INF-1400 Introduksjon til Python

16. januar, 2013

Informasjon

Det er lurt å undersøke om du bruker riktig versjon av Python før du begynner, boka bruker Python 3, så det kan bli lettere hvis du også bruker det (selv om det er små forskjeller). LG-laben har Python 3 installert på Linux, men ikke på Windows.

Hvor finner man informasjon om python? Et Google-søk på *python* og navnet på funksjonen eller det man ønsker å gjøre er ofte det enkleste. De første treffene er som oftest til den offisielle dokumentasjonen på `www.python.org` (som er veldig bra). En kort tutorial til python finnes her: `http://hetland.org/writing/instant-python.html`.

Oppgavene under er ment å få deg i gang med python-programmering. Gjør så mange som du rekker.

1 Lister og løkker

Oppvarming! Først, opprett ei tom fil med navnet `oppvarming.py`. Kjør scriptet med å skrive `python oppvarming.py` i terminalen. I fila skal du nå:

(a) Lage ei for-løkke som går 100 ganger fra 0 til 99 og printer alle tallene (ved bruk av Range).

(b) Modisfisere for-løkken til å kun printe partall.

(c) Lage ei liste med stringene “panther”, “tiger”, “leopard”, “snow leopard”, “windows vista”. Iterere over listen og print alle liste-elementene.

(d) Iterere over listen og print index fulgt av liste-element (hint: enumerate).

(e) Fjerne siste element i lista

(f) Sortere lista alfabetisk

2 Moduler

Bruk `oppvarming.py` eller lag ei ny fil om du vil.

(a) Importer matematikkbiblioteket (math)

(b) Bruk sinus, cosinus og/eller tangensmetoden(e).

(c) Skriv en funksjon som bruker matematikkbiblioteket til å regne ut kvadratroten av alle tallene fra 20 til 40, og skriver disse ut i terminalen.

3 Dictionary og pass by value/reference

(a) Definer en funksjon `make_phonebook()` som oppretter en dictionary og fyller den mdn noen navn(key) og tilhørende telefonnummer(value). Funksjonen skal returnere dictionaryen.

(b) Opprett en phonebook ved å kalle `make_phonebook()`. Iterer over navnene(keys) og skriv ut.

(c) Iterer over numrene(values) og skriv ut.

(d) Definer en funksjon `print_phonebook(phonebook)` som tar en dictionary som argument, og skriver denne ut på formen "*navn* har telefonnummer: *12345678*"

(e) Definer en funksjon `add_ccodes(phonebook)` som tar en dictionary som argument og legger til 47 foran alle telefonnumrene. Du kan anta at alle telefonnumre har 8 siffer.

(f) Kjør følgende kode. Hva skjer? Hvorfor?

```
name_numbers = make_phonebook()
print_phonebook(name_numbers)
add_ccodes(name_numbers)
print_phonebook(name_numbers)
```

4 Klasser og metoder

I denne oppgaven skal vi lage en enkel klasse som veldig enkelt simulerer en bankkonto. Opprett ei ny fil med navnet `account.py`

(a) Lag en klasse `Account` med metodene `deposit`, `withdrawal` og `status`. Account-klassen skal inneholde én variabel med navnet `value`. Ved å bruke metodene `deposit` og `withdrawal` skal du henholdsvis legge til og trekke fra en verdi fra `value`. `status`-metoden skal printe nåværende verdi av value.

(b) Opprett ei ny fil hvor du importerer klassen du nettopp har laget. Opprett et Account-objekt og bruk metodene til å legge til og trekke fra kontoen.

5 Guess a number!

I denne oppgaven skal vi lage et spill hvor datamaskinen plukker ut et tilfeldig tall mellom 0 og 100 og spilleren skal gjette tallet. Gjetter spilleren for høyt får han tilbakemelding om dette. På samme måte får spilleren vite om han har gjettet for lavt. Spillet avsluttes når spilleren har gjettet riktig tall.

For å løse oppgaven skal du lage en klasse Guess som ved initialisering velger et tilfeldig tall mellom 0 og 100. Klassen skal inneholde en metode `guess()` som tar inn tallet spilleren gjetter på. `guess()` returner -1, 0, eller 1, der -1 betyr at spilleren har gjettet for høyt, 1 betyr at spilleren har gjettet for lavt, og 0 betyr at spilleren har gjettet riktig. Du kan bruke `input("...")` for å lese inn tall fra terminalen.