**Assessment - My To-Do List**

*Read through this entire document very carefully before you start!*

# Problem Statement

Your goal is to create a to-do-list and associated operations using linked-lists in python. The To-Do list should have the capability of:

* 1. Adding/Creating a task
  2. Assigning a task-Id
  3. Removing a task
  4. Searching a task
  5. Marking a task completed
  6. Unmarking a task as completed
  7. List status showing all completed and incomplete tasks and associated details.

To solve this problem, create a linked-list of tasks and mark them as complete and incomplete. You'll be given a few commands in the input file.

## Operations:

* + 1. **def initiateToDoList(read\_input\_file):** This function reads the input file and creates a to-do list and all associated data structures and calls the necessary functions as mentioned in the input file.

Input: Input-File name with path. Output: None

* + 1. **def addTask(task\_string=””):** This function is called by initiateToDoList and creates a task along with a unique task-number and writes this task-number in the output file. Input: Task string.

Output: “ADDED:”<Unique Task-Number assigned to this task.>-<task-string>

* + 1. **def removeTask(task\_string=””, task\_number=””):** Description: This function is called by initiateToDoList and removes a task along with a unique task-number and writes this task number in the output file.

Input: Task string. OR task\_number.

Output: “REMOVED:”<Unique Task-Number assigned to this task.>-<task-string>

* + 1. **def searchTask(search\_string=””):** This function is called by initiateToDoList and find a task along with a unique task-number and writes this task-number in the output file. Input: Task string. OR task\_number.

Output: SEARCHED:<search-string>

<Task-Number>-<task-string>

* + 1. **def completeTask(task\_string=””, task\_number=””):** This function is called by initiateToDoList and marks a task along with a unique task-number as complete and writes this task-number in the output file.

Input: Task string. OR task\_number.

Output: “COMPLETED:”<Unique Task-Number assigned to this task.>-<task-string>

* + 1. **def incompleteTask(task\_string=””, task\_number=””):** This function is called by initiateToDoList and unmarks a task along with a unique task-number as complete and writes this task-number in the output file.

Input: Task string. OR task\_number.

Output: “UNCOMPLETED:”<Unique Task-Number assigned to this task.>-<task-string>

* + 1. **def statusTask():** This function is called by initiateToDoList and unmarks a task along with a unique task-number as complete and writes this task-number in the output file. Input:

Output: TASK STATUS:

<Task-Number assigned to this task.>-<task-string>

## Requirements:

* + - 1. Implement the above problem statement using **Python 3.7**
      2. Read the input from a file(**inputPS13.txt**), which contains the list of tasks and associated actions to be taken identified by relevant tags at the start of each line separated with a colon.
      3. You will output your answers to a file (**outputPS13.txt**) for each line.
      4. Perform an analysis for the features above and give the running time in terms of input size: n.

## Sample Input:

Input will be taken from the file(**inputPS13.txt**).

Add a Task: Complete Assignment. Add a Task: Return Library Books. Add a Task: Book a ticket.

Add a Task: Wish Birthday Remove Task: Wish Birthday

Mark Complete: Complete Assignment. Task Status:

Mark InComplete: Complete Assignment. Remove Task: Complete Assignment.

Add a Task: Call Home.

Add a Task: Read Algorithm book Remove Task: Call Home.

Search Task: book. Add a Task: wear mask,

*Note that the input/output data shown here is only for understanding and testing, the actual file used for evaluation will be different.*

## Sample Output:

Display the output in **outputPS13.txt**.

ADDED:TA3291-Complete Assignment. ADDED:TA8920-Return Library Book. ADDED:TA4836-Book a ticket.

COMPLETED:TA3291-Complete Assignment. TASK-STATUS:

Task-Number Task-String Task-Status

TA3291 Complete Assignment. C TA8920 Return Library Book. I TA4836 Book a ticket. I

UNCOMPLETED:TA3291-Complete Assignment. REMOVED:TA3291-Complete Assignment.

ADDED:TA3111-Call Home.

ADDED:TA3916-Read Algorithm book REMOVED:TA3111-Call Home.

SEARCHED:book

TA8920-Return Library Book. TA4836-Book a ticket.

ADDED:TA9374-wear mask,

*Note that the input/output data shown here is only for understanding and testing, the actual file used for evaluation will be different.*

# Deliverables

* 1. **InputPS13.txt** file used for testing
  2. **outputPS13.txt** file generated while testing
  3. **.py file** containing the python code. Create a single \*.py file for code. Do not fragment your code into multiple files.

# Instructions

* 1. It is compulsory to make use of the data structure(s) / algorithms mentioned in the problem statement.
  2. Ensure that all data structure insert and delete operations throw appropriate messages when their capacity is empty or full. Also ensure basic error handling is implemented.
  3. For the purposes of testing, you may implement some functions to print the data structures or other test data. But all such functions must be commented before submission.
  4. Make sure that your read, understand, and follow all the instructions
  5. Ensure that the input, prompt and output file guidelines are adhered to. Deviations from the mentioned formats will not be entertained.
  6. The input, prompt and output samples shown here are only a representation of the syntax to be used. Actual files used to evaluate the submissions will be different. Hence, do not hard code any values into the code.
  7. Run time analysis is to be provided in asymptotic notations and not timestamp based runtimes in sec or milliseconds.

# Instructions for use of Python:

* + 1. Implement the above problem statement using Python 3.7+.
    2. Use only native data types like lists and tuples in Python, do not use dictionaries provided in Python. Use of external libraries like graph, numpy, pandas library etc. is not allowed. The purpose of the assignment is for you to learn how these data structures are constructed and how they work internally.
    3. Create a single \*.py file for code. Do not fragment your code into multiple files.
    4. Do not submit a Jupyter Notebook (no \*.ipynb). These submissions will not be evaluated.
    5. Read the input file and create the output file in the root folder itself along with your .py file. Do not create separate folders for input and output files.

# Deadline

* 1. The strict deadline for submission of the assignment is **Friday, 13 Jan 2023**
  2. Late submissions will not be evaluated.

# \*\*\*\*\*\*\*\*\*\* All the Best \*\*\*\*\*\*\*\*\*\*