løse eksponentiell og logaritme likninger

Oppgavene under viser to tilnærmninger for å løse likninger med programmering

Oppgave 8.14

Skriv et program som bestemmer en heltallig (tilnærmet) løsning
av likningen

$x \cdot \lg(x) = 150$

når du får vite at løsningen ligger mellom 10 og 100.

Vi skal lage først en enkel algortime for å finne en tilnærmet heltall som løsning for likning ved hjelp av en while-løkke:

```python
import numpy as np

def f(x):
    return x*np.log10(x)

x = 10
while f(x) <= 150:
    x += 1

print(f"x = {x} er en tilnærmet løsning til x·lg(x) = 150 ")
```

```
x = 80 er en tilnærmet løsning til x·lg(x) = 150
f(80)= 152.24719895935547
```

Dette er tilnærming vi har tatt:

1. Vi starte med x = 10, siden løsning er mellom 10 og 100
2. Vi øker x med 1 så lenge f(x) < 150 og x <100 100
3. Vi stopper når f(x) er større en 150
4. Vi velger x-1 som en tilnærmet løsning til likning

Vi kunne ha skrevet samme funksjonalitet med en for-løkke

```python
#skriv din kode her
import numpy as np

def f(x):
    return x*np.log10(x)

for x in range(10, 101):
    if x > 150:
        break

print(f"x = {x} er en tilnærmet løsning til x·lg(x) = 150 ")
```

```
x = 100 er en tilnærmet løsning til x·lg(x) = 150
```

Vi får x= 80 og vi ser at f(80)= 152.25, dvs. den har vippet over 150. Dette er fordi løsning ikke vær en heltall. Kjør kode under og forklar at løsning finnes mellom 79 og 80.

```python
print(f"f(79))= {79*np.log10(79)}")
print(f"f(80)= {80*np.log10(80)}")
```

```
f(79))= 149.91254021194487
f(80)= 152.24719895935547
```

En andre alternativ som gir oss en bedre tilnærming til løsning er å enten øke med mindre enn 1 (f.eks. x = x + 0.001) eller bruke linspace for å lage verdier til x med desimaler. Når det blir større en 150 stopper vi opp, og runde opp når vi finne en tilnærmet verdi av nullpunkt.

```python
#skriv din kode her
import numpy as np

def f(x):
    return x*np.log10(x)

for x in np.linspace(10, 100, 1000):
    if f(x) > 150:
        break
print(f"x = {int(x)} er en tilnærmet løsning til x·lg(x) = 150 ")
```

```
x = 79 er en tilnærmet løsning til x·lg(x) = 150
```

Oppgave 8.18

Algoritme:

1. Sett x1 = 2
2. Sett x2 = 5
3. Regn m = (x1+x2)/2
4. Så lenge abs(f(m)) >= 0.01:

Funksjonen f er gitt ved $f(x) = x^3 - 4x^2 + x$, der f(2) < 0 og f (5) > 0.

Ta utgangspunkt i algoritmen ovenfor, og skriv kode der du bruker halveringsalgoritmen for å bestemme nullpunktet til f (x).

Siden f(2) < 0 og f (5) > 0 (skifte fortegn), vet vi at nullpunktet ligger i intervall [2,5].

```
def f(x):
    return x**3 -4*x**2 + x

x1 = 2
x2 = 5
m = (x1 + x2)/2

while abs(f(m)) >= 0.01:
    if f(x2)*f(m) < 0:
        x1 = m
    else:
        x2 = m
    m = (x1 + x2)/2


print(f"x={round(m,3)} er en tilnærmet verdi av en av nullpunkter til f (x) = x^3 - 4x^2 + x  ")
```

```
x=3.731 er en tilnærmet verdi av en av nullpunkter til f (x) = x^3 - 4x^2 + x
```

Hvis du har halveringsmetode ferdig skrevet som en funksjon, du kan skrive kode som følge:

```
from Standard_algoritmer import *

nullpunkt = halveringsmetode(f, 2,5)

print(f"x={round(nullpunkt,3)} er en tilnærmet verdi av en av nullpunkter til f (x) = x^3 - 4x^2 + x  ")
```

```
x=3.732 er en tilnærmet verdi av en av nullpunkter til f (x) = x^3 - 4x^2 + x
```

Oppgave 8.19

Gitt likningen $lg(x + 3) = x - 3$.
Du skal løse likningen ved hjelp av halveringsmetoden.
Du får vite at f (1) er positiv, og f (6.5) er negativ.
Bruk dette til å skrive et program som løser likningen ved hjelp av halveringsmetoden. (Løsningen med to desimalers nøyaktighet skal være x = 3,83).

```python
import numpy as np
def f(x):
    return np.log10(x + 3) - x + 3

a = 1
b = 6.5
nullpunkt = halveringsmetode(f, a, b, 1E-8)
print(f"x = {round(nullpunkt, 2)} er en tilnærmet verdi av nullpunkt")
```

```
x = 3.83 er en tilnærmet verdi av nullpunkt
```

Du kunne også ha bruk Newtonsmetode, som finne nullpunker til en funksjon.