STUDENT: **SARA-ELISE SØMME**
VEILEDER: GAUTE HOVTUN

Hvilke utfordringer og suksessfaktorer opplever norske lærere med implementering av programmering i matematikk?

Which challenges and success factors do Norwegian teachers experience with the implementation of programming in mathematics?

Masteroppgave i matematikk

År: 2023/2024

Grunnskolelærerutdanning for trinn 5-10

Institutt for grunnskolelærerutdanning, idrett og spesialpedagogikk

Fakultet for utdanningsvitenskap og humaniora

Universitetet
i Stavanger

Antall ord: 22 685
Antall vedlegg/annet: 5
EMNEORD: Programmering, Matematikk, Utfordringer, Suksessfaktorer, Implementering

Sammendrag

Det er flere som har gjennomført studier i ulike land om hvilke utfordringer og suksessfaktorer lærere opplever med implementering av programmering. (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). Igjennom denne studien er det forsket på «*Hvilke utfordringer og suksessfaktorer opplever norske lærere med implementering av programmering i matematikk*». Via intervju og modellen til Vinnervik (2022) for indre og ytre utfordringer har en rekke utfordringer og suksessfaktorer blitt identifisert. Av alle disse utfordringene og suksessfaktorene var det noen som skilte seg mer ut en andre. Lærerne viser å ha en utfordring med å forstå hva de digitale ferdigheter innebærer, og hva programmeringens plass i læreplanen er. De løfter frem utfordringene med mangel på tid til å hjelpe alle elevene, samt tidspresset med et fullstappet pensum. Selv om læreren implementerer programmering som en eget emne eller som en del av de andre emnene, synes fortsatt lærerne at det er tidspress i matematikk. Av suksessfaktorer var det kategorien *Undervisningstilnærming (I4)* fra modellen til Vinnervik (2022) som skilte seg ut. Denne kategorien var den eneste som hadde flere antall ytringer for suksessfaktorer enn antall ytringer for utfordringer. Blant de var suksessfaktorene kopiering og justering, læreverket Campus Inkrement, Spill-basert undervisning, samarbeidslæring. Ifølge lærerne løser samarbeidslæring problemer som tidsmangel, utstyrsmangel og generelt utfordringen med å strekke over alle elevene som trenger hjelp. Av forskningsbasert undervisningsmetoder er det kun Spill-basert læring som nevnes eksplisitt. Ingen av lærere løftet frem forskningsbaserte undervisningsmodeller for programmering i matematikk. Noe som strider imot hypotesen til denne studien.

Forord

Det har vært en læringsrik reise å skrive denne masteren. Da vi ikke har hatt mye programmering i vår grunnopplæring eller som en del av lærerutdanningen, tok jeg dette som en utfordring til å bli mer kjent med programmering i matematikk. Jeg skal jo tross alt stå i de samme skoene som alle disse lærerne som har stilt med egne utfordringer og suksessfaktorer til programmering i matematikk. Lite visste jeg om programmeringens historie og hvorfor elevene skal lære programmering annet enn at det er mye teknologi i verden. Jeg føler at denne masteren har gjort meg mer rustet og gitt meg et bedre ståsted i møte med programmering i matematikk, og ikke minst gitt meg motivasjon og en positiv innstilling.

Denne masteren er et resultat av flere personer som har støttet meg på denne reisen. I sammenheng med dette vil jeg takke min veileder Gaute Hovtun som har vist positivitet og gitt god støtte under hele masterprosjektet. Jeg vil takke alle informantene som sa seg villig til å stille til intervju.

Helt til slutt vil jeg takke hunden min Jeremy for alle timene han har stirret på meg mens jeg skrev denne masteren.

Sara-Elise Sømme

25.05.2024 Ålgård

Innhold

1. Introduksjon 6

1.1 Bakgrunn for valgt tema 6

1.2 Studiens hensikt, formål og problemstilling 6

1.3 Tidligere forskning 7

1.4 Oppgavens oppbygning 8

2. Teori 10

2.1 Programmering 10

2.1.1 Datamaskinens historie 10

2.1.2 Hva er programmering 11

2.1.3 Hvordan ser Programmering ut i skolen? 13

2.2 Hvorfor programmering i skoler 14

2.3 Programmering i læreplanen 15

2.3.1 Digitale ferdigheter og programmering 15

2.3.2 Problemløsing, Algoritmisk tenking og programmering 19

2.4 Programmering og Matematikk 21

2.4.1 Programmeringens plass i matematikk 21

2.4.2 Lærerrollen og samarbeid 23

2.5 Forskningsbasert undervisningsmetoder og modeller 24

2.6 Utfordringer og suksessfaktorer 26

2.6.1 Utfordringer 26

2.6.2 Suksessfaktorer 27

2.7 Analytisk rammeverk 29

3. Metode 32

3.1 Forskningsdesign 32

3.1.1 Kvalitativt design VS Kvalitativt design 32

3.1.2 Informanter 35

3.2 Dataanalyse 37

3.2.1 Del I – Teoridreven Innholdsanalyse, trinn 1 37

3.2.2 Del II - Teoridreven innholdsanalyse, trinn 2 39

3.2.3 Del III - Summativ innholdsanalyse 42

3.3 Forskningsetiske betraktninger 42

3.3.1 Etiske overveielser 42

3.3.2 Validitet og reliabilitet 43

4. Resultater 47
4.1 Indre utfordringer 47
4.1.1 Profesjonell kunnskap og forståelse 47
4.1.2 Profesjonell tilstrekkelighet 48
4.1.3 Profesjonell verdier og holdninger 50
4.1.4 Undervisningstilnærming 50
4.1.5 Eierskap 51
4.2 ytre utfordringer 52
4.2.1 Ressurser 52
4.2.2 Tidshåndtering 53
4.2.3 Praktisk side med implementering 54
4.2.4 Elevevaluering 55
4.2.5 Profesjonell utvikling og støtte 55
4.3 Indre suksessfaktorer 56
4.3.1 Profesjonell kunnskap og forståelse 56
4.3.2 Profesjonell tilstrekkelighet 56
4.3.3 Profesjonell verdier og holdninger 58
4.3.4 Undervisningstilnærming 58
4.3.5 Eierskap 60
4.4 Ytre suksessfaktorer 61
4.4.1 Ressurser 61
4.4.2 Tidshåndtering 62
4.4.3 Praktisk side med implementering 62
4.4.4 Elevevaluering 62
4.4.5 Profesjonell utvikling og støtte 63
5. Drøfting 64
5.1 Læreres forståelse av programmeringens plass i matematikken 64
5.2 Tidspress og implementeringsvalg 65
5.3 Undervisningsmetoder 67
5.4 Forskningsbasert undervisningsmodeller 70
6. Konklusjon og videre forskning 72
6.1 Konklusjon 72
6.2 Studiens svakheter og videre forskning 72
Referanser 74
Figurliste 78

Vedlegg **79**
Vedlegg 1 – Vinnervik (2022) 79
Vedlegg 2 – Finger & Houguet (2009) 80
Vedlegg 3 – Søknadsgodkjenning fra sikt 81
Vedlegg 4 – Informasjonsskriv og samtykkeskjema 82
Vedlegg 5 - Intervjuguide 85

1. Introduksjon

1.1 Bakgrunn for valgt tema

I løpet av 2020 kom den nye læreplanen LK20. Da den forrige læreplanen, LK06, har vært skolens grunnmur siden 2006, var det på tide med et nytt læreplanverk. Læreplanverket må kunne representere nåtiden og kunne speile samfunnet. Med den nye læreplanen programmering en ny sentral del og kan finnes i de grunnleggende ferdighetene og i fagene matematikk, naturfag og musikk. Utfordringen med å implementere programmering inn i skolen, er at det er svært få lærere som har hatt dette i egen utdanning. Johansen & Østfold (2020) skriver at programmering vil bli en utfordring, blant annet på grunn av mangel på programmeringskunnskap blant lærere. Lærere er usikker på hvordan de skal implementere programmering i matematikken, og på det tidspunktet fantes det ikke mye forskning om hvordan man skulle gjøre dette. Ertmer & Ottenbreit-Leftwich (2010, s. 277) påpeker at selv om lærere støtter ideen om denne nye definisjonen av undervisning, trenger de fortsatt konkete eksempler på hvordan dette ser ut i praksis. Både Ertmer & Ottenbreit-Leftwich (2010) og Finger & Houguet (2009) påpeker at lærere trenger en bedre forståelse av teknologi. Selv om flere elever har god kunnskap selv, mangler de kunnskapen til å tilrettelegge for elevsentrert læring.

Finger & Houguet (2009), Jones et al. (2004), Sentance & Csizmadia (2017) og Vinnervik (2022) har alle gjennomført forskning om hvilke utfordringer lærere har møtt på med implementering av teknologi /programmering og nevnt noen metoder som er benyttet for å møte disse utfordringene. Forskningen har funnet sted i deres respektive hjemland: Australia, New Zealand, England og Sverige. Det hadde da vært interessant å se på utfordringer i Norge, spesielt siden programmering ble en sentraldel av læreplanen i 2020. Da implementeringsprosessen i Norge har vart i 4 år, kan det også være interessant å se på suksessfaktorer.

1.2 Studiens hensikt, formål og problemstilling

Formålet med denne studien er å kartlegge utfordringer som matematikk lærere i Norge har støtt på rundt implementering av programmering i matematikk. I tillegg til å identifisere utfordringene vil det også bli undersøkt om noen har utviklet gode metoder eller løsningsstrategier som har bidratt til å overvinne noen av disse utfordringene. Hensikten med studien er å kunne tilby tips og metoder som kan være til hjelp for lærere som fortsatt sliter med implementering av programmering og ikke har fått effektive måter å håndtere disse utfordringene på.

Dette prosjektet har som mål å finne svar på problemstillingen:

«Hvilke utfordringer og suksessfaktorer opplever norske lærere med implementering av programmering i matematikk?».

Hypotesen til denne studien er at det er forventet at lærere har støtet på utfordringer i implementeringsprosessen, som for eksempel mangel på egen kunnskap og ferdigheter. Det forventes å få en innsikt til hvordan de har skaffet seg kunnskapen og ferdighetene samt hvordan de oppnår tilstrekkelig kompetanse til å undervise i faget på en effektiv måte. Når det gjelder strategier, er det ikke forventet at alle lærere har funnet løsninger på alle sine utfordringer. Noen strategier som kan bli benyttet, inkluderer bruk av eksterne ressurser, deltakelse i kurs og bruk av ulike forskningsbaserte undervisningsmetoder og modeller.

1.3 Tidligere forskning

Flere forskere har undersøkt utfordringer knyttet til programmering og teknologi i skolen (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). Finger & Houguet (2009) har studert indre og ytre utfordringer ved implementering av teknologi i Australia. De identifiserte flere utfordringer som blant annet lærernes kunnskap og forståelse av teknologi, ressursmangel og profesjonell utvikling ble fremhevet som gjentagende temaer. Sentance & Csizmadia, (2017) rapporterer de samme utfordringene i England som Finger & Houguet (2009), men deres forskning legger også vekt på utfordringer knyttet til differensiering, elevers forståelse og tekniske problemer. Vinnervik (2022) identifiserer flere av de samme utfordringene som Finger & Houguet (2009) og Sentance & Csizmadia (2017) i Sverige, inkludert mangel på tid, kunnskap og profesjonell tilstrekkelighet. En utfordring som skilte seg ut fra Finger & Houguet (2009) og Sentance & Csizmadia (2017) var at på grunn av mangel på undervisningsmateriale fra utdanningsforskning, vendte lærerne seg til generiske kodingsressurser. Dette resulterte i at undervisningen ble planlagt basert på hva verktøyene kunne tilby, ikke hva læreplanen foreskrev. Jones et al. (2004) sin forskning i New Zealand nevner også teknologisk kunnskap og ressurser som en utfordring, parallelt med funnene til Finger & Houguet (2009), Sentance & Csizmadia (2017), Vinnervik (2022), men de påpeker også utfordringer knyttet til en fullpakket læreplan og læreplanforståelse. Gjøvik & Torkildsen (2019) fremhever utfordringen knyttet til hva som vil skje med elever som ikke behersker

programmeringsferdighetene. Når det kommer til suksessfaktorene, dukker det opp flere undervisningsmetodene. Sentance & Csizmadia (2017) nevner den fysisk programmeringsmetoden Unplugged, Dolonen et al. (2019) presenterer FACT-metoden, PRIMM- modellen, scratch og fysisk programmering. Både Sentence & Csizmadia (2017), Gjøvik & Torkildsen (2019) og Dolonen et al. (2019) anbefaler at programmeringsundervisning består av fysisk programmering som baserer seg delvis på bruk av teknologi.

Identifisering av utfordringer kan gi lærere og forskere muligheten til å utvikle strategier for å overvinne dem (Finger & Houguet, 2009). Som tidligere nevnt har det blitt forsket på utfordringer knyttet til implementering av programmering i skolen i flere land. Australia, New Zealand og England har innført teknologi som et egent fag (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017). Sverige har implementert programmering i matematikkfaget og et egent teknologifag (Vinnervik, 2022), mens Norge har implementert programmering inn i matematikk, naturfag og musikk. Med andre ord er det betydelig forskning i ulike settinger som kan påvirke implementeringsprosessen og de ulike utfordringene som medfølger. Derfor er det nødvendig med økt kunnskap om hvilke utfordringer vi står overfor i Norge og hvordan norske lærere har overvunnet noen av disse?

Norge har sin egen implementeringsprosess, noe som kan resultere i unike utfordringer som skiller seg fra annen forskning. For å støtte lærer i undervisning og implementeringsprosessen deres, er det relevant med forskning basert på den norske skolegangen og deres implementeringsutfordringer. For at lærere skal få strategier og metoder som kan hjelpe dem i programmeringsundervisningen knyttet til matematikk, er det enda bedre hvis disse strategiene og metodene er forankret i en norsk kontekst. Dette i stedet for å kun basere seg på erfaringer fra andre land med egne kulturer og læreplaner.

1.4 Oppgavens oppbygning

Videre i denne oppgaven vil relevant teori presenteres. Denne vil løfter fokus datamaskinens historie, hva programmering er og hvordan programmering ser ut i skolen. Videre vil det bli gjennomgått programmeringens plass i skolen. Programmeringens plass læreplanen går i dybden på sammenhengen mellom programmering, digitale ferdigheter, algoritmisk tenkning og problemløsning. Videre skal det argumenteres for hvorfor programmering er en del av matematikken. Videre vil noen forskningsbaserte undervisningsmetoder og modeller bli presentert. Avslutningsvis for teoridelen vil utfordring og suksessfaktor defineres for å så

ende med presentering av det analytiske rammeverket til Vinnervik (2022). Metodedelen omhandler diskusjonene som konkluderer metodevalget for denne studien. I denne delen vil kvalitativ og kvantitativstudie definert og drøftes for å så konkludere på en kvalitativforskningsstudie rettet mot det fenomenologiske. Deretter skal ulike kvalitative forskningsmetoder definert og drøftes før det konkluderes i et semistrukturert lærerintervju. Etterfulgt vil Informantene bli presentert med en kort beskrivelse om deres bakgrunn. Informantene vil består av seks lærere på fire forskjellige skoler. To individuelle intervju og to gruppe intervju. Avsluttende til metodedelen vil inneholde en gjennomgang av de ulike analysemetodene. Dataene fra intervjueneia skal både analyseres via en teoridrevet innholdsanalyse hvor dataene blir plassert inn i Vinnerviks (2022) modell for indre og ytre utfordringer. Etterfulgt av en delvis summativ innholdsanalyse. Resultatdelen presenterer så de viktigste utfordringene og suksessfaktorene. Etterfulgt kommer en drøftingsdel hvor funnene fra resultatdelen bli drøftet opp mot relevant teori før det avsluttende vil komme en konklusjon og forslag til videre forskning.

2. Teori

2.1 Programmering

Da kjernen av denne studien er sentrert rundt programmering, må det bli rammet inn hva programmering er. For å kunne forstå dette trengs også kunnskapen til hvordan programmering og datamaskinen henger sammen. Det vil derfor bli tatt en kort og relevant gjennomgang av datamaskinens historie og oppbygning, samt virkemåte. Da datamaskinens historie er omfattende og detaljert, vil kun de mer overordnede punktene bli fremhevet. Det samme gjelder datamaskinens oppbygning. Kun de viktigste detaljene som har en sammenheng mellom datamaskinen og programmering vil bli løftet frem.

2.1.1 Datamaskinens historie

Skal en bygge en datamaskin må en først fokusere på å bygge en maskin som kan addere to tall sammen. En slik maskin kan da etterhvert utføre alle regneartene og kan utvikles til å bli den datamaskinen en har i dag (Petzold, 2000, s.135). I følge Copeland (2000) er det flere forskere som har æren for at datamaskinen finnes den dag i dag. Charles Babbage presenterte ideen om en mekanisk datamaskin som kunne utføre avanserte utregninger av ulike algoritmer. En såkalt kalkulator. Alan Turing oppfant prinsippene bak den moderne datamaskinen allerede i 1936. Hans tanke var å skape en maskin som hadde uendelig minne, muligheten til å lese av dette minnet og skrive ned det den finner ut. I 1941 kom den første funksjonelle elektromekaniske datamaskinen med navnet Zuse. I forhold til den mekaniske, besto hovedkomponentene i denne datamaskinen av brytere og var en program-kontrollert maskin. Det vil si at det gikk an å programmere den til å utføre ulike handlinger. For å programmere denne maskinen måtte brukeren fysisk flytte plugger, altså omkoble kablene. Alan Turing og Gordon Welchman bygget egen elektromekanisk datamaskin, «bombes», som kunne knekke enigma kodene tyskerne sendte ut over radio ved andre verdenskrig. Etter hvert så de at maskinen knakk kodene saktere enn ønsket og de trengte en maskin med mer fart. 1950 tallet kjørte pilot modell av Turings automatic computer engine (ACE) sitt første program og var den hurtigste datamaskinen så langt. Denne datamaskinen var i stand til å lagre program i et minne og deretter kjøre det. Av ulike årsaker forlot Turing arbeidet rundt ACE og var ikke en del av ferdigstillingen. I mellomtiden designet han et programmeringssystem som ble en del av den første kommersiell tilgjengelige datamaskinen. Denne maskinen het Feranti Mark I og ble først produsert i 1951 med 10 utgaver solgt.

Allan Turing tenkte allerede på kunstig intelligens (KI) i 1947 (Copeland, 2000; McCarthy, 2007). Ifølge Copeland (2000) var hans visjon at datamaskinen skulle kunne lære fra erfaringer og være i stand til å endre sine egne oppgaver. Det vil da si at KI ikke er noe nytt som kom med Chat GPT, Microsoft Cortana og andre lignende. Ifølge McCarthy (2007) mente Turing at maskiner som kunne oppføre seg som et menneske framfor brukeren var intelligente maskiner. Han nevner flere applikasjoner som er KI og blant de er applikasjoner som spill i form av å spille mot en robot og språkgjenkjenning som er tilgjengelig for alles bruk. Det første fungerende KI programmet var, ifølge Copeland (2000) en spillerobot i spillet dam. KI programmet var i stand til å spille et helt spill allerede i 1952. Programmet ble skrevet med oppmuntring fra Turing før det i senere tid ble lansert en egen bok om hvordan å programmere Feranti Mark I maskiner. På samme tidspunkt ble det også programmert en sjakkrobot som kunne analysere alle mulige trekk frem til en løsning. Begrensingen var at den ikke kunne kjøre et helt spill, men finne løsninger til problemet matt i to trekk. Turing arbeidet derimot med et program som kunne kjøre hele sjakkspill, men ble aldri ferdig med det.

2.1.2 Hva er programmering

Petzold (2000, s.31) drar frem analogien med en lysbryter. Er lyset på legger kontakten seg inn og elektronene får en sluttet krets slik at det blir lys. Dette kan tolkes som binært 1. Er lyset av legger kontakten seg ut og kretsen er ikke lenger sluttet. Binært vil det samsvare med 0. Er noe på representerer det 1, er noe av representerer det 0. Videre opplyser Petzold (2000, s. 33) at enn kan bruke denne tankegangen om lysebryteren til å sende morsekoder og er det som ligger til grunne for telegrafsystemet på 1800 tallet. Dette var en teknologi som kunne gi hurtig kontakt via langdistanse (Petzold 2000, s.41). I følge Petzold (2000, s. 46) er teknologien bak et telegrafsystemet en stor bryter.

Allerede rundt 1948 så matematikeren John Wilder Turkey en sammenheng mellom datamaskinen og de binære tallene. Han presenterte det forkortede ordet bit som skulle erstatte ordet binærtall. En bit har deretter også fått definisjonen for å være en bygge-brikke som inneholder informasjon (Petzold, 2000, s. 68 -70). En bit kan gi to forskjellige kodekombinasjoner 2^1=2. To bit gir 2^2 = 4 ulike kode kombinasjoner. Tre bit vil kunne gi 2^3 = 8 ulike kombinasjoner. Slik fortsetter det (Petzold, 2000, s.76). EAN kodene på matvarer består av bit og er derfor lesbar av en datamaskin. Tykk strek indikerer 11, tykt mellomrom indikerer 00, tyn strek 1 og tynt mellomrom 0. Altså det binære tall systemet (Petzold 2000, s.79 -80). Bit kan også representere ord, bilder, lyd, musikk og filmer og mye mer Petzold

(2000, s.85). Åtte bit heter byte og kan inneholde informasjon fra 00000000 - 11111111 binært altså tallene 0-255 i titallssystemet (Petzold, 2000, s.184). For å få frem størrelsen på dette bedre kan en sammenligne med nåtidens mobiler. De fleste mobiler i dag har en lagring fra 256 eller 512 GB (GigaByte). 256 GB er det samme som 2 199 023 255 552 bit og er en goddel større en 1 byte som består av 8 bit. For at datamaskinen skal kunne huske presenterer Petzold (2000, s.198) random access memory (RAM). Denne komponenten har åtte inputer som kan 'huske' 1 bit på hver input. Det finnes også andre typer RAM komponenter som kan huske mer enn 1 bit, men har fremdeles kun muligheten til å huske 8 slike separate bit-pakker. Det er viktig å notere seg at om strømmen skulle gått vil all minne i rammen forsvinne (Petzold, 2000, s.205). En av oppgavene til RAM er å inneholde maskinkoder som gir instruksjoner for hva CPUen skal utføre (Petzold, 2000, s.305)

Hardware eller maskinvare er alle komponentene som en kan finne i en datamaskin, mens software er programvaren. Skriver en et programvare, skriver en et dataprogram. I slike dataprogram kan en skille mellom koder og data. Dataene er det som kodene manipulerer og kodene er da instruksjoner om hva som skal skje med dataen. Prosessoren Central Prosessing Unit (CPU) kan assosieres som datamaskinens hjerne. Denne CPU responderer til maskinkoder som gjør det mulig for å skrive inn handlingene en ønsker datamaskinen skal utføre (Petzold, 2000, s.232 - 233). Maskinkoder består av de binære tallene og kan være slitsomt for en programmerer å arbeide med. Det finnes dermed et assembly språk som er en mellomting av binært og det engelske språket (Petzold, 2000, s.236). Dette språket blir sett på som et lavt-nivå språk da det står nært maskinkoden og hardwaren. Høyt-nivå språk er det dataspråket som assosieres med de andre programmeringsspråkene, noen av et høyere nivå enn andre. Eksempel på høyt-nivå språk er Python, R og Java. For at dataen skal kunne lese programmeringsspråk av et høyt-nivå trengs en kompilator. Dens oppgave er å oversette programmeringsspråket til maskinkode. Den positive siden med et høyt-nivå språk er at det er lettere for programmereren å forholde seg til og kan brukes på de fleste CPUer. Det er enklere å lære seg den type språk istedenfor Assembly språkene. Programmereren trenger heller ingen kunnskap om den underliggende strukturen til en datamaskin. Den negative siden er at et høy-nivå språk minsker kapasiteten til en CPU i forhold til assembly språkene. Dette da høyt-nivå språk også må oversettes for at CPUen kan forstå kodene (Petzold, 2000, s.352-353).

Programmering er det en gjør når en skal snakke med datamaskinen for å få den til å utføre ulike handlinger. Oftest programmerer en via et høyt nivå programmeringsspråk, eks Pyhton,

som igjen blir oversatt til maskinkode via en kompilator. Det vil da si at den mer menneskelige lesbare koden blir oversatt til bit som består av binærtall som datamaskinen kan lese. På denne måten vil datamaskinen kunne forstå og utføre den handlingen som programmere ba maskinen om å utføre.

2.1.3 Hvordan ser Programmering ut i skolen?

Av programmeringen som er tatt inn i skolen, kan en skille mellom blokkprogrammering og tekstprogrammering. Forskjellen mellom disse to typene er hvordan elevene programmerer. Ved blokkprogrammering, vil elevene pusle sammen ulike blokker som kan utføre ulike oppgaver (Taylor et al., 2010). På denne måten kan en få et program til å kjøre via disse blokkene. Tanken er at elevene skal kunne lære seg å programmere uten å måtte beherske og huske alle kodene slik som ved tekstprogrammering (Statped, 2021). Av blokkprogrammering finnes det både fysiskprogrammering og digitalprogrammering. Fysiskprogrammering innebærer at elevene får kjøre et program fysisk for å se resultatene. Et eksempel er Micro-Bit hvor elevene først programmerer et program og laster det over til en prosessor som er utstyrt med ulike sensorer og innganger. Dette gjør det mulig for enheten å oppfatte data utenifra som for eksempel bevegelse eller andre inputer. Lego Education er også en fysiskprogrammering på lik linje med Micro-bit bare med enda flere deler og kanskje flere muligheter til å lage et helt økosystem. Scratch er en digital plattform for programmering og er ofte forbunnet med å lage egne spill, animasjoner eller visuelle simulatorer. Scratch er et program laget av MIT og er bygget på ideene og arbeidet til Seymour Papert (Taylor et al., 2010). Ved tekstprogrammering må elevene huske kommandoene og skrive inn kodene slik som en programmerer. Det finnes flere programmeringsspråk som JavaScipt, c, Python og flere (Statped, 2021). I Norske skoler varierer bruken av blokk og Tekstprogrammering. Blokkprogrammeringsverktøyene varierer, men av tekstprogrammering er somoftest Python. Python er det programmeringsspråket en møter mest i arbeidslivet, og kan da være en relevant ferdighet for elevene å kunne i fremtiden (Haraldsrud et al., 2020, s. 14; Statped, 2021). Micro-bit har også en funksjon hvor elevene kan velge om de vil kode med blokk eller tekst. Elevene får da mulighet til å programmere Micro-Bitter med programmeringsspråket JavaScript. Det ligger innebygget funksjon hvor blokkprogrammet som eleven har programmert, bli oversatt til JavaScript. På denne måten får elevene se og oppleve sitt eget program i tekstformat. Noe som kan virke motiverende til å få elevene til å utforske mer av tekstprogrammeringen (Statped, 2021).

2.2 Hvorfor programmering i skoler

Det er ikke første gang programmering har forekommet i norske skoler. Helt tilbake til 80-tallet inntraff den første bølge med programmering i norske skoler. Via forskning mente Seymour Papert at unge elever var i stand til å programmere avanserte algoritmer (Dolonen et al., 2019). Forsøkene hans innebar at elever fikk programmere en skilpadde til å bevege seg på en skjerm. Sammen med programmering, matematikk og den konstruktivistiske læringsteorien, var intensjonen å gi elevene et annerledes læringsmiljø som skulle fremme engasjement, motivasjon og læring av matematikk (Forsström & Kaufmann, 2018). Han mente at slike læringsprosesser med utforsking og prøving kunne føre til utvikling av problemløsningsferdigheter og kognitiv utvikling. Dette var ikke en tanke han sto alene med i sin tid og var grunnlaget til at programmering i skoler både i USA og en rekke andre land. Allerede på 90 tallet begynte entusiasmen rundt programmering å dale. Mest fordi det ble vanskelig å finne støtte for Papert sin påstand (Dolonen et al., 2019; Forsström & Kaufmann, 2018). 40 år senere kommer andre bølgen med programmering i skolen. Denne gangen var ikke fokuset lenger på programmering og kognitiv utvikling, men programmering og problemløsing (Dolonen et al., 2019). I starten var det stor usikkerhet om matematikk skulle implementeres i allerede etablerte fag, som et eget fag eller en kombinasjon av disse (Dolonen et al., 2019; Kaufmann & Stenseth, 2021). Med en ny læreplan, Lk20, har Norge valgt å implementere programmering i matematikk, naturfag og musikk

En står nå ovenfor en fjerde industriell revolusjon. Revolusjonen bringer både utfordringer og muligheter som krever adaptering og forståelse av den nye teknologien For å kunne kontre disse endringene må en tilegne seg ferdighetene som kritisk tenkning, algoritmisk tenkning og kunne interdisiplinære disse (Dolonen et al., 2019; Forsström & Kaufmann, 2018). Det er altså skolens sosiale mandat som tilsier at det er skolens ansvar å kunne forme den fremtidige generasjonen. Skolen skal sørge for at enkelt individer blir borgere som har kunnskapen, ferdigheten og kompetansen som skal til å kunne bringe samfunnet videre. For å kunne få til dette må skolene tilpasses til den nye teknologien ved å integrere den i læringsmiljøet. Programmering er en ny måte for elevene å lære og forstå kunnskap og er så mye mer enn bare et verktøy. Noe som ikke synes igjen i den nye læreplanen LK20 da den presenterer programmering som et verktøy (Bungum & Vinnervik, 2022). I følge Haraldsrud et al. (2020, s.14) vil programmeringen kunne styrke fagene. Med dette mener de at programmering kan virke inn på dybdelæringen. Programmering kan gi muligheten til å utforske konsepter via flere metoder. Det kan også muliggjøre utforsking av konsepter som ikke har vært mulig før.

Dolonen et al. (2019) løfter også frem at skolen i dag har mer rom for prosjekter, noe som gir mulighet for tverrfaglighet. Programmeringen passer også fint inn i denne tilnærmingen.

En annen side av hvorfor programmering skal inn i skolen er å kunne forstå mer av den digitale verdenen. Hvordan kan det ha seg at alt i verden koster penger, men allikevel er det ulike digitale tjenester på nett som er helt gratis å bruke. Det koster penger å drive serverene til slike sider, så hvor får de inntektene fra? Ved å ha en forståelse for programmering kan en ha en større sjans til å forstå hvordan informasjon selges, og hvordan slike sider får inntekter. Barn som vokser opp nå, vokser opp sammen med teknologien. Det vil dermed ikke si at de har digital kompetanse. De har kompetanse som skal til for å kunne bruke teknologien, men har ikke forståelsen for hva som ligger bak det digitale samfunnet. De har heller ikke det som trengs for å kunne forholde seg til det (Haraldsrud et al., 2020, s. 15). Kunstig Intelligens som Chat GPT er et resultat av programmering på høyt nivå og Haraldsrud et al. (2020, s. 16) nevner hva det digitale samfunnet har å si for slik intelligens. De tar opp for eksempel Microsofts tidligere forsøk på en samtalerobot på Twitter som kun etter en dag hadde lært seg selv å bli en rasistisk kvinnehater. Dette setter da spørsmål til hva annet på nett kan ha å si for andre typer elektronikk. Det nevnes blant annet problematikken rundt selvstyrte biler som kan lære seg til å ta valg basert på selvlærte holdninger og verdier. Det kan da ende opp med å kunne ta valg som ikke er programmerte. Det er altså flere grunner for at programmering skal inn i skolen. Den ene er fordi programmering har mulighet til å berike fagene og kunne føre til algoritmisk tenkning, kritisk tenkning og dybdelæring. Den andre grunnen er at den teknologiske utviklingen og det digitale samfunnet krever at folk har en viss digital kompetanse. Med en slik kompetanse har folk et bedre grunnlag for å kunne reflektere og forstå hva som skjer på nettet og det som ligger bak.

2.3 Programmering i læreplanen

2.3.1 Digitale ferdigheter og programmering

Den grunnleggende ferdigheten digitale ferdigheter er ikke ny, men inkluderingen av programmering i de digitale ferdighetene er nytt med LK20. De grunnleggende ferdighetene er noe som skal opparbeides igjennom hele grunnskoleopplæringen og ligger derfor ikke som et kompetansemål (Utdanningsdirektoratet, 2020b). Utdanningsdirektoratet definerer digitale ferdigheter slik:

> *Digitale ferdigheter i matematikk innebærer å kunne bruke graftegner, regneark, CAS, dynamisk geometriprogram og programmering til å utforske og løse matematiske problemer. Videre innebærer det å finne, analysere, behandle og presentere informasjon ved hjelp av digitale verktøy. Utviklingen av digitale ferdigheter innebærer i økende grad å bruke og velge hensiktsmessige digitale verktøy som hjelpemiddel for å utforske, løse og presentere matematiske problemer.* (Utdanningsdirektoratet, 2020a)

Ifølge deres definisjon innebærer digitale ferdigheter kunnskapen om de ulike digitale verktøyen og kunne bruke de til å løse matematiske problemer. Programmering også likestilt med de andre ulike digitale verktøyene. Bungum & Vinnervik (2022) poengterer at programmering er et digitalt verktøy og dermed en del av den digitale ferdigheten. Men hva så med kompetansemålene? Hva sier de om digitale ferdigheter og programmering?

På Udir.no kan en sjekke kompetansemålene fra 1-10 trinn. Ved å sortere kompetansemålene etter grunnleggende ferdigheter kan en da få en oversikt over hvilke kompetansemål utdanningsdirektoratet mener hører til den grunnleggende ferdigheten digitale ferdigheter. Kompetansemålene sett nedenfor er hentet fra Utdanningsdirektoratet (2020b). Da elevene på 1. trinnet teller inn under kompetansemålene for 2. trinnet, er ikke dette trinnet med i tabellen. 2. trinnet har ingen kompetansemål som dekker digitale ferdigheter. Derfor starter tabellen nedenfor fra 3. trinn.

Trinn	**Kompetansemål hentet fra LK20 matematikk 1-10 trinn fra Udir.no**
3.trinn	• *«lage og følge regler og trinnvise instruksjoner i lek og spill knyttet til koordinatsystemet»*
4.trinn	• *«utforske, beskrive og sammenligne egenskaper ved to- og tredimensjonale figurer ved å bruke vinkler, kanter og hjørner»* • *«utforske og beskrive strukturer og mønstre i lek og spill»* • *«lage algoritmer og uttrykke dem ved bruk av variabler, vilkår og løkker»*
5.trinn	• *«diskutere tilfeldighet og sannsynlighet i spill og praktiske situasjoner og knytte det til brøk»* • *«lage og løse oppgaver i regneark som omhandler personlig økonomi»* • *«lage og **programmere** algoritmer med bruk av variabler, vilkår og løkker»*
6.trinn	• *«utforske og beskrive symmetri i mønstre og utføre kongruensavbildninger med og uten koordinatsystem»*

	• *«bruke ulike strategier for å regne ut areal og omkrets og utforske sammenhenger mellom disse»* • *«bruke variabler og formler til å uttrykke sammenhenger i praktiske situasjoner»* • *«bruke variabler, løkker, vilkår og funksjoner i **programmering** til å utforske geometriske figurer og mønstre»*
7.trinn	• *«utforske og bruke hensiktsmessige sentralmål i egne og andres statistiske undersøkelser»* • *«logge, sortere, presentere og lese data i tabeller og diagrammer og begrunne valget av framstilling»* • *«lage og vurdere budsjett og regnskap ved å bruke regneark med cellereferanser og formler»* • *«bruke **programmering** til å utforske data i tabeller og datasett»*
8.trinn	• *«utforske, forklare og sammenligne funksjoner knyttet til praktiske situasjoner»* • *«representere funksjoner på ulike måter og vise sammenhenger mellom representasjonene»* • *«utforske hvordan algoritmer kan skapes, testes og forbedres ved hjelp av **programmering**»*
9.trinn	• *«utforske og argumentere for hvordan det å endre forutsetninger i geometriske problemstillinger påvirker løsninger»* • *«tolke og kritisk vurdere statistiske framstillinger fra mediene og lokalsamfunnet»* • *«finne og diskutere sentralmål og spredningsmål i reelle datasett»* • *«utforske og argumentere for hvordan framstillinger av tall og data kan brukes for å fremme ulike synspunkter»* • *«beregne og vurdere sannsynlighet i statistikk og spill»* • *«simulere utfall i tilfeldige forsøk og beregne sannsynligheten for at noe skal inntreffe, ved å bruke **programmering**»*
10.trinn	• *«utforske og sammenligne egenskaper ved ulike funksjoner ved å bruke digitale verktøy»*

- *«utforske sammenhengen mellom konstant prosentvis endring, vekstfaktor og eksponentialfunksjoner»*
- *«hente ut og tolke relevant informasjon fra tekster om kjøp og salg og ulike typer lån og bruke det til å formulere og løse problemer»*
- *«planlegge, utføre og presentere et utforskende arbeid knyttet til personlig økonomi»*
- *«bruke funksjoner i modellering og argumentere for framgangsmåter og resultater»*
- *«modellere situasjoner knyttet til reelle datasett, presentere resultatene og argumentere for at modellene er gyldige utforske matematiske egenskaper og sammenhenger ved å bruke **programmering**»*

Som sett i tabellen øker antall kompetansemål som faller innenfor digitale ferdigheter i takt med trinnene. Det er ikke alle kompetansemålene som nevner programmering og kompetansemålene som eksplisitt nevner programmering starter fra 5 trinnet av. I følge Bungum & Vinnervik (2022) forventer ikke den Norske læreplanen at elevene skal bruke programmering som et verktøy til å arbeide med matematikk før 7 trinnet. 5-6 trinnet er tanken at elevene skal lære programmering slik at de er klar til å kunne bruke dette verktøyet fra 7 trinnet av. Selv om det bare er ett kompetansemål på hvert trinn som spesifikt nevner programmering som arbeidsmåte betyr ikke det at det kun er et kompetansemålet som er egnet til å bruke programmering. Bungum & Vinnervik (2022) mener at flere av kompetansemålene inviterer til å bruke programmering. Som en ser i tabellen ovenfor er det flere kompetansemål som ligger under digitale ferdigheter som ikke nevner programmering. Da står læreren fritt å kunne velge hvilke digitale verktøy som egner seg best til å de ulike kompetansemålene. Eks er kompetansemålet fra 4 trinnet *«lage algoritmer og uttrykke dem ved bruk av variabler, vilkår og løkker»*. Dette kompetansemålet inneholder ikke ordet programmering, men det stopper allikevel ikke opp for å kunne arbeide med kompetansemålet ved bruk av programmering. Det samme gjelder for de fleste andre kompetansemålene på tvers av trinnene. Selv om det ikke eksplisitt nevnes programmering, er det fortsatt en mulighet å arbeide med dette kompetansemålet via programmering. Det er bare opp til læreren å vurdere hvilke av disse kompetansemålene som eigner seg sammen med dette digitale verktøyet.

2.3.2 Problemløsing, Algoritmisk tenking og programmering

Når en tenker på algoritmisk tenking, eller computational thinking som det heter på engelsk, er det fort gjort å tenke at dette bør ha en sammenheng med datamaskiner og programmering. Eller for de som har erfaringer fra læreryrket en del år tilbake, vil kanskje kunne knytte dette begrepet sammen med standard algoritmer (Gjøvik & Torkildsen, 2019). I følge Bungum & Vinnervik (2022) er programmering ikke det samme som algoritmisk tenkning, men er et verktøy til å forstå og utvikle algoritmisk tenkning. Men hva er egentlig algoritmisk tenkning? Utdanningsdirektoratet definerer begrepet algoritmisk tenkning som følgende:

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

Utdanningsdirektoratet (2019) knytter sammen algoritmisk tenkning som en problemløsningsmetode. Dette da en side av algoritmisk tenkning er å arbeide med problemer på en systematisk måte. Sammen med begrepsforklaringen introduseres også nøkkelbegrepene: Logikk, Algoritmer, Dekomposisjon, Mønstre, Abstraksjon og Evaluering.

Figur 1: Den algoritmiske tenkeren [Illustrasjon], av Utdanningsdirektoratet, 27.03.2019, Utdanningsdirektoratet *https://www.udir.no/kvalitet-og-kompetanse/digitalisering/algoritmisk-tenkning/*

Videre forklarer de at Algoritmisk tenkning dreier seg om å kunne dekomponere problemene i mindre deler. Det dreier seg om analysering og organisering av informasjon slik at algoritmer kan oppstå, og på denne måten skape en vei til løsningene. Det å kunne modellere noe som representerer virkeligheten og problemstillingene. Det å kunne abstrahere og gjøre noe mer effektivt. Korte ned unødvendige steg i løsningsstrategiene. Det som de legger ekstra vekt på er det å se tilbake på arbeide. Som Polyas (1945) fjerde fase sier *looking back*, må elevene evaluere sitt eget arbeid. Er det riktig? Kan det effektiviseres, er det relevant? Utdanningsdirektoratet (2019) bruker ordet kognitiv kondis i sammen med algoritmisk tenkning da arbeidet er krevende og kan føles motarbeidende.

Hva sier internasjonal forskning om algoritmisk tenkning? Helt i starten må det noteres at den norske oversettelsen av algoritmisk tenkning kan virke noe misvisende. Algoritmisk tenkning er i teorien oversettelsen av Computational thinking. Den internasjonale utgaven computational thinking har en underkategori som heter algorithmic thinking. Det vil da si at algoritmisk tenkning og algorithmic thinking ikke helt er det samme (Gjøvik & Torkildsen, 2019).

Wing (2006) sier at computational thinking involverer prosesser som både en datamaskin og et menneske kan utføre. Allikevel er det er ikke ment at mennesker skal måtte tenke som datamaskiner. Hele tanken er en systematisk prosess til å kunne løse problemer, og dette krever kreativiteten til et menneske. Det nevnes også at computational thinking er så mye mer enn bare programmering da det er en tenkemåte som opererer på flere plan av abstraksjoner. Det er bare et menneskes kreativitet som kan begrense problemet og løsningene. Computational thinking er en ferdighet som alle bør har i det dagligdagse livet. Shute et al., (2017) definerer computational thinking slik: «*the conceptual foundation required to solve problems effectively and efficiently (i.e., algorithmically, with or without the assistance of computers) with solutions that are reusable in different contexts*.». En kan se en tydelig sammenheng med både Shute et al. (2017) og Wing (2006) som legger vekt på det å løse et problem både effektivt og på en fornuftig måte. Dette samsvarer også med Utdanningsdirektoratet (2019) sin definisjon. Selv om det finnes flere definisjoner både internt i Norge og på et internasjonalt nivå (Dolonen et al., 2019), er de enige om at computational thinking/ algoritmisk tenkning er en systematisk måte å tilnærme seg et problem.

2.4 Programmering og Matematikk

2.4.1 Programmeringens plass i matematikk

I følge (Forsström & Kaufmann, 2018) er det flere land i Europa som har valgt å implementere programmering i faget matematikk. Norge har valgt å gå i de samme sporene (Stenseth et al., 2019). Det er flere argumenter til hvorfor programmering skal inn i matematikken: Programmering kan gi mulighet for å koble matematikken til den virkelige verdenen, programmering kan virke motiverende for elevene til å lære matematikk, programmeringen kan virke på å øke den matematiske utøvelsen (Forsström & Kaufmann, 2018). Kaufmann & Stenseth (2021) mener at programmeringens plass i matematikk er knyttet til problemløsning og den algoritmiske tenkningen. Grover & Pea (2013) mener at programmering, matematisk tenkning og algoritmisk tenkning hører sammen. Ut i fra Grover & Peas (2013) argument kommenterer Stenseth et al. (2019) at programmering da må spille en viktig rolle i fremtidens digitale samfunn og kunnskap innenfor problemløsning, kreativitet og logikk. Elevene skal lære hvordan de kan bruke teknologien til samarbeid, kommunikasjon, kreativitet og problemløsning (Dolonen et al., 2019). I følge Forsström & Kaufmann (2018) har programmering en naturlig sammenheng med geometri, men at det

mangler videre forskning for å kunne knytte sammen programmering med de andre emnene. Selv om Forsström & Kaufmann (2018) kun nevner geometri kan en fortsatt argumenterer for at sannsynlighet og statistikk også har en naturlig sammenheng med programmering. Dette er da programmeringsspråket R har statistikk og sannsynlighet som et grunnlag for sin funksjon. Videre i Forsström & Kaufmann (2018) studie argumenteres det for at programmering kan utvikle problemløsningsevner, logisktenkning og kunne motivere elever til å lære matematikk. I følge Grover & Pea (2013) er programmering i matematikk en måte å utvikle algoritmisk tenkning. Sammen med dette mener Forsström & Kaufmann (2018) at elever må da kunne møte programmering i andre emner for å kunne utvikle disse ferdighetene.

I sin studie løfter Kaufmann & Stenseth (2021) frem to viktige spørsmål til programmering og problemløsning i matematikk. (1) Er problemet designet slik at programmeringen kan gi en bedre matematisk forståelse? (2) Eller brukes matematikk til å forbedre problemløsningsferdighetene i programmering. De argumenterer at denne todelingen av synspunkter er viktig for at læreren skal kunne gi gode råd til elevene i deres løsningsprosesser. Flere Europeiske lands argumenterer for at programmeringens plass i matematikk er at programmeringen kan styrke elevens problemløsningsevner. Til dette advarer de imot å tro at dette skjer automatisk og oppfordrer at fokuset ikke bør ligge rent på programmeringen. Dette da elevers problemløsningsevner og refleksjonsevner kan begrense deres evne til å løse oppgaven. Taylor et al. (2010) argumenterer for at elever kan utforske matematikken og programmeringen ved bruk av kreative verktøy, som for eksempel Scratch. De er fremdeles usikre på hvilken matematisk forståelse elevene sitter igjen med. Forsström & Kaufmann (2018) konkluderer også at programmering hører til i matematikk faget, men at det trengs mer forskning rundt tematikken for å kunne argumentere programmeringens plass.

Da programmering ofte blir knyttet sammen med problemløsning (Kaufmann & Stenseth, 2021), er det ikke unaturlig å sammenligne en programmeringsprosess med en problemløsningsprosess. I følge Polya (1945, s. 3 - 5) vil det ta tid for elevene å plukke opp problemløsningsferdigheten. Denne ferdigheten vil bedres etter å se andre løse problemer, eller selv løse problemer. Liljedahl (2021, s. 30) mener også at en må bruke tid og la elevene jobbe med oppgaver utenfor læreplanen før de kan være i stand til å jobbe med oppgaver i takt med læreplanen. Ved programmering er det den samme tankegangen. Det vil ta tid før eleven kan bruke programmeringen effektivt til å løse oppgaver. En må ta seg tid til å lære elevene det grunnleggende med programmering, før elevene er i stand til å bruke programmering som et verktøy.

2.4.2 Lærerrollen og samarbeid

Seysmour Papert assosierte læring via programmering sammen med den konstruktivistiske læringsteorien til Piaget. Han utviklet programmeringsspråket LOGO og undervisningsopplegg rundt denne læringsteorien. Han var den første som inkluderte programmering i matematikk, allerede i 1980. Hensikten hans var at elevene skulle lære, og bli motivert til å lære matematikk (Forsström & Kaufmann, 2018). Ved å benytte seg av samarbeidslæring kan ventetiden på hjelp minimeres og kunne motvirke elevenes frustrasjon eller kjedsomhet under problemløsning (Liljedahl, 2021, s. 149-150). Som resultat fra en litteraturgjennomgang av programmering i matematikk fant Forsström & Kaufmann (2018) at det var den sosiokulturelle læringsteorien og den konstruktivistiske læringsteorien som gikk igjen når det var snakk om programmering. Derfor anbefaler de å se på programmering som en samarbeidende prosess for å kunne bygge bedre forståelse for programmering i matematikk. Det å se på interaksjonene mellom elev, lærer og utstyr. Imsen (2015, s. 145 - 146) mener også at språket står sentralt i begge læringsteoriene når det kommer til kunnskapsbygging. Til og med ute i arbeidslivet hvor det jobbes med programmering er det normalt med samarbeid rundt det å kode. Ifølge forskning kan et program som er skrevet av flere enn en person være mer presise, da personer tenker ulikt og har ulike synspunkter (Haraldsrud et al., 2020, s. 170).

Forsström & Kaufmann (2018) viser til en gjenganger av lærers roller igjennom litteraturstudie sitt. I den ene artikkelen fra deres litteraturstudie kommenterer Taylor et al. (2010) at lærerens rolle i programmeringstimer går fra underviser til å guide og støtte elevene i deres arbeid. Det betyr at samarbeidslæringen ikke bare skjer mellom elevene, men også mellom lærer og elev. Lærerens rolle har noe å si for klasseromsmiljøet og samarbeidsevnen til elevene. Det er ikke bare læreren som trer inn i roller, elevene trer også inn i roller. Et klasserom som er preget av respekt hvor elever kan lytte til hverandres ideer og meninger, kan gi elever muligheten til å endre deres roller. Et eksempel er en elev som synes matematikk er vanskelig, kan kunne ta på seg en mer ledende rolle og kan stå ansvarlig for de matematiske ideene fra gruppa. På denne måten kan elevene lære fra hverandre og kan hjelpe hverandres problemløsningsevner (Forsström & Kaufmann, 2018). Taylor et al. (2010) argumenterer at elever som har en ‘‘lav’’ matematisk forståelse har muligheten til å utvikle en kompleks algoritmisk tenkning og bruke gode matematiske ideer. De vektlegger at de sosiale omgivelsene spiller sannsynligvis en viktig rolle i dette resultatet. De viser også til i sin forskning hvordan samarbeidslæring kan påvirke elevenes erfaring med programmering. Selv

om arbeidet var individuelt, ble arbeidet deres jevnlig tatt opp på tavlen framfor alle de andre elevene. Klassen fikk da diskutert prosjektet og resulterte med at alle elevene følte en slags deltagelse i alle sitt arbeid. Da er en tilbake til læringsteoriene, hvor elevene får brukt språket til å sette ord på egen kunnskap i denne kunnskapsdelings prosessen. Denne metoden å arbeide på kan igjen føre til dybdelæring (Haraldsrud et al., 2020, s. 170).

Stenseth et al. (2019) løfter også frem viktigheten av lærerens rolle i slike undervisningstimer. Læreren er mer en guide som har ansvar for elevenes samarbeid. De oppfordrer derfor at lærerne planlegger opplegget rundt samarbeidet mellom elevene og samarbeider mellom elev og lærer. De anbefaler også en kunnskapsdeling slik at elevene får sett hverandres løsninger. Kaufmann & Stenseth (2021) knytter lærerrollen opp mot den virkelige verdenen. Med å sammenligne lærerrollen med en klient i programmeringsyrket som skal ha fokus på kvaliteten på programmet, er det dermed nødvendig at læreren har god programmeringsferdigheter. I sin studie løfter Forsström & Kaufmann (2018) frem at lærerroller kan økende samarbeid i klasserommet. Hvilken rolle læreren tar på seg har noe å si for samarbeidslæringen. De anbefaler også å se på læringen på et klassenivå istedenfor et individuelt nivå. På denne måten kan læreren få en god innsikt og kunnskap over brukbarheten av programmering i matematikk

Forsström & Kaufmann (2018) argumenterer for at det trengs mer diskusjon over hvilke ferdigheter lærere må ha for å kunne undervise programmeringstimer i matematikk. Det må også diskuteres om programmering skal implementeres i lærerutdanning og får de som allerede er ferdig med lærerutdanning. De fleste aktive lærerne i Norge har ikke programmering i egen fagkrets, noe som er en utfordring når de da skal stå ansvarlig for opplæring av programmeringsferdighetene til elevene.

2.5 Forskningsbasert undervisningsmetoder og modeller

Ifølge Dolonen et al. (2019) er det lite forskning på undervisningsopplegg når det kommer til programmering. Taylor et al. (2010) stiller seg også kritisk da de argumenterer for at elever kan utforske matematikken og programmeringen ved bruk av kreative verktøy som Scratch, men at de fremdeles er usikre på hvilken matematisk forståelse elevene sitter igjen med. En av undervisningsoppleggene som Dolonen et al. (2019) presenterer som de synes virker lovende er PRIMM-modellen. Denne modellen er et planleggingsverktøy for programmeringstimer og er forankret i forskning (Matematikksenteret, u.å). I følge Dolonen

et al. (2019) vektlegger denne modellen samarbeid og har også vist mulighet for dybdelæring. Da flere elever opplever utfordring med å starte å programmere, skal dette verktøyet hjelpe elevene i gang. PRIMM består av 5 faser: Predict *(forutsi)*, Run *(kjøre)*, Investigate *(utforske)*, Modify *(modifisere)* og Make *(lage)*. Ved å introdusere elevene for et nytt program får læreren elevene til å *forutsi* hva programmet gjør. Ved å *kjøre* programmet vil elevene få en tilbakemelding på deres prediksjoner. Ved å la elevene *utforske* kan eleven få muligheten til å utvikle kodeforståelsen. Læreren kan hjelpe elevene med denne utforskningen ved å be de spore opp verdier til variabler i programmet. Dette kan gi elevene muligheten til logisk tenkning. De kan be elevene forklare hva programmet gjør eller skrive kommentarer til hva hver sekvens gjør. De kan be elevene feilsøke et program som i utgangspunktet er planlagt til å ikke fungere. De kan gi elevene et program og la elevene *modifisere* programmet til å utføre en annen oppgave. Siste steget er å la elvene bygge et program fra bunnen av som skal baseres på nye problemer og funksjoner (Matematikksenteret, u.å).

Sentance & Csizmadia (2017) nevner metoden Unplugged. Oftest består undervisningsmetoder til programmeringstimer av en slags type software, eksempelvis Scratch eller Codemonkey. Med Unplugged derimot er poenget å ikke bruke noen form for teknologi. Et eksempel kan være å bruke en kopp som en hukommelseslokasjon som elevene kan definere med en variabel. Gjøvik & Torkildsen (2019) nevner også en lignende variant av unplugged. Denne starter med helt analoge oppgaver som etter hvert kan utvikles til digitale oppgaver. Både Sentence & Csizmadia (2017), Dolonen et al. (2019) og Gjøvik & Torkildsen (2019) er enige rundt undervisningsmetoder som baserer seg på fysisk programmering. De anbefaler undervisningsmetoder som baserer seg delvis på bruk av teknologi som programmering av en robot eller et kretskort. Eksempel på dette er, Micro-Bit, Blue-Bot og Lego Education. Her får elevene programmere digitalt for å se programmet kjøre fysisk. Om dette er å få roboten til å kjøre en løype eller utføre ulike handlinger.

Kopiering og justering av andres program er fundamentalt i programmering. Det er slik programmerere også arbeider. Det vil da si at en slik måte å arbeide på ikke må bli sett på som juksing. Selv om elevene kopierer, trenger de fortsatt kunnskapen om det programmet de kopierer. Dette da de må ha den underliggende forståelsen for å kunne formulere hypoteser om hva som er feil eller må endres. Sett fra en matematisk siden kan en si at elevene arbeider på en måte som innebærer kritisk tenkning (Kaufmann & Stenseth, 2021). Denne metoden å arbeider på sklir også inn i PRIMM modellen som Matematikksenteret (u.å) presenterer.

Figueiredo & García-Peñalvo, (2020) løfter frem trenden med gamification av programmeringsundervisnings eller såkalt spill-basert læring. De påstår at denne undervisningsmetoden for programmeringen kan bidra til økt motivasjon hos elevene. Tanken er at elevens hverdag består mye av spill, så da må en slik undervisningsmetode føre til mer engasjement til å lære programmering. Zhan et al. (2022) viser til resultater fra egen studie hvor spill-basert læring kan ha en innvirkning på elevers motivasjon, akademisk prestasjon og tenkeferdigheter. Allikevel rapporterer de mangelen på kognitiv belastning og anbefaler læreren å fokusere på riktig bruk av scaffolding and fading. De argumenterer for at mangelen på dette kan være av at programmering enda er en nytt kunnskapsområdet og at elevene enda ikke har fått opparbeidet seg de nødvendige ferdighetene. Ved å søke på ordet spill dukker dette ordet opp i kompetansemålene for digitale ferdigheter på 3 – 5 trinn også et på 9 trinn. Noe som da gir rom for den nye trenden for spill-basertlæring i programmering.

2.6 Utfordringer og suksessfaktorer

2.6.1 Utfordringer

Fixdal (2018) skriver i sin artikkel om forskjellen på en utfordring og et problem. Hun definerer de to variantene følgende:

> *«Ordet «utfordring» beskriver noe man kan strekke seg etter, en situasjon der man får brukt sine ferdigheter og evner og kan presse grensene for hva man egentlig klarer. Det er noe det kan være mulig å oppnå …… Et problem er noe mer alvorlig. Det er noe som man ikke nødvendigvis kan løse ved å strekke seg litt lenger. Det er mer omfattende, og man kan trenge hjelp for å finne en løsning.»* (Fixdal, 2018)

I en sammenheng med implementering av programmering, vil det bli mer naturlig å bruke ordet utfordring. Dette er da fordi skolen som et kollektivt kan enda strekke seg til å finne løsninger. Det har ikke kommet til et nivå enda hvor det trengs eksterne ressurser til å hjelpe til.

Utfordringer som har dukket opp med tanke på programmering i matematikk er blant annet læreres kunnskapsbase (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). Lærere mener selv at de mangler kunnskap om programmering i matematikk, den såkalte content knowledge og pedagogical content knowledge. Flere lærere har ikke hatt programmering i egen utdannelse og mangler kunnskapen om hvordan anvende

dette til elever i matematikktimer (Finger & Houguet, 2009; Sentance & Csizmadia, 2017; Vinnervik, 2022). Lærere viser til utfordringer knyttet til egen usikkerhet innenfor programmering og føler selv at de ikke kan gi elevene det som de trenger (Finger & Houguet, 2009; Jones et al., 2004; Vinnervik, 2022). Det er utfordrende å holde elevens motivasjon oppe og generelt det å holde følge med elevene (Haraldsrud et al., 2020). De synes også at det er utfordrende å forstå læreplanen for hva elevene skal lære (Jones et al., 2004). Lærerne noterer også at eierskapet til matematikken ikke er den samme nå når programmering er en del av faget (Vinnervik, 2022)

Tid er også en utfordring. Lærere synes det er utfordrende med å få tid i programmeringstimer og generelt implementering av programmering i et fag som allerede fullstappet (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). Det er utfordrende å få tid til å strekke over alle elevene og gi de den støtten som de trenger (Haraldsrud et al., 2020; Statped, 2021). Det er utfordrende med resurser da det kan kreve mye utstyr for å ha programmeringstimer (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). Teknologiske problemer kan også være en utfordring (Haraldsrud et al., 2020; Sentance & Csizmadia, 2017). Elev evaluering er en annen utfordring (Finger & Houguet, 2009; Haraldsrud et al., 2020; Vinnervik, 2022). Hvordan skal lærerne vurdere elevenes kunnskap på en god måte? En annen utfordring som flere lærere løfter frem er mangel på profesjonell støtte. Flere lærere opplever det å ikke få den støtten som de trenger av ledelsen for å kunne implementere programmering inn i faget (Finger & Houguet, 2009; Sentance & Csizmadia, 2017; Vinnervik, 2022).

2.6.2 Suksessfaktorer

Det store norske leksikonet definerer suksessfaktor følgende:

> *«Suksessfaktorer er forhold som må ligge til rette for at prosjektresultatet skal oppfattes som vellykket. Mens suksesskriterier benyttes etter at prosjektet er avsluttes, gjelder suksessfaktorene under selve prosjektgjennomføringen»* (Rolstadås, 2023)

Programmering i matematikk er enda i en implementeringsprosess. Det er enda ikke vellykket innført og lærere opplever fortsatt en rekke utfordringer rundt implementeringsprosessen. Som sett i definisjonen ovenfor er suksessfaktor brukt i sammenheng av løpende prosesser som ikke er i mål enda. Ved å se på implementeringen av matematikk som en prosess som enda ikke har nådd målet, vil det da være passende å bruke

ordet *suksessfaktor* istedenfor *suksesskriterium*. Grunnen til at ordet *løsninger* ikke er gunstig er at ordet løsning vil virke misvisende. Sammenlignet med en matematisk utfordring, vil alle stegene en tar føre til en løsning. I denne konteksten vil lærere vise til trinnvise steg og det vil da bli mer naturlig med suksessfaktorer.

Suksessfaktorer som har dukket opp i studier varierer. Finger & Hoguet (2009) viser til flere suksessfaktorer fra et lederperspektiv. Suksessfaktorene som er interessante i denne studien, er suksessfaktorer som kan hjelpe lærerne. Suksessfaktorer som utgjør handlinger som læreren selv kan ta.

Haraldsrud et al. (2020, s. 164) anbefaler differensiering når det kommer til elevers motivasjon. Eksempel på dette kan være idebank (Haraldsrud et al., 2020, s. 164; Finger & Houguet, 2009). Det kan også hjelpe å informere elevene om at programmering er en stor verden som igjen kan gir rom for alle. Dette kan virke motiverende istedenfor demotiverende. Det er også anbefalt å være ærlig med elevene når det kommer til forskjellene i lærerens og elevenes kunnskap. Elevene kan fort komme i en situasjon hvor de kan mer en læreren eller at eleven trenger hjelp til noe som læreren ikke har svaret. Det er viktig å huske at selv om eleven kan mer programmering, er det fortsatt læreren som er den med den faglige kompetansen (Haraldsrud et al., 2020, s. 164). På denne måten vil læreren alltid kunne bistå elevene i programmeringen så lenge programmeringen er brukt for å tilegne seg faglig kunnskap.

Sentance & Csizmadia (2017) viser til lærere som opplever at det å bruke algoritmisk tenkning som modell har vært en suksessfaktor. Med dette menes det å legge vekt på alle stegene i den algoritmiske tenkningen til å arbeide ut ifra. De anbefaler også metoder hvor elever får programmere uten at de fysisk jobber med teknologi, også kalt unplugged activities. Dette kan for eksempel være i form av pen og papir. Dolonen et al. (2019) anbefaler også metoder for fysisk programmering. Det er type programmering hvor elever får arbeide med noe som skjer fysisk framfor seg og ikke bare på en skjerm.

For å kunne nå over flere av elevene i undervisning dukker det opp suksessfaktoren hvor en øver på elevers egen feilsøkingsevne fra starten av. Med dette menes at elevene blir vant til å spør hverandre, søke på nett og finne ut av feilmeldingene som programmene kan gi (Haraldsrud et al., 2020, s. 165; Sentance & Csizmadia, 2017). Eksempel på dette er å bruke de flinke elevene til å hjelpe de andre elevene. Dette kan i tillegg virke som differensiering, men anbefaler at dette ikke er hovedformen for differensiering til de flinkere elevene

(Haraldsrud et al., 2020, s. 164). Liljedahl (2021, s. 147 - 148) sier at det er viktig å kunne gi oppgaver med riktig utfordring i forhold til elevenes kunnskap. Såkalt differensiering. Er utfordringen for lav vil elever oppleve oppgaven som kjedelig, er utfordringen for stor kan dette føre til frustrasjon. Han løfter også frem problematikken om eleven må vente for lenge på hjelp fra læreren. Da kan eleven bikke over til kjedsomhet eller frustrasjon. Med andre ord er det utfordrende for læreren å holde elevene i *flow* om de ikke får hjulpet eleven i tide (Liljedahl, 2021, s. 149-150). Taylor et al. (2010) argumenterer at programmeringsoppgaver går under low floor, high ceiling oppgaver. Det vil da si at alle elvene skal i teorien klare å jobbe med programmeringsoppgavene og at de mer erfarne elevene også får sitt utbytte av undervisningen. Det vil igjen kunne tolkes som at programmering gjør differensiering lettere.

Suksessfaktorer med vurderinger i programmering kan være å la elevene presentere prosjektet framfor klassen. Da kan læreren stille oppfølgingsspørsmål, for å finne ut av hva elevene kan (Haraldsrud et al., 2020, s. 166). Utenom dette mener Haraldsrud et al. (2020, s166) at programmeringen blir representert i sammen med faget. Det vil si å ikke ha rene programmeringsprøver, men faglige prøver hvor programmering kommer inn. Dette kan komme inn som kortsvarsoppgaver, flervalgsoppgaver, oppgaver hvor elevene finner feilen, skriver om kodene, eller be elevene om å skissere ett program og forklare det (Haraldsrud et al., 2020, s. 166-168)

2.7 Analytisk rammeverk

Igjennom ett litteraturstudie har Finger & Houguet (2009) kartlagt utordringer med implementering av teknologi i skolen. Via denne studien presenterer de en selvlaget modell som skal representere en implementeringsprosess. Prosessen starter med den inkorporerte læreplanen til den erfarte læreplanen (se vedlegg 2). I denne modellen deles lærerutfordringer inn i to kategorier. En for indre utfordringer og en for ytre utfordringer. Ettersom flere land er i en ny implementeringsbølge med teknologi inn i skolen, har Vinnervik (2022) sett på utfordringer i Sverige. I sin studie har han basert seg delvis på Finger & Houguet (2009) sin modell for indre og ytre utfordringer. Han har valgt å kun trekke ut delen av deres modell som dekker utfordringer. Vinnervik (2022) presenterer dermed en egenlaget redigert versjon av Finger & Houguet (2009) modell. I sin modell velger Vinnervik (2022) å inkludere punktet E6 – profesjonell utvikling og støtte. Selv om Finger & Houguet (2009) modell ikke inneholder dette punktet, blir utfordringene fortsatt sortert inn i denne kategorien uten å være

en del av modellen. Det vil da falle naturlig for Vinnervik (2022) å presentere en forenklet modell som inneholder dette punktet.

Table 1 Intrinsic and extrinsic challenges for technology education curriculum implementation

Intrinsic challenges	Extrinsic challenges
I1. Professional knowledge and understanding	E1. Resources
I2. Professional adequacy	E2. Time management
I3. Professional attitudes and values	E3. Practicality of implementation
I4. Teaching approach	E4. Student assessment
I5. Ownership	E5. History and tradition
	E6. Professional development and support

Figur 2: *Vinnervik (2022)*

Av de indre utfordringene har vi, *Profesjonell kunnskap og forståelse (I1), Profesjonell tilstrekkelighet (I2), Profesjonell holdninger og verdier (I3), Undervisningstilnærming (I4) og Eierskap (I5).*

Med *Profesjonell kunnskap og forståelsen (I1)* menes utfordringer rettet mot kunnskapen og forståelse om programmering, programmeringsundervisning og programmeringspedagogikk. Dette kan for eksempel innebære lærers utfordring med teoretisk kunnskap og deretter kunne forankre metodiske valg i teorien. Med *profesjonell tilstrekkelighet (I2)* dreier det seg også om kunnskap, men mer rettet seg selv og elevene. Eksempel på dette kan være usikkerheten om en har god nok kunnskap til å gi elevene en god undervisnings. Kan en nok, er undervisningen tilfredsstillende, lærer elevene det de skal lære på en god og effektiv måte? Denne utfordringen har mye tilfelles med (I1) men fokuserer mer på hva læreren føler. Denne gir rom for lærerens usikkerhet og dårlig selvtillit. *Profesjonell holdninger og verdier (I3)* handler mer på skolens miljø. Hva sier kollegaer om programmering i matematikk? Mener de hele tanken er tull og tøys? Eller mener de at de forstår at elevene skal lære programmering men mener at matematikken ikke er det rette faget? Denne kategorien skal kunne romme utfordringer lærerne møter på med kollegaers holdninger og verdier, og hvordan dette kan påvirke implementeringsarbeidet av programmering i matematikk. *Utfordringer med undervisingstilnærming (I4)* går mer ut på hvilke utfordringer lærere erfarer knyttet til undervisningsmetoder i programmeringsundervising i matematikk. *Utfordring med Eierskap*

(I5) dreier seg om utfordringene lærerne har når det kommer til eierskap til programmering i matematikk.

Av de ytre utfordringer har vi, *Resurser (E1), Tidshåndtering (E2), Praktisk side med implementering (E3), Elevevaluering (E4), Historie og tradisjon (E5) og Profesjonell utvikling og støtte (E6).*

Utfordringer med *resurser (E1)* dreier seg om alt fra mangel på utstyr, penger, assistenter og annet. Utfordringer med *tidshåndtering (E2)* dreier seg om tidsperspektivet. Det er ikke ukjent for en skole å ha en fullpakket læreplan. Matematikk er et fag som er kjent for å ha kort tid til å gå i dybde på de forskjellige læreplanmålene. Det å presse programmering inn i matematikken setter enda mer press på tiden. Programmering i seg selv er også tidkrevende da det krever en del forarbeid for å kunne bruke det aktivt til å løse matematiske oppgaver. Utfordringer med den *praktiske siden med implementering (E3)* er problematikken rundt det å implementere programmering i matematikken, med fokus på den praktiske siden. Dette kan for eksempel være utfordringer rundt starten av implementeringsarbeidet i 2020, hvordan skolen jobbet rundt dette. Utfordringer med *elevevaluering (E4)* er det å vite hvordan elevene ligger an opp imot kompetansemålene. Hvordan kan en lærer finne ut hvordan elevene ligger an? Hvordan kan en jevnlig kontrollere at elevene lærer det de skal? *Utfordringer med historie og tradisjon (E5)* har for det meste en bakgrunn i tidligere forsøk med implementering av teknologi inn i skolen. Da kan en risikere at kollegaer er negative til programmering, da implementeringen første gangen ikke virket. Da teknologi ble forsøkt implementert i de Norske skolene på 80 tallet, har de fleste lærerne fra den tiden gått av med pensjon eller nærmer seg den. Denne kategorien ses da på som ikke aktuell i Norsk kontekst og vil da ikke bli brukt videre i analysen. Utfordringer med *profesjonell utvikling og støtte (E6)* dreier seg om utfordringer rundt støtte fra både kollegaer og ledelsen. Eksempel på dette er skoler som er dårlige på å støtte lærerne i deres implementeringsarbeid. Det kan også innebære skoler som nedprioriterer å sende lærere som ønsker å utvikle seg på kurs.

3. Metode

3.1 Forskningsdesign

3.1.1 Kvalitativt design VS Kvalitativt design

Kvalitativ datainnsamlingsmetode og kvalitativ datainnsamlingsmetode er begge metoder for å samle empiri. Selv om begge er i stand til å samle inn empiri, har de ulike egenskaper og har ulike bruksområder. Derfor er det ikke uvanlig at disse forskningsmetodene kombineres. Når en forsker skal velge en av disse forskningsmetodene er det viktig å reflektere over hvilke metode som egner seg best med tanke på problemstillingen (Postholm & Jacobsen, 2018, s. 110).

Det kvalitative forskningsdesignet

En kvalitativ datainnsamlingsmetode henter inn informasjon om virkeligheten basert på ord og språk (Postholm & Jacobsen, 2018, s. 89). Denne metoden er ofte brukt om en skal ha fokus på virkeligheten og meningsskaping. Kvalitativ datainnsamlingsmetoder består av strategiene observasjon, intervju eller en kombinasjon av disse (Postholm & Jacobsen, 2018, s. 113). Observasjon handler om oppfatning og forståelse og ikke bare det å se. Det vil si at denne strategien kan bære store preg av forskerens subjektivitet. Deltakernes mening har heller ingen innflytelse og det er kun forskeren som skal analysere og tolke det som skjer. Ved å velge en kombinasjon av observasjon og intervju, kan deltakernes meninger også komme til syne. På denne måten vil forskerens subjektivitet og tolkninger ikke stå alene. En slik kombinasjon kan virke støttende for hverandre. Observasjon kan støtte opp det kommende intervjuet med å gi et bedre grunnlag for informasjonen som dukker opp. Intervjuet kan også støtte opp den kommende observasjonen med at forskeren sitter inne med viktig informasjon som kan ha innflytelse på det som skal observeres (Postholm & Jacobsen, 2018, s. 114-115).

Ved å bruke intervju som strategi er fokuset på utviklingen av kunnskapen om et tema. Et intervju kan holdes både fysisk, digitalt, muntlig og skriftlig (Postholm & Jacobsen, 2018, s. 117). Postholm & Jacobsen (2018, s. 118 -119) presenterer ulike former for intervju. Det fenomenologiske intervjuet tar utgangspunkt i personers erfaringer og opplevelse. Da er både muntlig informasjon og kroppsspråk data som skal samles inn. Det vil da si en kombinasjon av observasjon og intervju. Med et slikt intervju er det dermed viktig at informanten har erfaringer med det problemstillingen er ute etter. I et slikt intervju deltar ofte 3-10 deltakere eller 5 – 25 deltakere og at deltakerne er valgt ut ifra de samme kriteriene. Da denne formen

for intervju skal ha fokus på å samle inn erfaringer med fenomen og hendelser er det viktig at intervjuer stiller både hva og hvorfor spørsmål. Dette for å kunne samle inn nok informasjon til å danne et bilde over det informanten har erfart. Det narrative intervjuet fokuserer på å dokumentere detaljerte historier og erfaringer. Det kan komme fra enkeltindivid eller liten gruppe. Datamaterialet kan bestå av disse fortellingene i form av transkripsjoner eller tekst sendt fra informantene. I en slik form for intervju er det informanten som styrer intervjuet. Det er da viktig at forsker fokuserer på å bygge tillitten fra informantene slik at de føler seg komfortable til å snakke åpent. Igjennom hele intervjuet må forskeren arbeide med å lytte, engasjere og oppmuntre informantene i deres fortellinger. Postholm & Jacobsen (2018, s. 120-121) presenteres ulike strukturer for intervju. Det strukturerte intervjuet handler om å bruke samme intervjuguide i alle intervjuene. Intervjudeltakerne har ingen innflytelse på intervjuets gang. Forskeren skal holde seg nøytral og ikke stille improviserte spørsmål. Intervjuets gang kan sammenlignes med at forskeren snakker ut ifra et manus. Det ustrukturerte intervjuet er det motsatte fra det strukturerte med at forskeren ikke stiller med planlagte spørsmål. Fokuset til forskeren er å lytte til det informanten kommer med av informasjon. Ofte er det denne formen for intervju som blir kombinert med observasjon, da intervjuet kan bestå av lærere som forteller om undervisningspraksis. Det semi-strukturerte intervjuet handler om kunnskapen bygget av både forsker og informant. Det vil si at forskeren stiller med forslag til spørsmål, men spørsmålene trenger ikke å bli brukt. Intervjuet er åpent for at informanten kan komme med informasjon som forskeren ikke hadde tenkt over på forhånd. På lik linje kan forskeren også stille improviserte spørsmål. Hele intervjuet er preget av å følge flyten og stille spørsmål som føles naturlig i settingen. Slik form for intervju blir ofte benyttet i fenomenologisk og narrativt intervju.

Det kvantitative forskningsdesignet

En kvantitativ metode henter inn informasjon om virkeligheten basert på antall svar (Postholm & Jacobsen, 2018, s. 89). Med en slik metode har en mulighet til å få en oversikt over hva mange mener om et tema. Funnene i studien vil da kunne representere virkeligheten bedre enn i en kvalitativ studie med få informanter. Denne datainnsamlingsstrategien innebærer at ønsket informasjon er forhåndsbestemt av forskeren (Postholm & Jacobsen, 2018, s. 165). Derfor krever en kvantitativ metode ofte mer forarbeid enn en kvalitativ metode. Det vil si at om en skulle brukt spørreundersøkelse med lukket svaralternativ, må forskeren bearbeide spørsmål og svaralternativene grundig. Er det feil i studien etter den er lansert er det vanskelig å rette opp i disse i etterkant (Postholm & Jacobsen, 2018, s. 166).

Hovedformen for datainnsamlingsstrategier i en kvantitativ studie er spørreundersøkelse. Spørreundersøkelsen kan være i form av web-basert, sendt privat til epost, telefon intervju og personlig intervju. Spørreundersøkelser som er web-basert, er enklere å få flere informanter og et bredere spekter av kandidater. Dette da lenke til spørreundersøkelse kan bli lagt ut på sosiale medier og spre seg videre. Telefon intervju og personlig intervju skiller seg fra det kvalitative med at det fortsatt er en spørreundersøkelse. Informantene får svaralternativ og kan ikke komme andre svar enn det som er oppgitt (Postholm & Jacobsen, 2018, s. 184 -185).

Forskningsdesign for denne studien

Det kunne vært interessant å benytte kvantitativmetode for å hente inn data da forskningsspørsmål baserer seg på norske lærere. Med en slik metode kan en få god spredning og god dekning, som da kan gi en god representasjon av Norge. Allikevel fokuserer forskningsspørsmålet «*Hvilke utfordringer og suksessfaktorer opplever norske lærere med implementering av programmering i matematikk*?» på lærerens subjektivitet. Det er vil bli vanskelig å kunne forme et spørreskjema som kan fange opp alle lærernes forskjeller og opplevelser. Da lærere oppfatter ulike utfordringer og suksessfaktorer ville det også vært ønskelig med muligheten om å stille oppfølgingsspørsmål. Et Spørreskjema har ikke denne muligheten. En kvantitativ metode vil da ikke være gunstig til å fange opp informasjonen som kan svare på problemstillingen.

Den kvalitative strategien observasjon kan gi et bilde over utfordringene og suksessfaktorer som skjer i en kort periode. Det vil da være mulighet å fange opp utfordringer og suksessfaktorer som læreren selv ikke har lagt merke til. Allikevel vil ikke observasjon kunne gi et klart bilde og svar til problemstillingen. Det er begrensninger for hvilke utfordringer og suksessfaktorer som vil kunne dukke opp på en kort periode. Forskningsspørsmålet gir også rom for hele implementeringsprosessen siden 2020. Det vil si at erfaringene som læreren har bygget opp over tid ikke vil komme synlig frem ved observasjon. Dermed er intervju en strategi som kan være gunstig å få hente inn data som kan svare på problemstillingen. Av de ulike intervjuformene vil både det narrative og det fenomenologiske intervjuet være interessant. Da den aktuelle dataen har røtter i lærers erfaringer og følelser vil det fenomenologiske intervjuet virke mer gunstig. Av de ulike intervjuetformene vil ikke et strukturert intervju egne seg bra da det på lik linje som et kvantitativt spørreskjema skaper lite rom for at kandidaten kan komme med egne påstander. Et ustrukturert intervju kan også være utfordrende da informanten kan trenge litt hjelp til å vite hva de skal snakke om. Med et

semi-strukturert vil forskeren sitte klar med spørsmål om informanten skulle trenge det for å fortelle om egne erfaringer. En slik struktur vil gi informantene muligheten til å snakke fritt og forskeren kan stille passende spørsmål til det informanten sier.

Denne studien vil da være av et kvalitativt forskningsdesign hvor fokuset ligger på det fenomenologiske og intervjustrukturen er av et semi-strukturert intervju. Da fokuset ligger på detaljerte uttalelser istedenfor korte svar vil det bli utviklet en intervjuguid. Intervjuguiden er formet av to åpne spørsmål. Ett omhandler de indre utfordringer med programmering i matematikk og det andre omhandler de ytre utfordringer. I tillegg til de to hovedspørsmålene inneholder intervjuguiden tilleggsspørsmål som er knyttet opp til Vinnerviks (2022) modell for indre og ytre utfordringer (se vedlegg 1). Disse tilleggsspørsmålene skal kunne hjelpe som supplement, om det er noe lærerne ikke fikk sakt via de åpne spørsmålene. Intervjuet starter med at intervjuets forløp og intervjuguiden blir forklart. Lærerne vil ha intervjuguiden tilgjengelig framfor seg igjennom helle intervjuet slik at de selv kan støtte seg på det.

Det er viktig å ha fokus på en trygg og åpen atmosfære. Da forskningsspørsmålet er av et sårt tema for mange kan det være krevende og skummelt for en lærer å svare åpent på spørsmålene. Da er det viktig at intervjuer har fokus på å bygge en trygg atmosfære. For å få til dette må intervjueren virke lyttende og positiv til det lærerne sier, og motivere dem igjennom hele intervjuet. Da noen av utfordringene kan ha røtter i kollegiet og arbeidsplassen, forsikres informantene om at det som blir sagt er anonymt og at lydopptak vil bli slette etter transkriberingen. Transkriberingen vil heller ikke inneholde noe navn eller lokasjoner som skal kunne spores tilbake til dem. Målet med et kvalitativt intervju er å samle inn informasjon om hvilke utfordringer lærere har støtt på med implementeringen av programmering i matematikk og hvilke suksessfaktorer de har benyttet seg av.

3.1.2 Informanter

Å finne aktuelle kandidater som også er villige til å stille til et intervju er utfordrende. Dette da programmering i matematikk er utfordrende og skummelt for mange. Etter hvert ble det identifisert aktuelle kandidater ved å sende epost/ ringe til rektorer ved barne- og ungdomskoler. Skolene fikk en beskrivelse av hvilken bakgrunn som er ønsket av informantene og det ble funnet seks aktuelle lærere som selv ønsket å stille til intervju. Det ble gjennomført 4 intervjuer på 4 forskjellige skoler. To av intervjuene var individuelle intervju og de andre to var gruppeintervju. Av de seks intervjukandidater er det tre stykk som har bakgrunn fra 1-7 trinn og tre stykk som har bakgrunn fra 8-10 trinn. Tre av kandidatene

sier selv at de er utrygge med programmering i matematikk, mens de andre tre er mer trygge. Alle kandidatene unntatt lærer 5 har jobbet siden implementeringen av programmering i matematikk ved LK20.

Lærer 1

Denne læreren jobber på en 1-10 skole og har utdanning fra 1-7. Selv har hun erfaringer med mellomtrinnet og har jobbet siden før implementering av LK20. Hun har deltatt på kursing innenfor microbit og lego-Education. Selv om hun er litt usikker på sine egne programmeringskunnskaper er hun positivt innstilt til programmering.

Lærer 2

Denne læreren jobber på en ungdomsskole 8-10 trinn og har selv utdanning fra 5-10 trinn. Hun har nylig byttet jobb og snakker med erfaringer fra begge ungdomsskolene. Hun har også jobbet fra før implementeringen av LK20. Hun har vært på et programmeringskurs for en del år siden og skal på nytt kurs i slutten av dette året. Selv om hun er usikker og utrygg i programmering men er positivt innstilt til å lære seg programmering da det er en del av framtiden.

Lærer 3

Denne læreren jobber på en 1-10 skole sammen med lærer 4. Hun har utdanning fra 1-7 og har jobbet siden før implementeringen av LK20. Hennes arbeid innebærer både matematikklærer og en rolle i ledelsen. Dette gjør at hun har et annet synspunkt en de andre informantene. Hun har kun et kurs i fra første året med programmering i matematikk hvor kommunen sendte hun og klassen sammen på et programmeringskurs via vitensenteret. Selv om hun er utrygg og ikke ha mye kunnskap innenfor programmering, er hun fortsatt positivt innstilt til programmeringen.

Lærer 4

Denne læreren jobber på samme barneskole som lærer 3 på en 1-10 skole. Han har selv utdanning fra 1-7 og har i nyere tid tatt videreutdanning til master. Han har jobbet som lærer fra før implementering av LK20. I forbindelse med etterutdanningen har han vært med på egne programmeringskurs og synes selv han har god kunnskap med programmering. Han er positivt innstilt til programmering og synes det er gøy.

Lærer 5

Denne læreren jobber sammen med lærer 6 på en ungdomsskole. Tidligere har hun jobbet på en barneskole. Hun har ikke jobbet igjennom hele implementeringsprosessen av Lk20. Utdanningen hennes består av matematikk og pedagogikk istedenfor den tradisjonelle lærerutdanningen. I forbindelse med matematikkutdanningen har hun jobbet mye med Python. Generelt er hun en av de mer trygge lærerne, men hun er fortsatt usikker når det kommer til hvordan hun kan implementere programmering i matematikk. Hun er en av de få informantene som trekker Python inn i programmering.

Lærer 6

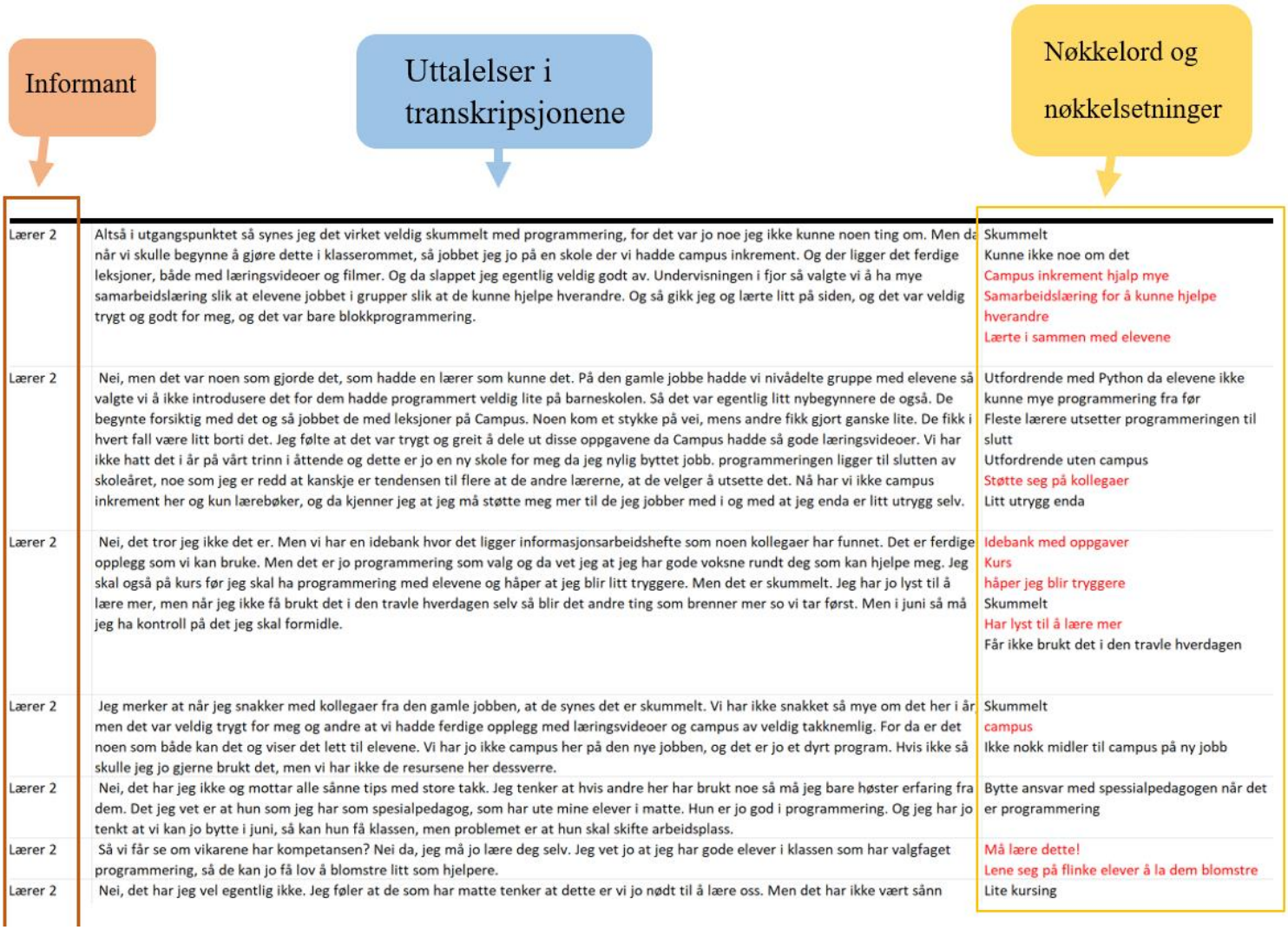
Denne læreren jobber på en ungdomsskole sammen med lærer 5 og har jobbet der side han ble ferdig utdannet. Da han har jobbet siden startfasen av implementering av Lk20, har han erfaring av nesten hele implementeringsprosessen av programmering i matematikk. Han har ingen kursing i programmering, men har brukt fellestiden på skolen til å jobbe med kompetansepakken fra Udir. Selv om han er trygg, er han også litt usikker.

3.2 Dataanalyse

Metodene benyttet i dette prosjektet er delt inn i tre deler. Først skal det skje en teoridreven innholdsanalyse som er delt inn i del I og del II, deretter skal det skje en summativ innholdsanalyse for å supplere funnene. Begrepene for metodene er tatt fra Fauskanger & Mosvold (2014), men forklaringen av analysemetodene er basert på Hsieh & Shannon (2005). Helt til slutt skal de indre utfordringene og suksessfaktorene sammenlignes med hverandre, deretter de ytre utfordringene og suksessfaktorene.

3.2.1 Del I – Teoridreven Innholdsanalyse, trinn 1

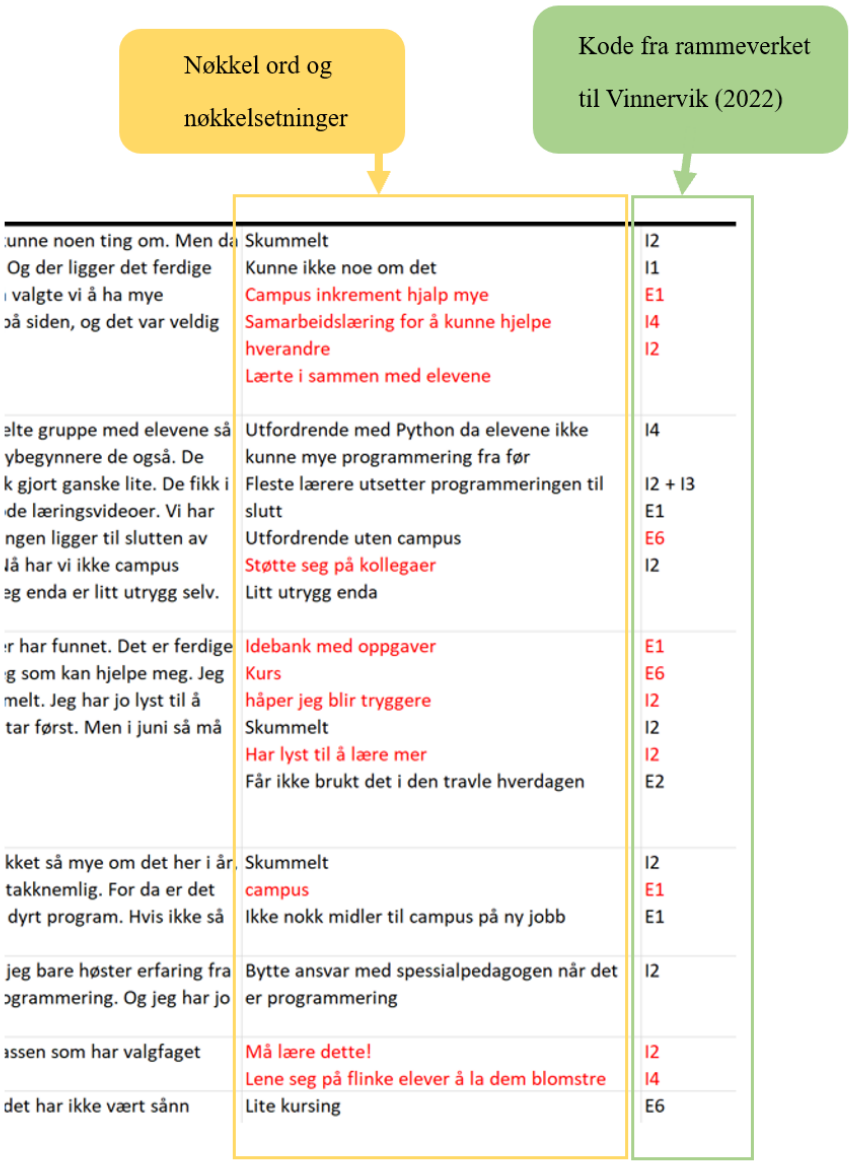
En teoridreven innholdsanalyse kan gjennomføres på to måter ifølge Hsieh & Shannon (2005). Den første metoden vil gå inn i datamaterialet for å først fremheve ønsket data, for å deretter sortere det som er fremhevet inn i kodene fra teorien. Metode to vil sortere data inn i koder fra teorien uten å fremheve. I dette prosjektet skal metode en brukes. Dette for å gjøre det lettere å kode, og generelt forsikre seg om at analysen via rammeverket blir riktig. Første gjennomgangen av datamaterialet vil relative nøkkelord eller nøkkelsetninger fremheves. Nøkkelordene eller nøkkelsetninger fanger essensen i utfordringen eller suksessfaktorene lærerne nevner. Disse nøkkelordene eller nøkkelsetningene blir notert inn i en kolonne på høyre side av utsagnene fra lærerne. Svart farge indikere utfordring og rød farge indikerer suksessfaktor.



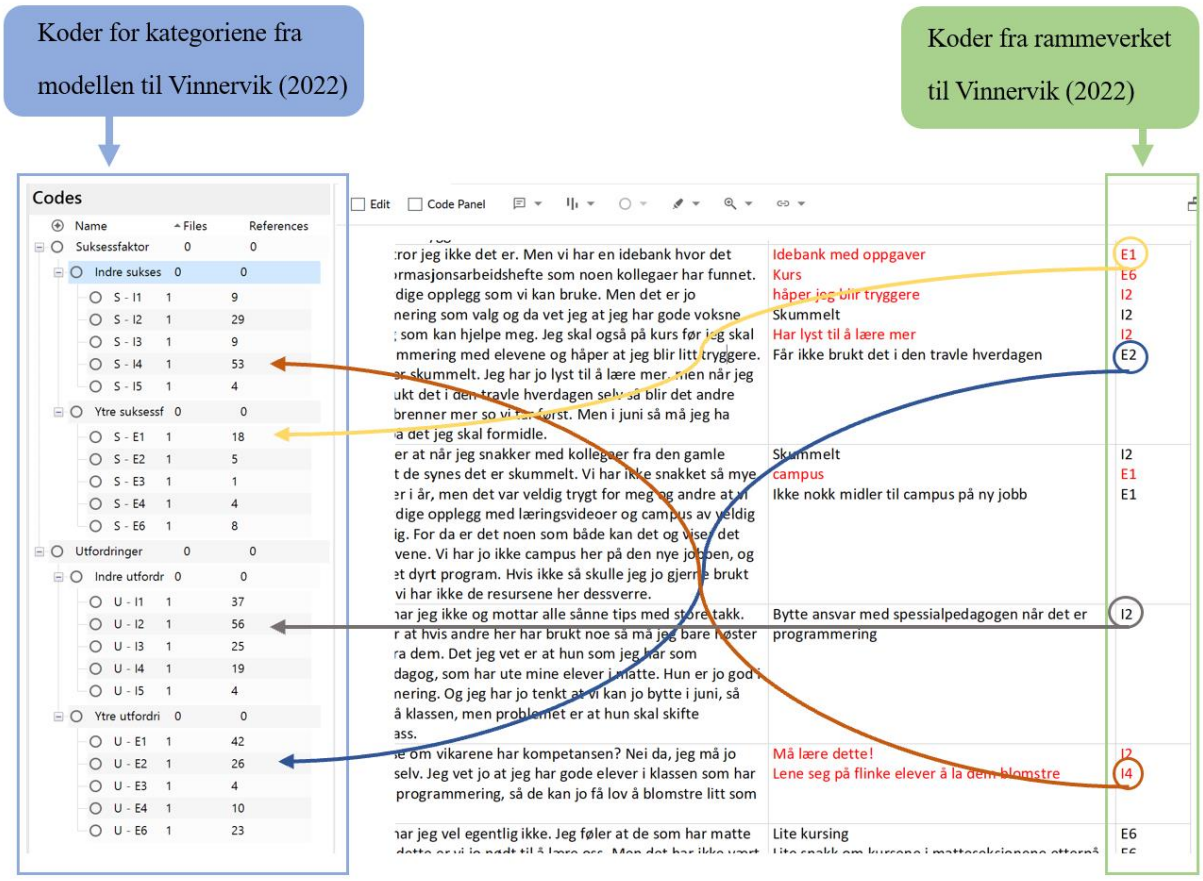
Lærer 2	Altså i utgangspunktet så synes jeg det virket veldig skummelt med programmering, for det var jo noe jeg ikke kunne noen ting om. Men da når vi skulle begynne å gjøre dette i klasserommet, så jobbet jeg jo på en skole der vi hadde campus inkrement. Og der ligger det ferdige leksjoner, både med læringsvideoer og filmer. Og da slappet jeg egentlig veldig godt av. Undervisningen i fjor så valgte vi å ha mye samarbeidslæring slik at elevene jobbet i grupper slik at de kunne hjelpe hverandre. Og så gikk jeg og lærte litt på siden, og det var veldig trygt og godt for meg, og det var bare blokkprogrammering.	Skummelt Kunne ikke noe om det Campus inkrement hjalp mye Samarbeidslæring for å kunne hjelpe hverandre Lærte i sammen med elevene
Lærer 2	Nei, men det var noen som gjorde det, som hadde en lærer som kunne det. På den gamle jobbe hadde vi nivådelte gruppe med elevene så valgte vi å ikke introdusere det for dem hadde programmert veldig lite på barneskolen. Så det var egentlig litt nybegynnere de også. De begynte forsiktig med det og så jobbet de med leksjoner på Campus. Noen kom et stykke på vei, mens andre fikk gjort ganske lite. De fikk i hvert fall være litt borti det. Jeg følte at det var trygt og greit å dele ut disse oppgavene da Campus hadde så gode læringsvideoer. Vi har ikke hatt det i år på vårt trinn i åttende og dette er jo en ny skole for meg da jeg nylig byttet jobb. programmeringen ligger til slutten av skoleåret, noe som jeg er redd at kanskje er tendensen til flere at de andre lærerne, at de velger å utsette det. Nå har vi ikke campus inkrement her og kun lærebøker, og da kjenner jeg at jeg må støtte meg mer til de jeg jobber med i og med at jeg enda er litt utrygg selv.	Utfordrende med Python da elevene ikke kunne mye programmering fra før Fleste lærere utsetter programmeringen til slutt Utfordrende uten campus Støtte seg på kollegaer Litt utrygg enda
Lærer 2	Nei, det tror jeg ikke det er. Men vi har en idebank hvor det ligger informasjonsarbeidshefte som noen kollegaer har funnet. Det er ferdige opplegg som vi kan bruke. Men det er jo programmering som valg og da vet jeg at jeg har gode voksne rundt deg som kan hjelpe meg. Jeg skal også på kurs før jeg skal ha programmering med elevene og håper at jeg blir litt tryggere. Men det er skummelt. Jeg har jo lyst til å lære mer, men når jeg ikke få brukt det i den travle hverdagen selv så blir det andre ting som brenner mer so vi tar først. Men i juni så må jeg ha kontroll på det jeg skal formidle.	Idebank med oppgaver Kurs håper jeg blir tryggere Skummelt Har lyst til å lære mer Får ikke brukt det i den travle hverdagen
Lærer 2	Jeg merker at når jeg snakker med kollegaer fra den gamle jobben, at de synes det er skummelt. Vi har ikke snakket så mye om det her i år men det var veldig trygt for meg og andre at vi hadde ferdige opplegg med læringsvideoer og campus av veldig takknemlig. For da er det noen som både kan det og viser det lett til elevene. Vi har jo ikke campus her på den nye jobben, og det er jo et dyrt program. Hvis ikke så skulle jeg jo gjerne brukt det, men vi har ikke de resursene her dessverre.	Skummelt campus Ikke nokk midler til campus på ny jobb
Lærer 2	Nei, det har jeg ikke og mottar alle sånne tips med store takk. Jeg tenker at hvis andre her har brukt noe så må jeg bare høster erfaring fra dem. Det jeg vet er at hun som jeg har som spesialpedagog, som har ute mine elever i matte. Hun er jo god i programmering. Og jeg har jo tenkt at vi kan jo bytte i juni, så kan hun få klassen, men problemet er at hun skal skifte arbeidsplass.	Bytte ansvar med spessialpedagogen når det er programmering
Lærer 2	Så vi får se om vikarene har kompetansen? Nei da, jeg må jo lære deg selv. Jeg vet jo at jeg har gode elever i klassen som har valgfaget programmering, så de kan jo få lov å blomstre litt som hjelpere.	Må lære dette! Lene seg på flinke elever å la dem blomstre
Lærer 2	Nei, det har jeg vel egentlig ikke. Jeg føler at de som har matte tenker at dette er vi jo nødt til å lære oss. Men det har ikke vært sånn	Lite kursing

3.2.2 Del II - Teoridreven innholdsanalyse, trinn 2

Andre gjennomgangen vil alle nøkkelordene og nøkkelsetningene få en tilhørende kode fra Vinnersviks (2022) rammeverk. De etablerte kodene fra del I vil så bli videre sortert inn i Vinnerviks (2022) kategorier for indre og ytre utfordringer (se vedlegg 1). Kodene kan bli satt inn i en eller flere kategorier, alt etter hvor de hører til. Dette vil først skje manuelt hvor kodene vil stå i en egen kolonne på høyreside av nøkkelordene. For å indikere hvilket nøkkelord og nøkkelsetning som koden henger sammen, vil de bli notert inn på samme linje. Er koden svart indikerer det utfordring, er den rød indikerer det en suksessfaktor. Dette er for å kvalitetssikre at nøkkelordene og setningene havner inn i rett kode i Nvivo, men også for å sikre at datamaterialet ikke er tilfeldig plassert inn i koder alt etter dagsform.

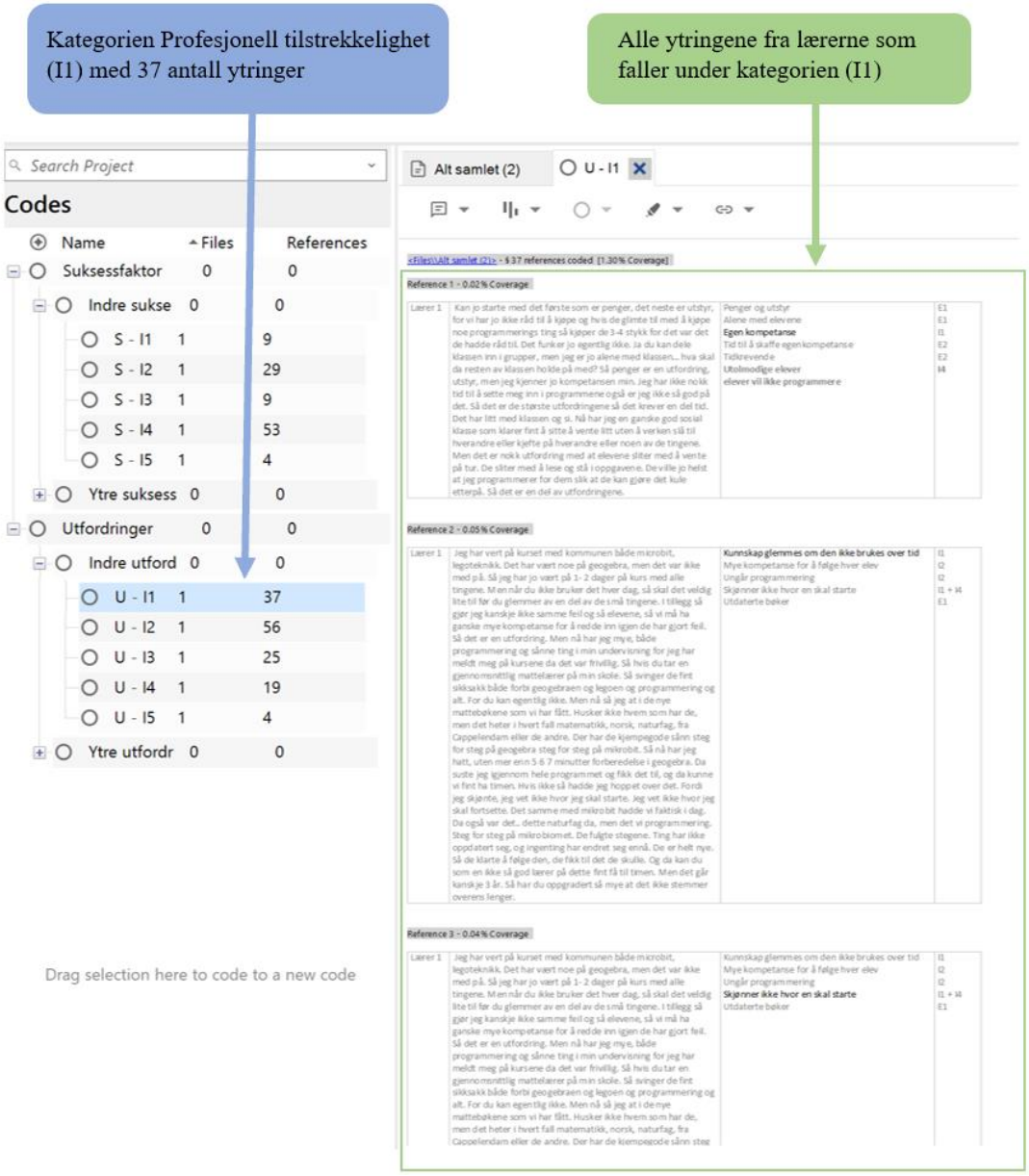


Etter dette er det klart til å bli satt inn i riktig kategori i programmet Nvivo. I programmet ligger koder for kategoriene til Vinnerviks (2022) modell for indre og ytre utfordringer klar til å inneholde nøkkelordene eller nøkkelsetningene fra del I. Sett til venstre ligger kodene for utfordringer og suksessfaktorer klar. Hver utfordring og suksessfaktor vil bli plassert i en eller flere av kodene. Røde nøkkelord eller nøkkelsetninger vil bli sortert inn i kodene for som representerer suksessfaktorer. De svarte nøkkelordene eller nøkkelsetningene vil bli sortert inn i kodene som representerer utfordringer. Fra del I er materialet allerede plassert inn i de forskjellige kategoriene til vinnervik (2022) og er da klar til å bli satt rett inn i tilhørende kode i NVIVO. Dette gir en mulighet for å kunne rette opp feil eller reflektere enda en gang over hvilke kategorier utfordringen eller suksessfaktoren hører til.



Etter å ha satt alle kodene inn i NVIVO sine koder for kategoriene til Vinnerviks (2022) rammeverk, kan en få en samlende oversikt. Hver av NVIVO sine koder vil kunne gi en oversikt over alle utfordringene som lærerne nevner igjennom datamaterialene. Ved å trykke innpå en av disse kodene, vil det dukke opp et vindu med en oversikt over alle utfordringene

eller suksessfaktorene som hører til. Dette vil bli brukt i resultatdelen og diskusjonsdelen. Ettersom noen utfordringer har røtter i flere koder, vil de noen ganger bli presentert i flere av vinnerviks (2022) kategorier i resultatdelen. En av kategoriene i resultatdelen vil som oftest gå mer i dybden på den gitte utfordringen enn det som vil bli presentert i de eventuelt andre kategoriene.



3.2.3 Del III - Summativ innholdsanalyse

Summativ innholdsanalyse løfter fokuset på antall ganger ord dukker opp i et datamateriale. Det som gjør at summativ innholdsanalyse er en kvalitativ metode og ikke kvantitativ, er fokuset på konteksten rundt ordene og ikke bare antall ganger ordet blir nevnt (Hsieh & Shannon, 2005). Slik den summativ innholdsanalyse blir utført i denne studien, ville den passet i både en kvalitativ og kvantitativ design. Det er fordi denne studien fokuserer på antall ord da dette kan supplere funnene. Lærerne nevner mye følelser og problematikk rundt programmering i matematikk. Det vil da være interessant å se hvilke ord som dukker opp, og hva det sier om programmering i matematikk. Det skal derfor bli gjennomført en summativ innholdsanalyse som skal kunne hjelpe i å berike analysen og funnene. Programmet Nvivo har muligheten til å telle ordene i transkripsjonen, og grupperer stammende ord. Ettersom programmet ikke er norsk, er det også begrensninger for hvilket nivå den klarer å gruppere ordene. Det kan dukke opp ord i grupperingen som ikke burde være en del av den. Programmet har heller ingen mulighet for å filtrere ut bindende ord. Da ordet «tid» er det eneste viktige ordet som dukker opp når programmet leter etter ord på tre eller flere bokstaver, ble løsningen å la programmet lete etter ord med fire eller mer bokstaver.

3.3 Forskningsetiske betraktninger

3.3.1 Etiske overveielser

Siden dataene kommer fra intervju av lærere, er det viktig å anonymisere kandidatene. Under transkriberingsprosessen må det sikres at lærernes, skolenes og andre detaljer som kan spores tilbake til intervjuobjektene, ikke kan spores opp. Det må sendes inn en søknad til SIKT for å få tillatelse til å behandle lydopptak. Denne søknaden må bli godkjent (se vedlegg 3) før informantenes informasjon kan lagres på en riktig måte.

Det er viktig at informantenes personlige data blir lagret på riktig måte. Dette innebærer at alle transkripsjonene ikke inneholder noe personlig informasjon. Alle navn skal bli erstattet med fiktive navn. I dette tilfellet blir navnet lærer 1 – 6. Alle stedsnavn skal byttes ut med ikke gjenkjennbare navn og det samme med navn på skoler, elever og kollegaer. Det skal ikke være noen informasjon som skal kunne spores tilbake til informanten og de rundt seg. Lagring av lydopptak skal skje i en egen kryptertsky, Nexcloud. Dette er av sikkerhetsmessige årsaker, da pc og telefon kan bli hacket. I en kryptertsky ligger dataen

sikrere og dermed mer beskyttet. Det er ingen andre som skal ha tilgangen til den nettskyen og all informasjon om informantene slettes etter prosjektet er slutt.

Det er viktig at lærerne blir opplyst om sine rettigheter innenfor dette prosjektet. Informanten skal få muntlig beskjed om hva studien innebærer, hvordan opplysningene deres blir lagret og deres rettigheter til å trekke seg når som helst. Bestemmer de seg for å trekke seg i løpet av prosjektet, skal all informasjon og data som er lagret av dem slettes. Informantene skal også få utdelt et informasjonsskriv og samtykke skjema (se vedlegg 4) som skal underskrives før intervjuet kan starte. Med dette vil de samtykke at deres data blir lagret så lenge prosjektet varer og at de er kjent med deres rettigheter.

3.3.2 Validitet og reliabilitet

Postholm & Jacobsen (2018, s. 222 - 223) løfter frem hvordan indre validitet, ytre validitet og reliabilitet danner studiens samlende troverdighet. Forskere som overholder disse faktorene og forklarer tiltakene for å sikre kvaliteten i forskningsprosessen, vil troverdigheten til studien bedres. For at en forsker skal kunne reflektere over studiens samlende troverdighet presenterer de to spørsmål. Spørsmål 1 omhandler studiets validitet og spørsmål 2 omhandler studiens relabilitet. Spørsmål 1 gjør at forskeren må reflektere over begrensninger med egen studie. *«Hvilke begrensninger som er knyttet opp til egen forskning»* (Postholm & Jacobsen, 2018, s. 222). Videre deler de validitet inn i indre og ytre validitet. Indre validitet handler om dataen som er samlet inn i sammenheng med denne studien er det samme som det en sier en har funnet (Postholm & Jacobsen, 2018, s. 222 - 223). Det handler om hvor godt virkeligheten samsvarer med det en studerer, og hvilket ståsted en har for å komme med uttalelser om studien (Postholm & Jacobsen, 2018, s. 229). Ytre validitet handler om hvor stor grad resultatene kan reproduseres. Hvor mye av resultatene samsvarer om studien blir reprodusert under andre omstendigheter (Postholm & Jacobsen, 2018, s. 222-223). Kan studien bli reprodusert på en annen skole? Et annet sted i landet eller til og med i et annet land? (Postholm & Jacobsen, 2018, s. 238). Spørsmål 2 gjør at forskeren må reflektere over hvordan forskningsmetoden kan ha påvirket resultatene i studien. *«Hvordan han eller hun gjennomgår sin måte å gjennomføre forskning på kan ha påvirket de endelige resultatene»* (Postholm & Jacobsen, 2018, s. 222). Er det mulig å at en annen forsker reproduserer studien på et annet tidspunkt? De nevner at en det er vanskelig å reprodusere en kvalitativ studie, da det er mange varierende faktorer. Med dette menes at hver person alltid er i endring og forskeren sitter inne med egne verdier og teorier (Postholm & Jacobsen, 2018, s. 223-224). Forskeren må reflektere over hvilket forhold informantene har for problemstillingen. Det må

reflekteres over hvilken kontekst studien er satt i. Har tid og rom noe å si for studien og datasamlingen? Det må også reflekteres over hvem som ikke deltar i forskningen som kunne ha kommet med interessante synspunkt i forhold til forskningsspørsmålet. Også må det reflekteres over om alt det viktige har blitt registrert (Postholm & Jacobsen, 2018, s. 225 - 228)

Postholm & Jacobsen (2018, s. 263) kaller det å inkludere flere datakilder for triangulering. Målet er å kunne beskrive virkeligheten fra ulike sider. I dette prosjektet betyr det at informantene ikke kun representerer matematikklærere, men også noen fra ledelsen. En av informantene i denne studien har en rolle som både matematikklærer og en rolle i ledelsen. Dette vil kunne gi en innsikt av problemstillingen sett fra ledelsesperspektivet fra en skole, og ikke kun igjennom øynene fra en lærer. På denne måten vil en få en bredere innsikt til virkeligheten.

Utfordringen med intern validitet er seleksjonsskjevhet. Informantene kan enten over eller underrepresenterer kunnskapsnivået til de generelle lærerne innenfor programmering i matematikk. Optimalt ville da vært at halvparten av informantene var de som overrepresenterte og resten var av de som underrepresenterer. På denne måten vil genomsnittet kunne representerer den generelle læreren. Dette er dermed utfordrende å vite på forhånd av et intervju, da en ikke vet hvor grensen er når det kommer til overrepresentering eller underrepresentering. Målet er da å finne informanter hvor halvparten er mer trygge innenfor programmering i matematikk og den resterende halvparten representerer de som er mindre trygge. På denne måten får en informasjon fra begge sider, og kan treffe bredere på hvilke utfordringer og suksessfaktorer lærere har møtt på i Norge. Det er dermed viktig at informantene er åpne om det de mener. Det kan hende at informantene svarer det de tror en som forsker vil hører og det er ikke optimalt for studien. For å unngå dette blir det presisert at alle informanter er verdifulle uansett kunnskapsnivå. Noen informanter er utrygge i deres egen kunnskap og er disse som kan være i faresonen for å pynte på deres opplevelse. Det er da viktig å betrygge disse informantene og støtte dem slik at de kommer med de reelle svarene.

Når det kommer til ekstern validitet, er utfordringen om studien kan gjenskapes. Kan studien generalisere de samme resultatene i en større studie med langt flere informanter? Flere av lærerne i Norge er satt i lik situasjon etter implementeringen av programmering via LK20. Det er lett å tenke at geografiske forskjeller ikke burde spille mye inn på resultatene da

informantene i denne studien allerede representerer flere skoler. Allikevel er det faktorer som gjør at det kan spille en større faktor på resultatene allikevel. Det er ikke alle skoler som for eksempel har muligheten til at alle elevene har egne digitale enheter. Dette vil kunne stille som en stor utfordring som ikke vil være relevant på alle skoler. Med andre ord kan geografiske forskjeller føre til ulike utfordringer og suksessfaktorer. Utfordringen med studiens problemstilling og dens reproduserbare, er at den har kjerne i læreres opplevelser med utfordringer og suksessfaktorer med programmering i matematikk. Det vil si at problemstillingen er subjektiv for lærerne. Selv om en skulle ha klart å finne lærere med lik bakgrunn og bevare det samme forholde av diversitet, vil det bli utfordrende å generere de samme resultatene. Det er mye av en lærers personlighet som får rom i en slik studie da den tar utgangspunkt i lærers subjektivitet. Det vil ikke dermed si at resultatene vil variere mye. Flere av informantene i denne studien opplever mye av de samme utfordringene, og har også noen felles suksessfaktorer. Det vil da kunne si at i en større studie kan resultatene kunne samsvare, men at en må ta i betraktning at en informants subjektivitet vil kunne føre til litt variasjon.

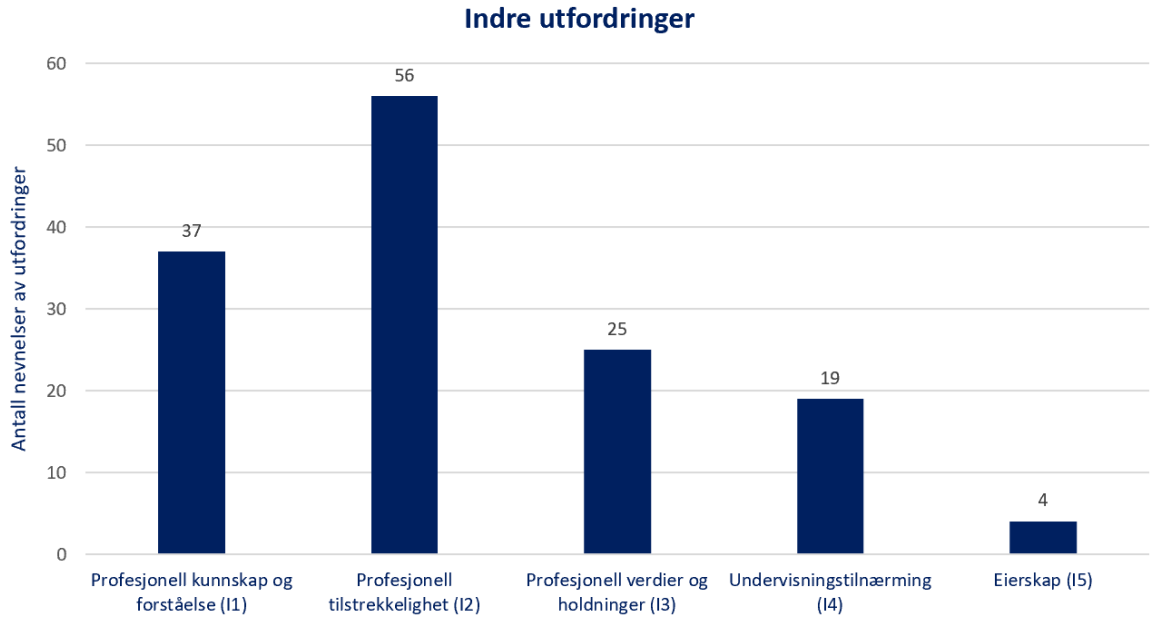
Datamaterialet blir analysert to ganger på forskjellige dager. Kodene i Nvivo vil også bli sjekket for feil en annen dag en dette igjen. Resultatdelen blir skreven på et senere tidspunkt. Med andre ord vil datamaterialet blir gjennomgått flere ganger på ulike tidspunkt og i ulike settinger. Dette gjør at resultatene og funnene er mer troverdige da de er mer konsistente og reproduserbare. Postholm & Jacobsen (2018, s. 224) mener derimot at reprodusering av studiet ikke har noen sammenheng med pålitelighet. De mener pålitelighet handler mer om forskerens og undersøkelsens påvirkningskraft til resultatene. Denne studiens kjerne handler om å identifisere og plassere læreres utfordringer og suksessfaktorer med programmering i matematikk inn i modellen til Vinnervik (2022) (se vedlegg 1). Dette gir rom for egen tolkning på hvor de ulike utfordringene og suksessfaktorene hører til i kategoriene i modellen til Vinnervik (2022). Det vil si at en annen person kan plassere utfordringene og suksessfaktorene i forskjellige kategorier. Spesielt når utfordringer og suksessfaktorer ofte hører til i flere kategorier. Da vil det kunne oppstå ulik tolkning og meninger om hvilke kategori utfordringene og suksessfaktorene hører mest til. For å kunne ha lik forståelse for hvor utfordringene og suksessfaktorene hører til, må en starte med modellen til Vinnervik (2022) og teorien bak. En gjennomgang av Finger & Houguet (2009) litteraturstudie, som danner grunnlaget for modellen til Vinnervik (2022), kan gi et bedre grep på hva de ulike kategoriene til Vinnerviks (2022) modell rommer. I tillegg til dette blir det tatt en

gjennomgang av lignende studier som har basert seg på samme modell, og sett på deres tolkning av kategorienes innhold.

4. Resultater

4.1 Indre utfordringer

Indre Utfordringer	Hvor mange av lærerne nevner utfordringer	Antall ytringer (utfordringer)
Profesjonell kunnskap og forståelse (I1)	1,2,3,4,5,6	37
Profesjonell tilstrekkelighet (I2)	1,2,3,4,5,6	56
Profesjonell verdier og holdninger (I3)	1,2,3,4,5,6	25
Undervisningstilnærming (I4)	1,2,3,5,6	19
Eierskap (I5)	1,4,5,6	4



4.1.1 Profesjonell kunnskap og forståelse

Av alle de indre utfordringene er det denne kategorien som har nest mest antall ytringer, kun forbigått av Profesjonell tilstrekkelighet (I2). Det indikerer at denne kategorien, fra de indre utfordringene, kan oppleves som mer vanskelig for lærerne.

Hvordan implementere programmering i matematikk?

Alle seks lærerne nevner at de synes det er utfordrende å vite hvordan de skal bygge opp et godt undervisningsforløp med programmering i matematikk. Det å vite hva elevene bør kunne for å utføre ulike programmeringsoppgaver er utfordrende. Hva er det grunnleggende? Hva bør de lære først? Når er elevene klare for diverse ulike utfordringer?

> Lærer 1: Fordi jeg skjønte, jeg vet ikke hvor jeg skal starte. Jeg vet ikke hvor jeg skal fortsette.

Lærer 1, 2 og 3 velger å implementere programmering som et egent emne. Det vil si at programmeringen blir likestilt med de andre emnene som for eksempel geometri. Lærer 4, 5 og 6 velger derimot å implementere programmering som en del av matematikken hvor de prøver å relatere programmeringen til alle de eksisterende emnene. I forbindelse med dette løfter alle lærerne opp utfordringen med tid. Dette blir tatt opp igjen i kategorien Tidshåndtering (E2). Lærer 5 og 6 som jobber aktivt med å implementere programmering som en del i alle emnene, sier at de synes det er utfordrende å relatere programmering til alle emnene da det ikke kommer like naturlig. I motsetning til lærer 5 og 6 mener både Lære 1, 3 og 4 mener at Excel og Geogebra inngår i programmering. Allikevel synes de at Geogebra og Excel er mye enklere å implementere enn programmering.

> Lærer 5: En ting er å lære de ulike formene for programmering, men det å flette det inn slik at det blir en naturlig del slik som det er med Geogebra og Excel.

Det er kun lærer 5 som har hatt skrifts programmering i egen utdannelse. Selv om hun har god kunnskap i Python synes hun fortsatt det er krevende å løse matematiske oppgaver ved hjelp av programmering. Dette mener hun også gjenspeiles hos elevene og at det kun er de mer erfarne elevene som kan utfordres til å bruke skrifts programmering som et verktøy til å løse noe matematisk.

> Lærer 5: å utfordre elevene til å se om de kan bruke programmering til å løse en oppgave er for de mer erfarne og engasjerte. Det å få hele klassen med på noe sånt er vi ikke i nærheten av. Jeg skulle ønske det, men det er skikkelig vanskelig.

4.1.2 Profesjonell tilstrekkelighet

Den profesjonelle tilstrekkeligheten er en utfordring som lærerne kjenner mest på. Lærerne er usikre rundt det å ha programmering i matematikktimer. Er opplegget bra nok? Har jeg god nok kunnskap? Kan jeg gi det elevene trenger innenfor programmering i matematikk?

Programmering er skummelt

Alle lærerne nevner flere ganger at programmering er skummelt. Ordet *skummelt* dukker opp 15 ganger i transkripsjonene, som en av de mest nevnte ordene. Programmering er skummelt

da ingen av lærerne har vært borti programmering bortsett fra i skolesettingen. Fire av lærerne har kun erfaring fra kursing via skolen og to av dem har hatt programmering i egen utdannelse.

> Lærer 3: … Altså programmering er noe som er helt nytt, så det er jo ikke noe som har vært en del av min lærerutdanning. Jeg har hatt noe kurs gjennom vitensenteret, så utfordringen er å prøve å lære seg noe nytt på egen hånd, for å så kunne videreføre det.

Det er utfordrende å ta det første steget mot en programmeringstime og krevende å fortsette med de tunge stegene. Det er også utfordrende å måtte omforme egen kunnskap til noe lærbart for elevene, spesielt om en selv er usikker på din egen kunnskap. Selv om flere av lærerne sitter inne med mer kunnskap enn flere andre, kjenner de ikke nødvendigvis at de har kontroll på programmering som en del av matematikken. *Kontroll* er også et ord som dukker opp 19 ganger.

> Lærer 6: Du møter et tema som du skal undervise i noe som vertfall ikke jeg føler meg trygg nok til å si at sånn og sånn skal vi gjøre det.

Ordet *vanskelig* er det ordet som dukker opp mest med hele 28 ganger. Lærerne synes programmering er vanskelig. Alt fra kursing og til det å omforme sin egen kunnskap til noe som er lærbart. Alle seks lærerne synes programmering er krevende og at de selv ikke har god nok kunnskap. Dette nevnes også av de lærerne som virker som et støtteapparat for andre kollegaer. Lærer 3 nevner usikkerheten noen kan føle på når de ser at andre lykkes med programmering i matematikk. Det virker som en illusjon hvor en innbiller seg at de andre som bruker programmering er så mye flinkere. Dette gjør det ikke enklere for selvtilliten til denne læreren. Nevnt lærer har selv indre utfordringer i tillegg til å se hvordan de andre lærerne klarer seg så bra. Dette fører til utrygghet rundt hele programmeringen.

> Lærer 3: Du føler deg egentlig bare enda mer utrygg for du tror alle andre kan så mye mer.

Lærer 6 nevner at hun føler seg alene med sine utfordringer. Hun kjenner på utfordringen med å implementere programmering i matematikk på en god måte og føler seg alene i denne søken etter gode suksessfaktorer.

> Lærer 5: Jeg føler ikke jeg har gode nok kunnskap om hvordan en skal gjøre det. At jeg føler meg alene

4.1.3 Profesjonell verdier og holdninger

Nedprioritering og negativitet til programmering

De utfordringene som blir gjentatt av flere lærere er utfordringer med negative holdninger fra de eldre kollegaene. Både lærer 1, 5 og 6 rapporterer å ha opplevd negative holdninger fra disse lærerne. Lærer 5 og 6 snakker om denne utfordringen i forbindelse med oppstartsfasen av programmering i matematikk, men som har bedret seg i løpet av årene.

> Lærer 6: … jeg har opplevd før at enkelte ikke ser noe mening med å ta det inn i faget. Hvorfor skal det inn i faget? Hvilken nytte har det? En negativ holdning

Lærer 1 rapporterer at skolen hennes fortsatt har et miljø preget av negativ holdninger fra de eldre. Disse lærere unnviker å ha programmeringstimer så langt det lar seg gjøre.

> Lærer 2: På vår skole så kan man si det sånn vi har én som nærmer seg pensjon ... Han gidder ikke. Han har sagt rett ut, dette kommer ikke han til å gjøre.

Lærer 1-4 nevner en trend hvor mange velger å ta programmeringsdelen helt til slutt. Det kan være flere grunner til dette, som egen usikkerhet, holdninger eller ren uvitenhet. Det kan også være av taktiske grunner som fører til at programmering blir plassert som siste emne. Det kan hende at den tenkte årsplanen blir forskjøvet og da går det som oftest ut over det siste emnet. På denne måten viser noen lærere at de nedprioriterer programmering i forhold til de andre emnene i matematikk.

4.1.4 Undervisningstilnærming

Elevers motivasjon

Flere av lærerne nevner utfordringer som omhandler elever. Lærer 1, 2, 5 og 6 nevner utfordring rundt det å motivere elevene. De opplever at flere elever blir raskt umotivert og frustrert av programmering som gjør at de kan melde seg ut av timen.

> Lærer 1: … du ender jo opp med noen 1/5 av klassen som sitter og kjempegiret og prøver å feile og finne ut og kan jo hjelpe alle etter en liten stund. Så har du de som fant ja nå fikk jeg mulighet til å gjøre niks og nada. Så sitter enten og ser ut av vinduet eller finner nye internettsider, se på eller sitte og tegne eller lese. Så har du de som sitter og blir frustrert. Har du lyst å hive pc inn i

> veggen fordi de for eksempel ikke visste at det finnes en angreknapp eller alt forsvant, og alt var drit og feil farge her og feil strek der.

Jevnt over synes lærerne det er utfordrende å holde motet og humøret til elevene oppe. Lærer 5 og 6 nevner også problematikken med elever som trenger hjelp. Ofte må elevene sitte lenge å vente på veiledning, eller vente på at læreren feilsøker programmet. Dette opplever lærerne føre til demotiverte elever. Lærerne føler at elevene kobler lærerens tidsbruk til feilsøking sammen med at programmering er vanskelig og dette kan da videre påvirke elevenes motivasjon.

> Lærer 6: Da sliter det kanskje på motivasjon også. Hvor lenge orker dem å sitte å lete etter feilen som ikke funker før de går videre til neste eller bare skru av.

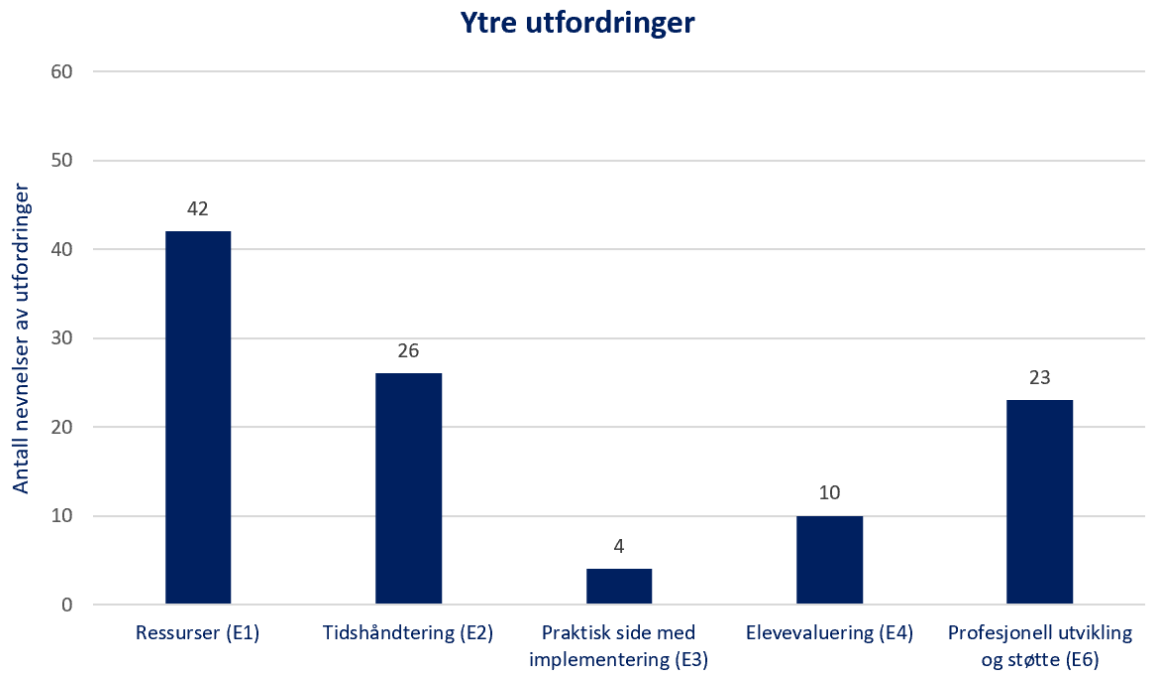
4.1.5 Eierskap

Generelt sett nevner lærerne lite utfordringer rettet spesifikt eierskap. Noen av lærerne nevner at eierskapet til programmering er mye mindre enn eierskapet til de andre emnene i matematikken. Lærer 4 er den eneste som føler litt eierskap, men allikevel ikke like god som med de andre emnene.

> Lærer 4: I forhold til en vanlig matte time hvor du lager noe fra bunnen av, fordi du har en stor idé om hvordan ting kan gjøres og bør gjøres, så føler ikke jeg at jeg har det (Eierskap) i programmering. Jeg er ikke så dyktig i programmering at jeg er på lik linje med en vanlig matte time.

4.2 ytre utfordringer

Ytre utfordringer	Hvor mange lærere nevner utfordringer	Antall ytinger (utfordringer)
Ressurser (E1)	1,2,3,4,5,6	42
Tidshåndtering (E2)	1,2,3,4,5,6	26
Praktisk side med implementering (E3)	5,6	4
Elevevaluering (E4)	1,3,5,6	10
Profesjonell utvikling og støtte (E6)	1,2,3,4,5,6	23



4.2.1 Ressurser

Av alle de ytre utfordringene som ble nevnt er det kategorien Ressurser (E1) som har flest ytringer fra lærere. Noe som indikerer at denne kategorien oppleves som mest utfordrende.

Penger og utstyr

Lærer 1,2,5 og 6 rapporterer at penger er en utfordring. Ikke alle skoler har penger til at det skal være en assistent med i timene. Det vil da si at mange lærere er alene med elevene i programmeringstimer. Mangel på penger fører også til en begrensing på hvilket utstyr skolen har, og i hvilket antall av hver sort. Lærer 1 nevner at det ikke er før nå at hun virkelig kan bruke utstyret aktivt i timene. Kommunen har nylig investerte mye penger for å få orden på programmeringsutstyret på skolen. Før dette satt hun med 3-4 stykk av hvert

programmeringsutstyr, noe som gjorde det utrolig vanskelig å bruke det i en klasse på 25-30 elever. Utfordringer med utstyret i seg selv er å ha kontroll på at det ikke er tatt i bruk en annen klasse når en selv trenger det. Det er også en utfordring med utstyr som ikke er komplett eller ikke virker som det skal.

Læreverk

Lærer 1 løfter frem utfordringen med læreverk som går ut på dato. Dette gjør det utfordrende å ha programmeringstimer om en ikke kan det så godt selv. Lærer 5 og 6 løfter frem utfordringen med å implementere programmering som en del av matematikken og ikke som et eget emne. Campus Inkrement og mange lærerbøker introduserer programmering som en egen bolk og fletter ikke programmeringen inn i de andre emnene. Skolestudio derimot fletter programmeringen inn i de andre emnene, men klarer ikke å gjøre det på en oppbyggende måte. Lærerne opplever at programmeringsoppgavene er for vanskelige og elevene ikke har forutsetningene som skal til for å løse disse oppgavene.

> Lærer 5: Og så har du skolestudio som gir programmeringsoppgaver underveis, føler jeg opplæringen mangler. Hvor har vi lært dette? Hvor kan du finne dette?

De samme lærerne synes også det er vanskelig å lære elevene tekst programmering da ingen av verktøyene har god støtte for dette. Skolestudio gjør det vanskelig med at en må forholde seg til flere program. Campus som inkluderer blokk programmering har ingen støtte for tekst programmering i det hele tatt.

4.2.2 Tidshåndtering

Alle seks lærerne nevner ordet *tidkrevende*. Igjennom alle transkripsjonene nevnes ordet spesifikt 15 ganger og ordet *tid* hele 46 ganger.

Planlegging

Flere av lærerne nevner at det er tidkrevende å planlegge programmeringstimer. Det krever mer tid å planlegge programmeringstimer enn ved vanlige matematikktimer. Det er mer tidkrevende da en må sjekke om alt utstyret som en skal bruke er på plass og fungerer som det skal. Undervisningsopplegget bør testes ut slik at kunnskapen om programmet er på plass i forkant av timen. Selve undervisningstimene kan da bli lite effektiv fordi det kan bli en del rydding av utstyr, både før og etter timen.

Hjelpe elevene

Lærere bruker lang tid når de skal hjelpe elevene. Det blir da vanskelig for lærerne å rekke over alle. Feilsøkingsfasen er noe lærerne sier er tidkrevende da de må kontrollere alle leddene nedover.

> Lærer 6: Vanlig regneoppgave, så kan du lettere kjenne igjen hvor feilen ligger, eller hvor det har skjedd en feil. Du må liksom igjennom hele feilsøkingsprosessen med mindre du finner frem fasiten.

Fullpakket pensum

Flere av lærerne nevner utfordringen med at matematikk allerede er full av emner som elevene skal igjennom på kort tid, og at programmering ikke hjelper med dette. Det blir dermed enda mer tidspress i et fag som allerede har nok tidspress fra før. Lærerne som nevner dette, er de som har programmering som en egen bolk. De ser på programmering som enda et emne som elevene skal gå igjennom. De lærerne som jobber aktivt med å flette programmeringen inn med de andre emnene, opplever også et mer tidspress. For å kunne trekke inn programmering igjennom alle emnene, trenger elevene repetering for å ta i bruk programmeringsferdighetene som ønsket.

> Lærer 6: Skal jeg bruke det på den måten i matematikk slik vi er nå, må jeg bruke tid på å repetere for å kunne bruke det, istedenfor en mer naturlig flyt.

4.2.3 Praktisk side med implementering

Denne kategorien nevnes ant minst, rett ovenfor utfordringer med eierskap. Dette kan være av ulike årsaker, eksempelvis hvor lærerne ligger an i implementeringsløpet. De fleste lærerne underviser programmering som et eget emne i matematikk. Lærer 5 og 6 derimot jobber aktivt med å implementere programmering som en del i alle emnene. Disse to lærerne synes dette arbeidet er utfordrende. De synes at læreverket ikke hjelper med denne jobben og at de føler seg alene i dette arbeidet. De skulle ønsket at programmering var like lett å implementere i matematikk som med Geogebra og Excel.

> Lærer 6: … Det som er med de andre digitale verktøy sånn Geogebra eller Excel. Det er mer naturlig å trekke inn andre oppgaver for det kjenner de til, og du følger deg trygg på at du kan lett trekke dette inn. Mens med programmering vet jeg ikke helt selv hvordan vi

> kan gjør det. Og så er det veldig usikker på hva hvem av elevene som kan nok til å bruke det? Så blir det et helt eget tema. Det blir ikke en del av matematikken.

4.2.4 Elevevaluering

Nesten alle lærerne synes det er vanskelig å evaluere elevene i programmering opp imot kompetansemålene. En lærer har prøvd Kahoot som en evalueringsform og en annen har prøvd egenevaluering. Allikevel er ikke dette løsningen.

Eksamensforberedelsen ble også en utfordring da programmeringsoppgavene så annerledes ut enn det elevene var vant med. Oppgavens utseende var generelt annerledes enn den blokkprogrammeringen som elevene hadde jobbet med på skolen. Det var mangel på generell informasjon angående om elevene ville møte tekstprogrammering eller blokkprogrammering på eksamen. Lærerne fikk vite svaret like før eksamen og dette skapte usikkerhet i hvordan de skulle jobbe med programmeringsoppgaver. Skulle de bruke tid på å introdusere tekstprogrammering eller bruke tiden videre på blokkprogrammering?

> Lærer 5: Det var veldig uklart for oss på hvordan det kommer til å bli på eksamen før nesten rett før, om de kom til å dra inn tekstprogrammering eller blokk programmering. For egentlig er det jo lagt opp til at det skulle være tekst. Men Campus tar jo bare blokk programmering. Så det var og enda er veldig uklart. Kan det plutselig komme full tekstprogrammering? At du hadde fått et program i Python som du skulle forklare eller hva er det de egentlig satser på? Det synes jeg er veldig uklart.

4.2.5 Profesjonell utvikling og støtte

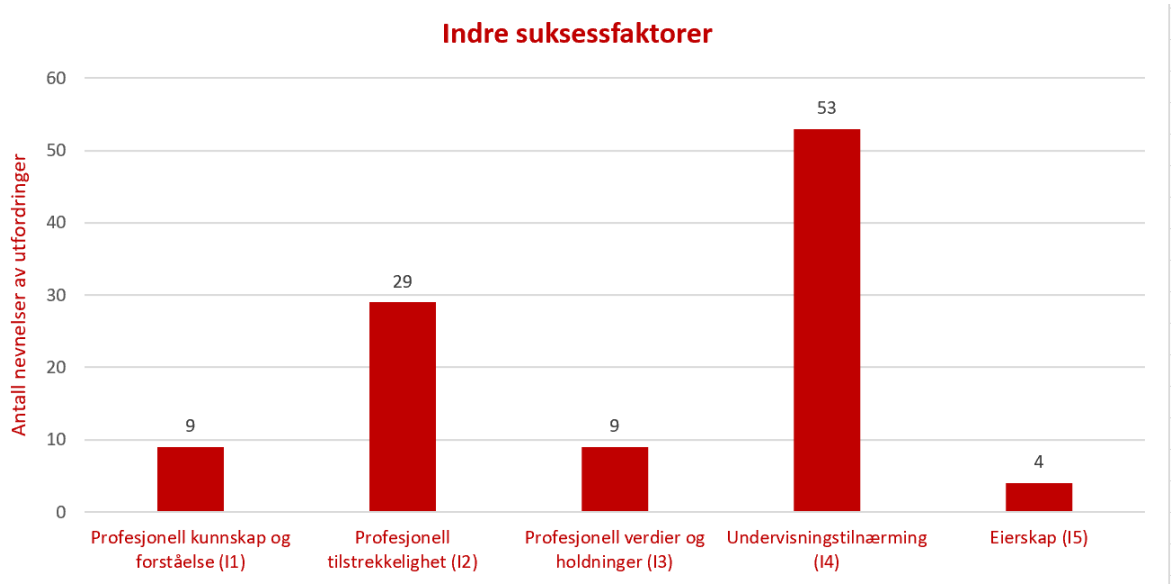
De fleste lærerne nevner at de savnet kursing og at det har vært dårlig opplegg rundt opplæring generelt. Lærer 1,2 ,5 og 6 nevner at det var ingen kunnskapsdeling etter kurs, noe som gjør at kursdeltakerne satt alene med kunnskapen. Lærer 5 og 6 nevner at de aldri ble sendt på programmeringskurs og har kun fått satt av tid til å jobbe med kompetansepakken fra Udir. Det de merker seg nå er at tilgangen på kurs blir dårligere. De er alltid fulle og da er det vanskelig å delta.

> Lærer 3: Noen av kursene blir fulle nesten før de blir lagt ut så det er egentlig utfordring.

Lærer 2 synes det var dårlig støtte fra ledelsen og at det var flere lærere som ikke fikk det de behøvde i starten og mangel på hjelp i sin usikkerhet.

4.3 Indre suksessfaktorer

Indre suksessfaktorer	Hvor mange lærere nevner suksessfaktor	Antall ytringer (suksessfaktor)
Profesjonell kunnskap og forståelse (I1)	2,3,4	9
Profesjonell tilstrekkelighet (I2)	1,2,3,4,5,6	29
Profesjonell verdier og holdninger (I3)	1,2,3,5,6	9
Undervisningstilnærming (I4)	1,2,3,4,5,6	53
Eierskap (I5)	2,4	4



4.3.1 Profesjonell kunnskap og forståelse

Det er få suksessfaktorer som blir nevnt, men den som er delt av flere lærere er klasseromserfaringen. Selv om programmeringsferdighetene ikke er av et ønsket nivå, hjelper det mye med klasseledererfaring. Det å kjenne klassen som et miljø og på individnivå. Kunnskapen om hvem som kan jobbe i grupper og hvor en skal plassere assistentene i klasserommet, er noe som kommer godt med i programmeringstimer.

4.3.2 Profesjonell tilstrekkelighet

Av suksessfaktorer som lærere nevner, var det denne kategorien som har nest flest antall ytringer, kun forbigått av Undervisningstilnærming (I4).

Ærlighet

Det flere av lærerne nevner som en suksessfaktor er det å være ærlig med elevene om egen programmeringsferdighet. Om en elev skulle stille et spørsmål som læreren ikke kan svaret på der og da, skal det ikke være flaut.

> Lærer 2: Så det å være såpass kule at det er helt greit å si at dette har jeg ikke helt kontroll på, men vi finner ut av det.

Bruke elevene

Det som også nevnes av alle seks lærerne er det å kunne lene seg på elevene. Det er ofte flere elever i klasserommet som er flinke i programmering. Enten fra å ha det som et valgfag eller fra egen personlig interesse. Disse elevene er verdifulle i programmeringstimer og kan være til hjelp for både medelever og deg selv som lærer. Alle lærerne nevner suksessfaktoren om det å ta lærdom fra elevene. Viser elevene ukjent kunnskap som læreren ikke har, må en ikke være redd for å ta til seg informasjonen.

> Lærer 5: …vi finner det ut sammen. Dere må lære meg. Egentlig veldig åpen om det for ellers hadde jeg nok ikke vært trygge nok til å stå der selv.

Ifølge Lærer 3 og 4 er det ofte de elevene som synes matematikk er utfordrende som viser seg å bli flinke i programmering. De opplever at elevene med IOP i matematikk synes programmering er gøy og blir fort flinke i programmering. Det er dermed mulig å la elever blomstre som kanskje ikke er så vant til å få blomstre i matematikk fra før av.

> Lærer 3: Så jeg har opplevd at elever som egentlig har IOP i faget, blir de sterkeste i programmering … Så har hatt flere elever på autisme spekteret som programmering har vært midt i blinken.

Tørre å prøve

Alle lærerne nevner suksessfaktoren med å kaste seg ut i det. Alle syntes programmering var skummelt og noen synes programmering fortsatt er skummelt. Om en aldri tårer å prøve, blir programmering heller aldri enklere. Det første steget er det som er det mest skremmende steget. Det er viktig å vite at en ikke trenger å ha stålkontroll for å kunne ha en programmeringstime.

> Lærer 5: Jeg tror terskelen for at mange skal ta det inn er fordi mange føler de må kunne det kjempegodt selv. Jeg føler at du klarer ikke å starte før du kaster deg ut i det.

4.3.3 Profesjonell verdier og holdninger

Det som lærerne nevner som en suksessfaktor i denne kategorien er de nyutdannede lærerne. Negativiteten rundt programmering kommer av at lærerne sliter med å se hvordan de skal bruke programmering i matematikk og hva som er nytten. De opplever at flere av de yngre, og generelt nyutdannede lærerne, har hjulpet med akkurat denne utfordringen. De bringer oftest en positiv holdning og har mer interesse for å finne ut hvordan de kan bruke programmering i matematikk.

> Lærer 6: Mens nå når det kommer en del yngre folk inn som lærere, så er det mer interesse for å kunne bruke det og kan se nytten i det.

Lærer 6 rapporterer at det har skjedd en vesentlig endring i holdninger og verdier sammenlignet til hvordan den var tidligere. Han har troen på at det vil på et tidspunkt være nok unge og nyutdannet lærere som kan omstille verdiene helt og at ting vil ro i land.

4.3.4 Undervisningstilnærming

Oppskrift basert

Flere av lærerne synes oppskrifter har virket som en suksessfaktor. Disse oppskrifter kan en finne på Superbit.no eller om de søker selv etter et opplegg på Youtube.com. Det de liker med oppskrifter er muligheten til å sortere inn etter vanskelighetsgrad. Dette gjør det lettere å differensiere. Lærer 3 har printet ut oppskriftene slik at de er lett tilgjengelige. Dette har hun gjort med tanke på de lærerne på skolen som synes programmering er svært utfordrende. Dette skal da kunne virke som en støtte uten at de trenger å ha den kompetente andre læreren ved siden av seg.

> Lærer 3: Jeg har jo følget nettside, hvor du følger steg på steg. Det er jo mikrobit og bitt-bot hvor vi har laget ferdig opplegg. Vi har laget ferdig opplegg som er laminert, og så har vi lagt det i rekkefølge inni koden skapet sånn at lærerne kan støtte seg på det.

Samarbeidslæring

De fleste lærerne løfter frem en form for samarbeidslæring som en suksessfaktor. Både i form av at læreren lærer sammen med elevene, eller elever som lærer av de andre elevene. Flere av

lærerne bruker gruppearbeid og læringsvenn. Lærer 6 bruker aktivt CL grupper som består av fire elever hvor det alltid vil være en elev som vil virke som den kompetent andre. Dette mener Lærer 6 løser problematikken med å rekke over alle eleven. Er det en gruppe som ikke får til problemet, så kan flere elever henge seg på i letingen etter en løsning. Da kan læreren rette fokuset mot andre problemer i mellomtiden.

> Lærer 6: Du er avhengig av å kunne bruke disse sterke elevene da du kan risikere å bruke hele timen på en oppgave. At den ene i CL gruppa kan komme bort å si at nå tror jeg vi fant feilen.

Lærer 5 liker å bruke elevene til å lære andre elever. Ofte kan de forklare kunnskapen på en bedre måte enn det hun selv kan. Hun har da spurt de elevene som hun vet brenner for programmering, om de har lyst til å delta i undervisningen og lære fra seg sin kunnskap.

> Lærer 5: I mange andre fag så kan jo elevene forklare det bedre enn det vi kan. For meg så er det mange elever som kan veldig mye. Både blokk og skrift programmering.

Prosjekt og spill-basert undervisning

Suksessfaktoren til lærer 4 er prosjekt og spill-basert undervisning. Han mener det ligger en suksess generelt i å undervise i noe elevene synes er gøy. Da mener han at spill er veien å gå når det gjelder programmering. Nettsidene han har brukt er Codemonkey.com og Scratch.com. Codemonkey er en nettside han bruker på småtrinnet, da dette er en enklere utgave av Scratch.

> Lærer 4: Det blir mye spill-basert blokkbasert. Der det er lett inngang og lett å få tak på det grunnleggende, men fortsatt er masse muligheter for elevene.

Campus Inkrement

Lærer 2 sverger til Campus Inkrement. Det har hjulpet henne mye med å kunne gi elevene en god undervisning, selv om hennes egen kunnskap ikke er på et ønsket nivå. Dette gjorde hennes start med programmeringsundervisning i matematikk mye enklere.

> Lærer 2: Og der ligger det ferdige leksjoner, både med læringsvideoer og filmer. Og da slappet jeg egentlig veldig godt av. Undervisningen i fjor så valgte vi å ha mye samarbeidslæring slik at elevene jobbet i grupper slik at de kunne hjelpe hverandre. Og så

> gikk jeg og lærte litt på siden, og det var veldig trygt og godt for meg

4.3.5 Eierskap

Av alle seks lærerne var det bare en som svarte at han følte litt eierskap. Han mener at hans suksessfaktor har en god del med egen interesse å gjøre. Han synes programmering er spennende og gøy, og har generelt mye mer programmering en det læreplanen legger opp til. En suksessfaktor som han peker på, er det å bygge videre på andres ideer. En må ikke si seg fornøyd med ett opplegg fra andre, men heller gjøre små endringer slik at det passer inn i egen undervisning.

> Lærer 4: Det er jo et emne som du må være litt interessert i, å lete litt på egen hånd. Kanskje mer enn andre deler av mattefaget. Det er en del tekniske ting som du må komme i gang med da før du kan gå i gang med elevene. Du trenger ikke å være utlært.

4.4 Ytre suksessfaktorer

Ytre suksessfaktorer	Hvor mange lærere nevner suksessfaktor	Antall ytringer (suksessfaktor)
Ressurser (E1)	1,2,3,4,5,6	18
Tidshåndtering (E2)	2,4,6	5
Praktisk side med implementering (E3)		0
Elevevaluering (E4)	4,5,6	4
Profesjonell utvikling og støtte (E6)	2,3,4	8

4.4.1 Ressurser

Koderom

Lærer 3 og 4 jobber på en skole som har dedikert et helt rom til å bli et offisielt koderom. Dette rommet inneholder ulike programmeringsutstyr og diverse tilbehør som en skulle trenge til programmering. Dette skal kunne gjøre det lettere for lærerne å utforske, og ikke minst bruke programmering i matematikktimene. I dette rommet ligger det også klart ulike undervisningsopplegg og oppgaver rangert etter vanskelighetsgrad. I løpet av dette året skal det monteres skjermer opp på alle veggene. Dette skal gjøre det lettere for elevene å programmere utstyret for å så teste det ut på det åpne gulvet.

Læreverk

Flere av lærerne nevner hvordan læreverk har vært en suksessfaktor for dem. Med et godt læreverk kan en slappe mer av, da en ikke trenger å være så avhengig av egen kunnskap.

Lærer 1 løfter frem boka Matematikk fra Cappelen Dam og sier at den har vært behjelpelig i hennes programmeringsundervising. Lærer 2 er meget fornøyd med Campus Inkrement da den er en støtte i hennes undervisning. Dette læreverket differensierer med å gi elevene oppgaver av riktig vanskelighetsgrad. Læreverket tilbyr en enkel måte for repetisjon om elevene skulle trenge det. Lærer 5 er også fornøyd med pakken Campus Inkrement kan tilby, men hun mener at det mangler støtte for tekstprogrammering. Løsningen hennes er å bruke Skolestudio og et ekstra program til selve tekstprogrammeringen. Dette synes hun er en tungvint løsning og lengter etter tilgangen hun hadde til Kikora. Dette læreverket tilbyr en helhetlig løsning på lik måte som Campus Inkrement, men har også den ekstra støtten for tekstprogrammering. I tillegg synes hun at Kikora tilbyr en bedre opplæring generelt. De legger matematikken og programmeringen frem på en oppbyggende og god måte slik at elevene møter oppgaver som de er i stand til å løse.

> Lærer 5: Med Kikora som vi brukte på den forrige skolen jeg jobbet, der var det et eget læringsverktøy for Python. Både blokkprogrammering og Python. Det savner jeg så masse, for det brukte jeg med 8 klassingene mine for 2 år siden. Det var mye enklere for der hadde de oppgaver og forklaringer. Dette har jeg savnet med campus og skolestudio. De mangler denne helheten.

4.4.2 Tidshåndtering

Lærerne kommer ikke med så mange suksessfaktorer når det gjelder tidshåndtering, men det som blir nevnt er det å prøve å samle timene. Programmeringstimer er ofte tidkrevende fordi det er så mye som må ryddes i forkant og i etterkant. Ved å ha dobbelttimer blir det mer tid til programmering og mindre rydding.

4.4.3 Praktisk side med implementering

Dette er et punkt hvor lærerne ikke kom med noen suksessfaktorer, men igjen presenterer lite utfordringer under denne kategorien generelt.

4.4.4 Elevevaluering

Kahoot og egenevaluering

Når det kommer til suksessfaktorer med elevevaluering, har lærerne testet ut flere varianter. Lærer 5 har pleid å bruke Kahoot som ikke gir innsikt i hva enkeltelever kan, men kan gi en pekepinn på hvordan klassen ligger an. Lærer 6 har pleid å bruke egenevaluering som han

opplever kan indikere hvordan eleven ligger an i forhold til kompetansemålet. Utfordringen med dette er elevens forhold til å kunne evaluere seg selv opp imot et kompetansemål.

Campus inkrement og prosjektbasert undervisning

Campus Inkrement har mulighet for å evaluere elevene. Den vil generere en prøve som viser hvordan elevene ligger an, både på klassenivå og individ nivå. Elevene får en oversikt over hva de bør jobbe videre med frem mot prøver og eksamen. Læreverker sorterer alle eksamensoppgaver inn etter emner slik at det blir oversiktlig for elevene. Lærer 4 anbefaler prosjekter da dette vil gi en løpende innsikt i hva elevene kan. Denne metoden kan gi mulighet for både underveisvurdering og sluttvurdering.

4.4.5 Profesjonell utvikling og støtte

Flere av lærerne nevner suksessfaktoren å ha støttende kollegaer. Det å kunne rådføre seg med andre lærere og få den støtten som en trenge. De gangene Lærer 2 ikke kunne hjelpe elevene var det god trygghet med å kunne støtte seg på en kollega.

> Lærer 2: Så min erfaring fra i fjor hvor andre elever i klassen heller ikke kunne det så va det bare å si at måtte spør kollegaen. Jeg kunne sende melding på teams i timen og så kunne han svare med en gang om han hadde mulighet. Det er jo vårt ansvar å hjelpe elevene med å finne svarene hvis de står fast.

5. Drøfting

Lærerne presenterer flere av de samme utfordringene som også dukker opp i annen forskning. De løfter blant annet opp utfordring med at programmering er skummelt. Dette da de ikke har hatt programmering i egen studie og at de ikke kan det godt nok (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). På lik linje som Haraldsrud et al., (2020) nevner disse lærerne også utfordring med elevers motivasjon. De er også enige med Finger & Houguet (2009) , Jones et al. (2004), Sentance & Csizmadia (2017) og Vinnervik (2022) angående utfordringer med ressurser som penger og utstyr. Når det kommer til suksessfaktorene var det mindre suksessfaktorer som også gikk igjen i andre studier. Årsak kan være at flere studier har utfordringer som fokus og inkluderer lite til ingen suksessfaktorer. Blant alle utfordringene og suksessfaktorene var det noen som var mer interessante enn andre. Noen av de interessante utfordringene og suksessfaktorene blir nevnt eksplisitt og andre ikke.

5.1 Læreres forståelse av programmeringens plass i matematikken

Lærer 1, 3 og 4 løfter frem læreplanverket, hvor lærer 1 synes det er utfordrende å forstå hva elevene skal lære. Denne utfordringen blir også nevnt av Jones et al. (2004). Lærer 3 og 4 kommenterer at det er få kompetansemål som går under programmering. At det er typisk kun et kompetansemål på hvert trinn som hører til programmering. Detter løfter opp tanken om at lærere har en utfordring med å vite forskjellen på digitale ferdigheter og programmeringens plass i læreplanen. Som nevnt tidligere er programmering sidestilt med de andre digitale verktøyene, eksempel Excel og Geogebra (Utdanningsdirektoratet, 2020a). Det vil da si at selv om det kun er et kompetansemål som tvinger læreren til å bruke programmering med elevene, er det fremdeles flere kompetansemål som kan brukes sammen med programmering. Som Bungum & Vinnervik (2022) sier, inviterer læreplanen til å kunne bruke programmering i flere av kompetansemålene. Dette er basert på at det er flere kompetansemål som faller under digitale ferdigheter. Da programmering er en av de digitale verktøyene, kan programmering også brukes i arbeidet med disse kompetansemålene. Det vil ikke si at programmering er det mest gunstigste digitale verktøyet å bruke til alle disse kompetansemålene, men at det er en mulighet. Lære 1, 3 og 4 mener også at Excel, Geogebra og de andre digitale verktøyene er programmering. Dette tyder da videre på at lærerne har en utfordring med å forstå forskjellen på programmering og digitale ferdigheter. Det som går igjen ved alle disse skolene er at de støtter seg mye til læreverk. Lærer 1 løfter fram boken

Matematikk fra Cappelen Damm som en av hennes suksessfaktorer med programmering i matematikk. Ved å se nærmere på denne boka inneholder den ikke noe form for programmering, kun Geogebra og regneark. Dette vil da tyde på at det er flere som sitter inne med en misforståelse av hva som inngår i programmering og hva de digitale ferdighetene er. Det virker som at det er en misforståelse om at de digitale verktøyene går under det å programmere. Ifølge læreplanen er ikke programmering, Excel og Geogebra det samme, men likestilte digitale verktøy under digitale ferdigheter (Utdanningsdirektoratet, 2020a).

Ved å legge fokuset på rene programmeringstimer i matematikk, er spørsmålet hvor den matematiske siden blir av undervisningstimen blir av. Lærer 1 og 3 løfter frem usikkerheten om hvor lang tid elevene skal få til programmeringen. Begge disse velger programmering som en egen bolk, og er mer fokusert på å lære elevene programmering enn å flette inn matematikken. Lærer 1 sier at elevene alltid kan jobbe videre med programmeringen, men synes det er utfordrende å vite om hun skal prioritere av de andre emnene istedenfor. Dette kan indikere at hun ikke ser den matematiske siden ved programmering i matematikktimer. Som Bungum & Vinnervik (2022) løfter frem at programmering ikke er det samme som algoritmisk tenkning, men er et verktøy til å forstå og utvikle algoritmisk tenkning. Andre studier peker også på at programmering, algoritmisk tenkning og problemløsning henger sammen (Kaufmann & Stenseth, 2021; Grover & Pea, 2013; Stenseth et al., 2019). Ved å la elevene møte programmeringen på en problemløsende måte, kan en argumentere for at matematikken er til stede i undervisningen.

5.2 Tidspress og implementeringsvalg

Lærere opplever flere utfordringer med tidspress når det kommer til undervisning i programmering i matematikk. På lik linje som Haraldsrud et al. (2020) og Statped (2021) løfter disse lærerne frem utfordringen med å ha nok tid til å strekke over alle elevene i programmeringsundervisning. Flere elever trenger hjelp og læreren opplever at det tar lenger tid å hjelpe elever i programmeringstimer enn i en vanlig matematikktime. Det samme gjelder feilsøking, da det tar lenger tid en i de andre emnene i matematikk. Denne ekstra tiden per elev fører til at det blir mangel på tid til alle elevene. En annen utfordring som faller under tidspress er et fullpakket pensum og denne utfordringen gjenspeiles i flere studier (Finger & Houguet, 2009; Jones et al., 2004; Sentance & Csizmadia, 2017; Vinnervik, 2022). Lærerne har valgt to forskjellige tilnærmingsmetoder for implementering av programmering i matematikk. Noen lærere inkludere et ekstra emne som elevene skal igjennom, mens andre

prøver å flette programmering inn i de andre emnene. Allikevel sier alle lærerne at det er tidkrevende med programmering i et allerede fullstappet pensum.

Lærer 1, 2 og 3 har valgt å implementere programmering som et eget emne. Dette setter tidspress da lærerne må få tid til et ekstra emne i årsplanen. Noe som går ut over de andre emnene. Lære 4, 5 og 6 har valgt å implementere programmering som en del av alle emnene. Ifølge lærer 5 og 6, er problemet med denne tilnærmingen at hver gang en skal ha programmering må elevene få tid til å friske opp programmeringsferdighetene sine. Får ikke elevene tiden til oppfriskning, kan det gå ut over det ønskede utfallet med timen. Det vil da si at ingen av tilnærmingene er bedre enn den andre i forhold til tidsbruk. Derimot vil lærer 5 og 6 foretrukket programmering som en egen bolk, da elevene får jobbe mye med programmering over en kortere tid. De velger fremdeles ikke å gjøre det slik da de mener implementeringen av programmering i matematikk tilsier at den skal være en del av helheten og kunne brukes som et verktøy til å løse noe matematisk. Forsström & Kaufmann (2018) argumenterer at det må finnes gode undervisningsmetoder for å inkludere programmering inn i de andre emnene. Dette kan tolkes som at programmering skal jevnlig brukes igjennom undervisningsforløpet og ikke som en egen bolk. På en annen side har vi Bungum & Vinnervik (2022) som tolker den norske læreplanen slik at elevene skal lære å programmere i 5-6 klasse. Deretter i 7 klasse lære å bruke programmering som et verktøy for å stå klar i 10 klasse til å bruke programmering som et verktøy. Via denne tolkningen kan en tenke seg at det blir mer naturlig å ha programmering som en egen bolk fra 5 klasse av og bevege seg mer og mer mot en mer integrert programmering frem til 10 klasse. Igjen så advarer Kaufmann & Stenseth (2021) om å ikke ha fokuset på rent på programmeringen.

Ved å se sammenheng i det lærerne sier og det forskning sier, samsvarer realiteten mest med hvordan Bungum & Vinnervik (2022) tolker lærerplanen. Ovenfor kan en se lærere på barneskoler som har en trend med å implementere programmering i matematikk som et eget emne. Ungdomskoler har en mer trend å implementere programmering som en del av helheten av matematikk, og prøver da å inkludere programmering i alle emnene. De som skiller seg ut fra mengden er lærer 2 som strider imot det som Bungum & Vinnervik (2022) tolker. Dette da hun arbeider på ungdomskolen, men implementerer programmering som en egen bolk ved slutten av året. Lærer 4 som også strider mot Bungum & Vinnervik (2022) tolkning, men støttes opp av Forsström & Kaufmann (2018) og Kaufmann & Stenseth (2021). Dette da han arbeider på barneskolen og implementerer programmering i matematikk som en

del av helheten. Lærer 5 og 6 blir støttet av alle tre da de arbeider på ungdomskolen og implementerer programmering som en del av helheten.

Dette kan da tyde på at implementeringsvalget bør være at elevene møter programmering i de fleste emnene. Da er det kanskje naturlig å lure på når elevene skal få lære seg å programmere slik at de står klar til å bruke programmering som et verktøy i matematikk. Til dette kan en da sammenligne programmering med problemløsning, som det også blir gjort av Kaufmann & Stenseth (2021). Både Liljedahl (2021, s. 147 - 148) og Polya (1945) sier at elvene må få bruke tid til å opparbeide seg metoden for å arbeide problemløsende. Det samme vil da kunne gjelde med programmering i matematikk. Læreren må påregne seg at det vil ta tid før elevene sitter inne med programmeringskunnskapene og må da legge til rette for dette. Liljedahl (2021, s. 147 - 148) argumenterer også for at læreren må påregne seg å la elevene arbeide med ikke læreplanmål før elevene kan arbeide med læreplanmål. Knytter en dette opp mot programmering, kan en tolke det som at elevene må arbeide med ren programmering i starten før de kan gå over til å programmere i matematikk. Da begynner en å nærme seg Bungum & Vinnervik (2022) tolkning av læreplanen.

5.3 Undervisningsmetoder

Lærerne kommer med flere suksessfaktorer, men det er få som overlapper. De suksessfaktorene som flere nevner er å tørre å prøve og det å ha et godt læreverk å lene seg på. Ved å se på alle kategoriene i Vinnerviks (2022) modell, er det den indre suksessfaktoren Undervingstilnærming (I4) som skiller seg ut. Ved å se på antall suksessfaktorer og antall utfordringer, kan en se at denne kategorien har flere suksessfaktorer enn utfordringer. Noe som skiller seg i fra de andre kategoriene. Nedenfor presenteres flere av undervisningsmetodene som de løfter frem.

Kopiering og justering

Lære 1 og 3 presenterer suksessfaktoren å la elevene følger oppskrifter. Lærer 1 velger oppskrifter da det letter hennes usikkerhet i hvordan hun kan ha en god programmeringsundervisning. Lærer 3 liker å bruke oppskrifter da det er enkelt å differensiere for elevene. Spørsmålet er om oppskrifter er læringsrikt for elevene, både for matematikkforståelsen og programmeringsferdigheten. I følge Kaufmann & Stenseth (2021) kan en slik prosess bli sett på som matematisk, da elevene har muligheten til å arbeide med kritisk tenkning. De anbefaler å la elevene kopiere program for å så gjøre justeringer med

programmet. På denne måten kopierer ikke elevene et program uten å ha forståelse for programmet. For å gjøre endringer som fører til ønsket handling, må elevene ha en viss forståelse for programmet som de skal justere. Det vil da si at forskning peker på at oppskrift-basert undervisnings er læringsrikt for elevene så lenge oppgavene ikke består av ren kopiering. Oppgavene må være laget slik at elevene får utforske og bygge en forståelse for hvordan programmet fungerer eller skal fungere. Verken Lærer 1 og 3 nevner noe om justering, kun kopiering, og da kan læringen frafalle ifølge Kaufmann & Stenseth (2021).

Samarbeidslæring

Flere av lærerne nevner samarbeidslæring som en suksessfaktor. Ifølge lærerne løser det problemer som tidsmangel, utstyrsmangel og generelt utfordringen med å strekke over alle elevene som trenger hjelp. Noen av lærerne bruker de flinkere elevene til å kunne bistå de andre elevene som trenger hjelp. På lik linje med Haraldsrud et al. (2020) og Statped (2021) løfter Lærer 5 og 6 frem at de synes det er utfordrende med elevers motivasjon i programmering. De opplever at elever må vente lenge på å få hjelp av læreren og det kan da føre til demotivering. I følge Liljedahl (2021, s. 147 - 148) er det viktig å ikke la elevene vente for lenge for da kan de vippe over på frustrasjon eller kjedsomhet. Til dette rapporterer de at samarbeidslæring har hjulpet med denne utfordringen. De benytter seg av Cl grupper, som baserer seg på at det alltid skal være minst en sterk elev på hver gruppe. Dette går rett inn under Vygotskys Sosiokulturelle teori, med den proksimale utviklingssonen. Elever som synes programmering eller generelt en oppgave er vanskelig, vil kunne støtte seg til den sterkere eleven og vil da kunne klare å løse vanskeligere oppgaver enn det de ville gjort alene. Taylor et al. (2010) argumenterer i sin studie at samarbeidslæring kan gi elever med en ‘‘lav’’ matematisk forståelse mulighet til å utvikle en kompleks algoritmisk tenkning og at de kan komme med gode matematiske ideer. Den eleven som stiller som den med kunnskapen i slike situasjoner, vil få erfart det å sette ord på egen kunnskap som igjen går inn under både den sosiokulturelle og konstruktivistiske læringsteorien. Via litteraturstudien til Forsström & Kaufmann (2018), viser en av resultatene at programmering får røtter i både den sosiokulturelle læringsteorien samt den konstruktivistiske læringsteorien. En ting som går igjen i begge teoriene er viktigheten med språket (Imsen, 2015, s. 145 - 146). Det at elevene får brukt språket sitt til å sette ord på sin egen kunnskap er med å bygge deres individuelle kunnskap. Ingen elever sitter med den eksakt samme kunnskapen og vil kunne stille med ulike løsningsstrategier og fremgangsmetoder i samarbeidssituasjoner. Elevene vil da kunne lære av hverandre ved å lytte til medeleven. Til og med Seymour Papert la opp sine

forskningstimer etter samarbeidslæring, slik at elevene fikk bruke språket sitt (Forsström & Kaufmann, 2018). Som en kan se så støtter teorien opp at samarbeidslæring er en suksessfaktor når det kommer til programmering i matematikk. Forsström & Kaufmann (2018) løfter frem hvordan elever også har roller i et klasserom. En elev som synes matematikk er vanskelig, kan kunne ta på seg en mer ledende rolle og kan da stå mer ansvarlig for de matematiske ideene i et gruppearbeid. Lærer 3 og 4 nevner hvordan de opplever at programmering er en suksessfaktor i seg selv når det kommer til elever på autismespekteret. De viser stor interesse og tar ofte programmeringen med hjem som en hobby og blir fort veldig flinke argumenterer lærer 4. Ifølge lærer 3, 4, 5 og 6 får elevene som synes matematikk er utfordrende også blomstre når det kommer til programmering i matematikk. Noe som de synes er veldig positivt at disse elevene også møter mestring i matematikkfaget.

Campus Inkrement

Fire av seks lære bruker Campus Inkrement og løfter det frem som en suksessfaktor. Lærer 1 baserte deler av programmeringsundervisningen på dette læreverket. Lærer 2, 5 og 6 baserte all matematikkundervisning og flere av disse lærerne kunne senke skuldrene sine med undervisning med dette læreverket. Campus viser til å kunne stille med en komplett pakke i forhold til undervisning, differensiering, underveisvurdering, vurdering og eksamenstrening. Med andre ord kan Campus Inkrement dekker det som lærerne trenger, og det som lærerne mener elevene trenger. En kritikk fra lærer 5 er at læreverket ikke støtter tekst programmering, noe som gjør det utfordrende med tanke på eksamen. Sammenligner en denne programmeringsundervingen med forskning strider denne metoden litt fra anbefalingene til Sentence & Csizmadia (2017), Dolonen et al. (2019) og Gjøvik & Torkildsen (2019). Fellestrekket mellom alle disse er anbefalingen om å la elevene arbeide med fysisk programmering. Campus Inkrement har ingen mulighet for fysisk programmering og tilbyr kun digitale arbeidsmetoder.

Spill basert programmeringsundervisning

Lærer 4 presenterer spill-basert læring som en suksessfaktor. For han har denne undervisningstilnærmingen gjort undervisning gøy både for han og elevene. Han opplever heller ikke problemer med elevers motivasjon i samme grad som de andre lærerne. Noe som også støttes av Figueiredo & García-Peñalvo (2020) og Zhan et al. (2022) i deres studier. Denne læreren bruker kun spill-basert undervisning og synes evaluering også blir enklere. Da

elevene programmerer et spill i Scratch vil eleven få masse underveisvurderinger. Han liker å la elene spille hverandres spill slik at de kan gi tilbakemeldinger på hverandres spill. En kan stille spørsmålet om denne formen for undervisnings blir veldig programmeringsorientert og lite matematikk. Nå arbeider denne læreren på småtrinnet og mellomtrinnet og bruker denne tilnærmingen til å lære elevene programmering. Uten å gå mer inn på samme tematikk som tidligere kan en utforske hvor matematikken i en slik undervisningstilnærming befinner seg. Hva om en sammenligner denne formen for spill-basert undervisning med algoritmisk tenkning. Ved å se på modellen for algoritmisk tenking (se figur 1) kan en argumentere for at alle arbeidsmåtene befinner seg i en slik undervisningstilnærming. Når elevene skal lage et eget spill fra bunnen av må de utforske, designe, oppdage, feilsøke, vise utholdenhet og samarbeide i form av kunnskapsdeling. Av nøkkelbegrepene må elevene tenke logisk igjennom hele prosessen. De må tenke algoritmisk da de må planlegge det som trengs i spillet. Eks spillebevegelser og poengsystem. Når elevene støter på problemer, må de dele opp problemet i mindre mer håndterbare problem. Eks karakteren til å sparke en ball. Da må eleven tenke på hva karakteren skal gjøre, hva ballen skal gjøre når karakteren sparker ballen. Eleven vil oppleve å se etter mønster for å kunne effektivisere programmet slik at sekvensene ikke blir lange og dermed også fjerne det som er unødvendig. Elevene vil også oppleve evaluering igjennom hele prosessen med å teste programmet, feilsøke, la medelever teste programmet osv. En kan da argumentere for at den algoritmiske tenkingen reflekteres i en slik undervisningstilnærming. Da Utdanningsdirektoratet (2019) definerer algoritmisk tekning som en problemløsningsmetode, kan en også si at spill-basert undervisningsmetode inneholder problemløsning. Både algoritmisk tekning og problemløsning er sterkt knyttet sammen med matematikken. Kaufmann & Stenseth (2021) argumenterer også for at programmering hører til i matematikken basert på den algoritmiske tenkningen. Da kan en også trekke konklusjonen over at det er matematikk til stede i spill-baserte undervisnings selv om de virker programmeringsorientert. Ordet spill dukker også opp i kompetansemålene som ligger under digitale ferdighet for 3 -5 trinn og 9 trinn. Noe som tyder på at den spill-baserte læringen i programmeringen også får plass i læreplanen.

5.4 Forskningsbasert undervisningsmodeller

Ingen av lærere løftet frem eksplisitt forskningsbaserte undervisningsmodeller for programmering i matematikk. Dolonen et al. (2019) nevner den lovende forskningsmodellen PRIMM og Sentance & Csizmadia (2017) nevner modellen for algoritmisk tenkning som en

suksessfaktor. Dette var noe av grunnlaget til hypotesen om at forskningsbaserte undervisningsmodeller ville bli nevnt av noen av lærerne. Allikevel presenterer noen lærere undervisningsmetoder som kan knyttes opp til algoritmisk tenkning og delvis PRIMM modellen. Den spill-baserte undervisningstilnærmingen til lærer 4 lar elevene få oppleve å predikere, kjøre, utforske modifisere og lage. Selv om elevene får oppleve disse delene er det fremdeles ikke på lik linje som med en i gjennomtenkt PRIMM basert undervisning. Som nevnt tidligere har den spill-baserte undervisningstilnærming mer sammenheng i modellen for den algoritmiske tenkningen. Lærer 1 og 3 som kjører oppskrift basert har også muligheten for å få til flere av fasene i PRIMM modellen. Men det samme gjelder her også med at elevene ikke får oppleve fasene til modellen som tenkt. For å konkludere var det forventet at lærere brukte forskningsbaserte undervisningsmetoder og modeller bevist. Men ut ifra disse seks lærerne så er det kun lærer 4 men den spill-baserte undervisningen som bruker en forskningsbasert tilnærming mer bevist.

6. Konklusjon og videre forskning

6.1 Konklusjon

Lærerne kom med flere utfordringer som også går igjen i andre studier. De løfter frem utfordring med egen usikkerhet, med elevers motivasjon, ressurser og problematikken rundt tilgangen på kurs. Det tolkes også at lærere har en utfordring med å forstå programmeringens plass i matematikk og læreplanen. Dette da det virker som noen lærere tror at programmer og digitale ferdigheter er det samme. De sidestiller ikke programmering som en digital ferdighet på lik linje som Geogebra og Excel, men mener Geogebra og Excel er programmering. Lærere opplever også at det utfordrende med tid når det gjelder programmering. Det er utfordrende å ha nok tid til å strekke over alle elevene som trenger hjelp. Det er tidkrevende uansett hvilke implementeringsvalg. Om læreren velger å implementere programmering som et eget emne eller som en del av de andre emnene er fortsatt utfordrende med tanke på tidsmangelen med et fullstappet pensum.

Suksessfaktorene de løfter frem er god klasseromserfaring, tørre å prøve, lene seg på elevene, ærlighet ovenfor elevene og støttende kollegaer. Kategorien *Undervisningstilnærming (I4*) er den eneste kategorien hvor antall suksessfaktorer er flere enn antall utfordringer. Suksessfaktorene de nevner er kopiering og justering, hvor elevene må følge en oppskrift og gjøre endringer. Samarbeidslæring presenteres også som en suksessfaktor hvor elevene må lene seg på hverandre, eller arbeide i grupper. Samarbeidslæring har vist positivt på tidsspekteret og elevenes ventetid på hjelp. Læreverket Campus Inkrement er en suksessfaktor da den blant annet tilbyr en komplett pakke for lærer og elev. Dette har ført til at om læreren selv ikke føler de har kontroll på programmering, kan de stole på at elevene får den undervisningen de skal. Siste suksessfaktoren er spill-basert undervisningstilnærming hvor elevers møte med spill i programmeringstimer kan øke motivasjonen. Det ble kun nevnt spill-basert undervisningstilnærming som en bevist forskningsbasert metode. Ellers var det ingen lærere som eksplisitte nevner forskningsbasert undervisningsmodeller og metoder.

6.2 Studiens svakheter og videre forskning

Svakheter med denne studien er antall informanter. Da problemstillingen er ute etter utfordringer i Norge er antall informanter for lite til å kunne representere norske lærere på en god måte. Alle informantene kommer også fra samme fylket, noe som heller ikke kan representere utfordringer og suksessfaktorer fra andre fylker og steder. Eksempelvis Lærere i

fylket Finnmark eller stedet Svalbard opplever sannsynligvis andre utfordringer og suksessfaktorer enn de som lærerne fra denne studie løfter frem.

Til videre forskning kan det være interessant å se på hvordan en kan implementere programmering som en naturlig del i de andre emnene i matematikk. Det kunne også vært interessant med en studie hvor en lærer baserer undervisningene sine på en forskningsbasert undervisningsmodell for programmering i matematikk. Eksempel PRIMM eller algoritmisk tenkning. For å så sammenligne disse undervisningene med en annen læreres undervisning uten slike modeller. Det kunne også vært interessant å sett på implementering av programmering via elevenes syn. Hva opplever elevene som utfordrende og suksess med implementering av programmering? Det er også faktumet av Norge er et av landene som har implementert programmering som en del av allerede eksisterende fag. Andre land har valgt å implementere programmering som et eget fag. Det kunne derfor vært interessant å sett hvordan et selvstendig programmeringsfag kan påvirke inkluderingen av programmering i matematikk.

Referanser

Bungum, B., & Vinnervik, P. (2022). *Computational thinking as part of compulsory education: How is it represented in Swedish and Norwegian curricula?* https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/3049986/Vinnervik_Bungum_NorDiNa_45_3_2022.pdf?sequence=2

Copeland, B. J. (2000). *The modern history of computing*. https://plato.stanford.edu/ENTRIES/computing-history/

Dolonen, J. A., Kluge, A., Litherland, K., & Mørch, A. I. (2019). *Litteraturgjennomgang av programmering i skolen*. https://www.duo.uio.no/handle/10852/76290

Ertmer, P. A., & Ottenbreit-Leftwich, A. T. (2010). Teacher Technology Change: How Knowledge, Confidence, Beliefs, and Culture Intersect. *Journal of Research on Technology in Education*, *42*(3), 255–284. https://doi.org/10.1080/15391523.2010.10782551

Fauskanger, J., & Mosvold, R. (2014). Innholdsanalysens muligheter i utdanningsforskning. *Norsk pedagogisk tidsskrift*, *98*(2), 127–139. https://doi.org/10.18261/ISSN1504-2987-2014-02-07

Figueiredo, J., & García-Peñalvo, F. J. (2020). Increasing student motivation in computer programming with gamification. *2020 IEEE Global Engineering Education Conference (EDUCON)*, 997–1000. https://gredos.usal.es/bitstream/handle/10366/143086/1433-pre.pdf?sequence=1

Finger, G., & Houguet, B. (2009). Insights into the intrinsic and extrinsic challenges for implementing technology education: Case studies of Queensland teachers.

International Journal of Technology and Design Education, *19*(3), 309–334. https://doi.org/10.1007/s10798-007-9044-2

Fixdal, T. (2018). Problem eller utfordring? *Tidsskrift for Den norske legeforening*. https://doi.org/10.4045/tidsskr.18.0150

Forsström, S. E., & Kaufmann, O. T. (2018). A literature review exploring the use of programming in mathematics education. *International Journal of Learning, Teaching and Educational Research*, *17*(12), 18–32.

Gjøvik, Ø., & Torkildsen, H. A. (2019). Algoritmisk tenkning. *Tangenten–tidsskrift for matematikkundervisning*, *30*(3), 31–37.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

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

Hsieh, H.-F., & Shannon, S. E. (2005). Three Approaches to Qualitative Content Analysis. *Qualitative Health Research*, *15*(9), 1277–1288. https://doi.org/10.1177/1049732305276687

Imsen, G. (2015). *Elevenes verden. Inføring i pedagogisk psykologi* (5. utgave). Universitetsforlaget.

Johansen, A.-K., & Østfold, H. i. (2020, juli 11). *– 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

Jones, A., Harlow, A., & Cowie, B. (2004). New Zealand teachers' experiences in implementing the technology curriculum. *International Journal of Technology and Design Education*, *14*, 101–119.

Kaufmann, O. T., & Stenseth, B. (2021). Programming in mathematics education. *International Journal of Mathematical Education in Science and Technology*, *52*(7), 1029–1048. https://doi.org/10.1080/0020739X.2020.1736349

Liljedahl, P. (2021). *Building Thinking Classrooms in Mathematics, Grade K - 12: 14 Teaching Practices for Enhancing Learning.* Corwin Press. https://web-p-ebscohost-com.ezproxy.uis.no/ehost/ebookviewer/ebook/bmxlYmtfXzI2Mzc3MTJfX0FO0?sid=f7026686-9b6c-40b8-a5d7-4fc73ddaad04@redis&vid=0&format=EB&lpid=lp_132&rid=0

Matematikksenteret. (u.å). *Didaktisk Grunnlag*. https://www.matematikksenteret.no/l%C3%A6ringsressurser-og-undervisningsopplegg/programmering/anbefalt-arbeidsmetodikk

McCarthy, J. (2007). *What is artificial intelligence*. http://cse.unl.edu/~choueiry/S09-476-876/Documents/whatisai.pdf

Petzold, C. (2000). *Code: The hidden language of computer hardware and software*. Microsoft Press. https://books.google.com/books?hl=en&lr=&id=iptCAwAAQBAJ&oi=fnd&pg=PT4&dq=%22Code:+The+Hidden+Language+of+Computer+Hardware+and+Software%22&ots=sf8MaqPXa2&sig=GcEtNdTik9DWzRyx8vSqHxm5pX4

Polya, G. (1945). *How to solve it: A new aspect of mathematical method*. Princeton university press. https://ebookcentral-proquest-com.ezproxy.uis.no/lib/uisbib/reader.action?docID=767228

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

Rolstadås, A. (2023). Suksesskriterium. *Store norske leksikon*. https://snl.no/suksesskriterium#:~:text=Suksessfaktorer%20er%20forhold%20som%2

0m%C3%A5,gjelder%20suksessfaktorene%20under%20selve%20prosjektgjennomf%C3%B8ringen.

Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies*, *22*(2), 469–495. https://doi.org/10.1007/s10639-016-9482-0

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational research review*, *22*, 142–158.

Statped. (2021, mars 1). *Programmeringsspråk*. https://www.statped.no/laringsressurser/teknologitema/programmering-for-barn-med-saerskilte-behov/programmering/programmeringssprak/

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

Taylor, M., Harlow, A., & Forret, M. (2010). Using a computer programming environment and an interactive whiteboard to investigate some mathematical thinking. *Procedia-Social and Behavioral Sciences*, *8*, 561–570. https://doi.org/10.1016/j.sbspro.2010.12.078

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

Utdanningsdirektoratet. (2020a). *Læreplan i matematikk 1.–10. Trinn (MAT01-05)*. Fastsatt som forskrift. Læreplanverket for Kunnskapsløftet 2020. https://www.udir.no/lk20/mat01-05/kompetansemaal-og-vurdering/kv19?lang=nob

Utdanningsdirektoratet. (2020b). *Overordnet del—Verdier og prinsipper for grunnskoleopplæringen—Grunnleggende ferdigheter*. Fastsatt som forskrift. Læreplanverket for kunnskapsløftet 2020. https://www.udir.no/lk20/overordnet-del/prinsipper-for-laring-utvikling-og-danning/grunnleggende-ferdigheter/

Vinnervik, P. (2022). Implementing programming in school mathematics and technology: Teachers' intrinsic and extrinsic challenges. *International Journal of Technology and Design Education*, *32*(1), 213–242. https://doi.org/10.1007/s10798-020-09602-0

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Zhan, Z., He, L., Tong, Y., Liang, X., Guo, S., & Lan, X. (2022). The effectiveness of gamification in programming education: Evidence from a meta-analysis. *Computers and Education: Artificial Intelligence*, *3*, 100096. https://doi.org/10.1016/j.caeai.2022.100096

Figurliste

Figur 1: Utdanningsdirektoratet (2019) *Algoritmisk tenkning* [Illustrasjon]. Hentet fra https://www.udir.no/kvalitet-og-kompetanse/digitalisering/algoritmisk-tenkning/

Vedlegg

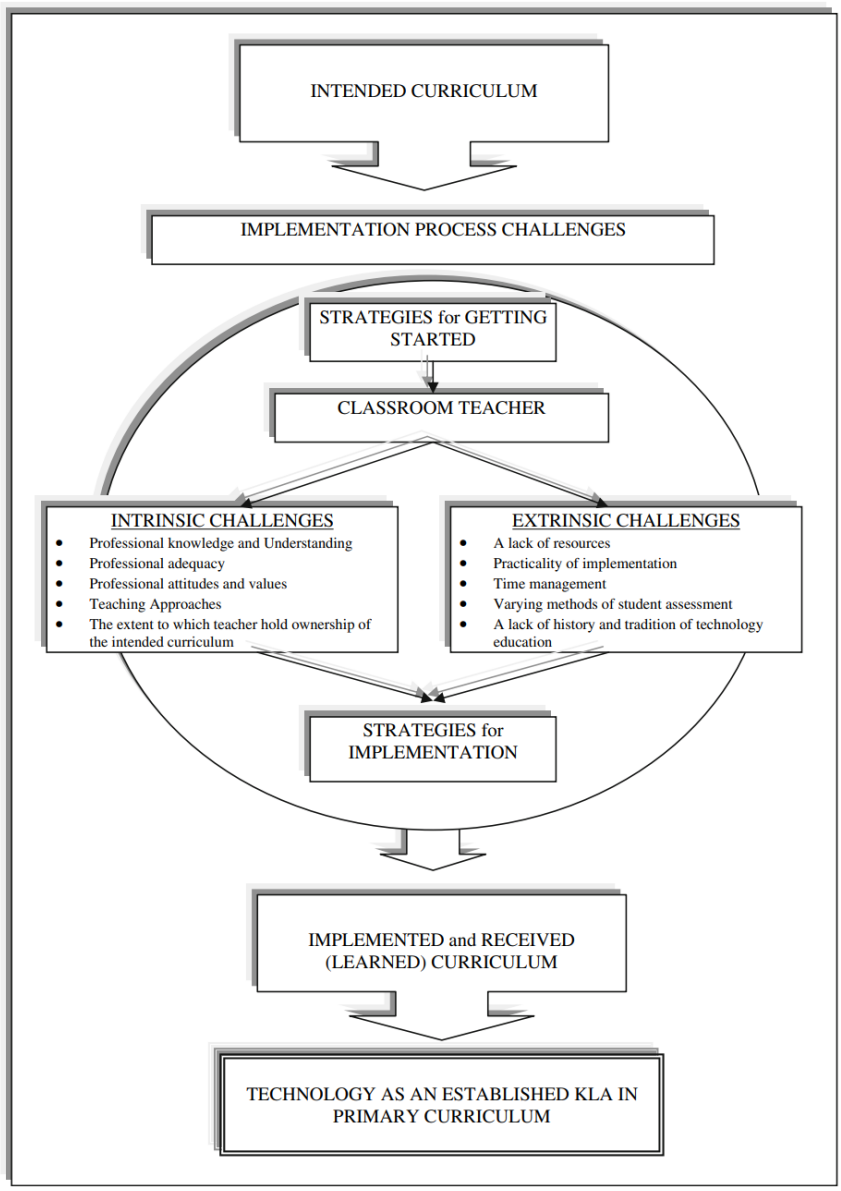
Vedlegg 1 – Vinnervik (2022)

Table 1 Intrinsic and extrinsic challenges for technology education curriculum implementation

Intrinsic challenges	Extrinsic challenges
I1. Professional knowledge and understanding	E1. Resources
I2. Professional adequacy	E2. Time management
I3. Professional attitudes and values	E3. Practicality of implementation
I4. Teaching approach	E4. Student assessment
I5. Ownership	E5. History and tradition
	E6. Professional development and support

Figur 3: Vinnervik (2022)

Vedlegg 2 – Finger & Houguet (2009)



Figur 4: Finger & Houguet (2009)

Vedlegg 3 – Søknadsgodkjenning fra sikt

Vurdering av behandling av personopplysninger

Skriv ut | 17.11.2023

Referansenummer	Vurderingstype	Dato
286578	Standard	17.11.2023

Tittel
Hvilke utfordringer opplever lærere med implementering av programmering i matematikk, og hvilke strategier har de benyttet seg av for å overkomme disse?

Behandlingsansvarlig institusjon
Universitetet i Stavanger / Fakultet for utdanningsvitenskap og humaniora / Institutt for grunnskolelærerutdanning, idrett og spesialpedagogikk

Prosjektansvarlig
Gaute Hovtun

Student
Sara-Elise Sømme

Prosjektperiode
01.12.2023 - 31.12.2024

Kategorier personopplysninger
Alminnelige

Lovlig grunnlag
Samtykke (Personvernforordningen art. 6 nr. 1 bokstav a)

Behandlingen av personopplysningene er lovlig så fremt den gjennomføres som oppgitt i meldeskjemaet. Det lovlige grunnlaget gjelder til 31.12.2024.

Meldeskjema

Kommentar
OM VURDERINGEN
Sikt har en avtale med institusjonen du forsker eller studerer ved. Denne avtalen innebærer at vi skal gi deg råd slik at behandlingen av personopplysninger i prosjektet ditt er lovlig etter personvernregelverket. Vi har nå vurdert at du har lovlig grunnlag til å behandle personopplysningene.

TAUSHETSPLIKT
Forskningsdeltagerne har yrkesmessig taushetsplikt. De kan ikke dele taushetsbelagte opplysninger med forskningsprosjektet. Vi anbefaler at du minner dem på taushetsplikten. Merk at det ikke er nok å utelate navn ved omtale av elever, pasienter el. Vær forsiktig med bruk av eksempler og bakgrunnsopplysninger som tid, sted, kjønn og alder.

FØLG DIN INSTITUSJONS RETNINGSLINJER
Det er institusjonen du er ansatt/student ved som avgjør hvordan du må lagre og sikre data i ditt prosjekt og hvilke databehandlere du kan bruke. Husk å bruke leverandører som din institusjon har avtale med (f.eks. ved skylagring, nettspørreskjema, videosamtale el.).

Personverntjenester legger til grunn at behandlingen oppfyller kravene i personvernforordningen om riktighet (art. 5.1 d), integritet og konfidensialitet (art. 5.1. f) og sikkerhet (art. 32).

MELD VESENTLIGE ENDRINGER
Dersom det skjer vesentlige endringer i behandlingen av personopplysninger, kan det være nødvendig å melde dette til oss ved å oppdatere meldeskjemaet. Se våre nettsider om hvilke endringer du må melde: https://sikt.no/melde-endringar-i-meldeskjema

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

Lykke til med prosjektet!

Vedlegg 4 – Informasjonsskriv og samtykkeskjema

Vil du delta i forskningsprosjektet

«Hvilke utfordringer opplever lærere med implementering av programmering i matematikk, og hvilke suksessfaktorer har dem benyttet seg av for å imøtekomme utfordringene?»

Dette er et spørsmål til deg om å delta i et forskningsprosjekt hvor formålet er å kartlegge utfordringer og suksessfaktorer benyttet for å imøtekomme utfordringene med implementering av programmering innenfor matematikk. I dette skrivet gir vi deg informasjon om målene for prosjektet og hva deltakelse vil innebære for deg.

Formål

Formålet med prosjektet er å kartlegge utfordringer og løsningsstrategier lærere har benyttet seg av som har ført til suksess med implementering av programmering innenfor matematikkfaget. Programmering ble en sentral del i matematikken i LK 20 og flere lærere har møtt på flere utfordringer rundt implementeringen. Prosjektet skal derfor kunne spre budskap til lærere som selv kjenner dem sliter med implementeringen av programmering og ønsker noen konkrete tips og metoder som kan hjelpe dem med akkurat dette.

Dataen som samles skal kunne brukes i å analyse problemstillingen «hvilke utfordringer opplever lærere med implementering av programmering i matematikk, og hvilke suksessfaktorer de har benyttet for å imøtekomme utfordringene?» i en masteroppgave.

Hvem er ansvarlig for forskningsprosjektet?

Fakultet for utdanningsvitenskap og humaniora / institutt for grunnskolelærerutdanning, idrett og spesialpedagogikk er ansvarlig for prosjektet.

Hvorfor får du spørsmål om å delta?

Aktuelle intervjukandidater er lærere som har erfaring med programmering i matematikk på 5-10 trinn etter den nye læreplanen LK20 ble igangsatt.

Hva innebærer det for deg å delta?

Hvis du/ dere velger å delta i prosjektet innebærer det et intervju på 45-60 min hvor det vil bli stilt spørsmål ut ifra en intervjuguide. Intervjuguiden vil inneholder spørsmål angående utfordringer og suksessfaktorer benyttet for å imøtekomme utfordringene med implementering av programmering i matematikk. Det vil bli tatt lydopptak av intervjuet.

- **Vårt personvernombud**: Ann Kristin Kolstø-Johansen, ann.k.kolsto-johansen@uis.no, 51832991

Hvis du har spørsmål knyttet til vurderingen som er gjort av personverntjenestene fra Sikt, kan du ta kontakt via:

- Epost: personverntjenester@sikt.no eller telefon: 73 98 40 40.

Med vennlig hilsen

Gaute Hovtun
Veileder

Sara-Elise Sømme
Masterstudent

Samtykkeerklæring

Jeg har mottatt og forstått informasjon om prosjektet Hvilke utfordringer opplever lærere med implementering av programmering i matematikk, og hvilke strategier har dem benyttet seg av for å overkomme disse ,og har fått anledning til å stille spørsmål. Jeg samtykker til:

- ☐ å delta i gruppeintervju med lydopptak
- ☐ at mine personopplysninger lagres frem til prosjektslutt den 03.juni 2024

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

---

(Signert av prosjektdeltaker, dato)

Det er frivillig å delta

Det er frivillig å delta i prosjektet. Hvis du velger å delta, kan du når som helst trekke samtykket tilbake uten å oppgi noen grunn. Alle dine personopplysninger vil da bli slettet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg.

Ditt personvern – hvordan vi oppbevarer og bruker dine opplysninger

Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler opplysningene konfidensielt og i samsvar med personvernregelverket.

- De som har tilgang til data materialet vil kun være meg og ingen andre.
- Navnet og kontaktopplysningene dine vil jeg erstatte med en kode som lagres på egen navneliste adskilt fra øvrige data. Data som inneholder personvern opplysninger vil bli lagret i en kryptertsky «nextcloud».
- Du vil ikke kunne bli gjenkjent i publikasjon av denne masteren

Hva skjer med personopplysningene dine når forskningsprosjektet avsluttes?

Prosjektet vil etter planen avsluttes når masteren er bestått som er 03.juni 2024. Etter prosjektslutt vil datamaterialet med dine personopplysninger anonymiseres. Dette vil si at alle videoklipp og lydopptak vil bli slettet.

Hva gir oss rett til å behandle personopplysninger om deg?

- Vi behandler opplysninger om deg basert på ditt samtykke.

På oppdrag fra Fakultet for utdanningsvitenskap og humaniora / institutt for grunnskolelærerutdanning, idrett og spesialpedagogikk har Sikt – Kunnskapssektorens tjenesteleverandør vurdert at behandlingen av personopplysninger i dette prosjektet er i samsvar med personvernregelverket.

Dine rettigheter

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

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

Hvis du har spørsmål til studien, eller ønsker å vite mer om eller benytte deg av dine rettigheter, ta kontakt med:

- Fakultet for utdanningsvitenskap og humaniora / institutt for grunnskolelærerutdanning, idrett og spesialpedagogikk:
    - Masterstudent: Sara-Elise Sømme, sa.somme@stud.uis.no, 93630231
    - Veileder: Gaute Hovtun, gaute.hovtun@uis.no, 51833483

Vedlegg 5- Intervjuguide

Indre utfordringer og strategier

Hvilke indre utfordringer har du opplevd med innføring av programmering i matematikkundervisningen? Med indre utfordringer mener jeg utfordringer som har røtter i deg seg, noe som du selv kan finne en løsning til. Har du noen suksesshistorier? Noen indre utfordringer som du/ dere har funnet en løsning til?

1. Når det kommer til hele prosessen rundt forberedelser til programmeringstimer i matematikk. Dette innebærer for eksempel planlegging av undervisning. Opplever du/ dere at du/ dere har tilstrekkelig med kunnskap og forståelse til å undervise programmering i matematikk? Er det noe du ønsker du hadde mer kunnskap om?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

2. Har dere møtt på utfordringer som er knyttet til egen usikkerhet i møte med programmering i matematikk? *(Professional adequacy)*
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

3. Hvilke utfordringer har du/ dere opplevd med egen /kollegers holdninger og verdier til programmering innenfor matematikk?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

4. Hvilke utfordringer har du/ dere opplevd med undervisningstilnærming?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

5. Hvilke utfordringer har du/ dere opplevd med eierskap til faginnholdet i en programmeringstime innenfor matematikk?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

Ytre utfordringer og strategier

Hvilke ytre utfordringer har du opplevd med innføring av programmering i matematikkundervisningen? Med ytre utfordringer tenker jeg på utfordringer som har røtter i det eksterne, noe du ikke har kontroll over eller kan gjøre noe med? Har du noen suksesshistorier? Noen utfordringer som du/ dere har funnet en løsning til?

6. Hvilke utfordringer har du/ dere møtt på når det gjelder resurser?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

7. Hvilke utfordringer har du/ dere opplevd med tidshåndtering?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

8. Hvilke ytre utfordringer har du/ dere opplevd med den praktiske siden av implementeringen?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

9. Hvilke utfordringer har du/ dere opplevd med elev evaluering?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

10. LK20 er det andre forsøket på å innføre programmering i norskeskoler. Første forsøket skjedde på 80 tallet i sammenheng med Seymore Paperet (Koschmann, 1997), hvor forsøket døde ut i løpet av 90 tallet. Har dere møtt på utfordringer som kan ha røtter fra 80 tallet? Hvilke utfordringer har du/ dere opplevd med programmeringens historie og tradisjon innenfor matematikk?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?

11. Hvilke utfordringer har du/ dere opplevd med profesjonell utvikling og støtte?
    a. Hvordan håndterte du/ dere utfordringen om dette?
    b. Kan du utdype mer om …
    c. Hva er det som hindrer i å finne en løsning?