Create a Small Project and Debug a Program

Step 1: Planning the Application

You'll create a small application using the use the Visual Studio Code console application you created at the start of the course. Your application allows users to manage a simple to-do list. The program will let users add tasks, view the list of tasks, and mark tasks as completed.

Remove any existing code in the Program.cs file of your console application and create all the code in each step in that file.

Step 2: Creating the Task List

Create an array to store the tasks and a variable to keep track of how many tasks have been added.

Instructions:

1. Define a class called ToDoList.
2. Create an array called tasks that can hold up to 10 tasks.
3. Create a variable called taskCount to keep track of how many tasks have been added.

Step 3: Adding a Task

Create a method to let the user add tasks to the list.

Instructions:

1. Inside the class definition, write a method called AddTask that prompts the user to enter a task and stores it in the tasks array.
2. Increase the taskCount variable each time a task is added.

Step 4: Viewing Tasks

Create a method to display all the tasks the user has added so far.

Instructions:

1. Inside the class definition and below any existing methods, write a method called ViewTasks that loops through the tasks array and prints each task.
2. Use the taskCount variable to know how many tasks to print.

Step 5: Marking a Task as Completed

Create a method to let the user mark a task as completed.

Instructions:

1. Inside the class definition and below any existing methods, write a method called CompleteTask that asks the user to select a task to mark as complete.
2. Update the selected task to show that it's done.

Step 6: Running the Program

Write the Main method that brings everything together and lets the user interact with the program.

Instructions:

1. Inside the class definition and below any existing methods, write the Main method that presents a menu to the user.
2. Use a loop to keep asking the user for input until they choose to exit.
3. To check your answer, run the Visual Studio Code console application. If you receive an error when you run the code, go to the reading on the next page to compare your code to the correct answer.

Step 7: Debugging Techniques

Introduce a common error that can happen in your program and walk through the process of finding and fixing it.

Instructions:

1. Update the CompleteTask method by adding a debugging line to print the taskNumber in the method.
2. Update the CompleteTask method to handle the error gracefully by showing the user an error message if the task number is out of range.
3. To check your answer, run the Visual Studio Code console application. If you receive an error when you run the code, go to the reading on the next page to compare your code to the correct answer.

Code:

```
class ToDoList
{
    static readonly string[] tasks = new string[10];
    static int taskCount = 0;

    static void AddTask()
    {
        if (taskCount >= tasks.Length)
        {
            Console.WriteLine("⚠ Task list is full.");
            return;
        }

        Console.Write("Enter a new task: ");
        string? task = Console.ReadLine();

        if (!string.IsNullOrWhiteSpace(task))
        {
            tasks[taskCount++] = task;
            Console.WriteLine("☑ Task added.");
        }
        else
        {
            Console.WriteLine("❌ Error: Task cannot be empty.");
        }
    }

    static void ViewTasks()
    {
        if (taskCount == 0)
        {
            Console.WriteLine("👎 No tasks to display.");
            return;
        }

        Console.WriteLine("\n📋 To-Do List:");
        for (int i = 0; i < taskCount; i++)
        {
            Console.WriteLine($"{i + 1}. {tasks[i]}");
        }
    }

    static void CompleteTask()
    {
        Console.Write("Enter the number of the task to mark as
completed: ");
        string? input = Console.ReadLine();

        if (int.TryParse(input, out int taskNumber))
        {
            Console.WriteLine($"🛠 DEBUG: You entered task number
{taskNumber}");

            if (taskNumber < 1 || taskNumber > taskCount)
            {
                Console.WriteLine("❌ Error: Invalid task number.");
                return;
```

```
            }

            tasks[taskNumber - 1] += " [Completed]";
            Console.WriteLine("☑ Task marked as completed.");
        }
        else
        {
            Console.WriteLine("❌ Error: Please enter a valid
number.");
        }
    }

    static void DeleteTask()
    {
        Console.Write("Enter the number of the task to delete: ");
        string? input = Console.ReadLine();

        if (int.TryParse(input, out int taskNumber))
        {
            if (taskNumber < 1 || taskNumber > taskCount)
            {
                Console.WriteLine("❌ Error: Invalid task number.");
                return;
            }

            for (int i = taskNumber - 1; i < taskCount - 1; i++)
            {
                tasks[i] = tasks[i + 1];
            }

            tasks[taskCount - 1] = null!;
            taskCount--;

            Console.WriteLine("🗑 Task deleted.");
        }
        else
        {
            Console.WriteLine("❌ Error: Please enter a valid
number.");
        }
    }

    static void Main(string[] args)
    {
        bool running = true;

        while (running)
        {
            Console.WriteLine("\n--- To-Do List Menu ---");
            Console.WriteLine("1. Add Task");
            Console.WriteLine("2. View Tasks");
            Console.WriteLine("3. Complete Task");
            Console.WriteLine("4. Delete Task");
            Console.WriteLine("5. Exit");
            Console.Write("Choose an option (1-5): ");

            string? choice = Console.ReadLine();

            switch (choice)
```

```
                {
                    case "1":
                        AddTask();
                        break;
                    case "2":
                        ViewTasks();
                        break;
                    case "3":
                        CompleteTask();
                        break;
                    case "4":
                        DeleteTask();
                        break;
                    case "5":
                        running = false;
                        Console.WriteLine("👋 Exiting program...");
                        break;
                    default:
                        Console.WriteLine("❌ Invalid option. Try again.");
                        break;
                }
            }
        }
    }
```