**



Figur 19: Fordeling av kognitive krav i hver oppgavekategori.

Tabell 19 leses nedover i kolonnene og viser blant annet at *diskusjon* inneholder ingen M-, 10 PUS-, 3 PMS- og 1 GM-oppgaver som er til sammen 14 POiM. Et viktig funn her er at det ikke eksisterer POiM i *nivå 4*, den vises derfor ikke i figuren. Andre funn tolkes sammen med avsnittet under.

Resultatene for oppgavekategorien *nivå 3*, leses av som 8 % M-, 8 %, PUS-, 58 % PMS- og 25 % GM-oppgaver. Det er mulig å se etter trender i de differensierte oppgavekategoriene *nivå 1, 2* og *3*. Dette er dessverre vanskelig for kategoriene *oppgaver* og *diskusjon*, da deres innhold er mer varierende enn de andre. De vil derfor tolkes uten trender.

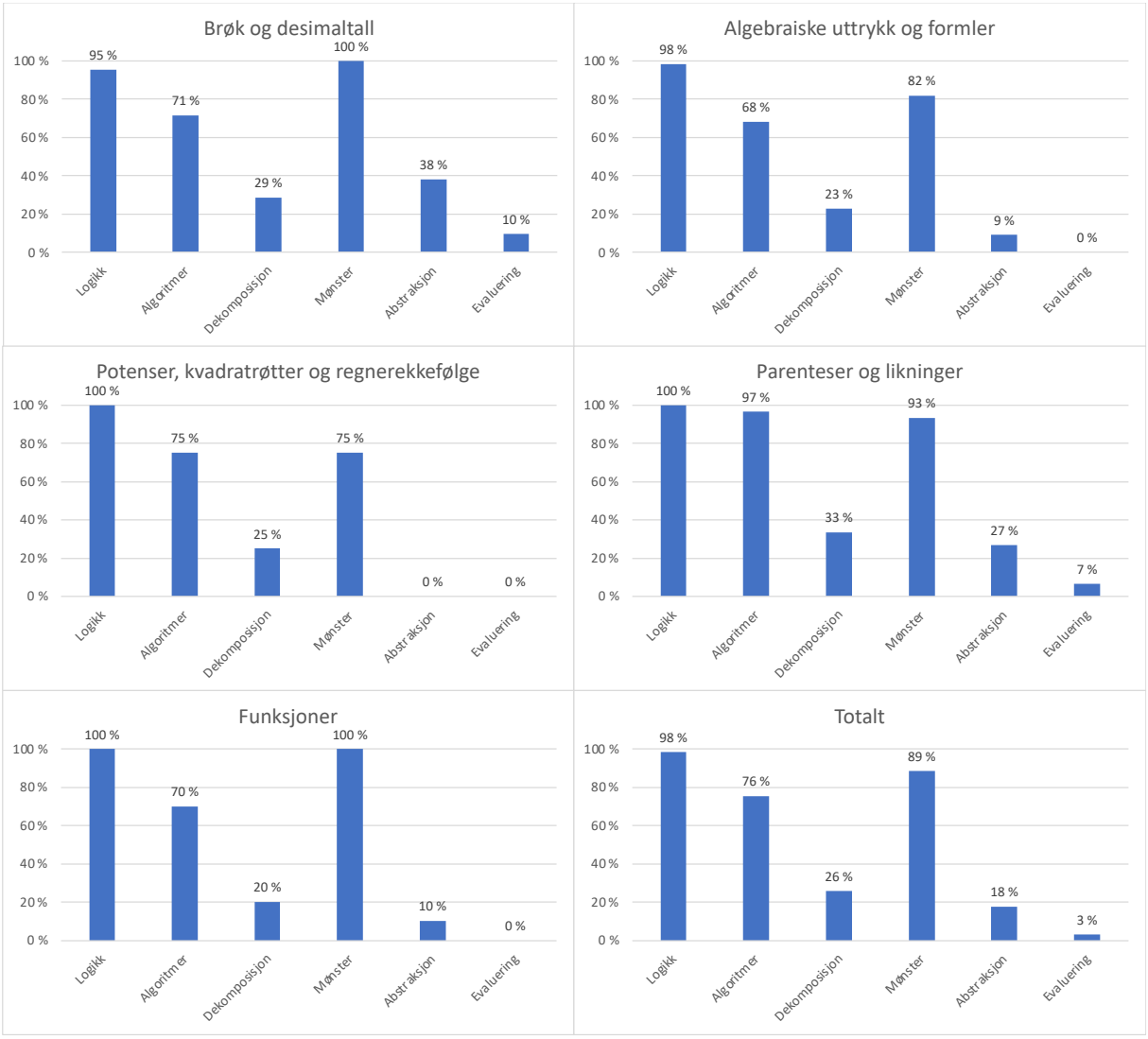
Kategoriene *oppgaver* og *nivå 1* inneholder flest M-oppgaver (45 og 52 %). De andre kategoriene inneholder mellom 0 og 16 % slike oppgaver. Det ser ut som at M-oppgaver har en tendens til å utgi færre av POiM jo høyere nivået av differensierte oppgaver er. Det skapes ingen tydelig trend i fordeling av alle kognitive krav for de nivådifferensierte kategoriene, men for *nivå 1* og *nivå 2* ser ut som at M- og PUS-oppgaver skifter vektlegging av deres fordeling. Dette betyr at *nivå 2* vektlegger flere PUS-oppgaver og færre M-oppgaver i forhold til *nivå 1*. I *nivå 3* er det lav fordeling av begge disse kognitive kravene. Det er interessante at *nivå 3* er den eneste oppgavekategorien som har høyere andel høyt enn lavt kognitivt krevende oppgaver. Dette ser vi spesielt gjennom PMS-oppgaver. For *nivå 1* og *nivå 2*, ligger slike oppgaver på 30 og 29 %. I *nivå 3* derimot, dobles fordelingen av PMS-oppgaver fra *nivå 2* til 58 %. I tillegg er *nivå 3*, sammen med *diskusjon*, de eneste kategoriene hvor GM-oppgaver blir gitt.

4.3.2 Nøkkelbegrep

I tillegg til *kognitive krav*, ble POiM undersøkt for *nøkkelbegrep* fra *den algoritmiske tenkeren* av Csizmadia et al. (2015) gjennom den vertikale analysen. Her ble POiM identifiserte med kvalitetene *logikk*, *algoritmer*, *dekomposisjon*, *mønster*, *abstraksjon* og *evaluering*. Resultatene fra denne undersøkelsen presentert gjennom Tabell 20 som viser antall tilfeller av *nøkkelbegrep* under hver av de fem matematiske temaene, deretter presenteres fordelingen av kvalitetene fra tabellen i Figur 20 for hvert tema og for alle POiM referert til som totalt. Diagrammene sine verdier, regnes ut ved å dele antall tilfeller av *nøkkelbegrep* på antall POiM for det korresponderende temaet.

Tabell 20: Programmeringsoppgavenes nøkkelbegrep fordelt ut over matematiske temaer.

	Brøk og desimaltall	Algebraiske uttrykk og formler	Potenser, kvadratrøtter og regnerekkefølge	Parenteser og likninger	Funksjoner	Totalt:
Logikk	20	65	4	30	10	**129**
Algoritmer	15	45	3	29	7	**99**
Dekomposisjon	6	15	1	10	2	**34**
Mønster	21	54	3	28	10	**116**
Abstraksjon	8	6	-	8	1	**23**
Evaluering	2	-	-	2	-	**4**
Antall oppgaver:	**21**	**66**	**4**	**30**	**10**	**131**



Figur 20: Fordelingen av programmeringsoppgavers kvaliteter av nøkkelbegrep for alle matematiske temaer.

Tabell 20 kan leses på to måter. Langs rader for å se hvilke matematiske temaer som inneholder et *nøkkelbegrep* eller nedover kolonner for å se hvilke *nøkkelbegrep* som finnes i ulike temaer. Et eksempel på sist nevnte ved *brøk og desimaltall*, kan leses som det inneholder tilfeller av *logikk* (20), *algoritmer* (15), *dekomposisjon* (6), *mønster* (21), *abstraksjon* (8) og *evaluering* (2). I tabellen vises antall tilfeller for alle POiM til høyre. Av 131 POiM inneholder 129 av dem kvaliteten *logikk*. *Mønster* (116) og *algoritmer* (99) er også nøkkelbegrep med flere tilfeller. Mer sjeldne tilfeller av nøkkelbegrep, er *dekomposisjon* (34), *abstraksjon* (23) og *evaluering* (4).

Figur 20 viser fordelingen av tilfeller med nøkkelbegrepkvaliteter til POiM for hvert *matematiske tema* og sammenlagt i «Totalt». Hvert diagram leses ved å se prosenten over

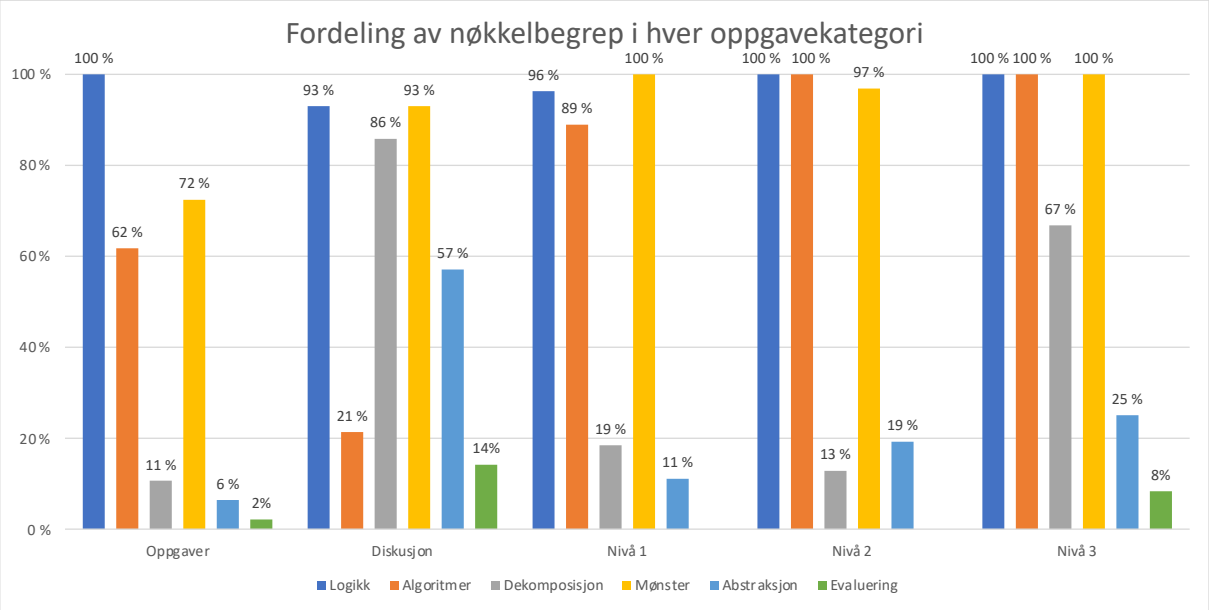
stolpen til nøkkelbegrepet nevnt nederst. *Brøk og desimaltall* kan dermed leses som at alle POiM i dette temaet inneholder nøkkelbegrepet *logikk* (95 %), *algoritmer* (71 %), *dekomposisjon* (29 %), *mønster* (100 %), *abstraksjon* (38 %) og *evaluering* (10 %). I avsnittet over ble tilfeller av *nøkkelbegrep* delt i to. Det eksisterer en gruppe nøkkelbegrep som er funnet ofte og en sjeldent, som sees igjen i diagrammene. *Nøkkelbegrep* med flere tilfeller varierer fra 68 til 100 %, færre tilfeller varierer fra 0 til 38 %.

Ved å gjennomføre en enkel test av resultatene for *nøkkelbegreper* til hver av de matematiske temaene i dette studiet, kan vi se hvilket tema som inneholder flest *nøkkelbegrep*. Testen baserer seg på å summere alle prosent for tilfeller av *nøkkelbegrep* fra Figur 20 og behandle de som poeng. Maksimalt poeng et tema kan få er 600. For å få en referanse å forholde poengene til, regnes det ut poeng av totalt først som resulterer i 310 poeng. Testen viser at POiM fra *brøk og desimaltall* får 343 poeng, *algebraiske uttrykk og formler* (280), *potenser, kvadratrøtter og regnerekkefølge* (275), *parenteser og likninger* (357) og *funksjoner* (300). *Parenteser og likninger* og *brøk og desimaltall* er derfor de temaene med flest tilfeller av nøkkelbegrep, er over gjennomsnittet og det er bare her *evaluering* eksister.

Funnene av *nøkkelbegrep* kan også settes opp mot *oppgavekategorien* de tilhører, slik som det ble gjort for de *kognitive kravene*. Det vises først oversikt gjennom Tabell 21, som viser *oppgavekategoriene* øverst og tilfeller *nøkkelbegrep* under. Resultatene vises deretter i Figur 21 som stolpediagram. Prosentene i diagrammet er regnet ut ved å dele antallet av et *nøkkelbegrep* på antall POiM i den aktuelle *oppgavekategorien*.

Tabell 21: Programmeringsoppgavenes nøkkelbegrep fordelt ut over oppgavekategorier.

	Oppgaver	Diskusjon	Nivå 1	Nivå 2	Nivå 3	Totalt:
Logikk	47	13	26	31	12	**129**
Algoritmer	29	3	24	31	12	**99**
Dekomposisjon	5	12	5	4	8	**34**
Mønster	34	13	27	30	12	**116**
Abstraksjon	3	8	3	6	3	**23**
Evaluering	1	2	-	-	1	**4**
Antall oppgaver:	**47**	**14**	**27**	**31**	**12**	**131**



Figur 21: Fordeling av nøkkelbegrep i hver oppgavekategori.

Tabell 21 kan leses nedover i kolonnene for å se hvilke nøkkelbegreper som er i en oppgavekategori. For eksempel i *nivå 1* er det tilfeller av *logikk* (26), *algoritmer* (25), *dekomposisjon* (5), *mønster* (27), *abstraksjon* (3) og ingen *evaluering* i 27 POiM. Den kan også leses langs rader, for å se hvor mange tilfeller det er av et nøkkelbegrep.

Figur 21 leses ved å se på stolpene over en *oppgavekategori* som har opp til 6 stolper hver. Disse er fargekodet og forklart nederst i figuren. *Nivå 1* blir lest som å inneholde *logikk* (96 %), *algoritmer* (89 %), *dekomposisjon* (19 %), *mønster* (100 %), *abstraksjon* (11 %) og ingen *evaluering*.

Noe som er verdt å bemerke seg fra disse resultatene, er at *logikk* forekommer nokså jevnt i alle kategorier med en variasjon mellom 93 og 100 %. Med unntak i *oppgaver*, så er det samme variasjon for *mønster*. *Algoritmer* er i 89 – 100 % av POiM for de differensierte nivåene, men 62 % i *oppgaver* og 21 % i *diskusjon*. Det disse tre nøkkelbegrepene har til felles, er at de alle forekommer i mer en halvparten av alle POiM. De resterende nøkkelbegrepene forekommer i under halvparten av dem. I likhet med funnet for *nøkkelbegrep* og *matematiske temaer*, er det et skille hvor noen *nøkkelbegrep* brukes ofte og sjeldent. Dekomposisjon brukes i tre oppgavekategorier bare i fra 11 til 19 % av POiM, men i *diskusjon* og *nivå 3* brukes det i 67 og 86 %. *Evaluering* brukes mest i *diskusjon* (14 %)

og *nivå 3* (8 %), det eksisterer ellers bare i *oppgaver* (2 %). Dette gjentas også for *abstraksjon*, hvor disse to oppgavekategoriene brukes dette ofte i forhold til de andre.

4.4 Oppsummering og spørsmål til drøfting

I kapittel 4.3.1 viste resultatene at fordelingen av POiM i matematiske temaene var ulik. Årsaker for hva som kan ha påvirket denne fordelingen drøftes i kapittel 5.1.

Figur 19 viste at GM-oppgaver kun fantes i oppgavekategoriene *diskusjon* og *nivå 3*. Den viste også at *nivå 3* var den eneste med POiM som hadde høyere fordeling av høyt kognitive krav enn lave. Disse resultatene drøftes i kapittel 5.2.

Fordelingen av kognitive krav viser også at *algebraiske uttrykk og formler* er det temaet med flest POiM. Av disse oppgavene er det ingen som kategoriseres som GM, men Tabell 18 viser at *parenteser og likninger* og *brøk og desimaltall* med færre POiM inneholder GM-oppgaver. I tillegg viser Tabell 20 at disse to temaene er de eneste som også inneholder nøkkelbegrepet *evaluering.* Hvorfor *algebraiske uttrykk og formler* ikke har kvalitetene GM og *evaluering* når de to andre temaene med nokså lik fordeling av lave og høye kognitive krav har det, drøftes i kapittel 5.3.

Figur 20 viste antall nøkkelbegrep funnet i POiM for hvert matematiske tema. Der var det tydelig at *logikk*, *mønster* og *algoritmer* var gjenkjent mye oftere enn de resterende (*dekomposisjon*, *abstraksjon* og *evaluering*). Figur 21 viser også at noen av de mer sjeldne nøkkelbegrepene brukes mer i oppgavekategoriene *diskusjon og nivå 3*. Denne forskjellen mellom ofte og sjeldent brukte nøkkelbegrep diskuteres i kapittel 5.3.

5 Diskusjon

Å analysere strukturen til læreverkene hadde tre hovedfunksjoner i dette studiet. Det var å kartlegge oppgavekategorier og leksjoner, danne oversikt over hvilke matematiske temaer som var mulige å identifisere POiM med, og hvilke teoretisk grunnlag elevene får før møtet med en POiM. Dette resulterte i at oppgavene fra begge læreverkene kunne grupperes som *oppgaver*, *diskusjon*, *nivå 1*, *nivå 2*, *nivå 3* og *nivå 4*. *Nivå 4* inneholdt ingen POiM. Dette viser, basert på oppgavekategorienes definisjon gitt i kapittel 4.1.2.5, at læreverkene legger mer vekt på å konstruere POiM som ikke overgår forventningsnivået til hva læreverkene mener kan forventes av elevene. Grunnstruktur ga også et overblikk over hvor faglig innhold som var relevant for horisontal og vertikal analyse befant seg. Kjennskap til plassering av oppgaver var nødvendig for å si hva som var gjennomgått i læreverkene ved lineær tilnærming og kunne kalles for elevenes teoretiske grunnlag.

5.1 Matematiske temaer for POiM

Oversikten over POiM sine matematiske temaer besvarer forskningsspørsmål 1 og stammer ut fra å se på læreverkets læringsmål og temaer. I det ene læreverket, *Matemagisk 8*, var POiM plasserte i ulike kapittel med egne matematiske temaer og læringsmål for programmering. Denne strukturen var ledende i avgjørelsen av hvilket matematisk tema de ulike POiM tilhørte fordi plasseringen av oppgavene i de gitte kapitelene var knyttet opp til spesifikke matematiske temaer. I *Campus Matte 8* var POiM ikke plassert under kapitler med matematiske temaer, derfor måtte oppgavene kategoriseres i tråd med metoden i kapittel 3.3.1.2 som tillater meg å forsøke å kategorisere oppgavene etter eksisterende temaer, eller etter de nye temaene (se Tabell 14). De endelige temaene som POiM fra *Campus Matte 8* kunne plasseres under vart identifisert gjennom den vertikale analysen. For alle POiM hvor dette måtte gjøres, kunne de plasseres i en av de eksisterende temaene. Dette resulterte i at *brøk og desimaltall*, *algebraiske uttrykk og formler*, *potenser, kvadratrøtter og regnerekkefølge*, *parenteser og likninger* og *funksjoner* var de matematiske temaene som POiM fra studiet er plassert under.

Resultatene fra Tabell 17 viser derimot at distribueringen av POiM mellom temaer i læreverkene er ulike. De to temaene som inneholder flest POiM fra *Matemagisk 8* er

algebraiske uttrykk og formler (43) og *brøk og desimaltall* (16). For *Campus Matte 8* er det også temaet *algebraiske uttrykk og formler* (23) i tillegg til *parenteser og likninger* (21). De to resterende temaene inneholdt til sammen 10 og 4 POiM. For å forklare hvorfor fordelingen av antall POiM er så ulik blant temaene, kan vi se på resultatene fra den horisontale analysen. I Tabell 11, som viser delkapitlene til *Matemagisk 8*, er det sammenheng mellom de tre matematiske temaene som har flest POiM og delkapitlene. For kapittel *Brøk og desimaltall*, *Algebraiske uttrykk og formler* og *Parenteser og likninger* finnes det egne delkapittel relatert til programmering (delkapittelkode: 2D, 3C og 6C) og er de eneste delkapitlene som hadde videoleksjoner om programmering (se Tabell 15). Det eksisterte ikke slike delkapittel for de to andre temaene i studiet. Når det gjelder *algebraiske uttrykk og formler* med flest POiM, kan dette forklares ved at det var det eneste kapittelet i *Matemagisk 8* som valgte å bruke programmering i to delkapittel. For *Campus Matte 8*, som ikke hadde et spesifikt matematisk tema knytt til det kapittelet POiM befant seg under, vart oppgavene plassert under de tre nevnte temaene i dette avsnittet. Dette indikerer at resultatene fra studiet viser til at *brøk og desimaltall*, *algebraiske uttrykk og formler* og *parenteser og likninger* er matematiske temaer som kan egne seg til å integrere programmering inn i matematikk for 8. trinn.

5.2 Kognitive krav for POiM

Funnene gjort gjennom den vertikale analysen i kapittel 4.3.1, viser at de fleste POiM fra de to læreverkene representerer det kognitive kravet PUS (41 %). Andre krav som ble i nokså stor grad vektlagt, var M (31 %) og PMS (25 %). Det er kun den siste kategorien, GM (3 %), som blir i liten grad vektlagt. Samlet gir resultatene for de *kognitive kravene* til POiM en fordeling på 72 % lavt og 28 % høyt kognitivt krevende. Dette betyr at de fleste POiM funnet i de to utvalgte læreverkene baserer seg på å lære bort hvordan man lager og bruker koder til å lage program innenfor matematisk kontekst. De høyt kognitivt krevende POiM forsøker derimot å skape sammenheng mellom matematiske ideer og konsepter gjennom programmering. Resultatets overvekt av lavt kognitivt krevende POiM kan ha sammenheng med at utvalget baserer seg på læreverk for det første trinnet av ungdomsskolen. Det vil derfor være en mulighet for at læreverkene er designet med utgangspunkt i å lære seg grunnlegendeferdigheter om hvordan programmering kan brukes i matematikk på 8. trinn og derfor gi mer kognitivt krevende POiM med mer fokus på utforsking for læreverkene på 9.

og 10. trinn. *Matemagisk 8* sin beslutning av å introdusere tekstprogrammering, som nevnt i kapittel 4.1.2.2, kan også ha påvirket denne fordelingen og støtter argumentet. Det vil derfor kunne være et større fokus på å lære seg å programmere i matematikk enn å utforske matematikk gjennom programmering på 8. trinn. *Campus Matte 8* introduserer ikke et nytt programmeringsspråk, blokkprogrammeringen er kjent fra deres læreverk på lavere trinn og gir derfor ikke støtte til argumentet.

Heimstad og Strand (2018) sitt studium resulterte i en fordeling på 73,7/26,3 %, (lave/høye kognitive krav) for alle oppgaver i læreverkene *Maximum* og *Faktor* for ungdomstrinnet. POiM sine *kognitive krav* ligger derfor nær deres funn som viser at kravene for POiM ligger innenfor et forventet gjennomsnitt av alle oppgaver fra deres studium. Det er derfor interessant å se om fordelingen av kognitive krav for studiets POiM kan sammenlignes med andre sine resultater knytt til spesifikke temaer. I likhet med meg, grupperte Heimstad og Strand (2018) oppgaver inn i matematiske temaer og viser til deres fordeling av kognitive krav. Deres temaer er sammensatt litt annerledes enn dette studiet, men representerer likevel sammenlignbare resultater. *Brøk og prosent* (84/16 %) (Heimstad & Strand, 2018, s. 84) som var et av deres temaer kan delvis sammenlignes med temaet *brøk og desimaltall* (71/29 %) fra mitt studie (se Figur 18). Ettersom fordelingen for POiM i *brøk og desimaltall* på 8. trinn vektlegger flere høyt og færre lavt kognitivt krevende oppgaver enn alle *brøk og prosent* oppgaver for 8. til 10. trinn, kan programmering i matematikk være en ressurs for å bidra til mer utforskende arbeid i dette temaet.

Temaet *algebra og likninger* (81/19 %) av Heimstad og Strand (2018, s. 84) kan sees i sammenheng med mine temaer *algebraiske uttrykk og formler* (71/29 %) og *parenteser og likninger* (73/27 %). Det kan også studiet til Hofstad og Liland (2021, s. 58-59). Deres studie viste fordelingen av kognitive krav for algebraoppgaver i læreverkene *Matemagisk 8* (55/45 %) og *Matematikk 8* (79/21 %). Utfra disse resultatene er det bare algebraoppgaver i *Matemagisk 8* som har en fordeling som vektlegger flere høyt kognitivt krevende oppgaver enn POiM under *algebraiske uttrykk og formler*, mens i *Matematikk 8* vektlegges det flere lavt kognitivt krevende oppgaver. POiM under *algebraiske uttrykk og formler* og *parenteser og likninger* inneholdt derimot flere høyt kognitivt krevende oppgaver enn alle oppgaver under *algebra og likninger* i læreverkene *Maximum* og *Faktor*. Dette viser at arbeid med

POiM ikke nødvendigvis bidrar med mer utforsking og mindre memorerende aktivitet for ulike matematiske temaer, det kan være avhengig av hvilke læreverk man henter oppgaver fra.

POiM sine *kognitive krav* i de ulike matematiske temaene har en nokså lik fordeling. Unntaket er POiM under *potenser, kvadratrøtter og regnerekkefølge*. Der var fordelingen lik mellom lavt og høyt. Det var bare fire POiM i dette studiet som kunne plasseres under dette temaet, det gjør utvalget er lite representativt for å drøftes opp mot andre temaer. Dette gjaldt også for *funksjoner* med ti oppgaver og en fordeling på 80/20 %. De andre temaene bestod av til sammen 117 POiM og har fra 71 til 73 % lavt kognitivt krevende POiM. Fordelingen av lave og høye kognitive krav er derfor relativt lik gjennomsnittet av fordelingen for alle POiM på 72/28 %.

Fordelingen endres drastisk om man ser på *oppgavekategorier*. For POiM med det kognitive kravet M, minker deres forekomst fra 52 % til 8 % når man går fra læreverkenes oppgavedifferensierings *nivå 1* til *nivå 3*. Ser vi på *nivå 3* er fordelingen av *kognitive krav* på 16/84 %. Dette er den eneste oppgavekategorien som har større andel høyt kognitivt krevende POiM i forhold til lavt. Dette samsvarer med Kongsnes og Wallace (2020, s. 4-5) og Thue et al. (2022) beskrivelse av læreverkenes oppgavekategorier. *Nivå 1* består av oppgaver med trening eller repetisjon av oppgaver som er gjort i fellesskap, dette vil resultere i flere M-oppgaver. *Nivå 2* baserer seg både på repetisjon og å ta et steg videre, dette resulterte i færre M- og flere PUS-oppgaver. Til slutt er *nivå 3*, som er mer utfordrende og krever mestring av de lavere nivåene. Disse resulterte i et lavt antall M- og PUS- oppgaver og stort antall PMS-oppgaver.

GM-oppgaver eksisterte kun i *diskusjon* og *nivå 3*. Smith og Stein (1998) og Valenta (2016) beskrev selv det kognitive kravet GM som oppgaver hvor det kreves utforsking rundt flere elementer og sammensetting av dette på et komplekst nivå for å skape en løsning. Dette ville også kreve at elevene tenker over egen utvikling. Sammen med Kongsnes og Wallace (2020, s. 4-5) og Thue et al. (2022) beskrivelse av oppgavekategoriene gitt i kapittel 4.1.2.5, går oppgavene i *nivå 3* over forventningsnivået til hva læreverkene forventer av elevene. Oppgaver med denne beskrivelsen i *nivå 3* vil dermed kunne knyttes opp mot beskrivelsen

av GM-oppgaver, siden de krever høy kognitiv aktivitet for å løses. *Diskusjon* ble beskrevet som en aktivitet hvor elevene diskuterer matematikk og forklarer hvordan man tenker. I siste trinn av problemløsningsprosessen til Polya (2014), vil refleksjon over løsningen kunne bidra til egen utvikling av matematisk forståelse. Gjennom samtaler, hvor elevene skal diskutere og forklare deres synspunkt, må de reflektere over egen tekning for å sette ord på deres oppfatninger og formidle dette videre. Det skapes derfor en mulighet for at GM-oppgaver kan oppstå gjennom *diskusjon*.

Ettersom at resultatene viser overvekt av lavt kognitivt krevende POiM, kan man se på hva som kjennetegner dem i dette studiet. Hva er det som gjør at flertallet av POiM går under lave kognitive krav? Gjennom innholdsanalysen påvirket det teoretiske grunnlaget ofte hvilket *kognitivt krav* en POiM ble identifisert med. Leksjoner, videoleksjoner, eksempler, formler og regler fra læreverkene ga mye nødvendig informasjon og støtte for å skape en løsning til POiM i analysen. I de fleste tilfellene, kunne man finne forklaringer eller eksempler for hvordan løsningsprogrammet kunne skrives. Matematiske elementer, knytt til programmet, ble ofte forklart gjennom videoleksjoner. Mye av fokuset til POiM var derfor på å lære seg ulike koders funksjoner og bruksområder gjennom memorering eller gjentakelse av program. Læreverket *Campus Matte* 8 inneholdt 49 av 131 POiM i dette studiet. Til disse var det 12 videoleksjoner på til sammen 23 minutter og 58 sekunder (se Tabell 16) fordelt på to delkapittel. Om vi ser på hva gjennomsnitts lengden for videoleksjoner per delkapittel er, får vi 11 minutter og 59 sekunder. I tillegg var flere av deres POiM nokså like i oppgaveteksten og kunne løses med like strategier. Det er derfor en mulighet at læreverket ønsket å bruke deres ressurser til å gi elever opplæring i hvordan man lager og bruker strategiene til å løse problemer. Et resultat av dette er at det er mer fokus på reproduksjon og strategiers bruksområdet som resulterer i flere POiM av lave kognitive krav.

Matemagisk 8 inneholdt 82 av de 131 POiM i dette studiet. I bokens første møte med programmering forklarte lærerveilederen at de valgte å introdusere *Python* tekstprogrammering (Aunivers, 2021). Det dette betyr for elevene, er at dersom de har følget *Matemagisk* sitt læreverk for lavere trinn er dette kanskje deres første møte med tekstprogrammering. Læreverket har dermed valgt å bruke en del av oppgavene deres med programmering til å lære nødvendige koder og hvordan de fungerer. I sammenheng med

funnene over, kan dette være en årsak for mengden av lavt kognitivt krevende POiM. Dette fører til at det blir en del repetisjon av skriving og bruk av program uten betydelig matematisk sammenheng. For POiM av høyrere kognitive krav, var det større søkelys på å utforske matematiske problem gjennom programmering.

For matematikkfaget viser resultatene hvordan POiM legger til rette for utforsking av strategier og framgangsmåter for å finne sammenhenger og mønstre mellom strategi og matematikk. Selv om 72 % av POiM fra dette studiet er av lave kognitive krav og ikke knyttes direkte til utforsking men til prosedyre, støtter de slik aktivitet basert på Kieran (2013) og Opheim og Simensen (2020, s. 107) argumentasjon. Deres argumenter viser til viktigheten av å balansere ulike tilnærminger, i dette tilfellet kognitive krav hvor de lave vektlegges mest, for å trene på prosedyreorienterte ferdigheter som er nødvendig for utforsking. POiM av lave kognitive krav, som jobber med teknikker for å skrive program for kjente matematiske fenomener, er derfor nødvendig for å støtte og komplementere ferdigheter som trengs for å utføre en utforskende tilnærming gjennom høye kognitive krav. Dette betyr i følge Opheim og Simensen (2020, s. 102) at programmering i matematikk har en mulighet til å tilrettelegge for utvikling av rike erfaringer og dybdelæring i matematikk gjennom arbeid med POiM av ulike kognitive krav og kjerneelementet *utforsking og problemløsning*. At læreverkene inneholder flere POiM av lave kognitive krav kan også bety at de ønsker å tilby en variasjon av POiM som vektlegger både programmeringsaktiviteter som trener på grunnlegendeferdigheter og utforsking av matematiske sammenhenger og ideer.

5.3 Algoritmisk tenkning i POiM

I den vertikale analysen ble POiM også undersøkt for hvilke *nøkkelbegreper* fra *den algoritmiske tenkeren* (Csizmadia et al., 2015) elever oppfordres til å bruke gjennom løsningen. Av alle POiM i dette studiet var *logikk* (98 %), *mønster* (89 %) og *algoritmer* (76 %) de tre mest identifiserte nøkkelbegrepene. De tre siste er identifisert i betydelig mindre grad: *dekomposisjon* kunne identifiseres i 26 % av programmeringsoppgavene, *abstraksjon* i 18 % og *evaluering* i bare 3 %.

Logikk forekommer i nesten alle POiM i dette studiet. Csizmadia et al. (2015) definerte *logikk*, forklart i kapittel 2.5.1, som å skape mening til problemet gjennom analyse av det og

finne relevant informasjon for å se for seg en løsningsstrategi. Gjennom resonnering av problemet får elevene brukt tidligere kunnskap i en ny kontekst (Csizmadia et al., 2015). I kapittel 5.2 ble det diskutert hvorfor flertallet av POiM gikk under lave kognitive krav. Her var en av argumentene at det ene læreverket valgte å introdusere tekstprogrammering gjennom *Python*. Dette førte til at mye av innholdet om programmering i leksjonene, videoleksjonene og eksempler forsøker å formidle og utforske grunnleggende koder i programmeringsspråket. Elevene vil derfor i de fleste tilfeller kunne bruke *logikk* i løsningen av POiM fordi de kan analysere problemet og resonnere rundt hvordan tidligere kunnskap kan brukes i løsningen. Dette gjelder også for *mønster*. Csizmadia et al. (2015) inkluderte å bruke tidligere løsninger av andre problemer for å skape en løsning av et nytt problem i deres definisjon av nøkkelbegrepet *mønster* (se kapittel 2.5.1). Her kan man se sammenhenger og likheter mellom tidligere løsninger for å løse hele eller deler av det nye problemet (Csizmadia et al., 2015). Ettersom POiM ofte var repeterende, diskutert i kapittel 5.2, vil elevene ofte kunne bruke løsninger fra andre problemer som grunnlag for en ny løsning av et annet problem.

Algoritmer var den tredje største forekomsten i studiet med 76 %. For 8. trinn er det et kompetansemål som inneholder programmering. «Utforske hvordan algoritmer kan skapes, testes og forbedres ved hjelp av programmering» (Kunnskapsdepartementet, 2020). *Algoritmer* ble definert som å lage og vite trinn-for-trinn med klare definisjoner hvordan man kan gjennomføre og løse et problem (Csizmadia et al., 2015). Funnene fra studiet viser at gjennom løsning av POiM fra læreverkene *Matemagisk 8* og *Campus Matte 8*, kan *algoritmer* brukes ofte og at kompetansemålet over blir ivaretatt. Studiets funn presenterer ikke en oversikt over om det er å lage, teste eller forbedre algoritmer som gjennomføres oftest, men gjennom analyseprosessen var det oftest testing og laging av algoritmer som var aktuelt. Dette kan sammenlignes med mengden lavt kognitivt krevende POiM i kapittel 5.2. For å forbedre algoritmer gjennom programmering, må man utforske og forstå hvorfor algoritmen er som den er basert på underliggende matematiske ideer og konsepter (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 5). Dersom det var flere POiM som baserte seg på forbedring av algoritmer gjennom programmering, ville det muligens resultert i en høyere andel av høyt kognitivt krevende oppgaver.

Koblingene gjort i Figur 2 viser at de tre oftest identifiserte nøkkelbegrepene gir støtte til de to første stegene i problemløsningsmodellen *å forstå problemet* og *lage en plan* av Polya (2014). Det tredje steget, *gjennomføre planen,* er bare dekket gjennom *arbeidsmåter*. Det er kun *å se tilbake,* det siste steget i modellen, som disse nøkkelbegrepene ikke knyttes opp mot. Det vil si at dersom elevene jobber med slike POiM vil de få trening i flere steg av problemløsningsmodellen, men hele problemløsningsprosessen dekkes ikke gjennom bare disse nøkkelbegrepene.

Hvorfor er det et stort skille av ofte og sjeldne tilfeller av *nøkkelbegrep*? Basert på argumentene fra de to avsnittene over, ble *logikk* og *mønster* brukt ofte fordi POiM var repeterende og løsningen kunne knyttes opp mot leksjoner og tidligere løsninger. *Algoritmer* kunne knyttes opp mot et kompetansemål og ble derfor brukt ofte. Hva med de andre nøkkelbegrepene? For å diskutere hvorfor disse brukes mer sjeldent, vil jeg ta fram *arbeidsmåter* og *type svar* fra Tabell 17. Av totalt 131 POiM, er 93 under å *skape* og 107 under *kun svar*. Gjennom den vertikale analysen var flere av disse POiM direkte og klare i oppgaveteksten. Typisk formidling av oppgavetekst var «lag et program som gjør ___». Både *dekomposisjon* og *abstraksjon* ble definert av Csizmadia et al. (2015) som nøkkelbegrep hvor man må bryte ned problemet i mindre deler og skille vekk unødvendig informasjon for å skape mening til et problem. Når oppgaveteksten til disse POiM er så konkrete, er det få eller ingen deler man kan bryte ned. Dette gjelder også for fjerning av unødvendig informasjon, som vil være vanskelig når det blir oppgitt lite fra før. *Dekomposisjon* og *abstraksjon* forekommer, i dette studiet, oftere i POiM som krever at elevene må lokalisere deler av koder for å gjøre endringer der.

I testen som regnet ut poeng av Figur 20, var *parenteser og likninger* og brøk og *desimaltall* de med flest poeng. Disse to var også de eneste temaene som inneholder tilfeller av *evaluering*. POiM som inneholdt *evaluering*, eksister kun under arbeidsmåtene å *fikle*, *feilsøke* og *samarbeide*. Resultatene stemmer overens med koblingene mellom komponenter fra *den algoritmiske tenkeren* (Csizmadia et al., 2015) og problemløsningsmodellen til Polya (2014) som vist i kapittel 2.5.3. Tabell 17 viser at 93 av 131 POiM går under å skape. Ingen av disse inneholder *evaluering*. Ved å bruke POiM som bruker andre *arbeidsmåter* enn å *skape*, vil det i forhold til resultatene i dette studiet,

muligens øke tilfeller hvor bruk av *evaluering* skjer i løsningen. Dette henger sammen med trinnet *å se tilbake* av Polya (2014), hvor å se tilbake på løsningen og tenke over om den er god nok kan bidra til å økt bruk av nøkkelbegrepet. Gjennom arbeidsmåtene å *fikle*, *feilsøke* og *samarbeide*, vil man kunne tenke over en løsning, gjøre endringer og eventuelt diskutere løsningen med andre og finne forbedringspotensial i løsningen. En ting som igjen bør nevnes, er at i dette studiet analyserte jeg POiM for om nøkkelbegrepene forekommer naturlig gjennom oppgaveteksten. Elevene kan, om de ønsker, selv evaluere egne løsninger uten at oppgavene oppfordrer til det. Studiet følger oppbyggingen av læreverket og kan derfor ikke danne en konklusjon på dette.

Videre kan vi se om dette kan kobles opp mot resultatene av at *algebraiske uttrykk og formler*, som har rundt halvparten av POiM i studiet, ikke inneholdt noen GM-oppgaver i kapittel 4.3.1. Både *brøk og desimaltall* og *parenteser og likninger* inneholder *evaluering* og GM-oppgaver. *Algebraiske uttrykk og formler* inneholdt ingen av delene. Vil det bety at dersom læreverkene tilrettelegger for mer *evaluering* i oppgaveteksten til POiM, så vil antall GM-oppgaver øke? Ut i fra teori i kapittel 2, kan elevene utvide sin egen matematiske forståelse gjennom refleksjon i *evaluering* (Polya, 2014), og GM-oppgaver vil kreve at elevene tenker over egen utvikling (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 7). Det er derfor en mulighet for at *evaluering* og GM kan knyttes tett opp mot hverandre. Studiets funn inneholder ikke nok GM-oppgaver til å kunne konkludere om dette har en virkning.

På samme måte som GM-oppgaver var kun funnet i *diskusjon* og *nivå 3*, er det disse to oppgavekategoriene som har POiM med flest identifiserte nøkkelbegrep av *dekomposisjon, abstraksjon og evaluering*. Flere av argumentene brukt for de kognitive kravene kan nok trekkes inn her. Blant annet ble beskrivelsen av *nivå 3* diskutert opp mot høyt kognitivt krevende oppgaver. For at elevene skal finne en løsning må de utvikle en dypere forståelse av matematiske ideer og konsepter (Smith & Stein, 1998, s. 348; Valenta, 2016, s. 5-6). *Nivå 3* kan inneholde flere tilfeller av *dekomposisjon* i forhold til andre oppgavekategorier, fordi å skape en dypere matematisk forståelse krever at elevene kan begrunne hvorfor ulike deler av løsningen er konstruert slik som den er. Gjennom samtale i *diskusjon*, må elevene forklare hvordan de tenker og diskutere deres syn med andre (Kongsnes & Wallace, 2020; Thue et al., 2022). Basert på Csizmadia et al. (2015) definisjon av nøkkelbegrepene, kan elevene

gjennom samtale bryte ned problemet eller løsningen i ulike deler og forklare deres relevans. Her må elevene skape mening til dette ved å påpeke hva som er viktig og uviktig informasjon. Når en løsning blir lagt fram kan den evalueres om den er god nok eller om den bør endres. *Dekomposisjon*, *abstraksjon* og *evaluering* kan derfor passe naturlig inn i *diskusjon* og GM-oppgaver.

For matematikk vil resultatene av *nøkkelbegrep* funnet i POiM vise hvilke deler av problemløsningsmetoden *algoritmisk tenkning* som oftest blir tilrettelagt. De tre nøkkelbegrepene som er gjenkjent oftest vil, med bakgrunn i deres definisjoner forklart i kapittel 2.5.1, bidra til å utvikle elevers ferdigheter til å løse praktiske problemer med matematikk og trene på problemløsningsprosedyren. POiM med mer tilrettelegging for *dekomposisjon* og *abstraksjon,* som er mindre gjenkjente *nøkkelbegrep* i dette studiet, vil bidra med å utvikle ferdigheter til å bryte ned komplekse problemer og å analysere deres deler. *Evaluering* i studiets POiM, som ble gjenkjent i 3 % av alle POiM, er den eneste koblingen av *nøkkelbegrep* i Figur 2 som knyttes opp mot Polya (2014) trinn *å se tilbake*. Det vil si at de fleste POiM gjennom oppgavetekst ikke tilrettelegger for at elevene selv må vurdere løsningen deres for effektivitet og gyldighet og å lære av deres eventuelle feil eller mangler. Studiet tar ikke høyde for hvordan POiM blir utført i praksis. Det er derfor en mulighet for at *evaluering* er en kvalitet som kan bli utført i større grad på egent initiativ fra elevene, det ser vi gjennom resultatene og koblingene til arbeidsmåtene *fikle, feilsøke og samarbeide* i Figur 2. Likevel dekker problemløsningsaktiviteten gitt i læreverkenes POiM kjerneelementet *utforsking og problemløsning*, men oppgavene legger mer opp til trening i problemløsningsprosessen enn vurdering av dens gyldighet. Med utgangspunkt i Torkildsen (2017, s. 2) fremmer problemløsningsaktiviteten funnet i studiets POiM til forståelse og dybdelæring i matematikk.

5.4 Rammeverket

Rammeverket har bidratt til å danne klare rammer for hvordan innholdet i undersøkelsen skulle analyseres. Det har nok vært til min fordel at deler av rammeverket har blitt utforsket og prøvd ut av Charalambous et al. (2010) da flere argumenter for valgmuligheter var definert på forhand av analysen. Strukturen gjorde det enkelt for meg å utvide rammeverket slik at det kunne besvare det siste forskningsspørsmålet til studiet.

Noe som har vert utfordrende med bruken av rammeverket, er å klassifisere *kognitive krav* for noen av POiM. Selv med hjelp av forhandsdefinerte kriterier for de *kognitive kravene*, skapte programmeringen nye elementer som måtte tas i betraktning ved avgjørelsen. I noen tilfeller ville programmeringsaktiviteten muligens kunne kreve høyere kognitiv aktivitet enn det matematiske ved oppgaven. Avgjørelsen måtte derfor tas gjennom tolking av løsningsprosessen og de kognitive kravenes definisjoner som ivaretar både krav til programmering og det matematiske. Ved å gjennomføre flere undersøkelser på POiM ville det vert mulig å sammenligne krav og programmeringsaktivitet for å definere mer spesifikke kriterier som tilfredsstiller både programmering og matematikk. Med tydeligere rammer og kriterier som tar hensyn til programmering, ville relabiliteten kunne økes i studier som undersøker *kognitive krav* for POiM.

En annen utfordring med studiets rammeverk var å definere hvilket matematisk tema en POiM tilhørte. Selve rammeverket viser til hvilke muligheter jeg har for å identifisere temaet, men ikke hvordan POiM med flere temaer skal vektlegges. Dersom en oppgave kan knyttes opp til for eksempel både likninger og brøk, hva gjør man da når det allerede eksisterer definerte temaer for hver av dem? I tråd med studiets kvalitative preg, som tillater meg å gjøre tolkninger, har jeg i slike situasjoner forsøkt å knytte oppgavens tema opp mot hva som vektlegges mest. Det betyr at jeg måtte vurdere hvilket av temaene er overordnet og hvilke kommer i tillegg, derfor var det mulig å klassifisere oppgavene i en overordnet og ganske bred kategori. Om studien skulle ha fokus på å kartlegge alle nyanser av matematiske temaer berørt i POiM, så skulle det vært slik at en oppgave kunne høre til flere temaer. Dette kan være interessant å gjøre, men da kreves det en metode som bearbeider og presiserer definisjon av temaer. Det blir også et annet utbytte i en slik studie. En konsekvens av dette er at POiM kan i noen tilfeller også være gyldig i et annet tema enn den som er tilordnet oppgaven.

6 Avslutning

Gjennom dette studiet, har det blitt gjennomført en innholdsanalyse av POiM hentet fra læreverkene *Matemagisk 8* og *Campus Matte 8*. Analysemodellen er inspirert av *horisontal* og *vertikal* analyse av Charalambous et al. (2010). I den horisontale analysen ble læreverkenes bakgrunnsinformasjon og struktur undersøkt for å kartlegge i hovedsak hvilke mulige matematiske temaer POiM kunne plasseres under og danne begrunnelser for valgene gjort i den andre analysen. I den vertikale analysen ble POiM undersøkt med intensjon for å verifisere dens matematiske tema, identifisere hvilke *kognitive krav* av Smith og Stein (1998) og *nøkkelbegreper* av Csizmadia et al. (2015) de inneholdt.

Det første forskningsspørsmålet var «Hvilke matematiske temaer er POiM plassert under?». Studiet viser at oppgaver fra *Matemagisk 8* og *Campus Matte 8* for 8. trinn, som var klassifisert som POiM, kunne plasseres under fem matematiske temaer: *brøk og desimaltall*, *algebraiske uttrykk og formler, potenser, kvadratrøtter og regnerekkefølge*, *parenteser og likninger* og *funksjoner*. Fordelingen av POiM i hver av temaene er nokså ulikt, *parenteser og likninger* og *funksjoner* inneholder vesentlig færre antall POiM i forhold til de tre andre temaene.

Neste forskningsspørsmål var «Hvilke kognitive krav stiller formuleringen av POiM til elever?». Studiet viset at læreverkene har POiM med flest lave (72 %) i forhold til høye kognitive krav (28 %). Uavhengig av lave og høye krav, identifiserer POiM som memorering (31 %), prosedyrer uten sammenheng (41 %), prosedyrer med sammenheng (25 %) og gjøre matematikk (3 %). Fordelingen av lave og høye kognitive krav er nokså lik gjennom alle oppgavekategoriene med unntak i *nivå 3*, hvor fordelingen har en overvekt av høye krav.

For å svare på det siste forskningsspørsmålet «Hvilke nøkkelbegrep for algoritmisk tenkning kan man gjenkjenne gjennom arbeid med POiM?», var de seks nøkkelbegrep fra den algoritmiske tenkeren delt opp i to kategorier: de ofte brukte og sjeldent brukte. *Logikk*, *algoritmer* og *mønster* er brukt i 76 – 98 % av alle programmeringsoppgaver og brukes derfor ofte. *Dekomposisjon*, *abstraksjon* og *evaluering* brukes i kun 3 – 26 % av alle programmeringsoppgavene og derfor sjeldent.

Disse resultatene besvarer min problemstilling:

Hvilke kvaliteter har programmeringsoppgaver i matematikk på tvers av to læreverk?

For integrering av programmering i matematikk viser studiet at undervisning om *algebraiske uttrykk og formler, brøk og desimaltall* og *parenteser og likninger* er temaer som kan være gode til å trekke programmering inn i matematikkundervisningen på 8. trinn. Det kan også innføres i *parenteser og likninger* og *funksjoner*, men med færre oppgaver fra de to læreverkene i forhold til de andre temaene. Fordelingen av lave og høye kognitive krav i deres POiM viser at oppgaver kan gis til ulike formål. De lave kognitive kravene vektlegger mer grunnlegendeferdigheter om hvordan man kan bruke programmering i matematikk, mens de høye kognitive kravene fokuserer på å utvikle matematisk forståelse og sammenheng gjennom programmering. Studiets resultater viser at fordelingen av kognitive krav for de tre nevnte matematiske temaene i læreverkene er nokså like og de tilbyr derfor variasjon av POiM. Deres kvaliteter for *algoritmisk tenkning* vektlegger mest problemløsningsaktivitetene *å forstå problemet, lage en plan* og *gjennomføre planen* av Polya (2014). Oppgaveteksten til læreverkenes POiM oppfordrer i mindre grad til *å se tilbake*, det kan derfor være lurt å oppfordre eller tilrettelegge for samtaler eller diskusjon rundt løsninger av POiM for å øke nøkkelbegrepet *evaluering* gjennom undervisningen.

For veien videre kunne det vert interessant å gjennomføre en lik undersøkelse på de andre alternativene av læreverk som er presentert i kapittel 3.1. Ved å gjøre dette og kombinere resultatene fra dette studiet dannes det en større og mer nøyaktig oversikt over kvaliteter som POiM kan ha. Dette kan også gjøres for andre trinn, som for eksempel på læreverkene for 10. trinn. Da kan man se om fordelingen av kvaliteter øker ved progresjon fra 8. til 10. trinn. Resultater fra en slik sammenligning vil også muligens kunne vise om programmering i matematikk er en ressurs som kan bidra til å utvikle elevers ferdigheter i utforsking og problemløsning i andre matematiske temaer.

En annen retning som kunne vert interessant å gå, er å undersøke om mer vektlegging av *evaluering* vil ha en effekt på antallet GM-oppgaver. Dette var et interessant funn for

diskusjon som jeg ikke kunne konkludere på i dette studiet. Ved å designe eller endre POiM til å vektlegge mer *evaluering*, så kunne dette testes og sammenlignes for å se om de får et høyere nivå av *kognitivt krav*.

Litteratur

Aschehoug. (u.å.). *Matematikk*. Aschehoug Skole. Hentet 25.11.2022 fra https://skole.aschehoug.no/ungdomsskole/matematikk

Aunivers. (2021). *Matemagisk 8-10 Digital lærerveiledning til bok*. Aschehoug Undervisning. https://aunivers.no/fagpakker/realfag/matemagisk-8-10#!/aarstrinn8/1317/til-laereren

Bloom, B. S., Engelhart, M. D., Furst, E. J., Hill, W. H. & Krathwohl, D. R. (1956). *Taxonomy of Educational Objectives: The Classification of Educational Goals*. Longmans, Green. https://books.google.no/books?id=hos6AAAAIAAJ

Cappelen Damm. (u.å.). *Alt på ett sted*. Hentet 25.11.2022 fra https://skolen.cdu.no/_/hva-er-skolen-629f475704f7531dafdf0791

Charalambous, C. Y., Delaney, S., Hsu, H.-Y. & Mesa, V. (2010). A Comparative Analysis of the Addition and Subtraction of Fractions in Textbooks from Three Countries. *Mathematical Thinking and Learning*, *12*, 117-151.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C. & Woollard, J. (2015). Computational thinking: A guide for teachers.

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

Hana, G. M. (2014). *Matematiske tenkemåter*. Caspar Forlag AS.

Heimstad, C. A. & Strand, K. (2018). *Kognitive utfordringer i to norske lærebokserier fra ungdomsskolen – en mixed methods studie* [Masteroppgave, Norges Arktiske Universitet]. Munin. https://munin.uit.no/bitstream/handle/10037/13791/thesis.pdf?sequence=2&isAllowed=y

Hofstad, O., Bergljot & Liland, J., Christian. (2021). *En studie av algebraoppgaver i to lærebøker for 8.trinn basert på den nye læreplanen.* [Mastergrad, NORD Universitet]. Nord Open. https://nordopen.nord.no/nord-xmlui/bitstream/handle/11250/2773989/HofstadLiland.pdf?sequence=1&isAllowed=y

Hsieh, H.-F. & Shannon, S. (2005). Three Approaches to Qualitative Content Analysis. *Qualitative health research*, *15*, 1277-1288. https://doi.org/10.1177/1049732305276687

Inkrement. (u.å.). *CampusMatte 8-10*. Hentet 25.11.2022 fra https://campus.inkrement.no/Home/CampusMatte_8_10

Johannessen, A., Christoffersen, L. & Tufte, P. A. (2021). *Introduksjon til samfunnsvitenskapelig metode* (6. utg.). Abstrakt forlag.

Johansen, A.-K. (2020). Programmering vil bli en utfordring for lærere. https://forskning.no/barn-og-ungdom-hogskolen-i-ostfold-matematikk/programmering-vil-bli-en-utfordring-for-laerere/1711838

Johnson, K., Herr, T. & Kysh, J. (2018). *Crossing the river with dogs: Problem solving for college students*. John Wiley & Sons.

Jones, D. L. & Tarr, J. E. (2007). An Examination of the Levels of Cognitive Demand Required by Probability Tasks in Middle Grades Mathematics Textbooks. *Statistics education research journal*, *6*(2), 4-27. https://doi.org/10.52041/serj.v6i2.482

Kidsakoder. (u.å.). *Om LKK*. Hentet 25.11.2022 fra https://www.kidsakoder.no/om-lkk/

Kieran, C. (2013). The False Dichotomy in Mathematics Education Between Conceptual Understanding and Procedural Skills: An Example from Algebra. I K. R. Leatham

(Red.), *Vital Directions for Mathematics Education Research* (s. 153-171). Springer. https://doi.org/10.1007/978-1-4614-6977-3

Kikora. (u.å.). *Om oss*. Hentet 25.11.2022 fra https://kikora.no/om-oss/

Kongsnes, A. L. & Wallace, A. K. (2020). *Matemagisk 8 Lærebok* (K. Kleivdal, Red. 1. Nynorsk. utg.). Aschehoug Undervisning.

Kotsopoulos, Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J. & Yiu, C. (2017). Pedagogical Framework for Computational Thinking. Digital Experiences in Mathematics Education, 3(2), 154–171. https://doi.org/10.1007/s40751-017-0031-2

Kunnskapsdepartementet. (2016). *Fag - Fordypning - Forståelse: En fornyelse av kunnskapsløftet* (Meld. St. nr. 28 (2015-2016)). Kunnskapsdepartementet.

Kunnskapsdepartementet. (2020). *Læreplan i matematikk 1.-10. trinn (MAT01-05)*. Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020. https://www.udir.no/lk20/mat01-05?lang=nob

Opheim, L. G. & Simensen, A. M. (2020). Matematikk - utforsking av mønstre og de store sammenhengene. I S. Bjørshol & R. Nolet (Red.), *Utforsking i alle fag* (s. 101-131). Cappelen damm.

Papert, S. (1980). Mindstorms Children, Computers and Powerful ideas. https://dl.acm.org/doi/pdf/10.5555/1095592

Polya, G. (2014). *How to Solve It: A New Aspect of Mathematical Method* (Expanded Princeton Science Library. utg.). Princeton University Press.

Postholm, M. B., Jacobsen, D. I. & Søbstad, R. (2018). *Forskningsmetode for masterstudenter i lærerutdanningen*. Cappelen Damm akademisk.

Sande, H. (u.å.). Programmering: noen sentrale begrep. *Realfagsløyper*. https://realfagsloyper.no/sites/default/files/2019-11/Programmering%20-%20noen%20sentrale%20begrep_0.pdf

Schoenfeld, A. H. (2014). What Makes for Powerful Classrooms, and How Can We Support Teachers in Creating Them? A Story of Research and Practice, Productively Intertwined. *Educational researcher*. https://hvl.instructure.com/courses/17168/files/folder/Undervisning/Uke%2045?preview=1557453

Schoenfeld, A. H. (2016). Learning to Think Mathematically: Problem Solving, Metacognition, and Sense Making in Mathematics (Reprint). *Journal of education (Boston, Mass.)*, *196*(2), 1-38. https://doi.org/10.1177/002205741619600202

Sevik et al. (2016). Notat nr.2: Programmering i skolen. https://www.udir.no/globalassets/filer/programmering_i_skolen.pdf

Silver, E. A. & Stein, M. K. (1996). The Quasar Project: The «Revolution of the Possible» in Mathematics Instructional Reform in Urban MIddle School. *Urban Education*, 476-521.

Smith, M. S. & Stein, M. K. (1998). Selecting and Creating Mathematical Tasks: From Research to Practise. *Mathematics Teaching in the Middle School*, *3*(5), 344-350.

Statped. (2021). *Programmering*. Statped. https://www.statped.no/laringsressurser/teknologitema/programmering-for-barn-med-saerskilte-behov/programmering/?depth=0#1

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

Thue, B. O., MOldeklev, R.-A. & Røyland, O. T. (2022). *Campus Matte 8*. Inkrement AS. campus.inkrement.no

Torkildsen, S. H. (2017). Matematisk problemløsing. https://www.matematikksenteret.no/sites/default/files/media/filer/MAM/Torkildsen%20Matematisk%20Problemløsing.pdf

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

Utdanningsdirektoratet. (2019b). Hva er kjerneelementer? https://www.udir.no/laring-og-trivsel/lareplanverket/stotte/hva-er-kjerneelementer/

Valenta, A. (2016). Kognitive krav i matematikkoppgaver. *Matematikksenteret: Nasjonalt senter for matematikk i opplæringen*. https://www.matematikksenteret.no/sites/default/files/media/filer/MAM/Valenta%20Kognitive%20krav%20i%20matematikkoppgaver_0.pdf

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35. DOI: 10.1145/1118178.1118215

Wing, J. M. (2010). Computational Thinking: What and Why? https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Woods, D. R. (2000). An Evidence-Based Strategy for Problem Solving. *Journal of engineering education (Washington, D.C.)*, *89*(4), 443-459. https://doi.org/10.1002/j.2168-9830.2000.tb00551.x

Vedlegg

Resultater fra den vertikale analysen

Læreverk	Matemagisk 8
Kapittel	Brøk og desimaltall
Delkapittel	Programmering i Python
Læringsmål	* Utforske og lage en algortime for et program som utfører en regneoperasjon. * Bruke Python og lage et kalkulatorprogram som henter verdier fra brukeren, utfører en regneoperasjon og skriv svaret til skjermen.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	2.86	BD	SKAPE	lo	al		mø			F	PUS
Diskusjon	s. 90	BD	TW	lo		de	mø	ab	ev	F	PUS
Oppgaver	2.90a	BD	FIKLE	lo			mø			F	PUS
Oppgaver	2.90b	BD	FS	lo			mø	ab	ev	F	PUS
Oppgaver	2.90c	BD	FS	lo			mø			F	PUS
Diskusjon	s. 91	BD	FS	lo	al	de	mø	ab		V	PUS
Oppgaver	2.91c	BD	SKAPE	lo	al		mø			S	PMS
Oppgaver	2.92b	BD	SKAPE	lo	al		mø			S	M
Nivå 1	2.95	BD	SKAPE				mø			S	M
Nivå 1	2.96b	BD	FIKLE	lo			mø			S	M
Nivå 1	2.96c	BD	SKAPE	lo	al		mø			F	PUS
Nivå 1	2.97	BD	SKAPE	lo	al	de	mø			S	M
Nivå 1	2.98	BD	SKAPE	lo	al		mø	ab		S	PUS
Nivå 2	2.100	BD	SKAPE	lo	al		mø			S	PUS
Nivå 2	2.101a	BD	SKAPE	lo	al		mø	ab		S	PUS
Nivå 2	2.101b	BD	SKAPE	lo	al		mø	ab		S	PUS

Læreverk	Matemagisk 8
Kapittel	Algebraiske uttrykk og formler
Delkapittel	Programmering med løkker
Læringsmål	* Utforske og lage program i Python som gjenntar ein kodesekvens flere ganger. * Vurdere i hvilken situasjon det er lurt å bruke for-løkker, og når det er lurt å bruke while-løkker.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	3.27b	AUF	FIKLE	lo						S	M
Oppgaver	3.28b	AUF	FIKLE	lo						S	M
Oppgaver	3.29b	AUF	FIKLE	lo						S	M
Oppgaver	3.29c	AUF	FIKLE	lo						S	M
Oppgaver	3.30b	AUF	FIKLE	lo						S	M
Oppgaver	3.30c	AUF	FIKLE	lo						S	M
Oppgaver	3.31b	AUF	FIKLE	lo						S	M
Oppgaver	3.31c	AUF	FIKLE	lo						S	M
Diskusjon	s. 112 a	AUF	TW	lo		de	mø	ab		F	PUS
Diskusjon	s. 112 b	AUF	TW	lo		de	mø	ab		F	PUS
Oppgaver	3.32b	AUF	FIKLE	lo						F	PUS
Oppgaver	3.33b	AUF	FIKLE	lo						F	PUS
Oppgaver	3.33c	AUF	FIKLE	lo						F	PUS
Diskusjon	s.112a	AUF	TW	lo		de	mø			F	PUS
Diskusjon	s.112b	AUF	TW	lo		de	mø			F	PUS
Diskusjon	s.112c	AUF	TW	lo		de	mø			F	PUS
Diskusjon	s.112d	AUF	TW	lo		de	mø			F	PUS
Diskusjon	s.114	AUF	TW				mø	ab		F	PUS
Oppgaver	3.34b	AUF	SKAPE	lo		de	mø			F	PUS
Nivå 1	3.35'	AUF	SKAPE	lo	al		mø			S	M
Nivå 1	3.36'	AUF	SKAPE	lo	al		mø			S	M
Nivå 2	3.37'	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 2	3.38a	AUF	SKAPE	lo	al		mø			F	PMS
Nivå 2	3.38b	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	3.38c	AUF	FIKLE	lo	al	de	mø			S	PUS
Nivå 2	3.38d	AUF	FIKLE	lo	al		mø			S	PUS

Læreverk	Matemagisk 8
Kapittel	Algebraiske uttrykk og formler
Delkapittel	Figurtall
Læringsmål	* Bruke programmering og regneark til å beskrive og skrive ut tallmønster.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	3.41g	AUF	SKAPE	lo	al		mø			S	PUS
Oppgaver	3.42e	AUF	SKAPE	lo	al		mø			S	M
Oppgaver	3.43e	AUF	SKAPE	lo	al		mø			S	PUS
Oppgaver	3.44a	AUF	SKAPE	lo	al		mø			S	PMS
Oppgaver	3.44b	AUF	SKAPE	lo	al		mø			S	M
Oppgaver	3.44c	AUF	SKAPE	lo	al		mø			S	PUS
Oppgaver	3.44d	AUF	SKAPE	lo	al		mø			S	M
Oppgaver	3.44e	AUF	SKAPE	lo	al		mø			S	PMS
Oppgaver	3.44f	AUF	SKAPE	lo	al		mø			S	M
Oppgaver	3.44g	AUF	SKAPE	lo	al		mø			S	PUS
Oppgaver	3.44h	AUF	SKAPE	lo	al		mø			S	M
Nivå 1	3.45d	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	3.47d	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	3.47f	AUF	SKAPE	lo	al	de	mø			S	PMS
Nivå 2	3.47g	AUF	SKAPE	lo	al	de	mø			S	M
Nivå 2	3.47h	AUF	SKAPE	lo	al	de	mø			S	PUS
Nivå 2	3.49'	AUF	SKAPE	lo	al		mø	ab		S	PMS

Læreverk	Matemagisk 8
Kapittel	Potenser, kvadratrøtter og regnerekkefølge
Delkapittel	Potenser og kvadratrøtter
Læringsmål	* Bruke programmering og regneark til å skildre og skrive ut tallmønster.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	4.8e	PKR	SKAPE	lo	al		mø			S	PUS
Oppgaver	Riskorn (3)e	PKR	SKAPE	lo	al		mø			S	PUS
Oppgaver	Riskorn (4)c	PKR	FIKLE	lo		de				F	PMS
Oppgaver	Riskorn (5)a	PKR	SKAPE	lo	al		mø			S	PMS

Læreverk	Matemagisk 8
Kapittel	Parenteser og likninger
Delkapittel	Å løse likninger med programmering
Læringsmål	* Lage, forbedre og programmere algoritmer som løser ligninger.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	6.30a	PL	FIKLE	lo	al					S	M
Oppgaver	6.30e	PL	FIKLE	lo	al		mø	ab		S	PUS
Diskusjon	s. 194 (1)	PL	TW	lo		de				F	PUS
Oppgaver	6.31	PL	SKAPE	lo	al		mø			S	PUS
Diskusjon	s. 194 (2)	PL	TW	lo	al	de	mø	ab	ev	F	GM
Nivå 1	6.33	PL	SKAPE	lo	al		mø			S	M
Nivå 2	6.35b	PL	SKAPE	lo	al		mø			S	PUS
Nivå 3	2c	PL	FIKLE	lo	al	de	mø			S	PMS
Nivå 3	3	PL	FIKLE	lo	al	de	mø		ev	S	PUS

Læreverk	Matemagisk 8
Kapittel	Hva er en funksjon?
Delkapittel	Funksjonsmaskiner
Læringsmål	* Lage et Python-program som fungerer som en funksjonsmaskin.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	7.7a	F	SKAPE	lo			mø			S	PUS
Oppgaver	7.7b	F	FIKLE	lo			mø			S	M
Oppgaver	7.8b	F	SKAPE	lo	al	de	mø			S	M
Oppgaver	7.8c	F	FIKLE	lo	al	de	mø			F	PUS
Oppgaver	7.8d	F	SKAPE	lo	al		mø			S	PUS
Oppgaver	Koder (4)d	F	SKAPE	lo	al		mø			F	PUS
Nivå 1	7.17	F	SKAPE	lo			mø			S	M
Nivå 2	7.23	F	SKAPE	lo	al		mø	ab		S	PMS
Nivå 2	7.24a	F	SKAPE	lo	al		mø			S	PMS
Nivå 2	7.24b	F	SKAPE	lo	al		mø			S	PUS

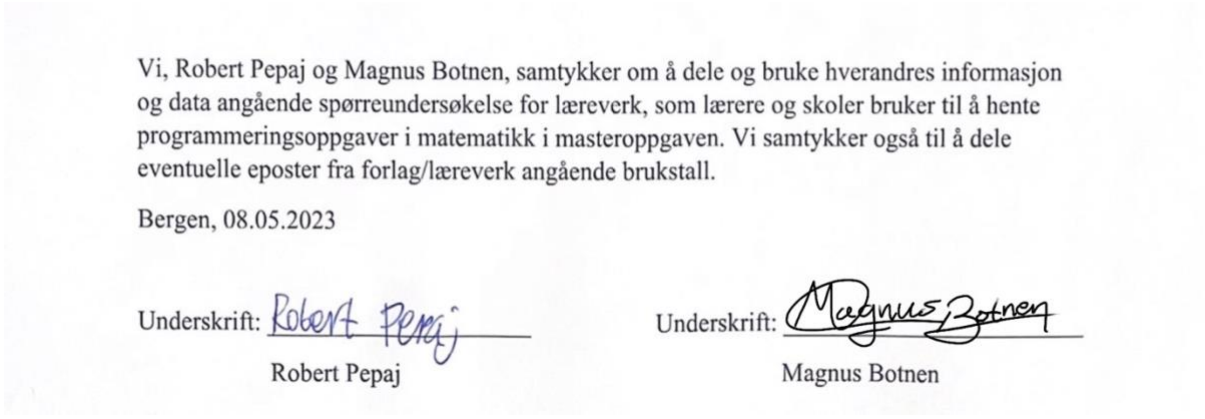
Læreverk	Campus Matte 8
Kapittel	Programmering
Delkapittel	Kalkulatorer
Læringsmål	* Jeg kan lese og forstå programmer som utfører matematisk beregninger. * Jeg kan skrive programmer som utfører matematiske beregninger. * Jeg kan vurdere om programmer som utfører matematiske beregninger fungerer feilfritt.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	Kontroll 1	AUF	SKAPE	lo	al		mø			S	M
Oppgaver	Kontroll 2	AUF	SKAPE	lo	al		mø			S	M
Oppgaver	Kontroll 3	AUF	SKAPE	lo	al		mø			S	PMS
Diskusjon	(1)	AUF	FIKLE	lo		de	mø	ab		S	PMS
Diskusjon	(2)	AUF	FIKLE	lo		de	mø	ab		S	PMS
Nivå 1	1b	AUF	SKAPE	lo	al		mø			S	M
Nivå 1	1c	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 1	1d	AUF	SKAPE	lo	al		mø			S	M
Nivå 1	1e	AUF	SKAPE	lo	al		mø			S	M
Nivå 1	2b	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 1	2c	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 1	2d	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 1	2e	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 2	3a	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 2	3b	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	3c	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	3d	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	4a	AUF	SKAPE	lo	al					S	PMS
Nivå 2	4b	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 2	4c	AUF	SKAPE	lo	al		mø			S	PUS
Nivå 3	5a	AUF	SKAPE	lo	al	de	mø			S	PMS
Nivå 3	5b	AUF	SKAPE	lo	al	de	mø			S	PMS
Nivå 3	5c	AUF	SKAPE	lo	al		mø			S	PMS
Nivå 3	6a	BD	SKAPE	lo	al		mø			S	PMS
Nivå 3	6b	BD	SKAPE	lo	al	de	mø			S	GM
Nivå 3	6c	BD	SKAPE	lo	al	de	mø			S	GM

Læreverk	Campus Matte 8
Kapittel	Programmering
Delkapittel	Problemløsning
Læringsmål	* Jeg kan lese og forstå et program som benytter "gjett og sjekk" metoden. * Jeg kan lage dataprogrammer som løser problemer ved hjelp av "gjett og sjekk" metoden. * Jeg kan forbedre mine egne algoritmer slik at programmene mine finner løsningen raskere.

Kategori	Oppgave nr	Tema	Arbeidsmåter	Logikk	Algoritmer	Dekomposisjon	Mønster	Abstraksjon	Evaluering	Type svar	Kognitive krav
Oppgaver	Kontroll 2	PL	SKAPE	lo	al		mø			S	M
Oppgaver	Kontroll 4	PL	SKAPE	lo	al	de	mø	ab		S	M
Diskusjon	(2)	PL	FIKLE	lo	al		mø			S	PMS
Nivå 1	1b	PL	SKAPE	lo	al		mø			S	M
Nivå 1	1c	PL	SKAPE	lo	al		mø			S	M
Nivå 1	1d	PL	SKAPE	lo	al		mø			S	PMS
Nivå 1	1e	PL	SKAPE	lo	al		mø			S	PMS
Nivå 1	1f	PL	SKAPE	lo	al		mø			S	M
Nivå 1	2a	PL	SKAPE	lo	al	de	mø			S	PMS
Nivå 1	2b	PL	SKAPE	lo	al	de	mø			S	PUS
Nivå 1	2c	PL	SKAPE	lo	al	de	mø	ab		S	M
Nivå 1	2d	PL	SKAPE	lo	al	de	mø	ab		S	PUS
Nivå 2	3a	PL	SKAPE	lo	al		mø			S	M
Nivå 2	3b	PL	SKAPE	lo	al		mø			S	PUS
Nivå 2	3c	PL	SKAPE	lo	al		mø			S	M
Nivå 2	4a	PL	SKAPE	lo	al		mø			S	PUS
Nivå 2	4b	PL	SKAPE	lo	al		mø			S	M
Nivå 2	4c	PL	SKAPE	lo	al		mø	ab		S	M
Nivå 2	4d	PL	SKAPE	lo	al		mø	ab		S	PMS
Nivå 3	5a	PL	SKAPE	lo	al		mø			S	M
Nivå 3	5b	PL	SKAPE	lo	al	de	mø	ab		S	GM
Nivå 3	6a	BD	SKAPE	lo	al		mø	ab		S	PMS
Nivå 3	6b	BD	SKAPE	lo	al	de	mø	ab		S	PMS

Samtykke mellom meg og Robert Pepaj angående innsamling av data til hvilke læreverk som brukes til å hente programmeringsoppgaver i matematikk



Vi, Robert Pepaj og Magnus Botnen, samtykker om å dele og bruke hverandres informasjon og data angående spørreundersøkelse for læreverk, som lærere og skoler bruker til å hente programmeringsoppgaver i matematikk i masteroppgaven. Vi samtykker også til å dele eventuelle eposter fra forlag/læreverk angående brukstall.

Bergen, 08.05.2023

Underskrift: Robert Pepaj

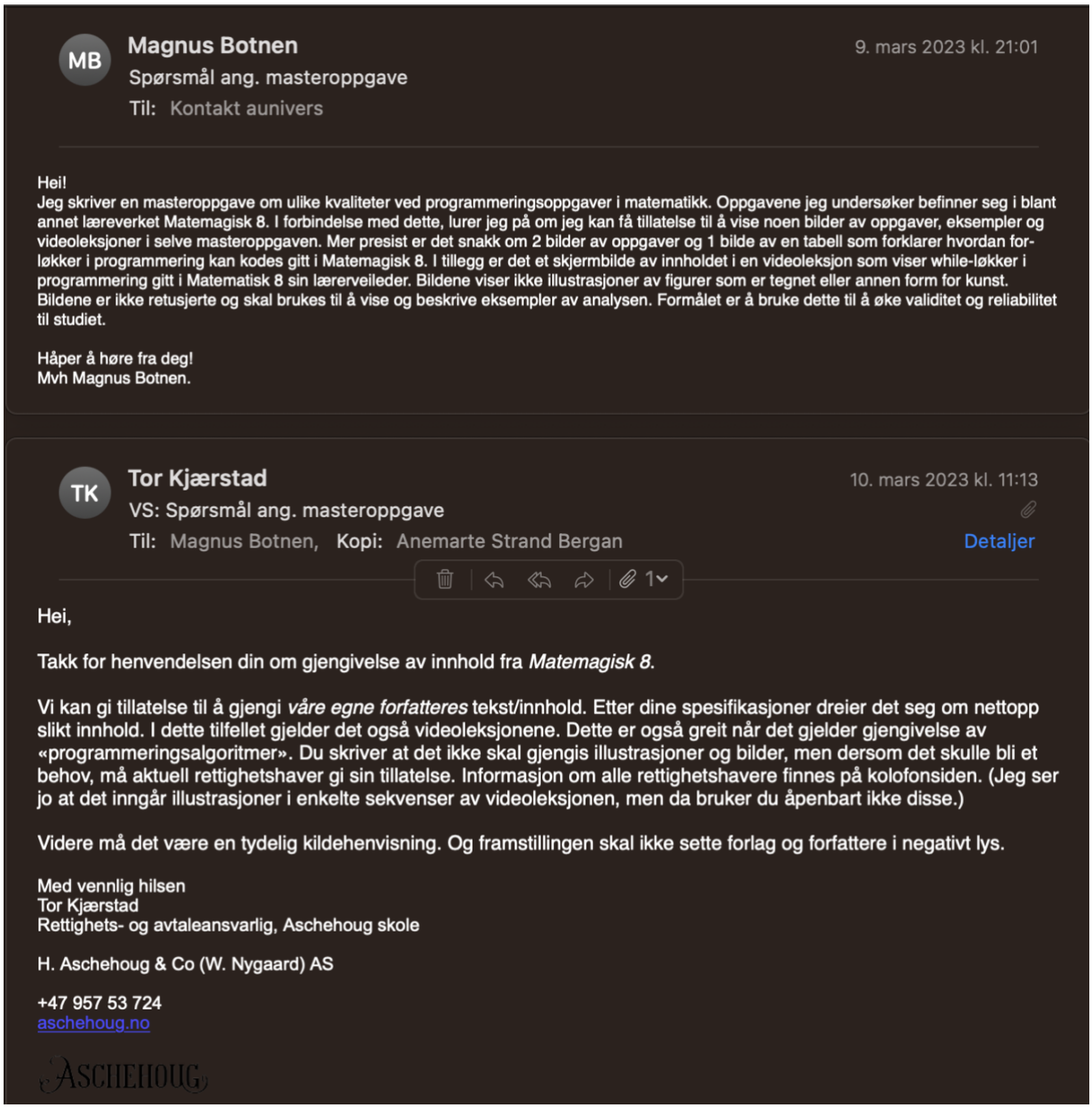
Robert Pepaj

Underskrift: Magnus Botnen

Magnus Botnen

E-poster til forlag og skoler

Avtale for bruk av bilder og skjermbilder fra læreverkene



Magnus Botnen 9. mars 2023 kl. 21:01
Spørsmål ang. masteroppgave
Til: Kontakt aunivers

Hei!
Jeg skriver en masteroppgave om ulike kvaliteter ved programmeringsoppgaver i matematikk. Oppgavene jeg undersøker befinner seg i blant annet læreverket Matemagisk 8. I forbindelse med dette, lurer jeg på om jeg kan få tillatelse til å vise noen bilder av oppgaver, eksempler og videoleksjoner i selve masteroppgaven. Mer presist er det snakk om 2 bilder av oppgaver og 1 bilde av en tabell som forklarer hvordan for-løkker i programmering kan kodes gitt i Matemagisk 8. I tillegg er det et skjermbilde av innholdet i en videoleksjon som viser while-løkker i programmering gitt i Matematisk 8 sin lærerveileder. Bildene viser ikke illustrasjoner av figurer som er tegnet eller annen form for kunst. Bildene er ikke retusjerte og skal brukes til å vise og beskrive eksempler av analysen. Formålet er å bruke dette til å øke validitet og reliabilitet til studiet.

Håper å høre fra deg!
Mvh Magnus Botnen.

Tor Kjærstad 10. mars 2023 kl. 11:13
VS: Spørsmål ang. masteroppgave
Til: Magnus Botnen, Kopi: Anemarte Strand Bergan Detaljer

Hei,

Takk for henvendelsen din om gjengivelse av innhold fra *Matemagisk 8*.

Vi kan gi tillatelse til å gjengi *våre egne forfatteres* tekst/innhold. Etter dine spesifikasjoner dreier det seg om nettopp slikt innhold. I dette tilfellet gjelder det også videoleksjonene. Dette er også greit når det gjelder gjengivelse av «programmeringsalgoritmer». Du skriver at det ikke skal gjengis illustrasjoner og bilder, men dersom det skulle bli et behov, må aktuell rettighetshaver gi sin tillatelse. Informasjon om alle rettighetshavere finnes på kolofonsiden. (Jeg ser jo at det inngår illustrasjoner i enkelte sekvenser av videoleksjonen, men da bruker du åpenbart ikke disse.)

Videre må det være en tydelig kildehenvisning. Og framstillingen skal ikke sette forlag og forfattere i negativt lys.

Med vennlig hilsen
Tor Kjærstad
Rettighets- og avtaleansvarlig, Aschehoug skole

H. Aschehoug & Co (W. Nygaard) AS

+47 957 53 724
aschehoug.no

Magnus Botnen 9. mars 2023 kl. 20:46
Spørsmål ang. masteroppgaver
Til: Lars Unneberg

Hei!
Det er Magnus Botnen her, en av masterstudentene som har fått tilgang til Campus Matte for å studere kvaliteter ved programmeringsoppgaver i matematikk. Jeg lurer på om jeg kan få tillatelse til å vise noen skjermbilder av oppgaver og videoleksjoner i selve masteroppgaven. Bildene er ikke retusjerte og skal brukes til å vise og beskrive eksempler av analysen. Formålet er å bruke dette til å øke validitet og reliabilitet til studiet.

Håper å høre fra deg!
Mvh Magnus Botnen.

Lars Unneberg 10. mars 2023 kl. 11:00
Sv: Spørsmål ang. masteroppgaver
Til: Magnus Botnen

Hei Magnus,
du kan "sitere" fra Campus, dvs at du kan ta enkeltbilder som eksemplifiserer. Du kan ikke kopiere hele gjennomganger. Min erfaring fra tilsvarende oppgaver er at et utvalg bilder er tilstrekkelig.

Vennlig hilsen
Lars Unneberg
Inkrement

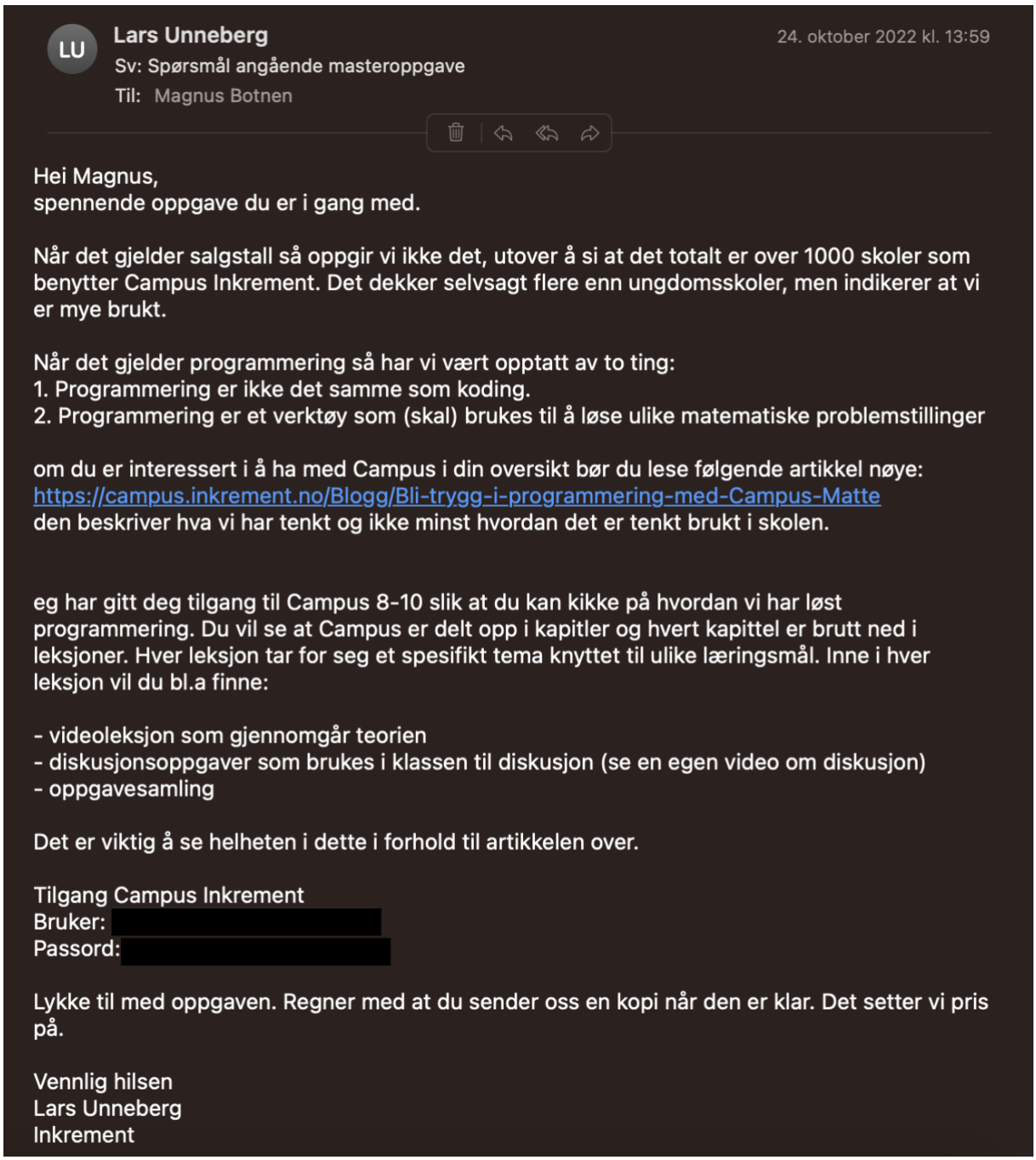
Spørsmål om hvilke læreverk skoler bruker til programmering i matematikk

Hei, eg er 5. års student på grunnskolelærerutdanningen 5.-10. ved HVL Bergen.
Dette studieåret skal eg skrive ei masteroppgåva i matematikkdidaktikk der mitt tema er programmering i matematikk på ungdomstrinnet. Eg skal undersøkje korleis programmeringsoppgåvene i matematikk, legger opp til kjerneelementet *utforsking og problemløysing*. I forbindelse med dette, ynskjer eg å sjå kva lærebokforlag og digitale plattformar som vert nytta på ulike skular i undervisninga, programmering i matematikk, på 8.- 10. trinnet. Denne informasjonen vil hjelpa meg i å velja ut kva for nokre læreverk eg skal undersøkja vidare samt sjå på ulikskapar.

Spørsmål som eg ynskjer svar på:
Kva for eit læreverk og/eller digital plattform nyttar skulen i matematikk og programmering på 8. – til 10. trinn.

Håper å høyra frå deg.
Mvh Magnus Botnen

Spørsmål til forlag om hvor mange skoler som bruker deres læreverk i matematikk



LU **Lars Unneberg** 24. oktober 2022 kl. 13:59
Sv: Spørsmål angående masteroppgave
Til: Magnus Botnen

Hei Magnus,
spennende oppgave du er i gang med.

Når det gjelder salgstall så oppgir vi ikke det, utover å si at det totalt er over 1000 skoler som benytter Campus Inkrement. Det dekker selvsagt flere enn ungdomsskoler, men indikerer at vi er mye brukt.

Når det gjelder programmering så har vi vært opptatt av to ting:
1. Programmering er ikke det samme som koding.
2. Programmering er et verktøy som (skal) brukes til å løse ulike matematiske problemstillinger

om du er interessert i å ha med Campus i din oversikt bør du lese følgende artikkel nøye:
https://campus.inkrement.no/Blogg/Bli-trygg-i-programmering-med-Campus-Matte
den beskriver hva vi har tenkt og ikke minst hvordan det er tenkt brukt i skolen.

eg har gitt deg tilgang til Campus 8-10 slik at du kan kikke på hvordan vi har løst programmering. Du vil se at Campus er delt opp i kapitler og hvert kapittel er brutt ned i leksjoner. Hver leksjon tar for seg et spesifikt tema knyttet til ulike læringsmål. Inne i hver leksjon vil du bl.a finne:

- videoleksjon som gjennomgår teorien
- diskusjonsoppgaver som brukes i klassen til diskusjon (se en egen video om diskusjon)
- oppgavesamling

Det er viktig å se helheten i dette i forhold til artikkelen over.

Tilgang Campus Inkrement
Bruker: [redacted]
Passord: [redacted]

Lykke til med oppgaven. Regner med at du sender oss en kopi når den er klar. Det setter vi pris på.

Vennlig hilsen
Lars Unneberg
Inkrement

Skolen - Cappelen Damm 24. oktober 2022 kl. 16:56
SV: Spørsmål angående masteroppgave CD:05500002
Til: Magnus Botnen

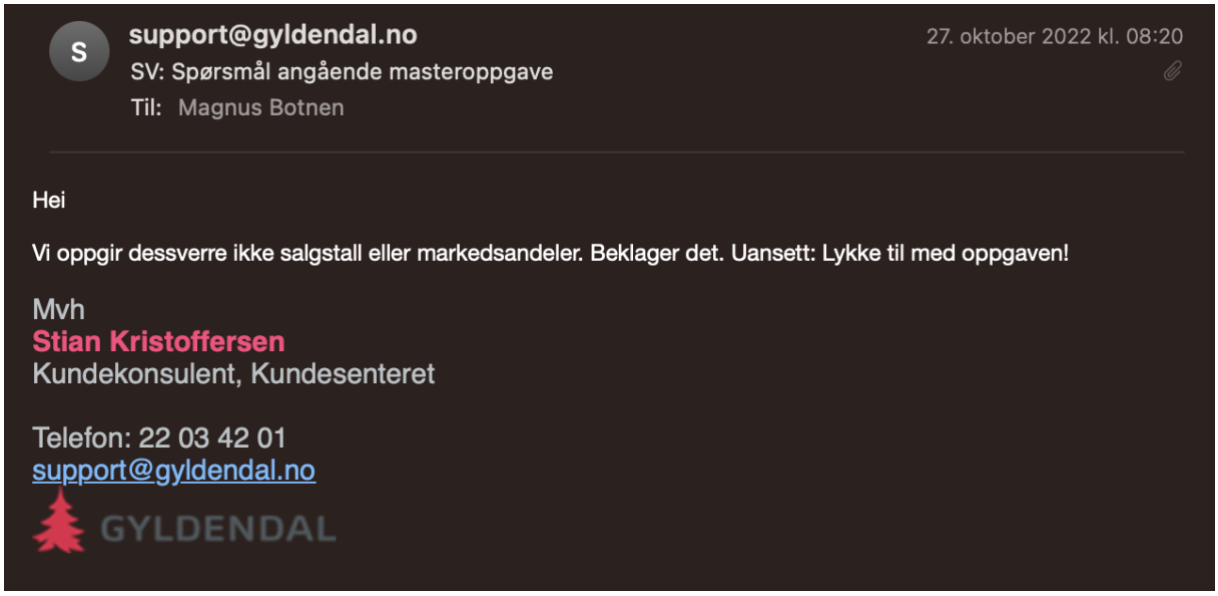
Hei Magnus!

Vi oppgir ikke salgstall på produkter, men vi kan si såpass som at Skolen fra Cappelen Damm er svært mye brukt, og mer konkret kan vi si at annenhver elev bruker Skolen fra Cappelen Damm på utrinn.

Om du ønsker en tilgang må dette gå via en ansatt ved HVL Bergen (veilederm amanuensis, høgskolelektor etc.). Vi har våre egne testbrukere som gir tilgang til Skolen fra CD, og disse må vi verne veldig godt om. Dersom du ønsker en tilgang må den være tidsbegrenset og aldri deles med andre.

Om du kan sende en e-post til meg der du også kobler på en ansatt fra HVL Bergen, og der du skriver hvor lenge du kal ha tilgangen, så kan jeg ordne det. da må det også bekreftes at du kun skal bruke den selv i forbindelse med din masteroppgave. Send e-posten til adressen min som står i avsenderfeltet :-)

--
Med vennlig hilsen
Dag Petter Breivik
Cappelen Damm AS
Telefon: +47 901 37 811
Sentralbord: +47 21 61 65 00
E-post: dag.petter.breivik@cappelendamm.no
www.cdu.no



support@gyldendal.no 27. oktober 2022 kl. 08:20
SV: Spørsmål angående masteroppgave
Til: Magnus Botnen

Hei

Vi oppgir dessverre ikke salgstall eller markedsandeler. Beklager det. Uansett: Lykke til med oppgaven!

Mvh
Stian Kristoffersen
Kundekonsulent, Kundesenteret

Telefon: 22 03 42 01
support@gyldendal.no
GYLDENDAL

support@kikora.no 21. november 2022 kl. 14:42

RE: Spørsmål angående masteroppgave [ref:_00Db0KFbP._5005p2oomb9:ref]

Til: magnusbot@outlook.com

Hei Magnus!

Beklager sent svar - ansvarlig for denne henvendelsen gikk ut i sykemelding, og vi fanget den ikke opp før nå. Beklager det!

Det er omtrent 370 ungdomsskoler i Norge som har Kikora som læremiddel.

Mvh Kikora support

Kontakt aunivers <kontakt@aunivers.no> fre. 11. nov. 2022 på 12:53

Til: Robert Pepaj

Hei,
vi oppgir ikke markedsandeler, men kan bekrefte at Matemagisk 8-10 er mye brukt og har en betydelig markedsandel.
--
Prefer fewer emails from me? Click here

> Vis opprinnelig melding