Answer Script

Question No. 01

1. Implement a template based Queue using a dynamic array which supports the enqueue, dequeue and front operations. **15**

Answer No. 01

Code:

```
#include<bits/stdc++.h>
using namespace std;

template <class T>
class Queue
{
public:
    T *a;
    int array_cap;
    int l, r;
    int sz;

    Queue()
    {
        a = new T[1];
        array_cap = 1;
        l = 0;
        r = -1;
        sz = 0;
    }

    void remove_circular()
    {
        if(l>r)
        {
            T *temp = new T[array_cap];
            int index = 0;
            for(int i=1; i<array_cap; i++)
            {
                temp[index] = a[i];
                index++;
            }
            for(int i=0; i<=r; i++)
            {
                temp[index] = a[i];
                index++;
        }

        swap(a, temp);
        l = 0;
        r = array_cap - 1;
        delete [] temp;
    }
}

void increase_size()
{
    remove_circular();
    T *temp = new T[array_cap*2];
    for(int i=0; i<array_cap; i++)
    {
        temp[i] = a[i];
    }
    swap(a, temp);
    array_cap = array_cap * 2;
    delete [] temp;
}

void enqueue(T value)
{
    if(sz == array_cap)
    {
        increase_size();
    }
    r++;
    if(r == array_cap)
    {
        r = 0;
    }
    a[r] = value;
    sz++;
}

void dequeue()
{
    if(sz == 0)
    {
        cout<<"Queue is empty!\n";
        return;
    }
    l++;
    if(l == array_cap)
    {
            l = 0;
        }
        sz--;
    }

    T front()
    {
        if(sz == 0)
        {
            cout<<"Queue is empty!\n";
            return -1;
        }
        return a[l];
    }
};

int main()
{
    Queue <int> q;

    q.enqueue(5);
    q.enqueue(6);
    q.enqueue(7);

    cout<<q.front()<<"\n";
    q.dequeue();
    cout<<q.front()<<"\n";
    q.dequeue();
    cout<<q.front()<<"\n";
    q.dequeue();
    cout<<q.front()<<"\n";

    return 0;
}
```

Question No. 02	
2. Implement Template based Stack using a singly linked-list.	**15**
Answer No. 02	

Code:

```
#include<bits/stdc++.h>
using namespace std;

template <class T>
class Node
{
public:
    T data;
    Node* next;
};

template <class T>
class SinglyLinkedList
{
public:
    Node<T>* head;
    int sz;

    SinglyLinkedList()
    {
        head = NULL;
        sz = 0;
    }

    Node<T>* CreateNewNode(T value)
    {
        Node<T> *newNode = new Node<T>;
        newNode->data = value;
        newNode->next = NULL;
        return newNode;
    }

    void InsertAtHead(T value)
    {
        sz++;
        Node<T>* a = CreateNewNode(value);
        if(head == NULL)
        {
            head = a;
            return;
        }
        a->next = head;
        head = a;
    }

    void DeleteAtHead()
    {
        if(head == NULL)
        {
            return;
        }
        sz--;
        Node<T>* a = head;
        head = a->next;
        delete a;
    }

    T getSize()
    {
        return sz;
    }

    void Traverse()
    {
        Node<T>* a = head;
        while(a != NULL)
        {
            cout<<a->data<<" ";
            a = a->next;
        }
        cout<<"\n";
    }

};

template <class T>
class Stack
{
public:
    SinglyLinkedList<T> dl;

    Stack()
    {

    }
    T top()
    {
        if(dl.getSize() == 0)
        {
            cout<<"Stack is empty!\n";
            return -1;
        }
        return dl.head->data;
    }

    void push(T val)
    {
        dl.InsertAtHead(val);
    }

    void pop()
    {
        if(dl.getSize() == 0)
        {
            cout<<"Stack is empty!\n";
            return;
        }
        dl.DeleteAtHead();
    }
};

int main()
{
    Stack<int> st;

    st.push(3);
    cout<<st.top()<<"\n";
    st.push(4);
    cout<<st.top()<<"\n";
    st.push(5);
    cout<<st.top()<<"\n";

    st.pop();
    cout<<st.top()<<"\n";
    st.pop();
    cout<<st.top()<<"\n";
    st.pop();
    cout<<st.top()<<"\n";
    return 0;
}
```

Question No. 03

3. Write a program to convert an infix expression to a postfix expression. The expression will contain the following characters [a-z , + , - , * , / , (,)]. **15**

Sample Input	Sample Output
a+(b+c)*d-e	abc+d*+e-
(a+b)*(c+d)	ab+cd+*

Answer No. 03

Code:

```
#include<bits/stdc++.h>
using namespace std;

int prec(char ch)
{
    if (ch == '+' || ch == '-')
        return 1;
    else if (ch == '/' || ch == '*')
        return 2;
    else
        return -1;
}

string infixToPostfix(string s)
{
    stack<char> st;
    string ans = "";

    for (int i = 0; i < s.length(); i++)
    {
        char ch = s[i];

        if ((ch >= 'a' && ch <= 'z'))
            ans += ch;
        else if (ch == '(')
            st.push('(');
        else if (ch == ')')
        {
            while (st.top() != '(')
            {
                ans += st.top();
                st.pop();
            }
            st.pop();
        }
        else
        {
            while (!st.empty() && prec(s[i]) <= prec(st.top()))
            {
                ans += st.top();
                st.pop();
            }
            st.push(ch);
        }
    }

    while (!st.empty())
    {
        ans += st.top();
        st.pop();
    }

    cout<<ans<<"\n";
}

int main()
{
    string s;
    cin >> s;  /// a+(b+c)*d-e  (a+b)*(c+d)
    infixToPostfix(s);
    return 0;
}
```

Question No. 04

4. Evaluate it using stack. All the numbers are single digit numbers in the input so you don't have to worry about multi digit numbers.

15

Sample Input	Sample Output
4+(5+6)*8-1	91
(2+4)*(5+6)	66

Congratulations you just built a mini calculator if you solved it correctly.

Answer No. 04

Code:

```
#include<bits/stdc++.h>
using namespace std;

int prec(char ch)
{
    if (ch == '+' || ch == '-')
        return 1;
    else if (ch == '/' || ch == '*')
        return 2;
    else
        return -1;
}

string infixToPostfix(string s)
{
    stack<char> st;
    string ans = "";

    for (int i = 0; i < s.length(); i++)
    {
        char ch = s[i];

        if ((ch >= '0' && ch <= '9'))
            ans += ch;
        else if (ch == '(')
            st.push('(');
        else if (ch == ')')
        {
            while (st.top() != '(')
            {
                ans += st.top();
                st.pop();
            }
            st.pop();
        }
        else
        {
            while (!st.empty() && prec(s[i]) <= prec(st.top()))
            {
                ans += st.top();
                st.pop();
            }
            st.push(ch);
        }
    }

    while (!st.empty())
    {
        ans += st.top();
        st.pop();
    }

    return ans;
    //cout<<ans<<"\n";
}

void calculate(string s)
{
    stack<int> st;
    int sz = s.size();
    int c = 0;
    for (int i = 0; i <sz; i++)
    {
        if (isdigit(s[i]))
            st.push(s[i] - '0');
        else
        {
            int num1 = st.top();
            st.pop();
            int num2 = st.top();
            st.pop();
            if( s[i] == '+')
                st.push(num1 + num2);
            else if( s[i] == '-')
            {
                st.push(num2-num1);
            }
            else if( s[i] == '*')
                st.push(num1 * num2);
            else if( s[i] == '/')
            {
                st.push(num2/num1);
            }
            else
            {
                cout<<"Invalid Expression";
                c = 1;
                break;
            }
        }
    }
    if(c==0)
        cout<<st.top()<<"\n";
}

int main()
{
    string s;
    cin >> s;
    string ans = infixToPostfix(s);
    //cout<<ans<<"\n";

    calculate(ans);

    return 0;
}
```

Question No. 05

5. Implement Template based Deque using a doubly linked-list which supports push_front, push_back, pop_back, pop_front, front, back operations. **15**

Answer No. 05

Code:

```
#include<bits/stdc++.h>
using namespace std;

template <class T>
class Node{
public:
    T data;
    Node* prev;
    Node* next;
};

template <class T>
class Deque{
public:
    Node<T>* head;
    Node<T>* tail;
    int sz;

    Deque()
    {
        head = NULL;
        tail = NULL;
        sz = 0;
    }

    Node<T>* CreateNewNode(T value)
    {
        Node<T>* newNode = new Node<T>;
        newNode->data = value;
        newNode->next = NULL;
        newNode->prev = NULL;
        return newNode;
    }

    ///Pushback
    void push_back(T value)
    {
```

```
    Node<T>* newNode = CreateNewNode(value);
    if(sz == 0)
    {
        head = newNode;
        tail = newNode;
        sz++;
        return;
    }
    tail->next = newNode;
    newNode->prev = tail;
    tail = newNode;
    sz++;
}

///PushFront
void push_front(T value)
{
    Node<T>* newNode = CreateNewNode(value);
    if(sz == 0)
    {
        head = newNode;
        tail = newNode;
        sz++;
        return;
    }
    head->prev = newNode;
    newNode->next = head;
    head = newNode;
    sz++;
}

///Pop back
void pop_back()
{
    if(sz == 0)
    {
        cout<<"Deque is empty!\n";
        return;
    }
    if(sz == 1)
    {
        delete tail;
        head = NULL;
        tail = NULL;
        return;
    }
    Node<T>* a = tail;
    tail = tail->prev;
    delete a;
    tail->next = NULL;
}

///Pop front
void pop_front()
{
    if(sz == 0)
    {
        cout<<"Deque is empty!\n";
        return;
    }
    if(sz == 1)
    {
        delete tail;
        head = NULL;
        tail = NULL;
        sz--;
        return;
    }

    Node<T>* a = head;
    head = head->next;
    delete a;
    head->prev = NULL;
    sz--;
}

///Front
T front()
{
    if(sz == 0)
    {
        cout<<"Dequq is empty!\n";
        return -1;
    }
    return head->data;
}

///back
T back()
{
    if(sz == 0)
    {
        cout<<"Deque is empty!\n";
        return -1;
        }
        return tail->data;
    }
};

int main()
{
    Deque<int> d;
    d.push_back(5);
    d.push_back(10);
    d.push_back(15);

    cout<<"Back: "<<d.back()<<" Front: "<<d.front();

    d.push_front(20);
    cout<<"\nBack: "<<d.back()<<" Front: "<<d.front();

    d.pop_front();
    cout<<"\nBack: "<<d.back()<<" Front: "<<d.front();

    d.pop_back();
    cout<<"\nBack: "<<d.back()<<" Front: "<<d.front();

    return 0;
}
```

Question No. 06

6. Given a string, check if it's a palindrome using a Deque.

Sample Input	Sample Output
abcba	Yes
abcca	No

Hint: Check the first and last character. If they are equal then pop them and continue this process until the string becomes empty. **15**

Answer No. 06

Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   string s;
   cin>>s;

   deque<char> dq;

   int i = 0;
   while(s[i] != '\0')
   {
      if((s[i]>='a' && s[i]<='z') || (s[i]>='A' && s[i]<='Z'))
      {
         dq.push_back(s[i]);
      }
      i++;
   }

   int flag = 0;
   while(dq.size() > 1)
   {
      if(dq.front() != dq.back())
      {
         flag = 1;
         break;
      }
      dq.pop_front();
      dq.pop_back();
   }

   if(flag == 0)
```

```
        cout<<"Yes\n";
    else
        cout<<"No\n";

    return 0;
}
```

Question No. 07

7. Write a function **void deleteValue(list<int> & l , int value)’** -> This function will delete the first occurrence of the element that is equal to the input **value** from the stl list.
   Sample Input: STL list containing [7, 3, 8, 4, 5, 4], value : 4
   Sample Output: STL list containing [7, 3, 8, 5, 4] **10**

Answer No. 07

Code:

```
#include<bits/stdc++.h>
using namespace std;

void print(list<int>&l)
{
    auto a = l.begin();

    while(a != l.end())
    {
        cout<<*a<<" ";
        a++;
    }

    cout<<"\n";
}

void deleteValue(list<int> &l, int value)
{
    auto it = l.begin();

    int flag = 0;
    while(it != l.end())
    {
        if(*it == value)
        {
            l.erase(it);
            flag = 1;
            break;
        }
        it++;
    }
    if(flag == 0)
        cout<<"Value not found in the STL list!\n";
}
```

```
int main()
{
    list<int> l;

    ///Push Back
    l.push_back(7);
    l.push_back(3);
    l.push_back(8);
    l.push_back(4);
    l.push_back(5);
    l.push_back(4);

    print(l);

    ///Delete
    deleteValue(l, 4);
    print(l);

    return 0;
}
```