Muhammad Mufeez

BCS-3J

DS Lab 06

Q1:

Code:

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

using namespace std;
template <typename T>
class Stack
{
    T *arr;
    int size;
    int top;

public:
    Stack(int s)
    {
        size = s;
        top = -1;
        arr = new T[size];
    }

    void push(T c)
    {
```

```
        if (top < size - 1)
        {
            arr[++top] = c;
        }
        else
        {
            cout << "Stack OverFlow" << endl;
        }
    }

    void pop()
    {
        if (top >= 0)
        {
            top--;
        }
        else
        {
            cout << "Stack is Empty" << endl;
        }
    }

    T peek()
    {
        if (top >= 0 && top < size)
        {
            return arr[top];
```

```
        }
        else
        {
            cout << "Stack is Empty" << endl;
            return -1;
        }
    }

    bool isEmpty()
    {
        return top == -1;
    }
};

int main()
{
    string st = "BORRORROB";
    Stack<char> st1(st.length());
    string ans;
    for(int i=0;i<st.length();++i){
        st1.push(st[i]);
    }



    for(int i=st.length();i>0;--i){
        ans += st1.peek();
```

```
        st1.pop();
    }




    if (ans==st)
        cout << "It is a palindrome" << endl;
    else
        cout << "It is not a palindrome" << endl;
}
```

Output:

```
It is a palindrome

--------------------------------
Process exited after 0.06433 seconds with return value 0
Press any key to continue
```

Q2:

Code:

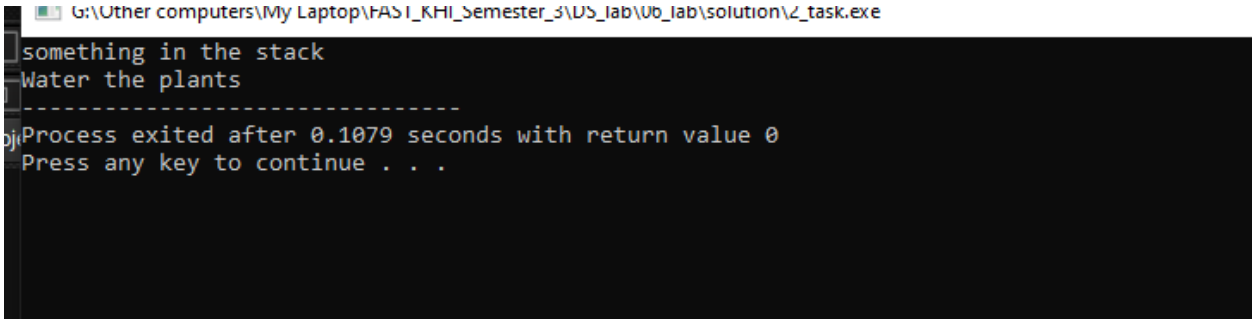
```
#include <bits/stdc++.h>



using namespace std;

template <typename T>

class Stack

{

    T *arr;

    int size;

    int top;
```

```
public:
    Stack(int s)
    {
        size = s;
        top = -1;
        arr = new T[size];
    }

    void push(T c)
    {
        if (top < size - 1)
        {
            arr[++top] = c;
        }
        else
        {
            cout << "Stack OverFlow" << endl;
        }
    }

    void pop()
    {
        if (top >= 0)
        {
            top--;
        }
        else
```

```
        {
            cout << "Stack is Empty" << endl;
        }
    }

    T peek()
    {
        if (top >= 0 && top < size)
        {
            return arr[top];
        }
        else
        {
            cout << "Stack is Empty" << endl;
            return T();
        }
    }

    bool isEmpty()
    {
        return top == -1;
    }
};

int main()
{
    Stack<string> stack1(5);
```

```
    stack1.push("Water the plants");
    stack1.push("Do the lab tasks :(");
    stack1.pop();


    if(stack1.isEmpty()){
        cout<<"Nothing in the stack"<<endl;
    }
    else{
        cout<<"something in the stack"<<endl;
    }


    cout<<stack1.peek();
}
```

Output:



Q3:

Code:

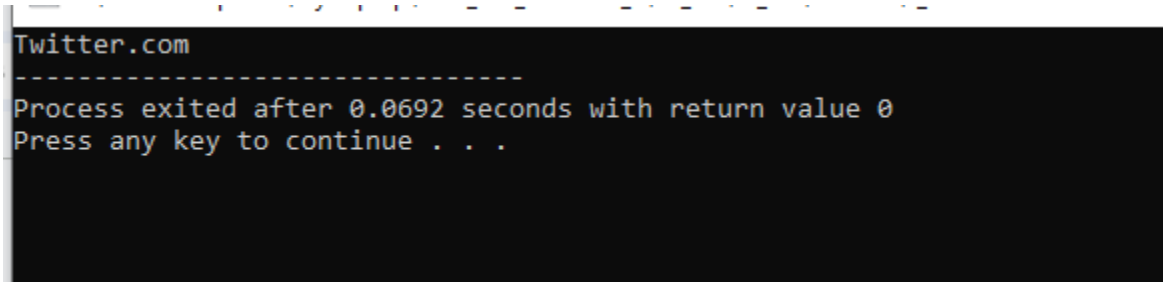
```
#include <bits/stdc++.h>


using namespace std;
```

```
template <typename T>
class Node
{
public:
    T url;
    Node *next;
    Node *prev;
    Node(T url)
    {
        this->url = url;
        this->next = NULL;
        this->prev = NULL;
    }
};


template <typename T>
class LLStack
{
    Node<T> *top;


public:
    LLStack(T url)
    {
        top = new Node<T>(url);
    }


    void visit(T homepage)
```

```
{
    Node<T> *newNode = new Node<T>(homepage);
    top->next = newNode;
    newNode->prev = top;
    top = newNode;
}

T back(int steps)
{
    while (steps)
    {
        if (top->prev)
            top = top->prev;
        else
            break;
        steps--;
    }
    return top->url;
}
T forward()
{
    if (top->next)
        top = top->next;
    return top->url;
}
T current()
{
```

```
        if (!top)
            return T();
        return top->url;
    }
    ~LLStack()
    {
        while (top && top->prev)
        {
            Node<T> *prev = top->prev;
            top = top->prev;
            delete prev;
        }
    }
};

int main()
{
    LLStack<string> web1("Google.cam");
    web1.visit("Facebook.com");
    web1.visit("Twitter.com");
    web1.visit("Linkedin.com");
    web1.visit("Instagram.com");

    web1.back(2);
    cout << web1.current();
    return 0;
}
```

Output:



Q4:

Code:

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


using namespace std;

template <typename T>

class Stack

{

    T *arr;

    int size;

    int top;


public:

    Stack(int s)

    {

        size = s;

        top = -1;

        arr = new T[size];

    }
```

```
    void push(T c)
    {
        if (top < size - 1)
        {
            arr[++top] = c;
        }
        else
        {
            cout << "Stack OverFlow" << endl;
        }
    }

    void pop()
    {
        if (top >= 0)
        {
            top--;
        }
        else
        {
            cout << "Stack is Empty" << endl;
        }
    }

    T peek()
    {
        if (top >= 0 && top < size)
```

```
        {
            return arr[top];
        }
        else
        {
//            cout << "Stack is Empty" << endl;
            return T();
        }
    }

    bool isEmpty()
    {
        return top == -1;
    }
};

int returnPrecedence(char c)
{
    if (c == '^')
    {
        return 3;
    }
    else if (c == '*' || c == '/')
    {
        return 2;
    }
    else if (c == '+' || c == '-')
```

```
    {
        return 1;
    }
    return -1;
}
string infixToPostfix(string exp)
{
    int len = exp.length();
    string ans = "";
    Stack<char> stack(len);
    for (int i = 0; i < len; ++i)
    {
        if ((exp[i] >= 'a' && exp[i] <= 'z'))
        {
            ans += exp[i];
        }
        else if (exp[i] == '(')
        {
            stack.push(exp[i]);
        }
        else if (exp[i] == ')')
        {
            while (!stack.isEmpty() && stack.peek() != '(')
            {
                ans += stack.peek();
                stack.pop();
            }
```

```
            if (!stack.isEmpty())
                stack.pop();
        }
        else
        {
            while ((returnPrecedence(stack.peek()) >=
returnPrecedence(exp[i])) && !stack.isEmpty())
            {
                ans += stack.peek();
                stack.pop();
            }
            stack.push(exp[i]);
        }
    }
    while (!stack.isEmpty())
    {
        ans += stack.peek();
        stack.pop();
    }
    return ans;
}

int main()
{
    string equation = "a+b*(c^d-e)^(f+g*h)-i";
    cout << "Post Fix: " << infixToPostfix(equation);
    return 0;
```

```
};
```

Output:

```
Post Fix: abcd^e-fgh*+^*+i-
--------------------------------
Process exited after 0.0223 seconds with return value 0
Press any key to continue . . .
```

Q5:

Code:

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


using namespace std;

template <typename T>

class Node

{

public:

   T data;

   Node *next;

   Node(T data)

   {

        this->data = data;

        this->next = NULL;

   }

};


template <typename T>

class LLStack
```

```
{
public:
   Node<T> *head;
   // int Max;
   int curr;
   LLStack()
   {
       // this->Max = Max;
       head = NULL;
       curr = 0;
   }
   void push(T data)
   {
       Node<T> *newNode = new Node<T>(data);
       newNode->next = head;
       head = newNode;
       ++curr;
   }
   T peek()
   {
       if (curr == 0)
       {
           cout << "Stack is empty" << endl;
           return T();
       }
       return head->data;
   }
```

```
    void pop()
    {
        if (curr == 0)
        {
            cout << "Stack is empty" << endl;
            return;
        }
        head = head->next;
        --curr;
    }
    bool isEmpty() { return curr == 0; }
    ~LLStack()
    {
        while (head)
        {
            Node<T> *prev = head->next;
            head = head->next;
            delete prev;
        }
    }
};


template <typename T>
class Stack
{
    T *arr;
    int size;
```

```
    int top;

public:
    Stack(int s)
    {
        size = s;
        top = -1;
        arr = new T[size];
    }

    void push(T c)
    {
        if (top < size - 1)
        {
            arr[++top] = c;
        }
        else
        {
            cout << "Stack OverFlow" << endl;
        }
    }

    void pop()
    {
        if (top >= 0)
        {
            top--;
```

```
        }
        else
        {
            cout << "Stack is Empty" << endl;
        }
    }

    T peek()
    {
        if (top >= 0 && top < size)
        {
            return arr[top];
        }
        else
        {
            cout << "Stack is Empty" << endl;
            return -1;
        }
    }

    bool isEmpty()
    {
        return top == -1;
    }
};

int returnPrecedence(string c)
```

```
{
    if (c == "^")
    {
        return 3;
    }
    else if (c == "*" || c == "/")
    {
        return 2;
    }
    else if (c == "+" || c == "-")
    {
        return 1;
    }
    return -1;
}
vector<string> infixToPostfix(vector<string> &exp)
{
    int len = exp.size();
    vector<string> ans;
    LLStack<string> stack;
    ans.push_back(exp[0]);
    ans.push_back(exp[1]);
    for (int i = 2; i < len; ++i)
    {
        if ((exp[i] >= "0" && exp[i] <= "999"))
        {
            ans.push_back(exp[i]);
```

```
        }
        else if (exp[i] == "(")
        {
            stack.push(exp[i]);
        }
        else if (exp[i] == ")")
        {
            while (!stack.isEmpty() && stack.peek() != "(")
            {
                ans.push_back(stack.peek());

                stack.pop();
            }
            if (!stack.isEmpty())
                stack.pop();
        }
        else
        {
            while (!stack.isEmpty() && (returnPrecedence(stack.peek()) >=
returnPrecedence(exp[i])))
            {
                ans.push_back(stack.peek());

                stack.pop();
            }
            stack.push(exp[i]);
        }
```

```
    }
    while (!stack.isEmpty())
    {
        ans.push_back(stack.peek());

        stack.pop();
    }
    return ans;
}
double evaluatePostfix(vector<string> &postfix)
{
    LLStack<double> stack;
    stack.push(stod(postfix[2]));
    stack.push(stod(postfix[3]));

    for (int i = 4; i < postfix.size(); ++i)
    {
        if (postfix[i] != "+" && postfix[i] != "-" && postfix[i] != "*" &&
postfix[i] != "/" && postfix[i] != "^")
        {
            stack.push(stod(postfix[i]));
        }
        else
        {
            double operand2 = stack.peek();
            stack.pop();
            double operand1 = stack.peek();
```

```
            stack.pop();
            double result = 0;

            if (postfix[i] == "+")
            {
                result = operand1 + operand2;
            }
            else if (postfix[i] == "-")
            {
                result = operand1 - operand2;
            }
            else if (postfix[i] == "*")
            {
                result = operand1 * operand2;
            }
            else if (postfix[i] == "/")
            {
                result = operand1 / operand2;
            }

            stack.push(result);
        }
    }
    return stack.peek();
}
int main()
{
```

```
   vector<string> equation = {"x", "=", "12", "+", "13", "-", "5", "*",
"(", "0.5", "+", "0.5", ")", "+", "1"};


   vector<string> postfix = infixToPostfix(equation);
   for (int i = 0; i < equation.size(); ++i)
   {
       cout << postfix[i] << " ";
   }
       cout <<endl;


   cout<< postfix[0]<<" "<<postfix[1]<<" "<<evaluatePostfix(postfix);
   return 0;
};
```

Output:

```
cd "/media/mufeez/work1/FAST_KHI_Semester_3
b/06_lab/solution/"5_task
mufeez@mine:/media/mufeez/work1/FAST_KHI_Se
& g++ 5_task.cpp -o 5_task && "/media/mufee
x = 12 13 + 5 0.5 0.5 + * - 1 +
x = 21mufeez@mine:/media/mufeez/work1/FAST_
```

Q6:

Code:

```
/*
The data structure used for the described problem is queue data structure
```

```
push will add the item and pop will remove it


Each element in the queue is an item, to remove multiple items, call the
pop function that number of time



function display data will print all orders
*/


#include <bits/stdc++.h>
using namespace std;
template <typename T>
class CircularQueueArray
{
public:
   int start;
   int end;
   int size;
   int max;
   T *arr;


   CircularQueueArray(int max)
   {
       this->max = max;
       start = end = -1;
       size = 0;
       arr = new T[max];
```

```
}

void push(T data)
{
    if (start == -1 && end == -1)
    {
        start = end = 0;
        arr[end] = data;
        ++size;
    }
    else
    {
        end = (end + 1) % max;
        arr[end] = data;
        ++size;
    }
}

void pop()
{
    if (size == 0)
    {
        cout << "Queue is empty" << endl;
        return;
    }
    start = (start + 1) % max;
    --size;
```

```
    }

    T top()
    {
        if (start != -1)
            return arr[start];
        return T();
    }

    void displayData(){
        if(size ==0) cout<<"No items ordered"<<endl;
        int count= (start<= end) ? end - start: start-end;
        for(int i=0;i<count ;++i){
            cout<<"Item "<<i<<" :"<<arr[i]<<endl;
        }
    }
};

int main()
{
    return 0;
}
```

Q7:

Code:

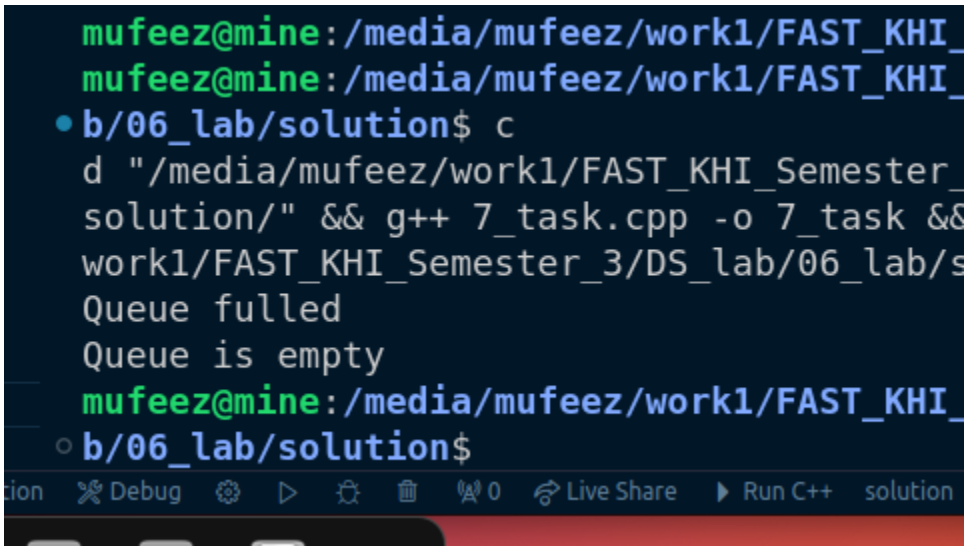
```cpp
// C++ Program to implement a queue using array

#include <bits/stdc++.h>

using namespace std;

template <typename T>

class Queue

{

public:

   int start;

   int end;

   int size;

   int max;

   T *arr;


   Queue(int max)

   {

       this->max = max;

       start = end = -1;

       size = 0;

       arr = new T[max];

   }


   void push(T data)

   {

       if (size == max - 1)

       {

           cout << "Queue fulled " << endl;

       }
```

```
    else if (start == -1 && end == -1)
    {
        start = end = 0;
        arr[end] = data;
        ++size;
    }
    else
    {
        end++;
        arr[end] = data;
        ++size;
    }
}

void pop()
{
    if (size == 0)
    {
        cout << "Queue is empty" << endl;
        return;
    }
    start++;
    --size;
}

T top()
{
```

```
        if (start != -1)
            return arr[start];
        return T();
    }
    int getSize(){ return size;}
};

int main()
{
    Queue<int> cashier(7); // max size of queue
    cashier.push(13);
    cashier.push(7);
    cashier.push(4);
    cashier.push(1);
    cashier.push(6);
    cashier.push(8);
    cashier.push(10);
    // above are enqueue operations, below are dequeue

    cashier.pop();
    cashier.pop();
    cashier.pop();
    cashier.pop();
    cashier.pop();
    cashier.pop();
    cashier.pop();
    return 0;
```

```
}
```

Output:



Q9:

Code:

```
/*
input values:


nothing
something
anything
*/


#include <bits/stdc++.h>


using namespace std;
```

```
template <typename T>
class Node
{
public:
   T data;
   Node *next;
   Node(T data) : data(data)
   {
       next = NULL;
   }
};
template <typename T>
class QueueLL
{
public:
   int size;
   Node<T> *start;
   Node<T> *end;

   QueueLL()
   {
       size = 0;
       start = NULL;
       end = NULL;
   }

   void push(T data)
```

```
{
    if (start == NULL)
    {
        start = new Node<T>(data);
        end = start;
        ++size;
    }
    else
    {
        Node<T> *temp = new Node<T>(data);
        end->next = temp;
        end = end->next;
        ++size;
    }
}

void pop()
{
    if (start)
    {
        Node<T> *temp = start;
        start = start->next;
        --size;
        delete temp;
        return;
    }
    cout << "Queue is empty" << endl;
```

```
    }

    T top()
    {
        if (start)
        {
            return start->data;
        }
        cout << "Queue is empty" << endl;
        return T();
    }

    void ProvideService()
    {
        if (!start)
            return;
        cout << "Hello Mr, " << start->data << endl;
        cout << "Enter name of book you want to borrow or return: " <<
endl;
        string str;
        getline(cin, str);
        cout << "Patron serviced" << endl;
        pop();
    }
    int getSize() { return size; }
};
int main()
```

```
{
    QueueLL<string> Library;
    // adding new patron
    Library.push("Ahmed");
    Library.push("Ali");
    Library.push("Hafeez");
    Library.push("Mubeen");
    Library.ProvideService();
    cout<<Library.getSize()<<endl;
    Library.ProvideService();
    cout<<Library.getSize()<<endl;
    return 0;
};
```

Output:

```
b/06_lab/solution$ cd "/media/mufeez/work1/FAST_KHI_Semes
ter_3/DS_lab/06_lab/solution/" && g++ 9_task.cpp -o 9_tas
k && "/media/mufeez/work1/FAST_KHI_Semester_3/DS_lab/06_l
ab/solution/"9_task
Hello Mr, Ahmed
Enter name of book you want to borrow or return:
nothing
Patron serviced
3
Hello Mr, Ali
Enter name of book you want to borrow or return:
something
Patron serviced
2
mufeez@mine:/media/mufeez/work1/FAST_KHI_Semester_3/DS_la
```