edbookPython

March 31, 2024

[24]:
```
import math
from tabulate import tabulate
from urllib.request import urlopen
```

1 Verkefni: Annars stigs jafna

Skrifið forrit sem leysir annars stigs jöfnu

$$ax^2 + bx + c = 0$$

Forritið á að lesa inn a, b og c með `input`-skipunum (með viðeigandi beiðnum til notanda). Ef jafnan hefur tvær lausnir á forritið að skrifa „Lausnirnar eru:“ og síðan lausnirnar, ef hún hefur eina lausn á að skrifa hana með viðeigandi skýringu og ef engin lausn er skal skrifa skilaboð um það. Lausn eða lausnir eru gefnar með formúlunni

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Ef stærðin undir kvaðratrótinni er neikvæð þá er engin lausn, ef hún er núll er ein lausn, annars tvær. Gerið auk þess ráð fyrir þeim möguleika að a sé 0. Ef b er ekki líka 0 þá er jafnan fyrsta stigs og forritið á að skrifa að svo sé, ásamt lausninni (sem er þá ein). Látið b og c vera gefin með afmælisdegi ykkar (dagur og mánuður) og prófið forritið fyrir fjórar mismunandi jöfnur, fyrsta stigs jöfnuna $bx + c = 0$ og annars stigs jöfnur sem hafa enga, eina og tvær lausnir þar á meðal jöfnuna $x^2 + bx + c = 0$. Setjið viðeigandi skjölunarstreng fremst í forritið.

[1]:
```
a = int(input("Veldu tölu: "))
b = int(input("Veldu tölu: "))
c = int(input("Veldu tölu: "))

d = (b**2 - (4 * a * c))

if a==0:
    print("Þetta er línuleg jafna með lausnina: ")

elif d<0:
    print("Það er engin lausn")
else:
```

```
    print("Fyrri lausn er: ", (-b + math.sqrt(d))/(2*a))
    print("Seinni lausn er: ", (-b - math.sqrt(d))/(2*a))
```

```
Það er engin lausn
```

2 Töluleg heildun

2.1 A. Samsett trapisuregla

Í sýnidæmi og æfingu í kafla 4.9 voru búin til föll til að nálga heildi með flatarmáli einnar eða tveggja trapisa. Föllin voru prófuð með heildunum:

$$(*) \qquad \int_1^2 \frac{\sin(x)}{x} dx \qquad \text{og} \qquad (**) \qquad \int_0^1 e^x dx$$

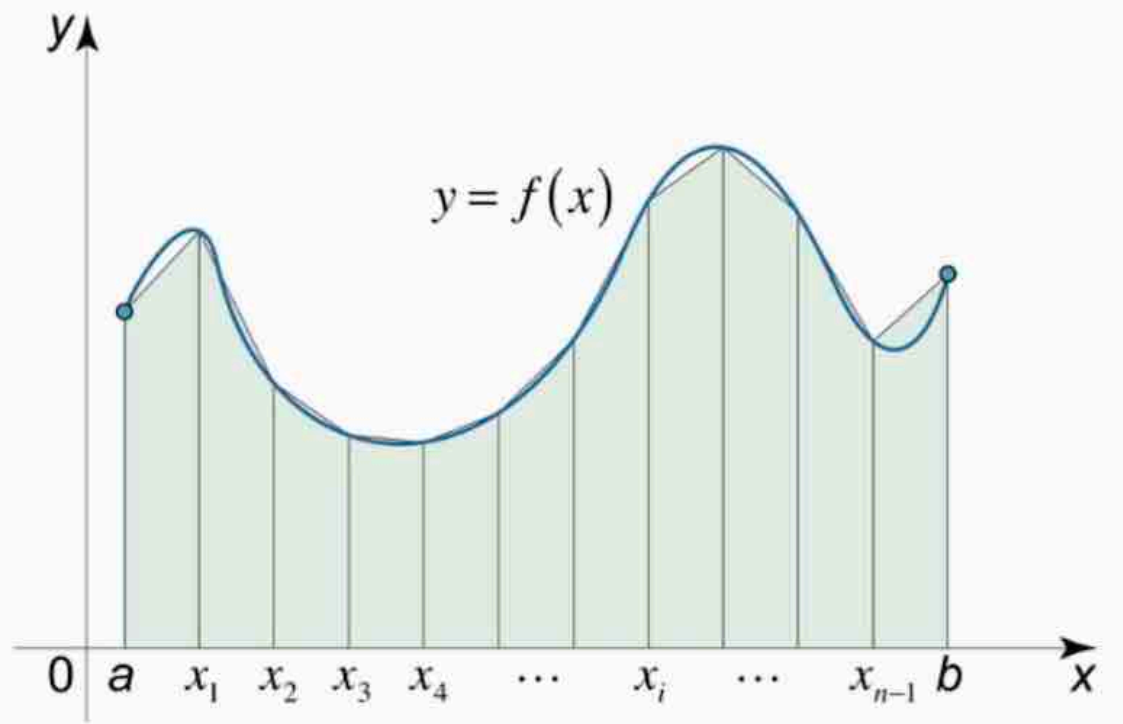
og fengust eftirfarandi nálganir:

	(*)	(**)
1 trapisa	0.648	1.859
2 trapisur	0.657	1.754
rétt gildi	0.6593299	1.7182818

Enn betri nálgun fæst með samsettri trapisureglu sem notar n trapisur. Forritið hana og prófið með (*) og (**) og $n = 4$. Kannið líka hve stórt n þarf að vera til að fá (næstum) 7 rétta aukastafi, sbr. uppgefin rétt gildi. Hér eru formúlurnar sem þarf að nota ásamt skýringarmynd:

Bilinu $[a, b]$ er skipt upp í n hlutbil sem hvert hefur lengd $x = \Delta\frac{b-a}{n}$.

Skiptipunktarnir eru: $a = x_0 < x_1 < x_2 < \cdots < x_n = b$ þar sem $x_i = a + i\Delta x$.



þá fæst nálgunin:

$$\int_a^b f(x)\,dx \approx T_n = \frac{\Delta x}{2}(f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(x_n))$$

[4]:
```
def trap1(f,a,b,n):
    '''heildi f frá a til b með ósamsettri trapisureglu'''

    s = f(a)+f(b)
    dx = (b-a)/n
    for i in range (1,n):
        xi = a+i*dx
        s += 2*f(xi)
    return dx/2*s

def g(x): return math.sin(x)/x
def h(x): return math.e**x

I1 = trap1(g,1,2,4)
I2 = trap1(h,0,1,4)
print(I1, I2)
```

```
0.6586304715647094 1.7272219045575168
```

2.2 B. Simpsons-regla

Skrifa skal forrit til að nálga heildi með svonefndri Simpsons-regla. Í trapisureglu er heildisbilinu skipt í n

hlutbil, fallið sem heilda skal nálgað með beinum línustrikum og heildi þess nálgað með flatarmálinu undir þessum línustrikum. Í Simpsonsreglu er fallið hinsvegar nálgað (eða brúað eins og það er kallað) með parabólum og heildið nálgað með flatarmálinu undir þeim. Skoðið endilega Wikipedíugrein um aðferðina.

Simpsons-formúlan er eftirfarandi:

$$\int a^b f(x)\,dx \approx \frac{\Delta x}{3}\left(f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + \cdots + 4f(x_{n-1}) + f(x_n)\right)$$

þar sem Δx og x_i eru eins og í A-lið og n

er slétt tala.

Skrifið fall simpson(f,a,b,n) sem nálgar heildið af f frá a til b með samsettri Simpsons-reglu með n hlutbilum.

Prófið með heildunum (*) og (**) úr A-lið með 4 hlutbilum. Ef rétt er forritað ætti að fást (∗) 0.65933 og (∗∗) 1.71832. Kannið líka hve stórt þarf að vera til að fá alla 7 aukastafina sem gefnir eru í töflunni í A-lið rétta. Heildið a lokum eitthvert sjálfvalið fall þar sem afmælisdagur ykkar kemur við sögu.

[3]:
```
def simpson(f,a,b,n):
    dx = (b-a)/n
    s = f(a)+f(b)
    for i in range (1,n):
        xi = a+i*dx
        if i%2==0:
            s += 2*f(xi)
        else:
            s += 4*f(xi)
    return (dx/3)*s


def g(x): return math.sin(x)/x
def h(x): return math.e**x
def z(x): return math.sin(2*x)*math.cos(2*x)

I1 = simpson(g,1,2,2)
I2 = simpson(h,0,1,4)
I3 = simpson(z,24,93,11)
print(f"Simpsons-reglan á fyrri jöfnu tveimur hlutblium hefur svarið {I1},␣
 ↪hinsvegar hefur seinni jafnan svarið {I2} ef skipt er í 4 hlutbita.")
print(f"Ég er fæddur 24.11.1993 og nota því jöfununa með þeim dagsetningum.␣
 ↪Svarið er því {I3}")
```

```
Simpsons-reglan á fyrri jöfnu tveimur hlutblium hefur svarið 0.6593510548608137,
hinsvegar hefur seinni jafnan svarið 1.718318841921747 ef skipt er í 4 hlutbita.
Ég er fæddur 24.11.1993 og nota því jöfununa með þeim dagsetningum. Svarið er
því 33.14483895863248
```

3 Ýmis dæmi

3.1 A. Hitastigum breytt

Hitastig í Fahrenheitgáðum er $T_F = \frac{9}{5}T_C + 32$ þar sem T_C er hitastigið í Celcíusgráðum. Skrifið forrit sem skrifar út töflu til að breyta milli skalanna sem nær frá –30°C til 50oC og hleypur á 5oC. Á Makka fæst gráðumerki með option-T og á Windows með Alt-0176. Taflan gæti byrjað svona:

°C	°F
-30	-22
-25	-13
-20	-4
-15	5
...	...

[3]:
```
def CelciusToFahrenheit():
    ctof = []
    for i in range(-30, 51, 5):
        x = ((9 / 5) * i) + 32
        ctof.append([i, x])
    print(tabulate(ctof,headers=["Celsius","Fahrenheit"]))

CelciusToFahrenheit()
```

```
  Celsius    Fahrenheit
---------  ------------
      -30           -22
      -25           -13
      -20            -4
      -15             5
      -10            14
       -5            23
        0            32
        5            41
       10            50
       15            59
       20            68
       25            77
       30            86
       35            95
       40           104
```

```
45              113
50              122
```

3.2 B. Rætur, lograr, kvaðröt

1. Skrifið forritsbút með for-lykkju sem finnur kvaðratrætur, náttúrulega lograr, og önnur veldi talnanna 1–10. Notið F-strengi til að skrifa (fallega) töflu yfir niðurstöðuna. Hafið 2 aukastafi í kvaðratrótunum og logrunum.
2. Endurtakið með while-lykkju.

[4]:
```
def RLK(n, k):
    for i in range(n,k+1):
        squareroot = math.sqrt(i)
        logarithm = math.log(i)
        square = math.pow(i, 2)
        print(f"Ferningsrót {i} er {squareroot:.2f}, lógaritmi {i} er␣
 ↪{logarithm:.2f} og {i} í öðru veldi er {square}")
    while n<=k:
        squareroot = math.sqrt(n)
        logarithm = math.log(n)
        square = math.pow(n, 2)
        print(f"Ferningsrót {n} er {squareroot:.2f}, lógaritmi {n} er␣
 ↪{logarithm:.2f} og {n} í öðru veldi er {square}")
        n+=1
RLK(1,10)
```

```
Ferningsrót 1 er 1.00, lógaritmi 1 er 0.00 og 1 í öðru veldi er 1.0
Ferningsrót 2 er 1.41, lógaritmi 2 er 0.69 og 2 í öðru veldi er 4.0
Ferningsrót 3 er 1.73, lógaritmi 3 er 1.10 og 3 í öðru veldi er 9.0
Ferningsrót 4 er 2.00, lógaritmi 4 er 1.39 og 4 í öðru veldi er 16.0
Ferningsrót 5 er 2.24, lógaritmi 5 er 1.61 og 5 í öðru veldi er 25.0
Ferningsrót 6 er 2.45, lógaritmi 6 er 1.79 og 6 í öðru veldi er 36.0
Ferningsrót 7 er 2.65, lógaritmi 7 er 1.95 og 7 í öðru veldi er 49.0
Ferningsrót 8 er 2.83, lógaritmi 8 er 2.08 og 8 í öðru veldi er 64.0
Ferningsrót 9 er 3.00, lógaritmi 9 er 2.20 og 9 í öðru veldi er 81.0
Ferningsrót 10 er 3.16, lógaritmi 10 er 2.30 og 10 í öðru veldi er 100.0
Ferningsrót 1 er 1.00, lógaritmi 1 er 0.00 og 1 í öðru veldi er 1.0
Ferningsrót 2 er 1.41, lógaritmi 2 er 0.69 og 2 í öðru veldi er 4.0
Ferningsrót 3 er 1.73, lógaritmi 3 er 1.10 og 3 í öðru veldi er 9.0
Ferningsrót 4 er 2.00, lógaritmi 4 er 1.39 og 4 í öðru veldi er 16.0
Ferningsrót 5 er 2.24, lógaritmi 5 er 1.61 og 5 í öðru veldi er 25.0
Ferningsrót 6 er 2.45, lógaritmi 6 er 1.79 og 6 í öðru veldi er 36.0
Ferningsrót 7 er 2.65, lógaritmi 7 er 1.95 og 7 í öðru veldi er 49.0
Ferningsrót 8 er 2.83, lógaritmi 8 er 2.08 og 8 í öðru veldi er 64.0
Ferningsrót 9 er 3.00, lógaritmi 9 er 2.20 og 9 í öðru veldi er 81.0
Ferningsrót 10 er 3.16, lógaritmi 10 er 2.30 og 10 í öðru veldi er 100.0
```

3.3 C. Lograr og veldi

Til að reikna logra (lógaritma) hefur Python þrjú föll: Náttúrulegur logri fæst með `log(x)`, tíu-logri fæst með `log10(x)` og loks fæst logri með grunntölu 2 með `log2(x)`. Fastinn fæst með `math.e`, og vísisfallið, e^x, fæst með `exp(x)`. Reiknið: 1. ln(1) 2. lnexp(3) 3. log101000 4. log28

(Ætti að gefa 0, 3, 3 og 3)

[ ]:
```
def LogAndPower(log, logexp, log10, log2):
    logarithm = math.log(log)
    logexponent = math.log(math.exp(logexp))
    logarithm10 = math.log(log10, 10)
    logarithm2 = math.log(log2, 2)
    print(f"""Lógaritmi af {log} er {logarithm:.0f}, lógaritmi af e í veldinu␣
 ↪{logexp} er {logexponent},
lógaritmi af {log10} með grunntöluna 10 er {logarithm10:.0f} og lógaritmi af␣
 ↪{log2} með grunntöluna 2 er {logarithm2:.0f}""")
LogAndPower(1, 3, 1000, 8)
```

3.4 D. Rúmmál kúlu

Þetta dæmi og það næsta (taflborðið) eru æfingar í útprentun, en ekki síður æfingar í að byggja upp forrit úr minni einingum, sem maður prófar hverja fyrir sig og setur svo saman. 1. Skrifið Python fall sem reiknar og skilar rúmmáli kúlu með radíus r. Prófið með kúlu með radíus 2 (ætti að skila 33.51) 2. Skrifið fall sem reiknar og skilar radíus hrings með gefið ummál. Prófið. 3. Skrifið fall sem tekur inn eðlismassa hlutar í g/cm^3 og rúmmáli hans í m^3 og skilar þyngd hans í kg. Notið fallið til að reikna þyngd gulltenings sem er 20 cm á kant (eðlismassi gulls er 19.30; ætti að skila 154.4 kg). 4. Jörðin er um það bil kúla með ummál 40 þúsund km og eðlisþyngd hennar er 5.5 g/cm^3. Notið föllin í liðum 1–3 til að finna massa jarðar í tonnum.

Látið öll forritin skrifa stuttan skýringartexta með því sem er prentað.

[ ]:
```
def volumeOfSphere(radius):
    return (4/3)*math.pi*(radius**3)

def radiusOfSphere(circumference):
    return circumference/(2*math.pi)

def weightOfGold(side, densityOfGold):
    return side * densityOfGold*1000

radius = 2
print(f"Rúmmál kúlu með radíus {radius} er ", round(volumeOfSphere(radius), 2))

circumference = 100
print(f"Radíus hrings með ummálið {circumference} er",␣
 ↪round(radiusOfSphere(circumference), 2))
```

```
side = 20
densityOfGold = 19.30
print(f"Eðlismassi gulltenings sem er {side} cm á kant og hefur eðlisþyng␣
 ↪{densityOfGold} er {weightOfGold((20/100)**3, 19.3):.2f}")

circumference = 40000
density = 5.5
radiusOfEarth = radiusOfSphere(circumference*10**4)
volueOfEarth = volumeOfSphere(radiusOfEarth)
massOfEarth = volueOfEarth * (density*10**3)
print(f"Massi jarðar með ummálið {(circumference/1000):.0f} km og eðlismassan␣
 ↪{density} g/cm^3 er {massOfEarth/(10**6)} tonn")
```

3.5 E. Taflborð

Lokamarkmiðið þessa dæmis er að búa til fall sem prentar út nxn taflborð fyrir slétta tölu n sem lítur svona út þegar $n - 8$:

```
+-----------------+
|   X   X   X   X |
| X   X   X   X   |
|   X   X   X   X |
| X   X   X   X   |
|   X   X   X   X |
| X   X   X   X   |
|   X   X   X   X |
| X   X   X   X   |
+-----------------+
```

Það eru þrjár gerðir af línum:

a. Efsta og neðsta línan, með +, $2n + 1$ striki og aftur +.

b. Línur eins og önnur línan með þrjú bil og X endurtekið $\frac{n}{2}$ sinnum.

c. Línur eins og þriðja línan með X og þrjú bil endurtekið $\frac{n}{2}$ sinnum.

1. Skrifið þrjú föll, til að prenta línur af tagi a, b og c. Prófið hvert um sig fyrir $n = 2$ og $n = 8$.
2. Skrifið fall sem kallar á föllin í lið 1. og prentar nxn borð. Látið fallið skrifa villuboð og hætta ef n er ekki slétt tala > 2. Prófið með $n = 8$

Munið að nota `*`-virkjann, sem fjölfaldar strengi.

[ ]:
```
def Chess(n):
    if n<2 or n%2!=0:
        return ValueError("Please pick a number higher than 2 that is not an␣
 ↪even number.")
    lines = "-"*2*(n+1)
    newline = lines.replace("-", '', 1)
    print("+", newline, "+", sep='')
    for i in range(0,n):
```

```
        print("|", end=' ')
        for t in range(0,n):
            if (i+t)%2==0:
                print(" ", end=' ')
            else:
                print("X", end=' ')
        print("|")
    print("+", newline, "+", sep='')
Chess(int(input()))
```

3.6 F. Viðsnúningur lista

1. Skrifið fall hali(L) sem skilar hala L, þ.e. lista með öllum stökum nema því fremsta.
2. Skrifið fall hausaftast(L) sem skilar nýjum lista þar sem haus L hefur verið færður aftast. Ef kallað er með $L = [1, 2, 3, 4]$ ætti fallið að skila $[2, 3, 4, 1]$. Prófið líka með lista búnum til útfrá afmælisdegi (t.d. 3.8.1999 $\rightarrow [3, 8, 99]$).
3. Skrifið fall snúavið(L) sem snýr við lista. Hér er reiknirit:

```
fall snúavið(L)
    n := lengd L
    M := tómur listi
    fyrir i = n-1,n-2,...,0:
        setja i-ta stak L afast í M
    skila M
```

Prófið að snúa við $L = [1, 2, 3, 4]$ sem ætti að skila $[4, 3, 2, 1]$ og líka afmælisdagalistanum úr lið 2.

[13]:
```
#1
def hali(L):
    return L[1::]

#2
def hausaftast(L):
    L.append(L.pop(0))
    return L

#3
def snuavid(L):
    n = len(L)
    M = []
    for i in range(n-1,-1,-1):
        M.append(L[i])
    return M

L=[1,2,3,4]
print(hausaftast(L))

birthday = [24,11,93]
```

```
print(hausaftast(L))

L=[1,2,3,4]
print(snuavid(L))

birthday = [24,11,93]
print(snuavid(birthday))
```

```
[2, 3, 4, 1]
[3, 4, 1, 2]
[4, 3, 2, 1]
[93, 11, 24]
```

4 Orðaleikur

1. Skrifið fall `telja(s)` sem byrjar á nota `s.split()` til að búa til lista af einstökum orðum í `s` og telur síðan hve margir stafir eru í hverju orði. Fallið á að skila lista með þessari talningu. Þannig ætti

   `telja("Afi minn og amma mín")`

   að skila listanum [3,4,2,4,3]. Prófið líka að telja stafi í nafninu ykkar (t.d. telja("Kristján Jónasson")). Líklega borgar sig að búa fyrst til tóman lista og fara svo í lykkju og bæta nýrri talningu afast í listann í hverri umferð (t.d. með `append`).
2. Búið til fall `hrópa(s)` sem skrifar `s` út með stórum stöfum og tveimur upphrópunarmerkjum. Hrópið svo nafnið ykkar :)
3. Til að athuga hvort stafur `c` sé hástafur má spyrja: `c == c.upper()`. Skrifið fall `stórir(s)` sem skilar lista af rökgildum með i-ta gildið satt ef i-ta orðið í `s` byrjar á stórum staf. Prófið með sjálfvöldu dæmi og sýnið niðurstöðuna.

[5]:
```
def telja(s):
  d=[]
  w= s.split()
  for i in w:
    d.append(len(i))
  return d

def hropa(s):
  return s.upper()+"!!"


def storir(s):
  p= []
  w=s.split()
  for i in w:
    p.append(i[0].isupper())
  return p
```

```
print(telja("Afi minn og amma mín"))
print(hropa("Magnús Daníel Budai Einarsson"))
print(storir("pRufa Þetta Eru sTorir Stafir"))
```

```
[3, 4, 2, 4, 3]
MAGNÚS DANÍEL BUDAI EINARSSON!!
[False, True, True, False, True]
```

5 Kvaðratrót

Um útreikning kvaðratrótar er fjallað í kafla 7.5 í Think Python kennslubókinni. Þar stendur að hægt sé að nota Newtons aðferð, og að ef byrjað er með einhverja nálgun x við $\sqrt{a}$ þá fáist betri nálgun, y með því að reikna:

$$(*) \qquad y = \frac{x + a/x}{2}$$

Formúluna má m.a. rökstyðja þannig að ef x er nákvæmt gildi þá er $x^2 = a$ svo að $x = \frac{a}{x}$. Hinsvegar ef x er aðeins minna en kvaðratrótin þá þá verður $\frac{a}{x}$ aðeins stærra en hún (og öfugt) og því ætti meðaltal x og $\frac{a}{x}$ að vera betri nálgun. Svo má athuga hve nálgunin er góð með því að skoða muninn á x og y. Ef t.d.

$$(**) \qquad |x - y| < \varepsilon$$

þar sem $\varepsilon = 10^{-4}$ er hún orðin nokkuð góð. 1. Skrifið Python-fall, krót(a) sem útfærir þessa hugmynd. Byrjið með upphafsgildið $x = 1$ og finnið svo betri og betri lausn með því að nota while-lykkju sem reiknar $(*)$ aftur og aftur og heldur áfram þangað til $(**)$ er uppfyllt. Prófið að reikna $\sqrt{9}$ og $\sqrt{10}$ (rétt gildi 3.16227766017).
2. Búið nú til nýja útgáfu af fallinu sem er með tvo inntaksstika, töluna a og nákvæmnikröfu eps, og telur auk þess hve margar ítrekanir eru teknar. Látið það skila bæði lokanálguninni og ítrekanafjöldanum (sbr. fyrra sýnidæmið í kafla 4.7). Skrifið niðurstöður með hæfilegum skýringartextum: lokanálgun, ítrekanafjölda, og muninn á réttri kvaðratrót og lokanálgun. Prófið með nokkrum mismunandi gildum á (m.a. eitthvað mjög stórt gildi) og , (m.a. gildi sem er ekkert mjög lítið, t.d. 0.1 eða 0.01). Bætið við textareit og skrifið örfá orð um niðurstöðu þessarar prófunar.

[6]:
```
def krot(a):
    x=1
    eps = 1e-4
    y=2
    while abs(x - y) >= eps:
        x=y
        y = (x + a/x)/2
    return y

def krot2(a, eps):
    x=1
    y=2
    ite = 0
```

```
    while abs(x - y) >= eps:
        x=y
        y = (x + a/x)/2
        ite += 1
    return y, ite
```

6 Meðaltal og staðalfrávik

Skrifið fall `tölfræði(x)` sem skilar pari `(m, s)` með meðaltali og staðalfráviki stakanna í listanum `x` með því að nota formúlurnar:

$$m = \frac{1}{n}\sum_{i=0}^{n-1} x[i]$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=0}^{n-1}(x[i]-m)^2}$$

Prófið með `x = [3,7,7,1]` (á að skila `[4.5, 3.0]`) og líka með lista `[d,m,á]` búnum til úr fæðingardegi.

[15]:
```
def tolfraedi(x):
    n = len(x)
    sums = 0
    for i in range(n):
        sums += x[i]
    m = sums/n
    sums = 0
    for i in range(n):
        sums += (x[i] - m)**2
    s = math.sqrt(sums/(n-1))
    return [m,s]

x=[3,7,7,1]
print(tolfraedi(x))

birthday = [24,11,93]
print(tolfraedi(birthday))
```

```
[4.5, 3.0]
[42.666666666666664, 44.07191093353377]
```

7 Bóluröðun

Hér er reiknirit sem raðar n-staka lista x með bóluröðun bubble sort, sem snýst um að rúlla í gegn um stökin og ef tvö samliggjandi stök eru í öfugri röð þá er víxlað á þeim. Þetta er endurtekið þar til öll stökin eru í röð. Minnstu stökin bobbla smám saman eins og loftbólur fremst í listann.

`víxlað = satt`

```
meðan víxlað
   víxlað = ósatt
   fyrir i=1,...,n-1:
      ef x[i-1] > x[i] þá
         víxla á x[i-1] og x[i]
         víxlað = satt
```

Þýðið þetta reiknirit yfir í Python-fall `bóluröðun(x)`. Athugið að til að víxla á breytum `x` og `y` má nota `(x,y) = (y,x)`. Prófið með því að raða listanum `[3,8,1,2,5,4]`.

[16]:
```
def bolurodun(x):
    reversed = True
    n = len(x)
    while (reversed):
        reversed = False
        for i in range (1,n):
            if x[i-1]> x[i]:
                x[i-1], x[i] = x[i], x[i-1]
                reversed = True
    return x

x = [3,8,1,2,5,4]

print(f"Ef við notum bóluröðun á eftirfarandi lista: {x} þá fáum við útkommuna␣
↪{bolurodun(x)}")
```

```
Ef við notum bóluröðun á eftirfarandi lista: [3, 8, 1, 2, 5, 4] þá fáum við
útkommuna [1, 2, 3, 4, 5, 8]
```

8 Pólhnit

Lesið aðeins um pólhnit á Wikipediu áður en þið spreytið ykkur á eftirfarandi verkefni.

1. Skrifið fall `pol2rec(r,theta)` sem skilar pari `(x,y)`, til að breyta úr pólhnitum yfir í rétthyrnd hnit með eftirfarandi formúlum:

$$x = r\cos\theta$$
$$y = r\sin\theta$$

   Prófið að breyta $(r, \theta) = (4, 35°)$ yfir í rétthyrnd hnit x og y (á að gefa $x = 3.277, y = 2.294$). Munið að nota `math.radians` til að breyta gráðum í radíana.

2. Skrifið svo annað fall, `rec2pol(x,y)` sem skilar `(r,theta)` og breytir í hina áttina með formúlunum:

$$\theta = \text{atan2}(y, x)$$
$$r = \sqrt{x^2 + y^2}$$

   Prófið að breyta svarinu sem fékkst í lið 1 aftur til baka í pólhnit.

[19]:
```
def pol2rec(r,theta):
    return r * math.cos(math.radians(theta)), r * math.sin(math.radians(theta))

print(pol2rec(4,35))

def rec2pol(x,y):
    return math.ceil(math.sqrt(x**2+y**2)), math.degrees(math.atan2(y,x))

x,y= pol2rec(4,35)
print(rec2pol(x,y))
```

```
(3.276608177155967, 2.294305745404184)
(4, 35.0)
```

9 Innsetningarröðun og Pythontutor

1. Ein einfaldasta röðunaraðferðin er innsetningarröðun (insertion sort). Forritið fyrsta reikniritið í lýsingu hennar á Wikipediu í Python falli. Notið paragildisgjöf, `(x,y) = (y,x)` til að víxla (swap) í 5. línu reikniritsins. Prófið að raða listanum $[3, 8, 1, 2, 5, 4]$. Leysið verkefnið með falli sem hefur listann sem raða skal sem inntaksstika og skilar röðuðum lista. Gætið þess að breyta ekki inntaksstikanum inni í fallinu.
2. Á vefsíðan pythontutor.com er hægt að keyra forrit skref fyrir skref og skoða hvernig gildi á breytum breytast. Setjið forritið sem þið skrifuðuð í lið 1 inn á þessa síðu og prófið ykkur áfram.

[20]:
```
def insertsort(A):
  i = 1
  while i < len(A):
    j = i
    while j > 0 and A[j-1] > A[j]:
        A[j-1],A[j] = A[j],A[j-1]
        j -= 1
    i += 1
  return A

print(insertsort([3,8,1,2,5,4]))
```

```
[1, 2, 3, 4, 5, 8]
```

10 Töluleg diffrun

1. Eftirfarandi formúlu má nota til að nálga afleiðu falls í punkti a:

$$f'(a) \approx \frac{f(a+h) - f(a-h)}{2h}$$

þar sem h er lítil tala. Skrifið fall `töldiff(f, a, h)` sem reiknar slíka nálgun og skilar henni. Prófið með $f(x) = \sqrt{x}, a = 4$ og $h = 10^{-4}$ (sem ætti að skila tölu sem er nálægt $\frac{1}{2\sqrt{4}}$).

2. Skrifið nú töflu yfir nálgunina fyrir $h = 10^{-k}, k = 1, ...10$, og látið fylgja með dálk sem sýnir hve skekkjan er mikil. Merkið með $*$ línuna þar sem skekkjan er minnst.

[21]:
```
def toldiff(f,a,h):
    return (f(a+h)-f(a-h))/(2 * h)

def f(x):
    return math.sqrt(x)

print(f"Svarið við lið a er {toldiff(f,4,10**-4)}\n")
k=1
k2=[]
while k<=10:
    k2.append(toldiff(f,4,10**-k))
    k+=1
minimum = min(k2, key=lambda x:abs(x-0.25))
print("k          útreikningur                mismunur")
print("-"*50)
for k,i in enumerate(k2):
    if i==minimum:
        stringI = str(i)
        stringI += "*"
        print(f"{k+1:<3}{stringI:^30}{i-0.25:>10}")
    else:
        print(f"{k+1:<3}{i:^30}{abs(0.25-i):>10}")
```

```
Svarið við lið a er 0.2500000000205116

k          útreikningur                mismunur
--------------------------------------------------
1       0.25001953659254283      1.9536592542834796e-05
2        0.2500001953130382      1.953130381870949e-07
3       0.25000000195318783      1.953187833692027e-09
4        0.2500000000205116      2.0511592424554692e-11
5       0.2500000000016378*      1.637801005927031e-12
6        0.2499999999239222      7.607781071783393e-11
7       0.24999999959085528      4.091447181053809e-10
8        0.249999987378402       1.2621597988982103e-08
9       0.25000002068509275      2.0685092749772593e-08
10      0.25000002068509275      2.0685092749772593e-08
```

11 Uppflettitafla afturábak

Hugsum okkur að ísl_ens sé uppflettitafla sem geymir íslensk-enska orðabók. Hún gæti t.d. innihaldið pörin:

”reipi” → ”rope”
”hús → ”house” og
”rauður” → ”red”.

Ef við vilum búa til ensk-íslenska orðabók gætum við snúið töflunni við og fengið pörin ”rope” → ”reipi”, ”house” → ”hús” o.s.frv. 1. Skrifið fall snúavið(U) sem snýr uppflettitöflu við á þennan hátt og skilar viðsnúnu töflunni. Gerið ráð fyrir að taflan U hafi hvergi sama gildi fyrir tvo mismunandi lykla. Prófið með orðaskránni að framan og búið til ens_ísl. 2. Skrifið nýja útgáfu af fallinu, snúavið2(U), sem er ekki með slíkri einkvæmnitakmörkun. Það á að skila nýrri uppflettitöflu V sem er þannig að ef g er gildi svarandi til tveggja mismunandi lykla, U[x] = U[y] = g þá á gildi V fyrir lykilinn g að vera listi með x og y, V[g] = [x,y]. Bætið nú tveimur pörum við ísl_ens:

”tómarúm” → ”vaccum”
”ryksuga” *to* ”vaccum”

og prófið snúavið2.

[22]:
```
def snuavid(U):
    return {value: keyname for keyname, value in U.items()}

ordabok = {"reipi":"rope", "hús":"house", "rauður":"red"}

print(snuavid(ordabok))

def snuavid2(U):
    L={}
    for keyname, value in U.items():
        if value in L:
            L[value].append(keyname)
        else:
            L[value] = [keyname]
    return L

ordabok = {"reipi":"rope", "hús":"house", "rauður":"red", "tómarúm":"vaccum",␣
 ↪"ryksuga":"vaccum"}

print(snuavid2(ordabok))
```

```
{'rope': 'reipi', 'house': 'hús', 'red': 'rauður'}
{'rope': ['reipi'], 'house': ['hús'], 'red': ['rauður'], 'vaccum': ['tómarúm',
'ryksuga']}
```

12 Shell-röðun

Árið 1959 kom út grein eftir Donald L Shell með hraðvirku reikniriti til að raða tölum (eða öðrum hlutum) sem síðar var kallað Shell-sort. Það er áhugavert að skoða greinina, sérstaklega hvernig reikniritið er sett fram með svonefndu flæðiriti, en í byrjun tölvualdar voru þau algeng. Reikniritið vinnur með minnkandi runu af bilum (gaps), $b_1 > b_2 > ... > b_n = 1$. Í fyrstu umferð er sætum $0, b_1, 2b_1, ...$ raðað með innsetningarröðun (sjá verkefni 9), þvínæst sætum $1, 1+b_1, 1+2b_1, ...$, o.s.frv. þar til allar hlutrunur sæta með millibili b_1 eru komnar í röð. Þetta er svo endurtekið fyrir öll bilin. Í upphaflegu grein Shells var b_k valið sem $\lfloor N/2^k \rfloor$ en síðan hafa menn þróað ýmsar aðrar bilarunur ($\lfloor x \rfloor = \text{int}(x)$; þ.e. sker aukastafi af x).

Hér er reiknirit sem Shell-raðar lista $a[0], a[1]...a[N-1]$ með upphaflegu bilarununni:

```
fyrir k = 1,2,3,...:
   bil = int(N/2^k)
   ef bil < 1 þá út úr lykkju
   fyrir i = 0, 1,..., bil-1:
       raða a[i], a[i+bil], a[i+2*bil],... með innsetningarröðun
```

Skrifið Python-fall sem Shell-raðar. Byrjið á að leysa verkefni 9 ef þið eruð ekki þegar búin að því. Það er hægt að nota heiltöludeilingu (`//`) til að reikna `bil` og hlutrunan í neðstu línunni fæst með `a[i:N:bil]`. Byggið á reikniritinu að framan (og alls ekki ná í tilbúið forrit af netinu). Prófið að raða listunum `[8,3,2]` og `[8,5,1,9,6,2,1,7,11,3]`, og auk þess einum sjálfvöldum lista með 12 tveggja stafa tölum.

[23]:
```
def insertsort(A):
  i = 1
  while i < len(A):
    j = i
    while j > 0 and A[j-1] > A[j]:
        A[j-1],A[j] = A[j],A[j-1]
        j -= 1
    i += 1
  return A


def shellsort(a):
  k = 1
  N = len(a)
  while(True):
    bil = N//2**k
    k+= 1
    if bil < 1:
      break
```

```
    for i in range(bil):
      b = insertsort(a[i:N:bil])
  return b

print(shellsort([8,3,2]))
print(shellsort([8,5,1,9,6,2,1,7,11,3]))
print(shellsort([67, 92, 18, 44, 67, 71, 77, 58, 29, 40, 11, 13]))
```

```
[2, 3, 8]
[1, 1, 2, 3, 5, 6, 7, 8, 9, 11]
[11, 13, 18, 29, 40, 44, 58, 67, 67, 71, 77, 92]
```

13 Skrá með íslenskum orðum

Þetta verkefni er lauslega byggt á verkefnum í 9. kafla Think Python kennslubókarinnar. Í skrá með veffang https://cs.hi.is/python/ord.txt eru 217 þúsund íslensk orð. Þið getið skoðað skrána með því að smella á hlekkinn. Í verkefninu sem hér fylgir þarf að nota ýmsar strengjaaðgerðir sem lýst er í kafla 5.2. 1. Opnið skrána með `urlopen` og lesið hana inn eins og sýnt er í kafla 8.2.4. Prentið út fyrstu 5 orðin og líka tíu þúsundasta hvert orð og öll orð sem eru lengri en 30 stafir. 2. Spegilorð (palindrome) er orð sem er eins lesið afturábak og áfram (t.d. kajak). Skrifið rökfall `spegilorð(s)` sem kannar hvort s sé spegilorð [rökfall er fall sem skilar `True` eða `False` og prófið]. Skrifið í framhaldi út öll spegilorð í skránni, 10 á hverja línu. Fyrsta linan ætti að verða:

`abba, afa, aga, agga, aka, ala, alla, ama, amma, ana,`

3. Finnið þau orð í skránni sem hafa einn sérhljóða og hámarksfjölda samhljóða.

[25]:
```
#1
f = urlopen("https://cs.hi.is/python/ord.txt")
ordList = []

for line in f:
   ordList.append(line.decode().strip())

fimmOrd = ""
for i in range(5):
    fimmOrd += ordList[i] + " "

print(fimmOrd)

for k,i in enumerate(ordList, 1):
    if k % 10000 == 0 or len(i) > 30:
        print(f'{k:6}', i)

#2
def palindrome(s):
    return (s==s[::-1])
```

```
count = 0

for i in ordList:
    if palindrome(i):
        if count == 10:
            print()
            count = 0
        else:
            print(i, end=", ")
            count +=1

#3
def longest(s):
    vowels = ['a', 'á', 'e', 'é', 'i', 'í', 'o', 'ó', 'u', 'ú', 'y', 'ý', 'æ',␣
 ↪'ö']
    longword = []
    longest_word_length = 0
    longest_words = []
    for i in s:
        word = i.lower()
        count = sum(1 for char in word if char in vowels)
        if count == 1:
            longword.append(word)
            word_length = len(word)
            if word_length > longest_word_length:
                longest_word_length = word_length
    for word in longword:
        if len(word) == longest_word_length:
            longest_words.append(word)
    print(longest_words)
longest(ordList)
```

```
abba abbadís abbadísar abbadísarinnar abbadísartíð
  3512 alþjóðaheilbrigðisstofnunarinnar
  3574 alþjóðasiglingamálastofnunarinnar
  5822 atvinnuleysistryggingasjóðurinn
 10000 barónessunni
 20000 bókmenntaheimurinn
 30000 eldvarpa
 39136 flugslysarannsóknarnefndarinnar
 40000 flögrað
 50000 galdrakerlingin
 60000 hafnarverkamannsins
 70000 hnýtta
 80000 illkvittnislega
 90000 konunglegan
100000 leiðbeiningu
110000 margnefndi
```

120000 nemann
121142 norðuratlantshafssjávarspendýraráðsins
121175 norðurheimskautsrannsóknaráðsins
130000 ramman
140000 sandhólum
150000 skynsemd
160000 stjórnarþátttöku
170000 sólarhofsins
172917 teiknimyndaævintýrapoppálfkonan
180000 tötralegum
190000 veðurratsjá
200000 árásargjörn
210000 útdauðar
abba, afa, aga, agga, aka, ala, alla, ama, amma, ana,
argra, assa, ata, axa, aða, gíg, gýg, illi, inni, iðaði,
kajak, kok, kák, kæk, kók, kúk, mm, muninum, munnum, munum,
mussum, natan, nón, píp, rabbar, radar, raddar, rafar, ragar, rakar,
rammar, rappar, rasar, rassar, ratar, raðar, rifir, riðir, ruddur, rullur,
runur, rár, rær, rór, rör, rýr, sinnis, stúts, summus, sás,
tillit, tæt, uku, ullu, undnu, unnu, unu, uxu, á, æ,
ísí, ó, óbó, ódó, óró, ý, ['bhmfólks', 'skrappst', 'skyggnst', 'strengst']