
INFORMATICS INSTITUTE OF TECHNOLOGY

# Foundation Certificate for Higher Education

**Module code:** DOC 334 Introduction to Programming in Python-P2

**Module Leader:** Mr. Nishan Saliya Harankahawa

**Assignment Number:** 1

**Assignment type:** Individual

**Title:** 20x2 Board Game in Python

**Student Name:** Tharusha Jayawardane

**Student ID:** 20220910

# Acknowledgements

I would like to express my deepest gratitude to my tutorial lecturer Ms Aloka for providing me with invaluable guidance, support and feedback throughout the course of this project. Your expertise and commitment to my work have been instrumental in shaping the outcome of this report.

I would also like to extend my appreciation to Mr Nishan, Mr Chathura and Ms Keerthiga for their help and support in lectures, which was crucial to the completion of this project. Your assistance and insights have been invaluable.

Lastly, I would like to express my gratitude to my family and friends for their unwavering support and encouragement throughout this project. Their love and encouragement gave me the motivation to persevere through the challenging times.

Thank you all for your contributions to the completion of this DOC 334 coursework report.

# Table of Contents

# 1. Introduction

This report is an overview of a Python program written with the purpose of implementing a code-cracking game concept called.

## 1.1 The main objective of the game Concept

The purpose of this assignment is to create and put into action a straightforward board game ("20 x 2") that can be played by a human player and a computer. A 20-block board is used in the game, and players roll dice to move their pawns. In order to win, a player must be the first to the 20th block or pass it. The player's piece can only enter the board after rolling a six on the dice, and the game features two black holes that penalize the player if they are landed upon.

## 1.2 Python Program Statements

Welcome to the game of 20 x 2! In this game, you will be playing against the computer on a board consisting of 20 blocks. The objective of the game is to be the first player to reach or pass the 20th block. But beware, there are black holes on blocks 7 and 14 that will send you back to block 1 if you land on them.

To start the game, you must roll a 6 on the dice. Once the game has started, each dice roll will determine how many blocks you move, with a dice value of 6 moving you 3 blocks, a dice value of 4 moving you 2 blocks, a dice value of 5 moving you 2 blocks (with 0.5 being neglected), and a dice value of 1 meaning no movement at all.

The game board will be displayed in the command console, with the first row of blocks representing your progress and the second row representing the computer's progress. Once a player reaches or passes the 20th block, the game will end and a text file with the game session details will be saved using the format YYYY_M_D_H_M.txt.

## 2 How does the program work (Algorithm)

1. Define variables and initialize them
   - Set the board size to 20 and create an empty board with dimensions 2x20
   - Define the player and computer positions, their number of moves, and other necessary variables
   - Define the black holes on the board

2. Define a function to display the board
   - Use a pretty table to create the board with the current positions of the player and computer

3. Define a function to roll the dice and return the number of blocks to move
   - Use the random module to generate a number between 1 and 6
   - Return the value of the dice divided by 2 (since each move takes two blocks)

4. Define a function for the human player's turn
   - Prompt the user to roll the dice and wait for their input
   - Call the rollDice() function to get the number of blocks to move
   - Move the player's position based on the number of blocks and check for black holes or the end of the board
   - Update the board with the new position of the player
   - Check if the player has won the game

5. Define a function for the computer's turn
   - Call the rollDice() function to get the number of blocks to move
   - Move the computer's position based on the number of blocks and check for black holes or the end of the board
   - Update the board with the new position of the computer
   - Check if the computer has won the game

6. Define a function to run the game loop

- Use a while loop to continue playing until a player win
- Alternate between the human player and the computer for each turn
- Display the board after each turn

7. Define a function for the welcome interface
- Prompt the user to enter their name
- Display a welcome message and the rules of the game

8. Call the welcome() function to start the game.

9. Call the functions to continue the game.

10. After a player win display game session details and write them into a text file and save.

# 3. The program's source code

```
import random
import math
import datetime
import time

from prettytable import PrettyTable


# Define variables to keep track of the players' positions
diceRoll=0
playerPosition = 1
computerPosition = 1
total_movesPlayer = 0
total_movesComputer=0
moveBlocks=0
diceValue=0
playerCount=0
computerCount=0
player_BH=0
computer_BH=0

blackHoles=[7,14]
boardSize=20


# Define a variable to keep track of the current player
turn=True


# Define the game board
board = [[0 for _ in range(20)] for _ in range(2)]
board[0][6] = 'O'
board[0][13] = 'O'
```

```
board[1][6] = 'O'
board[1][13] = 'O'

row1=["H"," "," "," "," "," "," "," ","O"," "," "," "," "," "," "," ","O"," "," "," "," "," "," "," "]
row2=["C"," "," "," "," "," "," "," ","O"," "," "," "," "," "," "," ","O"," "," "," "," "," "," "," "]

def gameBoard():
    board=PrettyTable()
    board.field_names =["
","1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20"]
    board.add_row(row1, divider=True)
    board.add_row(row2)

    print(board)


# Define to Roll the dice and determine the number of blocks to move
def rollDice():
    diceRoll = random.randint(1, 6)
    return diceRoll


# Define Human player's rolling
def humanPlayer_roll():
    print("Now you are playing !")
    time.sleep(1)
    input("Press enter key to start : ")
    print("The dice is rolling wait...")
    time.sleep(2)
    diceValue=rollDice()
    if diceValue == 6:
        print("\t\t\tYou rolled a 6! Now you are starting the game!")
        time.sleep(1)
        return True
    else:
```

```
        print(f"\t\t\tYou rolled a {diceValue}, Try again.")
        time.sleep(1)
        print(f"\t\t\tYou can\'t start the game with {diceValue}")
        return False


# Define Computer player's rolling
def computerPlayer_roll():
    time.sleep(2)
    print('Now It\'s computer\'s turn wait...')
    time.sleep(1)
    print("\t\t\tComputer is rolling the Dice.....")
    time.sleep(2)
    diceValue = rollDice()
    if diceValue == 6:
        print("\t\t\tComputer rolled a 6! Now the computer is starting the game!")
        return True
    else:
        print(f"\t\t\tComputer rolled a {diceValue}")
        print(f"\t\t\tComputer can\'t start the game with {diceValue}")
        return False


# Define Human player's position changing
def human_pMove(playerPosition,playerDiceValue):
    global player_BH
    row1[playerPosition]= " "
    currentPosition = playerPosition + (playerDiceValue // 2)
    if currentPosition in blackHoles:
        currentPosition = 1
        player_BH += 1
        print('\t\t\tOops!,You hit a black hole.')
    elif currentPosition > boardSize:
        currentPosition = boardSize
```

```
    row1[currentPosition]= "X"
    return currentPosition


# Defining Computer player's position changing
def computer_pMove(playerPosition, playerDiceValue):
    global computer_BH
    row2[playerPosition]= " "
    currentPosition = playerPosition + (playerDiceValue // 2)
    if currentPosition in blackHoles:
        currentPosition = 1
        computer_BH += 1
        print('\t\t\tOops!,Computer hit a black hole.')
    elif currentPosition > boardSize:
        currentPosition = boardSize
    row2[currentPosition]= "X"
    return currentPosition


# Welcome interface
def welcome():
    name=input("Enter your name : ").capitalize()
    print("---------------------- Hi !", name,",Welcome to the BoardGame (20 x 2) ----------------------
")
    print("\n")
    print("Better luck for a 6 !")
    print("\n")


#Load the game loop
welcome()

player=humanPlayer_roll()
print()
computer=computerPlayer_roll()
```

```
while True:
    if player:
        # player's turn
        print()
        playerCount += 1
        print(f"This is your turn -- {playerCount} ")
        input("\t\t\tPress enter key to roll the dice...")
        time.sleep(2)
        diceValue = rollDice()
        blocks=(diceValue // 2)
        print("\t\t\tYou rolled ", diceValue)
        time.sleep(1)
        print(f"\t\t\tYou moving {diceValue // 2} blocks forward")
        playerPosition = human_pMove(playerPosition, diceValue)
        print("\t\t\tyour position:", playerPosition)
        time.sleep(1)
        gameBoard()
        if playerPosition == boardSize:
            print("\t\t\tCongratulations, you won!")
            break
    else:
        print()
        player = humanPlayer_roll()


    if computer:
        # computer' turn
        print()
        computerCount += 1
        print(f'This is computer\'s turn ---- {computerCount}.')
        print('\t\t\tComputer is rolling the Dice...')
        time.sleep(2)
        diceValue = rollDice()
```

```
        blocks=(diceValue // 2)
        print("\t\t\tComputer rolled a", diceValue)
        time.sleep(1)
        print(f"\t\t\tComputer moving {diceValue // 2} blocks forward")
        computerPosition = computer_pMove(computerPosition, diceValue)
        print("\t\t\tComputer position:", computerPosition)
        time.sleep(1)
        gameBoard()

        if computerPosition == boardSize:
            print("\t\t\tSorry,Try again,The computer won this time.")
            break
    else:
        print()
        computer = computerPlayer_roll()


print()
print("Your details")
time.sleep(1)
print(f"\t\tTotal moves : {playerCount}")
print(f"\t\tTotal black hole hits : {player_BH} ")
if playerPosition == boardSize:
    print("\t\tyou won the game! (:")
else:
    print("\t\tYou loss the game,Try again ):")

print()
print("Computer details,")
time.sleep(1)
print(f"\t\t\tTotal moves : {computerCount}")
print(f"\t\t\tTotal black hole hits : {computer_BH}")
if computerPosition == boardSize:
    print("\t\t\tComputer Won the game ")
```

```
else:
    print("\t\t\tComputer loss the game")



# Save game details to a file
time = datetime.datetime.now()
file = time.strftime("%Y_%m_%d_%H_%M.txt")
with open(file, "w+") as file:
    file.write("Your details,")
    file.write(f"\n\t\tTotal moves : {playerCount}")
    file.write(f"\n\t\tTotal black hole hits : {player_BH} ")
    if playerPosition == boardSize:
        file.write("\n\t\tyou won! (:")
    else:
        file.write("\n\t\tYou defeated ,Try once again ):")

    file.write("\nComputer details")
    file.write(f"\n\t\tTotal moves : {computerCount}")
    file.write(f"\n\t\tTotal black hole hits : {computer_BH}")
    if computerPosition == boardSize:
        file.write("\n\t\tComputer Won the game ")
    else:
        file.write("\n\t\tComputer loose the game")
```

Copy & paste text from the VS Code interface. (Original code)

# 4. Screenshots of the program

Welcome screen &
the human player starting the game.

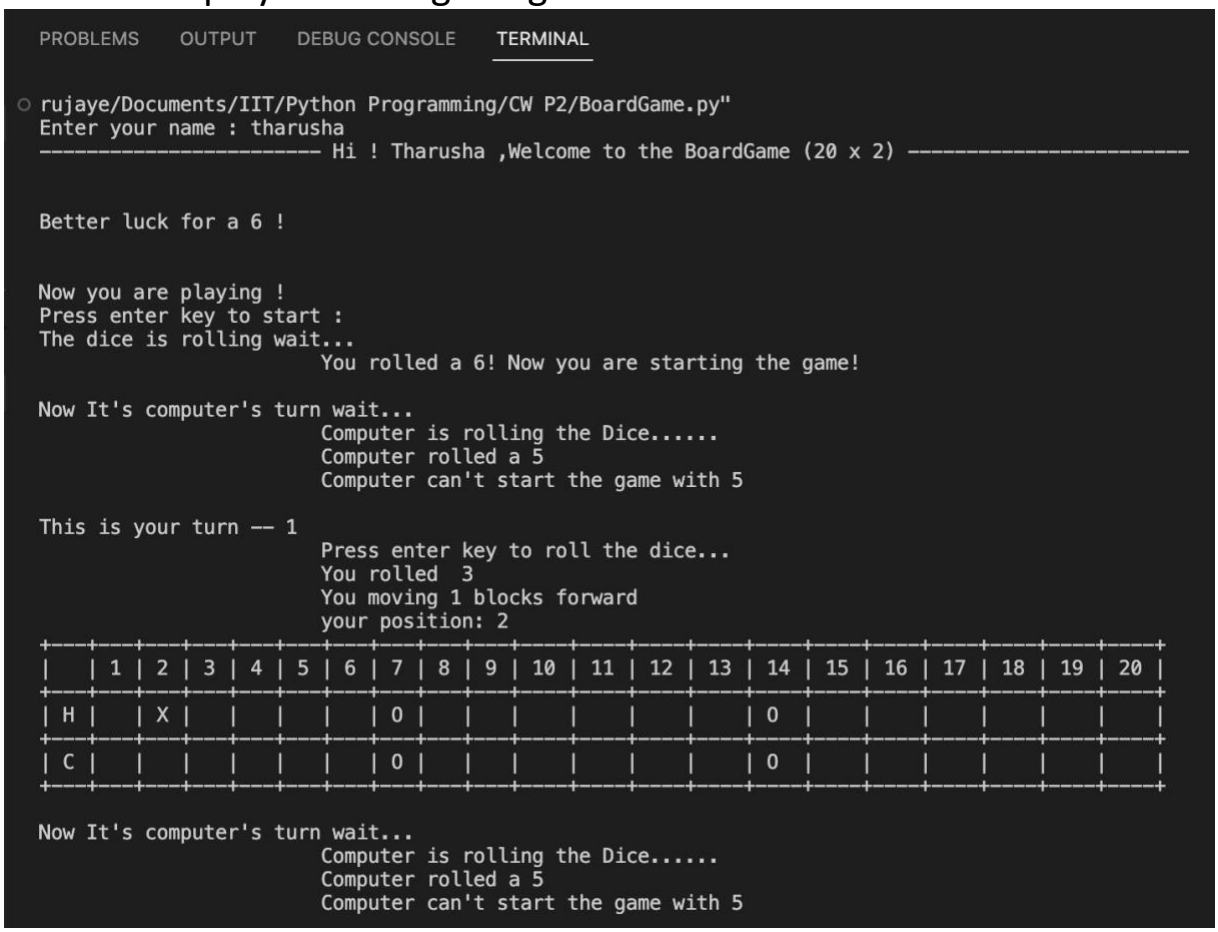


*Figure 1Welcome screen & the human player starting the game*

- The human player is starting the game after rolling the dice to 6.
- But the computer player can’t start the game because the dice is not rolled to 6.

The human player hits a black hole and returns to 1st position.

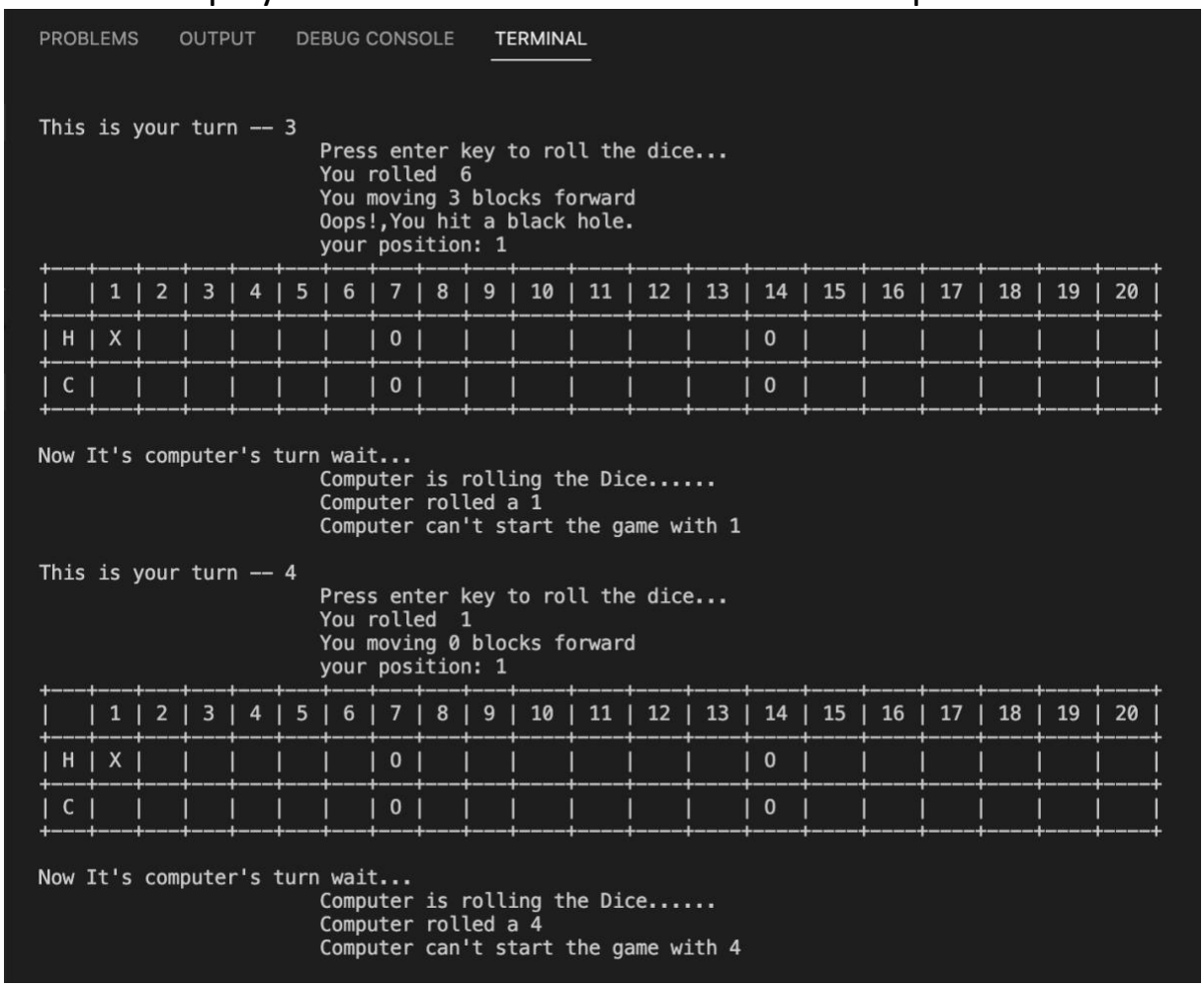


*Figure 2The human player hits a black hole and returns to 1st position*

- The computer player still can't start the game.

The computer is starting the game now.

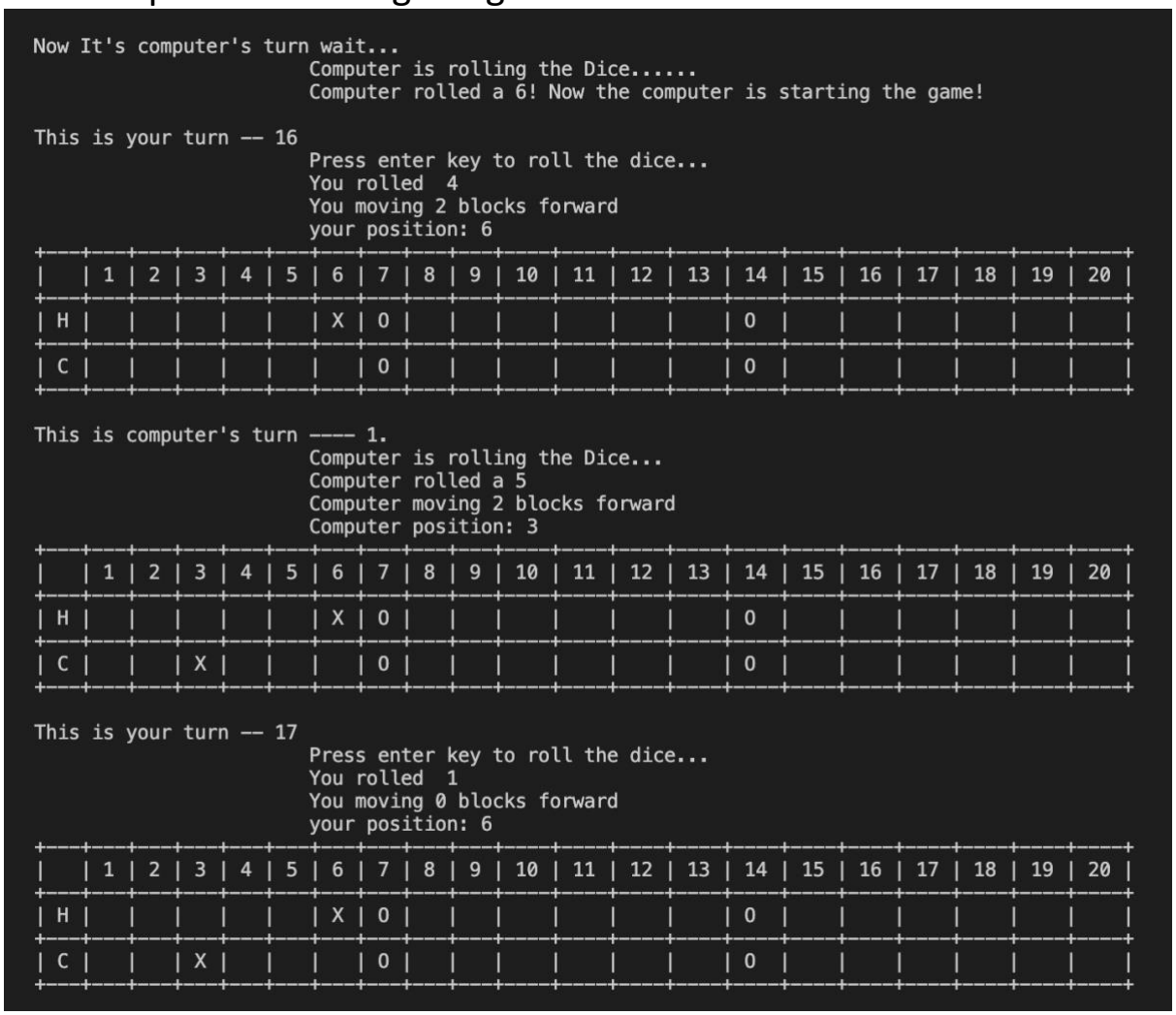


*Figure 3The computer player starting the game*

- Now the computer starting the game and the human player is in the 17th turn.

The computer hits a black hole and returns to the 1st position.


*Figure 4The computer hits a black hole and returns to the 1st position*

- The computer player hits a black hole, and the Human player is now at the 28th turn.

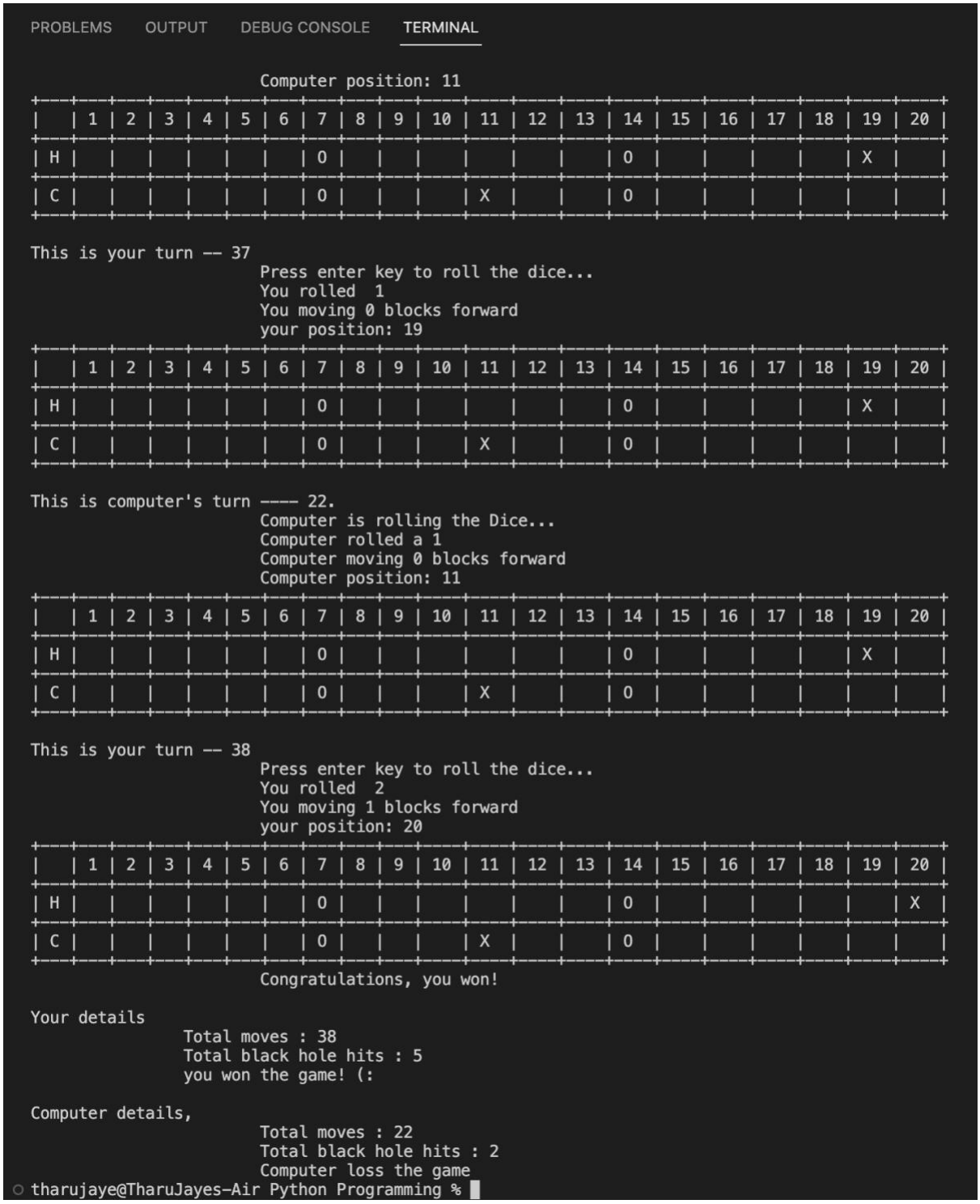


*Figure 5Game over and human player wins*

- The human player wins the game and displays the game session details at the end.

# 5. Test case

| Test case ID | Test Case Description | Pre-Conditions | Test Steps | Test Data | Expected Result | Post Condition | Actual Result | Status |
|---|---|---|---|---|---|---|---|---|
| 1 | Entering name | Need to type a name | 1. Enter the name | Name | Display the interface | Start the game | Displaying the interface | Pass |
| 2 | Dice rolling | Need to press enter key | 1.Press enter key And calling the function | Number of dice | Display dice rolled number | Display the player position | Displaying the rolled number | Pass |
| 3 | Game start | The dice rolled number must be 6 | 1.Press enter key to dice roll | Rolled dice number 6 | Start the game | Showing the position in game board | Starting the game | Pass |
| 4 | Game board with positions | Either 1 Player need to start the game | Calling functions | 1. Current positions of players<br><br>2.Black holes | Display Game board with player positions in a Pretty table | Displaying X in the current position of players | Displaying Game board with player positions in a Pretty table | Pass |
| 5 | Stop game after a player wins the game | Player position need to be 20 | Calling function | 1. Player position<br>2. Total moves<br>3. Total black hole hits | Display the winner and other details | Save game session to a text file | Displaying the winner and other details | Pass |
| 6 | Save the game session to a text file | Game needs to stop after a player win | Calling function | 1. Player position<br>2. Total moves<br>3. Total black hole hits | Save the game session details to a text file and save it | Closing the game | Saving the game session details to a text file and save it | Pass |

# 6. Conclusion

The program is a Python script implementing a board game that can be played between a human and the computer. The board game consists of a 20x2 board, and the players move their pieces based on the result of rolling a dice. The first player to reach the last block on the board wins. The game also has "black holes" that can cause the player's piece to move back to the start.

The script uses the prettytable module to display the game board, and the random module to simulate rolling the dice. It defines several functions to handle various aspects of the game, such as rolling the dice, moving the player's piece, and determining the winner.

The program begins by asking the user for their name and welcoming them to the game. It then starts the game loop, alternating between the human and computer players until one of them reaches the end of the board. During each turn, the player rolls the dice, moves their piece, and the game board is updated to reflect the current positions of the players. If a player's piece lands on a black hole, they are moved back to the start. The game also keeps track of the number of black holes each player has encountered.

At the end of the game, the program displays the winner and the number of turns it took to win.

The terminal was used to execute the program, which was entirely coded in VS Code.