C PROGRAMMING
ASSIGNMENT 03.03

Problem 1: Write a C program to check whether a given character is an alphabet, digit, or special character.

Solution Approach:

1. Prompt the user to input a single character.
2. Use conditional checks to determine the type of the character:
    - If the character is between 'A' and 'Z' or between 'a' and 'z', it is an alphabet.
    - If the character is between '0' and '9', it is a digit.
    - Otherwise, it is a special character.
3. Print the type of the character.

Test Cases:

- **Test Case 1:**
  Input: `A`
  Expected Output:
  ```
  The character is an alphabet.
  ```
- **Test Case 2:**
  Input: `5`
  Expected Output:
  ```
  The character is a digit.
  ```
- **Test Case 3:**
  Input: `#`
  Expected Output:
  ```
  The character is a special character.
  ```

Problem 2: Write a C program to check whether a given alphabet is a vowel or a consonant.

Solution Approach:

1. Prompt the user to input a single alphabet character.
2. Convert the character to lowercase if it is in uppercase for uniformity.
3. Check if the character is a vowel ('a', 'e', 'i', 'o', 'u').
4. If it is a vowel, print that it is a vowel; otherwise, print that it is a consonant.

Test Cases:

- **Test Case 1:**
  Input: `E`
  Expected Output:
  ```
  The alphabet is a vowel.
  ```
- **Test Case 2:**
  Input: `b`
  Expected Output:
  ```
  The alphabet is a consonant.
  ```
- **Test Case 3:**
  Input: `y`
  Expected Output:
  ```
  The alphabet is a consonant.
  ```

Problem 3: Write a C program to calculate profit or loss based on the cost price (CP) and selling price (SP) of a product.

Solution Approach:

1. Prompt the user to input the cost price (CP) and selling price (SP) of a product.
2. Calculate the difference between SP and CP.
3. If SP > CP, print the profit amount.
4. If SP < CP, print the loss amount.
5. If SP == CP, print that there is no profit or loss.

Test Cases:

- **Test Case 1:**
  Input: `CP = 100, SP = 150`
  Expected Output:
  ```
  Profit = 50
  ```
- **Test Case 2:**
  Input: `CP = 200, SP = 180`
  Expected Output:
  ```
  Loss = 20
  ```
- **Test Case 3:**
  Input: `CP = 50, SP = 50`
  Expected Output:
  ```
  No profit, no loss.
  ```

Problem 4: Write a C program to calculate and print the electricity bill of a customer. The program should capture the customer ID, name, and units consumed from the keyboard and calculate the total bill based on the following charges:

Unit	Charge/Unit
Up to 199	₹1.20
200 - 399	₹1.50
400 - 599	₹1.80

600 and above	₹2.00

If the bill exceeds ₹400, a surcharge of 15% will be applied. The minimum bill should be ₹100.

Solution Approach:

1. Capture the customer's ID, name, and units consumed.
2. Calculate the bill based on the units consumed:
    - For units up to 199, charge ₹1.20 per unit.
    - For units between 200 and 399, charge ₹1.50 per unit.
    - For units between 400 and 599, charge ₹1.80 per unit.
    - For units 600 and above, charge ₹2.00 per unit.
3. Check if the bill exceeds ₹400 and, if so, apply a 15% surcharge.
4. Ensure the final bill is at least ₹100.
5. Print the customer's details along with the total bill amount.

Test Cases:

- **Test Case 1:**
  Input: `Customer ID: 1001, Name: John, Units: 150`
  Expected Output:

```
Customer ID: 1001
Name: John
Units Consumed: 150
Total Bill: ₹180.00
```

- **Test Case 2:**
  Input: `Customer ID: 1002, Name: Jane, Units: 450`
  Expected Output:

```
Customer ID: 1002
Name: Jane
Units Consumed: 450
Bill: ₹810.00
Surcharge: ₹121.50
Total Bill: ₹931.50
```

- **Test Case 3:**
  Input: `Customer ID: 1003, Name: Emily, Units: 50`
  Expected Output:

```
Customer ID: 1003
Name: Emily
Units Consumed: 50
Total Bill: ₹100.00
```

Problem 5: Write a C program to accept a grade (E, V, G, A, F) and print the equivalent description.

Grade	Description
E	Excellent
V	Very Good
G	Good
A	Average
F	Fail

Solution Approach:

1. Prompt the user to input a grade.
2. Use a switch-case or if-else statements to map the grade to its corresponding description.
3. Print the description.

Test Cases:

- **Test Case 1:**
  Input: `Grade: E`
  Expected Output:
  `Description: Excellent`
- **Test Case 2:**
  Input: `Grade: V`
  Expected Output:
  `Description: Very Good`
- **Test Case 3:**
  Input: `Grade: A`
  Expected Output:
  `Description: Average`
- **Test Case 4:**
  Input: `Grade: X`
  Expected Output:
  `Invalid grade entered.`