# Velkommen til tolvte gruppetime!

Gruppe 8

Uke 6: skop (scope) og parameteroverføring



# Dagens gruppetime

- **Første time:**
  - Kahoot!
  - Oppsummering

- **Andre time:**
  - Oppgaver

# Oppsummering uke 1-2 (intro, programflyt)

- input() fra bruker
- Variabler og verdier
  - Variabler peker på verdier og ikke andre variabler
- Forskjellen på å multiplisere float/int og string
- Holde styr på boolske verdier
- Ulike feilmeldinger som NameError, TypeError, SyntaxError

# Oppsummering uke 3-4 (samlinger, løkker)

- Liste, mengde, ordbok
- Nøstede samlinger
- Tjenester ulike objekter/samlinger tilbyr (.get(), .sort())
- Pass på metoder som returnerer None! (tall.sort() returnerer ingenting)
- Iterere med for-løkker
  - Forskjell "for i in range" og "for elem in elements"
- Bruk av while og for i ulike sammenhenger

# Oppsummering uke 5-6 (funksjoner, parametere, filer, skop, parameteroverføring)

- Argument vs. parameter
- Funksjoner vs. prosedyrer
- Returverdi og return
- Lese og skrive til fil (open, close)
- Ulike filer (.txt, .csv)
- Synlighet av variabler (+ mutable, immutable)
- Bruke hovedprogram()
- Teknikk for å løse tekstoppgaver

# Oppgaver

- Mange av oppgavene er basert på tidligere eksamensoppgaver
- Finn en god teknikk for hvordan å løse tekstoppgaver! (Se gjerne slides fra gruppetimen forrige uke)
- Om du ønsker kan du jobbe med Trix, oblig eller oppgaver fra tidligere gruppetimer som du mener gir deg mer nytte
- Lykke til! :D

# Oppgave 1

Skriv en funksjon, *tell_grader(fag, bsc, msc)*, som tar inn tre argumenter av datatypen string og sjekker om en bachelorgrad (bsc) og/eller en mastergrad (msc) er innen et bestemt fag.

Dersom fag har samme verdi som bsc og msc, skal funksjonen returnere 2. Dersom fag har samme verdi som enten bsc eller msc, skal funksjonen returnere 1. Dersom fag hverken er lik bsc eller msc, skal funksjonen returnere 0.

For eksempel skal kallet tell_grader("informatikk", "informatikk", "informatikk") returnere 2, mens kallet tell_grader("historie", "informatikk", "informatikk") skal returnere 0. Test funksjonen med flere eksempler på fag, bsc og msc i et hovedprogram som er definert som hovedprogram().

# Oppgave 2

Skriv en funksjon, *forkort_lagliste(lagliste)*, som tar som argument en liste av strenger som er lagnavn og returnerer en ny liste hvor ingen lagnavn finnes flere ganger i lista.

Med andre ord, dersom samme streng opptrer flere ganger i lista som funksjonen mottar som argument, skal denne bare opptre én gang i lista som blir returnert.

For eksempel skal kallet forkort_lagliste(["Brann", "Molde", "Brann"]) returnere en liste ["Brann", "Molde"]. Test funksjonen med flere eksempler på lagliste i et hovedprogram som er definert som hovedprogram().

# Oppgave 3

Programmet på neste slide leser inn fra tekstfil hvor mye henholdsvis Peter og Pål har hatt i ferieutgifter. Koden kjører og gir riktig svar, men det er en del unødvendige gjentakelser.

Du skal skrive en modifisert versjon av programmet, slik at det skriver ut det samme som det opprinnelige, men med færre gjentakelser i koden.

# Oppgave 3 (program til oppgave 3)

```
def hovedprogram():
    fn_peter = "Peter.txt"
    tot_peter = 0
    for line in open(fn_peter):
        utgift_peter = int(line)
        tot_peter += utgift_peter
        print("Peter har brukt: ", tot_peter)

    fn_paul = "Paul.txt"
    tot_paul = 0
    for line in open(fn_paul):
        utgift_paul = int(line)
        tot_paul += utgift_paul
        print("Paul har brukt: ", tot_paul)
hovedprogram()
```

# Oppgave 4

Skriv en funksjon, *fjern_vokaler(setning, vokalliste)*, som tar inn en setning (datatype string) og en liste med vokaler (hvor hvert element i lista er en string av lengde 1).

Funksjonen skal returnere en kopi av setning, hvor alle vokaler fra vokalliste er fjernet. Du kan anta at både setning og vokalliste kun inneholder små bokstaver.

For eksempel skal kallet fjern_vokaler("ha det fint", ["a", "e", "i", "o", "u"]) returnere teksten "h dt fnt". Test funksjonen med flere eksempler på setning og vokalliste i et hovedprogram som er definert som hovedprogram().

# Oppgave 5

Skriv en funksjon, *er_sortert(talliste)*, som tar en liste med tall som argument og som returnerer en boolsk verdi.

Funksjonen skal sjekke om alle tallene i lista er i stigende rekkefølge (sorterte). Dersom alle verdiene er i stigende rekkefølge, skal funksjonen returnere True, ellers skal funksjonen returnere False. Anta at alle tallene i lista er ulike. Funksjonen trenger altså ikke ta hensyn til eventuelle like verdier.

For eksempel skal er_sortert([2, 4, 6, 8, 10]) returnere True og er_sortert([7, 3, 1, 9]) returnere False. Test funksjonen med flere tallister i et hovedprogram som er definert som hovedprogram().

# Oppgave 6

Denne oppgaven bygger videre på **oppgave 5**. Lag et hovedprogram som legger inn alle sorterte lister i én ordbok og alle usorterte lister i en annen ordbok. Nøklene i ordboka for de sorterte listene er på formen 1s, 2s, 3s, o.s.v. I ordboka for de usorterte listene er nøklene på formen 1u, 2u, 3u, o.s.v.

Når de to ordbøkene er laget, skal dere bruke ordbøkene til å lage en liste med lengde lik 2. Det første elementet i lista er en liste med sorterte lister, og det andre elementet i lista er en liste med usorterte lister.

Til slutt skal programmet skrive ut en oversikt over de sorterte listene og en oversikt over de usorterte listene. Husk passende overskrifter i utskriften, og sørg for at hver av listene skrives ut på nye linjer. Dere skal ikke skrive ut elementene i de sor

# Ferdig?

- Tenk tilbake på ukene til nå, hva er noe du sliter med i IN1000?
  - Jobb med det
  - Bruk filter på Trix
- Sammenlign løsningen din med løsningsforslag på neste slides (disse er ikke synlige før slutten av timen)

# Løsningsforslag oppgave 1

```
def tell_grader(fag, bsc, msc):
    antall = 0
    if fag == bsc:
        antall += 1
    if fag == msc:
        antall += 1
    return antall

def hovedprogram():
    print(tell_grader("informatikk", "informatikk", "informatikk"))
    print(tell_grader("informatikk", "informatikk", "fysikk"))
    print(tell_grader("informatikk", "fysikk", "informatikk"))
    print(tell_grader("historie", "informatikk", "informatikk"))

hovedprogram()
```

# Løsningsforslag oppgave 2

```
def forkort_lagliste(lagliste):
    ny_liste = []
    for lag in lagliste:
        if lag not in ny_liste:
            ny_liste.append(lag)
    return ny_liste

def hovedprogram():
    print(forkort_lagliste(["Brann", "Molde", "Brann"]))
    print(forkort_lagliste(["Odd", "Odd", "Odd"]))
    print(forkort_lagliste([]))
    print(forkort_lagliste(["Vikings", "Brann", "Rosenborg"]))

hovedprogram()
```

# Løsningsforslag oppgave 3

```
def skriv_utgifter(filnavn, navn):
    total = 0
    fil = open(filnavn)
    for linje in fil:
        total += int(linje)
        print(navn + " har brukt: ", total)
    fil.close()

def hovedprogram():
    skriv_utgifter("Peter.txt", "Peter")
    skriv_utgifter("Paul.txt", "Paul")

hovedprogram()
```

# Løsningsforslag oppgave 4

```
def fjern_vokaler(setning, vokalliste):
    resultat = ""
    for bokstav in setning:
        if bokstav not in vokalliste:
            resultat += bokstav
    return resultat

def hovedprogram():
    vokaler = ["a", "e", "i", "o", "u"]
    print(fjern_vokaler("ha det fint", vokaler))
    print(fjern_vokaler("aeiou", vokaler))
    print(fjern_vokaler("python", vokaler))
    print(fjern_vokaler("ha det fint", ["a", "i"]))

hovedprogram()
```

# Løsningsforslag oppgave 5

```
def er_sortert(talliste):
    for i in range(len(talliste) - 1):
        if talliste[i] > talliste[i + 1]:
            return False
    return True

def hovedprogram():
    print(er_sortert([2, 4, 6, 8, 10]))
    print(er_sortert([7, 3, 1, 9]))
    print(er_sortert([5]))
    print(er_sortert([]))
    print(er_sortert([1, 2, 4, 3]))

hovedprogram()
```

# Løsningsforslag oppgave 6

Lang kode, her er link: https://tinyurl.com/mr2933xb