**developer guide for the following class(.cs file):**

Logic.cs, Operation.cs, OperationHandler.cs, Storage.cs, Task.cs

**Architecture:**

The program serves as the main entry of the whole project. It will call UI and then Logic will be called. The creation of Logic will be followed by creation of OperationHandler.

The Logic control the sequence of command processing: first call commandparser to parse command, then get operation from it and pass operation to OperationHandler to execute then get result, which can serve as feedbacks to user in UI.

The Operation is definition for different types of operations. The OperationHandler execute operation and get feedbacks.

The Task is definition of different types of tasks.

The storage save information of tasks into xml files.

**Main User Interface**

***Logic.cs*:**

Attributes

|  |  |
| --- | --- |
| Variables | Explanation |
| OperationHandler operationHandler; | Instance of OperationHandler that execute operations |
| List<Task> taskList; | List of tasks to record user data |
| Functions | Explanation |
| public string ProcessCommand(string input); | Processes command from ui and return feedbacks |

Constructor

|  |  |
| --- | --- |
| Initializations | Explanation |
| operationHandler = new OperationHandler(); | Creates instance of operationhandler |
| taskList = new List<Task>(); | Creates instance of tasklist |

***Operation:***

Subclasses

|  |  |
| --- | --- |
| Classes | Constructor |
| OperationAdd | **OperationAdd(Task setTask);** |
| OperationDelete | **OperationDelete(int index);** |
| OperationDisplay |  |
| OperationSearch | **OperationSearch(string searchString);** |
| OperationModify | **OperationModify(int Previous, Task Revised);** |
| OperationUndo: | **OperationUndo().** |

***OperationHandler***

Attributes

|  |  |
| --- | --- |
| Variables | Explanation |
| List<Task> lastListedTasks; | List of the most recent tasks to record user data |
| Stack<Operation> undoStack; | Stack of operation for undo |
| Stack<Operation> redoStack; | Stack of operation for redo |
| Storage storageXML; | Instance of storage |
| Functions | Explanation |
| string Execute(Operation operation, ref List<Task> taskList) | Processes command from ui and return feedbacks |

Constructor

|  |  |
| --- | --- |
| Initializations | Explanation |
| lastListedTasks = new List<Task>(); | Creates instance of tasklist |
| undoStack = new Stack<Operation>(); | Creates instance of undostack |
| storageXML = new Storage(); | Creates instance of storage |

***Task:***

Constructor:

public TaskFloating(string TaskName)

public TaskDeadline(string TaskName, DateTime EndTime)

public TaskTimed(string TaskName, DateTime StartTime, DateTime EndTime)

***Storage:***

Constructor: public Storage().

Function: void WriteXML(TaskList taskList).

**Design descriptions**

**(class diagrams, sequence diagrams, notable algorithms ...)**

**Code examples**

**Instructions for testing**

Since no unit test has been written, simple test can be implemented using code which has been commented in the program.cs.

These codes can check add, display and delete function and get feedback in console.