Danmarks Tekniske Universitet

Kursusnavn	Introduktion til programmering og databehandling Programmering og databehandling (andet programmeringssprog)
Kursusnummer	02631, 02632, 02633, 02634, 02692, 02694
Hjælpemidler	Alle hjælpemidler er tilladt
Varighed	2 timer
Vægtning	Opgaverne vægtes ens

Indhold

Aflevering

Du skal aflevere dine løsninger elektronisk:

1. Du kan uploade dine løsninger individuelt på CodeJudge (`dtu.codejudge.net/prog-aug16/assignment`) under *Afleveringer/Exam*. Når du afleverer en løsning på CodeJudge bliver det test-eksempel som fremgår i opgavebeskrivelsen kørt på din løsning. Hvis din løsning består denne ene test, vil den fremgå som *Submitted*. Det betyder at din løsning består på dette ene test-eksempel. Du kan uploade til CodeJudge så mange gange som du ønsker under eksamen.

2. Du skal uploade alle dine løsninger på CampusNet. Hver assignment skal uploades som en separat .py fil med samme navn som funktionen i opgaven:

   (a) `confidence.py`
   (b) `weekday.py`
   (c) `symmetrize.py`
   (d) `voldif.py`
   (e) `stringcompare.py`

   Filerne skal afleveres separat (*ikke* som en zip-fil) og skal have præcis disse filnavne.

Efter eksamen vil dine løsninger blive automatisk evalueret på CodeJudge på en række forskellige tests for at undersøge om de generelt fungerer korrekt. Bedømmelsen af din aflevering foretages udeklukkende på baggrund af hvor mange af de automatiserede test der bestås.

- Du skal sikre dig at din kode følger specifikationerne nøje.
- Hver løsning må ikke indeholde nogen yderligere kode udover den specificerede funktion.
- Husk at du kan tjekke om dine løsninger følger specifikationerne ved at uploade dem til CodeJudge.
- Bemærk at alle vektorer og matricer anvendt som input eller output skal være numpy arrays.

Assignment A Confidence interval

Når du har et antal støjfyldte observationer (repræsenteret som en vektor x af decimaltal) kan et simpelt konfidens-interval for middelværdien beregnes med følgende udtryk:

$$m \pm 2\frac{s}{\sqrt{n}} \tag{1}$$

hvor m er middelværdien, s er standard-afvigelsen, og n er antallet af observationer. Vi benytter følgende definitioner:

$$m = \frac{\sum_{i=1}^{n} x_i}{n}, \qquad s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - m)^2}{n-1}}, \tag{2}$$

hvor x_i er observationerne.

Problemdefinition

Skriv en funktion med navnet `confidence` der tager som input en vektor `x` og returnerer den nedre og øvre grænse for konfidens-intevallet som en vektor `conf` med længde to. Vektorens første element skal være den nedre grænse og dens andet element den øvre grænse.

Løsningsskabelon

```
def confidence(x):
  #insert your code
  return conf
```

Input

`x`	Observationer (vektor med decimaltal).

Output

`conf`	Nedre og øvre grænse for konfidensinterval (vektor med længde 2).

Eksempel

Forestil dig følgende input vektor $x = [1,\ 2,\ 4,\ 3,\ 1]$. Middelværdien og standard-afvigelsen kan bregegnes som

$$m = \frac{1+2+4+3+1}{5} = 2.2, \qquad s = \sqrt{\frac{(1-2.2)^2 + (2-2.2)^2 + (4-2.2)^2 + (3-2.2)^2 + (1-2.2)^2}{5-1}} = 1.3038,$$

Konfidens-intervallet er således givet ved

$$2.2 \pm 2 \cdot \frac{1.3038}{\sqrt{5}} \quad = \quad 2.2 \pm 1.1662$$

og funktionen skal derfor returnere vektoren $[1.0338,\ 3.3662]$.

A

Assignment B Day of the week

Den følgende formel kan benyttes til at beregne ugedagen for en vilkårlig dato:

$$w = \left(d + C + y + \left\lfloor \frac{y}{4} \right\rfloor\right) \bmod 7 \tag{3}$$

Input til beregningen er dato-tallet $d \in \{1 \ldots 31\}$, måneds-tallet $m \in \{1 \ldots 12\}$ og de sidste to cifre i årstallet $y \in \{0 \ldots 99\}$. C er en måneds-kode (month code) som kan bestemmes ud fra m ved hjælp af tabellen nedenfor. Notationen $\lfloor \cdot \rfloor$ betyder *nedrunding* til nærmeste lavere heltal (floor-funktionen) og mod er modulo operatoren (rest efter heltals-division).

Month (m)	1	2	3	4	5	6	7	8	9	10	11	12
Month code (C)	6	2	2	5	0	3	5	1	4	6	2	4

Resultatet af beregningen er en ugedags-kode w (weekday code) som svarer til følgende ugedags-navne (weekday name):

Weekday code (w)	0	1	2	3	4	5	6
Weekday name	`Sun`	`Mon`	`Tue`	`Wed`	`Thu`	`Fri`	`Sat`

Problemdefinition

Skriv en funktion med navnet `weekday` der tager som input dato-, måneds- og årstal, og returnerer ugedagens navn som en streng skrevet præcis som i tabellen ovenfor.

Løsningsskabelon

```
def weekday(d, m, y):
  #insert your code
  return name
```

Input

`d`	Dato-tal (heltal, $1 \ldots 31$).
`m`	Måneds-tal (heltal, $1 \ldots 12$).
`y`	Årstal (heltal, $0 \ldots 99$).

Output

`name`	Ugedagens navn (streng).

Eksempel

Forestil dig datoen 21. august 2016, som er repræsenteret ved inputtet $d = 21$, $m = 8$, $y = 16$. Når vi slår $m = 8$ op i måneds-kode-tabellen fås $C = 1$. Ugedags-koden kan da beregnes som:

$$w = \left(21 + 1 + 16 + \left\lfloor \frac{16}{4} \right\rfloor\right) \bmod 7 = (21 + 1 + 16 + 4) \bmod 7 = 42 \bmod 7 = 0 \tag{4}$$

Navnet på ugedagen kan nu findes ved at slå ugedags-koden op i tabellen, og strengen `Sun` er således det endelige resultat.

B

Assignment C Matrix symmetrization

I lineær algebra er en symmetrisk matrix en kvadratisk matrix som er lig sin egen transponerede. Givet en arbirær kvadratisk matrix x, kan vi konstruere en symmetrisk matrix y som følger:

For *hvert element* (i, j) *i matricen.*
- **Hvis** $i = j$
  - Sæt $y_{i,j} = x_{i,j}$.
- **ellers**
  - Sæt $y_{i,j} = x_{i,j} + x_{j,i}$.

Problemdefinition

Skriv en funktion med navnet `symmetrize` der tager som input en kvadratisk matrix x og returnerer en symmetriseret matrix y beregnet ifølge ovenstående algoritme.

Løsningsskabelon

```
def symmetrize(x):
  #insert your code
  return y
```

Input	
`x`	Matrix der skal symmetriseres (kvadratisk matrix).

Output	
`y`	Symmetriseret matrix (symmetrisk matrix).

Eksempel

Forestil dig følgende input matrix:

$$x = \begin{bmatrix} 1.2 & 2.3 & 3.4 \\ 4.5 & 5.6 & 6.7 \\ 7.8 & 8.9 & 10.0 \end{bmatrix}.$$

Ifølge algoritmen skal output-matricen være:

$$y = \begin{bmatrix} 1.2 & 6.8 & 11.2 \\ 6.8 & 5.6 & 15.6 \\ 11.2 & 15.6 & 10.0 \end{bmatrix}.$$

C

Assignment D Volume difference

En hyperkugle er en generalisering af en cirkel (2-d) og en kugle (3-d) til et n-dimensionelt rum. Volumen af en hyperkugle er givet ved:

$$V_s = \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)} R^n, \tag{5}$$

hvor n er dimensionaliteten af rummet, R er hyperkuglens radius og $\Gamma(\cdot)$ er gamma-funktionen, som er implementeret i Python som funktionen `math.gamma`.

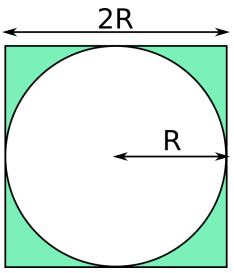
En hyperkugle med radius R kan rummes i en hyper-kube med sidelængde $2R$. Volumen af en sådan hyper-kube er givet ved:

$$V_c = (2R)^n. \tag{6}$$

Forskellen mellem de to volumener er givet ved

$$V_d = V_c - V_s, \tag{7}$$

og er illustreret ved det farvede areal i figuren nedenfor, som viser det 2-dimensionelle tilfælde.



Problemdefinition

Skriv en funktion med navnet `voldif` der tager radius R og dimensionaliteten n som input, og returnerer forskellen mellem volumen af hyper-kuben og hyper-kuglen, V_d.

Løsningsskabelon

```
def voldif(R, n):
  #insert your code
  return Vd
```

Input	
`R`	Radius (ikke-negativt decimaltal).
`n`	Dimensionalitet (positivt heltal).

Output	
`Vd`	Forskel mellem volumen af hyper-kube og hyper-kugle (decimaltal).

Eksempel

Forestil dig en radius på $R = 5$ and $n = 2$ dimensioner. Volumenerne (som faktisk er arealer i det 2-dimensionelle tilfælde) kan beregnes som:

$$V_s = \frac{\pi^{\frac{2}{2}}}{\Gamma(\frac{2}{2}+1)} 5^2 \approx 78.54, \qquad V_c = (2 \cdot 5)^2 = 100,$$

og forskellen, som er det endelige resultat, er givet ved

$$V_d = 100 - 78.54 = 21.46\,.$$

D

Assignment E String comparison

Et mål for dis-similaritet mellem tekst-strenge kan for eksempel bruges til at sammenligne to strenge fra forskellige data-kilder. I denne opgave vil vi arbejde med et simpelt mål defineret som antallet af forskellige bogstaver `a`–`z` der optræder i en og kun en af de to strenge. Hvis, for eksempel, den ene streng indeholder to `a`'er og den anden ikke indeholder nogen, tæller det som en forskel på 1, og hvis den ene streng indeholder to `a`'er og den anden indeholder et `a`, tæller det ikke som en forskel idet begge strenge indeholder bogstavet `a`. Du kan antage at input kun indeholder små bogstaver `a`–`z`.

Problemdefinition

Skriv en funktion med navnet `stringcompare` der tager som input to strenge og returnerer deres dis-similaritet som defineret ovenfor.

Løsningsskabelon

```
def stringcompare(string1, string2):
  #insert your code
  return disSimilarity
```

Input	
`string1`, `string2`	Strenge der skal sammenlignes (streng).

Output	
`disSimilarity`	Dis-similaritet (heltal).

Eksempel

Forestil dig at sammenligne de to strenge `aardvark` og `artwork`. Bogstaverne `a`, `r` og `k` optræder i begge strenge og kan derfor ignoreres. De to bogstaver `d` og `v` optræder kun i den første streng, og de tre bogstaver `t`, `w` og `o` optræder kun i den anden streng. Derfor er dis-similariteten $2+3=5$, og funktionen skal returnere tallet 5.

E