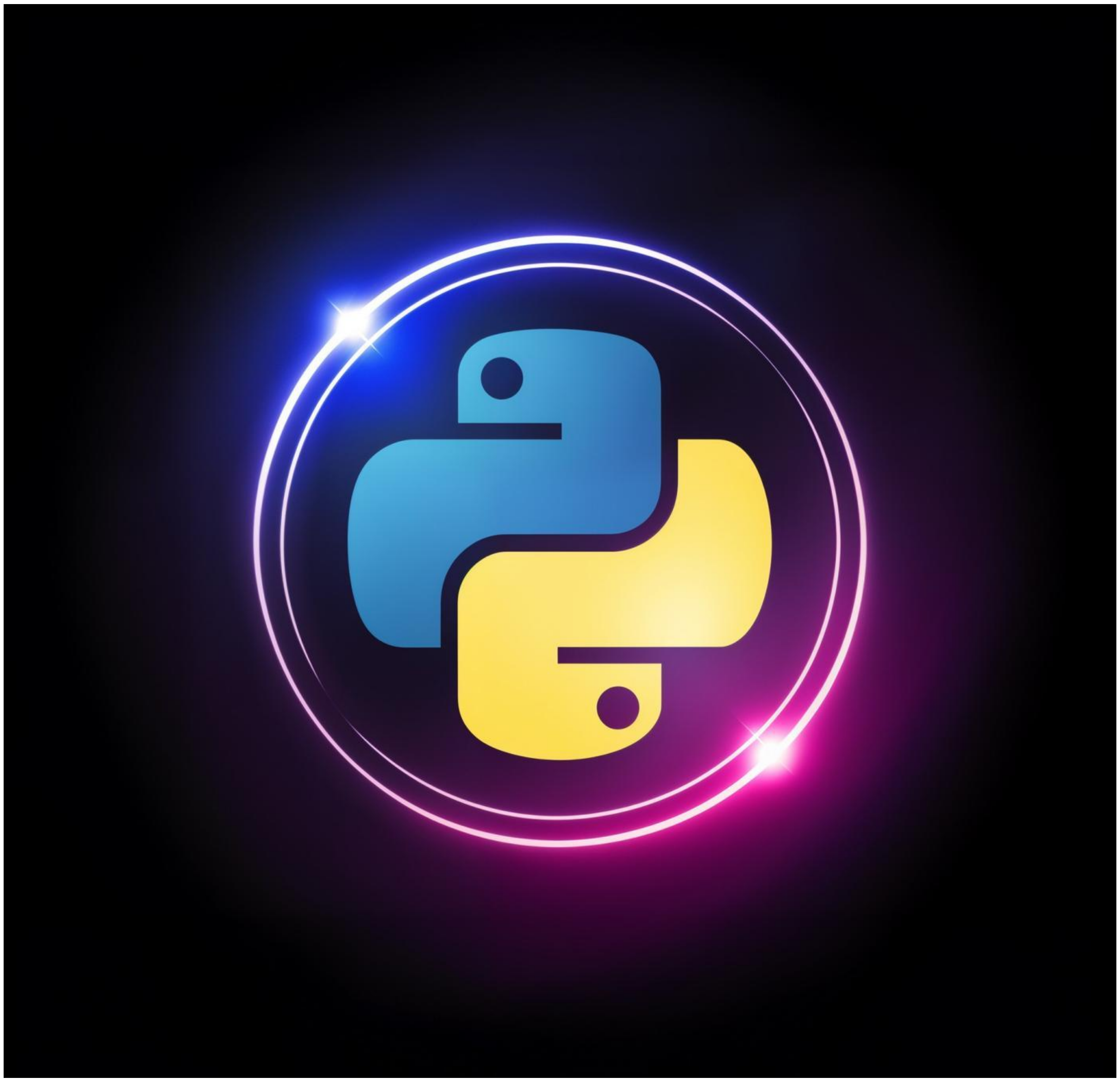


Table of Content

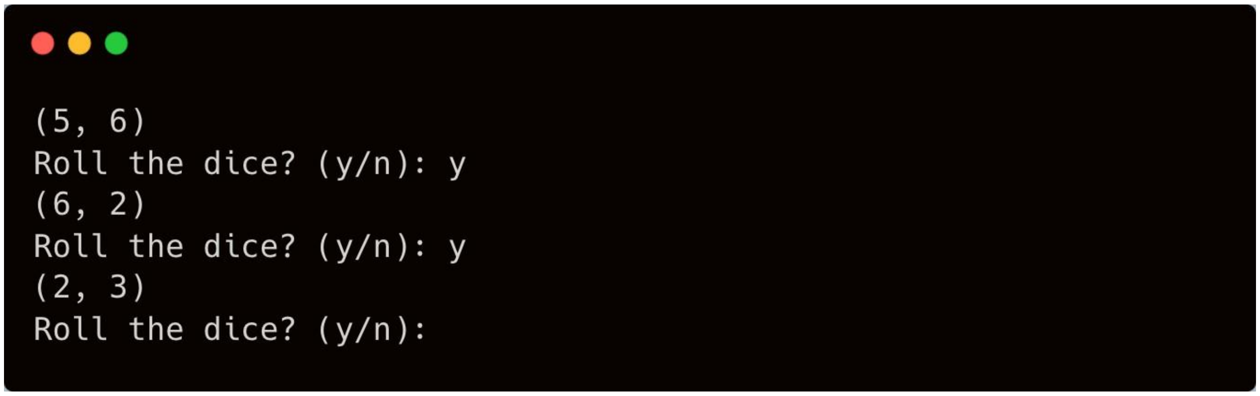
Project List

Below is a list of all the projects included in this course, along with their difficulty levels. Use this as a guide to track your progress and see how your skills are advancing as you move through the course.

Project	Difficulty
Dice Rolling Game	2/10
Number Guessing Game	3/10
Rock, Paper, Scissors Game	3/10
QR Code Generator	3/10
Currency Converter	3/10
Quiz Game	3/10
Tic Tac Toe Game	4/10
Todo List Application	4/10
Simple Text Editor	4/10
Pig Dice Game	4/10
Cows and Bulls Game	4/10
Password Strength Checker	4/10
Password Generator	5/10
Word Guessing Game	5/10
Slot Machine Game	5/10
ATM Simulation	6/10

Dice Rolling Game

Write a program that simulates rolling a pair of dice. Each time the program runs, it should randomly generate two numbers between 1 and 6 (inclusive), representing the result of each die. The program should then display the results and ask if the user would like to roll again.



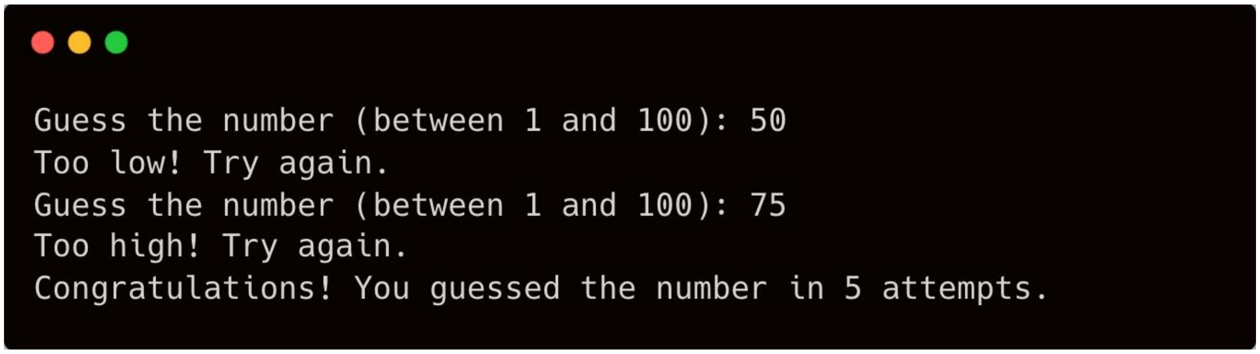
```
(5, 6)
Roll the dice? (y/n): y
(6, 2)
Roll the dice? (y/n): y
(2, 3)
Roll the dice? (y/n):
```

Optional Enhancements

- Modify the program so the user can specify how many dice they want to roll.
- Add a feature that keeps track of how many times the user has rolled the dice during the session. This will require a counter that increments each time the dice are rolled.

Number Guessing Game

Write a program to have the computer randomly select a number between 1 and 100, and then prompt the player to guess the number. The program should give hints if the guess is too high or too low.



```
Guess the number (between 1 and 100): 50
Too low! Try again.
Guess the number (between 1 and 100): 75
Too high! Try again.
Congratulations! You guessed the number in 5 attempts.
```

Optional Enhancements

- Allow the user to specify the minimum and maximum values for the number range before the game starts. This gives the player more control over the difficulty level.

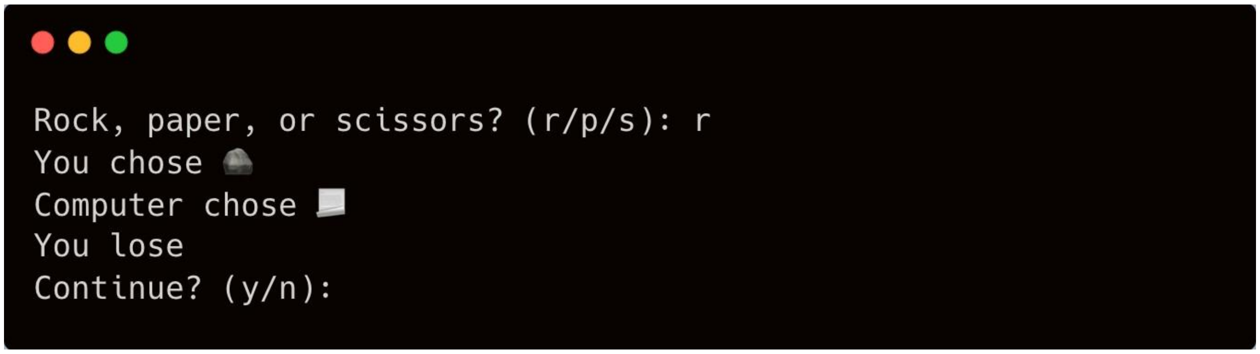
- Implement a feature that limits the number of guesses a player can make. If the player runs out of attempts, the game should end, and the correct number should be revealed.

- Add a feature that keeps track of the fewest attempts it took to guess the number correctly. The program should display this "best score" at the end of each game.

Rock, Paper, Scissors Game

Write a program to simulate a game of Rock, Paper, Scissors.

The game will prompt the player to choose rock, paper, or scissors by typing 'r', 'p', or 's'. The computer will randomly select its choice. The game will then display both choices using emojis and determine the winner based on the rules.



```
Rock, paper, or scissors? (r/p/s): r
You chose 🪨
Computer chose 📃
You lose
Continue? (y/n):
```

Optional Enhancements

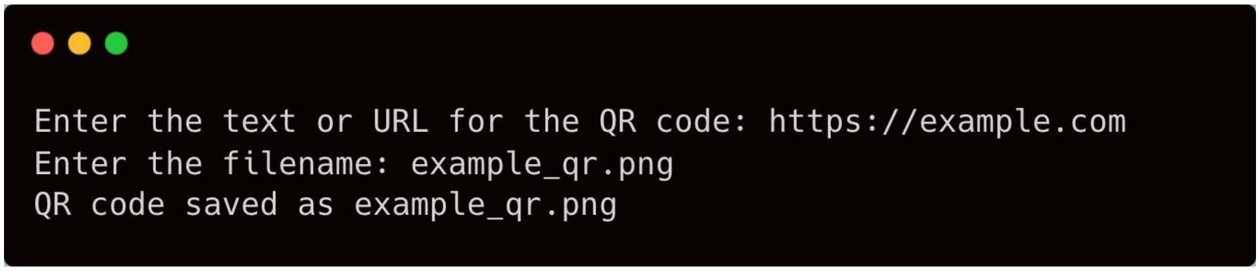
- Modify the game so that the first player (or computer) to win two out of three rounds is declared the overall winner. This adds a competitive aspect to the game.

- Keep a tally of how many times the player wins, loses, or ties with the computer. Display these statistics at the end of the game.

- Add an option for two players to play against each other, taking turns to input their choices. The program should then determine the winner based on their inputs.

QR Code Generator

Write a program to generate a QR code based on user input, such as text or a URL. The QR code should be saved as an image file that can be scanned with a smartphone.



```
Enter the text or URL for the QR code: https://example.com
Enter the filename: example_qr.png
QR code saved as example_qr.png
```

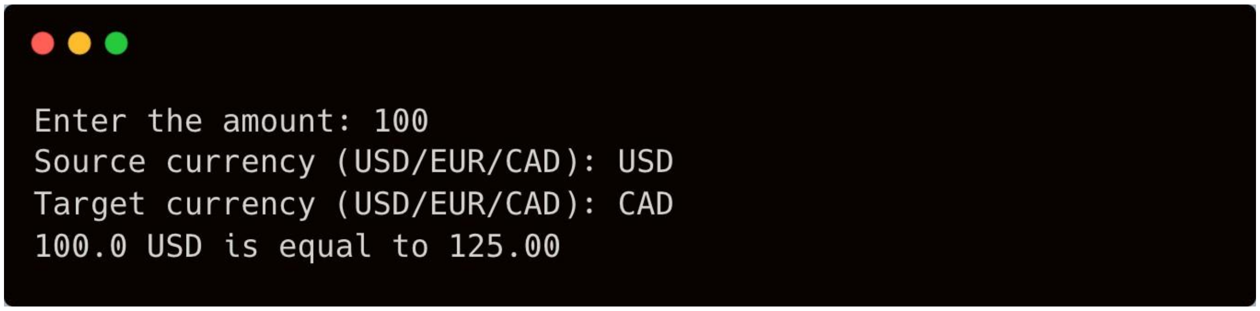
Optional Enhancements

- Add options for the user to choose the color of the QR code. This will allow users to generate QR codes that match their style or branding.

- Implement a feature that lets the user generate multiple QR codes at once by providing a list of URLs or texts. Each QR code should be saved with a unique filename.

Currency Converter

Write a program to convert an amount of money from one currency to another using fixed exchange rates. The user inputs the amount and selects the currencies for conversion.



```
Enter the amount: 100
Source currency (USD/EUR/CAD): USD
Target currency (USD/EUR/CAD): CAD
100.0 USD is equal to 125.00
```

Optional Enhancements

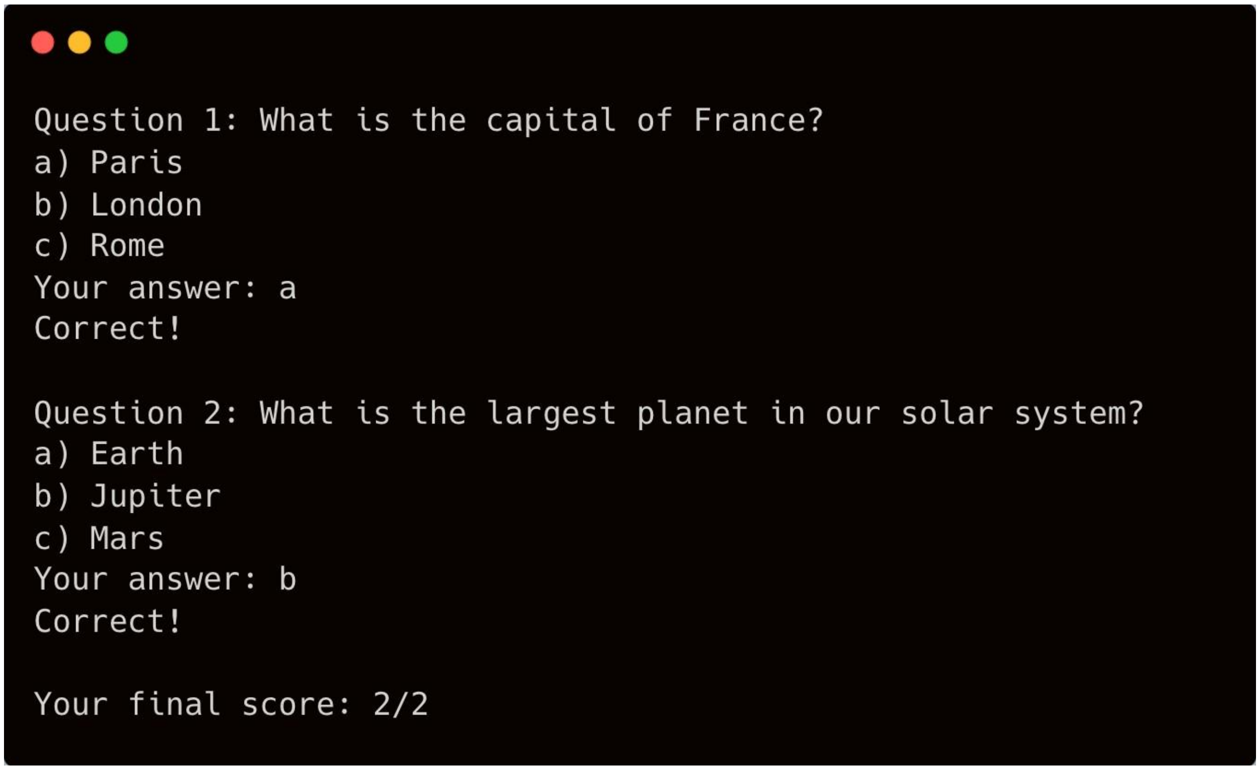
- Modify the program so that the user can see the equivalent amount in several different currencies at the same time. For example, converting 100 USD to EUR, CAD, and GBP all at once.

- Expand the list of available currencies for conversion. This might involve adding more fixed exchange rates to the program.

- Keep a history of the most recent conversions made during the session and display this history at the end of the program.

Quiz Game

Write a program to create a quiz where the player answers multiple-choice questions. The program should evaluate the player's answers and provide a score at the end.



```
Question 1: What is the capital of France?
a) Paris
b) London
c) Rome
Your answer: a
Correct!

Question 2: What is the largest planet in our solar system?
a) Earth
b) Jupiter
c) Mars
Your answer: b
Correct!

Your final score: 2/2
```

Optional Enhancements

- Allow the player to choose from different categories of questions, such as history, science, or geography.

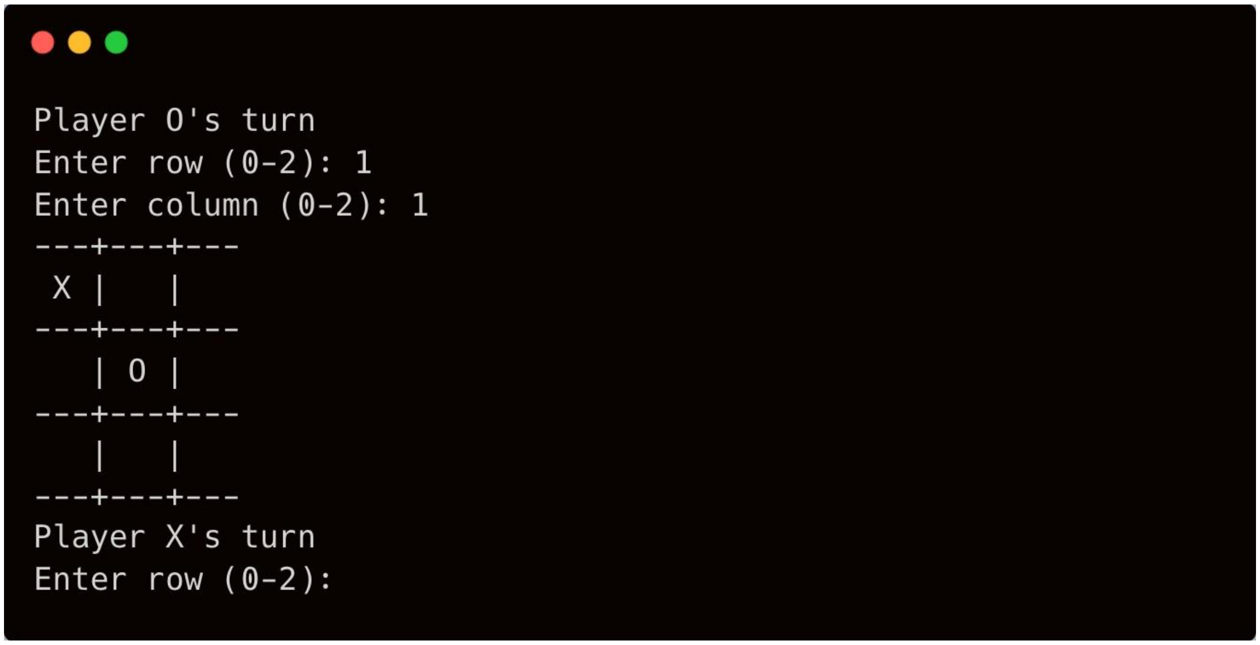
- Implement different difficulty levels by varying the complexity of the questions.

- Modify the game to import questions from a CSV or JSON file, allowing for easy updates and additions to the quiz content.

Tic Tac Toe Game

Write a program to create a classic Tic Tac Toe game. Two players take turns marking Xs and Os on a 3   3 grid. The first player to align three of their marks either vertically, horizontally, or diagonally wins the game.

The board is printed after each move, with colored marks to distinguish between the players. The game checks for a winner after each move and ends when a player wins or when the board is full, resulting in a draw.



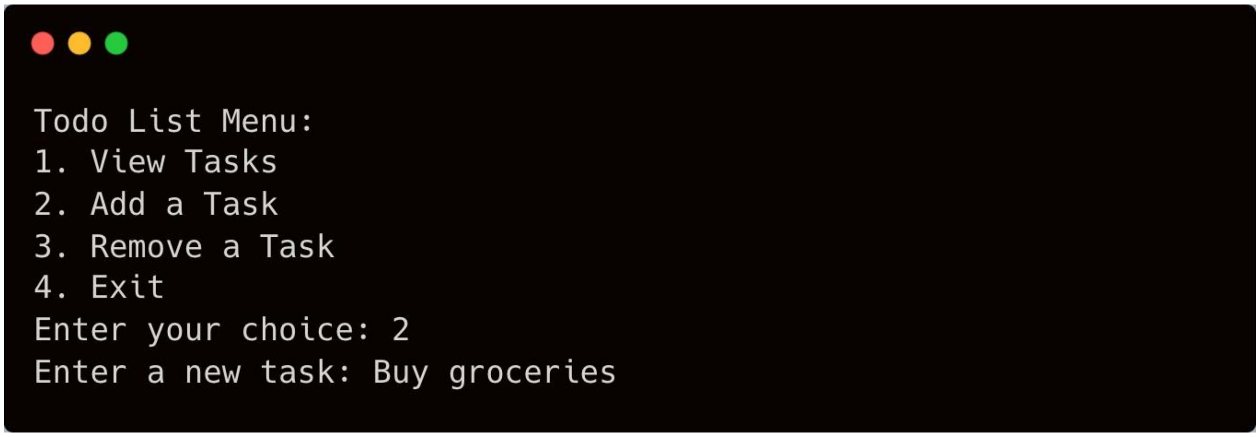
```
Player O's turn
Enter row (0-2): 1
Enter column (0-2): 1
---+---+---
 X |   |
---+---+---
   | O |
---+---+---
   |   |
---+---+---
Player X's turn
Enter row (0-2):
```

Optional Enhancements

- Add a scoring system that tracks the number of games won by each player across multiple rounds.
- Allow players to start a new game without restarting the program.
- Modify the game to allow different board sizes (e.g., 4 4 or 5 5 for a more challenging experience.

To-Do List Application

Write a program to create a simple To-Do List application. The program allows users to add tasks, view the current list of tasks, and remove tasks once they are completed.



```
Todo List Menu:
1. View Tasks
2. Add a Task
3. Remove a Task
4. Exit
Enter your choice: 2
Enter a new task: Buy groceries
```

Optional Enhancements

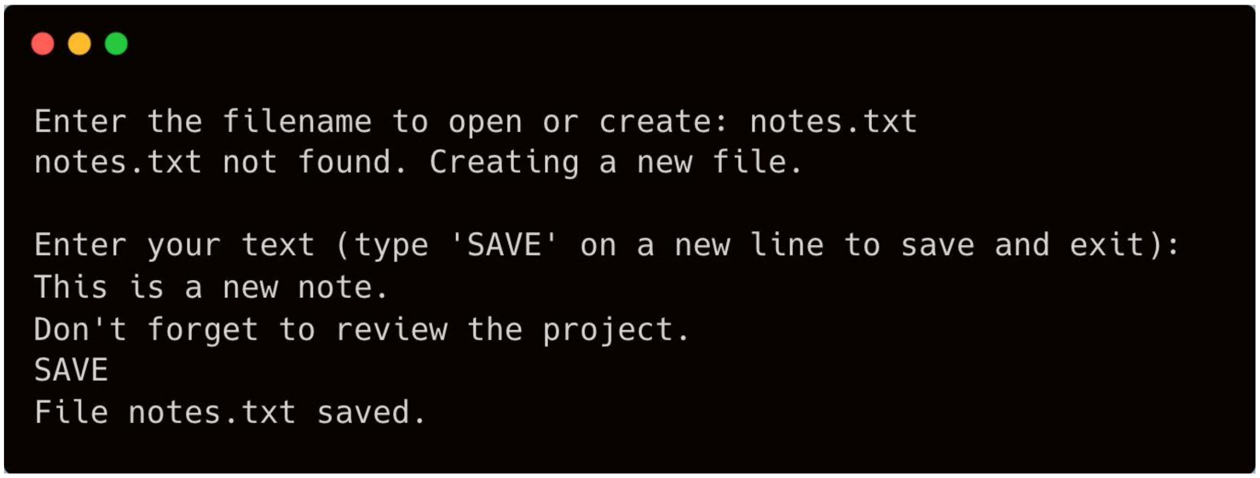
- Add an option in the menu that allows users to mark tasks as completed instead of removing them. Completed tasks can be displayed separately or removed from the list.

- Implement functionality that saves the list to a file and loads it when the program starts. This way, users can maintain their to-do list across different sessions.

- Allow users to categorize tasks (e.g., Work, Personal) and filter the list based on these categories. This adds an organizational element to the to-do list.

Simple Text Editor

Write a program to create a simple text editor. This program allows users to open an existing text file or create a new one, edit the content, and then save the changes by typing the SAVE command.



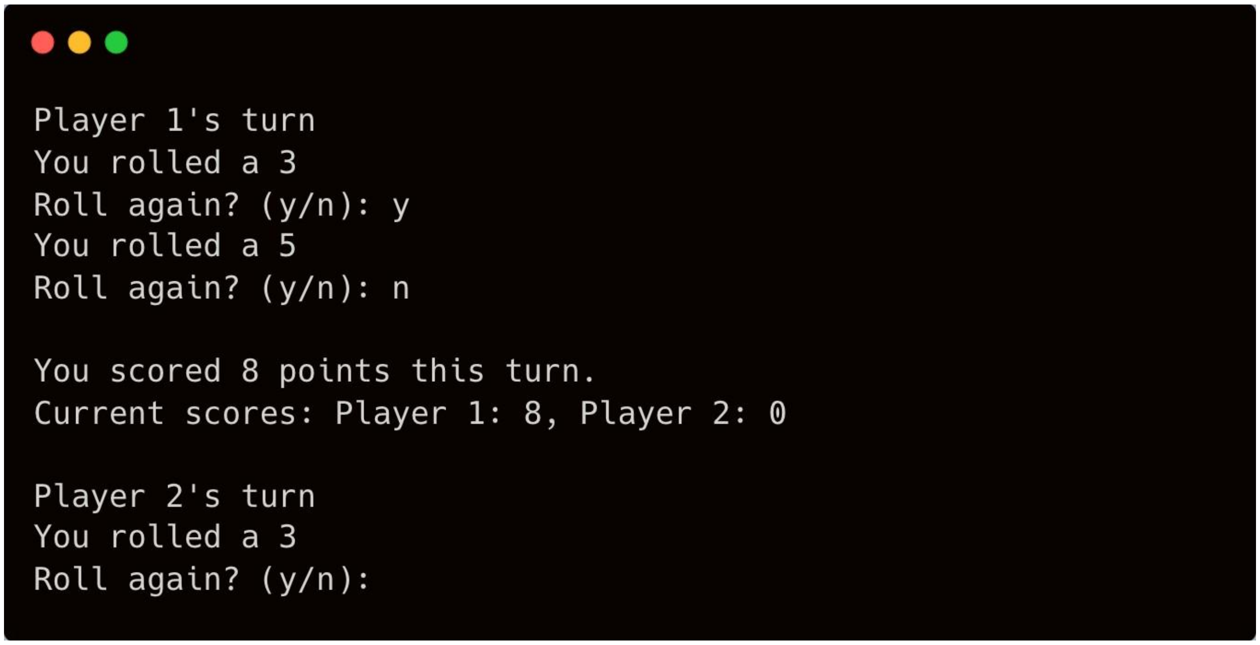
```
Enter the filename to open or create: notes.txt
notes.txt not found. Creating a new file.

Enter your text (type 'SAVE' on a new line to save and exit):
This is a new note.
Don't forget to review the project.
SAVE
File notes.txt saved.
```

Optional Enhancements

- Allow users to choose whether they want to overwrite the existing file content or append new text to the end of the file.
- Add functionality to search for specific words or phrases in the text and replace them with new content.

Pig Dice Game

The Pig Dice Game is a fun, strategic game where two players compete to be the first to reach 100 points. In this project, you'll build a simplified version of the game where each player rolls a die repeatedly to accumulate points. However, if they roll a 1, they lose all points accumulated in that turn and must pass the die to the other player. The game requires players to decide whether to keep rolling to increase their score or to hold and secure their points before risking a roll of 1.



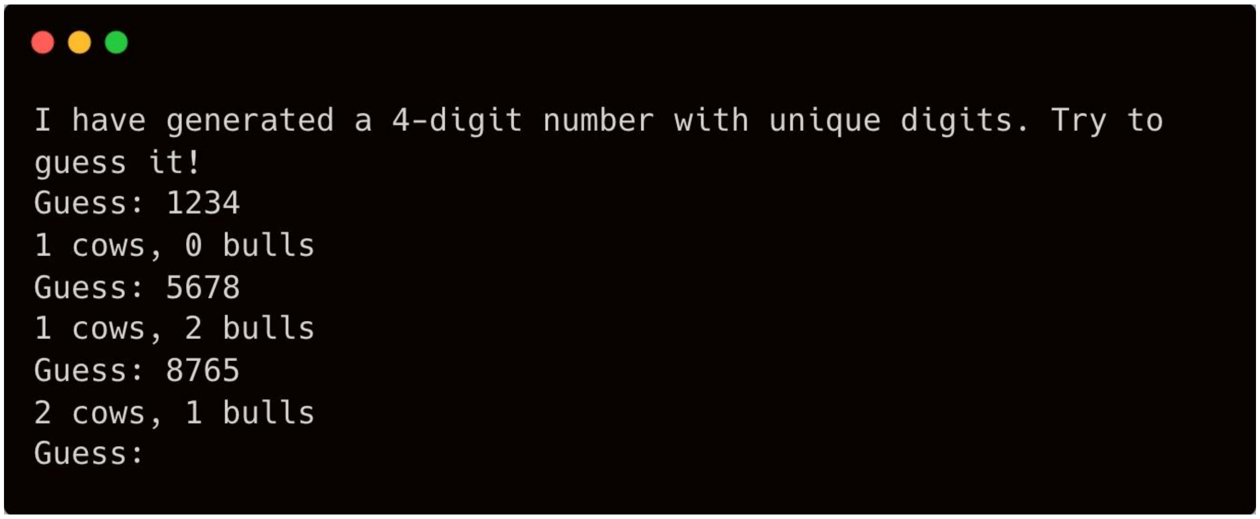
```
Player 1's turn
You rolled a 3
Roll again? (y/n): y
You rolled a 5
Roll again? (y/n): n

You scored 8 points this turn.
Current scores: Player 1: 8, Player 2: 0

Player 2's turn
You rolled a 3
Roll again? (y/n):
```

Optional Enhancements

- Allow players to set a custom target score before starting the game.
- Modify the program to support more than two players, with each player taking turns to roll the dice and accumulate points.
- Implement a feature that tracks the total score of each player over multiple rounds or games, allowing for a cumulative competition.
- Introduce a rule where rolling two 6s consecutively resets the player's score to 0, adding an extra layer of strategy.

Cows and Bulls Game

The Cows and Bulls Game is a fun number-guessing challenge that tests your logical thinking and problem-solving skills. In this project, you'll build a game where the computer comes up with a secret 4-digit number, and your job is to guess it. After each guess, you'll get feedback in the form of "cows" and "bulls"—a "bull" means you've guessed the right digit in the right spot, while a "cow" means the digit is correct but in the wrong spot.



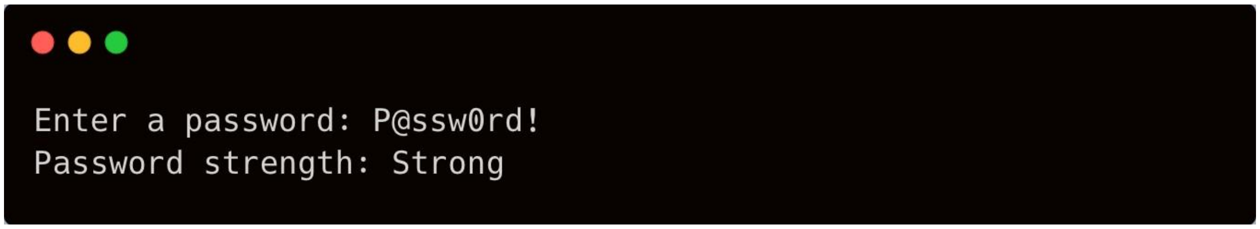
```
I have generated a 4-digit number with unique digits. Try to
guess it!
Guess: 1234
1 cows, 0 bulls
Guess: 5678
1 cows, 2 bulls
Guess: 8765
2 cows, 1 bulls
Guess:
```

Optional Enhancements

- Allow the player to choose a difficulty level at the start of the game, which changes the length of the secret number or the number of attempts allowed.
- Implement a system that offers hints after a certain number of incorrect guesses, providing more guidance to the player.

Password Strength Checker

Write a program to evaluate the strength of a password based on criteria like length, the inclusion of uppercase letters, numbers, and special characters. The program will then categorize the password as Very Weak, Weak, Medium, Strong, or Very Strong.



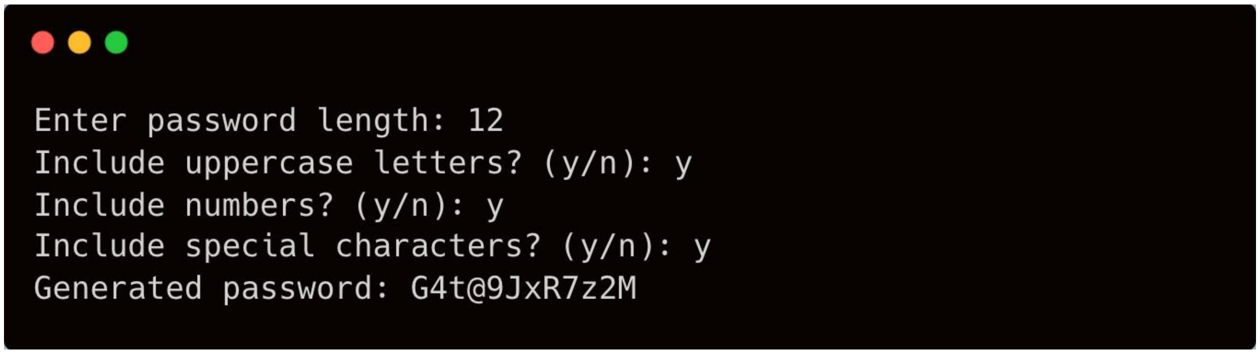
```
Enter a password: P@ssw0rd!
Password strength: Strong
```

Optional Enhancements

- After evaluating the password, give the user suggestions on how to make it stronger, such as adding more characters, including special symbols, or avoiding common words.

Password Generator

Write a program to generate a random password based on user-specified criteria, such as length, and whether to include uppercase letters, numbers, and special characters.



```
Enter password length: 12
Include uppercase letters? (y/n): y
Include numbers? (y/n): y
Include special characters? (y/n): y
Generated password: G4t@9JxR7z2M
```

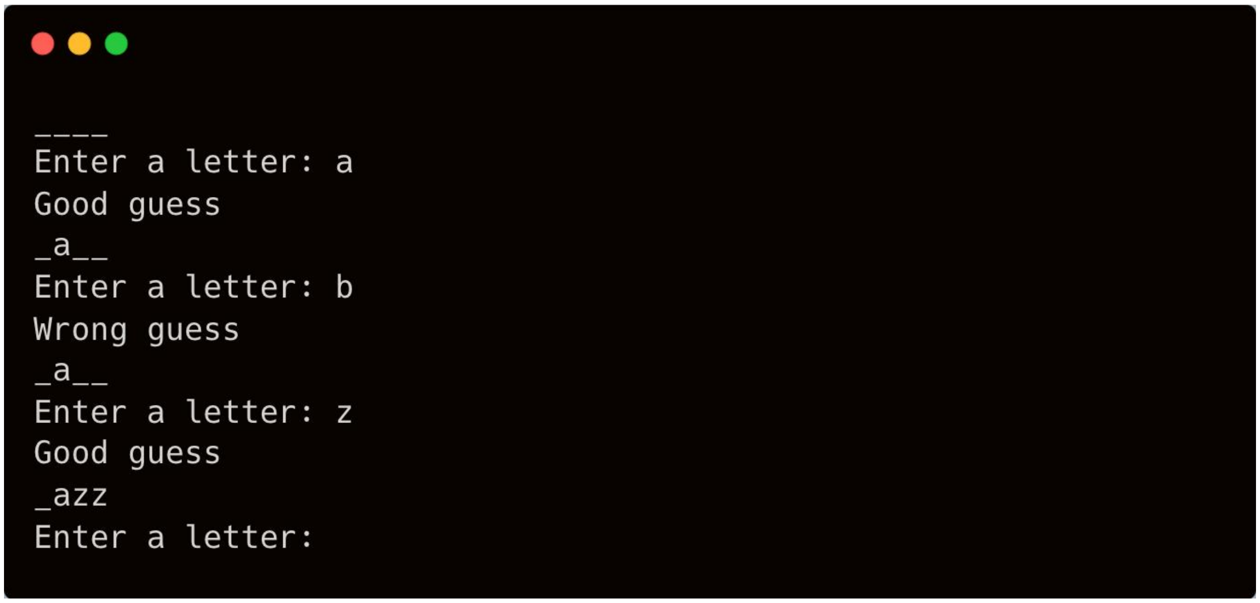
Optional Enhancements

- Modify the program so the user can specify how many passwords they want to generate at one time. The program should then create and display all the passwords.

- Add an option to save the generated passwords to a text file. This allows users to keep a record of their passwords for future reference.

Word Guessing Game

The Word Guessing Game is a fun and interactive project where players try to guess a secret word, one letter at a time. The word is selected randomly from a list of words stored in a text file. The player has six attempts to guess the word, with each incorrect guess reducing the remaining attempts. Correctly guessed letters are revealed in their respective positions, while incorrect guesses prompt the player to try again.



```
____
Enter a letter: a
Good guess
_a__
Enter a letter: b
Wrong guess
_a__
Enter a letter: z
Good guess
_azz
Enter a letter:
```

Optional Enhancements

- Implement a hint feature that the player can use once or twice per game to reveal a letter in the word. This can make the game a bit easier and more fun.

- Offer different difficulty levels that change the length of the words to be guessed. Longer words could be for more advanced players, while shorter words could be for beginners.

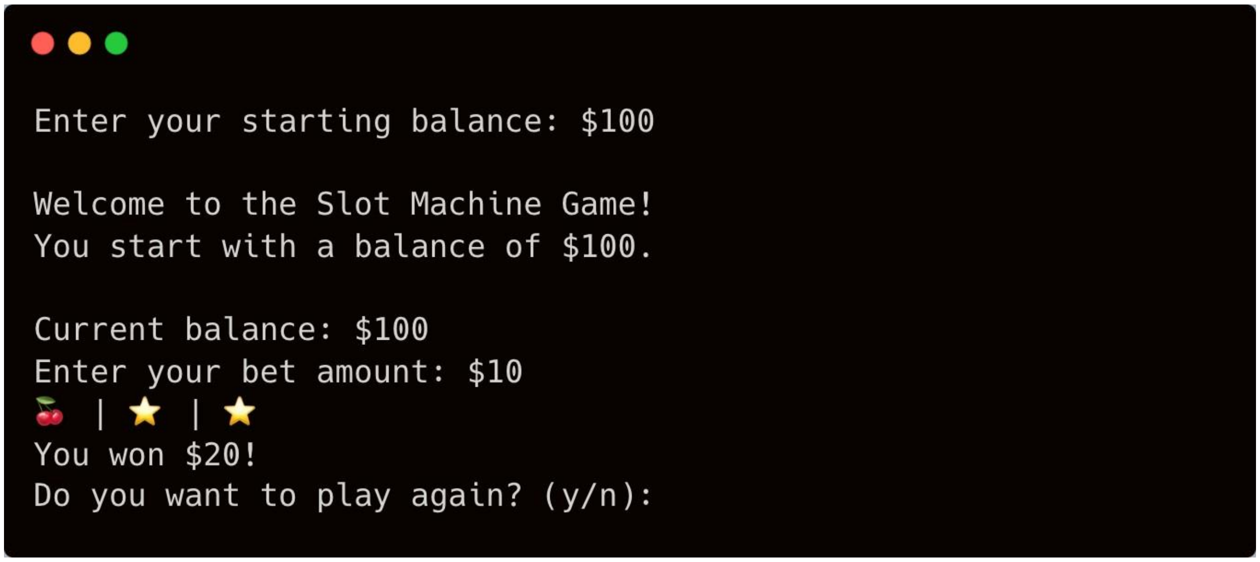
- Keep a record of how many games the player has won or lost during their session. Display this information at the end of each game.

Slot Machine Game

The Slot Machine Game is a fun and simple project that simulates a classic slot machine. The player starts with a balance, places bets, and spins the reels. If the symbols on the reels match, the player wins a payout based on their bet. The game continues until the player either runs out of money or decides to walk away with their winnings.

Payout Rules:

- If all three symbols match, the player wins 10 times their bet.
- If two out of three symbols match, the player wins 2 times their bet.
- If none of the symbols match, the player loses their bet.



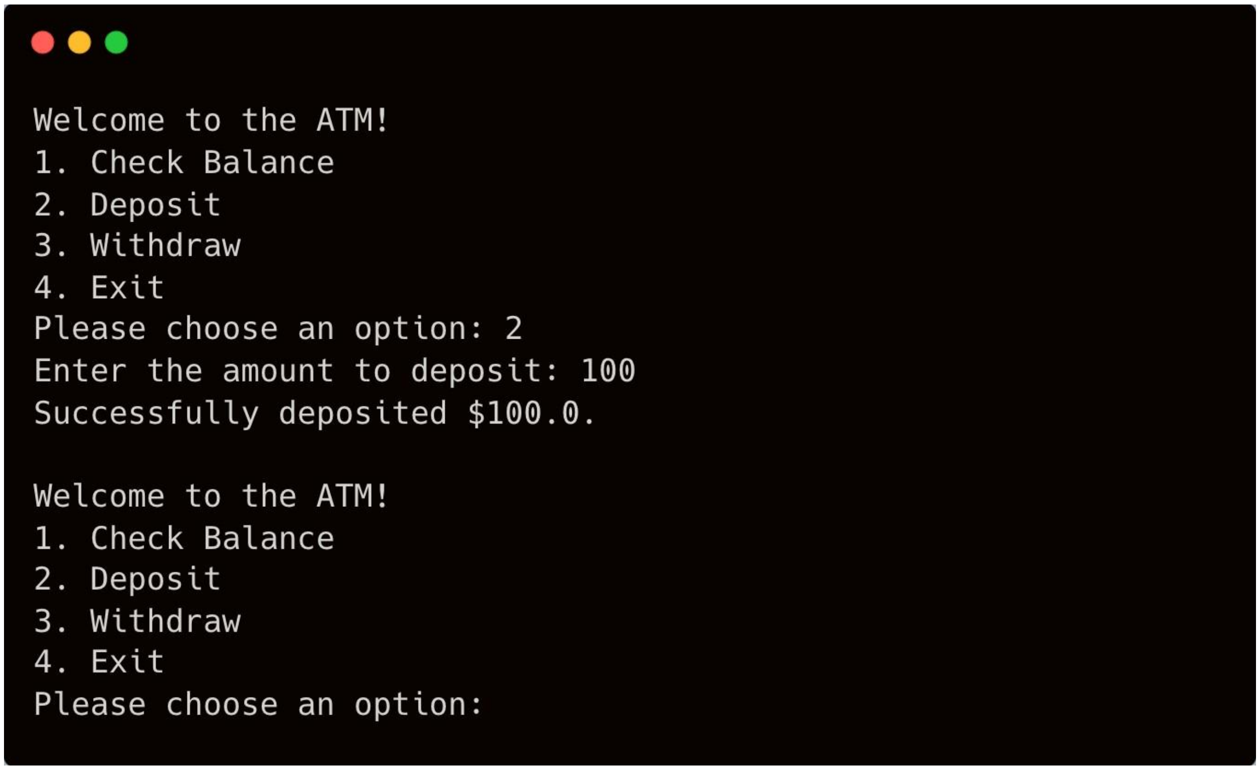
```
Enter your starting balance: $100

Welcome to the Slot Machine Game!
You start with a balance of $100.

Current balance: $100
Enter your bet amount: $10
🍒 | ⭐ | ⭐
You won $20!
Do you want to play again? (y/n):
```

Optional Enhancements

- Introduce new payout combinations, such as matching two specific symbols, for more chances to win.

ATM Simulation

The ATM Simulation project is a practical and engaging exercise where you create a simple ATM system. The system allows users to check their balance, deposit money, and withdraw funds. This project introduces basic Object-Oriented Programming OOP concepts by encapsulating the ATM's logic within a class and handling user interactions through a controller class.



```
Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please choose an option: 2
Enter the amount to deposit: 100
Successfully deposited $100.0.

Welcome to the ATM!
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Please choose an option:
```

Optional Enhancements

- Implement a PIN system where users must enter their PIN before accessing the ATM functions.
- Add a feature to keep track of all transactions (deposits and withdrawals) and allow users to view their transaction history.
- Allow the system to handle multiple user accounts, enabling users to log in and manage their individual balances.