IN1000: Obligatorisk innlevering 3

Frist for innlevering: 8.9.26, kl. 23:59 **Sist endret:** 1.9.26

Introduksjon

Det er kun quiz som må godkjennes denne uken, men programmeringsoppgavene er likevel nødvendige både for trening og som et mål på om du følger progresjonen i emnet. Ved å levere løsningene dine får du tilbakemeldinger som hjelper deg videre.

Oppgavene (og neste ukes undervisning) forutsetter at du har jobbet med ukens pensum til du har god kontroll på stoffet. Oversikt over undervisningen i IN1000 inkludert ekstratilbud finner du her. Undervisningsmateriale for hver uke finner du her eller fra Timeplanen (under Forelesninger).

For å få godkjent obligatoriske innleveringer må du innen fristen:

1. Besvare denne quizen i Nettskjema. Det er ingen krav til antall rette.
2. I Devilry (se nederst):
    a. Levere skjermbilde (.jpg) av svarsiden i Nettskjema
    b. Levere filer som angitt i oppgaver merket **obligatorisk** nedenfor (0 eller flere). Merk at koden må kjøre uten feil, eventuelt må du beskrive feilen og hva du tror kan hjelpe (se "Hvordan levere i Devilry" nederst i oppgaven)

Har innleveringen mindre feil eller mangler i obligatoriske oppgaver kan du få et (1) nytt forsøk - dette vil ha kort frist (3 dager), så sjekk e-post for tilbakemelding hver dag.

Les gjennom hver oppgave før du begynner å programmere, og forsøk gjerne å løse oppgavene på papir først (da kan det være lettere å konsentrere seg om en fremgangsmåte som gradvis beskrives i Python).

Dersom oppgaven sier du skal endre programmet, trenger du kun å levere den endrede løsningen.

Læringsmål

Du skal kunne holde styr på mange verdier ved hjelp av forskjellige samlinger (lister, mengder og ordbøker). Du skal vite at samlingene representeres av objekter som tilbyr tjenester i programmet. I tillegg skal du vise at du kan skrive litt mer komplekse programmer med kunnskapene du har tilegnet deg de siste ukene.

Oppgave 1 - Lister

Filnavn: *lister.py*

1. Lag en liste fylt med 3 tall du velger selv. Legg deretter til et nytt tall på slutten av lista. Skriv ut det første og det tredje elementet i lista.
2. Beregn summen av og produktet av (multipliser) alle tallene i lista, og legg disse to tallene (sum og produkt) i en ny liste. Lag så en tredje liste ved å slå sammen den opprinnelige tall-listen med listen som inneholder sum og produkt, og skriv ut denne lista (som altså skal ha seks tall).
3. Lag en ny, tom liste. Be brukeren om å oppgi 3 navn, og legg disse inn i lista.
4. Sjekk om ditt eget fornavn er blant de tre navnene brukeren oppga. Hvis brukeren

har gjort det, skal du skrive ut “Du husket meg!”. Hvis navnet ditt ikke finnes i lista, skal du skrive ut “Glemte du meg?”.

Syntes du denne oppgaven var vanskelig? Se Trix-oppgave 3.01, 3.03, 3.12
Syntes du denne oppgaven var enkel? Se Trix-oppgave 3.10

Oppgave 2: Samling

Filnavn: *samling.py*

1. Lag en samling som skal inneholde følgende butikkvarer og tilhørende pris:

   melk - kr 14.90,
   brød - kr 24.90,
   yoghurt - 12.90,
   pizza - 39.90.

   Hvilken type samling valgte du og hvorfor?
   Skriv ut innholdet i ordboken med en enkel print-setning (trenger ikke å bli pent).

2. Les inn to varenavn og tilhørende priser fra brukeren og legg disse til i samlingen. Skriv ut innholdet på nytt.
3. Spør brukeren hvilken vare de ønsker å kjøpe, og skriv ut prisen på den varen.

Syntes du denne oppgaven var vanskelig? Se Trix-oppgave 3.07, 3.04
Syntes du denne oppgaven var enkel? Se Trix-oppgave 3.13

Oppgave 3: Billettpris

Filnavn: *billettpris.py*
Nå skal du lage et system som beregner billettpris avhengig av kjøperens alder.

1. Programmet ditt må inneholde en prosedyre som spør brukeren om alder, og deretter skriver ut en billettpris som avhenger av den angitte alderen. Opprett tre priskategorier: barnebillett, standardbillett og seniorbillett (velg noen passende priser), og definér aldersgrupper som passer for disse kategoriene.
2. Test at prosedyren fungerer når du angir 3 forskjellige aldre for brukeren, én for hver aldersgruppe.

Syntes du denne oppgaven var vanskelig? Se Trix-oppgave 1.16, 2.03, 2.09.
Syntes du denne oppgaven var enkel? Se Trix-oppgave 3.14, 3.15

Oppgave 4: Matplan

Filnavn: *matplan.py*
Forestil deg at beboerne på et sykehjem får servert tre måltider i løpet av en dag. Du skal organisere informasjonen om fem beboere og hvilke måltider de får til frokost, lunsj og middag.

1. Lag en datastruktur som kobler hver beboer til en oversikt over hva vedkommende spiser til de tre måltidene. For eksempel: en person kan få brød til frokost, egg til lunsj og pølser til middag. Lag så en prosedyre som skriver ut navnene til alle beboerne og spør brukeren om å skrive navnet til en beboer i terminalen. Deretter skal prosedyren skrive ut matplanen til den valgte beboeren. Skriv ut en melding til brukeren hvis beboeren ikke er registrert.
2. Teoretisk spørsmål: Hvilken type samling (liste, mengde, ordbok) ville du brukt for å representere hver av de følgende eksemplene på data? Skriv litt om hvorfor, eventuelt med forbehold eller presiseringer.
    - Brukernavn på alle IN1000 studentene
    - Brukernavn og antall studiepoeng dette semesteret for hver student på IN1000
    - Alle vinnere i Lotto siste år (kun navn)
    - All mat noen gjester i et selskap er allergisk mot (for å planlegge menyen)

Syntes du denne oppgaven var vanskelig? Se Trix-oppgave 3.02, 3.11
Syntes du denne oppgaven var enkel? Se Trix-oppgave 3.13

Oppgave 5: Egen oppgave

1. Skriv oppgavetekst til en oppgave som handler om lister eller ordbøker. Et forslag kan være å skrive en quiz som i forrige uke, men ved å bruke lister. Alternativt et program som sjekker alle allergier (for eksempel for deltakere ved påmelding til et arrangement), og skriver ut en oversikt til slutt over hva slags mat som bør unngås.
2. Løs oppgaven! Du skal levere både oppgaveteksten og besvarelsen (skriv oppgaveteksten som kommentarer over løsningen din).

Krav til innlevering

- Programmer som leveres inn skal kjøre uten feil. Om du har feil du ikke selv finner ut av kan du få hjelp i gruppetimer, på Fredagspython eller i Discourse. Hvis du leverer programmer med feil, må dette kommenteres i Devilry (se 4c nedenfor).
- Koden din skal inneholde gode kommentarer som forklarer hva programmene gjør.
- Programmene skal ha gode utskriftsetninger som er enkle å forstå for bruker.

Hvordan levere i Devilry

1. Du skal IKKE lage en .zip-fil
2. Logg inn i Devilry.
3. Last opp alle filer som angitt i besvarte oppgaver.
4. Du **skal** også besvare noen spørsmål i kommentarfeltet i Devilry:
    a. Hvordan syntes du innleveringen var? Hva var enkelt og hva var vanskelig?
    b. Hvor lang tid (ca.) brukte du på innleveringen?
    c. Var det noen oppgaver du ikke fikk til? Hvis ja:
        i. For hvilke(n) oppgave(r) fungerer ikke det innleverte programmet ditt?
        ii. Hvorfor tror du programmet ikke fungerer?

    iii. Hva ville du gjort for å få programmet til å fungere hvis du hadde mer tid?

5. Husk å trykke **«Legg til innlevering eller spørsmål»** og sjekk deretter at innleveringen din er komplett. Du kan levere så mange ganger du vil frem til fristen, men **siste innlevering må inneholde alle filer som skal vurderes**.
6. Innleveringen med tilhørende Trix-oppgaver er minimum av hva du bør ha programmert i løpet av en uke - de fleste vil trenge en del mer trening for å få god kontroll på stoffet. Du finner flere oppgaver for denne uken her.

Finner du feil i oppgaveteksten, send en epost til siriamj@uio.no.