

**Hangman Game CodeAlpha Internship Task 1**

**Developer: Mithanya.M**

**Internship Batch: July 2025**

**Organization: CodeAlpha**

**INTRODUCTION :**

As part of my internship at CodeAlpha, I worked on developing a simple yet engaging Hangman Game using Python.

This project helped me strengthen my problem solving skills and understand the flow of user interaction in a program. I wrote clean and structured code without using any external libraries which gave me hands on experience in string manipulation, loops, and conditionals.

The game allows users to guess a hidden word letter by letter with a limited number of attempts. It was a great learning opportunity to build logic from scratch and handle different scenarios like repeated inputs and end-game conditions.

This project reflects my interest in coding and my goal to grow as a software developer.

**CODE:**

**def hangman():**

**print("Welcome to CodeAlpha Hangman Game!")**

**print("Guess the hidden word one letter at a time.\n")**

**# Word to guess (you can change for testing)**

**word = "PYTHON"**

**guessed = ["\_"] \* len(word)**

**attempts = 6**

**guessed\_letters = []**

**while attempts > 0 and "\_" in guessed:**

**print("Word: ", " ".join(guessed))**

**print("Attempts left:", attempts)**

**guess = input("Enter a letter: ").upper()**

**if guess in guessed\_letters:**

**print(" You already guessed that letter.\n")**

**continue**

**guessed\_letters.append(guess)**

**if guess in word:**

**print("Good guess!\n")**

**for i in range(len(word)):**

**if word[i] == guess:**

**guessed[i] = guess**

**else:**

**attempts -= 1**

**print(" Incorrect guess.\n")**

**if "\_" not in guessed:**

**print(" Congratulations! You guessed the word:", word)**

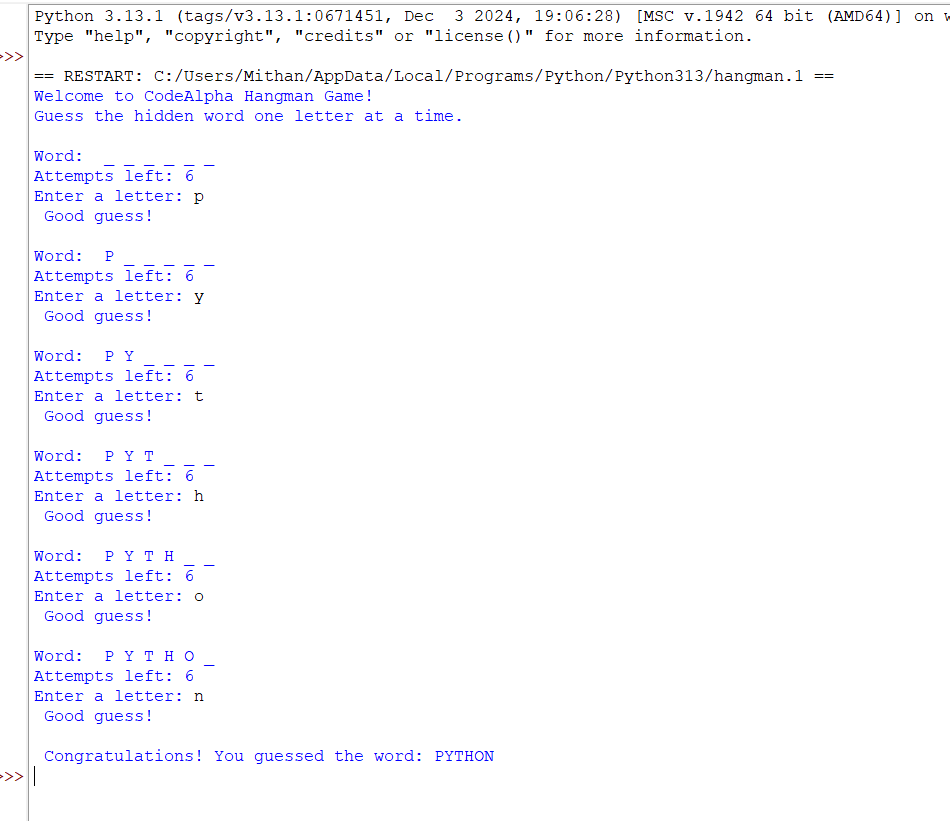
**else:**

**print(" Game Over! The word was:", word)**

**# Run the game**

**hangman()**

**OUTPUT:**

****

**Stock Portfolio Tracker CodeAlpha Internship Task 2**

**Developer: Mithanya.M**

**Internship Batch: July 2025**

**Organization: CodeAlpha**

**INTRODUCTION:**

As part of my internship at CodeAlpha, I developed a Stock Portfolio Tracker using Python.

This program helps users manage their stock investments by adding, viewing, and removing stocks. It calculates the total portfolio value dynamically.

I designed it to be simple and interactive using only core Python features without any external libraries. This project gave me hands-on experience with dictionaries, loops, conditionals, and user input handling.

It was a valuable step in improving my coding logic and preparing for real-world IT challenges.

**CODE:**

**def menu():**

**print("\n Stock Portfolio Tracker ")**

**print("1 Add Stock")**

**print("2 View Portfolio")**

**print("3 Remove Stock")**

**print("4 Exit")**

**portfolio = {}**

**while True:**

**menu()**

**choice = input("Select an option: ")**

**if choice == "1":**

**name = input("Enter stock name: ").upper()**

**try:**

**quantity = int(input("Enter quantity: "))**

**price = float(input("Enter price per stock (₹): "))**

**if quantity > 0 and price > 0:**

**portfolio[name] = {"quantity": quantity, "price": price}**

**print(f" {name} added to your portfolio.")**

**else:**

**print("Quantity and price must be positive numbers.")**

**except ValueError:**

**print(" Please enter valid numbers for quantity and price.")**

**elif choice == "2":**

**if not portfolio:**

**print(" Portfolio is empty.")**

**else:**

**total\_value = 0**

**print("\nYour Portfolio:")**

**for stock, data in portfolio.items():**

**value = data["quantity"] \* data["price"]**

**total\_value += value**

**print(f"{stock}: {data['quantity']} stocks @ ₹{data['price']} = ₹{value}")**

**print(f"Total Portfolio Value: ₹{total\_value}")**

**elif choice == "3":**

**name = input("Enter stock name to remove: ").upper()**

**if name in portfolio:**

**del portfolio[name]**

**print(f" {name} removed from your portfolio.")**

**else:**

**print(" Stock not found in portfolio.")**

**elif choice == "4":**

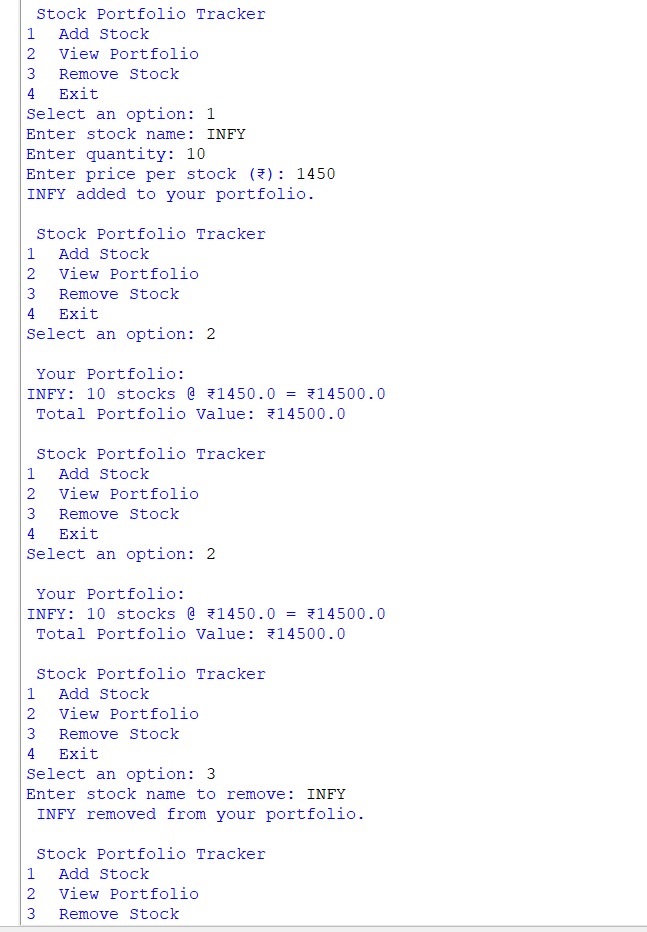
**print(" Exiting Portfolio Tracker. Goodbye!")**

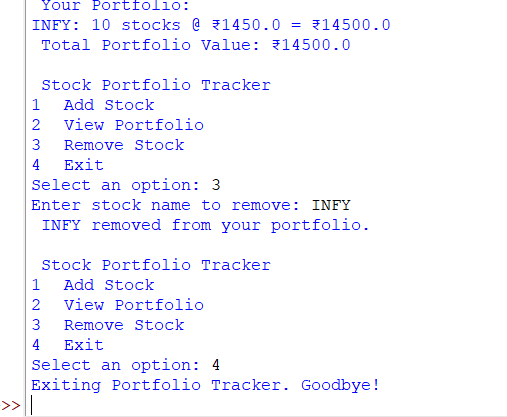
**break**

**else:**

**print("Invalid choice. Try again.")**

**OUTPUT:**

****

****