3.	**Write a program to implement singly linked list which supports the following operations:** **(i) Insert an element x at the beginning of the singly linked list** **(ii) Insert an element x at position in the singly linked list** **(iii)Remove an element from the beginning of the singly linked list** **(iv) Remove an element from position in the singly linked list.** **(v) Search for an element x in the singly linked list and return its pointer** **(vi) Concatenate two singly linked lists**	
4.	**Write a program to implement doubly linked list which supports the following operations:** **(i) Insert an element x at the beginning of the doubly linked list** **(ii) Insert an element x at position in the doubly linked list** **(iii)Insert an element x at the end of the doubly linked list** **(iv) Remove an element from the beginning of the doubly linked list** **(v) Remove an element from position in the doubly linked list.** **(vi) Remove an element from the end of the doubly linked list**	

	(vii) Search for an element x in the doubly linked list and return its pointer (viii) Concatenate two doubly linked lists	
5.	Write a program to implement circularly linked list which supports the following operations: (i) Insert an element x at the front of the circularly linked list (ii) Insert an element x after an element y in the circularly linked list (iii)Insert an element x at the back of the circularly linked list (iv) Remove an element from the back of the circularly linked list (v) Remove an element from the front of the circularly linked list (vi) remove the element x from the circularly linked list (vii)Search for an element x in the circularly linked list and return its pointer (viii) Concatenate two circularly linked lists	
6.	Implement a stack using Array representation.	
7.	Implement a stack using Linked representation	

8.	Implement Queue using Circular Array representation	
9.	Implement Queue using Circular linked list representation	
10.	Implement Double-ended Queues using Linked list representation	
11	Write a program to implement Binary Search Tree which supports the following operations: (i) Insert an element x (ii) Delete an element x (iii) Search for an element x in the BST and change its value to y and then place the node with value y at its appropriate position in the BST (iv) Display the elements of the BST in preorder, inorder, and postorder traversal (v) Display the elements of the BST in level-by-level traversal (vi) Display the height of the BST	

PRACTICAL-1

// 1. write a program to search an element from a list , Give user the option to perform Linear or Binary search .Use template function

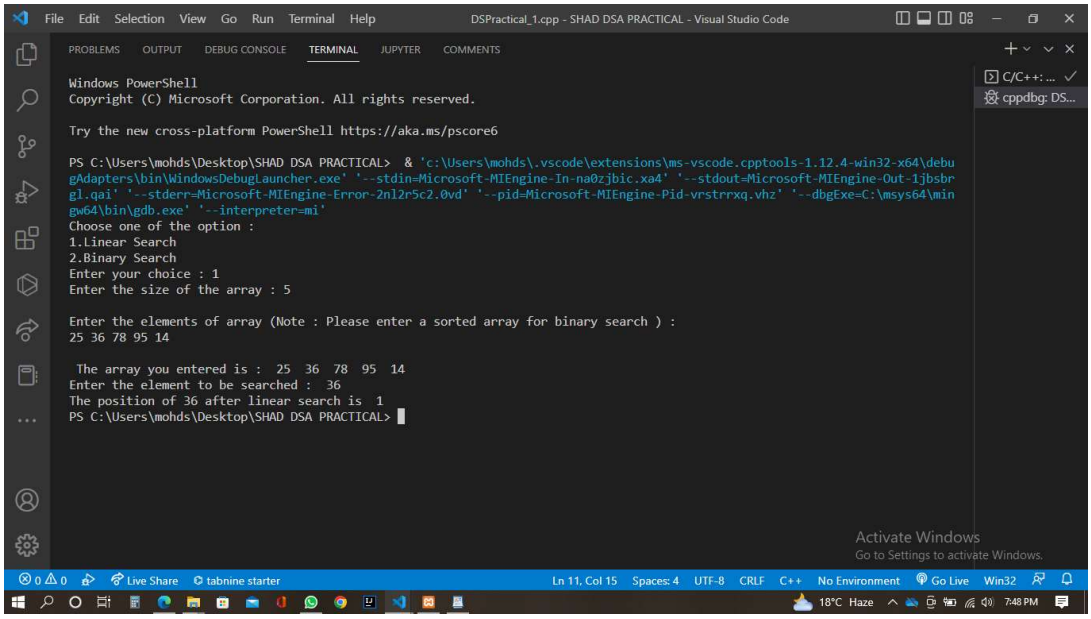
```
#include<iostream>

using namespace std;

template <class t1>

int LinearSearch(t1 arr[],int n,t1 key){

   for(int i=0;i<n;i++){

      if(arr[i]==key){

         return i;

      }

   }

   return -1;

}

template <class t2>

int BinarySearch(t2 arr[],int n,t2 key){

int s=0;
```

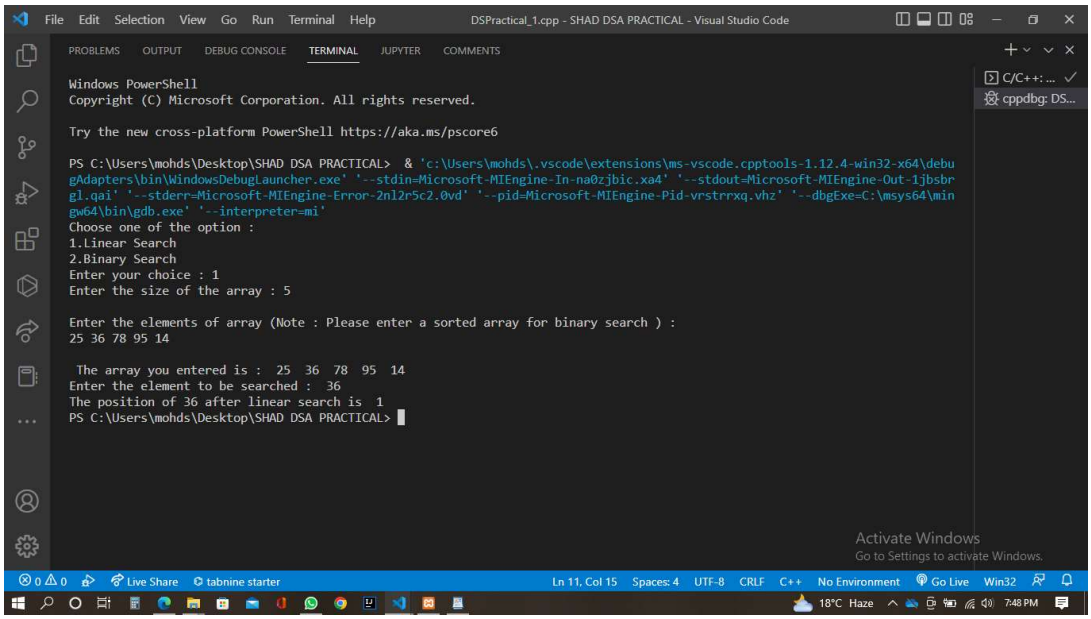
```
int e=n;
while(s<=e){
    int mid=(s+e)/2;
    if(arr[mid]==key){
        return mid;
    }
    else if(arr[mid]>key){
        e=mid-1;
    }
    else if(arr[mid]<key){
        s=mid+1;
    }
}
 return -1;
}
int menu(){
    cout<<"Choose one of the option : \n";
    cout<<"1.Linear Search\n";
    cout<<"2.Binary Search\n";
    cout<<"Enter your choice : ";
    int ch;
    cin>>ch;
    return ch;
}
```

```
int main(){
    int ch=menu();
    cout<<"Enter the size of the array : ";
    int n;
    cin>>n;
    int array[n];
    cout<<"\nEnter the elements of array (Note : Please enter a sorted array for binary search ) :
\n";
    for(int i=0;i<n;i++){
        cin>>array[i];
    }
    cout<<"\n The array you entered is :  ";
    for(int i=0;i<n;i++){
        cout<<array[i]<<"  ";
    }
    cout<<"\nEnter the element to be searched :  ";
    int key;
    cin>>key;

    if(ch==2){
    cout<<"The position of "<<key<<" after binary search is "<<BinarySearch(array,n,key)<<endl;
    }
    if(ch==1){
```

```
    cout<<"The position of "<<key<<" after linear search is  "<<LinearSearch(array,n,key)<<endl;
    }
    return 0;
}
```

OUTPUT





PRACTICAL-2

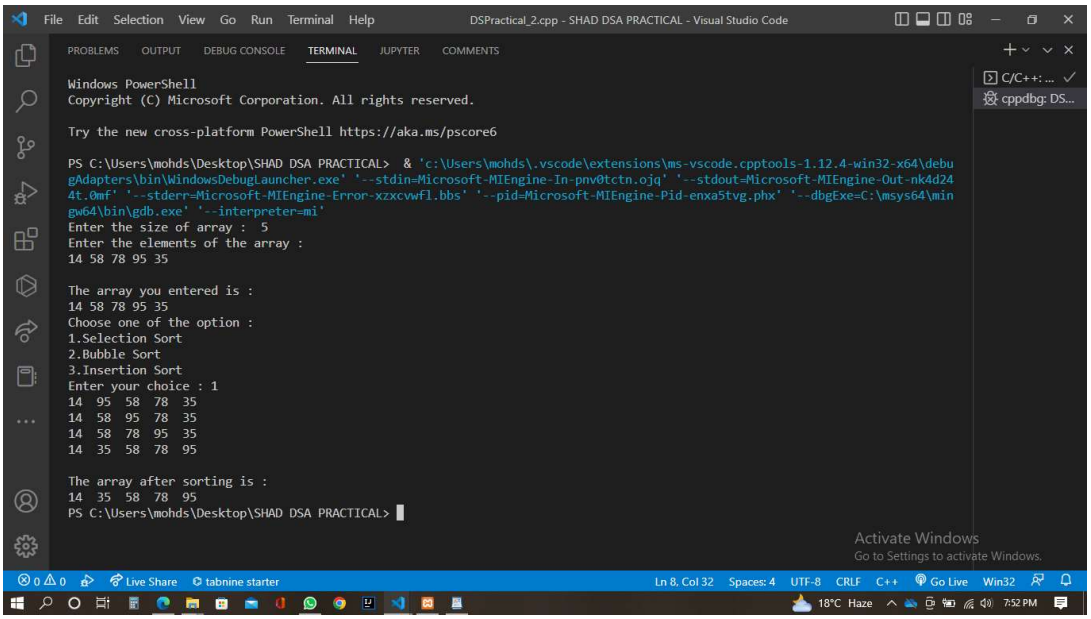
```
// WAP using templates to sort a list of elements . Give user the option to perform the sorting
//using Insertion Sort,Bubble Sort, or Selection Sort.
#include<iostream>
using namespace std;
template <class t1>
 void SelectionSort(t1 arr[],int n){
      for(int i=1;i<n;i++){
       for(int j=0;j<n-1;j++){
          if(arr[i]<arr[j]){
             int temp=arr[j];
             arr[j]=arr[i];
             arr[i]=temp;
          }
       }
       for(int i=0;i<n;i++){
          cout<<arr[i]<<"  ";
       }
       cout<<"\n";
   }
}
template <class t2>
void BubbleSort(t2 arr[],int n){
```

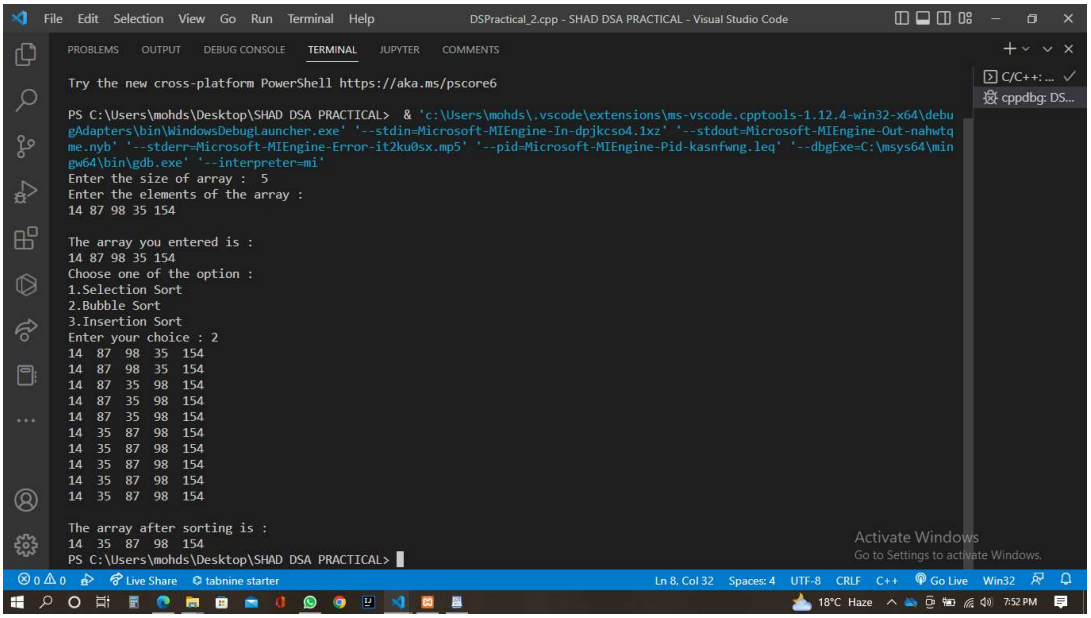
```
        int counter=1;
    while(counter<n){
    for(int i=0;i<n-counter;i++){
        if(arr[i]>arr[i+1]){
            int temp=arr[i];
            arr[i]=arr[i+1];
            arr[i+1]=temp;
        }

        for(int j=0;j<n;j++){
          cout<<arr[j]<<"  ";
        }
        cout<<"\n";
       }
         counter++;
}
}
template <class t3>
void InsertionSort(t3 arr[],int n){
for(int i=1;i<n;i++){
int current=arr[i];
int j=i-1;
while(arr[j]>current && j>=0){
    arr[j+1]=arr[j];
    j--;
}
```

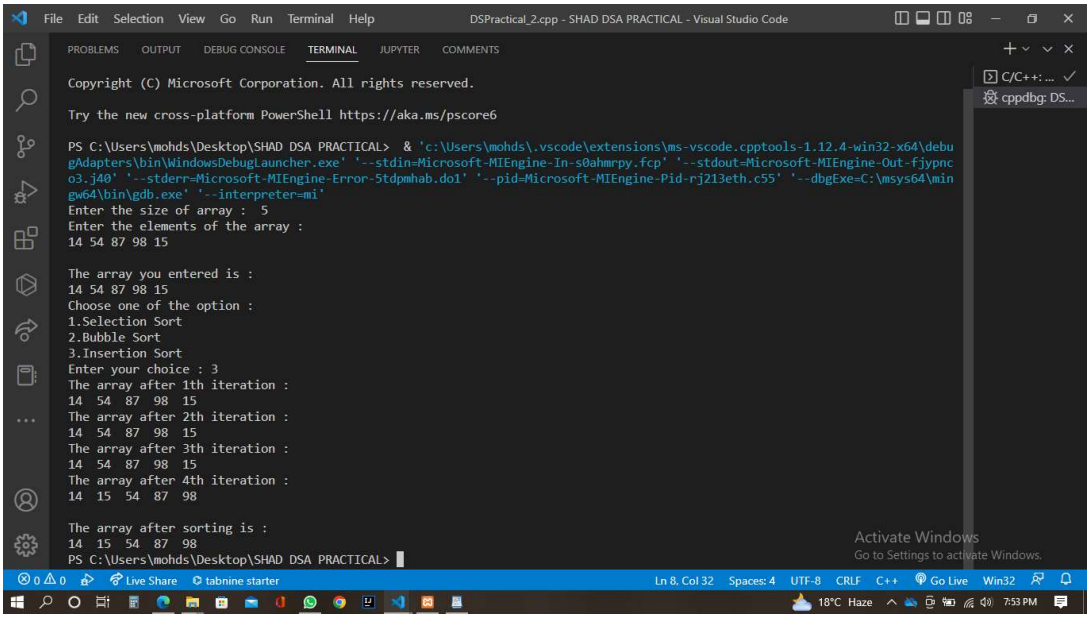
```
arr[j+1]=current;
cout<<"The array after "<<i<<"th"<<" iteration : \n";
for(int k=0;k<n;k++){
cout<<arr[k]<<"  ";
}
cout<<"\n";
  }
}
int menu(){
    cout<<"Choose one of the option : \n";
    cout<<"1.Selection Sort\n";
    cout<<"2.Bubble Sort\n";
    cout<<"3.Insertion Sort\n";
    cout<<"Enter your choice : ";
    int ch;
    cin>>ch;
    return ch;
}
int main(){
    cout<<"Enter the size of array :  ";
    int n;
    cin>>n;
    int array[n];
    cout<<"Enter the elements of the array : \n";
    for(int i=0;i<n;i++){
        cin>>array[i];
```

```
    }
    cout<<"\nThe array you entered is : \n";
    for(int i=0;i<n;i++){
        cout<<array[i]<<" ";
    }
    cout<<"\n";
    int ch=menu();
    if(ch==1){
        SelectionSort(array,n);
    }
    if (ch==2){
        BubbleSort(array,n);
    }if(ch==3){
        InsertionSort(array,n);
    }
     cout<<"\nThe array after sorting is : \n";
    for(int i=0;i<n;i++){
        cout<<array[i]<<"  ";
    }
    return 0;
}
```

OUTPUT


File Edit Selection View Go Run Terminal Help
DSPractical_2.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\mohds\Desktop\SHAD DSA PRACTICAL> & 'c:\Users\mohds\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-pnv0tctn.ojq' '--stdout=Microsoft-MIEngine-Out-nk4d244t.0mf' '--stderr=Microsoft-MIEngine-Error-xzxcvwfl.bbs' '--pid=Microsoft-MIEngine-Pid-enxa5tvg.phx' '--dbgExe=C:\msys64\mingw64\bin\gdb.exe' '--interpreter=mi'
Enter the size of array : 5
Enter the elements of the array :
14 58 78 95 35
The array you entered is :
14 58 78 95 35
Choose one of the option :
1.Selection Sort
2.Bubble Sort
3.Insertion Sort
Enter your choice : 1
14 95 58 78 35
14 58 95 78 35
14 58 78 95 35
14 35 58 78 95
The array after sorting is :
14 35 58 78 95
PS C:\Users\mohds\Desktop\SHAD DSA PRACTICAL>
C/C++: ...
cppdbg: DS...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 8, Col 32 Spaces: 4 UTF-8 CRLF C++ Go Live Win32
18°C Haze
7:52 PM


File Edit Selection View Go Run Terminal Help
DSPractical_2.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\mohds\Desktop\SHAD DSA PRACTICAL> & 'c:\Users\mohds\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-dpjkcso4.1xz' '--stdout=Microsoft-MIEngine-Out-nahwtqme.nyb' '--stderr=Microsoft-MIEngine-Error-it2ku0sx.mp5' '--pid=Microsoft-MIEngine-Pid-kasnfwng.leq' '--dbgExe=C:\msys64\mingw64\bin\gdb.exe' '--interpreter=mi'
Enter the size of array : 5
Enter the elements of the array :
14 87 98 35 154
The array you entered is :
14 87 98 35 154
Choose one of the option :
1.Selection Sort
2.Bubble Sort
3.Insertion Sort
Enter your choice : 2
14 87 98 35 154
14 87 98 35 154
14 87 35 98 154
14 87 35 98 154
14 87 35 98 154
14 35 87 98 154
14 35 87 98 154
14 35 87 98 154
14 35 87 98 154
14 35 87 98 154
The array after sorting is :
14 35 87 98 154
PS C:\Users\mohds\Desktop\SHAD DSA PRACTICAL>
C/C++: ...
cppdbg: DS...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 8, Col 32 Spaces: 4 UTF-8 CRLF C++ Go Live Win32
18°C Haze
7:52 PM



PRACTICAL-3

```
#include <iostream>
using namespace std;

// A linked list node
template <class N>
class Node
{
public:
    N data;
```

```
    Node<N> *next;
};

template <class N>
void insertAtHead(Node<N> **head_ref, N new_data)
{

    // 1. allocate node
    Node<N> *new_node = new Node<N>();

    // 2. put in the data
    new_node->data = new_data;

    // 3. Make next of new node as head
    new_node->next = (*head_ref);

    // 4. move the head to point
    // to the new node
    (*head_ref) = new_node;
}

template <class N>
void insertAfter(Node<N> *prev_node, N new_data)
{
    if (prev_node == NULL)
    {
```

```
        cout << "The given previous node cannot be NULL";
        return;
    }

    Node<N> *new_node = new Node<N>();

    // 3. put in the data
    new_node->data = new_data;

    // 4. Make next of new node
    // as next of prev_node
    new_node->next = prev_node->next;

    // 5. move the next of prev_node
    // as new_node
    prev_node->next = new_node;
}

template <class N>
void append(Node<N> **head_ref, N new_data)
{

    // 1. allocate node
    Node<N> *new_node = new Node<N>();

    // used in step 5
```

```
Node<N> *last = *head_ref;

// 2. put in the data
new_node->data = new_data;

// 3. This new node is going to be
// the last node, so make next of
// it as NULL
new_node->next = NULL;

// 4. If the Linked List is empty,
// then make the new node as head
if (*head_ref == NULL)
{
   *head_ref = new_node;
   return;
}

// 5. Else traverse till the last node
while (last->next != NULL)
{
   last = last->next;
}

// 6. Change the next of last node
last->next = new_node;
```

```
    return;
}

template <class N>
void deleteNode(Node<N> **head_ref, N key)
{

    // Store head node
    Node<N> *temp = *head_ref;
    Node<N> *prev = NULL;

    // If head node itself holds
    // the key to be deleted
    if (temp != NULL && temp->data == key)
    {
        *head_ref = temp->next; // Changed head
        delete temp;           // free old head
        return;
    }

    // Else Search for the key to be deleted,
    // keep track of the previous node as we
    // need to change 'prev->next'
    else
    {
        while (temp != NULL && temp->data != key)
```

```
        {
            prev = temp;
            temp = temp->next;
        }

        // If key was not present in linked list
        if (temp == NULL)
            return;

        // Unlink the node from linked list
        prev->next = temp->next;

        // Free memory
        delete temp;
    }
}
template <class N>
bool search(Node<N> *head, N x)
{
    Node<N> *current = head; // Initialize current
    while (current != NULL)
    {
        if (current->data == x)
            return true;
        current = current->next;
    }
```

```
    return false;
}
template <class N>
void reverse(Node<N> **head)
{
    // Initialize current, previous and next pointers
    Node<N> *current = *head;
    Node<N> *prev = NULL, *next = NULL;

    while (current != NULL)
    {
        // Store next
        next = current->next;
        // Reverse current node's pointer
        current->next = prev;
        // Move pointers one position ahead.
        prev = current;
        current = next;
    }
    *head = prev;
}
template <class N>
void concat(Node<N> *first, Node<N> **second)
{
    Node<N> *firstRef = first;
```

```
   // finding the lat node of first linked list
   while (firstRef->next != NULL)
   {
      firstRef = firstRef->next;
   }

   firstRef->next = *second;
}

// This function prints contents of
// linked list starting from head
template <class N>
void printList(Node<N> *node)
{
   while (node != NULL)
   {
      cout << " " << node->data;
      node = node->next;
   }
}

int main()
{
   cout<<"\t\t\t\t----------------NAME-RITESH------ROLL NO.-CSC/21/15-------
\n";
   Node<int> *first = NULL;
```

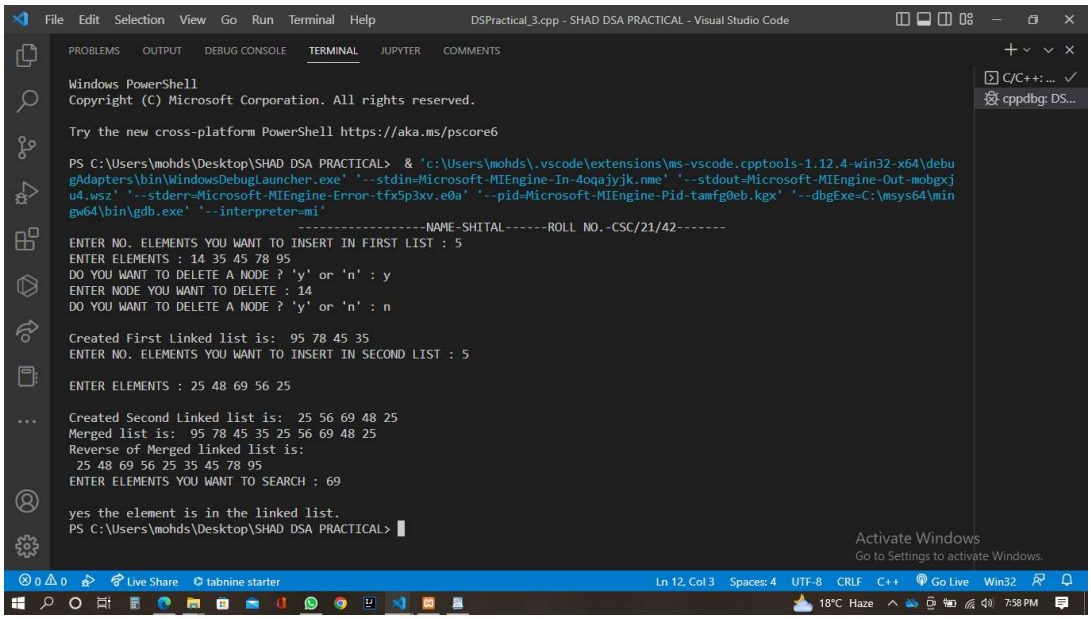
```
// Insert 6. So linked list becomes 6->NULL
int n,k;
cout<<"ENTER NO. ELEMENTS YOU WANT TO INSERT IN FIRST LIST : ";
cin>>n;
cout<<"ENTER ELEMENTS : ";
cin>>k;

append(&first,k);
for(int i=1;i<n;i++){
   cin>>k;
   insertAtHead(&first, k);

}
// append(&first, 6);

// // Insert 7 at the beginning.
// // So linked list becomes 7->6->NULL
// insertAtHead(&first, 7);

// // Insert 1 at the beginning.
// // So linked list becomes 1->7->6->NULL
// insertAtHead(&first, 1);

// // Insert 4 at the end. So
// // linked list becomes 1->7->6->4->NULL
```

```
// append(&first, 4);

// // Insert 8, after 7. So linked
// // list becomes 1->7->8->6->4->NULL
// insertAfter(first->next, 8);

// delete node which contains data 6
char c1;int d;
cout<<"DO YOU WANT TO DELETE A NODE ? 'y' or 'n' : ";
cin>>c1;
while(c1 == 'y'){
   cout<<"ENTER NODE YOU WANT TO DELETE : ";
   cin>>d;
   deleteNode(&first, d);
   cout<<"DO YOU WANT TO DELETE A NODE ? 'y' or 'n' : ";
   cin>>c1;
}

cout << "\nCreated First Linked list is: ";
printList(first);

Node<int> *second = NULL;
int n2,k2;
cout<<"\nENTER NO. ELEMENTS YOU WANT TO INSERT IN SECOND LIST : ";
cin>>n2;
cout<<"\nENTER ELEMENTS : ";
```

```
cin>>k2;

append(&second,k2);
for(int i=1;i<n2;i++){
   cin>>k2;
   insertAtHead(&second, k2);

}
// append(&second, 11);
// append(&second, 12);
// append(&second, 15);

cout << "\nCreated Second Linked list is: ";

printList(second);
cout << "\nMerged list is: ";
concat(first, &second);
printList(first);

reverse(&first);
cout << endl;
cout << "Reverse of Merged linked list is: " << endl;
printList(first);
int s2;
cout<<"\nENTER ELEMENTS YOU WANT TO SEARCH : ";
cin>>s2;
```

```
    bool x = search(first, s2);
    if (x = true)
    {
       cout << "\nyes the element is in the linked list.";
    }
    else
    {
       cout << "\nno the element is not in the linked list. ";
    }
    cout << endl;

    // Node<char> *c = NULL;
    // append(&c, 'p');
    // append(&c, 'z');
    // cout << "Linked list of characters : ";
    // printList(c);

    return 0;
}
```

OUTPUT



PRACTICAL-4

/*Write a program to implement Doubly Linked List using templates.

Include functions for insertion, deletion and search of a number, reverse the list.*/

```
#include <iostream>

using namespace std;

template <class t>

class node

{

public:

    t data;

    node *prev, *next;

};

template <class t>

class dlist

{
```

```
    int n;
    node<t> *first, *last;

public:
    dlist()
    {
        first = NULL;
        last = NULL;
    }

    // create function
    void create()
    {
        node<t> *current, *temp;
        char ch;
        // fflush(stdin);

        first = new node<t>;
        cout << "Enter data for first node:\n";
        cin >> first->data;
        current = first;
        first->next = NULL;
        first->prev = NULL;
        last = first;

        do
```

```
    {
        cout << "Want to enter more data:\n";
        cin >> ch;
        if (ch == 'y')
        {
            n = count();
            this->insert(n + 1);
        }
    } while (ch == 'y');
}

// display function
void display()
{
    node<t> *current;
    current = first;
    cout << "The data in linked list:\n";
    while (current != NULL)
    {
        cout << current->data << " <-> ";
        current = current->next;
    }
    cout << "\n";
}

// reverse function
```

```
void reverse()
{
    n = count();
    // fflush(stdin);
    node<t> *current;
    current = last;
    cout << "The data after reversing the linked list:\n";
    for (int i = 1; i <= n; i++)
    {
        cout << current->data << " -> ";
        current = current->prev;
    }
}

// count function
int count()
{
    int c = 0;
    node<t> *current;
    current = first;
    while (current != NULL)
    {
        c++;
        current = current->next;
    }
    return c;
```

```
}

// insert function
void insert(int n1)
{
    int b = count();
    if (n1 <= b + 1)
    {
        node<t> *current, *forward, *temp;
        current = first;
        temp = new node<t>;
        cout << "Enter data:\n";
        cin >> temp->data;
        temp->next = temp->prev = NULL;
        if (n1 == 1)
        {
            temp->next = first;
            first->prev = temp;
            first = temp;
        }
        else if (n1 <= b)
        {
            for (int i = 1; i < n - 1; i++)
                current = current->next;

            forward = current->next;
```

```
            temp->next = forward;
            current->next = temp;
            temp->prev = current;
            forward->prev = temp;
        }
        else
        {
            last->next = temp;
            temp->prev = last;
            last = temp;
        }
    }
    else
        cout << "Can't be inserted\n";
}

// search function
void search()
{
    int flag = 0;
    cout << "Enter data to be searched:\n";
    cin >> n;
    node<t> *current, *previ, *temp;
    int b = count();
    current = first;
    for (int i = 1; i <= b; i++)
```

```
        {
            if (current->data == n)
            {
                flag = 1;
                break;
            }
            current = current->next;
        }

        if (flag != 0)
        {
            previ = current->prev;
            int c;
            cout << "Data found:\nEnter what you wannna do:\n 1.delete data\n
2.replace it\n 3.do nothing\n";
            cin >> c;
            switch (c)
            {
            case 1:
                temp = current;
                if (current->next != NULL)
                {
                    current = current->next;
                    previ->next = current;
                    current->prev = previ;
                }
```

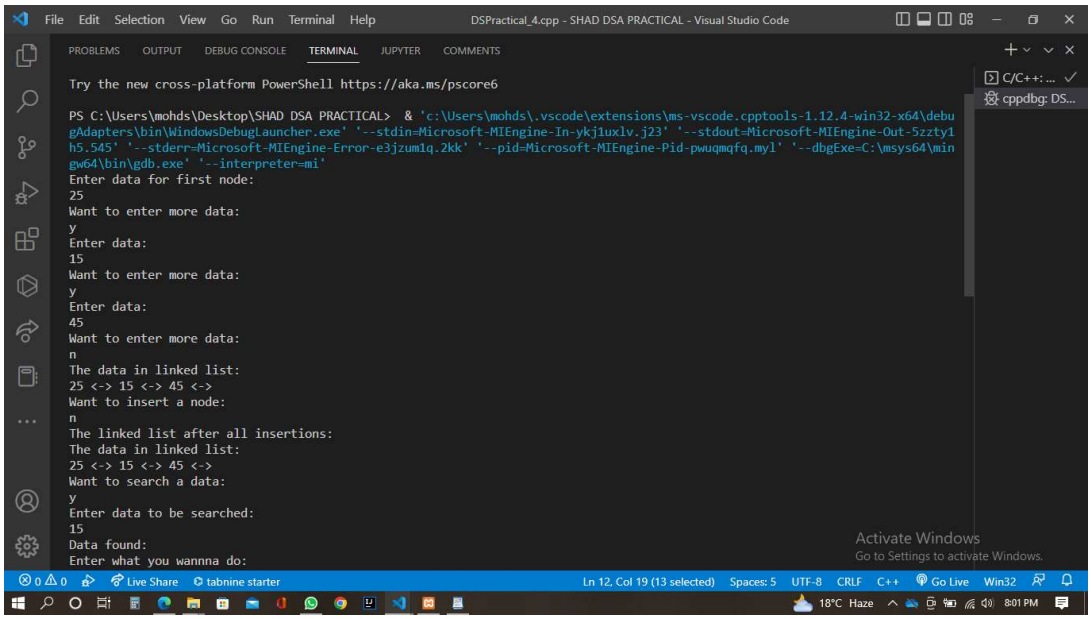
```
            else
            {
                previ->next = NULL;
                current->prev = NULL;
            }
            delete (temp);
            cout << "Data deleted:\n";
            this->display();
            break;
        case 2:
            cout << "Enter new data:\n";
            cin >> current->data;
            cout << "data replaced:\n";
            this->display();
            break;
        case 3:
            break;
        default:
            cout << "wrong choice:\n";
        }
        // getchar();
    }
    else
        cout << "Data not found:\n";
}
```

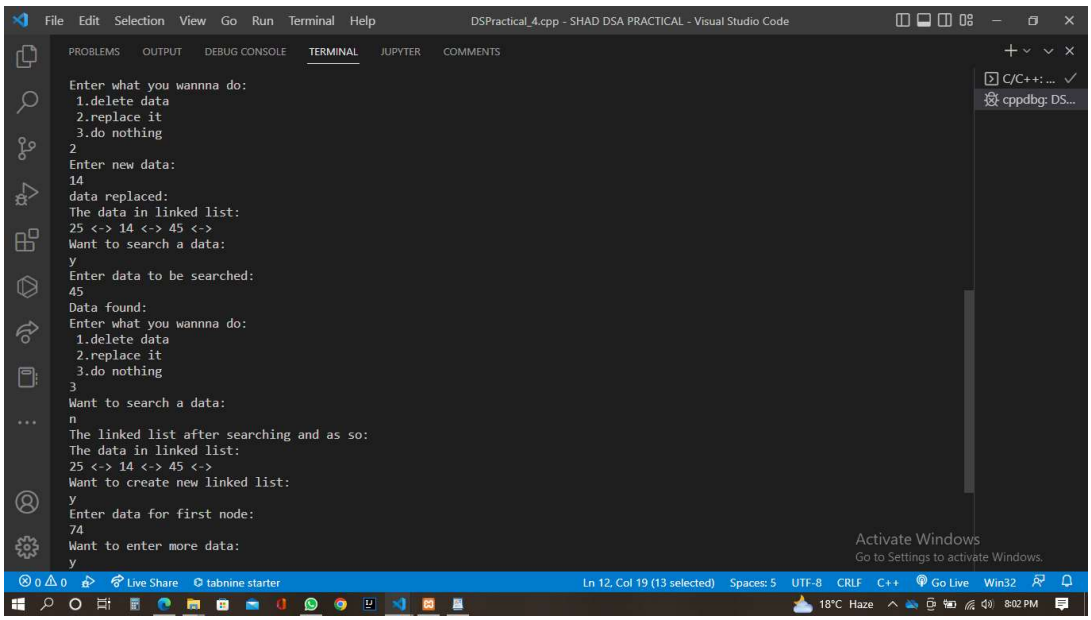
```
    // overloading + operator
    dlist operator+(dlist l)
    {
        dlist l6;
        l6.first = first;
        l6.last = last;
        l6.last->next = l.first;
        l.first->prev = l6.last;
        return l6;
    }
};

int main()
{
    int n;
    char ch;
    dlist<int> l1, l3, l2;
    l1.create();
    l1.display();

    // doing insertion
    do
    {
        cout << "Want to insert a node:\n";
        cin >> ch;
        if (ch == 'y')
```

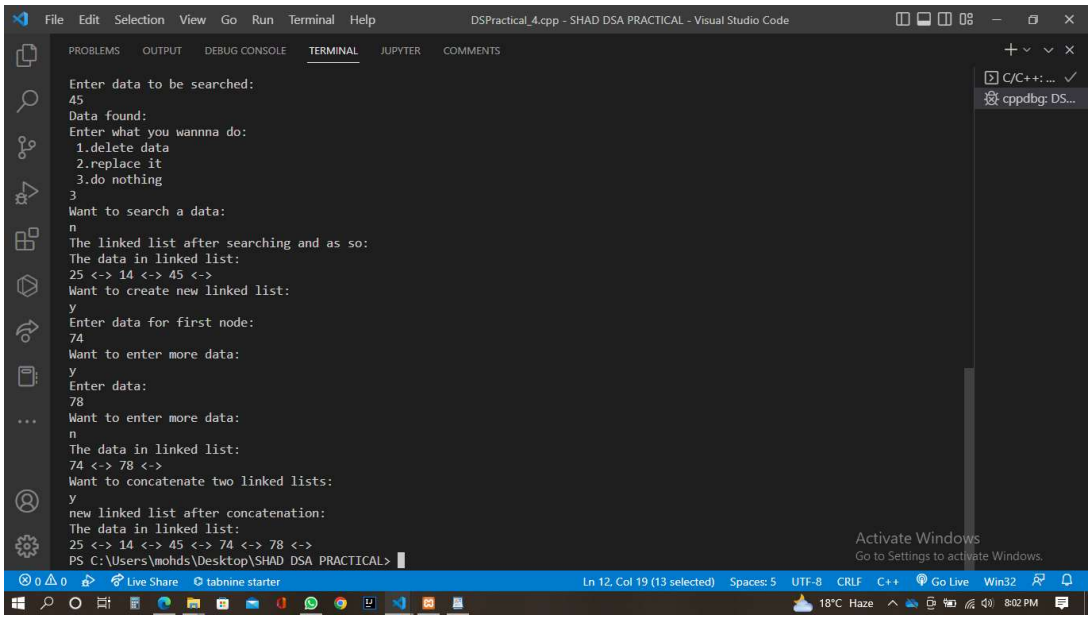
```
    {
        cout << "Enter the position of insertion;\n";
        cin >> n;
        l1.insert(n);
    }
} while (ch == 'y');

cout << "The linked list after all insertions:\n";
l1.display();

// doing searching,deleting,replacing
do
{
    cout << "Want to search a data:\n";
    cin >> ch;
    if (ch == 'y')
        l1.search();
} while (ch == 'y');

cout << "The linked list after searching and as so:\n";
l1.display();

// creating new linked list
cout << "Want to create new linked list:\n";
char cht;
cin >> cht;
```

```
    if (cht == 'y')
    {
        l2.create();
        l2.display();

        // concatenating strings
        cout << "Want to concatenate two linked lists:\n";
        cin >> ch;
        if (ch == 'y')
        {
            l3 = l1 + l2;
            cout << "new linked list after concatenation:\n";
            l3.display();
        }
    }

    return 0;
}
```

OUTPUT


File Edit Selection View Go Run Terminal Help
DSPractical_4.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\mohds\Desktop\SHAD DSA PRACTICAL> & 'c:\Users\mohds\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-ykj1uxlv.j23' '--stdout=Microsoft-MIEngine-Out-5zzty1h5.545' '--stderr=Microsoft-MIEngine-Error-e3jzum1q.2kk' '--pid=Microsoft-MIEngine-Pid-pwuqmqfq.myl' '--dbgExe=C:\msys64\mingw64\bin\gdb.exe' '--interpreter=mi'
Enter data for first node:
25
Want to enter more data:
y
Enter data:
15
Want to enter more data:
y
Enter data:
45
Want to enter more data:
n
The data in linked list:
25 <-> 15 <-> 45 <->
Want to insert a node:
n
The linked list after all insertions:
The data in linked list:
25 <-> 15 <-> 45 <->
Want to search a data:
y
Enter data to be searched:
15
Data found:
Enter what you wannna do:
C/C++: ...
cppdbg: DS...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 12, Col 19 (13 selected)
Spaces: 5
UTF-8
CRLF
C++
Go Live
Win32
18°C Haze
8:01 PM


File Edit Selection View Go Run Terminal Help
DSPractical_4.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
Enter what you wannna do:
1.delete data
2.replace it
3.do nothing
2
Enter new data:
14
data replaced:
The data in linked list:
25 <-> 14 <-> 45 <->
Want to search a data:
y
Enter data to be searched:
45
Data found:
Enter what you wannna do:
1.delete data
2.replace it
3.do nothing
3
Want to search a data:
n
The linked list after searching and as so:
The data in linked list:
25 <-> 14 <-> 45 <->
Want to create new linked list:
y
Enter data for first node:
74
Want to enter more data:
y
C/C++: ...
cppdbg: DS...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 12, Col 19 (13 selected)
Spaces: 5
UTF-8
CRLF
C++
Go Live
Win32
18°C Haze
8:02 PM



PRACTICAL-5

/*Write a program to implement Circular Linked List using templates.

Include functions for insertion, deletion and search of a number, reverse the list.*/

```
#include<iostream>

using namespace std;

template<class t>

class node

{

    public:

     t data;

     node *next;

};


template<class t>

class  clist
```

```
{
    int n;
    node<t> *first,*last;

    public:
        clist()
        {
            first=NULL;
        }

        //create function
        void create()
        {
            node<t> *current,*temp;

            cout<<"Enter how many nodes you want to enter in linked
list:\n";
            cin>>n;
            //fflush(stdin);

            if(n>0)
            {
                    if(first==NULL)
                    {
                            first=new node<t>;
                            cout<<"Enter data for first node:\n";
```

```
                cin>>first->data;
                //fflush(stdin);
                first->next=first;
            }
            current=first;
            for(int i=1;i<n;i++)
            {
                cout<<"Enter data:\n";
                temp=new node<t>;
                cin>>temp->data;
                //fflush(stdin);
                temp->next=current->next;
                current->next=temp;
                current=current->next;
            }
    last=current;
    }
}

//count the list
int count()
{
  node<t> *current;
  current=first;
  int c=0;
  while(current->next!=first)
```

```
    {
                    c++;
                    current=current->next;
    }
    c++;
    return c;
}



//insert function
void insert()
{
    cout<<"Enter the position of insertion;\n";
    cin>>n;
    int b=count();
    if(n<=b+1)
    {
        node<t> *current,*temp;
        current=first;
        temp=new node<t>;
        cout<<"Enter data:\n";
        cin>>temp->data;
        temp->next=NULL;
        if(n==1)
        {
```

```
                temp->next=first;
                first=temp;
                last->next=first;
            }
            else
            {
                for(int i=1;i<n-1;i++)
                current=current->next;

                temp->next=current->next;
                current->next=temp;
            }
    }
    else
    cout<<"Can't be inserted\n";
}


 //deletion now vomes
 void search()
 {
  int flag=0;
    cout<<"Enter data to be searched:\n";
    cin>>n;
    node<t> *current,*prev,*temp;
    int b=count();
```

```
current=first;
for(int i=1;i<=b;i++)
{
    if(current->data==n)
    {
        flag=1;
        break;
    }
    prev=current;
    current=current->next;
}
if(flag==1)
{
    int c;
    cout<<"Data found:\nEnter what you wannna do:\n 1.delete data\n 2.replace it\n 3.do nothing\n";
    cin>>c;
    switch(c)
    {
        case 1:temp=current;
            prev->next=current->next;
            delete(temp);
            cout<<"Data deleted:\n";
            break;
        case 2:cout<<"Enter new data:\n";
            cin>>current->data;
```

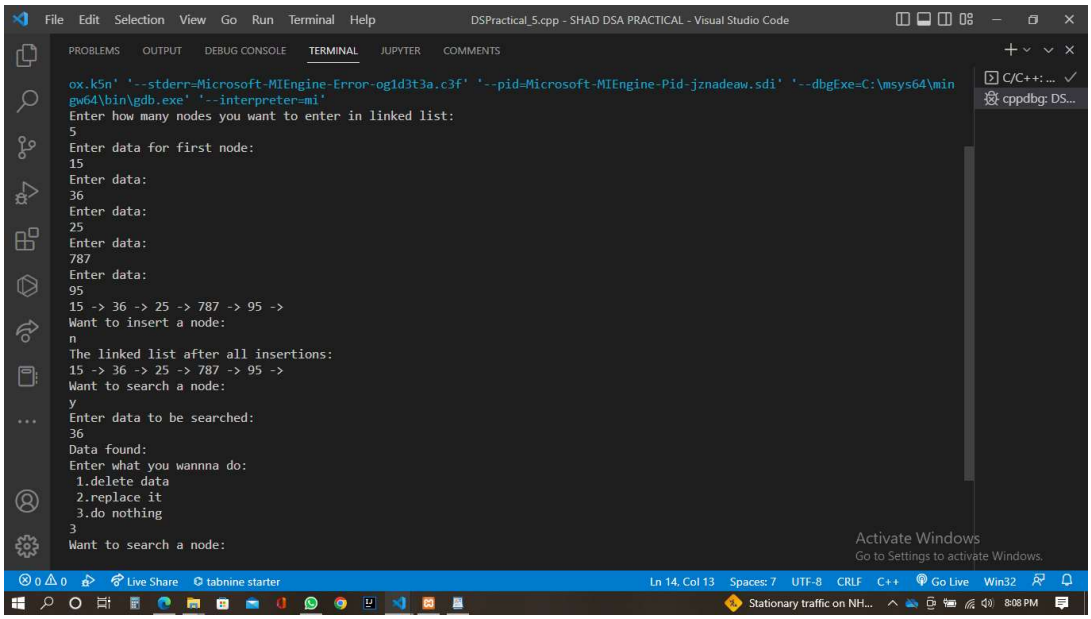
```
                    cout<<"data replaced:\n";
                    break;
                case 3:break;
                default:cout<<"wrong choice:\n";
            }
    }
    else
    cout<<"Data not found:\n";
 }

 //reverse function
void reverse()
{
    node<t> *a,*b,*temp;
    a=first;
    b=a->next;
    temp=b->next;
    a->next=NULL;
    while(temp!=first)
    {
            //fflush(stdin);
            b->next=a;
            a=b;
            b=temp;
            temp=temp->next;
            //this->display();
```

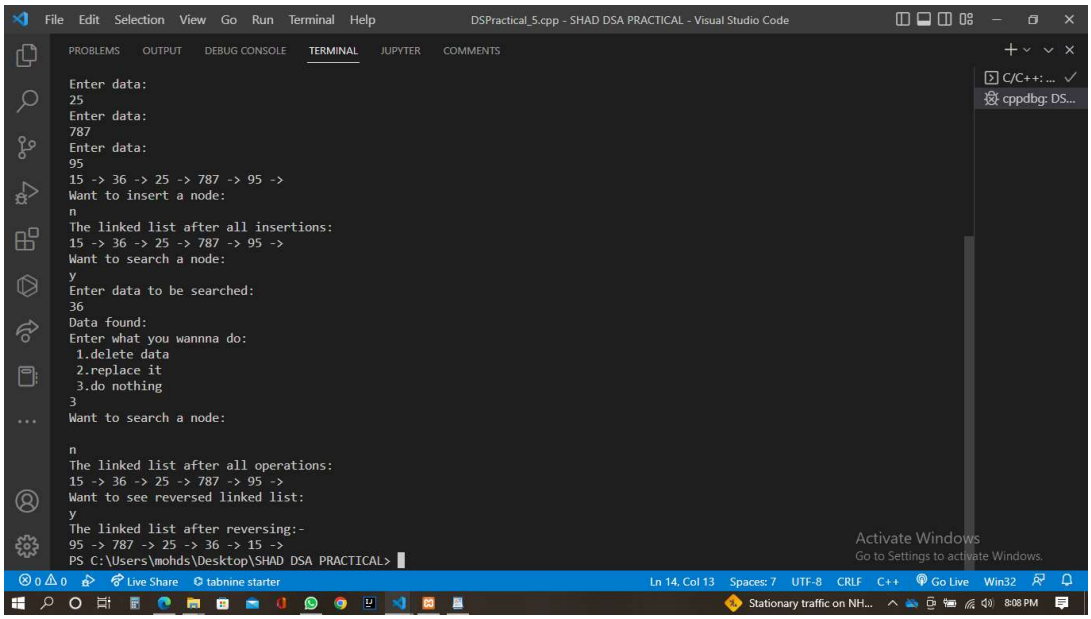
```
            }
            b->next=a;
            first->next=b;
            first=first->next;
          }


    //create display
          void display()
          {
               node<t> *current;
               current=first;
               while(current->next!=first)
               {
                              cout<<current->data<<" -> ";
                              current=current->next;
               }
               cout<<current->data<<" -> \n";
          }


};
int main()
{
   char ch;
   clist<int> l1;
```

```
l1.create();
l1.display();


//doing insertion
do
{
        cout<<"Want to insert a node:\n";
        cin>>ch;
        if(ch=='y')
        l1.insert();
}while(ch=='y');

cout<<"The linked list after all insertions:\n";
l1.display();
do
{
        cout<<"Want to search a node:\n";
        cin>>ch;
        if(ch=='y')
         l1.search();
}while(ch=='y');

cout<<"The linked list after all operations:\n";
l1.display();
```

```
cout<<"Want to see reversed linked list:\n";
cin>>ch;
if(ch=='y')
{
 l1.reverse();
}

cout<<"The linked list after reversing:-\n";
l1.display();

}
```

OUTPUT





PRACTICAL-6

// Perform Stack operations using Linked List implementation.

```
#include <iostream>
using namespace std;

class Node
{
public:
   int data;
   Node *next;
```

```
};

Node *top;

void push(int data)
{

   // Create new node temp and allocate memory in heap
   Node *temp = new Node();

   // Check if stack (heap) is full.
   // Then inserting an element would
   // lead to stack overflow
   if (!temp)
   {
      cout << "\nStack Overflow";
      exit(1);
   }

   // Initialize data into temp data field
   temp->data = data;

   // Put top pointer reference into temp next
   temp->next = top;

   // Make temp as top of Stack
```

```
    top = temp;
}

// Utility function to check if
// the stack is empty or not
int isEmpty()
{
    // If top is NULL it means that
    // there are no elements are in stack
    return top == NULL;
}

// Utility function to return top element in a stack
int peek()
{

    // If stack is not empty , return the top element
    if (!isEmpty())
        return top->data;
    else
        cout << "\nStack is empty! ";
    return -1;
}

void pop()
{
```

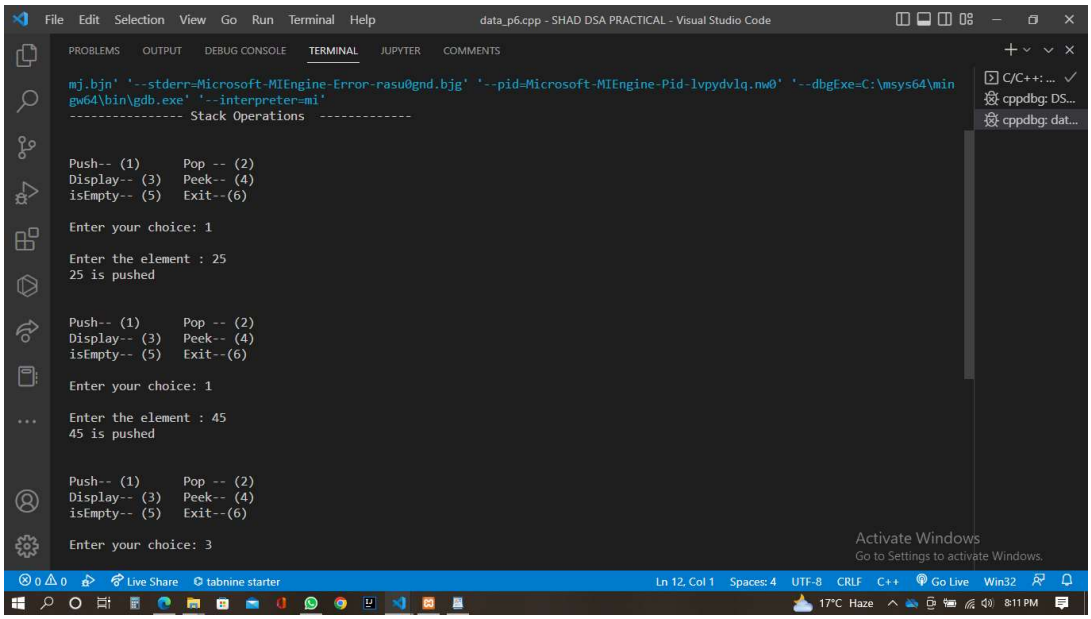
```
    if (top == NULL)
    {
        cout << "\nStack Underflow" << endl;
        exit(1);
    }
    else
    {

        // Assign second node to top
        top = top->next;
    }
}

void display()
{
    Node *temp;

    if (top == NULL)
    {
        cout << "\nStack Underflow";
        exit(1);
    }
    else
    {
        temp = top;
```

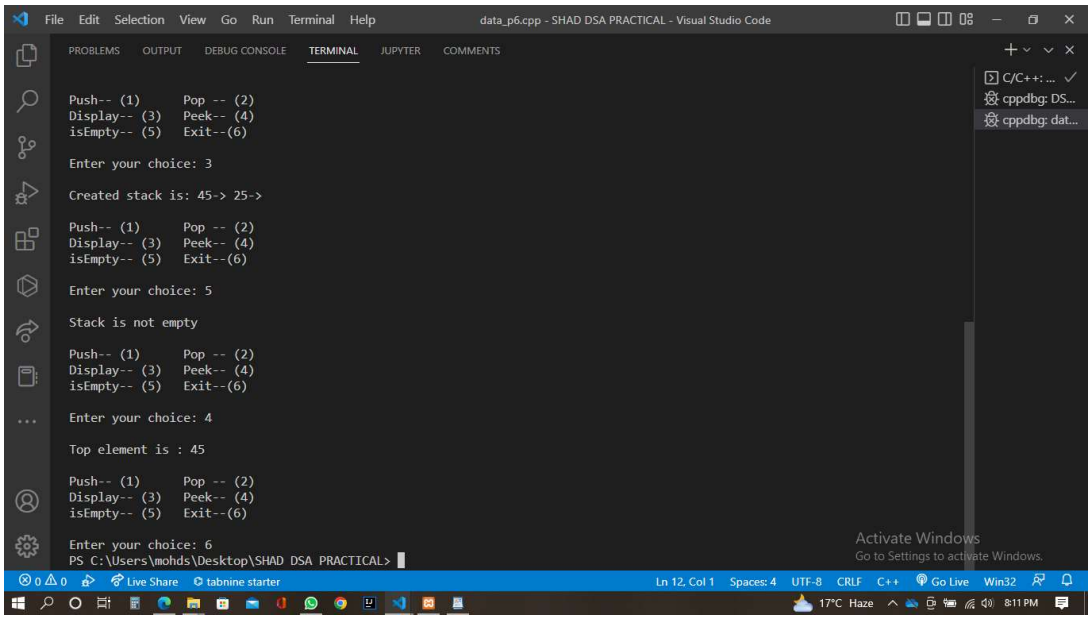
```
        while (temp != NULL)
        {
            cout << temp->data << "-> ";

            // Assign temp link to temp
            temp = temp->next;
        }
    }
}

int main()
{

    Node *head = NULL;

    int ch, element;
    cout << "---------------- Stack Operations  -------------" << endl;
    bool stop = false;
    do
    {
        cout << "\n\nPush-- (1)\tPop -- (2)\nDisplay-- (3)\tPeek-- (4)\nisEmpty--
(5)\tExit--(6)\n ";
        cout << "\nEnter your choice: ";
        cin >> ch;

        switch (ch)
```

```
{
case 1:
   cout << "\nEnter the element : ";
   cin >> element;
   push(element);
   cout << element << " is pushed\n ";
   break;
case 2:
   if (peek() != -1)
   {
      cout << peek() << " is Popped\n ";
      pop();
   }
   else
   {
      pop();
   }
   break;
case 3:
   cout << "\nCreated stack is: ";
   display();
   break;
case 4:
   cout << "\nTop element is : ";
   cout << peek();
   break;
```

```
        case 5:
           if (isEmpty() == 0)
           {
              cout << "\nStack is not empty ";
           }
           else
           {
              cout << "\nStack is empty ";
           }
           break;
        case 6:
           stop = true;
           break;

        default:
           cout << "Invalid option ! ";
           break;
        }
     } while (stop != true);

     return 0;
}
```

OUTPUT


File Edit Selection View Go Run Terminal Help
data_p6.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
mj.bjn' '--stderr=Microsoft-MIEngine-Error-rasu0gnd.bjg' '--pid=Microsoft-MIEngine-Pid-lvpydvlq.nw0' '--dbgExe=C:\msys64\mingw64\bin\gdb.exe' '--interpreter=mi'
---------- Stack Operations ----------
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 1
Enter the element : 25
25 is pushed
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 1
Enter the element : 45
45 is pushed
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 3
C/C++: ...
cppdbg: DS...
cppdbg: dat...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 12, Col 1 Spaces: 4 UTF-8 CRLF C++ Go Live Win32
17°C Haze
8:11 PM


File Edit Selection View Go Run Terminal Help
data_p6.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 3
Created stack is: 45-> 25->
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 5
Stack is not empty
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 4
Top element is : 45
Push-- (1) Pop -- (2)
Display-- (3) Peek-- (4)
isEmpty-- (5) Exit--(6)
Enter your choice: 6
PS C:\Users\mohds\Desktop\SHAD DSA PRACTICAL>
C/C++: ...
cppdbg: DS...
cppdbg: dat...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 12, Col 1 Spaces: 4 UTF-8 CRLF C++ Go Live Win32
17°C Haze
8:11 PM

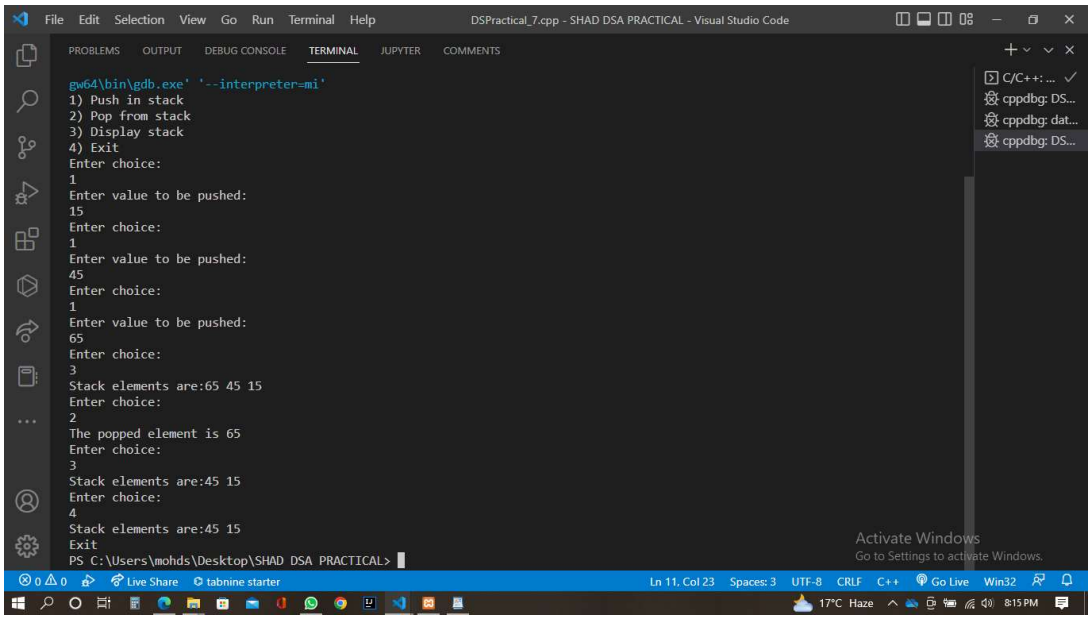
PRACTIAL-7

```
// Perform Stack operations using Array implementation
#include<bits/stdc++.h>
using namespace std;
int stackz[100], n=100, top=-1;

void push(int val) {
  if(top>=n-1)
  cout<<"Stack Overflow"<<endl;
  else {
    top++;
    stackz[top]=val;
  }
}

void pop() {
  if(top<=-1)
  cout<<"Stack Underflow"<<endl;
  else {
    cout<<"The popped element is "<< stackz[top] <<endl;
    top--;
  }
}
```

```
void display() {
  if(top>=0) {
    cout<<"Stack elements are:";
    for(int i=top; i>=0; i--)
    cout<<stackz[i]<<" ";
    cout<<endl;
  } else
  cout<<"Stack is empty";
}

int main()
{
  int ch, val;
  cout<<"1) Push in stack"<<endl;
  cout<<"2) Pop from stack"<<endl;
  cout<<"3) Display stack"<<endl;
  cout<<"4) Exit"<<endl;
  do {
    cout<<"Enter choice: "<<endl;
    cin>>ch;
    switch(ch) {
      case 1: {
        cout<<"Enter value to be pushed:"<<endl;
        cin>>val;
        push(val);
        break;
```

```
        }
        case 2: {
          pop();
          break;
        }
        case 3: {
          display();
          break;
        }
        case 4: {
          display();
          cout<<"Exit"<<endl;
          break;
        }
        default: {
          cout<<"Invalid Choice"<<endl;
        }
      }
    }while(ch!=4);


}
```

OUTPUT



PRACTICAL-8

```
#include <iostream>

using namespace std;

#define SIZE 100

int A[SIZE];

int front = -1;

int rear = -1;


//Function to check if queue is empty or not

bool isempty()

{

 if(front == -1 && rear == -1)

 return true;

 else

 return false;
```

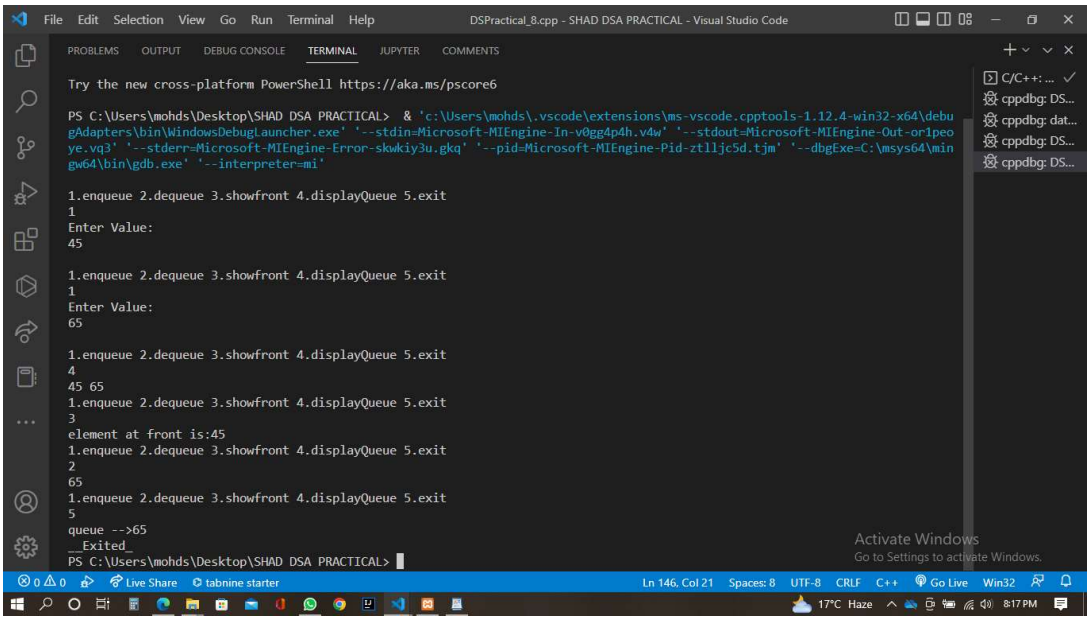
```
}
void displayQueue()
{
 if(isempty())
  cout<<"Queue is empty\n";
 else
 {
  int i;
  if( front <= rear )
  {
   for( i=front ; i<= rear ; i++)
   cout<<A[i]<<" ";
  }
  else
  {
   i=front;
   while( i < SIZE)
   {
   cout<<A[i]<<" ";
   i++;
   }
   i=0;
   while( i <= rear)
   {
   cout<<A[i]<<" ";
   i++;
```

```
  }
 }
}
}

//function to enter elements in queue
void enqueue ( int value )
{
 //queue is full
 if ((rear + 1)%SIZE == front)
   cout<<"Queue is full \n";
 else
 {
  //first element inserted
  if( front == -1)
    front = 0;
 //insert element at rear
 rear = (rear+1)%SIZE;
  A[rear] = value;
 }
}

//function to delete/remove element from queue
void dequeue ( )
{
 if( isempty() )
```

```
 cout<<"Queue is empty\n";
else
//only one element
if( front == rear )
 front = rear = -1;
else
 front = (front + 1)%SIZE;


 displayQueue();
}

//function to show the element at front
void showfront( )
{
 if( isempty())
 cout<<"Queue is empty\n";
 else
 cout<<"element at front is:"<<A[front];
}

//function to display queue


//Main Function
int main()
```

```
{
 int choice, value;
  do
  {
  cout<<"\n1.enqueue 2.dequeue 3.showfront 4.displayQueue 5.exit\n";
     cin>>choice;
  switch (choice)
  {
  case 1: cout<<"Enter Value:\n";
         cin>>value;
         enqueue(value);
         break;
  case 2: dequeue();
         break;
  case 3: showfront();
         break;
  case 4: displayQueue();
         break;
  case 5:
  cout<<"queue -->";
  displayQueue();
  cout<<"\n__Exited_";
  break;
  }
  } while (choice!=5);
}
```

OUTPUT



PRACTICAL-9

```
#include <iostream>
using namespace std;
#define SIZE 10
class deque
{
   int a[20], f, r;

public:
   deque();
   void insert_at_beg(int);
   void insert_at_end(int);
   void delete_fr_front();
   void delete_fr_rear();
   void show();
```

```
   void userInput();
};
deque::deque()
{
   f = -1;
   r = -1;
}
void deque::insert_at_end(int i)
{
   if (r >= SIZE - 1)
   {
      cout << "\n insertion is not possible, overflow!!!!";
   }
   else
   {
      if (f == -1)
      {
         f++;
         r++;
      }
      else
      {
         r = r + 1;
      }
      a[r] = i;
      cout << "\nInserted item is " << a[r];
```

```
    }
}
void deque::insert_at_beg(int i)
{
    if (f == -1)
    {
        f = 0;
        a[++r] = i;
        cout << "\n Inserted element is: " << i;
    }
    else if (f != 0)
    {
        a[--f] = i;
        cout << "\n Inserted element is: " << i;
    }
    else
    {
        cout << "\n Insertion is not possible, overflow!!!";
    }
}
void deque::delete_fr_front()
{
    if (f == -1)
    {
        cout << "deletion is not possible::deque is empty";
        return;
```

```
    }
    else
    {
        cout << "the deleted element is: " << a[f];
        if (f == r)
        {
            f = r = -1;
            return;
        }
        else
            f = f + 1;
    }
}
void deque::delete_fr_rear()
{
    if (f == -1)
    {
        cout << "deletion is not possible::deque is empty";
        return;
    }
    else
    {
        cout << "the deleted element is: " << a[r];
        if (f == r)
        {
            f = r = -1;
```

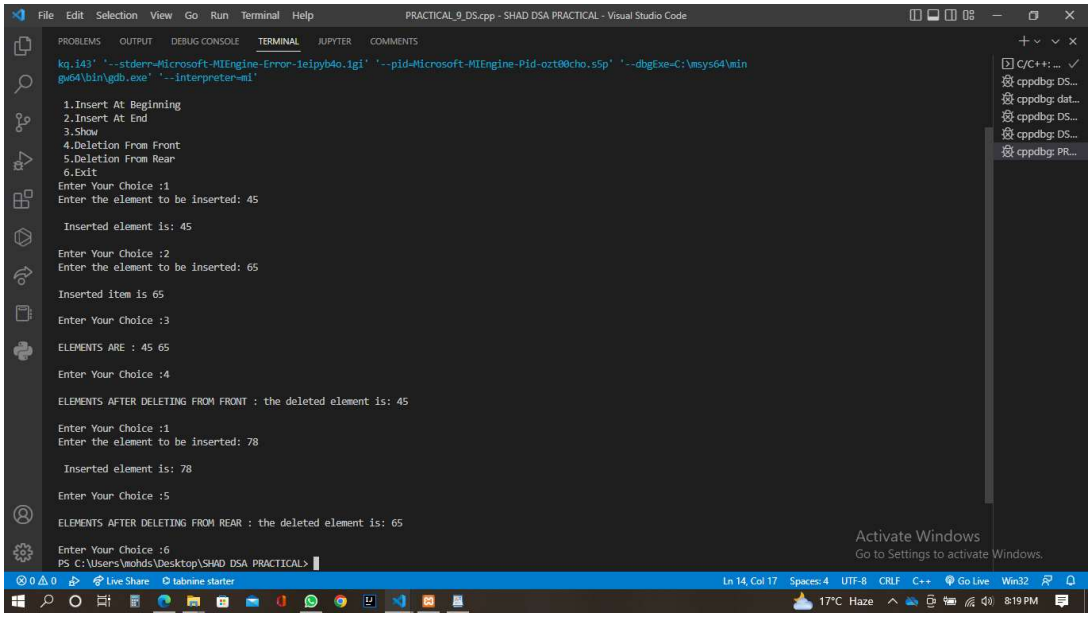
```
        }
        else
            r = r - 1;
    }
}
void deque::show()
{
    if (f == -1)
    {
        cout << "deque is empty! ";
    }
    else
    {
        for (int i = f; i <= r; i++)
        {
            cout << a[i] << " ";
        }
    }
}

void deque::userInput()
{
    int c, i;
    deque d;
    cout << "\n 1.Insert At Beginning";
    cout << "\n 2.Insert At End";
```

```
cout << "\n 3.Show";
cout << "\n 4.Deletion From Front";
cout << "\n 5.Deletion From Rear";
cout << "\n 6.Exit";
do
{
   cout << "\nEnter Your Choice :";
   cin >> c;
   switch (c)
   {
   case 1:
      cout << "Enter the element to be inserted: ";
      cin >> i;
      d.insert_at_beg(i);
      cout << endl;
      break;
   case 2:
      cout << "Enter the element to be inserted: ";
      cin >> i;
      d.insert_at_end(i);
      cout << endl;
      break;
   case 3:
      cout << "\nELEMENTS ARE : ";
      d.show();
      cout << endl;
```

```
            break;
        case 4:
            cout << "\nELEMENTS AFTER DELETING FROM FRONT : ";
            d.delete_fr_front();
            cout << endl;
            break;
        case 5:
            cout << "\nELEMENTS AFTER DELETING FROM REAR : ";
            d.delete_fr_rear();
            cout << endl;
            break;
        case 6:
            exit(1);
            break;
        default:
            cout << "Invalid choice! ";
            break;
        }
    }

    while (c != 7);
}
int main()
{
    deque x;
    x.userInput();
```

}

OUTPUT



PRACTICAL-10

```
#include <iostream>
using namespace std;
template<class T>
struct node
{
   T data;
   node* next;
};
template<class T>
class dequeue {
   private:
   node<T>* head=NULL;
   node<T>* end=NULL;
```

```
    public:


int push_back(T data){
    node<T>* temp=new node<T>();
    temp->data=data;
    temp->next=NULL;
    end=temp;
    if (head==NULL)
    {
        head=temp;

        return 0;
    }

    node<T>* temp1=head;
    while (temp1->next!=NULL)
    {
        temp1=temp1->next;
    }

    temp1->next=temp;

    return 0;
}
```

```
void pop_back()
{
   node<T>* elem=head;
   if(head==NULL)
   {
      cout<<"empty";
   }
   else if(head->next==NULL)
   {
      delete head;
      head=NULL;
      end=NULL;
   }
   else{

   while (elem->next!=end)
   {
      elem=elem->next;
   }
   elem->next=NULL;
   delete end;
   end=elem;
   }

}
void display()
```

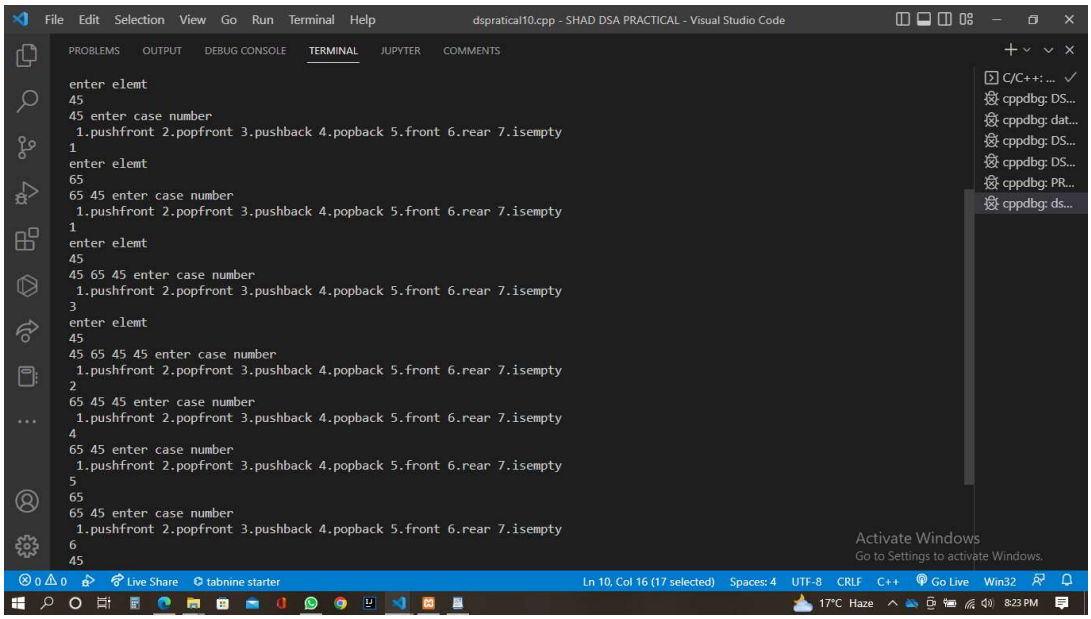
```
{   node<T>* temp=head;
    if(head==NULL){
        cout<<"empty"<<endl;
    }
    else{

    while (temp!=NULL)
    {
        cout<<temp->data<<" ";
        temp=temp->next;
    }
    }

}
void pushfront(T data)
{
    node<T>* temp=new node<T>();
    temp->data=data;

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

```
   temp->next=head;//connect temp to head locating element
   head=temp;

   }
}
void popfront()
{   node<T>* prevelem=head;
   node<T>* temp;//deleting elem
   temp=head;
   head=prevelem->next;
   delete temp;
}
void front(){
   cout<<head->data<<endl;
}
void rear(){
   cout<<end->data<<endl;
}

};

int main(){


   dequeue<int> *obj=new dequeue<int>() ;
```

```
   int n,i=1;
   int num,posi;
   while (i>0)
   {
      cout<<"enter case number\n 1.pushfront 2.popfront 3.pushback
4.popback 5.front 6.rear 7.isempty\n";
      cin>>n;
   switch (n)
   {
   case 1:
      cout<<"enter elemt\n";
      cin>>num;
      obj->pushfront(num);
      obj->display();
      break;

   case 2:
      obj->popfront();
      obj->display();
      break;

   case 3:
      cout<<"enter elemt\n";
      cin>>num;
      obj->push_back(num);
      obj->display();
```

```
    break;
  case 4:
    obj->pop_back();
    obj->display();
    break;
  case 5:
    obj->front();
    obj->display();
    break;
  case 6:
    obj->rear();
  case 7:
    obj->display();
    break;

  default:
    break;
  }
  }
}
```

OUTPUT



PRACTICAL-11

```
#include <iostream>
using namespace std;
struct node
{
   int data;
   node *right;
   node *left;
};
//by irtative
class binarysearchtree
{
   node* root=NULL;
```

```
public:
int insertion(int data){
   node *temp=new node();
   node *parent=root;
   temp->data=data;
   temp->left=NULL;
   if(root==NULL)
   {
      temp->right=NULL;
      root=temp;
   }
   else{
      while (temp!=NULL)
      {
      if(data<parent->data)//if parent data less than data move left
      {   if(parent->left==NULL)//if parent left is null than connt parent to
temp
         {
            temp->right=NULL;
            parent->left=temp;
            break;
         }
         parent=parent->left;
      }
      else if(data>parent->data)
      {   if(parent->right==NULL)
```

```
            {
               temp->right=NULL;
               parent->right=temp;
               break;
            }
            parent=parent->right;
         }
         else{
            cout<<"copy elemt found\n";
            return 0;
         }
         }
      }
      return 0;
   }
   //by recirsion
   node *rinsertiion(node *parent,node *temp)
   {
      if(parent==NULL)
      {
          temp->right=NULL;
          root=temp;
          return 0;
      }
      else if(temp->data<parent->data)//if parent data less than data move
left
```

```
        {   if(parent->left==NULL)//if parent left is null than connt parent to
temp
            {
               temp->right=NULL;
               parent->left=temp;
               return 0;
            }
            parent=parent->left;
        }
       else if(temp->data>parent->data)
        {   if(parent->right==NULL)
            {
               temp->right=NULL;
               parent->right=temp;
               return 0;
            }
            parent=parent->right;
        }
        else{
            cout<<"copy elemt found\n";
            return 0;
        }
        return rinsertiion(parent,temp);
   }
   node *search(int data)
   {
```

```
    node *parent=root;
    if(parent==NULL)
    {
      cout<<"NO\n";
      return NULL;
    }
    while (parent!=NULL)//infinte loop
      {
      if(data<parent->data)//if parent data less than data move left
      {   if(parent->left==NULL)//if parent left is null than connt parent to
temp
        {
          cout<<"NO\n";
          return NULL;
        }
        parent=parent->left;
      }
      else if(data>parent->data)
      {   if(parent->right==NULL)
        {
          cout<<"NO\n";
          return NULL;
        }
        parent=parent->right;
      }
      else{
```

```
            cout<<"copy elemt found\n";
            break;
        }
        }
    return parent;
}

int remove(int data)
{
    node *parent=search(data);
    node *actualdelete=parent;
    int min=0;
    if(parent==NULL)
    {
        cout<<"NOt exist\n";
    }
    else{
        if(parent->left==NULL && parent ->right==NULL)
        {
            delete parent;
        }
        else if(parent->left!=NULL )
        {
            parent=parent->left;
            int i=0,min=0;
            node *parentback;
```

```
        do
        {  if(i!=0)
           {
              parentback=parent;
           }
           min=parent->data;
           i++;
        }while (parent->right!=NULL&&(parent=parent->right));
        if(parent->left!=NULL)
        {
           parentback->right=parent->left;
        }
        delete parent;
        actualdelete->data=min;
     }
     else if(parent->left==NULL && parent ->right!=NULL)
     {
        parent=parent->right;
        do
        {
           min=parent->data;
        }while (parent->left!=NULL&&(parent=parent->left));
        delete parent;
        actualdelete->data=min;
     }
  }
```

```
    return 0;
}
void displayheight()
{
   cout<<height(root)<<endl;
}
int height(node *temp)
{
    if (temp==NULL)
    {
        return -1;
    }
    else{
        int ldepth=height(temp->left);
        int rdepth=height(temp->right);

        if (ldepth > rdepth)
        return (ldepth + 1);
    else
        return (rdepth + 1);
    }
    return 0;
}
void displaylevelbylevel(){
    node *temp=root;
    int h=height(temp);
```

```
   for (int i = 1; i < h; i++)
   {
      printcurrentlevel(temp,i);
   }
}
void printcurrentlevel(node *temp,int h)
{
   if(root==NULL)
   {
      return;
   }
   if(h==1)
   {
      cout<<root->data<<" ";
   }
   else if(h>1)
   {
      printcurrentlevel(temp->left,h-1);
      printcurrentlevel(temp->right,h-1);
   }
}
void displayinorder(){
   printinorder(root);
}
void printinorder(node* temp)
{
```

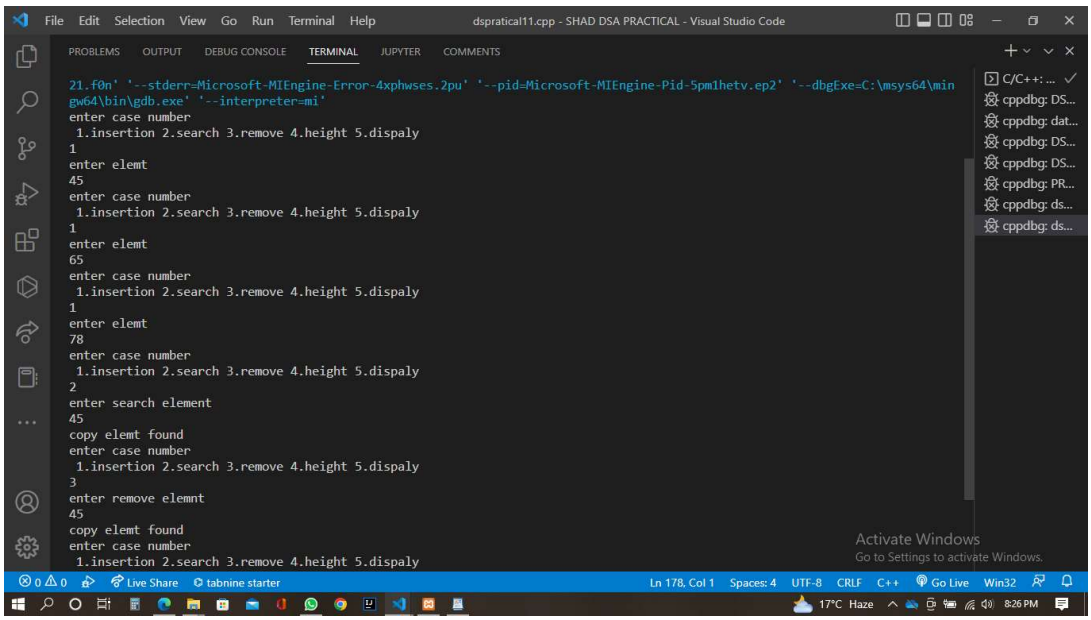
```
   if (temp==NULL)
   {
      return;
   }
   printinorder(temp->left);
   cout<<temp->data<<" ";
   printinorder(temp->right);
}
void displaypreorder(){
   printpreorder(root);
}
void printpreorder(node* temp)
{
   if (temp==NULL)
   {
      return;
   }
   cout<<temp->data<<" ";
   printpreorder(temp->left);
   printpreorder(temp->right);
}
void displaypostorder(){
   printpostorder(root);
}
void printpostorder(node* temp)
{
```

```
        if (temp==NULL)
        {
           return;
        }
        printpostorder(temp->left);
        printpostorder(temp->right);
        cout<<temp->data<<" ";
    }



};
int main(){
     binarysearchtree *obj=new binarysearchtree() ;
    int n,i=1;
    int num,posi;
    while (i>0)
    {
        cout<<"enter case number\n 1.insertion 2.search 3.remove 4.height
5.dispaly\n";
        cin>>n;
    switch (n)
    {
    case 1:
        cout<<"enter elemt\n";
        cin>>num;
        obj->insertion(num);
```

```
    break;
case 2:
    cout<<"enter search element\n";
    cin>>num;
    obj->search(num);
    break;
case 3:
    cout<<"enter remove elemnt\n";
    cin>>num;
    obj->remove(num);
    break;
case 4:
    cout<<"height of tree";
    obj->displayheight();
    break;
case 5:
    cout<<"output\n";
    obj->displaylevelbylevel();
    break;
case 6:
    cout<<"inorder\n";
    obj->displayinorder();
    cout<<"preorder\n";
    obj->displaypreorder();
    cout<<"postorder\n";
    obj->displaypostorder();
```

```
    }
    }
}
```

OUTPUT


File Edit Selection View Go Run Terminal Help
dspratical11.cpp - SHAD DSA PRACTICAL - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER COMMENTS
21.f0n' '--stderr=Microsoft-MIEngine-Error-4xphwses.2pu' '--pid=Microsoft-MIEngine-Pid-5pm1hetv.ep2' '--dbgExe=C:\msys64\mingw64\bin\gdb.exe' '--interpreter=mi'
enter case number
1.insertion 2.search 3.remove 4.height 5.dispaly
1
enter elemt
45
enter case number
1.insertion 2.search 3.remove 4.height 5.dispaly
1
enter elemt
65
enter case number
1.insertion 2.search 3.remove 4.height 5.dispaly
1
enter elemt
78
enter case number
1.insertion 2.search 3.remove 4.height 5.dispaly
2
enter search element
45
copy elemt found
enter case number
1.insertion 2.search 3.remove 4.height 5.dispaly
3
enter remove elemnt
45
copy elemt found
enter case number
1.insertion 2.search 3.remove 4.height 5.dispaly
C/C++: ...
cppdbg: DS...
cppdbg: dat...
cppdbg: DS...
cppdbg: DS...
cppdbg: PR...
cppdbg: ds...
cppdbg: ds...
Activate Windows
Go to Settings to activate Windows.
Live Share
tabnine starter
Ln 178, Col 1
Spaces: 4
UTF-8
CRLF
C++
Go Live
Win32
17°C Haze
8:26 PM