

School of Engineering and Technology

## Department of Computer Science and Engineering

Jain Global Campus, Kanakapura Taluk - 562112 Ramanagara District, Karnataka, India

### 2018-2019

**A Project Report on**

**“DOUBLY LINKED LIST”**

**Submitted in partial fulfilment for the requirements of Computer Graphics and Visualization Laboratory(16CIC505L)**

**BACHELOR OF TECHNOLOGY IN**

**CO M P U T E R SC I E N C E A N D EN GI N E E R I N G**

**Submitted by**

**DEVRAJ DHAR 16BTLCS001**

**YASH KUMAR JAIN**

**16BTTCS003**

**MANASH BHATTACHARJEE 16BT6CS076**

**Under the guidance of Mrs. Deepa.T.P., Ms. Anusha.S**

Assistant Professor

Department of Computer Science and Engineering

School of Engineering & Technology

**Jain University**



School of Engineering & Technology

## Department of Computer Science and Engineering

Jain Global campus Kanakapura Taluk - 562112 Ramanagara District Karnataka, India

# CERTIFICATE

This is to certify that the project work titled **“DOUBLY LINKED LIST”** is carried out by **Devraj Dhar(16BTLCS001), Yash Kumar Jain (16BTTCS003), Manash Bhattacharjee (16BT6CS076),** a bonafide students of 5th Semester Computer Science and Engineering, Bachelor of Technology at the School of Engineering & Technology, Jain University, Bangalore in partial fulfillment for the requirements of Computer Graphics and Visualization Laboratory (16CIC505L) during the year **2018-2019**.

### Mrs. Deepa.T. P Ms. Anusha.S Dr. Rajesh.S. L

Assistant Professor Dept. of ECE,

School of Engineering & Technology,

Jain University

Date:

Assistant Professor Dept. of ECE,

School of Engineering & Technology,

Jain University

Date:

H.O.D, CSE

School of Engineering & Technology, Jain University Date:

Name of the Examiner Signature of Examiner 1.

2.

# DECLARATION

We, **Devraj Dhar(16BTLCS001), Yash Kumar Jain (16BTTCS003), Manash Bhattacharjee(16BT6CS076)** are student’s of fifth semester B.Tech in **Computer Science and Engineering**, at School of Engineering & Technology,

**Jain University**, hereby declare that the project titled **“Doubly Linked List”** has been carried out by us and submitted in partial fulfilment for the requirements of Computer Graphics and Visualization Laboratory (16CIC505L) in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2018-2019**. Further, the matter presented in the project has not been submitted previously by anybody for the award of any degree or any diploma to any other University, to the best of our knowledge and faith.

Signature

Name: Devraj Dhar

USN : 16BTLCS001

Name: Yash Kumar Jain USN : 16BTTCS003

Name: Manash Bhattacharjee

USN : 16BT6CS076

PLACE : BANGALORE DATE : 06-12-18

**ACKNOWLEDGEMENT**

*It is a great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project work.*

*First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Jain University for providing us with a great opportunity to pursue our Bachelor’s Degree in this institution.*

*In particular we would like to thank* ***Dr. Hariprasad.S.A****,* ***Director****,* ***School of Engineering & Technology****,* ***Jain University*** *for his constant encouragement and expert advice. It is a matter of immense pleasure to express our sincere thanks to* ***Dr. Rajesh.S.L.***

***Head of the department****,* ***Computer Science and Engineering****,* ***Jain University,*** *for providing right academic guidance that made our task possible.*

*We would like to thank our guides Mrs. Deepa.T.P. and Ms. Anusha.S,* ***Assistant Professor****,* ***Dept. of Computer Science and Engineering****,* ***Jain University****, for sparing their valuable time to extend help in every step of our project work, which paved the way for smooth progress and fruitful culmination of the project.*

*We are also grateful to our family and friends who provided us with every requirement throughout the course.*

*We would like to thank one and all who directly or indirectly helped us in completing the Project work successfully.*

***i***

# ABSTRACT

Doubly Linked Lists are a type of data structure. In a doubly linked list, each node in the list stores the contents and the pointer or reference to the next node in the list. It does not store the pointer or reference to the previous node. To store a single linked list, only the reference or pointer to the first node in that list must be stored. The last node in a double linked list points to nothing. In this project nodes are created depending upon the options selected by the user (front, rear) and whole scenario is shown by full animation using OpenGl. The user can also delete the nodes from both front and rear and the nodes are deleted in the animation. Hence, the final list is shown by the animating window which is created by OpenGl library. The front node points or references to the next node and the next node references or points to its next node and hence the doubly linked list is formed.

**TABLE OF CONTENTS**

List of Figures i

Nomenclature used i

### Chapter 1 : Introduction I

* 1. **Introduction to Project 1**
  2. [**Problem Definition 1**](#_TOC_250002)
  3. [**Objectives 1**](#_TOC_250001)

Chapter 2 : Project Requirements II

* 1. HARDWARE AND SOFTWARES USED 2
  2. [TOOL DESCRIPTION 2](#_TOC_250000)

**Chapter 3 : Project Design**

* 1. **ALGORITHM**
  2. **FLOWCHART**
  3. **BLOCK DIAGRAMS**

**CHAPTER 4 : IMPLEMENTATION**

* 1. **C PROGRAMMING**
  2. **OPENGL**
  3. **LIST OF MAJOR FUNCTIONS USED**
  4. Pseudocodes

### III 3

**4**

**4**

**V**

5

5

5

6

### Chapter 5 : Results and Discussion VII

**CONCLUSIONS AND FUTURE SCOPE X**

**REFERENCES** XI

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| Fig. No. | Description of the figure | Page No. |
| 3.1 | Proposed Flow Chart | 4 |
| 3.2 | Proposed Block Diagram | 4 |
| 4.1(b) | Creation of Single Node | 7 |
| 4.2(a) | Inserting Front | 7 |
| 4.2(b) | Inserting Rear | 8 |
| 4.2(c) | Deletion Front | 8 |
| 4.3 | Deletion Rear | 9 |

**NOMENCLATURE USED**

|  |  |
| --- | --- |
| LCD | Liquid Crystal Display |
| GPRS | General Packet Radio Service |
| AT | Attention |
| GUI | Graphical User Interface |
| USB | Universal Serial Bus |
| SIM | Subscriber Identification Module |
| LF | Line Feed |
| CR | Carriage Return |
| PHP | Pre-Processor Hyper text |
| MySQL | My Structured Query Language |

**Chapter 1**

**Introduction**

### Description

This Doubly Linked List animation is an demonstration of how Linked List Data Structure is used and it is implemented in OpenGl .It is a type of Data Structure and it consists of nodes and each node contains the data of the node what it is storing and the address of the next node and new nodes can be inserted either by front side or by rear side and also nodes can be deleted either by front side or rear side.

### Problem definition

This system aims to solve the problems of the traditional method of explanation of Data Structure (Linked List) and make the teaching more effective and advanced. The main objective of this project is to give a demonstration on Doubly Linked List which is one of the main Data Structures.

### Objectives

The objectives of this project are to produce a animated system that:

* + - Attempts to solve the problems posed by the students or learners while learning the concept of Doubly Linked List.
    - Allows users to enter the nodes and delete the nodes according to the choice they choose.
    - Allows user to see the main result in animated format.

### Minimum System Requirements

This web application requires the following minimum hardware and software specifications for it to run:

* + - 1 GB RAM
    - Intel core i3 processor
    - Microsoft Visual Studio installed.
    - 10 GB storage or Higher.
    - Windows 7 64 bit or Higher.

1

# Chapter 2:

**Project Requirements**

### Hardware and Software used.

This project requires the following minimum hardware and software specifications for it to run:

* + - 1 GB RAM
    - Intel core i3 processor
    - Microsoft Visual Studio installed.
    - 10 GB storage or Higher.
    - Windows 7 64 bit or Higher.

### Tool Description:

**Microsoft Visual Studio**: Microsoft Visual Studio is an [integrated development](https://en.wikipedia.org/wiki/Integrated_development_environment) [environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) from [Microsoft](https://en.wikipedia.org/wiki/Microsoft). It is used to develop [computer programs](https://en.wikipedia.org/wiki/Computer_program), as well as [websites,](https://en.wikipedia.org/wiki/Web_site) [web apps,](https://en.wikipedia.org/wiki/Web_app) [web services](https://en.wikipedia.org/wiki/Web_service) and [mobile apps](https://en.wikipedia.org/wiki/Mobile_app). Visual Studio uses Microsoft software development platforms such as [Windows API](https://en.wikipedia.org/wiki/Windows_API), [Windows Forms](https://en.wikipedia.org/wiki/Windows_Forms), [Windows](https://en.wikipedia.org/wiki/Windows_Presentation_Foundation) [Presentation Foundation,](https://en.wikipedia.org/wiki/Windows_Presentation_Foundation) [Windows Store](https://en.wikipedia.org/wiki/Windows_Store) and [Microsoft Silverlight](https://en.wikipedia.org/wiki/Microsoft_Silverlight). It can produce both [native code](https://en.wikipedia.org/wiki/Machine_code) and [managed code](https://en.wikipedia.org/wiki/Managed_code). Visual Studio includes a [code](https://en.wikipedia.org/wiki/Code_editor)

[editor](https://en.wikipedia.org/wiki/Code_editor) supporting [IntelliSense](https://en.wikipedia.org/wiki/IntelliSense) (the [code completion](https://en.wikipedia.org/wiki/Code_completion) component) as well as [code](https://en.wikipedia.org/wiki/Code_refactoring) [refactoring.](https://en.wikipedia.org/wiki/Code_refactoring) [The integrated debugger](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio_Debugger) works both as a source-level debugger and a machine-level debugger. Other built-in tools include a [code profiler](https://en.wikipedia.org/wiki/Profiling_(computer_programming)), forms designer for building [GUI](https://en.wikipedia.org/wiki/GUI) applications, web designer, class designer and database schema designer.

2

# Chapter 3:

**Project Design**

### Algorithm

record Node

{

data; // The data being stored in the node

Node next // A reference to the next node, null for last node

}

record List

{

Node firstNode // points to first node of list; null for empty list

}

function insertAfter(Node node, Node newNode) // insert newNode after node newNode.next := node.next

node.next := newNode

function insertBeginning(List list, Node newNode) // insert node before current first node newNode.next := list.firstNode

list.firstNode := newNode

function removeAfter(Node node) // remove node past this one obsoleteNode := node.next

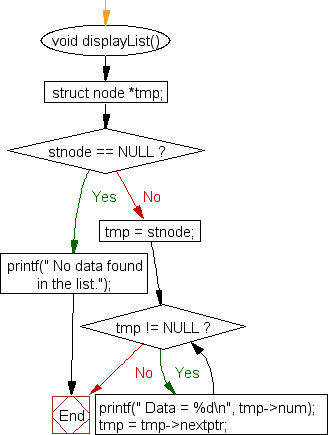
node.next := node.next.next destroy obsoleteNode

function removeBeginning(List list) // remove first node obsoleteNode := list.firstNode

list.firstNode := list.firstNode.next // point past deleted node destroy obsoleteNode

3

### Flowchart



**Fig 3.1: Flowchart of Doubly Linked List.**

* 1. **Block Diagram**



**Fig 3.2: Block Diagram**

4

# Chapter 4:

**Implementation**

This project of Doubly Linked List was implemented using the following technologies:

### C Programming

C is a general-purpose, imperative computer programming language, supporting structured programming, lexical variable scope and recursion, while a static type system prevents many unintended operations. By design, C provides constructs that map efficiently to typical machine instructions, and therefore it has found lasting use in applications that had formerly been coded in assembly language, including operating systems.

### OpenGL

Open Graphics Library (OpenGL) is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU), to achieve hardware- accelerated rendering. The OpenGL specification describes an abstract API for drawing 2D and 3D graphics. Although it is possible for the API to be implemented entirely in software, it is designed to be implemented mostly or entirely in hardware.

### List of Major Functions used:

* + - insert\_front (int item, NODE first) :

This function is used to insert a newly created node at the front end of the list hence making a new list and giving the newly created node the address of the next node.

* + - insert\_rear (int item, NODE first) :

This function is used to insert a newly created node at the rear end of the list hence making a new list and the node which is inserted at the rear end of the list does not contain any address value of any node hence it is null.

* + - NODE delete\_front (NODE first) :

This function is used to delete the first node of the list, the deletion contain deletion of the value and the deletion of the address value of the node and forming a new list.

* + - NODE delete\_rear (NODE first) :

This function is used to delete the last node of the list.

5

### Pseudocode

struct LinkedList{ int data;

struct LinkedList \*next;

};

typedef struct LinkedList \*node; //Define node as pointer of data type struct LinkedList node createNode(){

node temp; // declare a node

temp = (node)malloc(sizeof(struct LinkedList)); // allocate memory using malloc() temp->next = NULL;// make next point to NULL

return temp;//return the new node

}

node addNode(node head, int value){

node temp,p;// declare two nodes temp and p

temp = createNode();//createNode will return a new node with data = value and next pointing to NULL.

temp->data = value; // add element's value to data part of node if(head == NULL){

head = temp; //when linked list is empty

}

else{

p = head;//assign head to p while(p->next != NULL){

p = p->next;//traverse the list until p is the last node.The last node always points to NULL.

}

p->next = temp;//Point the previous last node to the new node created.

}

return head;

}

6

# Chapter 5:

**Results and Discussions:**

### Creating a single node

Figure 5.1 shows the single node created during the execution of the program using OpenGL.

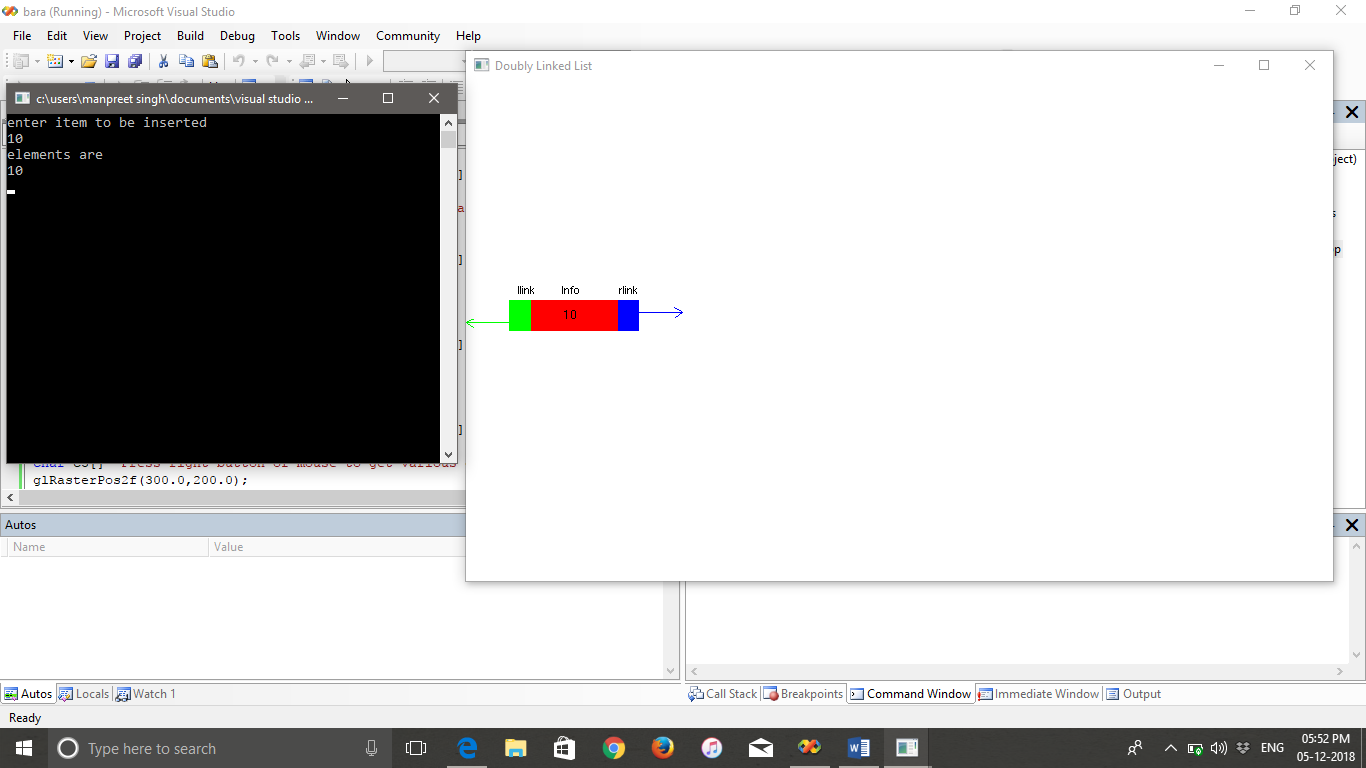


Fig. 5.1

Creation of Single node.

### 7

### 

### Inserting node from front

Figure 5.2 shows the single node inserted in front end during the execution of the program using OpenGL.

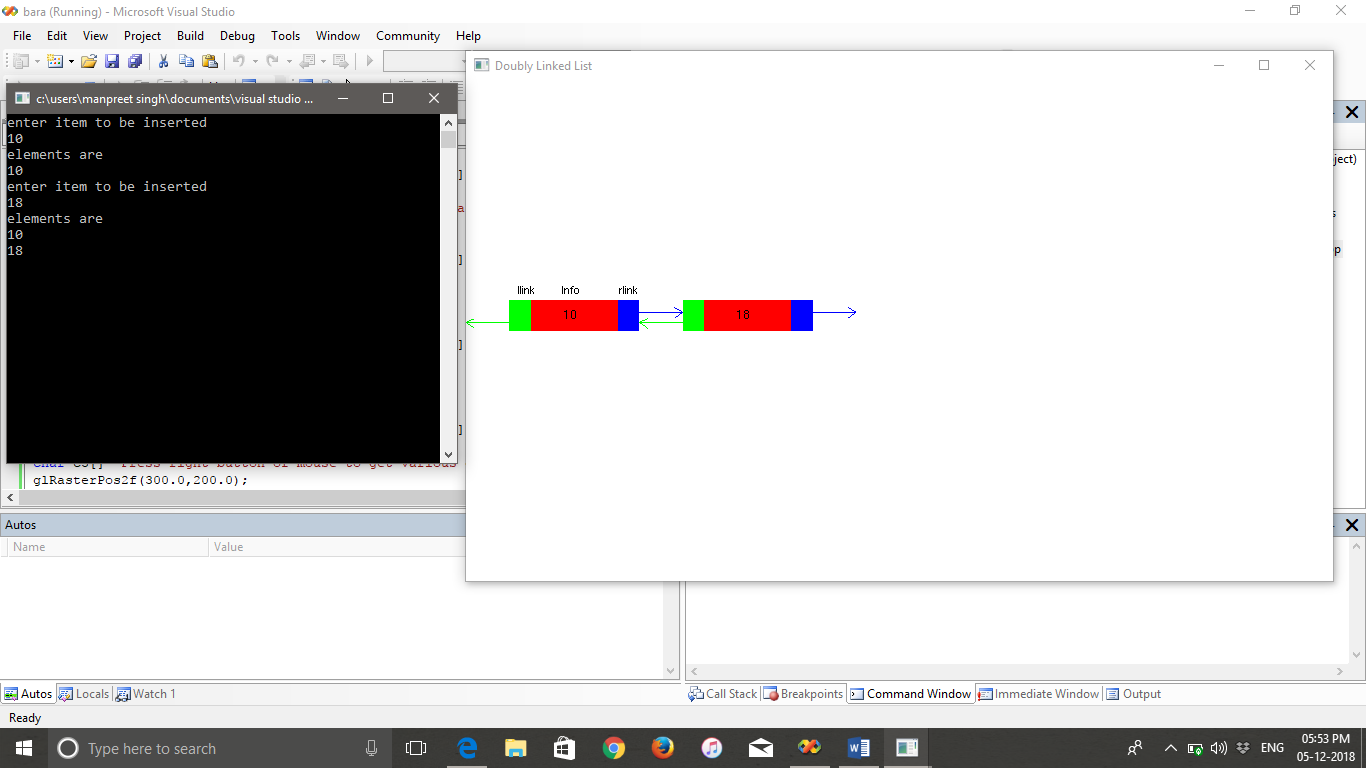


Fig. 5.2

Inserting Front.

### Inserting node from rear

Figure 5.3 shows the single node inserted in rear end during the execution of the program using OpenGL.

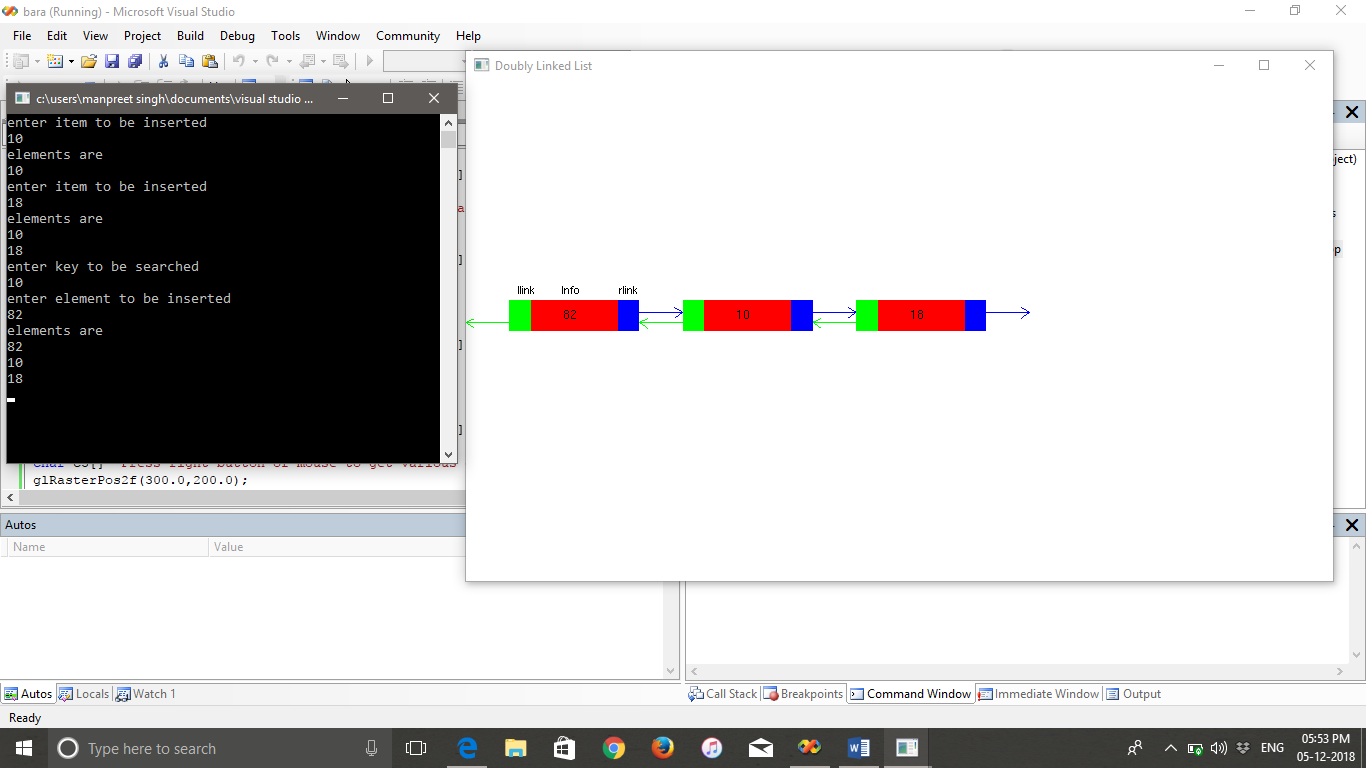


Fig. 5.3

Inserting Rear.

### Deleting node from front

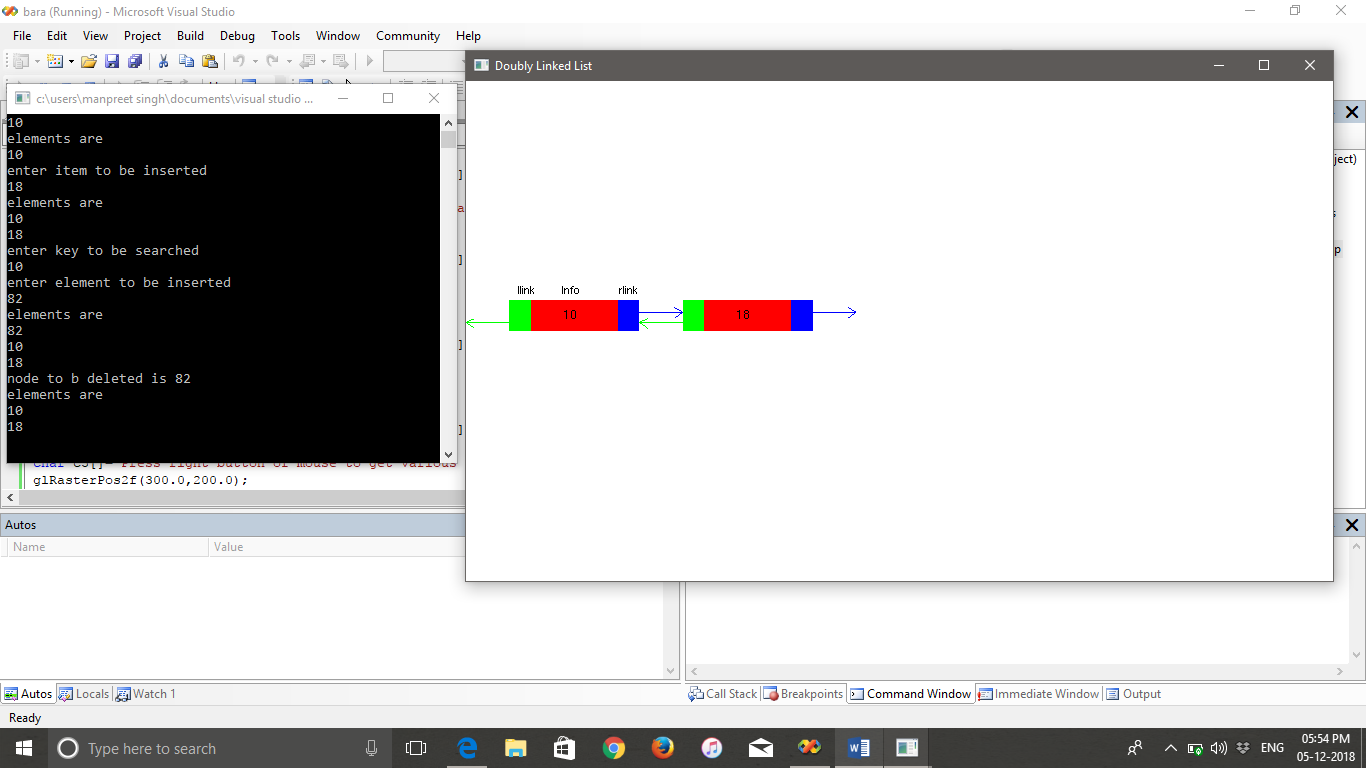
Figure 5.4 shows the single node deleted from front end during the execution of the program using OpenGL. 

Fig. 5.4

Delete Front.

8

### Deleting node from rear

Figure 5.5 shows the single node deleted from raer end during the execution of the program using OpenGL.

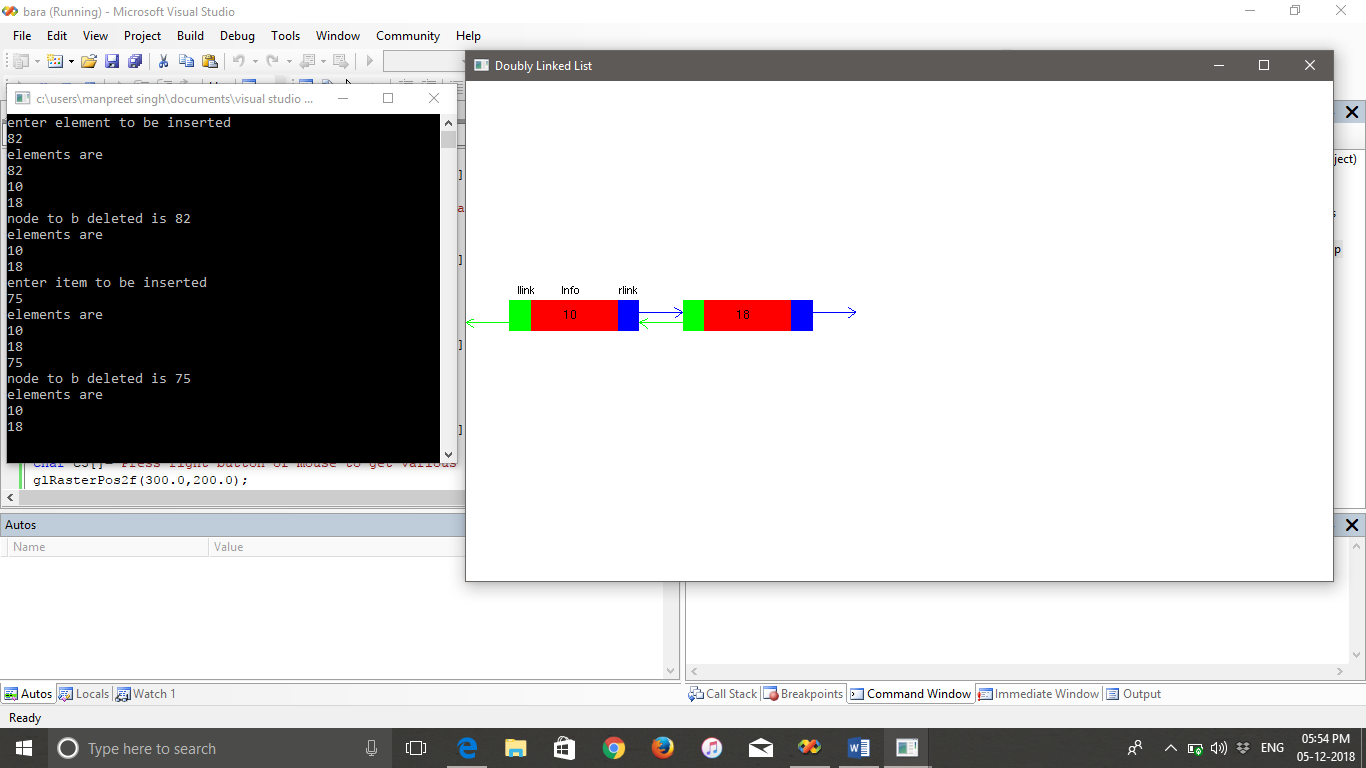


Fig. 5.5

Deleting Rear.

9

# Chapter 6:

**Conclusion**

The demonstration of Doubly linked list hopes to remove the redundancy and difficulty faced by the students or learner while understanding the concept of Doubly Linked List. To conclude, the demonstration of Doubly Linked List is easy to use, time-saving and immensely convenient.

### 6.1 Future Scope

This project has potential for being developed further in terms of making it more full- fledged. Animations and Graphics can be made better and there is lot of future scope in the animation for teaching and explaining the topics or presentation A better user interface could always be worked on and iterated in the future to increase learner satisfaction.

10

# REFERENCES

1. OpenGL-Projects, “JavaScript Reference” <https://www.openglprojects.com/jsref/>
2. Amos Q. Haviv “Building an Express opengl project”, MEAN OpenGL, Chapter 3, pages 49 – 77
3. Amos Q. Haviv “Introduction to C Programming”, MEAN C Programming, Chapter 2, pages 29 – 48
4. Avi Silberschatz, Henry F. Korth “Advanced Open GL”, An approach to OpenGL, Chapter 5 pages 157 – 216.

11