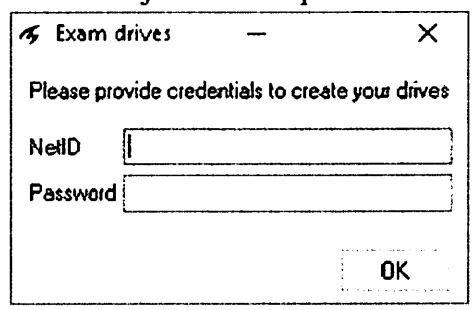
AM1090 Inleiding Programmeren
"Instructieblad"

Log in op de computer met

- Login naam: **`EWI-AM1090`**
- Password: **`Welcome4492@TU`**

Let op: het eerste teken van het password is een hoofdletter W en TU zijn ook hoofdletters.

Vul hierna je NetID en password in:



Klik op OK; bij de melding nogmaals op OK.

Sla documenten op op Private (P:) met de juiste namen

Dubbel klik op MyComputer. Dubbel klik op Shared (S:). Je ziet nu de documenten: **`A1.py`**, **`A2.py`** en **`A3.py`**. Kopieer deze documenten naar de Private (P:) schijf zoals hierna beschreven wordt. Selecteer de 3 files **`A1.py`**, **`A2.py`** en **`A3.py`** (met Shift-Click) en kopieer deze met CTRL + c. Ga naar This PC, dubbel klik op Private (P:) en plak de files met CTRL + v. **Kopieer geen files met slides of het boek naar Private (P:).** Deze files kun je openen vanaf de Shared (S:) schijf.

Klik voor iedere **`A?.py`** file twee keer op de file naam en verander de naam zoals hieronder beschreven. De nieuwe namen van de files **`A1.py`**, **`A2.py`** en **`A3.py`** moeten je naam en studienummer bevatten:
`A1`<achternaam><voorletters><voorvoegsels><studienummer>**`.py`**
De **`.py`** extensie staat er automatisch achter. Tik geen extra **`.py`** achter de filenaam. Hieronder staat een voorbeeld filenaam (let ook op de hoofdletters):

`A1NieuwenhuizenPRvan1234567.py`

Open de pdf-files met het boek "Think Python" en de college slides

Kopieer geen files met slides of het boek naar Private (P:) maar open de files vanaf Shared (S:). Ga naar This PC en dubbel klik op Shared (S:). Dubbel klik op `thinkpython_Python3.pdf` en kies Adobe Acrobat Reader DC. Open ook de andere pdf file met de slides op deze manier.

Start Spyder

Start Spyder door dubbel te klikken op het icoon links op je desktop. Open de **`.py`** bestanden door middel van File | Open in Spyder. De files kunnen niet worden geopend door dubbel klikken op een **`.py`** document.

Beëindigen van je sessie / inleveren van het werk

Sluit alle .py files, pdf files en alle windows.
Klik op MyComputer of kies Start | Programs | MyComputer.
Dubbel klik op Private (P:). Controleer of alle `.py` files met je naam en studienummer aanwezig zijn op Private (P:). **Op grond van deze files wordt je werk beoordeeld.** Je kunt overige bestanden op Private (P:) verwijderen, maar pas op dat je de bestanden die je wilt inleveren niet per ongeluk verwijdert.
Als je geen documenten met je naam en studienummer op Private (P:) hebt staan, dan heb je geen werk ingeleverd.

Log uit met Start | "TU vlam" | Sign Out

DELFT UNIVERSITY OF TECHNOLOGY
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

AM1090 Inleiding Programmeren
2e deeltentamen
1 februari 2024, 9:00 – 12:00 uur

Normering: cijfer is (aantal punten + 3) / 3

Opgave 1 **Puntenverdeling: a) 3 punten; b) 3 punten; c) 3 punten**

Gebruik bij deze opgave het bestand `A1.py` en breid dit bestand uit. Vul je naam en studienummer in, bovenaan in de file.

a. Implementeer een functie `create_dict(lst)` die bij een lijst `lst` met woorden een dictionary maakt en teruggeeft. De keys in de dictionary zijn de woorden uit `lst`. De waarden in de dictionary zijn tupels met twee gehele getallen. Het eerste getal is het aantal klinkers in het woord. Het tweede getal is het aantal medeklinkers in het woord. De letters a, e, i, o en u zijn klinkers. Alle andere letters zijn medeklinkers.

   Voorbeeld met `lst = ['aardbeiensoesje','boterkoek', 'eierkoek']`:
   Aanroep `create_dict(lst)` geeft uitvoer
   `{'aardbeiensoesje': (8, 7), 'boterkoek': (4, 5), 'eierkoek': (5, 3)}`

b. Implementeer een functie `klinkerrijk(d)` die bij een dictionary `d`, zoals gemaakt wordt in onderdeel a), een lijst van woorden bepaalt die meer klinkers bevatten dan medeklinkers. Deze lijst wordt door de functie teruggegeven.

   Voorbeeld met `d = {'aardbeiensoesje': (8, 7), 'boterkoek': (4, 5), 'eierkoek': (5, 3)}`:
   Aanroep `klinkerrijk(d)` geeft uitvoer
   `['aardbeiensoesje', 'eierkoek']`

c. Implementeer een functie `word_table(d)` die een tabel maakt (in een string) en die tabel teruggeeft. De string wordt dus **niet geprint** in de functie, maar wordt in het testprogramma door de aanroep `print(word_table(d))` geprint. De tabel moet er uitzien zoals in het onderstaande voorbeeld. Het aantal spaties tussen de kolommen mag iets afwijken van het gegeven voorbeeld, maar de tabel moet wel aan de volgende voorwaarden voldoen.
   - De kopregel moet aanwezig zijn.
   - De kolom met de woorden moet rechts uitgelijnd zijn.
   - De twee kolommen met getallen moeten ook rechts uitgelijnd zijn.
   - De kopregel staat boven de kolommen. De laatste letters van 'woord', '#klinkers' en '#letters' zijn ook rechts uitgelijnd met de woorden en getallen daaronder.

   Voorbeeld met `d = {'aardbeiensoesje': (8, 7), 'boterkoek': (4, 5), 'eierkoek': (5, 3)}`:
   Aanroep `print(klinkerrijk(d))` geeft in de console

```
          woord  #klinkers   #letters
 aardbeiensoesje          8         15
       boterkoek          4          9
        eierkoek          5          8
```

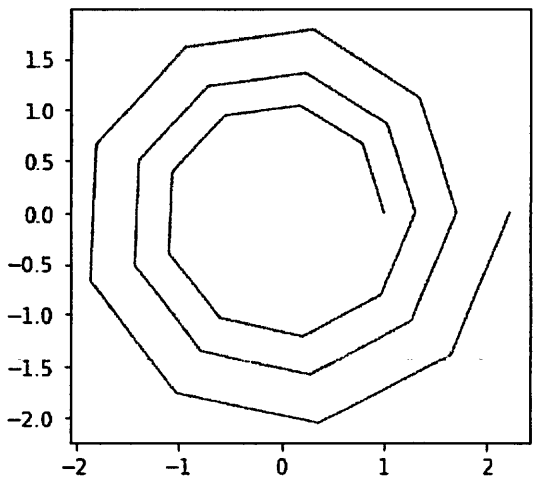
Opgave 2 **Puntenverdeling: a) 5 punten; b) 4 punten**

Gebruik bij deze opgave het bestand `A2.py` en breid dit bestand uit. Vul je naam en studienummer in, bovenaan in de file.

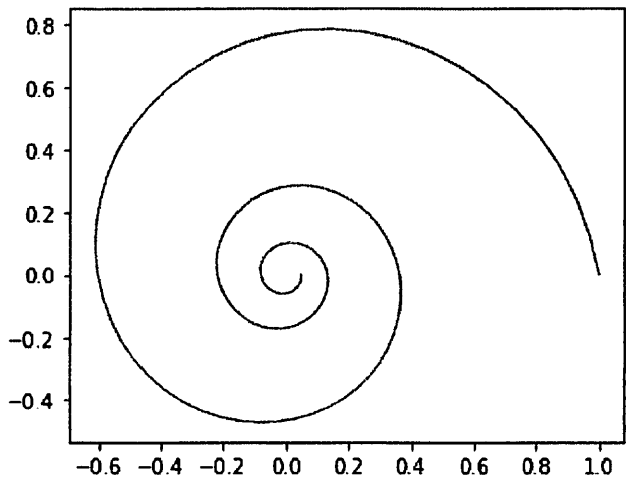
a. Implementeer een functie `Draw_spiral(n, f)` die met behulp van Matplotlib vanuit het beginpunt (1, 0) een naar buiten of naar binnen gerichte spiraal tekent rond het punt O = (0, 0) op de volgende manier:
   - De spiraal is opgebouwd uit precies `3n` lijnstukken.
   - Na `n` lijnstukken is er precies 1 maal rond het punt (0, 0) gedraaid. De volledige spiraal draait dus 3 maal rond de oorsprong. Zie ook de linker afbeelding voor de exacte manier waarop de draaiing moet plaatsvinden.
   - Ieder volgend punt van de kromme ligt op een afstand vanaf O die `f` maal zo groot is als de afstand van het vorige punt tot O.

   Voorbeelden:

   `Draw_spiral(9, 1.03)`



   `Draw_spiral(50, 0.98)`



b. Implementeer een functie `DragonCurve(start, n)`, waarin `n` de generatie is en `start` de string waarmee je begint. De functie geeft een string terug, namelijk:
   `DragonCurve ('F', 0)` geeft generatie 0 terug:
   `F`

   `DragonCurve('F',1)` geeft generatie 1 terug (`F` is vervangen door `F+G`):
   `F+G`

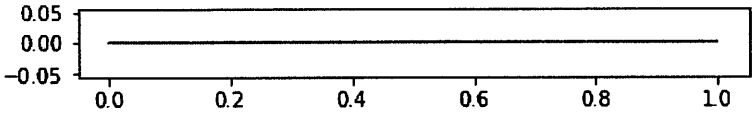
   `DragonCurve('F',2)` geeft generatie 2 terug (iedere `F` uit generatie 1 is vervangen door `F+G` en iedere `G` is vervangen door `F-G`):
   `F+G+F-G`

   `DragonCurve('F',3)` geeft generatie 3 terug (iedere `F` uit generatie 2 is vervangen door `F+G` en iedere `G` is vervangen door `F-G`):
   `F+G+F-G+F+G-F-G`

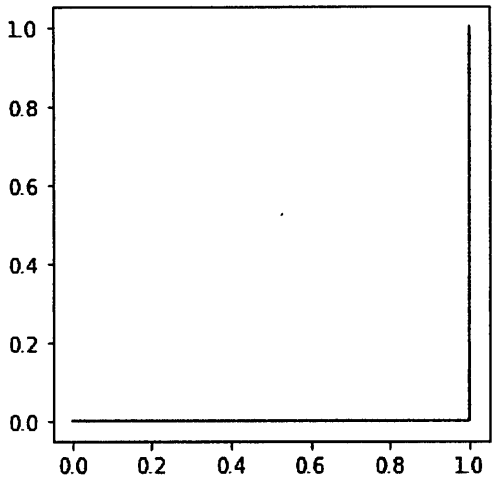
   De generatie 0 string is gelijk aan `start` en voor `start` zullen we in de aanroepen `'F'` gebruiken. De string wordt gemaakt door `n` keer **parallel**, d.w.z. tegelijkertijd, voor iedere letter `F` in de string van de vorige generatie deze letter `F` te vervangen door de string `'F+G'` en iedere `G` door `'F-G'`.

   In het testprogramma is een functie `Draw_Lsystem` gegeven. De door `DragonCurve` teruggegeven strings zijn zodanig dat bij de aanroepen `Draw_Lsystem(string, d_ang)` in het testprogramma de onderstaande figuren worden geplot:

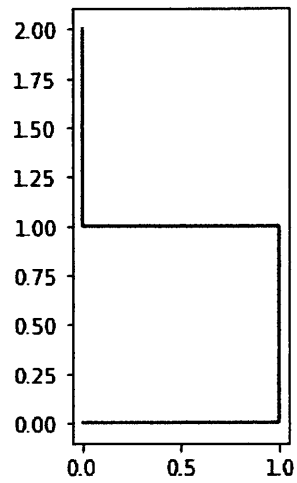
met `string = DragonCurve('F', 0)`



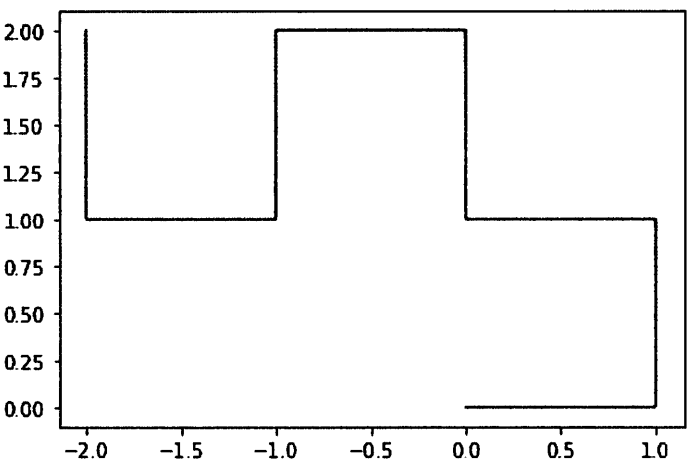
met `string = DragonCurve('F', 1)`



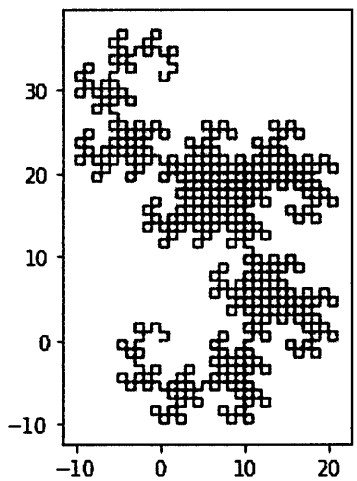
met `string = DragonCurve('F', 2)`



met `string = DragonCurve('F', 3)`



met `string = DragonCurve('F', 10)`



Opgave 3 **Puntenverdeling: a) 3 punten; b) 3 punten; c) 3 punten**

Gebruik bij deze opgave het bestand `A3.py` en breid dit bestand uit. Vul je naam en studienummer in, bovenaan in de file.

Tic tac toe (boter kaas en eieren) is een zeer eenvoudig spelletje. In deze opdracht ga je een gedeeltelijk al gegeven klasse uitbreiden zodat met het testprogramma het spel gespeeld kan worden. Gegeven is een klasse attribuut `game_start` en een constructor voor `TicTacToe` objecten. Met een al geïmplementeerde functie `play_game` kan het spel gespeeld worden, nadat je de klasse hebt uitgebreid. Echter, de aanroep van `play_game` is in het test programma uitgecommentarieerd, zodat je de methoden die je moet implementeren (eerst) los kunt testen. Als je uiteindelijk tijd over hebt, dan kun je het spel ook proberen in zijn geheel een keertje te spelen.

a. Implementeer in de klasse `TicTacToe` de methode die de `str` functie van Python overlaadt.
In het 3 bij 3 speelveld worden lege plaatsen weergegeven met een min teken en al gevulde plaatsen met een X of een O (afhankelijk van welke speler, X of O, zijn teken op dat veld heeft geplaatst). Onder het speelveld moet worden weergegeven wie er vervolgens aan zet is.

Voorbeeld (bij aanvang van het spel):

```
g = TicTacToe('X')
print(g)
```

resulteert in de uitvoer:

```
- - -
- - -
- - -
Player X to move.
```

Voorbeeld (zoals in het testprogramma):

```
print(g)
```

resulteert in de uitvoer:

```
- O -
- X -
- - X
Player O to move.
```

b. Implementeer de methode `move(self, r, c)` die een zet uitvoert, d.w.z.
   - een X of een O (afhankelijk van wie aan zet is) plaatst op positie `r,c`
   - de melding `"illegal move! Try again"` print als op positie `r,c` al een X of een O staat (en de zet dus niet uitgevoerd kan worden)
   - als de speler een legale zet heeft gedaan, de andere speler hierna aan zet laat zijn

c. Het spel heeft een winnaar als er 3 X-en op een rij, kolom of diagonaal staan of als er 3 O-s op een rij, kolom of diagonaal staan.
   Implementeer de methode `winner(self)` die bepaalt of er al een winnaar is. De methode geeft `'X'` of `'O'` terug als X of O de winnaar is en geeft `'-'` terug als er geen winnaar is.
   Je mag aannemen dat een eventuele winnaar altijd de speler is die de laatste zet heeft gedaan, maar je hoeft hier geen gebruik van te maken.

Bij juiste implementatie geeft het testprogramma onder elkaar de volgende uitvoer (hier in 3 kolommen):

```
TEST a) :
- - -
- - -
- - -
Player X to move.

================
- O -
- X -
- - X
Player O to move.

================

TEST b) :
[['-' '-' '-']
 ['-' '-' '-']
 ['-' '-' '-']]
================
After move of X,
g.move(1,2):
[['-' 'X' '-']
 ['-' '-' '-']
 ['-' '-' '-']]
================
After move of O,
g.move(3,3):
[['-' 'X' '-']
 ['-' '-' '-']
 ['-' '-' 'O']]

================
After illegal move of
X, g.move(1,2):
illegal move! Try again

[['-' 'X' '-']
 ['-' '-' '-']
 ['-' '-' 'O']]
================

TEST c) :
[['X' 'O' 'X']
 ['O' 'X' 'X']
 ['O' 'X' 'O']]
The winner is -
================
[['O' 'O' 'X']
 ['O' 'X' 'X']
 ['O' 'X' '-']]
The winner is O
================
[['X' 'O' 'X']
 ['O' 'X' '-']
 ['O' '-' 'X']]
The winner is X
================
```

Inleveren van je werk

Lees het laatste gedeelte van het instructieblad en voer alle handelingen uit om je werk in te leveren.
Je werk wordt uitsluitend aan de hand van de `.py` files **met je naam en studienummer** beoordeeld.
Controleer of er drie `.py` files met je naam en studienummer op `Private(P:)` staan.

Einde tentamen