



Daffodil
International
University

Daffodil International University

Daffodil Smart City (DSC), Savar, Dhaka

Department of Computing and Information System

Semester Project on

Data Structure [CIS - 122]

Semester Fall-2024

Prepared By	Submitted To
Name: Abdullah ID: 241-16-008 Batch: 19 (A) Department of CIS	Name: Mr. Md. Mehedi Hassan Designation: Lecturer Department of CIS Daffodil International University

Acknowledgement

I express my heartfelt gratitude to the Almighty Allah (SWT) for guiding me throughout this project and providing me the strength to complete it.

I would like to extend my sincere appreciation to my teacher, Md. Mehedi Hassan Sir, for providing me with the opportunity to work on this project and for his valuable guidance and feedback that have helped me grow and learn.

I would also like to thank my parents for their unwavering support, encouragement, and motivation throughout my academic journey. Their love and belief in me have been my biggest source of inspiration.

I am grateful to Daffodil International University for providing me with an environment that has helped me develop my knowledge and skills.

Lastly, I would like to dedicate this project to those who are differently abled and have a thirst for knowledge. Their determination and resilience are a true inspiration to us all.

Contents

Theory Part:

Task 1.....	4
Q1.....	4
Q2.....	8
Task 2.....	9
Q3.....	9
Q4.....	13
Q5.....	15
Task 3.....	16
Q6.....	16

Lab Part:

Task 1.....	16
QL1.....	16
QL2.....	20
Task 2.....	25
QL3.....	25

Theory Part

Task – 1

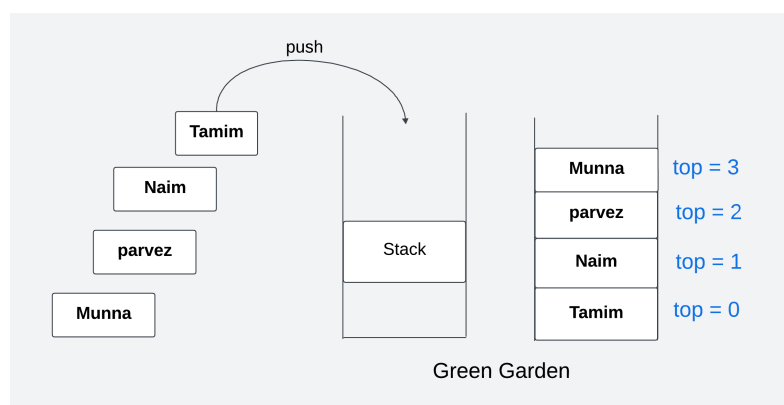
Q1. If **Tamim**, **Naim**, **Munna**, and **Parvez** are paying 110 Tk, 120 Tk, 140 Tk and 135 Tk respectively, in the green garden, therefore, implement the data structure (Campus Store) using another data structure (Green Garden). If the implementation is not possible, please explain the reason. [Maximum size of data structures is 4]

Ans:

The implementation is possible. Green Garden implement in **Stack** and Campus Store implement in **Queue**.

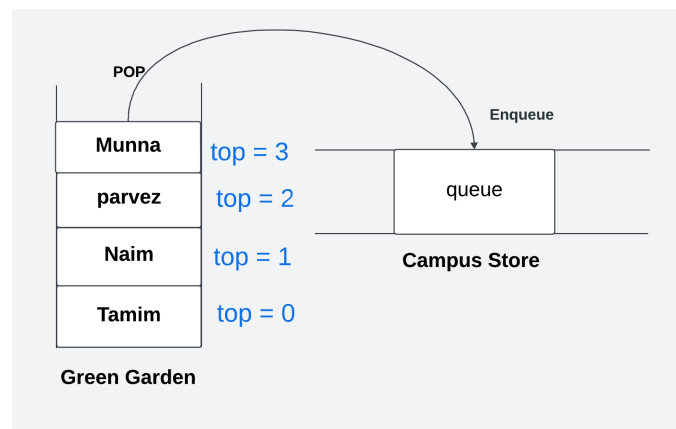
Green Garden

Tamim takes his lunch first, second **Naim**, third **Parvez** and **Munna** is the last and pay in the reverse order **Munna**, **Parvez**, **Naim**, **Tamim**. This looks like a **stack** behaviour (**LIFO - last in, first out**).



Munna makes his payment first in Green Garden and enters the campus store **first**. **Parvez** pays after **Munna** and enters the campus store second. The others

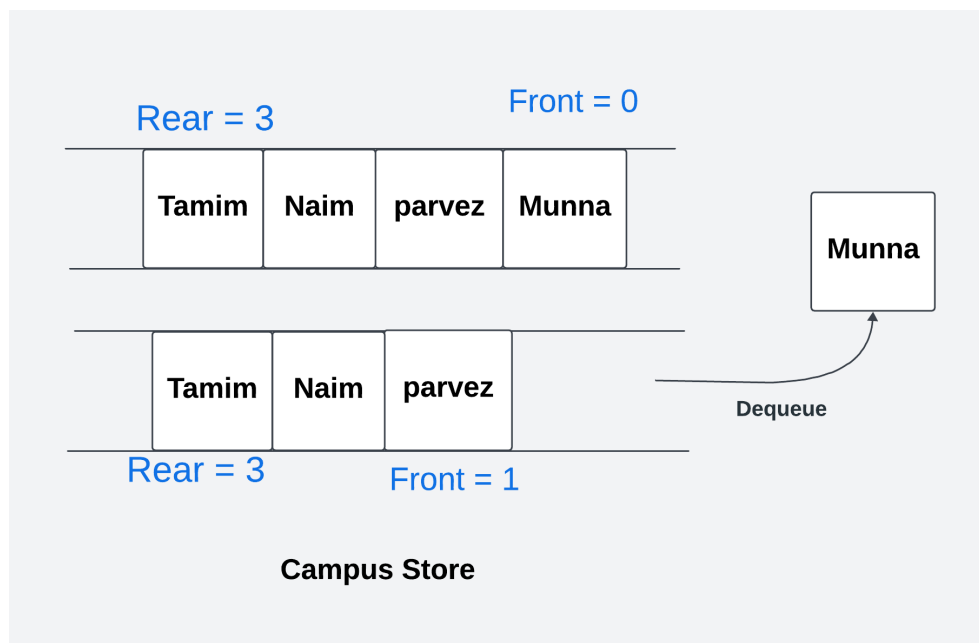
pay sequentially, each entering after **Parvez**. **Tamim** pays last and enters the store **last**.



Campus Store

A **queue** is an ordered list where elements are inserted at the REAR and deleted from the FRONT, following the **FIFO (First In, First Out)** principle.

Munna enters the campus store first, followed by **Parvez**, then **Naim**, and finally **Tamim**, who enters last. This entry order can be represented using queue operations.

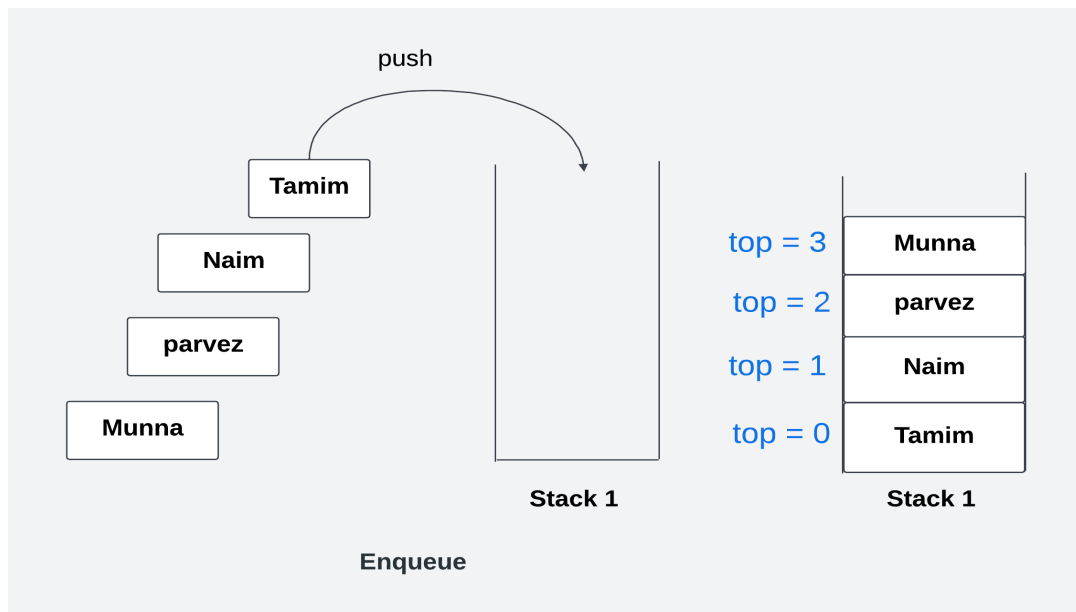


Yes, the implementation is possible. Implement **Queue**(Campus Store) using **Stack**(Green Garden).

Enqueue Operation (Entering the Campus Store in order):

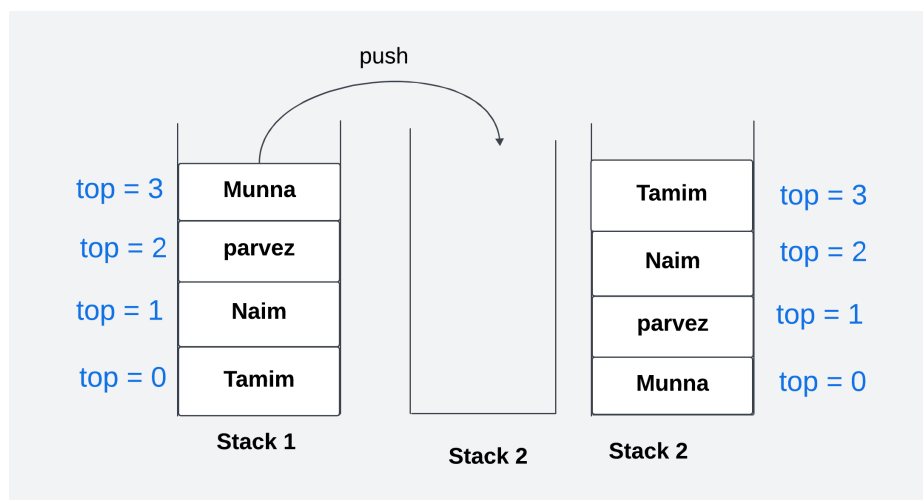
- When a new element is added to the queue, push it directly onto **stack1**.
- This keeps the elements in the order they arrive, with the newest element on top of **stack1**.

Tamim \Rightarrow Naim \Rightarrow Parvez \Rightarrow Munna

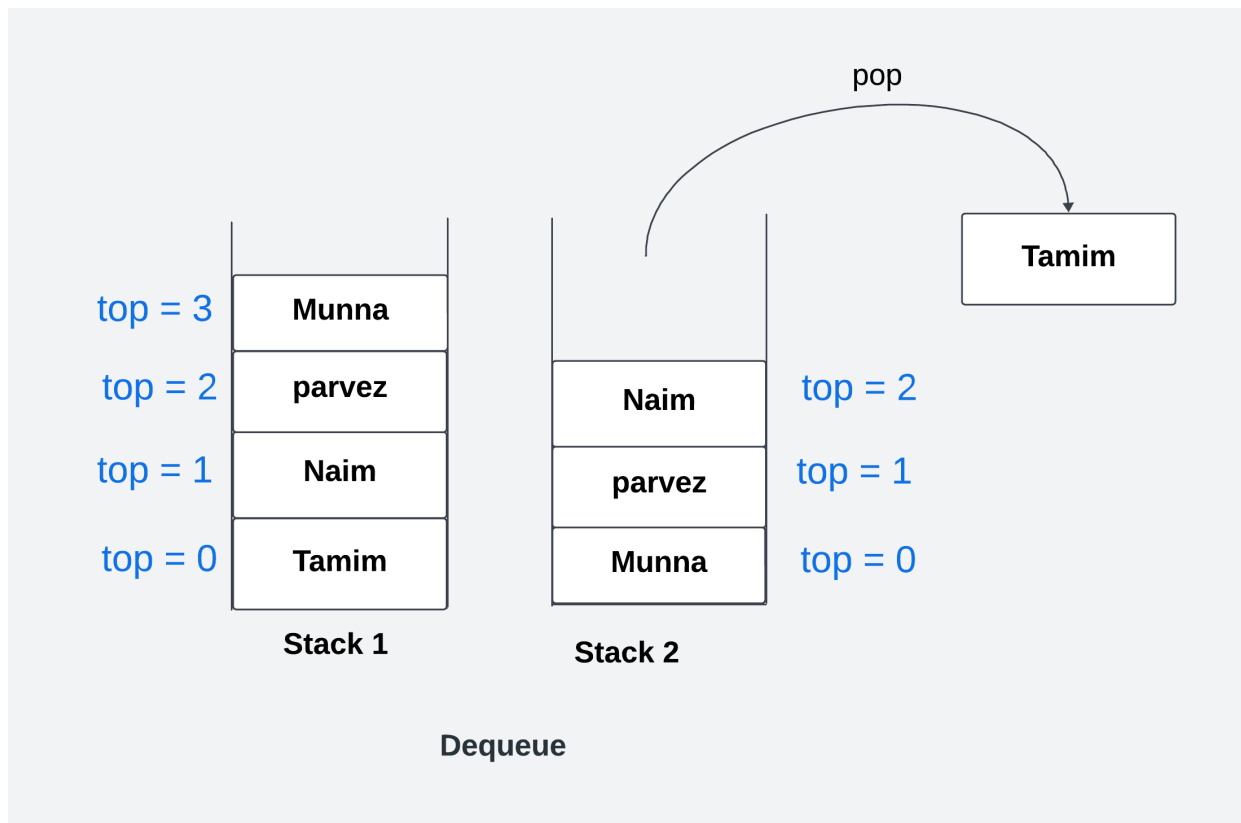


Deque Operation :

- We check **stack2**. If it's empty, we pop each element from **stack1** and push it into **stack2**.
- Now, **stack2** has the elements in reverse order: **Tamim** is now on top, followed by **Parvez**, **Naim**, and **Munna** on the bottom.



- Pop (remove) the top element from **stack2**. This is the element that has been in the queue the longest, maintaining **FIFO** order.



Q2. From Q1, if we find out the payment of Naim, what type of search is preferable? Justify your answer.

Ans:

To find **Naim's payment**, we could use either **Linear Search** or **Binary Search**. However, **Binary Search** is more suitable here because the data to be sorted, which is efficient and **time complexity** less than linear search.

Explanation:

Tamim(110) \Rightarrow Naim(120) \Rightarrow Parvez(135) \Rightarrow Munna(140)

Binary Search is preferable because it repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the naim(120) is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise, narrow it to the upper half. Repeatedly check until the value is found or the interval is empty. This method offers a time complexity of $O(\log n)$, which is more efficient than **Linear Search's** $O(n)$ for larger lists.

Since our payment list is already sorted, **Binary Search** is both feasible and efficient, allowing us to find **Naim's** payment quickly.

Task – 2

Q3. How did Mr. Himel solves the QA, QB, QC and QD?
Elaborately described with necessary explanation.

Ans:

QA: Write down the Pseudocode for traveling the entire linked list based on the scenario ['XX' = Your Name].

Ans:



```
Function(head){  
  Step 1: Set temp = head  
  Step 2: While (temp != NULL){  
            Print(temp->name)  
            temp = temp->next  
          }  
  Step 3: End
```

QB: Write the pseudocode to insert the name '**Abdullah**' into the linked list after *Your Name*, based on the given scenario.

Ans:



```
Linked_List_Insert (Head, Position_Name)
```

```
Step 1: Input New_name
```

```
Step 2: Set temp = Head
```

```
Step 3: Create New_Node with New_Node->Name = New_Name
```

```
Step 4: While (temp != NULL){
```

```
    If (temp->Name == Position_Name)
    {
```

```
        Set New_Node->Next = temp->Next
```

```
        Set temp->Next = New_Node
```

```
        Return
```

```
    }
```

```
    Set temp = temp->Next
```

```
}
```

```
Step 5: End
```

QC: Write the pseudocode to delete *Your Name* from the linked list based on the given scenario.

Ans:



```
delete (head, position_name)
```

```
Step 1: Set temp = Head  
       Set ptr = NULL
```

```
Step 2: While (PTR->Next != NULL){  
  
        If(PTR->Name == Position_Name)  
        {  
            ptr->Nex = temp->Next  
            Free temp  
            Return  
        }  
        ptr = temp  
        temp = temp->next  
    }
```

```
Step 3: End
```

QD: Is it possible to display the names (see **Figure 01**) in reverse order? If **yes**, please explain how this can be achieved with a proper explanation. If **no**, please justify your reasoning.

Ans:

Yes, it is possible to display names in reverse order. Here's how:

Recursion

- **Concept:** Use a recursive function to traverse the linked list to the end and print names in reverse order during the return phase.

- **Process:**

1. **Start at Head:** Call the recursive function with the head node.
2. **Base Case:** If the current node is NULL, return.
3. **Recursive Call:** Call the function with the next node.
4. **Print on Return:** Print the current node's name after the recursive call returns.

- **Output for “Mehedi -> Israfil -> Abdullah -> Faruk”:**

- **Result:** Faruk -> Abdullah -> Israfil -> Mehedi

Q4. How does **Mr. Mihal** solve the challenge provided by the course instructor? A detailed description is needed.

Ans:

Step 1: Select 15 Distinct Integer Numbers

Constraints:

1. The first number must be **50**.
2. The fifth number must be **91**.
3. The seventh number is **08**.
4. Exactly **three numbers** must be **less than 40**.
5. **No more than four numbers** can be **greater than 100**.

Example Selection:

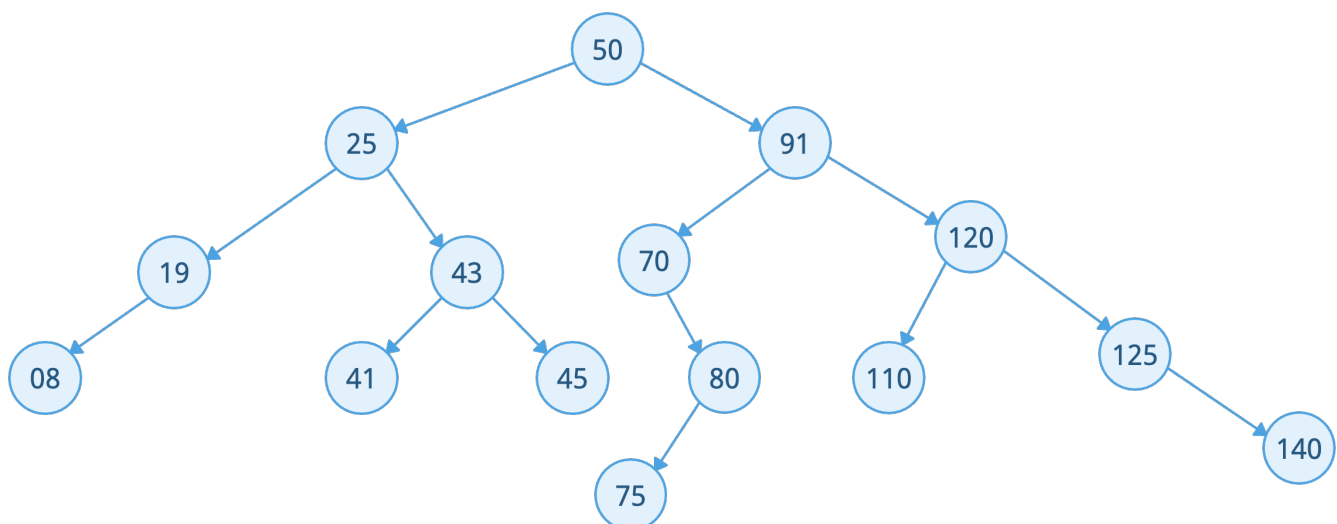
• **Selected Numbers:** 50, 25, 19, 43, 91, 43, 08, 120, 125, 140, 110, 70, 80, 75, 45

Step 2: Construct the Binary Search Tree (BST)

Insertion Process:

Root: Since the first number is the root. So the root is 50.

- Insert each number into the BST according to the binary search rules:
- If the search value is smaller than its parent, move to the left subtree.
- Otherwise, move to the right subtree.



Step 3: Finding the Student ID Using Binary Search

Objective: Use binary search to locate a specific value 08 in the BST.

Binary Search Steps:

1. **Start at Root (50):**

- Compare the search value (08) with the root node (50). Since 08 is less than 50, move to the **left child**.

2. **At Node (25):**

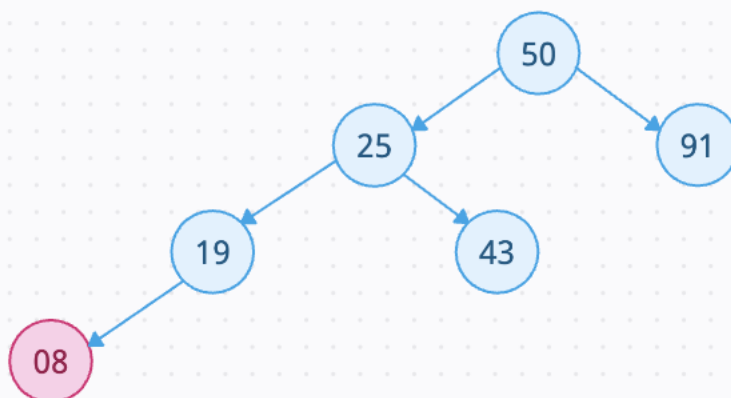
- Compare the search value (08) with the current node value (25). Since 08 is less than 25, move to the **left child**.

3. **At Node (19):**

- Compare the search value (08) with the current node value (19). Since 08 is less than 19, move to the **left child**.

4. **At Node (08):**

- Compare the search value (08) with the current node value (08). • The values match! The search is successful.



Q5. How did **Mr. Mihal** solve the difficulties faced by **Mr. Rafsan**? Describe with proper explanation.

Ans:

Mr. Rafsan's Difficulty:

- Incorrect Comparison
- Early Termination

Mr. Mihal could have helped Mr. Rafsan by:

1. Incorrect Comparison

- **Problem:** Rafsan made wrong decisions about moving left or right in the BST.
- **Solution:** Mihal explained the binary search logic:
- Move **left** if $08 < \text{current node}$.
- Move **right** if $08 > \text{current node}$.
- Stop when $08 == \text{current node}$.

2. Early Termination

- **Problem:** Rafsan stopped the search too soon, either at the wrong node or without exploring fully.
- **Solution:** Mihal clarified:
- Only stop when you find the value (08) or reach a NULL node (value not found).
- Follow the tree completely until one of these conditions is met.

Task – 3

Q6. Explain details about your theory work (Task-1 and Task-2) and make a short video (at least 4 minutes).

Ans:

https://drive.google.com/drive/folders/1jQSORKWveTGCLx3JNYBws_XJtqoPgYQA?usp=sharing

Lab Part

Task – 1

QL1. Implement the mentioned BST using the c/c++ programming language that is made by **Mr. Mihal**.

Ans:

```
#include <bits/stdc++.h>
```



```
using namespace std;

class node
{
public:
    int value;
    node *left;
    node *right;

    node(int val)
    {
        this->value = val;
        this->left = NULL;
        this->right = NULL;
    }
};

int main()
{
    node *root = NULL;
    int x;

    // insert
    cout << "Tree value(-1 to exit): ";
    while (true)
    {
        cin >> x;

        if (x == -1)
        {
            break;
        }

        if (root == NULL)
        {
            root = new node(x);
        }
        else
        {
            node *temp = root;
            while (true)
            {
                if (x < temp->value)
                {
                    if (temp->left == NULL)
```

```

        {
            temp->left = new node(x);
            break;
        }
        else
        {
            temp = temp->left;
        }
    }
    else
    {
        if (temp->right == NULL)
        {
            temp->right = new node(x);
            break;
        }
        else
        {
            temp = temp->right;
        }
    }
}

// traversal
queue<node *> q;
q.push(root);

while (!q.empty())
{
    node *curr = q.front();
    q.pop();

    // Print node
    if (curr->left != NULL)
    {
        cout << curr->left->value << "\t\t";
    }
    else
    {
        cout << "NULL\t\t";
    }

    cout << "Node value: " << curr->value << "\t\t";

    if (curr->right != NULL)

```

```

    {
        cout << curr->right->value << endl;
    }
    else
    {
        cout << "NULL" << endl;
    }

    // Add children
    if (curr->left != NULL)
    {
        q.push(curr->left);
    }
    if (curr->right != NULL)
    {
        q.push(curr->right);
    }
}

// Search
int key;
cout << "\nSearch value: ";
cin >> key;

bool found = false;
node *current = root;

while (current != NULL)
{
    if (current->value == key)
    {
        found = true;
        break;
    }
    else if (current->value > key)
    {
        current = current->left;
    }
    else
    {
        current = current->right;
    }
}

if (found)
{
    cout << "Found\n";
}

```

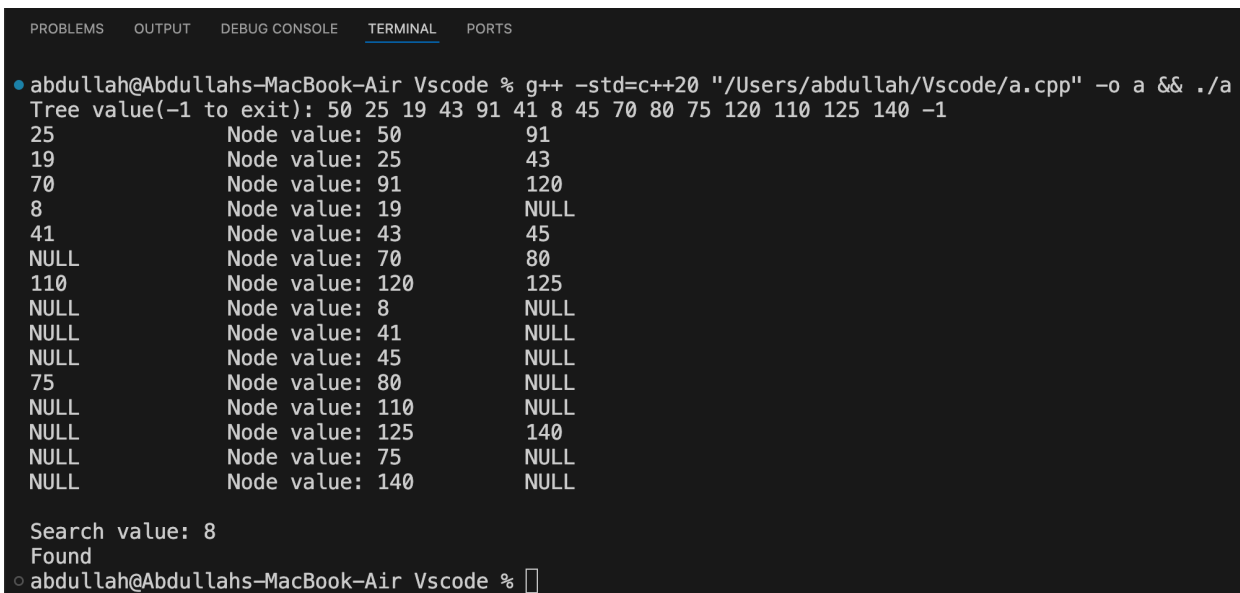
```

    }
    else
    {
        cout << "Not Found!\n";
    }

    return 0;
}

```

Output:



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
•abdullah@Abdullahs-MacBook-Air Vscode % g++ -std=c++20 "/Users/abdullah/Vscode/a.cpp" -o a && ./a
Tree value(-1 to exit): 50 25 19 43 91 41 8 45 70 80 75 120 110 125 140 -1
25      Node value: 50      91
19      Node value: 25      43
70      Node value: 91      120
8       Node value: 19      NULL
41      Node value: 43      45
NULL    Node value: 70      80
110     Node value: 120     125
NULL    Node value: 8       NULL
NULL    Node value: 41      NULL
NULL    Node value: 45      NULL
75      Node value: 80      NULL
NULL    Node value: 110     NULL
NULL    Node value: 125     140
NULL    Node value: 75      NULL
NULL    Node value: 140     NULL

Search value: 8
Found
○abdullah@Abdullahs-MacBook-Air Vscode %

```

QL2. Implement the *QB*, *QC* and *QD* using the c/c++ programming language that is made by **Mr. Himel**.

Ans:

```

#include <bits/stdc++.h>

using namespace std;

class node
{
public:

```

```

string name;
node *next;

node(string name)
{
    this->name = name;
    this->next = NULL;
}

};

int main()
{
    node *head = NULL, *newnode, *temp;
    string postion_name;

    // Input linked list creation
    while (1)
    {
        string name;
        cout << "Enter data (type 'exit' to end): ";
        cin >> name;

        if (name == "exit")
        {
            break;
        }

        newnode = new node(name);

        if (head == NULL)
        {
            head = newnode;
            temp = newnode;
        }
        else
        {
            temp->next = newnode;
            temp = newnode;
        }
    }

    while (true)
    {
        int choice;
        cout << "\nPerform operations on the Linked list:";
        cout << "\n1.Traversal\n2.Insert Name\n3.Delete Name\n4.Reverse
name\n5.Exit";
        cout << "\n\nEnter the choice: ";
    }
}

```

```

cin >> choice;

node *temp_insert, *new_node, *temp_delete, *ptr_delete, *prev,
*curr, *next;
string new_name;
bool found = false, deleted = false;

switch (choice)
{
case 1:
    // Traverse the linked list
    temp = head;
    while (temp != NULL)
    {
        cout << temp->name << " ";
        temp = temp->next;
    }
    cout << endl;
    break;

case 2:
    // Insert

    cout << "Which name want to insert after: ";
    cin >> postion_name;
    cout << "Enter the new name: ";
    cin >> new_name;

    temp_insert = head;
    new_node = new node(new_name);

    while (temp_insert != NULL)
    {
        if (temp_insert->name == postion_name)
        {
            new_node->next = temp_insert->next;
            temp_insert->next = new_node;
            break;
        }
        temp_insert = temp_insert->next;
    }
    cout << "Insert Successfully";

    break;

case 3:
    // Delete

```

```

    cout << "Select the name which you want to delete: ";
    cin >> postion_name;

    temp_delete = head;
    ptr_delete = NULL;

    while (temp_delete != NULL)
    {
        if (temp_delete->name == postion_name)
        {
            ptr_delete->next = temp_delete->next;
            delete temp_delete;
            break;
        }
        ptr_delete = temp_delete;
        temp_delete = temp_delete->next;
    }
    cout << "Delete Successfully";
    break;

case 4:
    // Reverse

    prev = NULL;
    curr = head;

    while (curr != NULL)
    {
        next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
    head = prev;

    cout << "Successfully reverse";
    break;

case 5:
    return 0;

default:
    cout << "Invalid choice!" << endl;
    break;
}
}
}

```

Output:

- At first, we have to Insert value, when we write “exit” then stop insert value.

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
• abdullah@Abdullahs-MacBook-Air Vscod % g++ -std=c++20 "/Users/abdullah/Vscod
Enter data (type 'exit' to end): Mehedi
Enter data (type 'exit' to end): Israfil
Enter data (type 'exit' to end): Abdullah
Enter data (type 'exit' to end): Faruk
Enter data (type 'exit' to end): exit

Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 1
Mehedi Israfil Abdullah Faruk

```

- When we stop adding the value, we again insert my name after "Abdullah".

```

Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 2
Which name want to insert after: Abdullah
Enter the new name: abdullah
Insert Successfully
Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 1
Mehedi Israfil Abdullah abdullah Faruk

```

- Delete my name from the linked list.


```

Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 3
Select the name which you want to delete: abdullah
Delete Successfully
Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 1
Mehedi Israfil Abdullah Faruk

```

➤ Display the names in reverse order.

```

Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 4
Successfully reverse
Perform operations on the Linked list:
1.Traversal
2.Insert Name
3.Delete Name
4.Reverse name
5.Exit

Enter the choice: 1
Faruk Abdullah Israfil Mehedi

```

QL3. Explain details about your lab work (Task-1) and make a short video (at least 4 minutes).

Ans:

https://drive.google.com/drive/folders/1HL3kFNgfUYP9kVDYsc3QzI5XvslceUN8?usp=drive_link