Øving 4

INF620 - Høstsemesteret 2021

Øvingsoppgavene er ikke obligatoriske, men vi anbefaler likevel at du gjør de og leverer de innen fristen — Den eneste måten å lære å programmere på er ved å programmere. Ved å gjøre oppgavene får du også testet deg selv og sjekket at du forstår begrepene. Du skal levere én zip-fil, `oving4.zip`, som inneholder de 3 filene `oppg1.py`–`oppg3.py`. For å komprimere en eller flere filer til en zip-fil høyreklikker du filene (i dette tilfellet `oppg1.py`–`oppg3.py`) i maskinens filnavigasjonsprogram og velger `Komprimer` eller `Send til` → `Komprimert mappe`. Frist: Torsdag 7. oktober kl 23:59

1 Vokaler

Svar leveres på fil med navn `oppg1.py`

a) Skriv en funksjon `vokaler` som tar en streng som parameter, og returnerer en delstreng som består av vokalene i strengen. Eksempler på kjøring av funksjonen fra konsollen i Spyder:

```
In[1]: vokaler('Eirik')
Out[1]: 'Eii'
In[2]: vokaler('Åse')
Out[2]: 'Åe'
```

Hint: Bruk den logiske operatoren `in` og hjelpestrengen 'aeiouyæøå'.

b) Skriv en funksjon `antall_vokaler` som tar en streng som parameter, og returnerer antall vokaler i strengen. Eksempel på kjøring av funksjonen fra konsollen i Spyder:

```
In[3]: antall_vokaler('Dusan')
Out[3]: 2
```

2 Terninger og spill

Svar leveres på fil med navn `oppg2.py`

a) Skriv en funksjon `terninger` som returnerer samlet verdi av kast med `n` terninger. Verdien på `n` skal gis som parameter til funksjonen.

b) Skriv en funksjon `stigespill` som tar navn på en spiller (`navn`) og antall terninger (`n`) som parametre. Funksjonen skal la spilleren starte med 0 poeng, og kaste de `n` terningene så mange omganger som hun/han ønsker, eller til spillet avbrytes. I hver omgang blir samlet verdi lagt til poengene spilleren hadde fra før. Hvis verdien av de `n` kastene i en omgang er mindre enn verdien av de `n` kastene i omgangen før, avbrytes spillet, og spilleren blir stående med 0 poeng. Funksjonen skal returnere antall poeng spilleren fikk.

Eksempel 1 på dialog (her vil funksjonen returnere 0):

```
Dag har 0 poeng. Flere kast (j/n)? j
Dag fikk 17 poeng med 4 terninger.
Dag har 17 poeng. Flere kast (j/n)? j
Dag fikk 12 poeng med 4 terninger.
Resultat for Dag: 0 poeng.
```

Eksempel 2 på dialog (her vil funksjonen returnere 43):

```
Sandra har 0 poeng. Flere kast (j/n)? j
Sandra fikk 11 poeng med 4 terninger.
Sandra har 11 poeng. Flere kast (j/n)? j
Sandra fikk 16 poeng med 4 terninger.
Sandra har 27 poeng. Flere kast (j/n)? j
Sandra fikk 16 poeng med 4 terninger.
Sandra har 43 poeng. Flere kast (j/n)? n
Resultat for Sandra: 43 poeng.
```

c) Skriv en funksjon `turnering` som lar flere spillere delta i stigespillet. Funksjonen skal først lese inn antall terninger det skal spilles med. Deretter skal den lese inn navn på en spiller, og la henne/han spille ferdig. Dette skal gjentas like til en tom streng oppgis som navn. Da skal funksjonen skrive ut navn på vinneren av turneringen, dvs. spilleren som fikk den høyeste poengsummen, og antall poeng vinneren fikk. Første spilleren som oppnådde den høyeste poengsummen kåres til vinner dersom flere spillere fikk samme poengsum.

Eksempel på dialog:

```
Antall terninger: 4
Første spiller (avslutt med tom streng): Mathias
Mathias har 0 poeng. Flere kast (j/n)? j
Mathias fikk 10 poeng med 4 terninger.
Mathias har 10 poeng. Flere kast (j/n)? j
Mathias fikk 15 poeng med 4 terninger.
Mathias har 25 poeng. Flere kast (j/n)? j
Mathias fikk 15 poeng med 4 terninger.
Mathias har 40 poeng. Flere kast (j/n)? n
Resultat for Mathias: 40 poeng.

Mathias leder med 40 poeng. Neste spiller (avslutt med tom streng): Mathilde
Mathilde har 0 poeng. Flere kast (j/n)? j
Mathilde fikk 12 poeng
Mathilde har 12 poeng. Flere kast (j/n)? j
Mathilde fikk 19 poeng
Mathilde har 31 poeng. Flere kast (j/n)? j
Mathilde fikk 16 poeng
Resultat for Mathilde: 0 poeng.

Mathias leder med 40 poeng. Neste spiller (avslutt med tom streng):
Mathias vant turneringen med 40 poeng.
```

Funksjonene skrevet i denne oppgaven testes ved å kalle de i slutten av programfilen `oppg2.py`.

3 Smittespredning

Svar leveres på fil med navn `oppg3.py`

En smittsom sykdom herjer befolkningen. Dersom antall syke denne uken er s, vil antall syke neste uke være $s + \alpha s - \beta$ (rundet av til nærmeste heltall og ikke mindre enn 0), der α uttrykker antall smittetilfeller hver syk person i gjennomsnitt forårsaker på en uke, og β uttrykker hvor mange som kan kureres på en uke.

a) Skriv en funksjon `neste` som returnerer antall syke neste uke. Funksjonen skal ha parametrene `s`, `alpha` og `beta`, som gir verdiene på hhv. s, α og β. Eksempler på kjøring av funksjonen fra konsollen i Spyder:

```
In[4]: neste(100, 0.2, 25)
Out[4]: 95
In[5]: neste(18, 0.2, 25)
Out[5]: 0
```

b) Skriv en funksjon `utvikling` som skriver ut sykdomsutviklingen i `n` uker framover. Parametre til funksjonen skal være de samme som parametrene

til `neste`, i tillegg til `n`. Funksjonen skal skrive ukenummeret (regnet fra i dag) og antall smittede på tabulert form. Eksempel på kjøring av funksjonen fra konsollen i Spyder:

```
In[6]: utvikling(100, 0.2, 25, 10)
Uke   Syke
  0    100
  1     95
  2     89
  3     82
  4     73
  5     63
  6     51
  7     36
  8     18
  9      0
 10      0
```

c) Hvis $\beta > \alpha s$, vil antall smittede avta, og før eller siden være så lavt som eller lavere enn halvparten av hva det er i dag. Ellers vil antall smittede bare vokse. Skriv en funksjon `halveringstid` som returnerer antall hele uker som går før antall smittede har nådd halvparten av hva det er i dag. Dersom dette aldri skjer skal funksjonen returnere `None`. Parametrene til funksjonen skal være de samme som parametrene til `neste`. Eksempler på kjøring av funksjonen fra konsollen i Spyder:

```
In[7]: print(halveringstid(100, 0.2, 25))
7
In[8]: print(halveringstid(100, 0.2, 15))
None
```