Tekst ABC

Skrevet av: Ole Kristian Pedersen, Kodeklubben Trondheim

Kurs: Python
Tema: Tekstbasert
Fag: Programmering
Klassetrinn: 8.-10. klasse

Introduksjon

I denne oppgaven skal vi gjøre enkle operasjoner på tekst, som å endre størrelsen på bokstavene og telle ord.

I Python lagrer vi tekst til en **variabel** slik som dette:

```
streng = "teksten er her"
```

Variabelen kalles da **tekststreng** eller bare **streng**, som er typen til variabelen.

Store og små bokstaver

En måte å endre tekst på er ved hjelp av funksjoner. Innebygget i Python finnes flere slike funksjoner. Gitt at vi har tekst i en variabel kalt `s`, så kan vi bruke disse funksjonene: `s.lower()`, `s.upper()`, `s.title()`, `s.swapcase()` og `s.capitalize()`.

Legg merke til at funksjonen kalles **på** strengen - `s.lower()` - istedenfor å **gi** strengen til funksjonen - `lower(s)`.

Her er noen eksempler på hvordan funksjonene brukes (legg merke til hvilke bokstaver som er store og små i utskriften):

```
>>> s = "Per og Ada"

>>> s.upper()        # store bokstaver
'PER OG ADA'

>>> s.lower()        # små bokstaver
'per og ada'

>>> s.capitalize()   # første bokstav er stor
'Per og ada'

>>> s.title()        # første bokstav i hvert ord er stor
'Per Og Ada'

>>> s.swapcase()     # bytter stor og små
'pER oG aDA'
```

Her er noen eksempler på hva funksjonene kan brukes til:

- `s.capitalize()` brukes når vi ønsker stor forbokstav kun i begynnelsen av teksten:

  ```
  >>> sentence = "dENne sETNinGeN har IKKE riKTige bokSTAVstØrR
  eLSER."
  >>> sentence.capitalize()
  'Denne setningen har ikke riktige bokstavstørrelser.'
  ```

- `s.title()` kan brukes når vi skal skrive filmtitler:

  ```
  >>> movie_title = "star wars: a new hope"

  >>> movie_title.title()
  'Star Wars: A New Hope'
  ```

- `s.upper()` og `s.lower()` kan brukes når vi ønsker å sammenlikne tekst uten å ta hensyn størrelsen på bokstavene:

  ```
  >>> answer = "JA"

  >>> answer == "ja"  # JA og ja er ikke lik
  False
  ```

```
>>> answer = "JA"

>>> answer.lower() == "ja"  # konverter JA til ja for testen
True
```

Du må huske på at disse funksjonene **ikke** endrer på variabelen. Derfor må du lagre resultatet i en ny variabel om du vil beholde endringen din:

```
>>> s = "tekst"

>>> s.upper()  # Vi endrer ikke på variabelen!!
'TEKST'

>>> s  # Fremdeles små bokstaver
'tekst'

>>> s = s.upper()  # Nå endrer vi på variabelen

>>> s  # Denne gangen er det store bokstaver
'TEKST'
```

Lag et program som skriver ut filmtitler med store bokstaver først i hvert ord.

Programmet skal se slik ut:

```
>>>
Skriv inn en filmtittel: alice in wonderland
Alice In Wonderland
```

Dette må du gjøre:

- [ ] Be om at brukeren skriver inn en filmtittel.
- [ ] Lagre filmtittelen i en variabel.
- [ ] Manipuler strengen slik at resultatet blir som beskrevet over.
- [ ] Skriv ut den nye strengen.

Telling av tekst

Ved hjelp av `s.count()` kan vi finne ut om en streng inneholder en bestemt tekst og hvor mange ganger den finnes i strengen. For eksempel så inneholder strengen `"Hei verden!"` teksten `"verden"` en gang.

Tenk deg at du ønsker å finne ut hvor mange kommaer som er i `"A, B, C, D, E, F, G, H, I, J, K, L"`. Det er enkelt å telle for hånd, men ikke like gøy som å la datamaskinen gjøre det:

```
>>> s = "A, B, C, D, E, F, G, H, I, J, K, L"

>>> s.count(",")
11
```

Vi kan også telle tekst som er lengre, for eksempel `"Per"`:

```
>>> s = "Per, Ada, Kim, Per, Kim, Per"

>>> s.count("Per")
3
```

Lag et program som teller hvor mange ord det er i det brukeren skriver inn. Antall ord kan regnes ut ved å telle antall mellomrom, og deretter legge til 1. Forstår du hvorfor man må legge til 1?

Slik skal programmet se ut:

```
>>>
Skriv inn en streng: Hei på deg
Du skrev inn 3 ord.
```

Dette må du gjøre:

- [ ] Be brukeren om tekst.
- [ ] Lagre teksten til en variabel.
- [ ] Regn ut hvor mange ord som er i teksten.

- [ ] Skriv ut hvor mange ord teksten inneholder.

  Hint: husk å konvertere fra tall til tekst med `str()` -funksjonen.

Erstatte tekst

Vi kan bruke `s.replace()` for å bytte ut tekst i en streng med en annen tekst. Hva om vi vil bytte ut alle kommaer med semikolon?

```
>>> s = "A, B, C, D, E, F, G, H, I, J, K, L"

>>> s.replace(",", ";")
'A; B; C; D; E; F; G; H; I; J; K; L'
```

Her får `s.replace()` to argumenter - først teksten vi skal erstatte i strengen `s`, og så teksten vi skal erstatte med. Vi kan også bruke `s.replace()` for å fjerne tekst. Vi kan for eksempel fjerne alle mellomrom:

```
>>> s = "1 2 3 4 5"

>>> s.replace(" ", "")
'12345'
```

Noen operativsystemer og programmer oppfører seg rart dersom man lager filnavn med mellomrom i. Du vil derfor lage et program som bytter ut alle mellomrom med en understrek. I tillegg skal du sørge for at det bare blir brukt små bokstaver i filnavnet. Det skal fungere som i programmet under:

```
>>>
Skriv inn et filnavn: Min HemMelige FIL.txt
min_hemmelige_fil.txt
```

Dette må du gjøre:

- [ ] Spør brukeren om et filnavn

- [ ] Endre filnavnet slik som beskrevet ovenfor

- [ ] Skrive ut det nye filnavnet

Lisens: CC BY-SA 4.0 (http://creativecommons.org/licenses/by-sa/4.0/deed)