# 23p-0685 Raqeeb Task #5
# Question #1:

```
#include<iostream>

using namespace std;

class Nodes
{
   public:
   string info;
   Nodes *next; //It will point to the object of the class made in the main function

   Nodes(string value)
   {
      info= value;
      next=NULL;

   }

};

class Linked_List
{
   private:
      Nodes *top;
      int length;

   public:
      Linked_List()
      {
         length=0;
         top =NULL;

      }

   void push(string value)
   {
```

```
        Nodes *n= new Nodes(value);
        Nodes *curr=top;
        if(top==NULL)
        {

            top=n;

        }
        else if(top!=NULL)
        {
            n->next=curr;
            top=n;
        }

        length++;
    }

//pop function to remove the last value from the stack
    string pop()
    {
        if(length==0)
        {
            return " ";
        }
        Nodes *curr=top;
        top=curr->next;
        length-=1;
        string value;
        value=curr->info;
        delete curr;
        return value;

    }
    string peek()
    {
        Nodes *cur=top;
        return cur->info;

    }
```

```
//To clear the whole stack
    void clear()
    {
        int size=length;
        for(int i=0;i<=size;i++)
        {
            pop();
        }

    }

//checking whether the stack is empty or not
    bool is_empty()
    {
        if(length==0)
        {
            return true;
        }
        else
            return false;
    }



};



int main(void)
{
    Linked_List Forward_stack,Backward_stack;

    //Variables for different purpose
    int user_input;
    string new_url;
    bool is_back_visit=false;
    bool is_forward_visit=false;
    int Num_forward=0;
    int Num_backward=-1;
    int page_num=0;
    string temp;
```

```
    string temp2;

//infinite loop for user control
    while(1)
    {
       //menu
    cout<<"Web page Browing Operations:\n1)Visit new page\n2)Press Back Button\n3)Press forward button\n4)Display the page number you are in\n5)Exit\nuserinput:";

        cin>>user_input;
        if(user_input==1)
        {

           cout<<"Enter the url of page:";
           cin>>new_url;
           temp=new_url;
           Backward_stack.push(new_url);
           //to remove the whole stack values
           Forward_stack.clear();
           Num_backward+=1;
           if(Num_backward>0)
               is_back_visit=true;
           Num_forward=0;
           //condition, so user cannot move forward
           is_forward_visit=false;
           cout<<"\nVisited to new Page:"<<temp<<endl<<endl;
           ++page_num;
        }
        else if(user_input==2 && Backward_stack.is_empty() != true)
        {
           if(page_num>1)
           {
              Num_backward-=1;
              Num_forward+=1;
              temp2=Backward_stack.pop();
              //to get the current page we are in
              temp= Backward_stack.peek();
              cout<<"\nBack to previous page:"<<temp<<endl;
              if(temp2!=" ")Forward_stack.push(temp2);
```

```
        page_num-=1;
        if(Num_backward==0)
           is_back_visit=false;

        is_forward_visit=true;
        }
        else if(page_num==1)
        {
           cout<<"\nNo previous page to move back!!!\n";
        }

}
else if(user_input==3 && Forward_stack.is_empty() !=true)
{
   Num_backward+=1;
   if(Num_forward!=0)
      Num_forward-=1;
   temp=Forward_stack.peek();
   temp2=Forward_stack.pop();

   //if stack is not empty
   if(temp2!=" ")
      Backward_stack.push(temp2);
   //if we reached the end page the no more forward movements
   if(Num_forward==0)
      is_forward_visit=false;
   is_back_visit=true;
   cout<<"\nMoved to forward page:"<<temp<<endl;
   ++page_num;

}
//Display the current page
else if(user_input==4)
{
   if(page_num!=0)
   cout<<"\n\nYou are currently in Page:"<<temp<<endl<<endl;
   else
   cout<<"\nFirst visit the new page first!!!\n";
}
//if user directly try to move forward without visiting new page
```

```
        else if(user_input==3 && is_forward_visit==false)
        {
            cout<<"\nNo forward page to move forward!!!\n";
        }
        //if user directly try to move backward without visiting new page
        else if (user_input==2 && is_back_visit==false)
        {
            cout<<"\nNo page available to move backward!!!!"<<endl;
        }
        else if(user_input==5)
        {
            break;
        }


    }


    return 0;
}
```

**OUTPUT:**

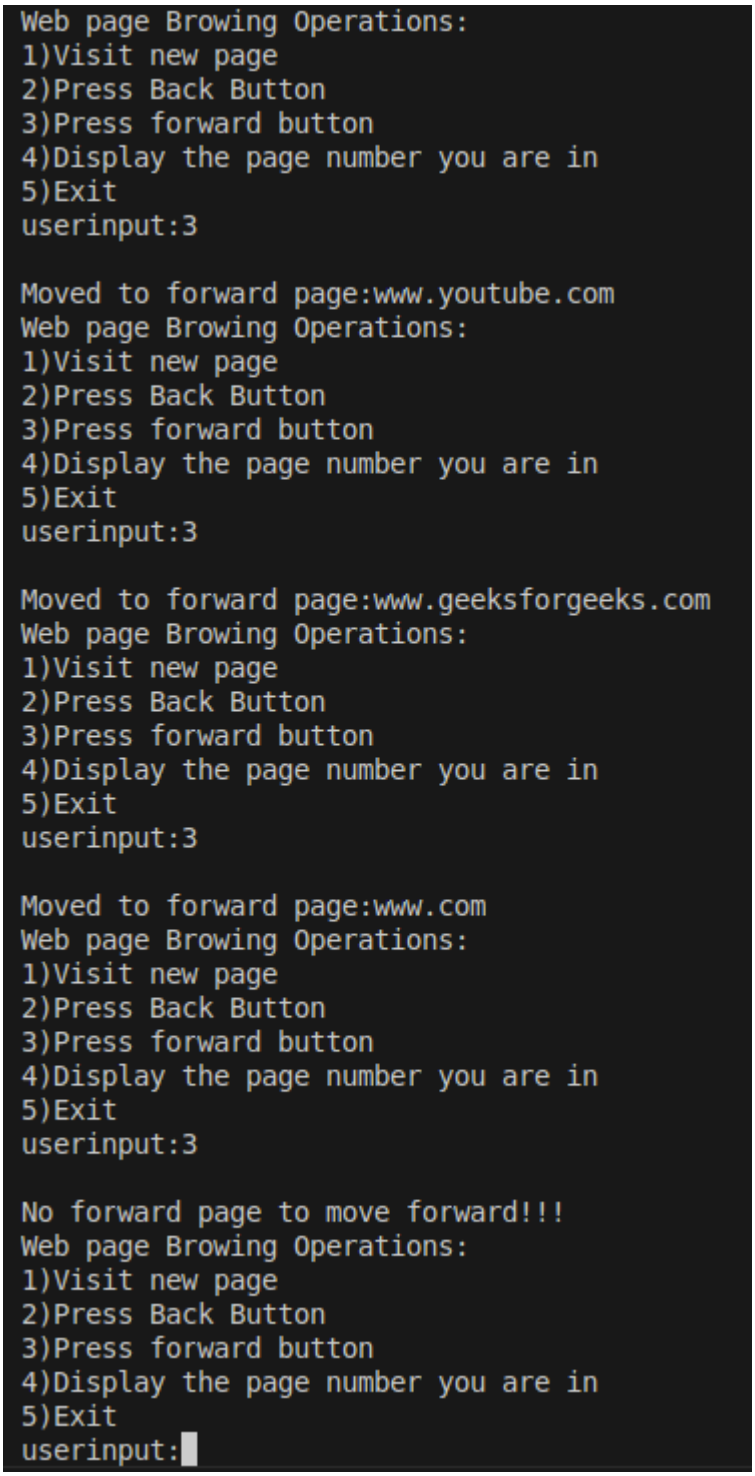
**1)**

```
Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:1
Enter the url of page:www.google.com

Visited to new Page:www.google.com

Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:1
Enter the url of page:www.youtube.com

Visited to new Page:www.youtube.com

Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:1
Enter the url of page:www.geeksforgeeks.com

Visited to new Page:www.geeksforgeeks.com

Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:1
Enter the url of page:www.com

Visited to new Page:www.com

Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:
```

**2)**

```
Visited to new Page:www.com

Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:2

Back to previous page:www.geeksforgeeks.com
Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:2

Back to previous page:www.youtube.com
Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:2

Back to previous page:www.google.com
Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:2

No previous page to move back!!!
Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:2

No previous page to move back!!!
Web page Browing Operations:
1)Visit new page
2)Press Back Button
3)Press forward button
4)Display the page number you are in
5)Exit
userinput:
```

**3)**




# Question #2:

#include<iostream>
using namespace std;

```
class Nodes
{
   public:
   char info;
   Nodes *next;

   Nodes(int value)
   {
      info= value;
      next=NULL;

   }

};

class Linked_List
{
   private:
      Nodes *top;
      int length;

   public:
      Linked_List()
      {
         length=0;
         top =NULL;


      }

   void push(int value)
   {


      Nodes *n= new Nodes(value);
      Nodes *curr=top;
      if(top==NULL)
      {
```

```
        top=n;

    }
    else if(top!=NULL)
    {
        n->next=curr;
        top=n;
    }

    length++;
}

char pop()
{
    if(length==0)
    {
        return 'N';
    }
    Nodes *curr=top;
    top=curr->next;
    length-=1;
    return curr->info;
    delete curr;

}
void display()
{
    if(length!=0)
{    Nodes *curr=top;
    for(int i=0;i<length;i++)
    {
        cout<<(curr)->info<<endl;
        curr=curr->next;
    }
    }
    else
    {
        cout<<"\nThe linkedlist stack is empty"<<endl;
    }
```

```
    }

bool Bracket_validator(string value)
    {
        bool t=true,f=false;
    char
B1_open='(',B1_close=')',B2_open='[',B2_close=']',B3_open='{',B3_close
='}';

        for(int i=0;i<value.size();i++)
        {
            if(i==0)
           {
            if(value[i] != B1_close && value[i] != B2_close && value[i] !
=B3_close)
                {
                    if(value[i]==B1_open || value[i]==B2_open ||
value[i]==B3_open)
                        push(value[i]);
                }
            else
            {
                return f;
            }
        }
        else
        {
            if(value[i] == B1_close || value[i] == B2_close || value[i]
==B3_close)
            {
                char poped_value=pop();
                if(value[i]==B1_close && poped_value==B1_open ||
value[i]==B2_close && poped_value==B2_open || value[i]==B3_close
&& poped_value==B3_open)
                {
                    continue;
                }
                else
                {
```

```
                return f;
            }
        }
        else
        {
            if(value[i]==B1_open || value[i]==B2_open ||
value[i]==B3_open)
                push(value[i]);
        }

    }

  }
   if(length==0)
     return t;
  else
     return f;
}

};



int main(void)
{
 Linked_List LD_1;
  while(1)
  {
  cout<<"Enter the brackets to check its validity:";
  string user_input;
  cin>>user_input;
  if(LD_1.Bracket_validator(user_input)==1)
  {
     cout<<"true"<<endl;
     cout<<"\nWant to end(1/0):";
     cin>>user_input;
     if(user_input=="1")
        break;
  }
```

```
    else
    {
        cout<<"false"<<endl;
        cout<<"\nWant to end(1/0):";
        cin>>user_input;
        if(user_input=="1")
            break;
    }
    }

    return 0;
}
```

**OUTPUT:**