

2022/2023

Lab 2: Singly Linked list

FAKULTI TEKNOLOGI KEJURUTERAAN KELAUTAN DAN INFORMATIK

**DATA STRUCTURE & ALGORITHM**



**VERSION 1**

**Name:** ARUN MUGILAN  
**Matric Number:** S63746   
**Lab: MP3**    
**Date:** 8/11/2022

STUDENT INFORMATION

**Please fill in your personal details:**

**Name: ARUN MUGILAN**

**MATRIC NUMBER:S63746**

**GROUP:K2**

**LAB:MP3**

**DATE:8/11/2022**

Table of Contents

[INSTRUCTIONS 1](#_Toc16069964)

[TASK 1: IMPLEMENTATION OF SINGLY LINKED LIST 2](#_Toc16069965)

[TASK 2: APPLICATION OF SINGLY LINKED LIST](#_Toc16069966) 7

# INSTRUCTIONS

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Fakulti Teknologi Kejuruteraan Kelautan dan Informatik, Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedar manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual.

This laboratory manual is for use by the students of the Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu (UMT) only. It is not permissible to print and distribute this manual without the official authorisation of the author.

Please follow step by step as described in the manual.

# TASK 1: implementation of singly linked list

## Objective

In this lab, we will learn the following topics:

* A linked list
* Implementation of linked list using java
* Insert Node at the beginning of the list
* Traverse List
* Delete Node from a list

## task description

In each of the topics, students should implement the tasks step by steps in order to have better understanding of singly linked list.

## Estimated Time

[60 Minutes]

### DEfinition of linked list

A linked list is just a chain of nodes, with each subsequent node being a child of the previous one. Many programs rely on linked lists for their storage because these don't have any evident restrictions. There is no limit (other than the amount of memory) on the number of elements they can store.

### Singly linked lists

**Basics:**

* A singly linked list is a concrete data structure consisting of a sequence of nodes
* It has a head node pointer indicating the first node in list.
* It could have optionally a tail pointer node indication the last node in list.
* Each node stores
  + Element (data)
  + Link to the next node

**Operations:**

The common operations of Singly linked list are:

1. Insertion (or Add):

* Add first
* Add last
* Add middle (after existing node); example:

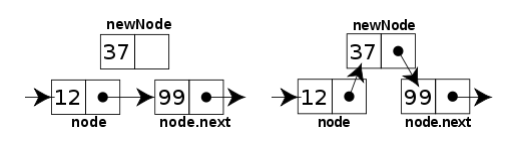


Figure 1: An example of insertion a new node in the middle of linked list

1. Deletion (or Remove):

* Delete first
* Delete last
* Delete after existing node; example:

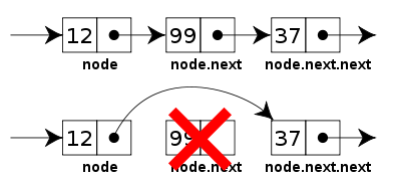


Figure 2: An example of insertion a new node in the middle of linked list

### Some common handling methods of Singly/Doubly linked list are:

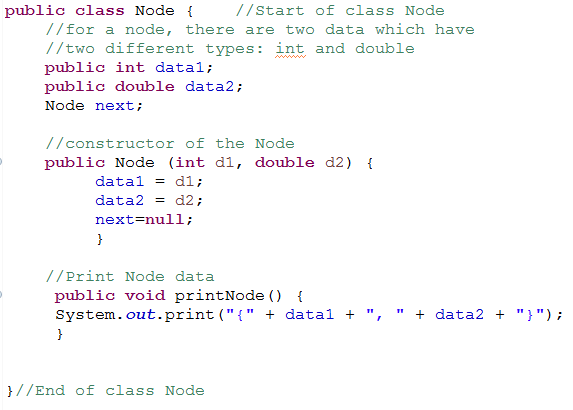
1. Print (Or Show):
2. Print all list elements
3. Print certain node
4. Search for an element
5. Find list size (if no size variable in list class)
6. Reverse the linked list [step 2]

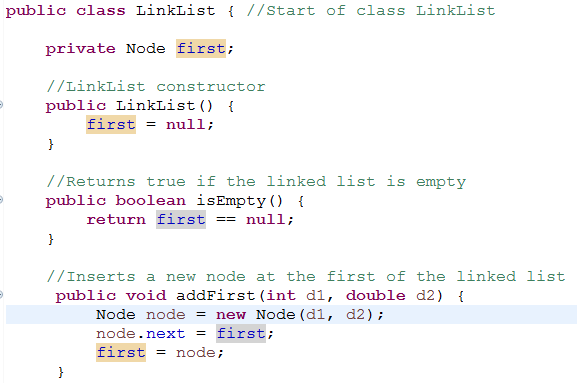
### ACTIVITIES

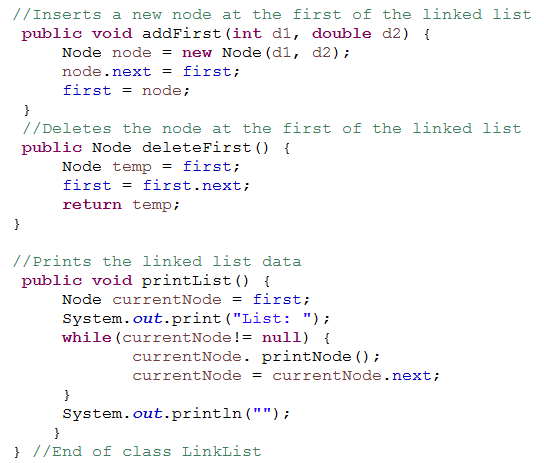
**Activity 1:**

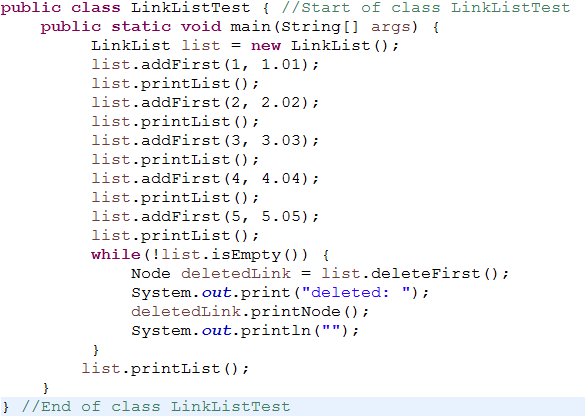
Apply and test the Linked List implementation bellow:

// the code below is a simple example of a linked list that inserts a new link at the beginning of the list, deletes from the beginning of the list and loops through the list to print the links contained in it.





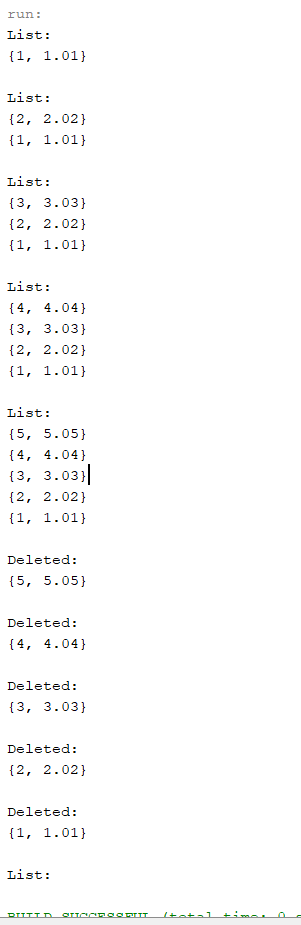




Activity 2: Compile the program. Execute the program and record the results.

Upload the screenshot using the control box provided below:

**Answer:**

****

# TASK 2: APPLICATION of singly linked list

## task description

Based on the previous code, you may use it to complete the following tasks.

## Estimated Time

[120 Minutes]

1. Write the Node class consists of two components of a node (i.e.: element, next), with a default construct and a constructor that accepts an item assigned to the initially declared element variable.
2. Write a class called **MyLinkedList**. The class should have the following:
   1. Default constructor
   2. Nodes for head and tail
3. Implement the following methods (some methods had already been shown in Task 1)
   1. public void addFirst(Node e)
   2. public void addLast(Node e)
   3. public void add(int index, Node e)
   4. public Node removeFirst()
   5. public Node removeLast()
   6. public void printList()
   7. public void reverse()
4. Write a test program called TestLinkedList that creates a list from MyLinkedList class. Using the methods in (3) , do the following:
   1. Add these elements to the linked list using addFirst() method according to the order : a, b, c, d, e
   2. Print all the elements in the list.
   3. Delete the last value.
   4. Print current list.
   5. Add ‘f’ at the second position in the linked list.
   6. Print current list.
   7. Add ‘g’ at the end of the list.
   8. Print current list.
   9. Delete the first element in the current list.
   10. Print current list.
   11. Reverse the list.
   12. Print the list.
5. Compile the program. Execute the program and record the results.

Table

Description automatically generated

Table

Description automatically generated

Graphical user interface, table

Description automatically generated

1. Submit your codes at epembelajaran. Please ensure your codes are submitted to the correct group.