Programmeringsövningshäfte

Övningarna i det här häftet är till stor del av matematisk karaktär, men är sorterade efter olika programmeringskoncept. Svårighetsgraderna markeras med asterisk (*) där ju fler asterisker desto högre är svårighetsnivån.

Innehåll

Räkna med Python

1. En rätvinklig triangel har en katet som är 3cm, och en hypotenusa som är 5cm. Skriv ett program som beräknar den andra katetens längd. (*)

---

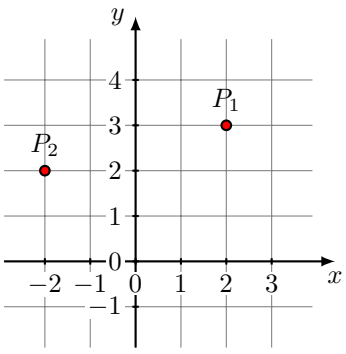
2. Skriv ett program som låter användaren mata in basen och höjden av en triangel. Programmet ska räkna ut triangelns area. (*)

---

3. pH-värdet är ett logaritmiskt mått på surheten och definierat som: $pH = -\lg\{H^+\}$, där $\{H^+\}$ är aktiviteten av vätejoner H^+. Skriv ett program som beräknar pH-värdet för vätejonsaktiviteten $1 \cdot 10^{-7}$.

---

4. Avståndsformeln (x_1, x_2) $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$, är en tillämpning av Pythagoras sats och kan användas för att räkna avståndet mellan två punkter (x_1, y_1) och (x_2, y_2) i ett koordinatsystem. Skriv ett program som:



   (a) beräknar avståndet mellan P_1 och P_2 (*)

   (b) beräknar avståndet d mellan punkterna $(2, 1)$ och $(1, 0)$ (*)

(c) låter användaren mata in punkterna och beräknar avståndet mellan dem. (*)

If-satser

1. Skriv ett program som låter användaren mata in ett tal och skriv ut om talet är positivt negativt eller noll. (*)

---

2. Skriv ett program som låter användaren mata in ett tal och skriv ut om talet är jämnt, udda eller delbart med 5. (*)

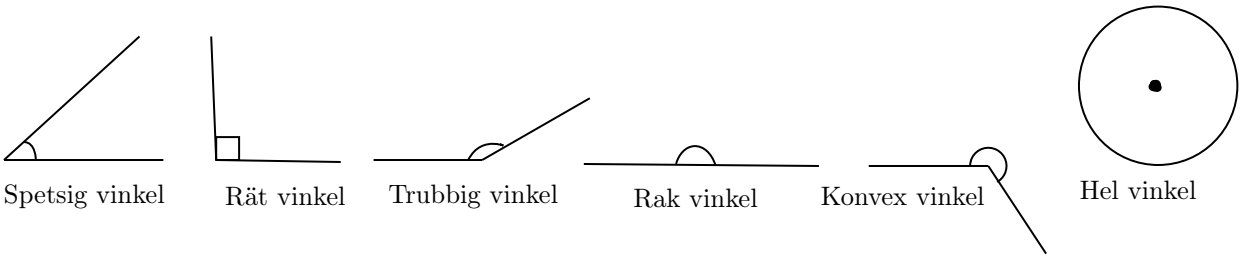
---

3. Skriv ett program som låter användaren mata in två tal och skriv därefter ut det största talet. (*)

---

4. Skriv ett program som låter användaren mata in en vinkel och skriv ut om vinkeln är:

   - spetsig
   - rät
   - trubbig
   - rak
   - konvex
   - hel



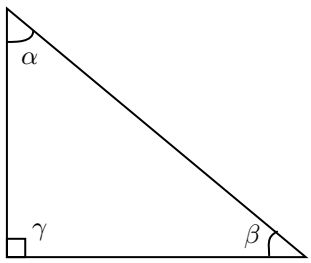
(*)

---

5. Skriv ett program som låter användaren mata in en radie och därefter beräknar cirkelns omkrets och area om radien är positiv. Om radien är negativ ska ett felmeddelande skrivas ut. (*)

---

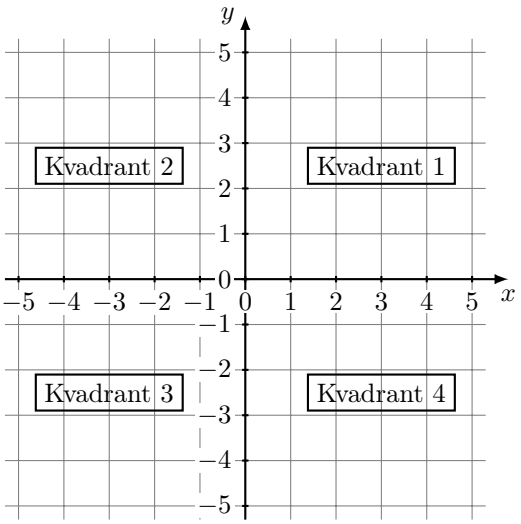
6. Skriv ett program som låter användaren mata in tre vinklar på en triangel. Avgör om triangeln har en rät vinkel.



(*)

---

7. Skriv ett program där användaren matar in x-koordinaten för en punkt och y-koordinaten för punkten. Programmet ska därefter avgöra vilken kvadrant punkten ligger i.



(*)

---

8. I SAS är reglerna för handbaggagestorlek följande:

   - max vikt: 8kg
   - max dimension: 55cm x 40cm x 23cm (längd x bredd x höjd)

   Skriv ett program som låter användaren mata in vikten, bredden, längden och höjden på ett bagage. Programmet ska avgöra om bagaget är godkänt eller inte. (*)

---

9. Skriv ett program som låter användaren mata in ett tal. Programmet ska skriva vad absolutbeloppet av talet är. $|x| = \begin{cases} x, & x \geq 0 \\ -x, & x < 0 \end{cases}$ (*)

---

10. En andragradsfunktion kan generellt beskrivas med $f(x) = ax^2 + bx + c$, där a, b, c är konstanter, $a \neq 0$. Skriv ett program som löser $f(x) = 0$, där:

    (a) $\begin{cases} a = 1 \\ b = -1 \\ c = -2 \end{cases}$

    Tips: kvadratkomplettera och lös algebraiskt ut x, alternativt PQ-formeln. Analysera diskriminanten mha if-satser. (**)

    (b) användaren matar in värden på a, b, c. Testa olika värden för a,b,c och reflektera kring när du får två, en, respektive inga reella lösningar.

```
Rötter till andragradsfunktioner

Ange ett värde för a: 1
Ange ett värde för b: 2
Ange ett värde för c: 1


x1=x2=-1.0
```

```
Ange ett värde för a: 1
Ange ett värde för b: -1
Ange ett värde för c: -2


x1 = 2.0
x2 = -1.0
```

```
Rötter till andragradsfunktioner

Ange ett värde för a: 3
Ange ett värde för b: 1
Ange ett värde för c: 2.3


Saknar reella lösninggar
```

(**)

11. Skriv ett program som låter användaren mata in en punkt (x_1, y_1) och låt programmet avgöra om punkten ligger på linjen som beskrivs av funktionen $f(x) = 3x + 5$. (**)

Repetitionssatser

I Python används for- och while-satser för att repetera kod. På engelska heter det loops och svenska kan ordet slinga användas.

For-sats

1. Skriv ett program med hjälp av for-sats som:
   (a) skriver ut talen $1, 2, \ldots, 10$ (*)
   (b) skriver ut talen $-10, -9, \ldots, 9, 10$ (*)
   (c) skriver ut talen $10, 9, \ldots, 1, 0$ (*)
2. Skriv ett program som skriver ut alla primtal från 1 till 100 med hjälp av en for-sats. (*)

---

3. Skriv ett program som beräknar följande aritmetiska summa med hjälp av for-sats:

   $s = 1 + 2 + 3 + \ldots + 99 + 100$ (*)

---

4. Skriv ett program som beräknar följande aritmetiska summa med hjälp av for-sats:

   $s = 1 + 3 + 5 + \ldots + 99$ (*)

---

5. Skriv ett program som:
   (a) Skriver ut femmans multiplikationstabell från 0 till 10 (*)
   (b) Låter användaren mata in vilken tabell, start- och slut för tabellen. (*)
   (c) Skriver ut hela multiplikationstabellen från 0 till 10. Tips: använd dig av nästlade for-satser och följande print: print(f"{$tal :< 4$}" , end = "") (**)

---

6. Skriv ett program som beräknar följande geometriska summa med hjälp av for-sats:

   $s = 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots + \frac{1}{2^n}$

   Testa olika värden på n och dra slutsats vad summan konvergerar mot. (**)

---

7. Skriv ett program som beräknar $n!$ (n-fakultet), dvs .

   $n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (n-1) \cdot n$

   Låt användaren mata in ett heltal n. (**)

---

8. Skriv ett program som beräknar följande summa med hjälp av for-sats.

   $$s = 1 + 1 + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \ldots + \frac{1}{n!}$$

   Testa olika värden på n och dra slutsats vad summan konvergerar mot. (**)

While-sats

1. Skriv ett program som beräknar följande aritmetiska summa med hjälp av while-sats:

   $$s = 1 + 2 + 3 + \ldots + 99 + 100$$ (*)

---

2. Skriv ett program som beräknar följande aritmetiska summa med hjälp av while-sats:

   $$s = 1 + 3 + 5 + \ldots + 99$$ (*)

---

3. Skriv ett program som beräknar följande geometriska summa med hjälp av while-sats:

   $$s = 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots + \frac{1}{2^n}$$

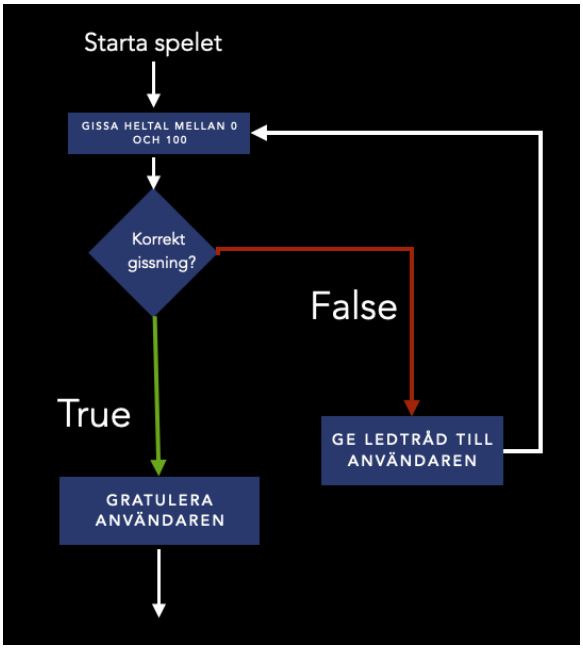
   Testa olika värden på n och dra slutsats vad summan konvergerar mot. (**)

---

4. Skriv ett program som skriver ut värden på $f(x) = x^2 - 3x$ för $-2 \leq x \leq -2$ med intervall på 0.1. Använd en while-sats. (**)

---

5. Skriv ett program som implementerar följande flödesschema:



(**)

Strängar

1. Skriv ett program som låter användaren mata in sitt namn och programmet skriver ut antalet bokstäver i namnet. (*)

---

2. Skriv ett program som räknar antalet ord i denna mening: ”En bild säger mer än tusen ord, en matematisk formel säger mer än tusen bilder” (*)

---

3. Ett palindrom är en följd av tecken som är densamma när man läser framlänges som baklänges (man räknar inte med skiljetecken och mellanslag). Exempel på palindrom är ”Anna”, ”Ni talar bra latin” och ”bjkjb”.
   (a) Skriv ett program som låter användaren mata in en följd av tecken utan mellanslag och skiljetecken och skriv ut om följden är ett palindrom. (*)
   (b) Skriv ett program som låter användaren skriva in en följd av tecken inklusive mellanslag och skiljetecken och skriv ut om följden är ett palindrom. (*)

---

4. Skriv ett program som räknar antalet vokaler i detta citat: ”Do not worry about your difficulties in Mathematics. I can assure you mine are still greater.” (**)

---

5. Skriv ett program som låter användaren mata in ett ord.
   (a) Kryptera därefter meddelandet genom att ersätta varje bokstav med nästa bokstav. Är bokstaven Ö ska den ersättas med A. Exempel: ”höjd” $\rightarrow$ ”iake” (**)
   (b) Dekryptera nu meddelandet. Exempel ”lösmfl” $\rightarrow$ ”kärlek” (**)
   (c) Låt användaren skriva in ett meddelande, välja kryptera/dekryptera och utför kryptering-en/dekrypteringen. (**)

Listor

1. Skriv ett program som simulerar 10 tärningskast, lägger in i en lista och :
   (a) sortera den från minsta talet till största (*)
   (b) sortera den från största talet till minsta (*)

---

2. Använd list comprehension för att skapa en lista med kvadrater från -10 till 10, dvs $100, 81, \ldots, 81, 100$. Rita därefter dess graf. (*)

---

3. Använd en 2D lista för att skapa koordinaterna för ett schackbräde. (*)

---

4. Skapa en 2D lista med dimensionerna 10x10 som ska representera en spelplan och fyll med nollor. Slumpmässigt placera ut 30 ettor som ska representera bomber i spelplanen.
   (a) Låt användaren mata in koordinater (x,y), $0 \leq x < 10$, $0 \leq y < 10$ tills dess att man träffar en etta. (**)
   (b) Låt användaren ha 3 liv. Varje gång hen träffar en bomb ska ett liv tas bort, klarar hen 10 omgångar utan att träffa någon bomb har hen vunnit spelet. (**)
   (c) Ge förslag på hur man kan utveckla vidare programmet och ge det ett försök. (**)

Funktioner

1. Formeln för aritmetisk summa är:

$$s_n = a_1 + a_2 + \cdots + a_n = \frac{n(a_1 + a_n)}{2}$$

   (a) Skriv en funktion som beräknar aritmetisk summa. Funktionen ska heta aritmetisk_summa och ha parametrarna n, a_1, a_n. (*)

   (b) Använd din funktion för att beräkna summan $s_{100} = 1 + 2 + 3 + \cdots + 100$ (*)

   (c) Låt användaren mata in a_1, a_n, n och programmet ska anropa funktionen aritmetisk_summa och därefter skriva ut resultatet på ett snyggt formaterat sätt. (*)

---

2. Gör en funktion som beräknar summan:

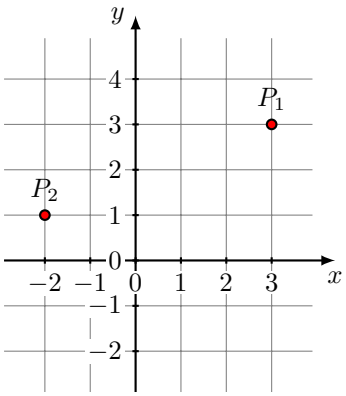
$$s = 1 + 3 + 5 + \cdots + (2n + 1)$$

   (a) Låt användaren mata in ett n och programmet ska skriva ut resultatet på ett snyggt formaterat sätt (*)

   (b) Gör om funktionen så att användaren kan mata in ett startvärde och ett n. Funktionen ska ge resultatet på ett snyggt formaterat sätt. (*)

---

3. Punkterna P_1 och P_2 är markerade i koordinatsystemet.



   (a) Beräkna avståndet mellan dessa punkter (*)

   (b) Skriv en funktion som tar in två punkter som inparametrar och som returnerar avståndet. (*)

   (c) Låt användaren mata in två punkter och använd funktionen för att beräkna avståndet mellan dessa punkter. (*)

---

4. Skapa en funktion som tar in en summa pengar som inparameter. Den ska skriva ut lämpliga sedlar/mynt som representerar denna summa. Ex 5839 ska skriva ut: 5 tusenlappar, 4 tvåhundralappar, 1 tjugolapp, 1 tia, 1 femokrona och 4 enkronor. (*)

---

5. Skapa följande funktioner som tar in ett heltal n som parameter:

   (a) och returnerar $n!$ (n-fakultet). $n! = n \cdot (n-1)(n-2)\ldots 2 \cdot 1$ (*)

(b) och beräknar följande summa $s = 1 + 1 + \frac{1}{2!} + \frac{1}{3!} + \cdots + \frac{1}{n!}$ (**)

(c) och beräknar följande summa $s = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots + \frac{(-1)^{n+1}}{2n-1}$ (**)

Testa dina funktioner med olika värden på n och se vilka resultat du får.

Felhantering

1. Finn felen och rätta dem för att beräkna avståndet mellan punkten (x,y) och origo i ett kartesiskt koordinatssystem. Notera att det inte enbart är syntaxfel.

```
impor numpy as np

def distance(x, y):
  reurn np.sqrt(x+y)

print(distance([0.5,0.5]))
```

(*)

---

2. Finn felen och rätta dem. Ändra endast funktionen, rör inte testprogrammet.

```
def ar_fyrsiffrigt(tal):
  if tal//1000 < 10 :
    return True
  else:
    return False

# testprogram
testtal = [100, 231, 10000, 10001, -1000, 102313]

for t in testtal:
  if ar_fyrsirigt(t):
    print(f"{t} är fyrsiffrigt")
  else:
    print(f"{t} är inte fyrsiffrigt")
```

(*)

---

3. Kokchun är en klumpig och ovan datoranvändare som inte brukar åka spårvagn. Skriv ett program som frågar efter antalet gånger han vill åka spårvagn, engångskostnad och månadskostnad. Därefter ska det räkna ut om man tjänar på att köpa månadskort.

   Gör ett användarvänligt program som ger tydliga felmeddelanden när han skriver fel och frågar igen när han gör fel. (*)

---

4. Gör ett program som låter användaren mata in 5 tal, programmet ska hitta och skriva ut största talet, minsta talet, medelvärdet och medianen. Programmet ska hantera fel, ge tydliga felmeddelanden och frågar igen när man gör fel. Exempel:

```
Ange 5 tal med ett mellanslag mellan varje två tal: 1 2 3 jag är cool
Fel format, formatet ska vara: tal1 tal2 tal3 tal4 tal5
Ange 5 tal med ett mellanslag mellan varje två tal: 1 3 15 6 -4 -4
Du angav 6 tal, du ska ange 5 tal
Ange 5 tal med ett mellanslag mellan varje två tal: 2 2 1 1 -2
Du angav talen: [2.0, 2.0, 1.0, 1.0, -2.0]
Största talet är: 2.0
Minsta talet är: -2.0
Medelvärdet är: 0.8
Medianen är: 1.0
```

(**)

Filhantering

1. I den här uppgiften ska du skapa en textfil vid namn diceRoll.txt från Python. Skriv lämpliga rubriker till varje deluppgift i textfilen.
   (a) Simulera 10 tärningskast och skriv dem till diceRoll.txt. (*)
   (b) Sortera tärningskasten från (a) och skriv ut dem på separat rad i diceRoll.txt. (*)
   (c) Räkna antalet femmor i kasten och skriv ut på separat rad i diceRoll.txt (*)

   Exempel på textfil:

```
Simulera 10 tärningskast: [1, 5, 5, 6, 1, 4, 2, 1, 2, 2]
Kastet sorterat: [1, 1, 1, 2, 2, 2, 4, 5, 5, 6]
Antalet femmor: 2
```

---

2. Läs in filen ”Provresultat.txt” i Python.
   (a) Läs in filen printa ut texten i terminalen. (*)
   (b) Skapa nya rader i samma fil och skriv personerna och deras resultat i alfabetisk ordning där. (*)
   (c) Skapa ytterligare nya rader i samma fil och sortera personerna efter betyg. Betygsgränserna är: $F < 20, E : 20 - 29, D : 30 - 39, C : 40 - 49, B : 50 - 59, A : 60 - 70$ (**)

3. Läs in filerna ”NPvt19Ma2A.txt” och ”NPvt19Ma2C.txt” och rita cirkeldiagram för varje betygskategori för respektive fil. Använd: matplotlib pie chart och subplot. (*)

---

4. Vi ska simulera tärningskast och se hur sannolikheten för varje siffra förändras beroende på antalet simulerade kast. Skapa en textfil från Python vid namn ”simulering.txt” och skriv antalet ettor, tvåor, treor, fyror, femmor, sexor samt andelen vid simulering av 10, 100, 1000, 10000, 100000 antal kast. (**)

```
Antal tärningskast: 10
Ettor: 0, sannolikhet 0.0
Tvåor: 1, sannolikhet 0.1
Treor: 3, sannolikhet 0.3
Fyror: 3, sannolikhet 0.3
Femmor: 2, sannolikhet 0.2
Sexor: 1, sannolikhet 0.1

Antal tärningskast: 100
Ettor: 17, sannolikhet 0.17
Tvåor: 17, sannolikhet 0.17
Treor: 20, sannolikhet 0.2
Fyror: 17, sannolikhet 0.17
Femmor: 14, sannolikhet 0.14
Sexor: 15, sannolikhet 0.15

Antal tärningskast: 1000
Ettor: 160, sannolikhet 0.16
Tvåor: 181, sannolikhet 0.181
```

Dictionary

1. Skapa en dictionary av de kurser du läser just nu med kursnamn som key och antal poäng som value. Beräkna därefter summan av antalet poäng du läser just nu. (*)

---

2. Simulera 100000 tärningskast och spara antalet ettor, tvåor, ... i en dictionary. Skriv därefter ut frekvenstabellen i terminalen. Exempel på ett utfall: (*)

```
Frekvenstabell
1: 16692
2: 16494
3: 16619
4: 16943
5: 16761
6: 16491
```

---

3. Läs in filen ”pokemonlista.txt” i Python. Skapa en variabel pokedex som är en dictionary med key:value-paren ”pokemon”:”typ, index”. Exempel när man söker på ”Gastly” och ”Pikachu”. (*)

```
print(pokedex["Gastly"])
print(pokedex["Pikachu"])
```

```
Typ: Spöke/Gift, index: 92
Typ: Elektrisk, index: 25
```

---

4. Läs in filen ”morse.txt”, spara i en dictionary och skapa en funktion som låter användaren skriva in ett meddelande för att få det översatt till morsekod. Exempel: (**)

```
print(morse("SOS"))
print(morse("POKEMON"))
```

```
...---...
.------.-.------.
```

Numpy

1. Använd funktionen arange i numpy-modulen för att skapa följande arrays:
   (a) $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$ (*)
   (b) $[-10, -9, ..., 9, 10]$ (*)

---

2. Skapa en array med 10 element mellan 10 och 100 med jämn fördelning, dvs $[10, 20, 30, \ldots, 90, 100]$. Multiplicera fältet med 10. (*)

---

3. Skriv ett program som ritar grafen till funktionen $f(x) = 3x + 1$ för $0 \leq x \leq 10$. Namnge x-axeln med ”x” och y-axeln med ”y” och använd en lämplig titel. Tips: använd funktionen linspace() i numpymodulen och modulen matplotlib. (*)

---

4. Skriv ett program som ritar grafen till funktionen $f(x) = x^2 - 1$ för $-2 \leq x \leq 2$. Namnge x-axeln med ”x” och y-axeln med ”y” och använd en lämplig titel. Tips: använd funktionen linspace() i numpymodulen och modulen matplotlib. (*)

---

5. En raket har en rörelsebana som beskrivs av $h(t) = -4t^2 + 24t + 1$, $h(t)$ är höjden i meter och t är tiden i sekunder. Rita rörelsebanan och beräkna högsta höjden. Tips: använd matplotlib och linspace i numpymodulen. (*)
