USE CASE:

Design an application for Simulating Random Rolling Dice Using NumPy

The Dice Roll Simulation may be performed by selecting a random number between 1 and 6, which we can achieve using the random package in Python. we'll show you how to make a Dice Roll Simulator using NumPy

Game Rules:

Player will throw a dice and the output will be added to the current scores of the player (initially equal to zero). If the dice had output 6 then it would be thrown again (one dice: 6, one more turn: 4. Then the total would be 6+4 = 10). The sum of total will be throwing id the total score of the player with a particular number of trials.

The formal code structure is as:

player() class: This player class will be storing player name, its age and its color code for the game. There is a method called score which stores the attribute score associated with the player. Another method getscore() for calling the value of score stored.

game() class: This class represents the game and take input as the player (class type) and the number of trails. The method __init__() defines the attribute associated with the class type game. The method gaming() is consisting of the whole game.

dice() function: The function dice just give output as a random value from the number set [1,2,3,4,5,6]. This uses random.choice() function for performing this task.

```
import random
import numpy
def roll():
    return random.choice([1,2,3,4,5,6])

class player(object):
    def __init__(self, name, age, colour):
        self.name = name
        self.age = age
        self.colour = colour

    def score(self, score):
        self.score = score

    def getscore(self):
        return self.score

    def getname(self):
        return self.name
    def __str__(self):
        return 'NAME: ' + self.name + '\nCOLOUR: ' + self.colour + '\nSCORE: ' +
str(self.score)

class game(object):
    def __init__(self, playr, trails):
        self.trails = trails
        self.playr = playr
```

```
    def gaming(self):
        throw = 0
        score = 0
        for i in range(self.trails):
            throw = roll()
            if throw == 6:
                throw = throw + roll()
            score = throw + score
        return score

    def __str__(self):
        return self.playr.getname() + str(self.score)
tri = 123




zack = player('zack', 24, 'green')
johny = player('johny', 25, 'yellow')
kina = player('kina', 14, 'red')
usher = player('usher', 13, 'blue')
print("-----------LETs PLAy THIs GAMe----------\n" )
#zack.score(88)
#print(zack)
zackscr = game(zack, tri)
johnyscr = game(johny, tri)
kinascr = game(kina, tri)
usherscr = game(usher, tri)
```

```
scr = []
scr.append(zackscr.gaming())
scr.append(johnyscr.gaming())
scr.append(kinascr.gaming())
scr.append(usherscr.gaming())


scrsort = sorted(scr)
for el in scrsort:
   print(el)


zack.score(scr[0])
usher.score(scr[3])
kina.score(scr[2])
johny.score(scr[1])

#players = []
#players.append(zack.getscore())
#players.append(usher.getscore())
#players.append(kina.getscore())
#players.append(johny.getscore())

#for el in players:
#    print('--', el)
#print(scr[0])
print(zack, '\n')
```

```
print(kina, '\n')
print(johny, '\n')
print(usher, '\n')
```

OUTPUT

----------LETs PLAy THIs GAMe----------

485

489

491

525

NAME: zack

COLOUR: green

SCORE: 485

NAME: kina

COLOUR: red

SCORE: 491

NAME: johny

COLOUR: yellow

SCORE: 489

NAME: usher

COLOUR: blue

SCORE: 525