**Doubly Linked List**

This section will guide you to:

* Write a program in Java to traverse a doubly linked list in the forward and backward directions
* Use Eclipse (the popular text editor for Java programs)
* Push code to Git

This lab has three subsections, namely:

* + 1. Creating a new project in Eclipse
    2. Writing the program in Java
    3. Pushing the created files to Git

**Step 4.7.1:** Creating a new project in Eclipse

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next
* Type in any project name and click on Finish
* Select your project and go to File -> New -> Class
* Enter **DLL** in class name, check the checkbox **public static void main(String args)**, and click on Finish

**Step 4.7.2:** Writing the program in Java

You need to write the program to traverse a doubly linked list in the forward and backward directions.

public class DLL

{

     Node head;

class Node

{

         int data;

         Node prev;

         Node next;

Node(int d)

{

data = d;

}

     }

public void push(int new\_data)

     {

Node new\_Node = new Node(new\_data);

new\_Node.next = head;

         new\_Node.prev = null;

if (head != null)

             head.prev = new\_Node;

head = new\_Node;

     }

public void InsertAfter(Node prev\_Node, int new\_data)

     {

if (prev\_Node == null)

{

             System.out.println("The given previous node cannot be NULL ");

             return;

         }

Node new\_node = new Node(new\_data);

new\_node.next = prev\_Node.next;

prev\_Node.next = new\_node;

new\_node.prev = prev\_Node;

if (new\_node.next != null)

             new\_node.next.prev = new\_node;

     }

     void append(int new\_data)

     {

Node new\_node = new Node(new\_data);

   Node last = head;

new\_node.next = null;

if (head == null)

{

             new\_node.prev = null;

             head = new\_node;

             return;

         }

while (last.next != null)

             last = last.next;

last.next = new\_node;

new\_node.prev = last;

     }

public void printlist(Node node)

     {

         Node last = null;

         System.out.println("Traversal in forward Direction");

         while (node != null)

{

             System.out.print(node.data + " ");

             last = node;

             node = node.next;

         }

         System.out.println();

         System.out.println("Traversal in reverse direction");

         while (last != null)

{

             System.out.print(last.data + " ");

             last = last.prev;

         }

     }

public static void main(String[] args)

     {

DLL dll = new DLL

dll.append(6);

dll.push(7);

dll.push(1);

dll.append(4);

dll.InsertAfter(dll.head.next, 8);

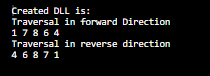
   System.out.println("Created DLL is: ");

         dll.printlist(dll.head);

     }

}

**Output:**



What is big O for inserting a value in a linked list? We are traversing a linked list here.