# Algoritmisk tenkning i LK20 - den norske konteksten

Skrevet av Renate Andersen, juni 2020

«Den algoritmiske tenkeren må være systematisk og analytisk i sitt arbeid, men det er minst like viktig å være skapende, eksperimenterende og åpen for alternative løsninger» (Utdanningsdirektoratet, 2020).

Innføringen av LK20 høsten 2020 gir klare føringer på at algoritmisk tenkning og programmering er en del av fremtidens kompetanser for elevene. Derfor er det ekstra viktig at elever får grunnleggende ferdigheter i dette slik at de er forberedt på fremtidens arbeidsliv. Slik kan de utvikle evne til å tenke kritisk rundt fordeler og ulemper med bruk av teknologi og programmering.

## Hvorfor algoritmisk tenkning?

Ved innføringen av LK20 kommer programmering og algoritmisk tenkning inn som en del av flere fag i skolen. Det er særlig i kompetansemålene i fagene matematikk, naturfag, kunst og håndverk og samfunnsfag vi kan se at det er fokus på algoritmisk tenkning og programmering.

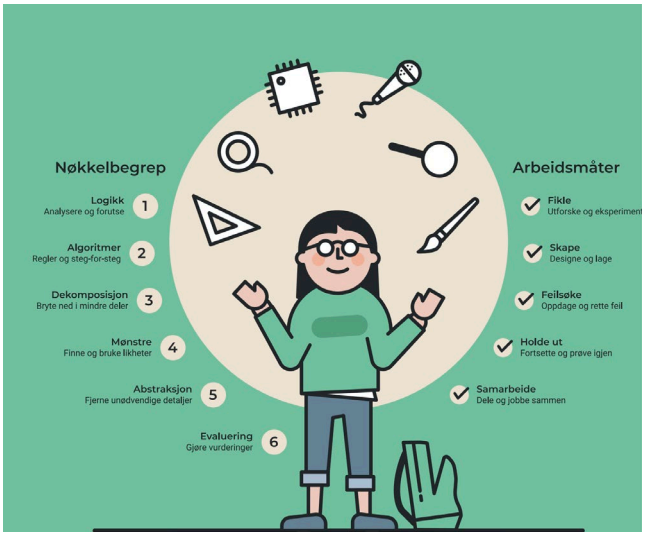
Selv om dette nå er en integrert del av fagene er det viktig å reflektere over hvorfor elevene skal lære algoritmisk tenkning og programmering? Utfra det som står i LK20 virker det som om formålet er at elevene skal bli gode problemløsere, ikke nødvendigvis at alle skal bli gode programmere. Et av hovedpoengene med det forsterkede fokuset på algoritmisk tenkning er nettopp at elevene skal forstå hvordan verden henger sammen og er bygget opp av algoritmer. Algoritmisk tankegang er et tankesett med fokus på problemløsning som fordrer å skape noe, prøve, fikle, feile og prøve igjen. Et sentralt element er at elevene lærer seg strategier for å løse problemer.

## Hva er algoritmisk tenkning?

*Algoritmisk tenkning* er et begrep som har fått stor oppmerksomhet den siste tiden. Det finnes flere ulike definisjoner som viser ulike nyanser av algoritmisk tenkning. Begrepet ble først definert av Papert (1980) som forholdet mellom programmering og «thinking skills». Wing (2006) tok begrepet et steg videre og definerer algoritmisk tenkning som hvordan man løser problemer, designer systemer og forstår menneskelig oppførsel ved å trekke på konsepter som hører til programmeringsfaget. Fokus her vil være på Udir sin definisjon av algoritmisk tenkning siden det er den som legges til grunn i LK20:

«Å tenke algoritmisk er å vurdere hvilke steg som skal til for å løse et problem, og å kunne bruke sin teknologiske kompetanse for å få en datamaskin til å løse (deler av) problemet. I dette ligger også en forståelse av hva slags problemer/oppgaver som kan løses med teknologi og hva som bør overlates til mennesker» (Utdanningsdirektoratet, 2020). Algoritmisk tenkning er den norske oversettelsen av det engelske «computational thinking».

Utdanningsdirektoratet presenter en sentral modell som de kaller den algoritmiske tenkeren som illustrerer noen viktige nøkkelbegrep som inngår i algoritmisk tenkning og typiske arbeidsmåter den algoritmiske tenkeren bruker for å løse problemer. Sentrale begreper er logikk, algoritme, dekomposisjon, mønstre, abstraksjon og evaluering. Arbeidsmåtene som den algoritmiske tenkeren fremhever er fikle, skape, feilsøke, holde ut og samarbeide.




Algoritmisk tenkning er en problemløsningsmetode som innebærer at man tilnærmer seg problemer på en systematisk måte, både når man formulerer hva man ønsker å løse og når man foreslår løsninger. Å tenke algoritmisk innebærer derfor å vurdere hvilke steg som skal til for å løse et problem, og bruke sin teknologiske kompetanse for å få en datamaskin til å løse deler av eller hele problemet (Utdanningsdirektoratet, 2020). Her ligger det en forståelse av hva slags problemer som kan løses med teknologi og hva som kan løses av mennesker. Dette vil si at algoritmisk tenkning handler å om å bryte ned komplekse problem til mindre, mer håndterlige delproblem som lar seg løse. Det innebærer å organisere og analysere informasjon på en logisk måte og å lage fremgangsmåter (algoritmer) for å komme frem til ønsket løsning. Det handler også om å lage abstraksjoner og modeller av den virkelige verden ved å fjerne unødvendige detaljer og fokusere på det som er relevant for den aktuelle problemstilling og løsning (Utdanningsdirektoratet, 2020). I dagens digitale samfunn hvor vi har enkel og rask tilgang på store mengder informasjon er det ekstra viktig å kunne kritisk vurdere hva som er relevant og riktig informasjonen i en gitt kontekst.

## Hvordan undervise i algoritmisk tenkning?

Begrepene i modellen ovenfor, «Den algoritmiske tenkeren», kan karakteriseres som abstrakte og høynivå begreper. Selv om de er veldig viktige, kan de ikke isolert sett forklare hvordan algoritmisk tenkning kan bli integrert som en del av klasserommet og hvordan det kan bli integrert i klasserommet på en didaktisk måte. Sevik (2017) understreker at hvis elevene skal utvikle algoritmisk tankegang innebærer det å tilnærme seg problemer på en systematisk måte og foreslå løsninger som kan bruke datamaskiner til å løse (deler av) dem. Det vil si både å: 1) tenke på hvilke steg som skal til for å løse et problem, og 2) bruke sin teknologiske kompetanse for å få datamaskinen til å løse problemet. Derfor anbefales det ifølge Sevik (2017) å starte med å trene elevene til å tenke algoritmisk ved å fokusere på det vi kaller *analog programmering*.

Analog programmering betyr at vi jobber med grunnleggende programmeringskonsepter uten tekniske virkemidler eller utstyr. Analog programmering kan derfor defineres som steg for steg beskrivelser av hverdagslige hendelser beskrevet i vårt naturlige hverdagsspråk. Eksempler på det kan hvordan lage en kopp te, pusse tenner eller lage seg en brødskive med syltetøy. Mannila (2017) sier at analog programmering er å skape algoritmer og programmere uten bruk av datamaskiner basert på en hverdagslig hendelse. Til dette kan man benytte seg av konkreter som erstatning for en datamaskin.

**Oppgave for å trene algoritmisk tankegang:**

- Be elevene jobbe sammen i par for å lage en algoritme for en hendelse – mulige hendelser de kan velge blant er: Gå ut av sengen, vaske hendene, lage frokost, skrive et brev, ta et bad eller se en film på Netflix.
- Be elevene dele algoritmen med hverandre – og prøve ut den andre eleven sin algoritme for å se om medelever klare å tolke oppgaven riktig.

I denne oppgaven får elevene erfaring med at et naturlig språk ofte ikke er presist nok. Det vil kreve et mer nøyaktig programmeringsspråk for å forklare hva en datamaskin skal gjøre for oss.

Selv om det er nytt at algoritmisk tenkning og programmering kommer inn i skolen med LK20, så er det viktig å huske på at algoritmebegrepet ikke er nytt. Som Mannila (2017) sier: «Vi har alle arbeidet med algoritmer på ulike måter siden vi var barn, vi har bare ikke kalt det for å følge en algoritme». Det er derfor bra å begynne å starte med å aktivt diskutere alle disse prosesser og praksiser med elevene.

**Refleksjonsspørsmål**

1) På hvilken måte kan algoritmisk tenkning integreres som en del av undervisning i et fag? Kan du gi et eksempel fra, f eks. matematikk, naturfag eller kunst og håndverk?
2) Hvordan skrive et program for å låse opp ytterdøren, børste tenner og lage en brødskive med syltetøy?
3) Kan du gi et eksempel på en ting du gjør i hverdagen som forutsetter at du bruker algoritmisk tenkning?
4) Hvilke fordeler og ulemper er det ved å innføre av algoritmisk tenkning og programmering i skolen gjennom LK20?

**Referanser:**


- Mannila L. (2017) *Att undervisa i programmering i skolan : varför, vad och hur? /*. Upplaga 1. (Nordén L-Å, red.). Lund: Studentlitteratur.
- Sevik, K. (2015). Koding i skolen: notat fra Senter for IKT i utdanningen. Senter for IKT i utdanningen (forlag/utgiver), red. 2015:1–30. https://www.udir.no/globalassets/filer/programmering_i_skolen.pdf.
- Utdanningsdirektoratet. (2020, 10.juni). Algoritmisk tenkning. Hentet fra: https://www.udir.no/kvalitet-og-kompetanse/profesjonsfaglig-digital-kompetanse/algoritmisk-tenkning/
- Papert, S. (1980). *Mindstorms: Children, computers and powerful ideas*. Basic Books, New York.