Kursusnavn:	Programmering
Kursusnummer:	02002 og 02003
Eksamensdato:	Midtvejs prøve-eksamen, oktober 2023
Hjælpemidler:	Alle hjælpemidler, ingen internet
Eksamensvarighed:	2 timer for midtvejs prøve-eksamen (4 timer for ægte eksamen i December)
Vægtning:	Alle opgaver vægtes ens

Eksamensvejledning

Opgave 1: Renters rente (Compound Interest)

Opgave 2: Lagerstatus (Stock Status)

Opgave 3: Første alarm (First Alarm)

Opgave 4: Hyppigste efterfølger (Typical Successor)

Opgave 5: Fair terningekast (Dice Fairness)

Eksamensvejledning

Forudsætninger

For at kunne løse eksamensopgaverne skal du have en computer med Python, VSCode, kursusværktøjskasse og softwarepakker installeret.

Eksamensmateriale

Eksamensmaterialet består af:

- Eksamensteksten, et PDF-dokument med navnet indeholdende `midterm2023fall` tilgængeligt på dansk (dette dokument) og engelsk.
- Download-scriptet `download_midterm2023fall.py`.

Du skal gemme download-scriptet, åbne det i VSCode og køre det. Scriptet vil oprette de filer og mapper, som du skal bruge til at løse eksamensopgaverne, svarende til hvordan ugentlige øvelser blev oprettet. Download scriptet vil oprette følgende mapper og filer:

- En opgavemappe `02002students/cp/midterm2023fall/tasks/` indeholdende en Python-fil for hver opgave, der fungerer som skabelon for den kode, du skal skrive.
- En projektmappe `02002students/cp/midterm2023fall/project/` indeholdende en Python-fil til at teste og en Python-fil til at grade (bedømme) dine løsninger.

Hvis download-scriptet støder på et problem og ikke kan finde mappen `02002students`, vil det oprette filer og mapper på placeringen af download-scriptet.

Løsning af eksamensopgaver

Når du løser eksamensopgaver, skal du følge instruktionerne fra eksamensteksten og udfylde den medfølgende skabelon Python-filer fra opgavemappen.

Du kan teste dine løsninger ved at køre det udleverede testscript `midterm2023fall_tests.py`, som tester din løsning på et lille antal testcases.

At løse opgaverne og bruge testscriptet minder om, hvordan du har løst ugentlige øvelser og projekter. Til eksamen vil der dog blive brugt yderligere testcases under evalueringen efter eksamen.

Aflevering af løsning

For at aflevere din løsning skal lave en token fil ved at køre det udleverede grading script `midterm2023fall_grade.py`. Upload token-filen til eksamenssystemet.

Derudover, som en ekstra sikkerhed, indsend dine løsninger som én individuel Python-fil for hver opgave. Det vil sige, indsend de udfyldte Python-filer fra opgavemappen. Vi vil bruge Python-filerne, hvis vi støder på problemer med token-filen.

Kun relevant for eksamen i december. Når du bruger DE-systemet (digital eksamen) på `https://eksamen.dtu.dk/`, skal du vælge en fil som hovedfil, og de andre filer kan tilføjes som vedhæftninger. Vi foreslår, at du indsender token-filen som hovedfil, men dette er ikke et krav.

Opgave 1: Renters rente (Compound Interest)

Renters rente er et udtryk fra bankverden. Formlen for renters rente I efter et år er

$$I = P\left(1 + \frac{r}{n}\right)^n - P.$$

Her er P hovedstolen, r er rentesatsen (som decimaltal f.eks. 4% er givet som $r = 0.04$), og n er tilskrivningsfrekvensen.

Du skal skrive en funktion, der tager tre tal som input: `principal` for hovedstolen, `rate` for rentesats og `frequency` for tilskrivningsfrekvens. Funktionen skal returnere renters rente.

Overvej for eksempel inputtet:

```
>>> compound_interest(1500, 0.04, 8)
61.06056588815591
```

Da hovedstolen er 1500 (i en bestemt valuta, f.eks. DKK), rentesatsen er 0.04 (dvs. rentesats på 4%), og tilskrivningsfrekvensen er 8 (dvs. tilskrivning 8 gange i året), er renters rente

$$I = 1500\left(1 + \frac{0.04}{8}\right)^8 - 1500,$$

som er ca. 61.06, og det er det, din funktion skal returnere.

For at kunne aflevere din løsning skal du indsætte den i filen: `cp/midterm2023fall/tasks/compound_interest.py`.

Specifikationerne er:

`cp.midterm2023fall.tasks.compound_interest.`**`compound_interest`**(*principal*, *rate*, *frequency*)

Return the compound interest given principal, rate and frequency.

Parametre

- **`principal`** (`int`) – A positive integer, the principal sum.
- **`rate`** (`float`) – A positive float, the interest rate.
- **`frequency`** (`int`) – A positive integer, the compounding frequency.

Returtype
`float`

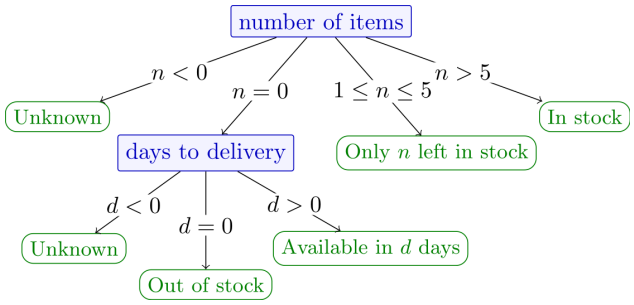
Returnerer
The compound interest.

Opgave 2: Lagerstatus (Stock Status)

Når kunder besøger en webshop for et bestemt produkt, ser de oplysninger om lagerstatus. Denne information er baseret på to tal: `number_of_items`, antallet af varer på lager i øjeblikket; og `days_to_delivery`, antallet af dage indtil levering af nye varer fra fabrikken, hvor 0 angiver, at der ikke forventes nye varer.

Vist tekst	Lagersituation
`In stock`	6 eller flere varer på lager
`Only <n> left in stock`	Mellem 1 og 5 varer på lager. Her er <n> antallet af varer på lager.
`Available in <d> days`	Ingen varer på lager, men nye varer forventes. Her er <d> antallet af dage til levering.
`Out of stock`	Ingen varer på lager, og der forventes ingen nye varer.
`Unknown`	Enten er antallet af varer negativt, eller også er der ingen varer på lager, og antallet af dage til levering er negativt.

En anden måde at repræsentere lagerstatus på er med et beslutningstræ:



Din opgave er at skrive en funktion, der tager `number_of_items` og `days_to_delivery` som input. Funktionen skal returnere en `str` med den tekst, der skal vises.

Overvej for eksempel inputtet:

```
>>> stock_status(4, 7)
'Only 4 left in stock'
```

Her, `number_of_items` er mellem 1 og 5, og derfor bør funktionen returnere strengen `"Only 4 left in stock"`.

For at kunne aflevere din løsning skal du indsætte den i filen: `cp/midterm2023fall/tasks/stock_status.py`.

Specifikationerne er:

`cp.midterm2023fall.tasks.stock_status.`**`stock_status`**`(`*number_of_items*, *days_to_delivery*`)`

Return stock status message given number of items and days to delivery.

Parametre

- **`number_of_items`** (`int`) – An integer, the number of items in stock.
- **`days_to_delivery`** (`int`) – An integer, the number of days to delivery.

Returtype

`str`

Returnerer

The stock status message.

Opgave 3: Første alarm (First Alarm)

Vandstanden i en flod måles (i meter) og registreres hver time. En alarm udløses, hvis en af følgende to betingelser er opfyldt:

1. Vandstanden er steget med mere end 0.2 meter i løbet af den sidste time, og den resulterende vandstand er højere end 1.5 meter.
2. Vandstanden er over 2.0 meter.

Vandstandsmålingerne er gemt i en liste kaldet `water_levels`. Din opgave er at skrive en funktion, der returnerer indekset for den første alarm. Hvis der ikke udløses en alarm, skal funktionen returnere -1. Alle uligheder er strenge ($<$ er streng mens $\leq$ ikke er det); for eksempel skal vandstanden være strengt over 2.0 meter for at udløse en alarm.

Overvej for eksempel inputtet:

```
>>> first_alarm([1.52, 1.29, 1.32, 1.18, 1.45, 1.63, 1.81, 1.95, 2.11, 2.09, 1.98, 1.3])
8
```

Stigningen i vandstanden mellem to på hinanden følgende timer er strengt taget kun større end 0.2 meter mellem målingerne ved indeks 3 og 4. Den resulterende vandstand ved indeks 4 er dog ikke strengt over 1.5 meter, så der udløses ingen alarm der. En alarm udløses ved indeks 8, fordi vandstanden strengt taget er over 2.0 meter. Derfor returnerer funktionen 8.

For at kunne aflevere din løsning skal du indsætte den i filen: `cp/midterm2023fall/tasks/first_alarm.py`.

Specifikationerne er:

`cp.midterm2023fall.tasks.first_alarm.`**`first_alarm`**`(`*water_levels*`)`

Return the index of the first alarm given the list of water levels.

Parametre
`water_levels` (`list`) – A list of floats, the water levels.

Returtype
`int`

Returnerer
The index of the first alarm.

Opgave 4: Hyppigste efterfølger (Typical Successor)

På forskellige sprog er nogle kombinationer af bogstaver mere almindelige end andre. På engelsk bliver bogstavet `q` for eksempel næsten altid efterfulgt af bogstavet `u`.

I denne opgave skal du skrive en funktion, der tager to strenge som input: `text`, en streng, der repræsenterer noget tekst; og `letter`, en streng, der indeholder et lille bogstav fra det engelske alfabet. Funktionen skal returnere det bogstav, der hyppigst kommer efter det givne bogstav i teksten. Hvis det givne bogstav ikke findes i teksten eller ikke efterfølges af noget andet bogstav, skal funktionen returnere en tom streng.

De følgende regler gælder:

- Inputteksten kan indeholde både store og små bogstaver, men du skal behandle dem alle som små bogstaver. For eksempel skal `A` og `a` behandles som `a`.
- Alle tegn, der ikke er bogstaver i det engelske alfabet, skal ignoreres. De skal dog behandles som brud i rækkefølgen. For eksempel, i `up-down`, er bogstavet `p` *ikke* efterfulgt af bogstavet `d`.
- Generelt kan to eller flere bogstaver være de hyppigste efterfølgere. I vores test er der altid *præcis én* hyppigst efterfølger. Derfor kan man gå ud fra, at hvis bogstavet findes i teksten og har efterfølgere, så er der præcis én, der er den hyppigste.

Overvej for eksempel inputtet:

```
>>> text = 'Hello world. This usual salutation usually starts programming.'
>>> typical_successor(text, 'l')
'l'
```

I denne tekst streng, bogstavet `l` optræder 7 gange. Det er 6 gange efterfulgt af et andet bogstav og en gang af et mellemrum. Bogstavet `l` efterfølges af `d` én gang, af `l` to gange, af `o` én gang, af `u` én gang, og af `y` én gang. Derfor skal `l` returneres.

For at kunne aflevere din løsning skal du indsætte den i filen: `cp/midterm2023fall/tasks/typical_successor.py`.

Specifikationerne er:

`cp.midterm2023fall.tasks.typical_successor.`**`typical_successor`**(*text*, *letter*)

Return the letter that most often follows the given letter in the text.

Parametre

- **`text`** (`str`) – A string, the text.
- **`letter`** (`str`) – A string, the letter.

Returtype
`str`

Returnerer
The letter that most often follows the given letter.

Opgave 5: Fair terningekast (Dice Fairness)

Du vil tjekke, om et sæt terningekast er fair ved at besvare tre spørgsmål: Hvilket tal optræder oftest? Hvor mange gange optræder det? Hvad er det forventede antal gange, hvert tal optræder?

Din opgave er at skrive en funktion, der som input tager en liste med tal fra 1 til 6 og returnerer en tuple med 3 elementer, der indeholder:

- Det tal, der optræder hyppigst blandt kastene.
- Antallet af gange, det hyppigste tal optræder.
- Det forventede antal gange et tal dukker op, som beregnes ved at dividere det samlede antal kast med 6.

Du kan antage, at inputlisten ikke er tom og kun indeholder tal fra 1 til 6. Hvis der er uafgjort hvilket er det hyppigste tal, skal funktionen returnere det mindste tal.

For eksempel kan du overveje følgende liste af terningekast:

```
>>> throws = [4, 2, 4, 4, 5, 6, 1, 2, 3, 4, 2, 3, 5, 5, 4, 4, 3, 2, 1, 4, 6]
>>> dice_fairness(throws)
(4, 7, 3.5)
```

Det hyppigste tal er 4, som forekommer 7 gange. Der er 21 kast, så det forventede antal gange for et tal at blive kastet er 21 divideret med 6, hvilket er 3.5. Funktionen skal derfor returnere tuplen `(4, 7, 3.5)`.

For at kunne aflevere din løsning skal du indsætte den i filen: `cp/midterm2023fall/tasks/dice_fairness.py`.

Specifikationerne er:

`cp.midterm2023fall.tasks.dice_fairness.`**`dice_fairness`**`(`*`throws`*`)`

Return the 3-element tuple containing dice statistics.

Parametre
`throws` (`list`) – A list of integers, the throws of a dice.

Returtype
`tuple`

Returnerer
A 3-element tuple containing information about the throws.