creativecapsule

Foundation Training '25

CC Programming-Foundations

Instructions

- Write Pseudo code for the below problems
- Test the Pseudo code with the tests provided in the problem statements to validate the logic
- Start coding only after the first two steps are completed.

Overview

This set contains easy problems, this will help you get started with desired language syntax in an incremental manner. Complete all problems in each section before attempting new section.

Section 1

Section 2

Task 1 :Calculate Age in Days

Problem Statement

Convert Age to Days
Create a function that takes the age in years and returns the age in days.

Notes:

- Use 365 days as the length of a year for this challenge.
- Ignore leap years and days between last birthday and now.
- Expect only positive integer inputs.

Expected output

Examples

```
Calculate_Age(23) → 8395
Calculate_Age(0) → 0
Calculate_Age(20) → 7300
```

Task 2 : Simple Calculator

Problem Statement

Write a program that takes two numbers and an operator (+, -, *, /) from the user and performs the calculation.

Expected output

Example

```
Input:
Enter first number: 10
Enter second number: 5
Enter operator (+, -, *, /): *

Output:
Result: 50
```

Task 3 :Reversing a String

Problem Statement

Create a function that accepts a string parameter and returns the reverse of the string.

Expected output

Examples

```
ReverseString('hello') → 'olleh'
ReverseString('goa') → 'aog'
ReverseString('India') → 'aidnI'
```

Task 4 : Sum of Digits

Problem Statement

Write a Python program to calculate the sum of digits of a given number.

Expected output

Example 1

```
Input:
Enter a number: 123

Output:
Sum of digits: 6
```

Example 2

```
Input:
Enter a number: 428

Output:
Sum of digits: 14
```

Task 5 : Find Prime Numbers in a Range

Problem Statement

The user must enter a **starting number** and an **ending number**.

Validate that:

- Both numbers are **positive integers**.
- The **starting number** is **less than or equal to** the ending number.

Expected output

Example 1

```
Input:
Enter starting number: 10
Enter ending number: 30

Output:
Prime numbers between 10 and 30 are:
11 13 17 19 23 29
```

Example 2

If the input is invalid:

```
Input:
Enter starting number: 50
Enter ending number: 30

Output:
Invalid range. Starting number must be less than or equal to ending number.
```

Task 6 : Number Guessing Game (with Attempt Limit)

Problem Statement

Create a number guessing game using Python where:

- The program randomly selects a number between 1 and 100.
- The user has **7 attempts** to guess the number.
- After each guess, print whether the number is too high, too low, or correct.
- Validates that input is an integer within range (1–100)
- If input is invalid, it should prompt again without counting that as an attempt
- Reveals the correct number if all attempts are used

Validate that:

- Withdrawal does not happen if the balance is insufficient, and display an appropriate message.

Expected output

(Note: The randomly selected number is shown here for demonstration purposes.)

Example 1

Let's assume the random number is 45

```
Input:
Guess the number between 1 and 100 (You have 7 attempts)
Attempt 1: Enter your guess: 30
Output: Too low.

Attempt 2: Enter your guess: 60
Output: Too high.

Attempt 3: Enter your guess: 45
Output: Correct! You guessed the number.
```

Example 2

Let’s assume the random number is 72

```
Input:
Guess the number between 1 and 100 (You have 7 attempts)
Attempt 1: 30 → Too low
Attempt 2: 40 → Too low
Attempt 3: 50 → Too low
Attempt 4: 60 → Too low
Attempt 5: 75 → Too high
Attempt 6: 73 → Too high
Attempt 7: 71 → Too low

Output:
You've used all attempts. The number was 72.
```

Example 3

Let’s assume the random number is 72

```
Input:
Guess the number between 1 and 100 (You have 7 attempts)
Attempt 1: Enter your guess: hello
Output: Invalid input. Please enter an integer between 1 and 100.

Attempt 1: Enter your guess: 105
Output: Invalid input. Please enter an integer between 1 and 100.

Attempt 1: Enter your guess: 50
Output: Too high.
```

Note: Only valid attempts are counted toward the 7.

Task 7 : ATM Simulation

Problem Statement

Write a program to simulate basic ATM operations:

- The user starts with a balance of **₹10,000**.
- Show a menu with the following options:
    1. Check Balance
    2. Deposit
    3. Withdraw
    4. Exit
- Validate for valid menu choices.
- Validate that deposit/withdrawal amounts are positive numbers.
- Ensure withdrawal doesn't exceed balance.

- Keep displaying the menu until the user chooses to **Exit**.

Expected output

Example 1: Valid Operations

```
Input:
--- ATM Menu ---
1. Check Balance
2. Deposit
3. Withdraw
4. Exit
Choose an option (1-4): 1

Output:
Current Balance: ₹10000.0
```

```
Input:
--- ATM Menu ---
Choose an option (1-4): 2
Enter amount to deposit: 2000

Output:
Deposited successfully.
```

```
Input:
--- ATM Menu ---
Choose an option (1-4): 3
Enter amount to withdraw: 1500

Output:
Withdrawal successful.
```

```
Input:
--- ATM Menu ---
Choose an option (1-4): 1

Output:
Current Balance: ₹10500.0
```

Example 2: Invalid Menu Choice

```
Input:
--- ATM Menu ---
Choose an option (1-4): 9

Output:
Invalid option. Please try again.
```

Example 3: Negative Deposit

```
Input:
--- ATM Menu ---
Choose an option (1-4): 2
Enter amount to deposit: -500

Output:
Invalid amount. Please enter a positive number.
```

Example 4: Withdraw More Than Balance

```
Input:
--- ATM Menu ---
Choose an option (1-4): 3
Enter amount to withdraw: 50000

Output:
Insufficient balance!
```

Task 8 : Sentence Value

Problem Statement

Write a program that prints total value of a sentence where

- a sentence can only contain alphabets from a to z (all smallcase),
- each alphabet represents a value starting from a = 11 , b= 12 , c=13 ...to z = 36
- treat uppercase letters as lowercase
- you are not allowed to use array to store values of each alphabet such as 11, 12, .. 36
- you are not allowed to use 26 If statements

Expected output

Example 1

```
Input:
Enter a sentence: Test

Output:
value of 'Test' is 104
```

Explanation: The word "Test" has a value of 30 + 15 + 29 + 30 = 104

Example 2

```
Input:
Enter a sentence: Hello World

Output:
value of 'Hello World' is 224
```

Explanation: Total value: 18 + 15 + 22 + 22 + 25 + 33 + 25 + 28 + 22 + 14 = 224
