Extraövningar i PYTHON

1. Skriv ett uttryck som är sant om ett tal R inte avviker mer än 0,05 från 12.

2. Låt `a, b, c` och `k` vara heltalsvariabler (`int`). För vilka värden på `a, b, c` och `k` är följande villkor i PYTHON sanna respektive falska?

   (a) `b<a and a<c`

   (b) `b<a or a<c`

   (c) `k==1 or k==6`

   (d) `k==1 and k==6`

   (e) `not (k==1 or k==6)`

3. Skriv uttryck som testar om

   (a) x inte tillhör intervallet $[-1, 5]$

   (b) $-\pi \leq x \leq \pi$

   (c) $x \in [0, 1]$ eller $y \in [-4, 4]$

   (d) punkten med koordinaterna (x, y) ligger i en cirkel med mittpunkt i origo $(0, 0)$ och radie 1.

4. För vilka tal `x` och `y` är uttrycken sanna? Svara med en geometrisk beskrivning (t.ex. ” rektangeln med hörn i ..”), eller rita en bild på papper.

```
abs(x) <= 1 and abs(y) <=2

abs(x) <= 1 or abs(y) <=2

not (abs(x) <= 1) and abs(y) <=2

not abs(x) <= 1 and abs(y) <= 2

not (abs(x) <= 1) and not (abs(y)<=2)

abs(x) <= y**2

x*y <= 1
```

5. Skriv ett program i PYTHON som omvandlar från svenska kronor till japanska yen. Läs in antalet svenska kronor med `input` och använd `print` för att skriva ut svaret. Antag att en svensk krona är 12,75 yen.

6. Sekvensen nedan ritar en röd ifylld oktogon. Använd `text` och placera ut ordet `STOP` mitt i figuren.

```
import numpy as np
import matplotlib.pyplot as plt
t=np.linspace(0,2*np.pi,9)+np.pi/8
x=np.cos(t); y=np.sin(t)
plt.fill(x,y,'red')
plt.axis('equal')
```

7. En 12:a är en segelbåt vars mått uppfyller formeln

$$R = \frac{L + 2d - F + \sqrt{(S)}}{2.37}$$

där L är längden i vattenlinjen, d är omfångsdifferensen midskepps, S är segelarean och F är fribordshöjden[1]. Skriv ett program i PYTHON som läser in värden på L, d, S och F från Console och som avgör om värdet på R avviker mer än 0.05 från 12.

8. Skriv ett program som upprepade gånger läser in ett värde på en radie r och som svarar med volymen för ett klot, $V = 4/3\pi r^3$. Om användaren anger en negativ radie ska programmet avslutas.

9. Skriv ett program som låter användaren mata in ett antal positiva tal, ett och ett, och som skriver ut medelvärdet av alla de inlästa talen. Inläsningen avslutas med att användaren matar in ett negativt tal. Det negativa talet ska inte tas med i medelvärdet.

10. Skriv ett program som beräknar summan $1 + 1/2 + 1/3 + \cdots + 1/100$. Använd en `for`-loop för att beräkna summan.

11. I följande loopar försöker man beräkna hur många termer man minst måste ta med för att $1 + 1/2 + 1/3 + \cdots \geq 5$. Ingen av looparna ger rätt svar. Förklara vad som är fel.

```
n = 0; sn = 0
while sn < 5:
  sn = sn+1/n
  n = n+1
print(n)
```

```
n = 1; sn = 1
while sn < 5:
  sn = sn+1/n
  n = n+1
print(n)
```

```
n = 1; sn = 0
while sn < 5:
  sn = sn+1/n
  n = n+1
print(n)
```

12. Det gäller att $\ln(2) = 1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \ldots$. Hur många termer i summan måste man ta med om man vill approximera $\ln(2)$ med 5 korrekta decimaler?

    Ledning: Använd en `while`-sats och addera term för term till en variabel `s`. Avsluta när $|s - \ln(2)| < 0.5 \cdot 10^{-5}$. Funktionen för `ln` heter `log` i `NumPy`.

13. Beräkna medelvärdet av elementen i ett fält. Använd `for`-loop

14. Skriv en `for`-loop som skapar ett fält som innehåller talen 1, 1, 2, 3, 5, 8, ... Från plats 3 och framåt är elementen summan av de två föregående elementen i fältet. Låt fältet bestå av 50 element.

15. En $n \times n$ matris $\mathbf{A}$ vars element ges av

$$a_{ij} = \frac{1}{i + j - 1}$$

kallas Hilbertmatris. Man kan skapa en Hilbertmatris med dubbla `for`-loopar som i sekvensen nedan. Om sekvensen körs med $n = 3$ får man matrisen till höger

[1] En beskrivning av hur man klassificerar 12:or finns på websidan `http:/www.12mrclass.com/about-us/class-rules/`

```
import numpy as np
n=3
A=np.zeros(n,n)
for i in range(n):
  for j in range(n):
    A[i,j]=1/(i+j+1)
```

$$\begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix}$$

I vilken ordning tilldelas elementen i matrisen värden?

16. Använd två loopar för att generera en utskrift i stil med

```
1 1
1 10
1 1000
1 10000
2 1
2 10
2 100
2 1000
2 10000
3 1
3 10

osv.

9 1
9 10
9 100
9 1000
9 10000
```

17. Skriv en funktion som beräknar volymen för ett klot. Låt funktionen ha en inparameter, radien till klotet.

    Använd funktionen och beräkna volymer för 9 klot med radier 2440, 6052, 6378, 3396, 69911, 58232, 25362, 24622 respektive 1188 km.

18. Skriv en funktion som avgör om en segelbåt är en tolva. Låt funktionen returnera sant om båten är en 12:a, falskt annars. (I uppgift 6 finns formeln som klassificerar om en båt är en 12:a).

19. Skriv en funktion `max2(x,y)` som tar två tal `x` och `y` som inparameterar och som returnerar det största värdet.

20. Skriv en funktion som heter `fml`. Funktionen ska en inparameter, en vektor med tal `x`. Funktionen ska returnera tre tal, `first`, `middle` och `last`, där `first` och `last` är första respektive sista elementet i vektorn. Om `x` innehåller ett udda antal element ska `middle` vara mittenelementet i vektorn. Om inparametern `x` innehåller ett jämnt antal tal ska `middle` vara medelvärdet av de två mittersta elementen i `x`.

21. Skriv en funktion som heter `vec_sums` som har två inparametrar, två vektorer `x` och `y`. Funktionen ska returnera två vektorer, `xpr2 = x+rev(y)` och `xmr2 = x-rev(y)`, där `rev(y)` betecknar den vektor man får om man reverserar elementen i `y`, så t.ex. `rev([1,4,3,2])` blir `[2,3,4,1]`. Om inparametrarna inte innehåller lika många element ska funktionen returnera en tom vektor.

22. Låt

$$f(x,y) = \sin(x) \cdot e^x + \sin(y) \cdot e^y$$

Skriv en funktion i PYTHON som heter `f` som beskriver funktionen f. Skriv sedan ett program som skapar en grid (ett rutnät) med gridpunkter (x_i, y_i) jämt fördelade över området $-2 \leq x \leq 2$, $-2 \leq y \leq 2$ och sedan beräknar funktionsvärden i alla gridpunkter. Griden skapas genom att bilda två matriser

$$X = \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix} \text{ och } Y = \begin{bmatrix} -2 & -2 & -2 & -2 & -2 \\ -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \end{bmatrix}$$

Sekvensen nedan ritar en funktionsyta[2] av $f(x,y)$

```
import matplotlib.pyplot as plt
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X,Y,f(X,Y))
```

Testa gärna den (fast du måste först definera funktionen `f` och matriserna `X` och `Y`. Testa också att göra griden tätare.

23. Skriv en funktion som heter `positive`. Funktionen ska ha en parameter, en matris `M` och ska returnera en logisk matris `P`. Elementet `P[i][k]` ska ha värdet `True` då motsvarande element i `M` är positivt. För övriga värden ska `P[i][k]` ha värdet `False`

[2]En funktionsyta är mängden av alla punkter (x,y,z) i rummet sådana att $f(x,y) = z$

Lösningsförslag till de flesta av övningarna

1. `abs(R-12)<0.05`

2. (a) Sant om a ligger mellan b och c, dvs sant om $b < a < c$. Villkoret kan också skrivas `b<a<c` i PYTHON
   (b) Vilkoret är falskt då $c < a < b$, sant annars.
   (c) Sant om `k` har något av värdna 1 eller 6.
   (d) Alltid falskt
   (e) Sant om $k \neq 1$ och $k \neq 6$

3. (a) `not -1<x<5`
   (b) `import numpy as np; -np.pi <= x <= np.pi`
   (c) `0 <= x <= 1 or -4 <= x <= 4`
   (d) `x**2+y**2<1`

4.

5. ```
   kr = float(input('Ange kronantal '))
   print('Det blir ',kr*12.75,' yen')
   ```

6. Utöka t.ex. med `plt.text(-0.5,-0.1,'STOP',fontsize=40,color='white')`

7. ```
   import numpy as np
   L=float(input('Längden i vattenlinjen: '))
   d=float(input('omfångsdifferensen midskepps: '))
   S=float(input('Segelarean: '))
   F=float(input('Fribordshöjden: '))
   R=(L+2*d-F+np.sqrt(S))/2.37
   if abs(R-12)<0.05:
       print('ja en 12:a ')
   else:
       print('nix inte en 12:a')
   ```

8. ```
   import numpy as np
   while 1:
       r=float(input('Ange en radie '))
       if r<0: break
       print('V =',4/3*np.pi*r**3)
   ```

9. ```
   m=0
   n=0
   while 1:
       t=float(input('Ange tal: '))
       if t<0: break
       m=m+t
       n=n+1

   if n>0:
      print('Medelvärdet = ',m/n)
   ```

10.
```
s=0
for n in range(100):
    s=s+1/(n+1)
```

11. Vänster: `n` har värdet 0 första varvet i loopen, så man delar med 0.
Mitten: `sn` har fel startvärde, så summan 1+1+1/2+1/3 .. beräknas.
Höger: `n` blir ett för stort. Rätt antal termer ska vara `n-1`

12. Addera term för term till `s`. Avsluta när $|s - \ln(2)| < 0.5 \cdot 10^{-5}$. Obs funktionen för `ln` heter `log` i `NumPy`

```
import numpy as np
tol=0.5e-5
s=0; n=0; d=1;
while np.abs(s-np.log(2))>tol:
    n=n+1
    s=s+(-1)**(n-1)/n
print('Antal termer ',n)
```

13.

14.
```
import numpy as np
f=np.zeros(50)
f[0]=1; f[1]=1
for i in range(2,np.size(f)):
    f[i]=f[i-1]+f[i-2]
```

15. Elementen får sina värden radvis. För varje varv i den yttre loopen körs den inre.

16.
```
for i in range(9):
    for j in [1,10,100,1000,10000]:
        print(i+1,j)
```

17.
```
import numpy as np
def volymer(r):
    return 4/3*r**3*np.pi

r=np.array([2440,6052,6378,3396,69911,58232,25362,24622,1188])
print(volymer(r))
```

18.
```
import numpy as np
def tolva(L,d,S,F):
  R = (L+2*d-F+np.sqrt(S))/2.37
  return abs(R-12)<0.05
```

19.
```
def max2(x,y):
    if x>y:
        return x
    else:
        return y
```

Om man ska beräkna det största av två tal i PYTHON är det förstås bättre att använda funktionen `max`. Tanken med övningen är att öva på att definera egna funktioner.

20.
```
def fml(x):
    n=len(x)
    if n%2==0:  # jämnt
      middle=(x[n//2-1]+x[n//2])/2
    else:
      middle=x[(n-1)//2]
    first=x[0]
    last=x[-1]   # sista elementet i en vektor indexeras -1
    return [first,middle,last]
```

Operatorn `//` är heltalsdivision. (Operatorn `/` är vanlig division).

21.
```
def vec_sums(x,y):
    n=len(y)
    if n!=len(x): return []
    revy=np.array(y.copy())
    for i in range(n):
        revy[n-i-1]=y[i]
    xpr2=x+revy
    xmr2=x-revy
    return [xpr2,xmr2]
```

22.
```
import matplotlib.pyplot as plt
import numpy as np
def f(x,y):
    return np.sin(x)*np.exp(x)+np.sin(y)*np.exp(y)

X=np.array([[-2,-1,0,1,2],[-2,-1,0,1,2],[-2,-1,0,1,2],[-2,-1,0,1,2],[-2,-1,0,1,2]])
Y=np.array([[-2,-2,-2,-2,-2],[-1,-1,-1,-1,-1],[0,0,0,0,0],[1,1,1,1,1],[2,2,2,2,2]])
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X,Y,f(X,Y))
```

23.
```
def positive(M):
    (m,n)=np.shape(M)
    P=np.ones((m,n))*False
    for i in range(m):
        for j in range(n):
            if M[i,j]>0:
                P[i,j]=True
    return P
```