**CREATE AN**

**YOUR OWN ADVENTURE**

**GAME**

NAME : BHARATHI KANNAN V

DATE OF SUBMITTED : 30-05-2024

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TOPICS** | **PAGE NO** |
| 1 | AIM OF THE PROJECT | 3 |
| 2 | PROBLEM STATEMENT | 3 |
| 3 | PROJECT DESCRIPTION | 3 |
| 4 | FUNCTIONALITICS | 4 |
| 5 | INPUT VERSATILITY AND ERROR HANDLING | 4 |
| 6 | CODE IMPLEMENTATION DESCRIPTION | 5 - 8 |
| 7 | RESULTS AND OUTCOME | 9 |
| 8 | CONCLUSION | 9 |

**INDEX**

**1. Aim of the Project:**

The aim of the Pythonland Adventure game is to provide an interactive and engaging text-based gaming experience where players embark on a quest to retrieve the stolen gold of the kingdom of Pythonland from the evil Lord Evilton. Through a series of choices and challenges, players navigate through the narrative-driven storyline, making decisions that affect the outcome of the game.

**2. Problem Statement:**

The kingdom of Pythonland is in peril as the King's gold has been stolen by the nefarious Lord Evilton. The player must step into the shoes of a brave adventurer and undertake the challenge of infiltrating Lord Evilton's castle, overcoming obstacles, and ultimately reclaiming the stolen treasure.

**3. Project Description:**

The Pythonland Adventure game is a text-based adventure implemented in Python. It begins with an introduction to the storyline, setting the stage for the player's quest. The player is then presented with choices at various decision points, each leading to different outcomes that determine the progression of the game.

**4. Functionalities:**

* **Acceptance of Challenge:**

Players decide whether to accept the quest to retrieve the stolen gold.

* **Decision Point:**

Players make choices at different stages of the game, influencing the direction of the narrative.

* **Outcome Resolution:**

The game resolves the consequences of the player's choices, leading to various endings based on their actions.

* **Play Again Option:**  After completing the game, players have the option to play again or exit.

**5. Input Versatility with Error Handling and Exception Handling:**

The game offers versatile input options, allowing players to interact using simple text inputs. Robust error handling ensures a smooth user experience by gracefully managing invalid inputs and guiding the player to enter valid responses. Exception handling is implemented to catch unexpected errors, preventing the game from crashing and ensuring uninterrupted gameplay.

**6. Code Implementation:**

The provided Python script defines the playGame() function, which orchestrates the gameplay. It utilizes while loops and if-else statements to handle player choices and game progression. Each decision point presents the player with clear options, and their input determines the subsequent actions and outcomes in the game.

**DESCRIPTION:**

First thing we want to do is give the player a little introduction to the game. You can type out your introduction in a multiline string, and then we want to print it to the player:

def playGame():

introduction = '''Hello and welcome to the Pythonland Adventure!

The kingdom of Pythonland needs your help! The evil Lord Evilton has stolen the King's gold,

and we need your help to get it back!'''

print(introduction)

We want the player to be able to start the game over at the end if they want to play again, for this we will use a **while loop.** We can keep track of if the user wants to keep playing in a variable stillPlaying. We will initialize it to True, and then change it to False once the user decides to stop playing. We will use this as our condition in our while loop.

**Here is our while loop:**

stillPlaying = True

while stillPlaying:

Now we can start giving the player choices! For this we will use the input() function that is already built into python. Here is an example of how we can use it:

stillPlaying = True

while stillPlaying:

choice = input("Do you accept the challenge? (y/n) ")

What we do next will depend on what the player picks, this is where we need to start using conditional blocks. Since there are only two options for this one, we can just use an **if-else** block.

if choice == "y":

print("You approach Lord Evilton's castle, but there are guards at the door. How do you want to proceed?")

choice = input("1: Throw a rock in the bushes to distract them \n2: Try to fight through them ")

We are currently coding the story for **if** the user inputs ‘y’ to continue on their adventure. We haven’t gotten to the else statements yet!

If you made a chart of your storyline, now is a good time to reference it! We can repeat the same process for both the if and else blocks, depending on what is happening in the story.

Now they will need to choose between option one 1: Throw a rock in the bushes to distract them or 2: Try to fight through them.

**Here’s what we can code next:**

if choice == "1":

print("The guards are distracted long enough for you to run into the castle! \n")

print("Inside the doorway you enter a room with three doors. Which door do you choose?")

choice = input("1: The red door, 2: The black door, 3: The blue door ")

Now we are giving the user three choices between a red, black and blue door. So in this case we need to use two **elif** blocks as well.

**Here’s what we can add to our code now:**

if choice == "1":

print("The red door leads directly into a pit of lava.")

elif choice == "2":

print("The black door leads to the throne room! Lord Evilton sits on his throne guarding his stolen gold.")

choice = input("Do you 1: Fight or 2: Run ")

if choice == "1":

print("You fight Lord Evilton and win! You grab the gold and escape to safety! The King thanks you.")

else:

print("You run from the castle to safety.")

elif choice == "3":

print("The blue door leads to the gold room, you grab the gold and escape to safety! The King thanks you.")

This section of code presents the player with three door options. The player’s choice determines what message is displayed.

If the player chooses the red door (option 1), the message “The red door leads directly into a pit of lava” is displayed, and the game is over.

If the player chooses the black door (option 2), the message “The black door leads to the throne room! Lord Evilton sits on his throne guarding his stolen gold” is displayed. The player is then asked whether they want to fight Lord Evilton or run away. If they choose to fight, they win the game and get the message “You fight Lord Evilton and win! You grab the gold and escape to safety! The King thanks you.” If they choose to run away, they get the message “You run from the castle to safety.”

If the player chooses the blue door (option 3), the message “The blue door leads to the gold room, you grab the gold and escape to safety! The King thanks you” is displayed, and the player wins the game.

**Now let’s add our final else statement**

else:

print("You are too weak, and you lose the battle.")

The last line else: print("You are too weak, and you lose the battle.") is a part of an if statement that checks if the player chooses option 2 to fight through the guards instead of option 1 to throw a rock to distract them.

If the player chooses option 2 and tries to fight through the guards but fails, the program executes the code block under the else statement, which prints the message “You are too weak, and you lose the battle.” This message indicates to the player that they have failed to pass the guards and have lost the game.

In summary, the else statement is executed when the player’s choice does not match the conditions specified in the previous if statement, in this case, when the player chooses to fight through the guards but fails.

**Finishing The Game:**

The last thing we need to do is make it so that the player can choose to keep playing or end the game once the story is finished. We can add a block like this at the end:

playAgain = input("Do you want to keep playing? (y/n) ")

if playAgain == "n":

print("Thanks for playing")

stillPlaying = False

So we will keep looping until the player chooses to end the game, otherwise they can just keep playing!

**Here is what the completed game looks like:**

def playGame():

introduction = '''Hello and welcome to the Pythonland Adventure!

The kingdom of Pythonland needs your help! The evil Lord Evilton has stolen the King's gold,

and we need your help to get it back!'''

print(introduction)

stillPlaying = True

while stillPlaying:

choice = input("Do you accept the challenge? (y/n) ")

if choice == "y":

print("You approach Lord Evilton's castle, but there are guards at the door. How do you want to proceed?")

choice = input("1: Throw a rock in the bushes to distract them \n2: Try to fight through them ")

if choice == "1":

print("The guards are distracted long enough for you to run into the castle! \n")

print("Inside the doorway you enter a room with three doors. Which door do you choose?")

choice = input("1: The red door, 2: The black door, 3: The blue door ")

if choice == "1":

print("The red door leads directly into a pit of lava.")

elif choice == "2":

print("The black door leads to the throne room! Lord Evilton sits on his throne guarding his stolen gold.")

choice = input("Do you 1: Fight or 2: Run ")

if choice == "1":

print("You fight Lord Evilton and win! You grab the gold and escape to safety! The King thanks you.")

else:

print("You run from the castle to safety.")

elif choice == "3":

print("The blue door leads to the gold room, you grab the gold and escape to safety! The King thanks you.")

else:

print("You are too weak, and you lose the battle.")

playAgain = input("Do you want to keep playing? (y/n) ")

if playAgain == "n":

print("Thanks for playing")

stillPlaying = False

playGame()

**7. Results and Outcomes:**

* The Pythonland Adventure game provides an immersive and enjoyable gaming experience, allowing players to embark on a thrilling quest to save Pythonland.
* With engaging narrative elements and interactive decision-making, players are immersed in the world of the game, making impactful choices that shape the storyline.
* The game's versatile input handling and robust error management ensure a seamless and enjoyable gameplay experience for users of all skill levels.

**8. Conclusion:**

The Pythonland Adventure game demonstrates the potential of text-based gaming to provide compelling and immersive experiences for players. Through engaging storytelling, interactive decision-making, and effective input handling, the game captivates players and offers an enjoyable journey through the kingdom of Pythonland.

**THANK**

**YOU !**