IT2-prøve, torsdag 20. mars 2025

Tid: 08.15 –10.25 (2 timar og 10 minutt)

Hjelpemiddel og kjelder:
Alle er tillatt, med unntak av kommunikasjon. Det er IKKJE lov til å spørre om hjelp i ulike forum, i chat, å bruke KI, Copilot eller liknande. Dette blir sett på som juks.

Du skal oppgje alle kjelder der du hentar kode frå på ein tydeleg måte, der det òg tydeleg går fram kva du har henta.

Alle legg mobilen på bordet, slår den på lydlaus, og lar den ligge der i løpet av dagen.

Ved oppstart viser du at du har slått av eller sletta extensions som Copilot, Gemini eller liknande. Felles for desse er at dei skriv/genererer kode for deg. Det kan bli tatt stikkprøvar undervegs.

Vurdering:
Vurderingskriterier finn du til slutt i dette dokumentet. Det er i tillegg oppgitt nokre endå meir konkrete kriterier/krav i sjølve oppgåvene.

Innlevering:
Marker alle mappene/filene med arbeidet ditt og i:

- Windows: Høgreklikk på mappene/filene og velg “Send til komprimert (zippet mappe)”. Hugs å endre namn på zip-fila (sjå under).
- MacOS: Høgreklikk på mappene/filene og velg “Komprimer (x) objekter”. Hugs å endre namn på zip-fila (sjå under).

Filnavn:

Ditt navn IT2 mars 2025

- Last opp zip-fila der prøven blei utlevert. Hugs å trykke på “Lever”-knappen når du er ferdig.

Oppgåve 1 (av 3): Drøfting (30 minutt)

I denne oppgåva skal du **velje mellom 1 av dei 2 alternativa du finn i vedlegga**.

Du skal skrive ein kort tekst på ca. 300 ord, der du skal vise at du kan drøfte/diskutere eit gitt tema og/eller sitat eller påstand.

Vurdering, drøfting:

Kompetansemål:

- utforske og vurdere muligheter, utfordringer og konsekvenser ved bruk av informasjonsteknologi i ulike sammenhenger
- drøfte etiske dilemmaer som oppstår som en konsekvens av informasjonsteknologi, både for individ og samfunn

Høg måloppnåing:

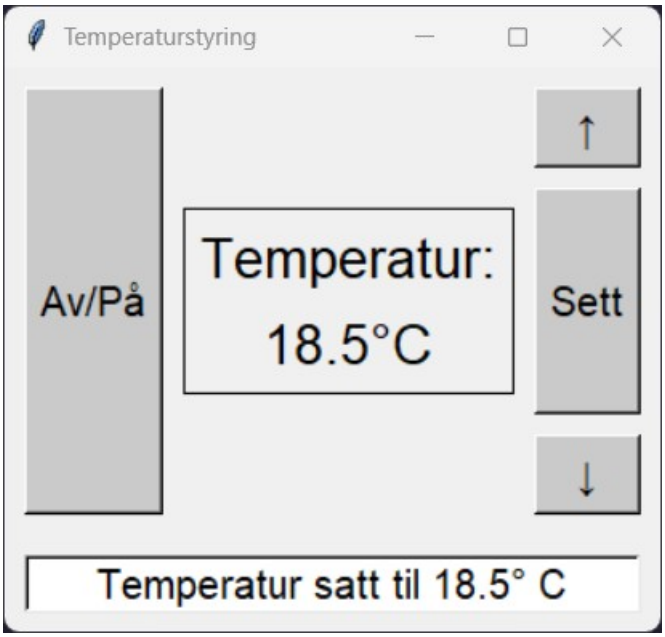
- Teksten skal vise god forståing for dei etiske problemstillingane og vurderer konsekvensar på både individ- og samfunnsnivå.
- Teksten viser eit bevisst forhold til fagterminologi, og uttrykkjer komplekse tankar på ein klar og overtydande måte.
- Teksten inneheld ulike relevante perspektiv, inkludert eigne refleksjonar.

Oppgåve 2 (av 3): GUI (50 minutt)

Deloppgåve A:

Gjenskap GUI-et for å kunne setje/stille inn temperatur på eit bad.

Her er designet du skal etterlikne:



Deloppgåve B:

Funksjonaliteten som skal implementerast er som fylgjer:

- Set ein startverdi for temperaturen på 20 grader celcius når programmet startar. Dette skal visast i den store boksen i midten.
- Når brukaren trykker på «opp»-knappen justeres temperaturen med ei halv grad opp.
- Når brukaren trykker på «ned»-knappen justeres temperaturen med ei halv grad ned.
- Knappen mellom «opp» og «ned» bekreftar/set temperaturen, og det blir skrive fylgjande setning ut på botn av appen: «Temperatur satt til 23.5° C».

Vurdering, GUI

Kompetansemål:

- utforske og vurdere alternative løsninger for design og implementering av et program
- vurdere brukervennligheten i egne og andres programmer og foreslå forbedringer
-

Høg måloppnåing:

- Du gjenskapar GUI frå skjermbiletet i deloppgåve A.
  - Du nyttar grid for å plassere elementer på riktig plass, og gjer dette på ein ryddig og oversikteleg måte.
- Du legg til funksjonalitet slik den er beskriven i deloppgåve B.
  - Feil blir handtert.

Oppgåve 3 (av 3): Datasett (40 minutt)

Ta utgangspunkt i den vedlagte CSV-fila.

Del 1:

- Utgangspunkt: «Protocol»
- Kva for tre (3) protokollar er brukt oftast i datasettet, og kor mange gonger blir desse brukt (forekomst)?

Del 2:

- Utgangspunkt: «Attack type»
- Finn ut, og skriv ut, kor mange ulike typar «Attack Type» det er.
- Finn ut, og skriv ut, kor stor prosentandel «DDoS» utgjer av totalen.
- Lag ein graf/visualisering som viser fordelinga av dei ulike typane «Attack Type».

Del 3:

- Utgangspunkt: «Device Information»
- Finn ut kva type operativsystem som blir angripe oftast, av «Macintosh» og «Windows».
- Lag ein graf/visualisering som viser fordelinga av spørsmålet over.

Vurdering, datasett:

Kompetansemål:

- bruke programmering til å innhente, analysere og presentere informasjon fra reelle datasett
- (gjøre rede for standarder for lagring, utveksling og sikring av ulike typer data)

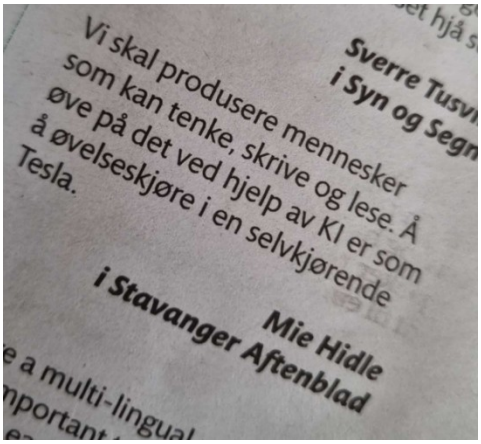
Høg måloppnåing:

- Du nyttar Pandas eller «vanilla» Python for å svare på dei ulike deloppgåvene. Desse teknikkane er jamnstilte.
- Koden unngår å gjere operasjonar fleire gonger enn naudsynt, og er effektiv.
- Visualiseringane er tydelege og gode, på ein slik måte at det dei skal framstille er lett å både sjå og forstå for den som ser på.

Vedlegg

Alternativ 1:
KI i tilpassa opplæring – for mykje eller for lite?

Scenario: Ein elev bruker ChatGPT til å få forklart matematikkoppgåver, men i staden for å lære, kopierer eleven berre svaret. Ein annan elev bruker KI for å få forslag til tekst, men redigerer og omformulerer det før innlevering.



Diskuter:

- Når blir KI ein støtte, og når blir det ein snarveg?
- Korleis kan elevar lære å bruke KI på ein måte som fremjar læring?
- Reflekter kring korleis dette kan påverke individ og samfunn.

Litteraturliste:

- Kunstig intelligens i skolen | udir.no
- Eventuelle eigne kjelder (som du oppgjer)

Alternativ 2:
Korleis kan kunstig intelligens bidra til å redusere einsemd og styrke sosiale relasjonar? Er det situasjonar der teknologien kan skape nye utfordringar for det sosiale livet?

Kva for etiske spørsmål oppstår når KI-assistentar imiterer menneskeleg kommunikasjon på ein så naturleg måte at ein kan forveksle dei med ekte menneske? Reflekter kring korleis dette kan påverke individ og samfunn.

Litteraturliste:

- Fireship: “AI influencers are getting filthy rich... let's build one”: https://www.youtube.com/watch?v=ky5ZB-mqZKM
- Fireship: “Our AI girlfriends just leveled up big time…” https://www.youtube.com/watch?v=UlVADiGfwWc
- Sesame: “Crossing the uncanny valley of conversational voice” https://www.sesame.com/research/crossing_the_uncanny_valley_of_voice
- Manus – the general AI Agent https://manus.im/
- Eventuelle eigne kjelder (som du oppgjer)

Vurdering, generelle kompetansemål

Kompetansemål	**Høy grad av måloppnåelse**
utforske og vurdere alternative løsninger for design og implementering av et program, vurdere og bruke strategier for feilsøking og testing av programkode, generalisere løsninger ved å utvikle og bruke gjenbrukbar programkode, vurdere brukervennligheten i egne og andres programmer og foreslå forbedringer,	Bruker hensiktsmessig mappe og filstruktur. Strukturerer og kommenterer egen programkode på en presis og konsis måte. Beskriver egne og andres algoritmer og gjør rede for virkemåte steg for steg. Vurderer effektiviteten av egne og andres algoritmer, og gjør avveininger i forhold til nytte vs. prestasjon. Bruker algoritmer for å analysere data og svare på problemstillingen. Vurderer ulike måter å strukturere og lagre informasjon, og kan benytte seg av disse etter behov.
Variablar	• Skriver velstrukturert kode med gode variabelnavn og hensiktsmessige datatyper. • Brukt på ein god måte for å styre programflyt og gjere programmet meir dynamisk. • (Tydelege navn som gjer meining)
Valgsetningar	• Brukt på ein god måte for å styre programflyt og gjere programmet meir dynamisk, og handterer unntak ein blir bedt om. • Handterer feil, og gjer tydelege feilmeldingar.
Løkker	• Løkker som dynamisk itererer gjennom data og “gjer ein jobb”, eksempelvis summerer, teller opp antal, samanliknar data og liknande. • Blir brukt til å begrense operasjonar (“så lenge som ... “).
Funksjonar	• Kan samle kode i funksjonar der det er hensiktsmessig, legg ikkje sentrale deler utanfor desse som gjer at programmet feilar. • Kan bruke funksjonar med parameter, og som kan returnere data. • Kan skrive generelle funksjonar som kan gjenbrukast. • (Tydelege navn som gjer meining)
Kolleksjonar/datastrukturar	• Kan vurdere kva for ein kolleksjon som egnar seg til ein gitt situasjon, og benytte seg av denne på ein god måte for å løyse ulike

	problemstillingar/oppgåver.
OOP	• Kan planlegge eit program med tydeleg UML-modell, og begrunne vala som er gjort, samt diskutere eventuelle andre løysingar. • Kan lage ei løysing som tek hensyn til problemstillingane i oppgåveteksten, og nyttar seg av ein god struktur mtp. klassar, arv og andre faktorar frå pensumet. • Får samanhengane mellom ulike klassar til å fungere (bok/bøker er ein del av eit bibliotek, ein bil kan parkere i eit parkeringshus osv.).
PyGame	• Kan lage interaktive applikasjonar og spel som tek hensyn til problemstillingane i oppgåveteksten. • Har god struktur i koden (testing på kriterier, oppdatering, teikning m.m. ligg på ”rett stad”). • Nyttar OOP.

Tips

- Løys ein og ein del, sjå gjerne på det i mange tilfeller som å få ei og ei linje med kode til å fungere. Få denne til å køyre før du går vidare. Etterkvart kan me sleppe oss meir laus, og skrive litt meir kode om gongen. Kvifor? Til dømes er det lettare å feilsøke, i mange tilfeller.
- Få grunnfunksjonaliteten på plass først, så kan du be om input, og gjere andre deler som typisk kan by på fleire problem. Eksempelvis kan du late som om du har fått noko input ved å definere ein variabel som du heilt tydeleg styrer i byrjinga av programmet.
- Ha ei testfil der du køyrer mindre deler av koden for å sjekke om til dømes oppsettet av ei løkke er feil, eller om det er andre deler av programmet som feilar.
