Danmarks Tekniske Universitet

Kursusnavn	Introduktion til programmering og databehandling
Kursusnummer	02633 (02631, 02632, 02634, 02692)
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

   https://dtu.codejudge.net/prog-jun20/assignments

   under *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 DTU's Onlineeksamen site. Hver assignment skal uploades som en separat .py fil med samme navn som funktionen i opgaven:

   (a) `stable_measurements.py`
   (b) `smooth_curve.py`
   (c) `slide_row.py`
   (d) `punctuation_check.py`
   (e) `stock_status.py`

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

Efter eksamen vil dine løsninger fra DTU Inside blive automatisk evalueret på CodeJudge på en række forskellige tests, for at undersøge om de generelt fungerer korrekt. Bedømmelsen af din aflevering foretages udelukkende 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, dog kan `import`-sætninger inkluderes.
- 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 1 Stable measurements

Vi betragter et eksperiment for at måle bakterievækst. Det er kendt, at måleintrumentet giver ustabile værdier tidligt i eksperimentet. Måleværdierne skal derfor deles i det ustabile og det stabile interval. De ustabile målinger skal fjernes fra data, så kun intervallet med stabile målinger beholdes. Det er besluttet, at det stabile interval starter med den første måleværdi, som er strengt større end alle tidligere måleværdier, og strengt mindre end alle senere måleværdier. Starten på det stabile interval kan ikke være den første eller den sidste af de oprindeliige måleværdier. Envidere man må godt antage at mindst tre målinger er taget.

Problemdefinition

Skriv en funktion `stable_measurements` som tager en vektor med alle måleværdier som input, og returnerer en vektor med alle stabile måleværdier. Hvis det stabile interval ikke kan findes (for eksempel, hvis alle måleværdierne er ens), skal funktionen returnere en tom vektor.

Løsningsskabelon

```
def stable_measurements(m):
  #insert your code
  return s
```

Input	
`m`	Oprindelige måleværdier, en vektor med mindst tre elementer.

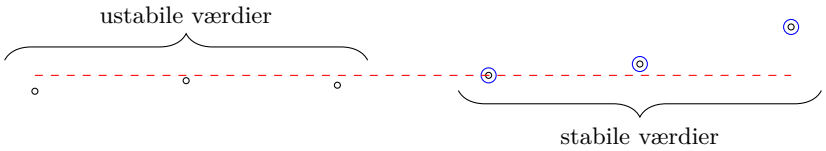
Output	
`s`	En vektor med stabile måleværdier.

Eksempel

Betragt sekvensen

$$\mathbf{m} = [4.3\,, \quad 5.7\,, \quad 5.1\,, \quad 6.4\,, \quad 7.9\,, \quad 12.8]$$

som også er vist i illustrationen.



Måleværdien 5.7 er her ikke starten på det stabile interval, da den er større end en senere måling på 5.1. Målæværdien 5.1 er ikke starten på det stabile interval, da den er mindre end en tidligere måling på 5.7. Måleværdien 6.4 er strengt større end alle tidligere målinger (5.7 og 5.1) og er stengt mindre end alle senere målinger (7.9 og 12.8), så 6.4 er starten på det stabile interval. Derfor

$$\mathbf{s} = [6.4\,, \quad 7.9\,, \quad 12.8]\,.$$

Assignment 2 Smooth curve

En lukket kurve er repræsenteret med en sekvens af 2D punkter $p_i = (x_i, y_i)$, $i = 1, \ldots, N$, som er forbundet med linjesegmenter, hvor det antages, at det sidste punkt er forbundet med det første. Kurven udglattes ved at flytte hvert punkt p_i en smule i retningen af en position midtvejs mellem dets to naboer p_{i-1} og p_{i+1}, hvor man skal sørge for at flytte det første og det sidste punkt korrekt. De nye koordinater for kurvepunkterne, hvor $i = 2, \ldots, N-1$, kan udregnes som

$$x_i^{\text{new}} = (1-\alpha)x_i + \alpha\frac{x_{i-1} + x_{i+1}}{2}, \qquad y_i^{\text{new}} = (1-\alpha)y_i + \alpha\frac{y_{i-1} + y_{i+1}}{2}$$

For det første og det sidste punk har vi

$$x_1^{\text{new}} = (1-\alpha)x_1 + \alpha\frac{x_N + x_2}{2}, \qquad y_1^{\text{new}} = (1-\alpha)y_1 + \alpha\frac{y_N + y_2}{2}.$$

og

$$x_N^{\text{new}} = (1-\alpha)x_N + \alpha\frac{x_{N-1} + x_1}{2}, \qquad y_N^{\text{new}} = (1-\alpha)y_N + \alpha\frac{y_{N-1} + y_1}{2}.$$

Parametren α angiver styrken af udglatningen og stilles normalt til en værdi mellem 0.1 og 0.5.

Problemdefinition

Skriv en funktion `smooth_curve` som tager som input: en matrix $\mathbf{C}$ af størrelsen $2 \times N$ med koordinaterne af kurvepunkterne (x_i i den første række, y_i i den anden række), og en skalar α. Funktionen skal returnere en matrix af den samme størrelse som $\mathbf{C}$, men koordinaterne af den udglattede kurve.

Løsningsskabelon

```
def smooth_curve(C, alpha):
  # insert your code
  return S
```

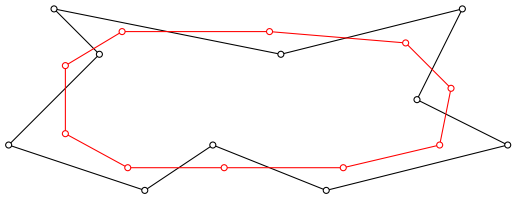
Input	
`C`	En matrix af størrelse $2 \times N$ med koordinaterne af kurvepunkterne, $N \geq 3$
`alpha`	Skalar med udglatningsparameter α (flydende tal).

Output	
`S`	En matrix af størrelse $2 \times N$ med koordinaterne af den udglattede kurve.

Eksempel

Betragt en kurve, som også er vist på illustrationen i sort, givet af

$$\mathbf{C} = \begin{bmatrix} 24 & 40 & 36 & 44 & 28 & 18 & 12 & 0 & 8 & 4 \\ 12 & 16 & 8 & 4 & 0 & 4 & 0 & 4 & 12 & 16 \end{bmatrix},$$



og en udglatningsparameter

$$\alpha = 0.5\,.$$

De nye koordinater af de første to punkter kan udregnes som

$$x_1^{\text{new}} = (1-0.5)24 + 0.5\frac{40+4}{2} = 23, \qquad y_1^{\text{new}} = (1-0.5)12 + 0.5\frac{16+16}{2} = 14\,,$$

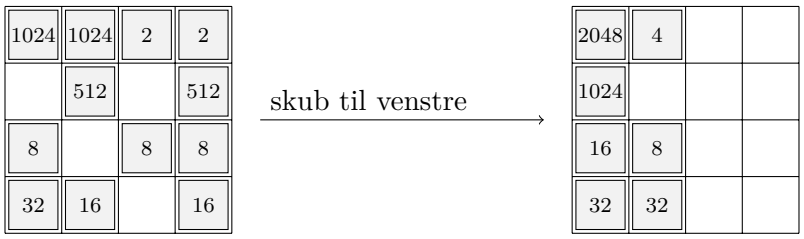
$$x_2^{\text{new}} = (1-0.5)40 + 0.5\frac{24+36}{2} = 35, \qquad y_2^{\text{new}} = (1-0.5)16 + 0.5\frac{12+8}{2} = 13\,,$$

og lignende udregning udføres of de resterende koordinater. Den udglattede kurve, vist på illustrationen med rødt, er

$$\mathbf{S} = \begin{bmatrix} 23 & 35 & 39 & 38 & 29.5 & 19 & 10.5 & 5 & 5 & 10 \\ 14 & 13 & 9 & 4 & 2 & 2 & 2 & 5 & 11 & 14 \end{bmatrix}.$$

Assignment 3 Slide row

2048 er et spil for en spiller, baseret på et 4×4 gitter, hvor der på nogle af gitterpladserne findes brikker med tal, som kan skubbes af spilleren. Når de skubbes, slåes brikker med ens værder sammen og deres værtier summeres, som vist nedunder.



Her betragter vi kun en horisontal række, og et skub af brikkerne til venstre. For dette er reglerne:

- Brikkerne flyttes så langt til venstre som muligt, indtil de stoppes af andre brikker, eller af kanten af gitteret.
- Hvis to brikker, som står ved siden af hinanden har den samme værdi, slåes de sammen til en brik med en værdi, som svarer til summen af værdierne på de to brikker. Den resulterede brik kan ikke slåes sammen med en anden brik.
- Hvis tre brikker ved siden af hinanden har den samme værdi, slåes kun de to brikker til venstre sammen. Hvis alle fire brikker i rækken har den samme værdi, bliver de første to og de sidste to slået sammen.

Problemdefinition

Skriv en funktion `slide_row` som tager en vektor med 4 elementer som input. Elementerne i vektoren er heltal, som repræsenterer brikker, og nul repræsenterer en tom plads på gitteret. Funktionen skal returnere en vektor med 4 elementer, som repræsenterer brikker efter et skub til venstre. Værdierne på brikkerne kan være vilkårlige heltal, og ikke kun potenser af 2 som i det originale 2048 spil.

Løsningsskabelon

```
def slide_row(R):
  # insert your code
  return S
```

Input	
`R`	En vektor med 4 elementer som repræsenterer en række af gitteret inden skubbet.

Output	
`S`	En vektor med 4 elementer som repræsenterer en række af gitteret efter skubbet.

Eksempel

Bætragt en række

$$\mathbf{R} = [8,\ 13,\ 0,\ 13]\,.$$

Kun brikken helt til højre kan flyttes til venstre, og rækken bliver $[8, 13, 13, 0]$. Nu står to brikker med værdien 13 ved sidden af hinanden, og de slåes sammen til en brik med værdien 26. Da ingenting kan flyttes mere til venstre, skal funktionen returnere

$$\mathbf{S} = \underline{[8,\ 26,\ 0,\ 0]}\,.$$

Assignment 4 Punctuation check

I trykte tekster skal nogle tegn altid efterfølges af et mellemrum. Det gælder for:

- Tegn som bruges til at afslutte en sætning: punktum (.), spørgsmålstegn (?) og udråbstegn (!).
- Tegn som indikerer en pause: komma (,), semikolon (;) og kolon(:).

For en automatisk evaluering af tekster, skal vi tælle alle tegnsætningsfejl, hvor tegn (.), (?), (!), (,), (;) og (:) *ikke* bliver efterfulgt af et mellemrum. Hvis den sidste karakter i teksten er en af de seks tegn, regnes det ikke som en fejl.

Problemdefinition

Skriv en funktion `punctuation_check`, som tager som input en streng som repræsenterer tekst, og returnerer antallet af tegnsætningsfejl.

Løsningsskabelon

```
def punctuation_check(text):
    #insert your code
    return n
```

Input	
`text`	Streng der repræsenterer en tekst.

Output	
`n`	Skalar med antal tegnsætningsfejl i teksten (heltal).

Eksempel

Betragt teksten givet med en streng

```
text = 'This is:a text, a text with many,many errors!  But,who cares?'
```

som har følgende fejl: `':a'`, `',m'` and `',w'`. Så funktionen burde returnere 3.

4

Assignment 5 Stock status

En web-shop har følgende information for hver af deres produkter: x er et antal af varer de har på lager; og d er antallet af dage indtil de nye varer ankommer på lager, hvor 0 indekerer, at ingen nye varer er på vej. For at informere kunderne om deres lagerstatus, bruger de reglen:

Besked	Lagerstatus
`In stock`	5 eller flere varer på lager.
`Only <x> items in stock`	1, 2, 3 eller 4 varer på lager, `<x>` er antal varer på lager, x.
`In stock again in <d> days`	Ingen varer på lager, men nye varer forventes, `<d>` er antal dage intil de nye varer ankommer, d
`Out of stock`	Ingen varer på lager, og der er ingen nye varer på vej.
`Unknown stock status`	Enten x eller d er negative.

Problemdefinition

Skriv en funktion `stock_status` som tager som input et tal x som repræsenterer antal varer på lager, og et tal d som repræsenterer antal dage indtil de nye varer ankommer. Funktionen skal returnere en streng med en besked, som angiver lagerstatus.

Løsningsskabelon

```
def stock_status(x, d):
  # insert your code
  return s
```

Input

`x`	Antal af varer på lager (heltal).
`d`	Antal af dage indtil de nye varer ankommer (heltal).

Output

`s`	En besked med et lagerstatus (streng).

Eksempel

Betragt $x = 3$ og $d = 0$. Da der er mellem 1 og 4 varer på lager, burde strengen

`Only 3 items in stock`

returneres af funktionen.