GA GOKSTAD AKADEMIET

Backend-Programmering, Emne 1
Grunnleggende Programmering

Cybersikkerhet, Emne 1
Introduksjon til Programmering

Python

Arbeidskrav 1

Tid:	29 August 2025 – 03 October 2025
Hjelpemidler:	Alle
Vedlegg:	**Fil:** bokutlån.csv
Innlevering:	Python filene leveres samlet i en .zip fil

1 Grunnprogrammering

Oppgave 1.1

Lag et program som ber brukeren om å skrive inn et positivt heltall (`int`) og finner summen av alle tall f ra 1 til dette tallet (inkluder også tallet i summen). Summen skal beregnes ved hjelp av en for-løkke.

Oppgave 1.2

Skriv et program som ber brukeren skrive inn to setninger. Programmet skal deretter sammenligne lengden på de to setningene og skrive ut hvilken som er lengst og antall karakterer det er i denne setningen.

Oppgave 1.3

Lag et program som ber brukeren skrive inn et tall og genererer multiplikasjonstabellen for dette tallet (fra 1 til 10). Eksempel:
Input: 3
Output:
3 * 1
3 * 2
3 * 3
Osv..

Oppgave 1.4

Lag et program som bytter plass på to elementer i en gitt liste. Programmet skal ta utgangspunkt i følgende liste:

```
fruits = ["eple", "banan", "appelsin", "drue", "kiwi"]
```

1. Be brukeren om å skrive inn to indekser (input) som angir hvilke elementer i listen som skal bytte plass
2. Bytt plass på elementene som ligger på de angitte indeksene
3. Skriv ut den oppdaterte listen

Hvis en eller begge indeksene er ugyldige (ikke i listen), skal programmet gi en passende feilmelding

Oppgave 1.5

Utvid Oppgave 1.3 slik at brukeren kan angi et intervall [m, n] i stedet for 1-10, og programmet skriver ut en pent formatert tabell for tallet i hele intervallet.

```
|  1  |  2  |  3  |
|-----|-----|-----|
|  1  |  2  |  3  |
|  2  |  4  |  6  |
|  3  |  6  |  9  |
```

2 Datastrukturer

Oppgave 2.1

Skriv et program som leser inn en dato i formatet "dd/mm/yyyy" fra brukeren. Programmet skal deretter sjekke om datoen er gyldig. Hvis datoen er ugyldig, skal programmet skrive en passende feilmelding.

Oppgave 2.2

Ta utgangspunkt i en liste der tekst og tall er plassert parvis med navn og alder slik:

```
["Cecilie", 28, "Bjørn", 30, "Tor", 24, "Anna", 25]
```

Skriv et program som splitter denne listen i to separert lister, en liste for navn og en liste for alder.

Oppgave 2.3

Ta utgangspunkt i listene fra Oppgave 2.2 og lag en dictionary der tekstverdier fra listen med navn blir nøkler og tallverdiene fra listen med alder blir verdier. Skriv ut innholdet av denne dictionary på formatet:

```
"Cecilie er 25 år"
"Bjørn er 30 år"
```

Oppgave 2.4

Skriv et program som sorterer denne dictionary etter alder hvor den eldste skal være først. Skriv ut resultatet.

Oppgave 2.5

Ta utgangspunktet i dictionary som er sortert på navn i Oppgave 2.4 og lag en ny liste med disse verdiene slik at listen får samme format som den i Oppgave 2.2, men sortert på navn.

```
["Anna", 25, "Bjørn", 30, "Cecilie", 28, "Tor", 24]
```

Oppgave 2.6

Konverter datasettet fra Oppgave 2.2 til en liste av dictionaries med formatet:

```
{"navn": "Cecilie", "alder": 28}
```

3 Funksjoner

Oppgave 3.1

Lag en funksjon som tar inn en streng og sjekker om det er en gyldig IPv4-adresse. En gyldig IPv4-adresse har fire deler atskilt med punktum (`_`), der hver del er et heltall mellom 0 og 255. Funksjonen skal returnere `True` hvis det er en gyldig adresse, og `False` ellers. Eksempel:

Input: `"192.168.0.1"` → Output: `True`
Input: `"256.100.50.0"` → Output: `False`

Oppgave 3.2

Lag en funksjon som tar inn to datoer som strenger på formatet "dd/mm/yyyy" og returnerer antall dager mellom dem. Hvis den første datoen er senere enn den andre, skal funksjonen fortsatt returnere antall dager (positivt tall). Eksempel:

Input: `"21/11/2024", "01/01/2024"` → Output: 325 dager

Oppgave 3.3

I dataprogrammering brukes fargekoder ofte for å representere farger i brukergrensesnitt og grafikk. To vanlige måter å representere farger på er:

1. **Hex-kode:** En fargekode som starter med `#` og består av seks tegn som representerer de røde, grønne og blå komponentene i fargen (for eksempel `#CD5C5C`). Hver komponent har to tegn (heksadesimalt), hvor:

   - CD representerer rødt (r),
   - 5C representerer rødt (g), og
   - 5C representerer blått (b),

2. **RGB-kode:** En fargekode representert som tre heltall, én for hver fargekomponent (rød, grønn, blå). For eksempel: `rgb(205, 92, 92)`.

Lag en funksjon `rgb_to_hex` som tar inn tre heltall (for eksempel `red=205`, `green=92`, `blue=92`) og returnerer tilsvarende Hex-kode som en streng (for eksempel `"#CD5C5C"`). Legg også til passende feilmelding om red, blue eller green parametere har ugyldig verdi.

Oppgave 3.4

Lag en funksjon som tar en annen funksjon som parameter, noen argumenter og et forventet resultat, og som returnerer sant hvis det faktiske resultatet stemmer overens med det forventede resultatet, ellers usant.

4 Opprette filstruktur og sortere filer

Oppgave 4.1

Lag en funksjon som oppretter en mappe kalt `Files` og genererer 30 tilfeldige filer med følgende filtyper i denne mappen: `.txt`, `.csv`, og `.log`. Hver fil skal ha:

- Et tilfeldig navn med mellom 5 og 10 tegn.
- En tilfeldig filtype valgt fra de tre typene.
- Innhold i disse filen er ikke viktig — lag gjerne disse filene uten innhold

```
Files
|-- G5zLehz4.txt
|-- iTwTrTkU.txt
|-- vPcaO7jR.csv
|-- 1Vgi2jbe.log
`-- 7NMaq7aa.csv

0 directories, 5 files
```

Oppgave 4.2

Lag en funksjon som leser filene i `Files` og sorterer dem i undermapper basert på filtype:

- Opprett en ny mappe kalt `SortedFiles`
- Opprett undermapper kalt `txt-files`, `csv-files`, og `log-files` inne i denne mappen kalt `SortedFiles`.
- Flytt filene fra `Files` til riktig undermappe i `SortedFiles` basert på deres filtype.

```
SortedFiles
|-- csv
|   |-- vPcaO7jR.csv
|   `-- 7NMaq7aa.csv
|-- log
|   `-- 1Vgi2jbe.log
`-- txt
    |-- G5zLehz4.txt
    `-- iTwTrTkU.txt

3 directories, 5 files
```

5 Fil Analyse

Du har fått tilgang til en fil som heter `bokutlån.csv`. Filen inneholder informasjon om bøker som har blitt lånt ut fra et bibliotek over en gitt periode. Informasjonen er registrert i følgende kolonner:

1. **Fornavn:** Fornavnet til personen som lånte boken.
2. **Etternavn:** Etternavnet til personen som lånte boken.
3. **Boktittel:** Navnet på boken som ble lånt.
4. **Sjanger:** Sjanger på boken som ble lånt (Fiksjon, Krim, Sakprosa, Fantasy).
5. **Lånedato:** Datoen boken ble lånt ut (format: dd/mm/yyyy).
6. **Låneperiode:** Antall dager boken er lånt for (standard: 14 dager).
7. **Forlenget:** Hvor mange ekstra dager lånet ble forlenget (kan være 0 hvis ingen forlengelse).
8. **Tilbakelevert:** En indikasjon på om boken ble levert tilbake i tide eller ikke (Ja eller Nei).

Programmet må kunne håndtere følgende situasjoner:

- **Manglende data:** Dersom noen av feltene i filen mangler data, skal programmet ignorere den aktuelle raden, men fortsatt kunne fullføre beregningene.
- **Ugyldige verdier:** Hvis en verdi i kolonnene som krever numeriske data (som forlengelse eller låneperiode) ikke er et tall, skal programmet håndtere dette og gi en melding som forklarer feilen.

Tips: Åpne filen med utf-8 encoding slik: `open(filename, 'r', encoding='utf-8')`

Analysere dataene i denne filen for å besvare følgende spørsmål ved hjelp av et Python-program:

Oppgave 5.1

Skriv et program som summerer opp antall dager lånene ble forlenget og skriv ut svaret.

Oppgave 5.2

Lag en funksjon som beregner hvor mange bøker som er lånt ut per sjanger (f.eks. `"Fantasy: 3, Krim: 5"`, osv.) og skriv ut svaret.

Oppgave 5.3

Beregn den gjennomsnittlige låneperioden i antall hele dager for alle bøker som er lånt ut, inkludert forlengelsene og skriv ut svaret.

Oppgave 5.4

Lag en funksjon som lister opp alle bøkene som ikke ble levert tilbake. Returner en liste med navnene på bøkene og hvem som lånte dem og skriv ut svaret.

Oppgave 5.5

Skriv en funksjon som finner hvilke bøker som har blitt lånt flest ganger. Funksjonen skal returnere en oversikt over boktitlene og antallet ganger de har blitt lånt ut. Hvis flere bøker har blitt lånt ut like mange ganger, skal de sorteres alfabetisk. Skriv ut svaret.