print("Hello")

print("Dette programmet er en glimrende kalkulator. La meg vise deg")

print(4+2)

print(10-4)

print(10\*3)

print(12/2)

print(2\*\*3)

print(2\*3-(2\*\*4-4))

#En annen morsom ting vi kan gjøre i Python er bruke variabler.

a = 5

b = 2

print(a+b)

print(a-b)

print(a\*b)

print(a/b)

print(a\*\*b)

#Noe enda morsommere vi kan gjøre er å bruke kommandoen input for å få brukeren til å skrive inn noe

#int er heltall

ola = int(input("Skriv inn et heltall: "))

fola = int(input("Skriv inn et annet heltall: "))

fasit = ola \* fola

print("Hvis du ganger sammen",ola,"med",fola,"får du",fasit)

#float er desimaltall

ola = float(input("Skriv inn et heltall: "))

fola = float(input("Skriv inn et annet heltall: "))

fasit = ola \* fola

print(f"Hvis du ganger sammen {ola} med {fola} får du {fasit}")

#Legg merke til at vi kan bruke f-strengen for å slippe å bruke så mange komma

#Vi kan få Python til å finne ut om tall er større eller mindre enn noe ved å bruke IF

b = int(input("Skriv inn et heltall: "))

if b > 0 :

print("Tallet ditt er større enn 0")

elif b < 0 :

print ("Tallet er mindre enn 0")

else:

print("Tallet ditt er null")

#Vi kan også sikre oss at brukeren skriver inn riktig tall

ola = float(input("Skriv inn et tall: "))

fola = int(input("Vil du gange tallet ditt med 10, 100 eller 1000?"))

# == betyr er lik ,( bare et = betyr at vi skal lage en variabel)

if fola == 10 :

svar = ola \* 10

print("Tallet ditt er", svar,)

elif fola == 100 :

svar = ola \* 100

print("Tallet ditt er", svar,)

elif fola == 1000 :

svar = ola \* 1000

print("Tallet ditt er", svar,)

else :

print("Du skrev inn feil tall. Prøv igjen")

#For-løkker kjører et fast antall ganger. Her bruker vi «range» for å lage tallrekker. Vi bestemmer henholdsvis hvor mange ganger, hvor det starter og steg-tallet. Dette gjøres ved å legge til flere tall med komma mellom

for n in range(100):

print(n)

for n in range(70,100):

print(n)

for n in range(0,100,10):

print(n)

#tenk deg en tall-linje som begynner på 0 og fortsetter 1,2,3,4 osv

#vi skal nå lage en program som summerer tallene på den linja helt opp til et antall ganger som brukeren bestemmer

ola = int(input("Skriv inn et heltall: "))

sum = 0

for banan in range(ola+1):

sum = banan + sum

print(f"Summen av {banan} tall i rekka er {sum}")

print(f"Summen av {banan} tall i rekka er {sum}")

#legg merke til hvordan resultatene endrer seg ved hvor du plasserer print i skriptet. Om du har det til venstre vises bare siste resultat. Om du legger den under for-løkka ser vi at resultatene kommer hver gang

#for-løkker er derfor glimrende å bruke til å lage oppgaver med renters rente

print("Dette programmet regner ut hva en startsum vokser til etter ett, to og opp til og med et valgt antall år med en valgfri rente per år.")

startsum = float(input("Skriv inn den summen du ønsker å sette inn:"))

maksår = int(input("Skriv inn hvor mange år summen skal stå totalt:"))

rente = float(input("Skriv inn hva renten skal være:"))

vekstfaktor = 1 + rente/100

for år in range(1, maksår + 1):

sum = startsum \* vekstfaktor\*\*år

print(f"Etter {år} år har {startsum:.2f} kroner vokst til {sum:.2f} kroner.")

#Der .2f bare betyr at man skal runde av til to desimaler og ha svaret på desimalform(f=fractions)

#Du kunne like gjerne skrevet det sånn her

#print("Etter",år," år har", round(startsum,2),"vokst til ",round(sum,2) , "kroner")

print(f"Renten er {rente} prosent.")

#while løkker kjøres så lenge en betingelse er sann

print("Dette programmet regner ut hvor lang tid det tar før et beløp har vokst seg til et annet beløp gitt en fast rente ")

startsum = float(input("Skriv inn den summen du ønsker å sette inn:"))

onsketsum = float(input("Skriv inn hva ønsket sum er: "))

rente = float(input("Skriv inn hva renten skal være:"))

vekstfaktor = 1 + rente/100

år=0

while startsum < onsketsum :

startsum = startsum \* vekstfaktor

år = år + 1

print(år) #gir antall år det tar

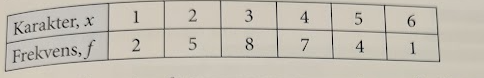
print(round(startsum,1)) #gir summen du ender opp med

#spyder kan importere biblioteker som gir deg flere kommandoer/muligheter en sånn er statistics, i dette biblioteket er vi interessert i gjennomsnitt, typetall, median og standardavvik. Du ser dem under hva de heter

from statistics import mean, mode, median, pstdev

#importerer gjennomsnitt, typetall, median og standardavvik.

#vi lager vår egen tabell basert på denne tabellen



frekvenstabell = [(1,2),(2,5),(3,8),(4,7),(5,4),(6,1)]

liste = []

#lager en tom liste

for (timer,antall) in frekvenstabell:

liste += antall\*[timer]

print(liste)

# += betyr at legger adderer inn observasjonene etter hvert som programmet leser av frekvenstabellen. Du ser det tydelig om du prøver med og uten + tegnet, men husk at print(liste) ikke må stå under liste. Ellers vises bare den fullførte lista. Står den under så blir den en del av løkka og du kan lett se hva += gjør, kontra bare +

print(liste)

#står den sånn så kommer bare den fullførte lista

#under regner vi ut litt av hvert

print(round(mean(liste),2))

print(mode(liste))

print(median(liste))

print(round(pstdev(liste),2))

#python kan så klart regne funksjoner. Def betyr at vi skal gi navn til funksjonen vår. Return er selve funksjonen. Man kan deretter printe ut et hvert svar av f(x) (altså y) ved å endre x-verdi

def f(x) :

return -2\*x\*\*2+3\*x-3

print (f(2))

def opprinneligverdi(år):

return 5000\*0.95\*\*år

nypris = round(opprinneligverdi(10),2)

print(nypris)

#og kan så klart også lage grafer, men da må vi importere biblioteker som gir oss tilgang til dette

import numpy as np

import matplotlib.pyplot as plt

#lager meg en funksjon

def f(x):

return -x\*\*2+2\*x+10

x = np.arange (-3,5,0.1) #bestemmer startverdi, sluttverdi og steg-verdi (bytt ut siste med 1 så ser du)

y = f(x)

plt.plot(x,y)

plt.show()

#vi kan så klart laste inn pi og kvadratrøtter

from math import pi, sqrt

radius = 2.6

volum = 4/3\*pi\*radius\*\*3

overflateareal = 4\*pi\*radius\*\*2

print("Volumet er", round(volum, 2))

print("Overflatearealet er", round(overflateareal, 2))

print(sqrt(9))