Ishoel

Python på eksamen

Det pågår en gradvis innføring av oppgaver som krever ferdigheter i tekstprogrammering på matematikkeksamen i grunnskolen (MAT0015). Udir presiserer på sine nettsider:

> Vi har en løpende vurdering på hvordan vi formulerer oppgaver med programmering som løsningsstrategi, og vi vil bruke kunnskap om eksamen til å finne gode løsninger på dette. (Utdanningsdirektoratet, 2024a)

Selv om det bare har blitt gjennomført tre ordinære eksamener i MAT0015 etter fagfornyelsen, foreligger det i tillegg flere eksempelsett som man kan se tilbake på for å spore utviklingen. Oppgavene som kan relateres til programmering, har inkludert blant annet algoritmisk tenkning, flytskjemaer, blokkprogrammering og eksempler på tekstprogrammering.

Ole Henrik Hjorthaug Ishoel
Lillestrøm realfagssenter
OleHenrikHjorthaug.Ishoel@lillestrom.kommune.no

Les artikkelen med løsninger av oppgavene på nettet:
http://tangenten.no/wp-content/uploads/2025/01/Ishoel-til-nettet.pdf

På eksamen skal elevene ha tilgang til programmeringsverktøy som støtter Python. Selv om det foreløpig ikke har kommet eksamensoppgaver som har krevet Python eksplisitt, åpner metodefriheten for at elever har kunnet bruke det i en rekke oppgaver likevel. Eksamensveiledningen fra 2024 presiserer:

> Kandidaten kan velge å bruke programmering som løsningsstrategi for å løse et matematisk problem. Det kan også være oppgaver hvor et matematisk problem er representert som en programkode. Kandidaten skal vurdere innholdet i koden og bruke denne til å løse det matematiske problemet. (Utdanningsdirektoratet, 2024b)

Denne formuleringen gir rom for tolkning, og hvilke oppgavetyper som kan komme på bakgrunn av slike formuleringer, vil alltid være noe uforutsigbart. Dette skaper usikkerhet blant lærere som forsøker å forberede elevene på det som måtte komme på eksamen. Den første eksamensoppgaven som faktisk inkluderte Python, ble gitt våren 2024. Oppgaven inneholdt imidlertid også blokkprogrammering, slik at elevene kunne bruke begge kodene for å forstå hvilken matematikk som lå til grunn. Det at Python var en del av eksamen, kan tolkes som et tegn på videre innføring. Etter alt å dømme vil kravene til elevenes Python-ferdigheter tilta.

Ishoel, O. H. H. (2025): Python på eksamen. *Tangenten – tidsskrift for matematikkundervisning, 36*(1), 36–40.

Utmerket måloppnåelse	God måloppnåelse	Lav måloppnåelse
Utforsker, utvider og modifiserer eksisterende kode. Utvikler programkode fra bunnen på en selvstendig og hensiktsmessig måte. Forstår programmer som en helhet.	Organiserer gitte kodelinjer i korrekt rekkefølge for å løse en oppgave. Vurderer, feilsøker og retter eksisterende kode. Forstår sekvenser av og/eller relaterte kodelinjer.	Sporer verdien til variabler i gitt kode og forklarer hva resultatet blir. Forklarer og kommenterer gitte kodelinjer. Forstår én kodelinje av gangen.

Tabell 1

Oppgaveutvalget under består ikke bare av «rene» algoritme- og programmeringsoppgaver. Det er også inkludert noen oppgaver som åpner for en løsningsmetode som benytter simulering eller numerisk tilnærming. På tross av at disse sannsynligvis ikke var ment for å løses ved hjelp av programmering i utgangspunktet, kan det ha didaktisk verdi å vise elevene hvordan man kan bruke programmering som metode for å løse dem. Løsningsforslaget som presenteres (i nettutgaven av artikkelen) til hver oppgave, er så klart kun ett av mange tenkelige, noe som illustrerer hvordan programmering stimulerer til utforsking og kreativitet. Løsningene er imidlertid ikke komplette, på den måten at de mangler utfyllende tekstforklaringer. Løsningene inkluderer skriptene, resultatene fra kjøringen av dem, en kommentar om oppgaven og forslag til eventuelle utvidelser.

Hvordan utviklingen av programmeringsoppgavene på eksamen vil skje fremover, er fortsatt usikkert og styres selvfølgelig av Udirs bestemmelser fra år til år. I mellomtiden kan vi forberede elevene på ulike anvendelser, og veilede elevene på hvordan de kan vise sin kompetanse i programmering der det er et hensiktsmessig verktøy. Dette krever imidlertid at vi introduserer og tydeliggjør variasjonen i oppgavetyper, vanskelighetsgrad og taksonomi.

For enkelhets skyld brukes videre i teksten kode som en fellesbetegnelse på Python-skript/program og blokkode. En mulig kategorisering av programmeringsoppgaver kan se slik ut:

Spore variabelverdier. Elevene skal identifisere og spore verdien til variabler i kode, og kunne si noe om hva resultatet blir.
Forklare og kommentere. Elevene skal forklare og kommentere kode, flytskjemaer og/eller pseudokode.
Vurdere og feilsøke. Elevene skal vurdere kritisk og rette feil i eksisterende kode.
Parsons-problemer. Elevene blir presentert for en rekke, gitte kodelinjer eller blokker som må settes i riktig rekkefølge for at programmet skal fungere i henhold til en oppgave.
Modifisere. Elevene skal utforske, utvide og/eller modifisere en gitt kode.
Utvikle. Elevene skal utvikle kode fra bunnen av i henhold til en gitt oppgave eller et problem.

Man kan selvfølgelig se for seg kombinasjoner av disse oppgavetypene, eller at de ulike brukes i deloppgaver for å skape en oppbygning.

Ulike oppgavetyper kan assosieres med ulik vanskegrad. En mulig taksonomi innen programmeringskompetanse som tar utgangspunkt i oppgavetypene over, kan se ut som i Tabell 1.

Merk: Denne taksonomien er basert på vurdering av «normalkode», som vil si kode som

Ishoel, O. H. H. (2025): Python på eksamen. *Tangenten – tidsskrift for matematikkundervisning, 36*(1), 36–40.

følger vanlige prinsipper og konvensjoner for effektive og funksjonelle programmer. Nyanser kan selvsagt forekomme, for eksempel hvis sporing av variabler settes opp komplekst med vilje, eller at man stiller svært lave kriterier for utvikling av programkode fra bunnen av. Så en alternativ taksonomi kunne tatt utgangpunkt i et slikt perspektiv, hvor samme type kompetanse deles inn i tre nivåer.

I det følgende kommer beskrivelse av åtte oppgaver som er valgt ut fra eksempelsettene og fra de ordinære eksamenssettene. De er presentert i kronologisk rekkefølge, altså etter når de er utgitt. På grunn av plassbegrensing er de kort beskrevet her, mens oppgavene i sin helhet med løsning og kommentar ligger på nettutgaven av artikkelen (på tangenten.no).

Oppgave 1

Eksempelsett «Vår 2021» Oppgave 4 (forutsatt heldigital eksamen).

Oppgaven går ut på at elevene skal tolke en algoritme som styrer en penn, og resultatet skal bli en Figur. Algoritmen er av typen «Start på (0,0), gjenta 4 ganger, gå frem» osv. Oppgaven er formulert som en flervalgsoppgave hvor elevene skal velge den figuren som passer best med algoritmen, og var opprinnelig tiltenkt en planlagt og varslet heldigital eksamen, men denne eksamensformen er skrinlagt inntil videre. Løsningen på oppgaven går ut på å oversette algoritmen til et skript som benytter Turtle-modulen for å lage figurene. Selv om vi her ikke gjenskaper en identisk kopi, vil vi få nok informasjon til å kunne identifisere rett svar.

Oppgave 2

Eksempelsett «Vår 2021» Oppgave 7, nytt sett datert 18.08.2021 Oppgave 5 og nytt sett datert 14.01.2022 Oppgave 5.

Oppgaven kommer i et par ulike varianter, men det de har til felles, er at elevene skal vurdere et flytskjema, i oppgaven noe upresist kalt algoritme, som vi gjenkjenner som en sjekk om hvorvidt en trekant er rettvinklet eller ikke. I flytskjemaet bes elevene om å legge inn lengden av den lengste siden først, og de to andre etterpå. Deretter kommer det en sjekk om hvorvidt Pytagoras' setning er oppfylt eller ikke. Flytskjemaet mangler bevisst hva som skal være output i de to tilfellene. Elevene bes om å forklare hva algoritmen undersøker, og om å gi eksempler på lengder som vil gi ulike outputer. Løsningen går ut på å lage et skript som gjennomfører sjekken og gir korrekt output.

Oppgave 3

Eksempelsett datert 18.08.2021 Oppgave 4 og nytt sett datert 14.01.2022 Oppgave 4.

I denne oppgaven, som kommer i to varianter med litt ulik ordlyd, blir elevene presentert for en likning med to ukjente, $(4 - a)(4 + b) = 8$, og skal bestemme tre løsninger til likningen. I utgangspunktet ser ikke denne oppgaven ut til å være ment for programmering, men metodefriheten gjør at vi like gjerne kan bruke programmering som noe annet. Elevene kan benytte ulike tilnærminger til en løsning, men en programmeringsløsning blir av typen «ved rå makt» (brute force), der man går gjennom mange potensielle løsninger og isolerer de som oppfyller likningen. En slags systematisert «prøve og feile»-metode som poengterer at eliminering av gale løsninger, kan være en effektiv (nok) vei til målet.

Oppgave 4

Eksempelsett datert 18.08.2021 Oppgave 7.

I denne oppgaven skal elevene sette opp et likningssett basert på panting av flasker. De får opplyst at Ali har pantet 51 flasker for 109 kroner, og de skal finne ut hvor mange små og store flasker Ali har pantet når panten på flaskene er hhv. 2 kroner og 3 kroner. Oppgaven er trolig ikke ment for programmering i utgangspunktet, da elevene blir bedt om å sette opp, forklare og løse et likningssett som metode. Løsningen med Python blir igjen å bruke rå makt, ved å sette opp alle kombinasjoner av store og små flasker i en dobbel for-løkke, for så å sette

Ishoel, O. H. H. (2025): Python på eksamen. *Tangenten – tidsskrift for matematikkundervisning, 36*(1), 36–40.

inn en if-test som benytter likningssettet som logisk test.

Oppgave 5

Eksamen 24.05.2022 Oppgave 3.

Oppgaven går ut på at blant fem kort som er nummerert 1–5, trekkes to kort tilfeldig. De fem kortene snus og blandes før trekkingen, og elevene skal summere tallene på de to kortene som trekkes. Elevene skal argumentere for at sannsynligheten er 40 % for at summen av de to trukne tallene er et partall. Denne oppgaven er trolig ikke ment for programmering, men programmering kan likevel brukes til å illustrere forskjellene på analytisk og numerisk tilnærming. Det som er den mest nærliggende tilnærmingen uten bruk av programmering, er kanskje å telle antall gunstige utfall og dele på mulige utfall, eller så kan man benytte en kombinasjon av produktsetningen og addisjonssetningen. Her kan man se for seg mange ulike måter å argumentere på. Python-løsningen går ut på å simulere situasjonen ved å lage et skript som «gjennomfører» en slik trekking et visst antall ganger. Brukeren kan velge antall trekninger i hver simulering, og vil legge merke til at andelen partall vil nærme seg 40 % jo flere trekninger man gjør.

Oppgave 6

Eksamen 24.05.2022 Oppgave 8.

I oppgaven blir elevene presentert for en Figur som viser en blokkbasert sekvens av instruksjoner, noe upresist kalt dataprogram. Variablene a og b settes til hhv. 4 og 5, før det kommer en blokk vi gjenkjenner som en løkke, som skal gjenta seg b ganger. I løkka skal det først tegnes et linjestykke med lengde a, og deretter kommer «snu» 360/b grader til høyre. Oppgaven ber elevene forklare hva som skjer når programmet blir kjørt, og deretter tegne figuren som programmet lager, samt sette riktige mål på figuren. Løsningen benytter også her Turtle-modulen, med den modifiseringen at verdien a må settes til 40 i stedet for 4, slik at figuren blir godt nok synlig. Algoritmen passer svært godt med syntaksen i Turtle, så her er det lav terskel for å overføre algoritmen til et slikt skript.

Oppgave 7

Eksamen 22.05.2023 Oppgave 5.

I denne oppgaven skal elevene utforske store talls lov ved å simulere kast med en terning. Programmet lar brukeren angi et tall for hvor mange ganger terningen skal kastes, og for hvert kast blir et tilfeldig tall mellom 1 og 6 valgt og lagt til en liste. Til slutt skrives hele listen ut i konsollen, uten noe mer bearbeidelse. Altså vil det komme en sekvens med tallene 1–6 som blir akkurat så lang som brukeren legger inn. I hvilken grad en slik tallsekvens vil kunne si noe om sannsynlighet, er uklart. Elevene på sin side blir bedt om å forklare «hva som skjer når programmet kjøres», og å velge en passende verdi for antall terningkast for å demonstrere at sannsynligheten for å få 1, 2, 3, 4, 5 og 6 øyne er omtrent like stor. Hva som er en passende, eller anbefalt, verdi i denne sammenhengen, kan være gjenstand for diskusjon, da det største problemet er at programmet ikke er egnet til å si noe om dette i utgangpunktet. Etter min mening hadde denne oppgaven fungert bedre som en «Modifisere»-oppgave, der elevene hadde kunnet utforske, utvide og/eller modifisere algoritmen til å bli funksjonell etter formålet. Løsningen er et Python-skript som representerer en slik modifikasjon.

Oppgave 8

Eksamen 21.05.2024 Oppgave 6.

Dette er altså den første gangen elevene møtte en oppgave som inkluderte ren Pythonkode. I oppgaven hadde elevene imidlertid tilgang på koden både i Python og som blokk. Det forelå også et resultat fra en kjøring av programmet. I koden tas det to verdier fra brukeren, radius for en «liten» kule, og radius for en «stor» kule. Deretter beregner koden volumet til de to kulene, og bestemmer forholdet mellom volumet til den store kula og volumet til den

Ishoel, O. H. H. (2025): Python på eksamen. *Tangenten – tidsskrift for matematikkundervisning, 36*(1), 36–40.

lille kula. Dette forholdet skrives ut, og kjøringen av koden viser at dersom man skriver inn radius 2 for den lille og 4 for den store, får man at forholdet mellom volumene blir 8. Elevene bes forklare «hva som skjer når koden kjøres», og «hvordan volumet til en kule endres når radiusen dobles». Løsningen bygger på at man simpelthen skriver av koden og gjør egne kjøringer for å vise sammenhengen. Det kan diskuteres om dette er fullverdig som løsning, eller om det kan kreves analytiske argumenter i tillegg.

Det finnes altså mange gode muligheter for å bruke tidligere oppgaver, enten fra eksempelsett eller ordinære eksamenssett, til å introdusere ulike oppgavetyper, vanskelighetsgrader og taksonomiske nivåer innen programmering. Slike oppgaver egner seg også godt til å tydeliggjøre forskjellen mellom analytiske og numeriske metoder i matematikk. Når elevene går på 10. trinn, har de vanligvis mer erfaring med analytiske metoder i matematikk enn med numeriske. En viktig del av elevenes programmeringskompetanse er derfor å lære å vurdere når og hvordan programmering kan brukes som et effektivt hjelpemiddel på del 2 av eksamen.

Hvis jeg til slutt skal vurdere vanskelighetsgraden på de oppgavene som kan klassifiseres som rene algoritme- eller programmeringsoppgaver på eksamen så langt, i lys av kriteriene for måloppnåelse tidligere i denne teksten, kan de strengt tatt anses for å kun måle lav kompetanse. Mange vil kanskje ikke være helt enige i dette perspektivet, da programmering kan oppleves som krevende i seg selv. Likevel er det slik at oppgavene så langt stort sett har handlet om å tolke eller spore variabelverdier og forklare ferdige algoritmer, flytskjemaer og programkoder. Variasjonen i oppgavetyper har derfor vært relativt begrenset. Derfor er det relevant å stille spørsmål om hvilke nye oppgavetyper som kan komme på fremtidige eksamenssett, og hva slags programmeringssyntaks oppgavene forutsetter at elevene behersker.

Referanser

Utdanningsdirektoratet. (2024a). *Slik endrer vi eksamen*. https://www.udir.no/eksamen-og-prover/eksamen/slik-endrer-vi-eksamen/#a210172

Utdanningsdirektoratet. (2024b). *Eksamensveiledning – om vurdering av eksamensbesvarelser MAT0015 Matematikk*.

Ishoel, O. H. H. (2025): Python på eksamen. *Tangenten – tidsskrift for matematikkundervisning, 36*(1), 36–40.