The Pig Game

***The Pig Game*** merupakan sebuah permainan dadu yang bisa dimainkan oleh 2 orang atau lebih. Setiap pemain akan mendapatkan kesempatan untuk mengocok dadu dan berapa pun angka yang muncul dari dadu tersebut pemain tersebut akan menyimpannya di dalam ***scoreboard*** mereka. Jika salah satu pemain bisa mengumpulkan poin hingga 100, maka mereka akan memenangkan pertandingan tersebut dan jika mereka mendapatkan angka 1 dari dadu tersebut maka skor yang sudah mereka kumpulkan akan kembali ke 0 dan giliran mereka berhenti.

Pig Game di dalam bahasa pemrograman Python.

import random

* This line of code imports the **random** module, which is a built-in Python module.
* The **random** module provides functions and methods for generating random numbers and performing randomization operations in Python.
* In this context, the **random** module is used to simulate randomness in the game, particularly for rolling a die.

def dice():

    min\_dice\_value = 1

    max\_dice\_value = 6

    dice = random.randint(min\_dice\_value, max\_dice\_value)

    return dice

* This code defines a Python function named **dice()** that simulates rolling a six-sided die.
* The function sets the minimum and maximum possible values for the die using the variables **min\_dice\_value** (1) and **max\_dice\_value** (6).
* It uses the **random.randint()** function from the **random** module to generate a random integer between the **min\_dice\_value** and **max\_dice\_value**. This simulates the roll of a six-sided die, and the result is stored in the **dice** variable.
* The function then uses the **return** statement to return the value of the rolled die when the function is called. This allows you to use the function to get a random die roll.

while True:

    player\_number = input("How many player will be participating in this game of pig: ")

    if player\_number.isdigit():

        player\_number = int(player\_number)

        if 2 <= player\_number <= 4:

            break

        else:

            print("The requirement to play the game is 2 - 4 people!")

    else:

        print("Invalid Input!")

print(f"Players registered : {player\_number} \n")

* This code segment is responsible for determining the number of players who will participate in the Pig Game.
* It uses a **while** loop with the condition **while True** to repeatedly ask the user for the number of players until a valid number (between 2 and 4) is provided.
* The **input()** function receives user input, and **player\_number** initially stores this input as a string.
* It checks if the input is a valid number using the **isdigit()** method.
* If the input is a valid number, it converts it to an integer using the **int()** function.
* It further checks if the number of players is within the range of 2 to 4. If so, it breaks out of the **while** loop.
* If the input is not a valid number or is outside the specified range, it provides feedback to the user via **print()** messages.
* Once a valid number of players is determined, it displays a message with the number of players registered.

winning\_score = 10

scoreboard = [0 for \_ in range(player\_number)]

* The code initializes two variables: **winning\_score** and **scoreboard**.
* **winning\_score** is set to 10, which means that the game will end once a player reaches a score of 10 or according to your liking.
* **scoreboard** is a list that will store the scores of each player. It's created using list comprehension and is initialized with zeros for each player. The number of zeros in the list is based on the value of the **player\_number** variable.

while max(scoreboard) < winning\_score:

* The code uses a **while** loop to simulate the game's progression.
* The condition **max(scoreboard) < winning\_score** checks whether the maximum score among all players (as determined by **max(scoreboard)**) is less than the **winning\_score**. This condition ensures that the game continues as long as no player has reached or exceeded the winning score.

for player\_index in range(player\_number):

        print(f"Player {player\_index + 1} turn!")

        print(f"Your current score is {scoreboard[player\_index]}")

        current\_score = 0

* The code uses a **for** loop to simulate the turns of each player in the Pig game. The loop iterates through **player\_index**, which refers to each player specified in the **player\_number** variable.
* It prints a message to announce whose turn it is to play. Adding 1 to **player\_index** is to adjust for Python's zero-based indexing.
* It also displays the current score of the player by accessing the **scoreboard** list using **player\_index**.
* **current\_score** is a placeholder variable used to track the score for the current player. It's initialized to 0 because all players start with a score of 0.

while True:

            user\_prompt = input("Would you like to roll: Type 'y' to continue! ").lower()

            if user\_prompt != "y":

                break

            dice\_value = dice()

            if dice\_value == 1:

                print("Oops! You rolled a 1. Your turn ends.")

                current\_score = 0

                break

            else:

                current\_score += dice\_value

                print("You rolled a ", dice\_value)

            print("Your total score is ", current\_score)

* This code segment uses a **while** loop with the condition **while True** to simulate a player's turn in the Pig Game.
* The loop continues as long as the player chooses to roll the die (by entering 'y') and doesn't roll a 1.
* It uses the **input()** function to prompt the player for their choice, and the input is converted to lowercase to ensure it's not case-sensitive.
* If the player enters anything other than 'y,' the loop breaks, ending their turn.
* If the player decides to roll the die (enters 'y'), the **dice()** function is called to obtain a random die value, which is stored in the **dice\_value** variable.
* If the **dice\_value** is 1, the player's turn ends, and their current score is reset to 0.
* If the **dice\_value** is not 1, it's added to the player's current score, and the value rolled is displayed.
* The player's total score is also displayed after each roll.

scoreboard[player\_index] += current\_score

print(f"Player {player\_index + 1} score is {scoreboard[player\_index]} \n")

* This code segment updates the player's score in the **scoreboard** list based on their **current\_score**.
* It uses the player's index (**player\_index**) from the **for** loop to identify the player whose turn just ended and adjust their score accordingly.
* The player's **current\_score** is added to their existing score in the **scoreboard** using the **+=** operator.
* It then uses the **print()** function to display a message indicating the player's updated score.

result = max(scoreboard)

winning\_player = scoreboard.index(result)

print(f"Player {winning\_player + 1} is won with the score of {result}")

* This code segment is responsible for determining the winner of the Pig game based on the scores in the **scoreboard** list.
* It uses the **max()** function to find the highest score among the players' scores stored in the **scoreboard** list. This value is stored in the **result** variable.
* It then uses the **index()** method of the list (in this case, the **scoreboard** list which is already stored in result) to find the position of the highest score in the list. This position represents the winning player.
* Finally, it uses the **print()** function to announce the winning player by displaying their player number and score.