Python Aflevering
”Hvordan man vinder øl på en bar spillet”

Christoffer Andersen
Webudvikling
Python Valgfag

Github:
https://github.com/Loopmootin/Python/tree/master/Assignments/Dice_Assignment

Til denne aflevering fik vi 3 specielle terninger i forskellige farver, med forskellige numre på, som vi skulle teste gennem et program skrevet i Python.

Spillet gik i enkelthed ud på at man skulle slå med alle 3 terninger eller 2 udvalgte og se hvilken terning som vandt.
Grunden til vi laver dette gennem et program, er at da vi i timen brugte næsten en halv time på at slå 200 gange, kan vi her gøre det over 100.000 gange på under et sekund og derved få en meget mere præcis statestik.

Jeg startede med at lave en klasse, hvori jeg kunne oprette mine terninger. Dette er måske en smule "overkill" eftersom der kun er 3 specifikke terninger, men gjorde det alligevel fordi jeg gerne ville have en toString og muligheden for at printe navnet på min terning sammen med værdierne på den.

Min terning som ses på billedet tager 3 værdier, farve, sider og en scorer, samt 3 get funktioner og 1 toString. Hvor farven og siderne måske giver sig selv, vil scoren sættes som en tom liste og senere bruges til at få tilføjet point, hver gang terningen vinder.

Billede 1

```
class Die:
    # constructor
    def __init__(self, color, sides, scores):
        self._color = color
        self._sides = sides
        self._scores = scores

    # get
    def getColor(self):
        return self._color

    def getSides(self):
        return self._sides

    def getScores(self):
        return self._scores

    # toString
    def toString(self):
        return self._color + " die has the sides " + str(self._sides)
        + " and an empty score list " + str(self._scores)
```

Derefter som ses på billede 2, lavede jeg en ny fil hvori jeg importerede Random og min nye klasse. Random skal bruges i denne opgave til at rulle med mine terninger.

Så opretter jeg mine 3 terninger som vi har fået givet i opgavebeskrivelsen og laver 2 input variabler. Disse inputs som henholdsvis vælger antallet af terninger og antallet af slag, er ikke nødvendige for opgaven, men jeg ville gerne give selve "spillerne" muligheden for at vælge, frem for at variablerne skulle ændres direkte i koden.

Billede 2

```
# imports
import random
from dice import Die

# creating the dice from Die class
red_die = Die("Red", [0, 0, 4, 4, 8, 8], [])
green_die = Die("Green", [2, 2, 3, 3, 7, 7], [])
blue_die = Die("Blue", [1, 1, 5, 5, 6, 6], [])

# calling the winner
the_winner = "Wait what, no winner??"

# input from user
numberofdice = int(input("Choose number of players(2 or 3): "))
numberofrolls = int(input("Choose number of rolls: "))
```

Hvis man vælger 2 terninger, skal de 2 spillere så, som set på billede 3, vælge en farve hver. Undervejs får man at vide hvilke sider den valgte terning har og så bliver funktionen roll_die_2() som ruller terningerne og giver point.

Billede 3

```
# chosing which function to run and what to print from the results
if(numberofdice == 2):
    choose_player1 = input("Choose which color die " +
                           "you want [Red, Green, Blue]: ")
    # calling the Die toString()
    if(choose_player1.lower() == "red"):
        print(red_die.toString())
    elif(choose_player1.lower() == "green"):
        print(green_die.toString())
    elif(choose_player1.lower() == "blue"):
        print(blue_die.toString())
    else:
        print("You must chose one of the 3 colors!")

    choose_player2 = input("Choose which color die " +
                           "you want [Red, Green, Blue]: ")
    # calling the Die toString()
    if(choose_player2.lower() == "red"):
        print(red_die.toString())
    elif(choose_player2.lower() == "green"):
        print(green_die.toString())
    elif(choose_player2.lower() == "blue"):
        print(blue_die.toString())
    else:
        print("You must chose one of the 3 colors!")

    # calling function
    roll_die_2()
```

Stadig i samme if statement, men efter roll_die_2() er blevet kaldt, bliver scorene tjekket og så får man resultatet fra begge spillere plus vinderen printet ud, som set på billede 4.

```
# checking and calling the winner
if(len(first_player.getScores()) > len(second_player.getScores())):
    the_winner = first_player.getColor() + " is the winner!"
else:
    the_winner = second_player.getColor() + " is the winner!"

print(str(first_player.getColor()) + " : " +
      str(len(first_player.getScores())))
print(str(second_player.getColor()) + " : " +
      str(len(second_player.getScores())))
print(the_winner)
```

Billede 4

Hvis spillerne derimod vælger at bruge alle 3 terninger, behøver man ikke vælge terninger, i stedet bliver funktionen roll_die_3() kaldt fra start og derefter gør den det samme som roll_die_2(), som ses billede 5.

Til sidst i det statement, hvis man ikke vælger 2 eller 3 terninger får man en fejl printet ud og spillet stopper.

```
elif(numberofdice == 3):
    # calling function
    roll_die_3()

    # checking and calling the winner
    if(len(red_die.getScores()) > len(green_die.getScores()) and
       len(red_die.getScores()) > len(blue_die.getScores())):
        the_winner = red_die.getColor() + " is the winner!"
    elif(len(green_die.getScores()) > len(red_die.getScores()) and
         len(green_die.getScores()) > len(blue_die.getScores())):
        the_winner = green_die.getColor() + " is the winner!"
    elif(len(blue_die.getScores()) > len(red_die.getScores()) and
         len(blue_die.getScores()) > len(green_die.getScores())):
        the_winner = blue_die.getColor() + " is the winner!"
    else:
        the_winner = "Something went very wrong here!"

    print(red_die.toString())
    print(green_die.toString())
    print(blue_die.toString())
    print(str(red_die.getColor()) + " : " + str(len(red_die.getScores())))
    print(str(green_die.getColor()) + " : " + str(len(green_die.getScores())))
    print(str(blue_die.getColor()) + " : " + str(len(blue_die.getScores())))
    print(the_winner)

else:
    print("Game doesn't work with " + str(numberofdice) + " dice")
```

Billede 5

Funktionen roll_die_2() sætter 3 først variabler i toppen, "i" bliver brugt som counter i et loop, "num" som bliver sat ud fra mængden af slag spillerne har valgt og 2 globale variabler som er spiller 1 og spiller 2. Derefter tjekker den om spiller 1 har valgt en valid terning, rød, grøn eller blå (se billede 6).

Billede 6

```
# if using 2 colors
def roll_die_2():
    i = 0
    num = numberofrolls
    # setting global to reuse same variable outside function
    global first_player, second_player

    # setting first players color
    if(choose_player1.lower() == "red"):
        first_player = red_die
    elif(choose_player1.lower() == "green"):
        first_player = green_die
    elif(choose_player1.lower() == "blue"):
        first_player = blue_die
    else:
        print("Something went wrong")
```

Derefter laver den det samme tjek på spiller 2, men udover det sikrer den sig også at det ikke er samme terning som spiller 1 har valgt (se billede 7).

```
# setting second players color and checking if not the same as first
if(choose_player2.lower() == "red" and choose_player1.lower() != "red"):
    second_player = red_die
elif(choose_player2.lower() == "green"
        and choose_player1.lower() != "green"):
    second_player = green_die
elif(choose_player2.lower() == "blue"
        and choose_player1.lower() != "blue"):
    second_player = blue_die
else:
    print("You need to choose a different color than player 1!")

# amount of rolls is decided by user when program is running
while i < num:
    roll1 = random.choices(first_player.getSides())[0]
    roll2 = random.choices(second_player.getSides())[0]
    if(roll1 > roll2):
        first_player.getScores().append("+")
    else:
        second_player.getScores().append("+")
    i = i + 1
```

Billede 7

Til sidst bliver et loop kørt det antal gange som spillerne har valgt, hvor den hver gang vælger et tilfældigt nummer fra hver terning og spilleren som har slået det højeste nummer får et point i sin score liste (se billede 7).

Funktionen roll_die_3() er i grove træk en simpel version af roll_die_2(), eftersom den ikke behøver tjekke hvilke terninger som er blevet valgt og kan derfor springe direkte i while loopet (se billede 8).

```python
# if using all 3 colors
def roll_die_3():
    i = 0
    num = numberofrolls

    # amount of rolls is decided by user when program is running
    while i < num:
        roll1 = random.choices(red_die.getSides())[0]
        roll2 = random.choices(green_die.getSides())[0]
        roll3 = random.choices(blue_die.getSides())[0]
        if(roll1 > roll2 and roll1 > roll3):
            red_die.getScores().append("+")
        elif(roll2 > roll1 and roll2 > roll3):
            green_die.getScores().append("+")
        elif(roll3 > roll1 and roll3 > 2):
            blue_die.getScores().append("+")
        i = i + 1
```

Billede 8

```
Choose number of players(2 or 3): 2
Choose number of rolls: 20000
Choose which color die you want [Red, Green, Blue]: Red
Red die has the sides [0, 0, 4, 4, 8, 8]
Choose which color die you want [Red, Green, Blue]: Blue
Blue die has the sides [1, 1, 5, 5, 6, 6]
Red : 8904
Blue : 11096
Blue is the winner!
```

Billede 9

```
Choose number of players(2 or 3): 3
Choose number of rolls: 20000
Red die has the sides [0, 0, 4, 4, 8, 8]
Green die has the sides [2, 2, 3, 3, 7, 7]
Blue die has the sides [1, 1, 5, 5, 6, 6]
Red : 8057
Green : 5986
Blue : 5957
Red is the winner!
```

Billede 10

På billede 9 og 10 kan man se outputtet fra spillene med henholdsvis 2 og 3 terninger.

Og i følge resultaterne, gennem mange tusinde af slag, vinder Rød hver gang i et spil med 3 terninger, men Blå slår Rød i et spil med 2, hvor at Grøn slår Blå og Rød slår Grøn.