

INF1820, V2017 – Oblig 1a

Tekst i Python

Innleveringsfrist: 3 februar

I denne oppgaven skal vi jobbe med tekst i Python. Vi må derfor kunne lese inn filer og jobbe med tekststrenger. Vi skal også gjøre et første forsøk på et program som utfører såkalt **tokenisering**, dvs bryter en tekst opp i ord. Tokenisering er et nødvendig utgangspunkt for de aller fleste språkteknologiske oppgaver. Vi kommer til å fortsette å jobbe med tokenisering i neste oblig.

Innlevering av oppgaven skjer i Devilry. Se emnesiden for mer informasjon om reglement rundt innlevering, samt bruk av Devilry. Registrer svarene dine i en fil som angir brukernavnet ditt slik:

```
oblig1a_brukernavn.py
```

En perfekt løsning av denne oppgaven er verdt 100 poeng. Koden må kunne kjøres på IFIs maskiner og må inneholde kommentarer som forklarer hva koden gjør.

1 Strenger i Python (60 poeng)

Python Library Reference inneholder en oversikt over metoder som er tilgjengelig for forskjellige datatyper. For strenger (akkurat som for lister og tupler), finner du relevant informasjon på: <http://docs.python.org/library/stdtypes.html> Alternativt kan du skrive inn

```
help(str)
```

i et Python-shell for å se en oversikt over metodene som er tilgjengelig for strenger. Observer at det er flere metoder enn de som vises på skjermen og bruk “space” for å navigere nedover. Bruk “q” for å avslutte.

```
| count(...)
|     S.count(sub[, start[, end]]) -> int
|
|     Return the number of non-overlapping occurrences of
|     substring sub in
|     string S[start:end]. Optional arguments start and end
|     are interpreted
|     as in slice notation.
```

Dette betyr at du for eksempel kan bruke count slik :

```
>>> mystring = "hello"
>>> mystring.count("l")
2
```

Bruk Pythons strengmetoder for å løse følgende oppgaver.

a) **Les inn en tekstfil** Filen `dev.txt` inneholder norsk avistekst. Last ned filen fra emnesiden.

- Python er i utgangspunktet ikke så glad i norske tegn, så vi må derfor ha en spesiell kommentar som første linje i fila for å få dette til å virke:
encoding: utf-8

- For å lese inn hele filen som en streng må du åpne den, lese den og lukke den:

```
f = open("<sti-til-filen>")
filinnhold = f.read()
f.close()
```

b) Tell forekomster

- Bruk strengmetoden `count` til å avgjøre hvor mange ganger bokstavsekvensen *er* forekommer i teksten. (Du trenger ikke bare telle ordet *er*, en forekomst av feks *hjerne* skal også telles.)
- Hvor mange ord i teksten slutter på *er*? (NB! Her er vi ute etter endelsen *er* og ikke bare en forekomst hvor som helst i et ord (dvs vi vil teller *tenker* som en forekomst men ikke *hjerne*). I denne oppgaven trenger du ikke å håndtere tegnsetting; tell bare ordene som slutter på *er* og overse de som slutter på *er*, eller *er.*, dvs forekomster av *er* etterfulgt av punktum, komma eller annen tegnsetting. **Hint:** Du kan bruke strengmetodene `split` og `endswith`

c) Hent ut deler av et ord

- I Python kan sekvenstypene (strenger, tupler og lister) manipuleres med slice-notasjon. Denne notasjonen lar oss lett hente ut en del av en større sekvens:

```
>>> liste = ["a", "b", "c", "d"]
>>> liste[2:4]
["c", "d"]
>>> liste[:-1]
["a", "b", "c"]
>>> streng = "abcde"
>>> streng[2:4]
"cd"
>>> streng[:-1]
"abcd"
```

Bruk slice-notasjon til å lage en liste som inneholder de to siste bokstavene fra hvert ord i teksten.

- Sett sammen listen med endelser fra forrige oppgave til en streng av endelser med mellomrom mellom.

2 Tokenisering (40 poeng)

- a) **Les inn filen som en liste** En annen måte å lese inn en fil i Python på er å lese inn filen `dev.txt` som en liste, der elementene i listen tilsvarer en linje i filen. Dette gjør vi med en `for`-løkke, slik:

```
f = open("<sti-til-filen>")
for line in f:
    ...
f.close()
```

Vi ønsker nå å få tilgang til alle ordene i teksten. I første omgang antar vi at ordene i teksten er adskilt av mellomrom. Utvid programmet ditt slik at det deler opp hver linje basert på mellomrom. Her skal du benytte deg av en `for`-løkke.

- b) **Telle linjer og ord** Du kan nå telle antall ord. Utvid koden din med funksjonalitet for å telle ord. Her skal du ikke bruke `count`-metoden, men heller benytte deg av en teller som inkrementeres for hver runde i løkken. Hvor mange ord inneholder filen? Skriv ut resultatet.