Assignment-2: Function

Problem 1. Food App Simulation

This **Food App Simulation** assignment requires you to create a simple console-based application that mimics the process of ordering food, providing feedback, and generating a receipt. The user will interact with the app by selecting a food item, giving feedback, providing a tip, and viewing a summary of their order.

Key Points to Understand:

1. **Welcome Screen**: The app greets the user and asks for their name.

2. **Food Menu**: The app displays a list of at least four food items with prices. The user selects one, and the app ensures their choice is valid.

3. **Feedback & Tip**: The user can provide optional feedback and choose to add a tip. If the user skips the tip, it defaults to zero.

4. **Receipt**: After placing the order, the app generates a receipt showing the user's name, the food item ordered, the tip, the total amount, and any feedback.

5. **Workflow**: The app needs to guide the user smoothly through each step, ensuring the flow is clear and handles invalid inputs effectively.

6. **Closing Message**: At the end, the app thanks the user for using the service.

Problem 2.Menu-Driven Program: Basic Calculator

Objective:

Create a simple console-based calculator that allows users to perform basic arithmetic operations like addition, subtraction, multiplication, and division**.**

Requirements:

1. **Menu Display:**
    - The program presents a menu with options for four operations (Add, Subtract, Multiply, Divide) and an option to exit.

2. **Input Handling:**
    - The user selects an operation by entering a number corresponding to the operation (1 for Add, 2 for Subtract, etc.).
    - The program prompts the user to enter two numbers for the selected operation.

3. **Arithmetic Operations:**
    - Based on the selected operation, the program performs the corresponding arithmetic calculation:
        - **Addition**: Adds the two numbers.
        - **Subtraction**: Subtracts the second number from the first.
        - **Multiplication**: Multiplies the two numbers.
        - **Division**: Divides the first number by the second (with a check for division by zero).

4. **Invalid Input Handling:**

    - If the user selects an invalid option or enters invalid numbers, the program prompts them again.

Problem 3:BMI (Body Mass Index) Calculator.

Objective: Develop a function that calculates the BMI based on a person's weight and height, then categorizes the result into standard BMI categories.

`weight` (in kilograms) and `height` (in meters). It computes the BMI using the formula. BMI=weight/(height^2)

After calculating the BMI, the function categorizes the result as follows:

- **Underweight:** BMI less than 18.5
- **Normal weight:** BMI between 18.5 and 24.9
- **Overweight:** BMI between 25 and 29.9
- **Obese:** BMI 30 or above

Implementation Steps:

1. **Input Validation:** Ensure that the `height` is a positive number. If not, display an error message and exit the function.
2. **BMI Calculation:** Compute the BMI using the provided formula.
3. **Categorization:** Determine the BMI category based on the calculated value.
4. **Output:** Display the BMI value and its corresponding category.

Problem 4:Dynamic Discount Calculation

Develop a program to calculate a **discount** based on the customer type and purchase amount.

- **Discount Rules**:
  - **Regular customers**:
    - 10% discount for purchases ≥ 500.
    - 5% discount for purchases < 500.
  - **Gold customers**:
    - 15% discount for purchases ≥ 500.
    - 10% discount for purchases < 500.
  - **Platinum customers**:
    - 20% discount for purchases ≥ 500.
    - 15% discount for purchases < 500.
- Input: Customer type and purchase amount.
- Output: The calculated discount percentage.

Problem 5:Student Grading System

Objective:

Design a system that calculates and displays a student's grade based on their marks using JavaScript.

Instructions:

1. **Create a Function**:
   - Write a function that determines the grade for a student based on their marks.
   - Use **arrow function syntax**.
2. **Grading Criteria**:
   - Marks between **90 and 100**: `A+`
   - Marks between **80 and 89**: `A`

    - Marks between **70 and 79**: `B`
    - Marks between **60 and 69**: `C`
    - Marks between **50 and 59**: `D`
    - Marks between **0 and 49**: `F`
    - Marks outside the range **0–100**: `Invalid Marks`
3. **Input Validation**:
    - Prompt the user to enter their marks using a dialog box.
    - Ensure the input is numeric and within the valid range.
4. **Output**:
    - Display the student's marks and grade in a formatted message.
    - If the input is invalid, show a message to the user to enter a valid number.

Problem 6: Seat Swapping Program

Objective:

Create a seat-swapping program using arrow functions to swap two seat allocations between passengers.

Instructions:

1. **Problem Understanding:**
    - Swap two seat numbers (e.g., "12A" and "14B") and display the seat assignments before and after swapping.
2. **Tasks:**
    - Write an arrow function to swap the two seat numbers and display the result.
    - Modify the function to return the updated seat numbers instead of printing them directly.

Problem 7: Hotel Booking System.

Objective:

Create a simple hotel booking system using arrow functions, which allows the user to select a room type and specify the number of nights. The program will calculate the total cost based on the room type and display a booking summary.

Instructions:

1. **User Details:**

    - Collects user details (name, mobile, age, address, and the number of rooms).
    - Allows the user to select a room type (Standard, Deluxe, Suite).
    - Asks for the number of nights to stay.
    - Calculates the total cost based on the selected room type, number of nights, and number of rooms.
    - Displays a booking summary or error messages for invalid inputs

2. **Rating System:**
    - **Standard Room**: $100 per night
    - **Deluxe Room**: $200 per night
    - **Suite Room**: $300 per night
    - If the room type is invalid, the user will be notified.

3. **Tasks:**

    - **Function Implementation:**
        - Implement an arrow function to calculate the total cost of the booking based on room type and nights.
        - Implement an arrow function to handle the booking process.
        - Implement an arrow function to start the booking simulation.
