Sist endret: 1. februar 2023

IN1000—Innleveringsoppgave 1

Introduksjon

Velkommen til den første innleveringsoppgaven i IN1000! Det er tre deloppgaver som skal løses, som hver teller ett poeng. Les gjennom hver oppgave før du begynner å programmere, og forsøk gjerne å løse oppgavene på papir først! Hvis du sitter fast på en oppgave bør du prøve å løse øvingsoppgavene i Trix (se lenke under hver oppgave) før du spør om hjelp.

For hvert program du skriver skal du legge ved en kommentar i toppen av fila som forklarer hva programmet gjør. Videre forventes det at du kommenterer koden underveis så det blir tydelig hva du har tenkt. Dersom oppgaven sier du skal endre programmet trenger du kun å levere den endrede løsningen. Andre viktige krav til innleveringen og beskrivelse av hvordan du leverer finner du nederst i dette dokumentet.

Læringsmål

Målet for disse oppgavene er at du skal ha kommet i gang med programmeringen. Du skal dessuten vise at du kan lagre variabler med verdier og gjøre beslutninger ved hjelp av *if*-tester.

Oppgave 1: Utskrift og innlesing med variabler

Filnavn: *variabler.py*

1. Lag en fil ved navn *variabler.py*.

2. Skriv et program i denne filen som skriver ut ”Hei Student!” til terminalen.

3. Endre programmet slik at du ber brukeren om å oppgi et navn i form av en tekststreng ved hjelp av funksjonen *input()*, og lagre denne verdien i en variabel *navn*. Skriv så ut “Hei “ og variabelen *navn*.

4. Utvid programmet ditt med to variabler. Du kan velge variabelnavn selv, men gi begge variablene heltallsverdier. Skriv ut variablene på hver sin linje.

5. Beregn differansen av de to variablene (den første minus den andre) og legg resultatet inn i en ny variabel. Skriv ut “Differanse:” etterfulgt av denne tredje variabelen.

6. Be bruker om å oppgi et nytt navn, og legg svaret i en ny variabel. Lag nok en variabel ved navn sammen, og gi den verdien av det første navnet etterfulgt av det andre navnet. Skriv ut *sammen* på en ny linje.

7. Du skal nå endre verdien av variabelen *sammen*. Dette skal du gjøre ved å slå sammen de to navnene som i forrige deloppgave, men denne gangen skal du legge til "og" med et mellomrom på hver side mellom dem. For eksempel: Dersom *sammen* først hadde verdien "OlaKari!" skal den nå ha verdien "Ola og Kari".

   Viktig: Du skal utvide programmet ditt ved å endre verdien av variabelen *sammen* på en ny linje, ikke endre linjen der du først definerte variabelen.

Synes du denne oppgaven var vanskelig?
Tren på Trix-oppgaver 1.01, 1.04, 1.06, 1.07 og 1.10
Synes du denne oppgaven var enkel?
Se Trix-oppgaver 1.05, 1.12 og 1.13

Oppgave 2: Beslutninger

Filnavn: *beslutninger.py*

1. Skriv et program som ber brukeren om å svare "ja" eller "nei" på om de kunne tenke seg en brus. Lagre svaret i en variabel.

2. Skriv en if-sjekk som tester hva brukeren har skrevet inn:

   a. Hvis brukeren har svart *"ja"* skal programmet skrive ut *"Her har du en brus!"*

   b. Hvis brukeren har svart *"nei"* skal setningen *"Den er grei."* skrives ut.

   c. Hvis brukeren har svart noe helt annet skal programmet skrive ut *"Det forstod jeg ikke helt."*

Synes du denne oppgaven var vanskelig? Se Trix-oppgave 1.08 og 1.09
Synes du denne oppgaven var enkel? Se Trix-oppgave 1.13, 1.14 og 1.16

Oppgave 3: Problemløsning med beslutninger

Filnavn: *dato.py*

1. Skriv et program som ber om og leser inn to datoer, angitt med heltall for dag og måned (to variable for hver dato, altså dag1, mnd1, dag2, mnd2). Eksempel *24* og *12* for 24. desember.

2. Skriv en if-test som tester hvilken dato som kommer først i samme år:
   a. Hvis den første datoen kommer først skal programmet ditt skrive ut *"Riktig rekkefølge!"*
   b. Hvis den siste datoen som skrives inn er en tidligere dato, skal programmet skrive ut *"Feil rekkefølge!"*
   c. Om datoene er like skrives *"Samme dato!"* ut.

3. Frivillig: Skriv et nytt program i filen *dato2.py* som gjør det samme som i punkt 2, men der brukeren skal skrive hver dato som ett heltall i stedet for to (altså dato1, dato2).

Gi brukeren eksempel på hvordan datoen bør skrives for å sikre at sammenligningen i programmet ditt kan gjøres enklere enn i punkt 2.

Synes du denne oppgaven var vanskelig? Se Trix-oppgave 1.09 og 1.11
Synes du denne oppgaven var enkel? Se Trix-oppgave 1.14, 1.15,1.17 og 1.18

Denne obligatoriske innleveringen med tilhørende Trix-oppgaver er minimum av hva du bør ha programmert i løpet av ei uke. Du finner flere oppgaver for denne uka her.