

S2

February 19, 2023

Heimadæmi S2

1 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 parabolum og heildið nálgað með flatarmálinu undir þeim. Skoðið endilega Wikipedíugrein um aðferðina.

Simpsons-formúlan er eftirfarandi:

$$\int_a^b f(x) dx \approx \frac{\Delta x}{3} (f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + \dots + 4f(x_{n-1}) + f(x_n))$$

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.

```
[18]: from math import sin
      from math import cos

      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

from math import sin, e

def g(x): return sin(x)/x
def h(x): return e**x
def z(x): return sin(2*x)*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 hlutblum 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 hlutblum 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

```

[225]: n=1
while round(simpson(g,1,2,n),7)!=0.6593299:
    n+=1
print(f"Það tekur {n} trapisur á fyrri jöfnu til þess að fá rétt gildi")

m=1
while round(simpson(h,0,1,m),7)!=1.7182818:
    m+=1
print(f"Það tekur {m} trapisur á seinni jöfnu til þess að fá rétt gildi")

```

Það tekur 10 trapisur á fyrri jöfnu til þess að fá rétt gildi
Það tekur 26 trapisur á seinni jöfnu til þess að fá rétt gildi

2 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 `boluroðun(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]`.

```
[227]: 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]`

3 Vaxtareikningur

3.0.1 1.

Skrifið fall með stika `u`, `p`, `k`, `m` sem reiknar heildarvexti, , af upphæð sem er á % vöxtum í ár og mánuði sky. formúlunni:

$$v = u(1 + a)^k(1 + \frac{am}{12}) - u, arsem a = \frac{p}{100}$$

Skerið af `aura` (með fallinu `int`), hafið viðeigandi skjölunarstreng í fallinu, og prófið það með því að reikna 2% vexti af 10000 kr. í 3 ár og 4 mánuði (ætti að gefa 682 kr.).

```
[2]: def interest(upphaed, prosent, ar, manudir):
    a = prosent/100
    return (upphaed*(1+a)**ar) * (1+(a*manudir)/12)-upphaed

print(f"Ef vextir eru 2% á 10.000kr sem ég legg inn þá eru heildarvextir á_
↪þeirri upphæð {int(interest(10000, 2, 3, 4))}kr. eftir 3. ár og 4 mánuði")
```

Ef vextir eru 2% á 10.000kr sem ég legg inn þá eru heildarvextir á þeirri upphæð 682kr. eftir 3. ár og 4 mánuði

3.0.2 2.

Reiknið heildarvexti til dagsins í dag ef 25000 kr. hefðu verið lagðar inn á 3% vexti á fæðingardegi ykkar (nálgði aldur ykkar í heilan mánuð), og reiknið jafnframt út hlutfall vaxtanna af upphaflegri upphæð. Notið `f`-strengi til að skrifa niðurstöðurnar með hæfilegum skýringartexta.

```
[14]: # Age: 29 year, 2 months

print(f"Ég er 29 ára og tveggja mánaða gamall og fæ því {int(interest(25000, 3, 29, 2))}kr. vexti ef ég hefði lagt inn 25.000 kr. við fæðingu.")
print(f"Vaxtarhlufall frá upphafi er: {(interest(25000, 3, 29, 2)/25000)*100:.2f}%")
```

Ég er 29 ára og tveggja mánaða gamall og fæ því 34208kr. vexti ef ég hefði lagt inn 25.000 kr. við fæðingu.
Vaxtarhlufall frá upphafi er: 136.83%

3.0.3 3.

Skrifið fall sem ákvarðar hve mörg ár og mánuði það tekur upphæð á % vöxtum að tvöfaldast (notið t.d. tvöfalda lykkju, og return á viðeigandi stað). Prófið með $n = 13$ (ætti að gefa 5 ár og 8 mánuði) og með gefnu með fæðingarmánuði ykkar (t.d. 8 fyrir ágúst).

```
[16]: def interestTime(upp, n):
    ar = 0
    v=0
    while (v<=upp):
        manudir = -1
        ar +=1
        while (v<=upp and manudir<12):
            manudir +=1
            v = interest(upp, n, ar, manudir)

    return ar, manudir

#Fæddur í Nóvember
print(f"Ef vextir eru 11% þá tekur það {interestTime(1000, 11)[0]} ár og {interestTime(1000, 11)[1]} mánuði að tvöfalda upphæðina.")
```

Ef vextir eru 11% þá tekur það 6 ár og 8 mánuði að tvöfalda upphæðina.