Berg

# Analog programmering

## Innledning

I denne artikkelen vil jeg undersøke hvilke fordeler elever og lærere opplever ved å inkludere analog programmering i opplæringen av programmering innenfor matematikk. Analog programmering betyr at en programmerer uten datamaskin. Datamaterialet mitt er hentet fra intervju med en lærer og spørreundersøkelse i en gruppe på 8. trinn der elevene i forkant hadde arbeidet på denne måten. I undervisningsopplegget hadde de fått innføring i for-løkker og brukt det til å forme geometriske figurer ved tekstbasert programmering.

Programmering er inkludert i fagfornyelsen i matematikk fra 1. til 10. trinn (Utdanningsdirektoratet, 2019a). Deler av argumentasjonen for programmeringens plass er at programmering trener elevene på algoritmisk tenkning[1], som blant annet innebærer å analysere informasjon, lage regler og steg-for-steg-fremgangsmåter, og bryte problemer ned i mindre deler (Utdanningsdirektoratet, 2019b). Et annet argument for programmeringens plass i fagfornyelsen er dens overførbarhet til andre kognitive ferdigheter som problemløsing, resonnering, kreativitet og matematiske ferdigheter (Scherer et al., 2018). Sist, men ikke minst har ferdigheter innenfor programmering en nytteverdi i seg selv. For å fungere i dagens samfunn omgir vi oss med ulike elementer som er programmert. Vi trenger kunnskap om programmering både for å forstå hvordan disse fungerer, og for å videreutvikle teknologi i fremtiden.

Det finnes ulike tilnærminger til hvordan programmering kan implementeres i skolen, både digitalt og analogt. Analog programmering refererer til «unplugged programming» (Humble et al., 2019a; 2019b), som i korte trekk går ut på å lage en fremgangsmåte eller løse et problem uten å bruke digitale hjelpemidler. Analog programmering finnes i mange ulike former, det kan være å styre en brikke ved hjelp av koder på et brettspill, programmere medelever til å utføre spesifikke oppgaver, eller pusle et puslespill uten at en selv ser puslespillet (Aranda & Ferguson, 2018).

Internasjonal forskning som undersøker hvordan lærere og elever i ungdomsskolen oppfatter analog programmering (Bell & Vahrenhold, 2018; Cortina, 2015; Humble et al., 2019a),

**Tonje Karoliussen Berg**
Universitetet i Sørøst-Norge
tonje.k.berg@usn.no

Dette er en fagfellevurdert artikkel på nivå 1. Tangenten er et sted der læreres og forskeres perspektiv på matematikkundervisning møtes og derfor har vi med praksisrelaterte forskningsartikler. Les mer i retningslinjene: www.caspar.no/nivaa1



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

indikerer at både elever og lærere ser fordeler ved analog programmering. Tilnærmingen gir mulighet til å trene algoritmisk tenkning (Bell & Vahrenhold, 2018; Cortina, 2015; Humble et al., 2019a), og den kan gi økt motivasjon for programmering (Bell & Vahrenhold, 2018; Alamer et al., 2015). I Norge er det gjort studier knyttet til læreres og elevers oppfatning av programmering knyttet til digitale plattformer, som Scratch (Giannakos et al., 2013). Det samme er imidlertid ikke gjort for analog programmering, og det er derfor interessant å undersøke om lærere og elever innenfor det norske skolesystemet har samme oppfatning knyttet til analog programmerings mulige fordeler som internasjonale studier indikerer.

I denne artikkelen undersøker jeg elevers og læreres oppfatninger rundt bruk av analog programmering i ungdomsskolen. Læreres og elevers oppfatning er viktig for å danne et bilde av hvordan de oppfatter eksisterende opplæring innenfor programmering, samt hvilken opplæring de tror er fordelaktig i norsk skole. Dette kan gi grunnlag for å utvikle didaktiske metoder til bruk i opplæringen. Jeg søker svar på følgende problemstilling: Hvilke fordeler erfares ved bruk av analog programmering i undervisning på 8. trinn?

## Teori

I det følgende viser jeg til forskning som belyser læreres og elevers oppfatning av fordeler ved analog programmering, og som samtidig legger vekt på læringsstiler, algoritmisk tenkning og motivasjon og mestring. Jeg ser på forskning som kan gi innsikt i analog programmerings betydning for (1) bruk av ulike læringsstiler i undervisningen om programmering, (2) utvikling av algoritmisk tenkning og (3) påvirkning på motivasjon og mestring blant elever. I tillegg knyttes relevant teori til de tre kategoriene.

I en svensk studie viser Humble et al. (2019a) at lærere i etterutdanning var positive til analog programmering i sin undervisning. Majoriteten av lærerne mente det var hensiktsmessig å bruke analog programmering som introduksjon og i kombinasjon med andre typer programmeringsaktiviteter. Samtidig viste lærerne usikkerhet knyttet til når denne formen for programmering kan brukes. Mange mente at det var mest aktuelt på lavere klassetrinn og for elever med begrenset kunnskap om programmering. Atlas et al. (2017) bekrefter disse funnene.

### Læringsstiler

Et argument for at analog programmering bør inkluderes i undervisningen, er at arbeidsformen bidrar til at ulike læringsstiler blir benyttet. Noen elever lærer best gjennom hva de hører (auditivt), andre ved å berøre og arbeide med konkreter (taktile). Noen husker best det de selv har sett (visuelle), mens andre har behov for fysisk aktivitet og egne praktiske erfaringer (kinestetiske) (Maugesten & Olafsen, 2017). Maugesten og Olafsen argumenterer for at en gjennom bruk av ulike læringsstiler kan få muligheter til å oppfylle kravene om tilpasset opplæring. Også Cortina (2015) fremhever at analog programmering kan bidra til at undervisningen treffer andre elever enn det digital programmering gjør. Ved å bruke analog programmering i undervisningen kan man tilpasse opplæringen gjennom taktile, auditive, visuelle og kinestetiske arbeidsformer. Noen forskere, for eksempel Bell og Vahrenhold (2018), bekrefter disse fordelene med analog programmering, og argumenterer for at opplæringen innenfor programmering bør inkludere både analog og digital programmering, samt at en kan trekke paralleller mellom disse. Ifølge Bell og Vahrenhold (2018) gjør analog programmering det mulig å også jobbe med programmering uten at elevene har tekniske ferdigheter innenfor programmering.

### Algoritmisk tenkning

Algoritmisk tenkning går blant annet ut på å identifisere og forstå et problem, kunne lage og implementere en algoritme for å løse problemet, samt kunne evaluere om løsningen er optimal



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

med tanke på problemet (Thomas et al., 2017). Det er enighet om at mange av ferdighetene innenfor algoritmisk tenkning kan trenes opp ved hjelp av arbeid med analog programmering (Bell & Vahrenhold, 2018; Cortina, 2015). Bell og Vahrenhold (2018) skriver at analog programmering i forkant av digital programmering er et lovende pedagogisk grep for å utvikle algoritmisk tenkning hos elever. Det kan hjelpe elever til å tenke nøye gjennom kodene de bruker, og hva de ulike kodene gjør, før de anvender koder på datamaskin. Dette kan gå tapt dersom man begynner direkte på den digitale programmeringen, fremholder Bell og Vahrenhold (2018). Humble et al. (2019) fant at lærere uttrykker tro på at analog programmering trener elevene på å vise og forklare en tenkt prosess, bryte ned et problem i mindre delproblemer og forklare og strukturere et matematisk konsept om til en kode. Aranda og Ferguson (2018) viser også til at datamaskinen i noen tilfeller kan bli en distraksjon underveis i læringen. Hvilke aspekter innenfor algoritmisk tenkning som trenes ved analog programmering, vil være avhengig av hvilken type oppgaver man jobber med, men en stor del av analoge programmeringsoppgaver er knyttet til problemløsing (Cortina, 2015). Analog tilnærming har med stor suksess blitt brukt i tilknytning til innlæring av spesifikke programmeringskonsept, for eksempel innlæring av løkker (Bell & Vahrenhold, 2018).

## Motivasjon og mestring

Teoretisk kan en skille mellom indre og ytre motivasjon. I denne artikkelen vil jeg kun gå inn i indre motivasjon fordi jeg er ute etter hvorvidt oppgaven i seg selv kan skape motivasjon for elevene. For å oppnå indre motivasjon må elever og lærere oppleve oppgaven som interessant og morsom. Elever som er indre motivert, setter i gang med oppgaver på egen hånd, foretrekker utfordrende oppgaver, stiller spørsmål, er utholdende, gjør mer enn det som kreves, og opplever glede ved å løse oppgaver (Wæge & Nosrati, 2018). Fordi «elevenes indre motivasjon har en tendens til å minke med økende alder i skolen generelt og i matematikk spesielt» (Wæge & Nosrati, 2018, s. 21), er det viktig å fokusere på oppgaver innenfor programmering som er motiverende, også på høyere klassetrinn. Indre motivasjon kan påvirkes av elevenes opplevelse av mestring, og analog programmering kan spille en sentral rolle. Mestring innenfor matematikk innebærer blant annet å mestre en matematikkoppgave, kunne stille matematiske spørsmål, forklare og forstå løsningsstrategier, argumentere, resonnere og forklare og bruke matematiske begreper (Wæge & Nosrati, 2018). En forutsetning for elevenes opplevelse av mestring er at de blir utfordret på et optimalt nivå. Utfordringer som er for vanskelige, kan føre til frustrasjon, mens for enkle oppgaver kan føre til kjedsomhet (Wæge & Nosrati, 2018). Hvorvidt elevene opplever mestring knyttet til arbeidet med oppgaven, er avgjørende for om den analoge programmeringsaktiviteten fører til motivasjon for oppgaven og i beste fall motivasjon for programmering generelt.

Bell og Vahrenhold (2018) og Alamer et al. (2015) fremhever at analog programmering kan skape engasjement og øke motivasjon for programmering generelt både hos lærere og elever. Bell og Vahrenhold (2018) og Cortina (2015) har i sine studier funnet at om undervisningen inneholder både analog og digital programmering, kan flere elevers motivasjonen for programmering øke enn om man utelukkende bruker digital programmering. Dette samsvarer med Alamer et al. (2015) sine funn, der jenter i ungdomsskole og videregående skole opplevde analog programmering som motiverende og morsomt.

## Metode

### Forskningsdesign

Innsamlingen av data er gjort hjelp av kvalitative undersøkelser. Jeg ønsket å samle mye informasjon om lærerens oppfatning knyttet til analog programmering, og gjorde derfor et semistrukturert intervju med læreren. Intervjuet varte i 30 minutter, der jeg som intervjuer



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

www.caspar.no/tangenten

tok notater underveis i samtalen. Det var ikke mulig å gjennomføre observasjon siden skolen var nedstengt som følge av covid-19. Av tidsmessige årsaker ble det valgt en anonym spørreundersøkelse for å hente inn informasjon fra elevene. De svarte på et digitalt spørreskjema, der to av spørsmålene var fritekstsvar, resten var avkrysningsspørsmål. Denne undersøkelsen ble gjennomført i etterkant av tredje og fjerde undervisningsøkt.

Avkrysningsspørsmålene innebar at elevene tok stilling til ulike påstander som var formulert både positivt og negativt ladet. Påstandene handlet i stor grad om hvordan elevene oppfattet den analoge delen av undervisningsopplegget, hvilket læringsutbytte de hadde, og hvordan den analoge delen forberedte dem på den digitale programmeringsøkten. Noen av spørsmålene handlet direkte om hvordan de oppfattet analog programmerings virkning på deres motivasjon, læringsstil og arbeid med algoritmisk tenkning. Avkrysningsskjema ga rom for ulike synspunkter knyttet til analog programmering ved at elevene svarte på spørsmålene «Hva var bra med undervisningsopplegget du akkurat gjennomførte? Hvorfor?». Spørreskjemaet erstattet begrepet analog programmering med «programmering for hånd», fordi læreren brukte denne betegnelsen da hun beskrev analog programmering. Alle elevene svarte på begge skjemaene.

Jeg ville legge til rette for at informanten fritt kunne gi uttrykk for sine tanker knyttet til analog programmering og startet derfor Intervjuguiden med åpne spørsmål. Den ble også brukt som et veiledende dokument underveis i intervjuet. Noen av oppfølgingsspørsmålene ble direkte knyttet til kategoriene læringsstiler, motivasjon og algoritmisk tenkning. Disse spørsmålene ble stilt i etterkant av at læreren hadde nevnt fordeler ved disse. Det kan være en svakhet at det ikke ble gjort opptak av intervjuet, men bare dokumentert ved notater underveis. Føring av notater kan gjøre flyten i intervjuet dårligere (Thagaard, 2010). Samtidig kan informanter bli påvirket av ikke-menneskelige faktorer, som lydopptaker, og bli mer bevisst på hvordan de ordlegger seg (Brinkmann & Kvale, 2015). I denne studien vurderte jeg det som viktigere at informanten formulerte seg fritt, enn å bygge analyser på informantens ordrette formuleringer.

**Bakgrunn for undersøkelsene**

Undervisningsopplegget ble gjennomført i en klasse med 21 elever. Klassens matematikklærer gjennomførte undervisning med implementering av for-løkker som mål. For-løkker er en operasjon der en gjentar en kommando et gitt antall ganger (Bueie, 2019). Denne typen løkker er hjelp til å refaktorere en kode. Det innebærer å omstrukturere en kode for å gjøre den enklere, noe som gjør det lettere å vedlikeholde og videreutvikle koden. Elevene skulle refaktorere koder av geometriske figurer. De hadde kunnskap om programmering, de hadde blant annet formet geometriske figurer ved tekstbasert koding, og jobbet litt med hvis-setninger. De hadde også tidligere arbeidet med analog programmering. Undervisningsopplegget ble gjennomført som fjernundervisning med tidsbruk som vist i Tabell 1.

| Undervisningsøkt | Tid | Opplegg |
|---|---|---|
| 1. time | 30 minutter | Introduksjonsvideo av for-løkker |
| 2. time | 60 minutter | Analogt undervisningopplegg |
| 3. time | 60 minutter | Fortsettelse av analogt undervisningsopplegg. Oppsummering |
| 4. time | 60 minutter | Digitalt undervisningsopplegg |

Tabell 1: Tidsbruk i undervisningsopplegget.



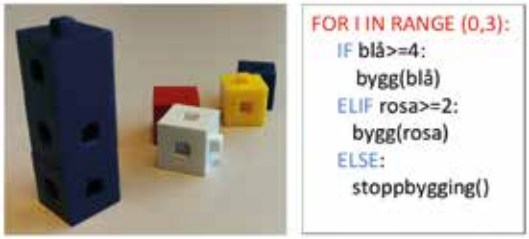
Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

Første undervisningsøkt introduserte for-løkker ved hjelp av hverdagslige eksempler (se figur 1) og Python-kode. Det analoge undervisningsopplegget gjennomførte elevene i digitale par der de tolket en utdelt pseudokode. En pseudokode er en uformell og hverdagslig kode av en algoritme (Dvergsdal, 2019). Ved hjelp av hvis-setninger og for-løkker gir pseudokoden en oppskrift på hvordan elevene skal bygge et tårn av multilinks (se figur 2).

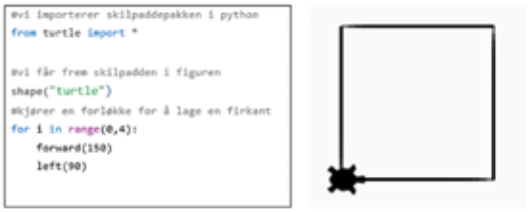
Som et forsøk på å kompensere for hjelpen læreren ville gitt i klasserommet, så elevene videoer med løsningsforslag underveis. I fjerde undervisningsøkt arbeidet de med å lage geometriske figurer digitalt via tekstbasert programmering for så å refaktorere dem ved hjelp av for-løkker (se figur 3).

```
GJENTA 52 ganger
    GJENTA 5 ganger
        —> spise frokost kl 07.30
    GJENTA 2 ganger
        —> spise frokost kl 09.00
```

Figur 1: Ett av de hverdagslige eksemplene som ble vist til elevene ved introduksjonen av for-løkker.




Figur 2: Elevene fikk et visst antall multilinks og bygget det blå tårnet ved å følge pseudokoden til høyre.




Figur 3: Til venstre er koden for et kvadrat, laget ved en for-løkke. Til høyre er figuren koden lager.

## Bearbeiding av data

Notatene fra intervjuet med læreren ble renskrevet rett etter innsamling for å få med detaljer. For å analysere dataene fra spørreskjemaene ble det laget tabeller for hvert spørsmål der prosentandelen med de ulike svarene kom opp. Disse tabellene gir et helhetsinntrykk av hvordan klassen oppfatter analog programmering. Samtidig ble enkeltbesvarelser gjennomgått for å se hvordan de ulike svarene samsvarte med hverandre.

Sortering av data og drøfting av resultat ble systematisert ved bruk av kategoriene læringsstiler, algoritmisk tenkning og motivasjon. Disse kategoriene var ikke fastlåste, men fungerte som et utgangspunkt for drøftingen. Funn som ikke ble dekket av disse kategoriene, ble også brukt i drøftingen. Presentasjon og drøfting av resultater tok i stor grad for seg dataene fra første avkrysningsskjema med elevene, samt intervju med læreren. Disse dataene viste seg mest relevante for å svare på problemstillingen. Der data fra spørreundersøkelsen etter den digitale økten er brukt, er dette oppgitt.

## Resultater og diskusjon

I denne delen redegjør jeg for og drøfter hvilke fordeler elevene og læreren ser ved analog programmering etter endt undervisningsopplegg. Resultatene sammenliknes med tidligere forskning, og jeg tar utgangspunkt i kategoriene: læringsstiler, algoritmisk tenkning og motivasjon og mestring. Hver kategori er strukturert slik at lærerens syn blir presentert før elevenes.

## Bruk av ulike læringsstiler

Læreren sier at analog programmering kan skape variasjon i matematikkundervisningen, og at det gir henne muligheter til å treffe flere elever med sitt undervisningsopplegg. Videre sier læreren at hun tror dette undervisningsopplegget i stor grad kan treffe elever med taktil læringsstil på en hensiktsmessig måte, og at dette gjelder analog programmering generelt. Lærerens oppfatninger kan sees i lys av



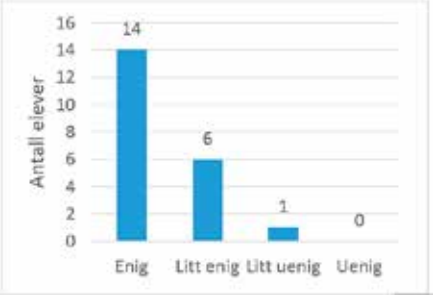
Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

Maugesten og Olafsens (2017) oppfordring om å etterstrebe en undervisning som inkluderer ulike læringsstiler for å tilpasse undervisningen til enkeltelever. Samtidig gir læreren uttrykk for at hun har erfart at det er hensiktsmessig å la innføringen av et emne være analogt, slik at elevene får tid til å forstå et tema innenfor programmering på en mer hverdagslig og helhetlig måte, før de begynner arbeidet digitalt. Læreren fremhever at dette vil gagne elever med lite mestring innenfor programmering. Dette kan sees i sammenheng med tidligere funn om at analog programmering kan bidra til at elever med lite tekniske ferdigheter i programmering likevel kan trene på å mestre aspekter innenfor programmering (Bell & Vahrenhold, 2018). Dette kan sees på som en form for tilpasset opplæring der en benytter seg av flere ulike læringsstiler. Læreren beskriver at hun vil bruke analog programmering som en integrert del av programmeringsopplæringen, hun vil veksle mellom å jobbe digitalt og analogt.

Læreren reflekterer også over hvorvidt analog programmering er aktuelt for å oppnå bruk av ulike læringsstiler i undervisningen i ungdomsskolen fremover. Elevene blir flinkere på det tekniske innenfor programmering, og hun tror det er mulig at analog programmering derfor vil utebli. Dette kan tolkes som at læreren har en forståelse av at analog programmering er aktuelt innenfor den grunnleggende opplæringen av programmering, noe som samsvarer med tidligere forskning (Humble et al., 2019a). Dette kan også tyde på at hun ikke synes fordelene ved analog programmering knyttet til variasjon av læringsstiler i undervisningen er så hensiktsmessig som først antatt. Læreren utelukker ikke at variasjon av læringsstilene også kan oppnås ved digital programmering.

Elevene underbygger at analog programmering kan skape variasjon i matematikkundervisningen ved at 20 elever bekrefter at de er enig eller litt enig i at oppgaven med programmering for hånd var en annerledes time enn vanlige matematikktimer (se figur 4). Når en elev skriver at «det var gøyere enn vanlig matte», kan det tyde på at det for denne eleven også skapte motivasjon. Det fremgår ikke hva eleven legger i «vanlig matte». Det kan være timer preget av stor variasjon i aktiviteter samt bruk av mange ulike læringsstiler.

Halvparten av elevene ville gått rett på programmering på data, uten å ha oppgaven med programmering for hånd først, mens den andre halvparten ville gjennomført det analoge opplegget også sett i etterkant. Denne tilbakemeldingen forteller at elevene foretrekker ulike fremgangsmåter, og kanskje at de også har ulike læringsstiler. De ulike svarene kan tyde på at man treffer flere ved å benytte seg av analoge undervisningsopplegg enn om man kun velger de digitale.




Figur 4: Elevsvar på påstanden «Oppgaven med programmering for hånd var en annerledes time enn vanlige matematikktimer».

## Utvikle algoritmisk tenkning

Læreren trekker frem at man gjennom det analoge undervisningsopplegget trener på leseferdigheter, og innenfor programmering uttrykker hun at dette handler om evnen til å lese en kode. Hun opplever at elevene blir «tvunget» til å bearbeide selve koden grundigere når de gjennomfører en analog programmeringsøkt enn en digital. Hun underbygger dette ved at elevene selv må komme frem til hva resultatet av en kode er, mens de ved digital programmering kan teste ut koden i større grad underveis i prosessen. Lesing av koden sammenlikner hun



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

med lesing av en bruksanvisning, som kan relateres til en algoritme. Det dreier seg om å lese en algoritme, forstå den og forutse hva den vil utrette, noe som er en tydelig del av ferdighetene innenfor algoritmisk tenkning. Dette kan sees i lys av Bell og Vahrenholds (2018) funn, der de tydeliggjør hvilket lovende pedagogisk grep det er å bearbeide kode på en grundig måte i opplæringen av programmering. I tillegg forteller læreren at det «gikk opp et lys» for mange elever underveis i undervisningsopplegget. Læreren uttrykker at hun tror denne grundige forståelsen handler om at de må bruke konkreter til å lage noe fysisk ut av en kode, noe også Bell & Vahrenhold (2018) understreker.

I forlengelse av dette redegjør læreren for hvordan opplegget også trener på teknisk koding ved at elevene ser en ferdig kode og tolker den gjentatte ganger. Teknisk koding kan i dette undervisningsopplegget forstås som hvordan man lager en for-løkke ved programmeringsspråket Python. Hun sier elevene i stor grad forsto hvordan intervallene en for-løkke avgrenser, fungerer, og at de forsto og kunne forestille seg at variabelen i løkken alltid legger på én hver gang den kjører. Læreren tror at elevene i større grad bruker for-løkker og variabler uten nødvendigvis å forstå logikken som ligger bak, ved digital programmering enn ved analog programmering. Denne forståelsen samsvarer med at analog programmering bidrar til innlæring av spesifikke programmeringskonsepter (Bell & Vahrenhold, 2018), som i dette tilfellet er for-løkker.

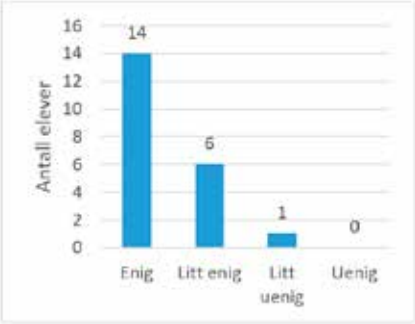
Senere i intervjuet trekker læreren også inn begrepet algoritmisk tenkning som en fordel ved det analoge undervisningsopplegget. Algoritmisk tenkning konkretiserer hun ved å trekke frem ferdigheter som at elevene trener på å bryte ned problemer i mindre biter, for så å forstå hvert delproblem – dette kalles dekomposisjon. Lærerens oppfatning av at analog programmering kan bidra til dekomposisjon, samsvarer med funnene til Humble et al. (2019a), som sier at analog programmering kan føre til at elevene må dele opp problemer i mindre delproblemer før de kan løse dem. Dekomposisjonen gjøres i undervisningsopplegget ved at elevene skal forstå ulike deler av koder før den til slutt forstår hele programmet. Læreren trekker også frem algoritmisk tenkning når det fokuserer på de delene av programmet som er essensielt for å utføre koden, der en fjerner unødvendig informasjon. Hun sier man trener på dette i undervisningsopplegget ved å fjerne overflødig informasjon fra de betingelsene i en hvis-setning som ikke oppfylles.

En kan stille spørsmål ved om denne formen for trening på algoritmiske ferdigheter ikke kan oppnås via digital programmering. Læreren som har deltatt i studien, er opptatt av at man jobber med algoritmisk tenkning på en annen måte ved analog programmering enn ved digital programmering. Hun begrunner dette ved å vise til at elevene får bruke mer tid på å forstå prinsippene bak koden, og med at de selv må skape resultatet av koden de tolker. Det står i motsetning til digital programmering, der resultatet av koden kommer opp på en skjerm, uavhengig av om eleven egentlig har forstått hva koden sier. Læreren mener elevene går gjennom en modningsprosess ved at de får bruke mye tid på å tolke koden og bygge den fysisk. Det samme understrekes hos Bell & Vahrenhold (2018), som skriver at analog programmering bidrar til å kunne lese og forstå digital kode, noe som kan tolkes i retning av at datamaskinen kan oppfattes som en distraksjon for elevene, heller enn at den er et nødvendig hjelpemiddel. Læreren sier også at ikke alle elever nødvendigvis har behov for denne formen for undervisning, og hun peker spesielt på elevene som mestrer programmering godt, men hun antyder samtidig at den analoge arbeidsmetoden gjør at flere av elevene i klassen forstår prinsippene bak algoritmisk tenkning. Det er viktig å understreke at det finnes avansert digital programmering som relativt enkelt kan illustreres ved hjelp av konkreter, og gjennom dette bidra til at elevene som mestrer programmering godt, kan ha nytte av analog programmering.

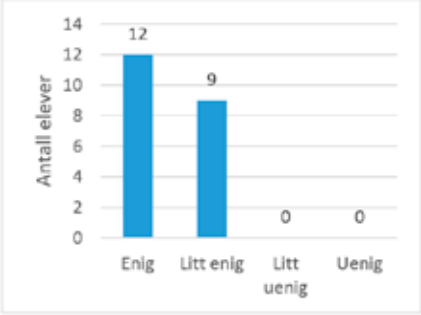


Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

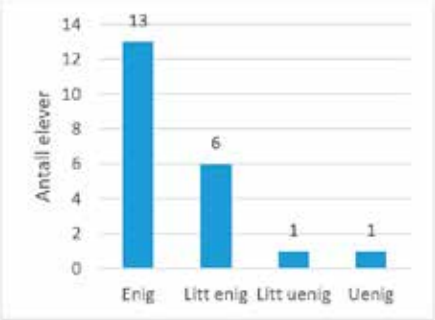
www.caspar.no/tangenten



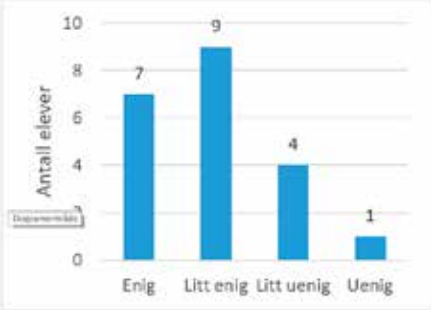

Figur 5: Elevsvar på påstanden «Programmering for hånd gjorde at jeg forsto hvordan for-løkker fungerer».




Figur 7: Elevsvar på påstanden «Programmering for hånd kan være en hjelp til å forstå prinsipper innenfor digital programmering».




Figur 6: Elevsvar på påstanden «Programmering for hånd gjorde at jeg forsto hvordan hvis-setninger fungerer».




Figur 8: Elevsvar på påstanden «Programmeringen for hånd gjorde at jeg forsto programmering på data bedre».

Lærerens oppfatninger av at analog programmering kan bidra til å utvikle algoritmisk tenkning, bekreftes av elevene. Av totalt 21 elever svarer henholdsvis 20 og 19 av disse at de er enig eller litt enig i at programmering for hånd gjorde at de forsto hvordan for-løkker og hvis-setninger fungerer (figur 5 og figur 6). Dette handler om å kunne lese en algoritme og analysere og forutsi hva som skjer når koden kjøres. Etter den analoge økta var alle elevene enige eller litt enige i at programmering for hånd kan være en hjelp til å forstå prinsipper innenfor digital programmering (figur 7). Etter endt digital økt var 16 av 21 elever enige eller litt enige i at programmeringen for hånd gjorde at de forsto programmering på data bedre (figur 8). Dette kan underbygges ved disse skriftlige tilbakemeldingene på det analoge undervisningsopplegget:

> «Det var artig å se hvordan for-løkker fungerte. Jeg lærte mye av å gjøre det for hånd, fordi det var lettere å forstå.»
> «Jeg lærte mer om hvordan programmering fungerer.»

Besvarelsene bekrefter at flertallet i denne elevgruppa tror at undervisningsopplegget førte til at de forsto prinsippene bak for-løkker og hvis-setninger bedre.



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

### Økt motivasjon og mestringsfølelse

Et av de viktigste resultatene den analoge undervisningsøkten gav, var ifølge læreren at elevene fikk motivasjon for programmering i sin helhet da de klarte oppgavene. Begrunnelsen for dette er at aktiviteten var lagt opp slik at de tekniske ferdighetene innenfor programmering ikke er like nødvendige som ved digital programmering. På denne måten kan flere elever ha forutsetninger for å få til oppgaven. Følelsen elevene får ved å løse en oppgave, kan knyttes til begrepet mestring, noe også læreren nevner i denne forbindelse. Den analoge tilnærmingen oppgaven har, kan gagne elevene som ellers opplever liten grad av mestring innenfor matematikk, hevder læreren. Læreren tror at mestringsfølelsen elevene fikk ved den analoge programmeringsøkten, kan ha skapt motivasjon for programmering generelt.

Dette punktet underbygges av elevenes svar på påstanden om oppgavens vanskelighetsnivå. Til tross for at mange fortalte læreren at de fikk feil på de første oppgavene da de sjekket med løsningsforslaget, syntes kun fem elever at den analoge oppgaven var for vanskelig. Wæge & Nosrati (2018) understreker at en forutsetning for elevenes mestringsfølelse er at oppgavene ikke oppleves for vanskelige slik at det oppstår frustrasjon hos elevene. Spørreundersøkelsen sier ingenting om hvor mange elever som syntes oppgavene var for enkle, men den forteller at ni av elevene syntes oppgaven var kjedelig, noe som kan være en følge av for få utfordringer. Samtidig er det lærerens oppfatning at mange elever fikk feil svar på første oppgave, men etter hvert mestret oppgaven godt. Dette kan bety at oppgaven var passe utfordrende for mange. Skriftlige tilbakemeldinger på opplegget underbygger også dette: «Det var bra at vi fikk bruke mer tid enn vi skulle, det var gøy å få det til», «Det var artig å se hvordan for-løkker fungerte … I tillegg syntes jeg det var gøy, så jeg fikk motivasjon til å fortsette selv når timen var over». Dette tyder på at aktiviteten skaper mestringsfølelse, som også hos noen elever førte til en indre motivasjon, ettersom svarene spesifikt viser at elevene ville fortsette også etter at timen var ferdig. Det er nettopp denne typen motivasjon som knyttes til ulike læringsfremmende initiativer fra elevenes side (Wæge & Nosrati, 2018).

Elevenes respons etter den analoge økta underbygger lærerens oppfatning av at undervisningen førte til høyere motivasjon for programmering generelt. Alle unntatt én svarer at de ønsker å lære mer om programmering. Samtidig svarer 13 elever at programmering for hånd gjorde at de fikk større motivasjon for programmering generelt, noe som er et tydelig flertall. Dette samsvarer med det Alamer et al. (2015) fant i sin studie av jenter på ungdomsskolen og videregående skole. Ni av elevene i min undersøkelse svarte at de syntes programmering for hånd med multilinks var kjedelig. Fem av disse var uenige i at de fikk større motivasjon for programmering generelt ved å programmere analogt, og fire av dem ønsket at aktiviteten skulle vært gjennomført digitalt. Dette kan tolkes på flere måter. Det kan se ut til at noen elever ville programmere digitalt og har høy motivasjon for det, men at de ikke fant motivasjon ved å programmere analogt, slik oppgaven er utformet. Noen opplever kanskje ikke motivasjon innenfor programmering, hverken ved analog eller digital programmering på generell basis. Andre elever kan ha høy motivasjon for å programmere fra før, men de opplevde ikke at denne aktiviteten gav mer motivasjon. Med dette som bakteppe finner jeg at flertallet av svarene fra elevene underbygger lærerens tro på at analog programmering skapte motivasjon for programmering generelt.

En faktor som taler imot at analog programmering skaper motivasjon for programmering, er at elevene ønsker å komme i gang med programmering på data og derfor blir utålmodige. Denne faktoren underbygges av at fire elever gav tilbakemelding om at opplegget ville vært bedre om det hadde vært gjennomført på data. I tillegg til de fire gav én elev følgende tilbake-



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

www.caspar.no/tangenten

melding: «Det var bra, men nå har jeg lyst til å prøve på pc.» Spørreundersøkelsen etter den digitale delen tyder på at elevene syntes den digitale delen, i likhet med den analoge delen, førte til mer motivasjon. Det var 19 elever som var enige eller litt enige i at den digitale programmeringen skapte større motivasjon for programmering generelt. Dette tyder på at både den analoge delen og den digitale delen fører til mer motivasjon innenfor programmering, noe som underbygges ved at i underkant av halvparten ville ha droppet den analoge programmeringsdelen, mens resterende ville beholdt den, også i etterkant av det digitale undervisningsopplegget. Om lag halvparten av elevene opplevde, i likhet med læreren, den analoge delen som motiverende. Det optimale er om man ved hjelp av både analoge og digitale arbeidsmetoder kan motivere alle elever i en klasse. Elevenes ulike svar tyder på at analog programmering kan skape motivasjon for programmering, men at man bør jobbe både analogt og digitalt for å nå de ulike elevene.

## Konklusjon

I denne studien har jeg kartlagt hvilke fordeler noen elever og én lærer ser ved å introdusere for-løkker ved hjelp av analog programmering på ungdomsskolen. Både elevene og læreren sett under ett viser at de er positive til analog programmering. Læreren gir uttrykk for at analog programmering kan bidra til å trene algoritmisk tenkning, skape motivasjon for programmering generelt samt legge til rette for ulike læringsstiler i undervisningen. Dette bekreftes langt på vei av elevene. At analog programmering ser ut til å påvirke motivasjonen for programmering generelt hos elevene, er i seg selv en grunn til å benytte analog programmering kontinuerlig i ungdomsskolen. Det ser også ut til at både elevene og læreren tror analog programmering har bidratt til at elevene har forstått hva de ulike løkkene utfører, og de gav uttrykk for at de gjennom denne erfaringen hadde et godt utgangspunkt for den digitale programmeringen. Dette betyr ikke at elevenes læringsutbytte ikke kunne vært oppnådd via digitale hjelpemidler. Det bekreftes i denne studien at variert bruk av læringsstiler har betydning, både for motivasjonens del og med hensyn til å tilpasse til ulike individer i klassen.

Funnene som vises i denne studien, er langt på vei de samme tendensene som man ser fra forskning i andre land, noe som tyder på at analog programmering bør inkluderes i ungdomsskolen. Studien gir grunnlag for videre diskusjoner i skole- og lærerutdanning, knyttet til hvordan programmeringskompetanse best kan utvikles. Jeg håper at funnene og undervisningsopplegget kan være til informasjon og inspirasjon for nåværende og fremtidige lærere som skal undervise i programmering i skolen. Det er imidlertid viktig å understreke at denne studien tar utgangspunkt i et lite utvalg informanter, og at det derfor er behov for videre forskning på elevers og læreres oppfatning av fordeler innenfor analog opplæring i programmering i norsk skole.

## Note

1 Oversettelse av «Computational thinking»

## Referanser


Alamer, R. A., Al-Doweesh, W. A., Al-Khalifa, H. S. & Al-Razgan, M. S. (2015). Programming unplugged: Bridging CS unplugged activities gap for learning key programming concepts. I J. E. Guerrero (red.), *Fifth International Conference on e-Learning* (s. 97–103). IEEE. https://doi.org/10.1109/ECONF.2015.27

Aranda, G. & Ferguson, J.P. (2018). Unplugged programming: The future of teaching computational thinking? *Pedagogika, 68*(3), 279–292. https://doi.org/10.14712/23362189.2018.859

Atlas, J., Bell, T. & Duncan, C. (2017). What do the teachers think? Introducing computational thinking in the primary school curriculum. I *Proceedings of the Nineteenth Australasian Computing Education Conference* (s. 65–74). Association for Computing Machinery. https://doi.org/10.1145/3013499.3013506




Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

Bell T. & Vahrenhold J. (2018). CS unplugged – how is it used, and does it work? I H. J. Böckenhauer, D. Komm & W. Unger (red.), *Adventures between lower bounds and higher altitudes. Lecture notes in computer science, 11011*. Springer, Cham. https://doi.org/10.1007/978-3-319-98355-4_29

Brinkmann, S. & Kvale, S. (2015). *Det kvalitative forskningsintervju* (3. utg.). Gyldendal Norsk Forlag.

Bueie, H. (2019). *Programmering for matematikklærere*. Universitetsforlaget.

Cortina, T. J. (2015). Broadening participation: Reaching a broader population of students through «unplugged» activities. *Communication of the ACM, 58* (3), 25–27. https://dl.acm.org/doi/pdf/10.1145/2723671

Dvergsdal, H. (2019, 4. august). Pseudokode. *Store norske leksikon*. https://snl.no/pseudokode

Giannakos, M. N., Jaccheri, L. & Proto, R. (2013). Teaching computer science to young children through creativity: Lessons learned from the case of Norway. I M. van Ekelen & E. Barendsen (red.), *Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research* (s. 103–111). Open Universiteit, Heerlen, Netherlands. https://dl.acm.org/doi/proceedings/10.5555/2541917

Humble, N., Mozelius, P. & Sällvin, L. (2019a). On the role of unplugged programming in K-12 education. I *European Conference on e-Learning 2019*. DOI: https://doi.org/10.34190/EEL.19.049

Humble, N., Mozelius, P. & Sällvin, L. (2019b). *Programmering i matte och teknik*. I Pedagogiska magasinet. http://miun.diva-portal.org/smash/get/diva2:1372308/FULLTEXT01.pdf

Maugesten, M. & Olafsen, A. R. (2017). *Matematikkdidaktikk i klasserommet*. Universitetsforlaget.

Scherer, R., Siddiq, F. & Sánchez Viveros, B. (2018). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of Educational Psychology, 111*(5), 764–792. http://dx.doi.org/10.1037/edu0000314

Thagaard, T. (2010). *System og innlevelse: En innføring i kvalitativ metode* (3. utg.). Fagbokforlaget.

Thomas, J. O., Rankin, Y. & Minor, R. & Sun, L. (2017). Exploring the difficulties African-American middle school girls face enacting computational algorithmic thinking over three years while designing games for social change. *Computer Supported Cooperative Work, 26*, 389–421. https://doi.org/10.1007/s10606-017-9292-y

Utdanningsdirektoratet (2019a). *Kompetansemål og vurdering*. https://www.udir.no/lk20/mat01-05/kompetansemaal-og-vurdering/kv16

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

Wæge, K. & Nosrati, M. (2018). *Motivasjon i matematikk*. Universitetsforlaget.



Berg, T.K. (2021). Analog programmering. *Tangenten – tidsskrift for matematikkundervisning, 32*(3), 42–52.

www.caspar.no/tangenten