Creating a Text based Dragon Game in Python

In this guide, you will learn how to create a simple text-based adventure game in Python. This game involves choosing between two caves, one containing a treasure and the other containing a dragon. If you choose the correct cave, you find the treasure; otherwise, you encounter the dragon.

Creating the Main Game Script

Let's start writing our game step by step. We'll explain each part of the code as we go along.

Importing Necessary Modules

First, we need to import some tools that will help us build our game.

```python
import random
```

Listing 1: Importing necessary modules

Explanation:

- `import random`: This line tells Python to use the `random` module, which helps us generate random numbers. We'll use this to randomly decide which cave has the treasure.

Getting the User's Choice

Next, we need to ask the player which cave they want to enter.

```python
userInput: str = input(”Which cave do you want to enter? (1 or 2)”)
```

Listing 2: Asking for user input

Explanation:

- `userInput: str`: This part creates a place to store the player's answer. The `str` means we expect the answer to be a string (text).
- `input("Which cave do you want to enter? (1 or 2)")`: This part shows a message asking the player to choose a cave and waits for their answer.

Choosing a Random Cave

Now, let's make the computer choose a random cave to hide the treasure.

```
number: int = random.randint(1, 2)
```

Listing 3: Choosing a random cave

Explanation:

- `number: int`: This creates a place to store a number. The `int` means we expect a whole number (integer).

- `random.randint(1, 2)`: This part makes the computer choose a random number, either 1 or 2. This number will represent the cave with the treasure.

Comparing Choices

Now, we need to compare the player's choice with the computer's random choice.

```
if userInput == str(number):
    print(”You have entered the cave with the treasure!”)
else:
    print(”You have entered the cave with the dragon!”)
```

Listing 4: Comparing choices

Explanation:

- `if userInput == str(number)`: This checks if the player's choice (userInput) is the same as the computer's choice (number). We use `str(number)` to convert the number to a string so we can compare it with the player's answer. You cannot compare a String with a Integer.

- `print("You have entered the cave with the treasure!")`: This prints a message saying the player found the treasure.

- `else`: If the player's choice does not match the computer's choice, this part runs.

- `print("You have entered the cave with the dragon!")`: This prints a message saying the player found the dragon.

Putting It All Together

Now, let's put everything together into a main function.

```
def main():
    userInput: str = input(”Which cave do you want to enter? (1 or 2)”)
    number: int = random.randint(1, 2)
    print(enter_cave(userInput, number))
```

Listing 5: Defining the main function

Explanation:

- `def main()`: This starts the definition of our main function. A function is a set of instructions that we can run by calling its name.
- The next Two lines are the same as we wrote before. They ask for the player's choice and generate a random number.
- The last line is printing out the result. Inside the print statement we are calling another function called `def enter_cave(userInput, number)` which we will discuss below.

Creating the enter_cave Function

We need another function to compare the player's choice with the computer's choice and show the appropriate message and picture.

```python
def enter_cave(userInput: str, number: int) -> str:
    if userInput == str(number):
        return "You have entered the cave with the treasure!"
    else:
        return "You have entered the cave with the dragon!"
```

Listing 6: Defining the enter_cave function

Explanation:

- `def enter_cave(userInput: str, number: int) -> str`: This starts the definition of our `enter_cave` function. It takes two inputs: `userInput` (a string) and `number` (an integer). It will return a string.
- The next lines inside the function are the same as we wrote before. They compare the choices and return the appropriate message.

Running the Game

Finally, we need to run the main function when the script is executed.

```python
if __name__ == "__main__":
    main()
```

Listing 7: Running the main function

Explanation:

- `if __name__ == "__main__"`: This line checks if the script is being run directly (not imported as a module).
- `main()`: This runs the main function.

Your `main.py` file should now look like this:

```python
import random
import render

def main():
    userInput: str = input(”Which cave do you want to enter? (1 or 2)”)
    number: int = random.randint(1, 2)
    print(enter_cave(userInput, number))

def enter_cave(userInput: str, number: int) -> str:
    if userInput == str(number):
        render.treasure()
        return ”You have entered the cave with the treasure!”
    else:
        render.dragon()
        return ”You have entered the cave with the dragon!”

if __name__ == ”__main__”:
    main()
```

Listing 8: Complete cave_game.py file

Adding Extra Flavour

To make the game more engaging, we'll add visual representations of the treasure and the dragon using ASCII art. We'll store these in a separate file called `render.py`.

1. Create a new file named `render.py`.
2. Add the following functions to display the treasure and dragon.

Your `render.py` file should now look like this:

```
def dragon():
    print("""
                                                                    ____________
                                 (`-..________....---''  ____..._.-`
                                  \\`._______.._,.---'''     ,'
                                  ; )`.      __..-'`-.      /
                                 / /     _.-' _,.;;._ `-._,'
                                / /   ,-' _.-'  //   ``--._``._
                              ,','_.-' ,-' _.- (( =-    -. `-._`-._____
                            ,;.''__..-'   _..--.\\.--'````--.._``-.`-._`.
             _          |\,' .-''        ```-'`---'`-...__,._  ``-.`-.`-.`.
  _     _.-,'(__)\__)\-'' `     ___  .          `     \      `--._
,',)---' /|)          `     `      ``-.   `     /     /     `     `-.
\_____--.  '`  `               __..-.  \     . (   < _...-----..._   `.
 \_,--..__. \\ .-`.\----'';``,..-.__ \  \      ,`_. `.,-'`--'`---''`.  )
           `.\`.\  `_.-..' ,'   _,-..'  /..,-''(, ,' ; ( _______`___..'__
                   ((,(,__(    ((,(,__,'  ``'-- `'`.(\  `.,..______   SSt
                                                      ``________..._``--.__
                                                                  )-.,'
                                                                   \\,'
    """)

def treasure():
    print(
    """
                            _.--.
                        _.-'_:-'||
                    _.-'_.-::::'||
               _.-:'_.-::::::'  ||
             .'`-.-:::::::'     ||
            /.'`;|:::::::'      ||_
           ||   ||::::::'     _.;._'-._
           ||   ||:::::'  _.-!oo @.!-._'-.
           \'.  ||:::::.-!()oo @!()@.-'_.|
            '.'-;|:.-'.&$@.& ()$%-'o.'\||
              `>'-.!@%()@'@_%-'_.-o _.|'||
               ||-._'-.@.-'_.-' _.-o  |'||
               ||=[ '-._.-\/.-'    o |'||
               || '-.]=|| |'|      o  |'||
               ||      || |'|        _| ';
               ||      || |'|    _.-'_.-'
               |'-._   || |'|_.-'_.-'
            jgs '-._'-.|| |'  `_.-'
                    '-.||_/.-'
    """)
```

To use the functions we created in `render.py` we need to import the file into our main file :

```
import random
import render  # This will allow us to use the functions we created
```

Listing 9: Importing necessary modules

Now we need to specify where and when the treasure and dragon functions from `render.py` need to be called :

```
if userInput == str(number):
    render.treasure() # This will display the treasure on success
    print(”You have entered the cave with the treasure!”)
else:
    render.dragon() # This will display the dragon on failure
    print(”You have entered the cave with the dragon!”)
```

Listing 10: Comparing choices

Go on, run the game see if it works .

Try It Yourself

What if we wanted to have three options instead of two, where only the correct one leads to the treasure and the other two lead to the dragon? Let's think about how to code this.

First, consider the steps you need to take:

1. Update the user input prompt to ask for a choice between three caves.

2. Generate a random number between 1 and 3 to determine the cave with the treasure.

3. Modify the comparison logic to handle three choices.

Here's how you can do it step by step:

Step 1: Updating the User Input Prompt

We need to update the user input to allow for three choices:

```
userInput: str = input(”Which cave do you want to enter? (1, 2, or 3)”)
```

Listing 11: Updating user input prompt

Step 2: Generating a Random Number

Now, generate a random number between 1 and 3:

```
number: int = random.randint(1, 3)
```

Listing 12: Choosing a random cave

Step 3: Modifying the Comparison Logic

The comparison logic stays stays the same:

```python
def enter_cave(userInput: str, number: int) -> str:
    if userInput == str(number):
        render.treasure()
        return "You have entered the cave with the treasure!"
    else:
        render.dragon()
        return "You have entered the cave with the dragon!"
```

Listing 13: Comparing choices with three options

Explanation:

Despite the number of choices increasing, the comparison logic would stay the same, as no matter the number of choices the user is given, only the number that matches the randomly generated number `number: int = random.randint(1, 3)` will return true for the `if userInput == str(number):` and all other numbers will return false and hence get the dragon.

Now, integrate these changes into the main function:

```python
def main():
    userInput: str = input("Which cave do you want to enter? (1, 2, or 3)")
    number: int = random.randint(1, 3)
    print(enter_cave(userInput, number))

if __name__ == "__main__":
    main()
```

Listing 14: Updated main function

Complete updated code:

```python
import random
import render

def main():
    userInput: str = input("Which cave do you want to enter? (1, 2, or 3)")
    number: int = random.randint(1, 3)
    print(enter_cave(userInput, number))

def enter_cave(userInput: str, number: int) -> str:
    if userInput == str(number):
        render.treasure()
        return "You have entered the cave with the treasure!"
    else:
        render.dragon()
        return "You have entered the cave with the dragon!"

if __name__ == "__main__":
    main()
```

Listing 15: Complete updated cave_game.py file

By following these steps, you have successfully expanded the game to include three choices.

Running the Game

To run the game, simply execute the `main.py` script. Open your terminal or command prompt, navigate to the directory where you saved the files, and type:

```
python main.py
```

Listing 16: Running the game

Follow the prompt to choose a cave and see if you find the treasure or encounter the dragon!

Congratulations! You've created a simple yet fun text-based adventure game in Python. Feel free to modify and tinker with it. Enjoy coding!