Mentimeter

IN1000 Gruppetime Uke 5

Grupp 4: Design

Mentimeter

Plan for i dag

- Prosedyrer meg parametere og argumenter
- Funksjoner og returverdier
- Fil-lesing
- Gjøre oppgaver sammen

Mentimeter

Parametere og argumenter

- Parametere og argumenter lar oss variere resultatet av en prosedyre
- Parametere er variablene vi definerer at prosedyren vår skal ta inn
- Argumenter er verdiene vi sender inn når vi kaller på prosedyren

```
#Parametere
def prosedyre(parameter1, parameter2):
    print(parameter1)
    print(parameter2)

argument1 = "Hei"
argument2 = "Hallo"

#Argumenter vi sender inn
prosedyre(argument1, argument2)
```

Mentimeter

Hva er en funksjon?

- Likt som prosedyrer, men de returnerer en verdi
- En returverdi er verdien vi får tilbake etter et funksjonskall
- input() er for eksempel en funksjon med returverdi. Den returnerer hva enn brukeren skriver inn
- Hvis man ønsker å returnere flere verdier kan man bruke en samling

```
#Parametere
def er_partall(tall):
    returverdi = tall % 2 == 0
    return returverdi

#Funksjonskallet blir om til returverdien
#er_partall(6) = True
print(er_partall(6))
```

Mentimeter

Litt mer om return

- Return vil stoppe funksjonen
- Prosedyrer vil returnere None, altså ingenting

```
def funksjon():
    return 5
    print("Hallo")
    tall1 = 2
    tall2 = 5
    print(tall1 + tall2)

def sjekk_tall(tall):
    if type(tall) != int:
        return False

    print("Resten av koden som kjører ellers")
```

Mentimeter

```python
#Alternativ 1: En enkel funksjon
def finn_gyldige_tall(tall_liste):
    resultat = []
    for tall in tall_liste:
        if tall > 0:
            if tall % 2 == 0:
                resultat.append(tall)
    return resultat


#Alternativ 2: Flere funksjoner
def er_positivt_og_partall(tall):
    return tall > 0 and tall % 2 == 0

def finn_gyldige_tall(tall_liste):
    resultat = []
    for tall in tall_liste:
        if er_positivt_og_partall(tall):
            resultat.append(tall)
    return resultat
```

Mentimeter

```python
def analyser_tekst(tekst):
    ord = tekst.split()
    antall_ord = len(ord)
    lengste = ""
    for o in ord:
        if len(o) > len(lengste):
            lengste = o
    vokaler = 0
    for bokstav in tekst.lower():
        if bokstav in "aeiouyæøå":
            vokaler += 1
    print(f"Antall ord: {antall_ord}")
    print(f"Lengste ord: {lengste}")
    print(f"Antall vokaler: {vokaler}")
```

Mentimeter

```python
def tell_ord(tekst):
    return len(tekst.split())

def finn_lengste_ord(tekst):
    ord = tekst.split()
    lengste = ""
    for o in ord:
        if len(o) > len(lengste):
            lengste = o
    return lengste

def tell_vokaler(tekst):
    vokaler = 0
    for bokstav in tekst.lower():
        if bokstav in "aeiouyæøå":
            vokaler += 1
    return vokaler

def analyser_tekst(tekst):
    antall_ord = tell_ord(tekst)
    lengste_ord = finn_lengste_ord(tekst)
    antall_vokaler = tell_vokaler(tekst)
    print(f"Antall ord: {antall_ord}")
    print(f"Lengste ord: {lengste_ord}")
    print(f"Antall vokaler: {antall_vokaler}")
```

Mentimeter

Metoder for filer

- Lese en fil:
  - fil = open("filnavn.txt, "r")
  - "r" står for read og er default verdi
  - open("filnavn.txt, "r") = open("filnavn.txt")
- Skrive til en fil (overskrive):
  - fil = open("filnavn.txt", "w")
  - "w" står for write
- Legge til mer til en fil
  - fil = open("filnavn.txt, "a")
  - "a" står for append
- Lukke en fil med fil.close()

Mentimeter

```python
#Åpner en eksisterende fil for å lese
fil = open("tekstfil.txt", "r")

#Vi kan iterere over hver linje i filen
for linje in fil:
    print(linje)

#Åpner en ny fil for å skrive til, må ikke eksistere fra før av
ny_fil = open("nyfil.txt", "w")

#Skriver en linje til filen
ny_fil.write("Ny tekst")
```

Mentimeter

Nyttige metoder når vi leser inn fra en fil

- linje.strip()
  - Hver linje har en "\n" etter seg
  - strip() kan også bli brukt for å fjerne andre uønskede ting
- linje.split()
  - Nyttig når det er mer data på samme linje

Mentimeter

```
#Åpner en eksisterende fil for å lese
fil = open("csv.txt", "r")

navn_liste = []

#Vi kan iterere over hver linje i filen
for linje in fil:
    linje_segmenter = linje.strip().split(",")
    print(linje_segmenter)

    if linje_segmenter[0] != "Navn":
        navn_liste.append(linje_segmenter[0])

print(navn_liste)
```

Mentimeter

Del 1 - Prosedyre med parameter

Lag en funksjon skriv_ut_tallkarakter() som tar inn en bokstav-karakter (A til F) og skriver ut tall-karakteren til terminalen. Bokstavkarakterer konverteres slik:

- A: 5.0
- B: 4.0
- C: 3.0
- D: 2.0
- E: 1.0
- F: 0.0

Mentimeter

Del 2 - Funksjon med parameter

Oppdater forrige prosedyre til å returnere flyttallet.

Lag en funksjon beregn_vekttall() Som tar inn en bokstav-karakter og antall studiepoeng for et fag. Funksjonen skal returnere tall-karakteren ganget med antall studiepoeng.

Mentimeter

Del 3 - Lesing av fil

Ta utgangspunkt i en csv-fil der hver rad inneholder en fagkode, antall studiepoeng, og en bokstavkarakter.

Lag en prosedyre som tar inn navnet til filen (parameter), og som skriver ut informasjon om alle fagene, inkludert fagkode, antall studiepoeng, karakter, og vekttall

csv-fil eksempel:

IN1000,10,B
IN1020,10,C
IN1030,10,C

Mentimeter

Del 4 - Skriving til fil

Lag en prosedyre *karakterutskrift()* som tar inn et filnavn å lese av, samt et filnavn å skrive til som parameter. Skriv én linje per fag til filen, som inkluderer fagkode, studiepoeng, bokstav-karakter, og vekttall.

Nederst i filen, skriv ut hvor mange studiepoeng studenten har totalt, og det vektede karaktersnittet.