UNIVERSITY OF THE PUNJAB
Session: BS (IT) Fall 2020 Morning
Course: Enterprise Systems
Assignment 03, Sessional Weightage: **0.5** out of 25

Guidelines and instructions:

- This is a graded assignment, you can collaborate to understand the concepts, but plagiarism is not allowed. You are directed to complete all tasks individually. Any case of plagiarism would result in 0 marks.
- Please follow the coding conventions
  - Indent your code properly
  - Use meaningful variable and method names, mind the usage of *PascalCase* and *camelCase* notation
  - Comment your code properly
- You are requested to
  - Create a new branch named **`Assignment-03`** in the repository you've created in the first assignment
  - If you have not created your repository then create one private repository on GitHub on this pattern: **`Rollnumber-EAD`** (e.g., **`BITF20M001-EAD`**)
  - Send GitHub link of your assignment at this email: bsef20m021@pucit.edu.pk
    - Subject line: '<Roll Number>-Assignment-02' (e.g. BITF20M001-Assignment-02)
- The submission deadline of this assignment is **Tuesday, 10 October, 2023 10:14:59 AM**
- This assignment should not take more than 90 minutes to complete
- Please note that the assignments submitted after the deadline will not be entertained

Happy coding 😊

Please write the code for the following statements: (4 marks + 11 marks = 15 marks)

1. Optional arguments (4 marks)

   a. Write a method named **`Greet`** with two optional parameters: **`greeting`** (string) and **`name`** (string). If no values are provided for these parameters when calling the method, it should display the message **`"Hello, World!"`** on the console. If values are provided, it should display a custom greeting message using the provided values, such as **`"[greeting], [name]!"`**. (1 mark)

   b. Create a method called **`CalculateArea`** for calculating the area of a rectangle. The method should have two optional parameters: **`length`** (double) and **`width`** (double), both with default values of **`1.0`**. The method should return the area of the rectangle. (1 mark)

   c. Create two overloaded methods: **`AddNumbers`**. The first method takes two integer parameters and returns their sum. The second method is an overload that takes three integer parameters, with the third parameter being optional and having a default value of 0. Implement the logic to add the numbers and return the sum. (1 mark)

   d. In C#, you can use optional arguments in constructors. Create a **`Book`** class, and define a constructor for the class that takes two parameters: **`title`** (string) and **`author`** (string). Make the **`author`** parameter optional with a default value of **`"Unknown"`**. Create instances of the **`Book`** class with and without specifying the author, and display their details. (1 mark)

2. Generics (11 marks)

   a. Create a generic class called **`MyList<T>`** that can store a list of elements of any data type **`T`**. Implement methods to add elements to the list, remove elements, and display the list. (3 marks)

b. Write a method that defines a generic method called **`Swap<T>`** that takes two parameters of type **`T`** and swaps their values. Test method with integer and string values. (1 mark)

c. Define a generic method that computes the sum of elements in an array of any data type, but restrict the method to accept only types that support addition (e.g., int, long, double). Use generic constraints to achieve this. (1 mark)

d. You are tasked with managing a student database system in a C# program. Each student has a unique student ID (integer) and a corresponding name (string). You have to use dictionary to store this information. (6 marks)

- Add the following student records to the dictionary:
  ```
  Student ID: 101, Name: "Alice"
  Student ID: 102, Name: "Bob"
  Student ID: 103, Name: "Charlie"
  Student ID: 104, Name: "David"
  ```

- Implement a function that allows the user to input a student ID and retrieve the corresponding student's name from the dictionary. Display an appropriate message on console if the student ID is not found.

- Implement a function to display the entire student database, showing both student IDs and names.

- Add a feature to update a student's name by inputting their student ID and the new name.

- Finally, create a menu-driven interface that allows the user to perform the following actions:
  ```
  1. View the student database
  2. Search for a student by ID
  3. Update a student's name
  4. Exit the program
  ```

- Your program should continue running until the user chooses to exit.