Assignment Questions

---

1. Write a program using a while loop to print the numbers from 1 to 10.

- **Hint: Use a counter variable initialized to 1. Increment the counter by 1 in each iteration and print it. Use a condition counter <= 10 for the loop.**

2. Modify the above program to skip printing the number 5 using the continue statement.

- **Hint: Add an if condition inside the loop to check if the counter equals 5. Use the continue statement to skip the rest of the loop body for that iteration.**

3. Write a program to sum up the digits of a number entered by the user until the sum becomes a single-digit number.

- **Hint: Use num % 10 to extract the last digit and num // 10 to remove it. Keep summing the digits in a loop. If the sum has more than one digit, repeat the process.**

  Example:

  num=534, now add 5+3+4

  sum = 12, since sum has more than 1 digit, sum the digits again, like 1+2

  sum = 3, finally sum has only one digit, exit the loop

4. Create a while loop that continuously takes user input and stops when the user enters "exit" (use break).

- **Hint: Use input() to read user input in the loop. Check if the input equals "exit". If true, use the break statement to exit the loop.**

5. Write a program to find the smallest divisor of a given number using a while loop.

- **Hint: Start with a divisor of 2 (smallest prime number) and increment it in each iteration. Use the modulus operator % to check if the number is divisible by the divisor.**

6. Create a program to print all even numbers between 1 and 20 using a while loop, skipping multiples of 6 (use continue).

- **Hint: Use a condition to check if the number is even (num % 2 == 0) and print it. If it's also a multiple of 6 (num % 6 == 0), use continue to skip printing.**

7. Write a program to reverse a number entered by the user using a while loop.

- **Hint: Use num % 10 to get the last digit and build the reversed number by multiplying the current reversed number by 10 and adding the digit. Divide the number by 10 in each iteration.**

8. Write a program to display the first 10 Fibonacci numbers using a while loop.

- **Hint: Start with two variables initialized to 0 and 1 (first two Fibonacci numbers). Print them and compute the next number by adding the previous two. Use a counter to stop after 10 iterations.**

9. Implement a program that continuously prompts the user to input a number. If the input is negative, skip it; if it's zero, stop the program.

- **Hint: Use input() to get numbers in a loop. Use if conditions to check for negative numbers (use continue) or zero (break to exit the loop). For positive numbers, print them.**

2. Functions

10. Write a function greet() that prints "Hello, World!" and call it.

11. Write a function introduce(name, age) that prints a greeting message including the person's name and age. Call the function with various names and ages.

12. Create a function multiply(a, b) that multiplies two numbers and prints the result. Call it with two integers and two floats.

13. Write a function describe_country(country="India") that prints "I live in [country]". Call the function with and without providing a country name.

14. Create a function calculate_tip(bill, tip_percentage=10) that prints the tip amount based on the bill and the tip percentage. Call it with and without the tip percentage.

15. Write a function print_items(*items) that prints each item on a new line. Call it with various numbers of arguments.

16. Create a function sum_numbers(*numbers) that calculates and prints the sum of all numbers passed as arguments. Call it with 3, 5, and 8 numbers.

Real-Life Scenario: Restaurant Order System

Problem Statement:

You are tasked with creating a simple menu-driven system for a restaurant that allows customers to order food. The system should display a menu of available food items, let the customer choose items, and calculate the total price of their order. The system should continue prompting for orders until the customer decides to exit.

Menu:

1. **Pizza** - $10
2. **Burger** - $5
3. **Pasta** - $7
4. **Fries** - $3
5. **Exit** - Exit the ordering system

Program Flow:

1. Display the menu and ask the customer to choose an item.
2. Based on the customer's choice, call a function to add the selected item to their order and update the total price.
3. If the customer chooses "Exit", exit the program and display the total price.
4. The program should continue to display the menu until the customer selects "Exit."
5. If the customer enters an invalid choice, print an error message and prompt for a valid choice.

Requirements:

1. Use a **while loop** to repeatedly show the menu until the user selects "Exit."
2. Use **break** to exit the loop when the customer chooses "Exit."
3. Implement each menu option as a separate function.
4. Keep track of the **total price** and show it when the customer exits.

Hints for Implementation:

- **Add Item Function**: Create a function add_item() that prompts the user to select an item and updates the total price.
- **Display Menu Function**: Create a function display_menu() to print the available food items and prices.
- **Input Validation**: Ensure that the customer's selection is a valid menu choice (1-5). If an invalid option is selected, ask the customer to try again.
- **Exit Function**: When the customer chooses "Exit", display the total cost of the meal and exit the loop using break.

Output seems like this:

```
Welcome to the Restaurant Ordering System!

Menu:
1. Pizza - $10
2. Burger - $5
3. Pasta - $7
4. Fries - $3
5. Exit

Please select an item (1-5): 1
You selected Pizza. Total price: $10

Menu:
1. Pizza - $10
2. Burger - $5
3. Pasta - $7
4. Fries - $3
5. Exit

Please select an item (1-5): 3
You selected Pasta. Total price: $17

Menu:
1. Pizza - $10
2. Burger - $5
3. Pasta - $7
4. Fries - $3
5. Exit

Please select an item (1-5): 5
Thank you for your order! Your total is $17. Goodbye!
```