# ICT208 Algorithms and Data Structures

# Assessment 2: Case Study 1 – To Do List

|  |  |  |  |
| --- | --- | --- | --- |
| **Linked Lists- Case study 1**  In this assessment, students will analyse a case and develop appropriate data structures and finally write code. | Due in week 7 | 40% | 1,2,3,4 |

**Case study:**

In this assessment you are required to develop a to-do list application in C++. The list must be implemented by linked list. A linked list is a linear data structure, in which the elements are not stored at contiguous memory locations. The elements in a linked list are linked using pointers as shown in the below image:

Diagram

Description automatically generated

Each element in the link is called a node which is composed of two parts:

1. Data stored by the node
2. Link pointed to the next node in the linked list

The final application must meet the following functional requirements:

1. The user can create unlimited to-do lists. Each list must have a unique name (i.e., two lists with same name are not allowed).
2. User can insert a new list at the start and delete the first list.
3. User can insert an item in a list specified by the user. User inputs the name of the list to insert the item. If the list exists, then the user has the option to either insert the item at the end of the list or after an item at an index specified by the user. If the list does not exist an appropriate message is shown.
4. User can delete an item from a list specified by the user. User inputs the name of the list to insert the item. If the list exists, then the user is prompted to enter the index of the item to delete. If the list does not exist an appropriate message is shown.
5. User can see the names of stored lists.
6. User must be able to display the items stored in a to-do list specified by the user.

The to-do list has the following functional requirements:

1. Each node in the list stores a string that is entered by the user. The stored string describes the user task.
2. There must be a way to delete last item from the list.
3. There must be a way to insert an item at the end of the list.
4. There must be a way to insert and delete an item at a specific index. You can assume that the first item in the list has an index 0.

Please note to score maximum marks you must implement all the requirements. However, you can still obtain a maximum of 70% marks if your application allows users to create only a single to-do list.

The following reference sites can be of helpful to enhance your understanding of Linked List class structure based on Node and associated methods:

1. <https://www.geeksforgeeks.org/data-structures/linked-list/>
2. <https://www.freecodecamp.org/news/data-structures-explained-with-examples-linked-list/>
3. <https://beginnersbook.com/2013/12/linkedlist-in-java-with-example/>

Submission requirement: upload the LinkedList.cpp to Moodle by week 7.

**Marking rubric**

|  |  |  |
| --- | --- | --- |
| Marking criteria | Lecturer expectation | Marks |
| Unlimited lists | You are required to develop appropriate data structures.  The method that checks lists must have unique name is correctly implemented. | 10  (5 + 5) |
| Declaration of node | Node contains correct data members with data types. | 15 |
| Insertion | Insert at front, end and after specific node methods are correctly implemented. | 30  (5+ 12.5+ 12.5) |
| Deletion | Delete first, last and a specific item method are correctly implemented. | 30  (5 + 12.5 + 12.5) |
| Search | The method that searches the nested list for the list is correctly implemented. | 5 |
| Display | Method that displays the names of the stored lists is correctly implemented  Method that displays a to-do list is correctly implemented. | 10  (5 + 5) |