1.1Simple_Stack_13

Code:-

```
#include<iostream>
using namespace std;
class nick
{
public:
int top,ele,choice,num,size,s[10];
public:
void push()
{
cout<<"push operation is selected\n";
if(top==size-1)
{
cout<<"stack exausted.....\n";
}
else
{
cout<<"enter the element you want to push into the stack\n";
cin>>num;
++top;
s[top]=num;
cout<<"the number is inserted into the stack:-\n";
}
}

void pop()
{
cout<<"you have selected pop operation\n";
if(top==-1)
{
```

Stack operations

```
cout<<"no element present in the stack\n";
}
else
{
ele=s[top];
top--;
}
}

void dis()
{

cout<<"display option is selected\n";
if(top==-1)
{
cout<<"no element is present in array\n";
}
else
{
for(int i=top;i>-1;i--)
{
cout<<i<<" : "<<s[i]<<"\n";
}
}
}

void get()
{
top=-1;
cout<<"enter the size of the stack:-\n";
cin>>size;

Stack operations
```

```
int s[size];
do
{
cout<<"enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-\n";
cin>>choice;

switch(choice)
{
case 1:
push();
break;

case 2:
pop();
break;

case 3:
dis();
break;

case 4:
break;

default:
cout<<"invalid choice ## enter right choice.....\n";
}
}
while(choice!=4);
}
};
```

Stack operations

```
int main()
{
nick o;
o.get();
}
```

Output:-

```
enter the size of the stack:-
3
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
1
push operation is selected
enter the element you want to push into the stack
1
the number is inserted into the stack:-
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
1
push operation is selected
enter the element you want to push into the stack
2
the number is inserted into the stack:-
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
1
push operation is selected
enter the element you want to push into the stack
3
the number is inserted into the stack:-
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
1
push operation is selected
stack exausted........
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
3
display option is selected
2 : 3
1 : 2
0 : 1
```

```
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
2
you have selected pop operation
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
2
you have selected pop operation
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
2
you have selected pop operation
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
2
you have selected pop operation
no element present in the stack
enter the option you want to select:-(1-push 2-pop 3-display 4-exit):-
```

1.2Balancing_Paranthesis_Stack_13

Code:-

```
#include<iostream>
#include<string>
using namespace std;
class nick
{
public:
int size,top,i;
string str;
char *a;
public:
void push(char c)
{
if(top==size-1);
else
{
a[++top]=c;
}
}
void pop()
{
```

Stack operations

```
if(top==-1);
else
{
top--;
}
}
void display()
{
if(top==-1)
{
cout << "empty array";
}
else
{
cout << "elements in array: \n";
for(int i=top;i>=0;i--)
{
cout<<a[i]<<"\n";
}
}
}
void mytech()
{
top=-1;
cout << "Enter a string: ";
cin >> str;
int b=str.length();
a=new char[b];
for(i=0;i<b; i++)
{
if(top==-1 && str.at(i)==')')
```

Stack operations

```
{
cout << "Invalid Input.";
break;
}
else
{
if(str.at(i)=='(')
{
push(str.at(i));
}
else if(str.at(i)==')')
{
pop();
}
}
}
if(i==b)
{
if (top==-1)
{
cout << "equal Paranthesis.\n";
}
else
{
cout << "Paranthesis not equal.\n";
}
}
}
};
int main()
{
```

Stack operations

```
nick o;
o.mytech();
}
```

Output:-

```
Enter a string: (a+b)*(a*b*c)
equal Paranthesis.

--------------------------------
```

```
Enter a string: (a+b)(a*b*c
Paranthesis not equal.

--------------------------------
```

1.3Infix_To_Postfix_Stack_13

Code:-

```
#include<iostream>
using namespace std;
class nick
{
char *s;
string input,output;
int top;
public:
mytech()
{
top = -1;
cout<<"Enter equation:-";
cin >>input;
input+=')';
s=new char[input.length()];
s[++top] = '(';
output="";
for(int i=0;i<input.length();i++)
```

Stack operations

```
{
if(input.at(i)=='(')
{
push(input.at(i));
}
else if(process(input.at(i))==true)
{
if(s[top]!='(' && (prior(input.at(i))<prior(s[top])))
{
output+=pop();
push(input.at(i));
}
else
{
push(input.at(i));
}
}
else if(input.at(i)==')')
{
while(s[top]!='(')
{
output += pop();
}
pop();
}
else
{
output+=input.at(i);
}
}
cout << output << endl;
```

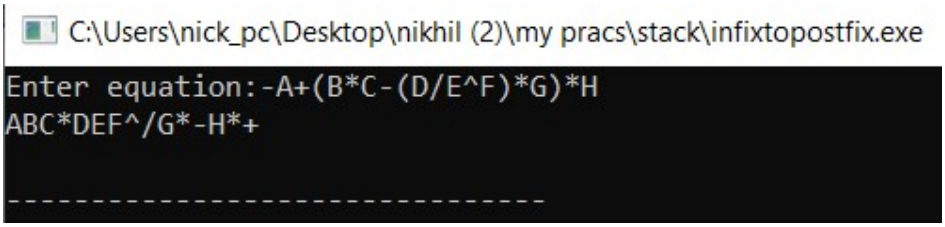
Stack operations

```
}
bool process(char c)
{
if ((c=='+')||(c=='-')||(c=='*')||(c=='/')||(c=='^'))
{
return true;
}
else
{
return false;
}
}
int prior(char op)
{
if(op=='^')
{
return 4;
}
else if(op=='/')
{
return 3;
}
else if(op=='*')
{
return 2;
}
else if(op=='+')
{
return 1;
}
else if(op=='-')
```

Stack operations

```
{
return 0;
}
}
void push(char c)
{
s[++top] = c;
}
char pop()
{
return s[top--];
}
};
int main()
{
nick o;
o.mytech();
}
```

Output:-



1.5Stack_Link_List_13

Code:-

```
#include<iostream>
#include<malloc.h>
using namespace std;
struct node{
   int data;
```

Stack operations

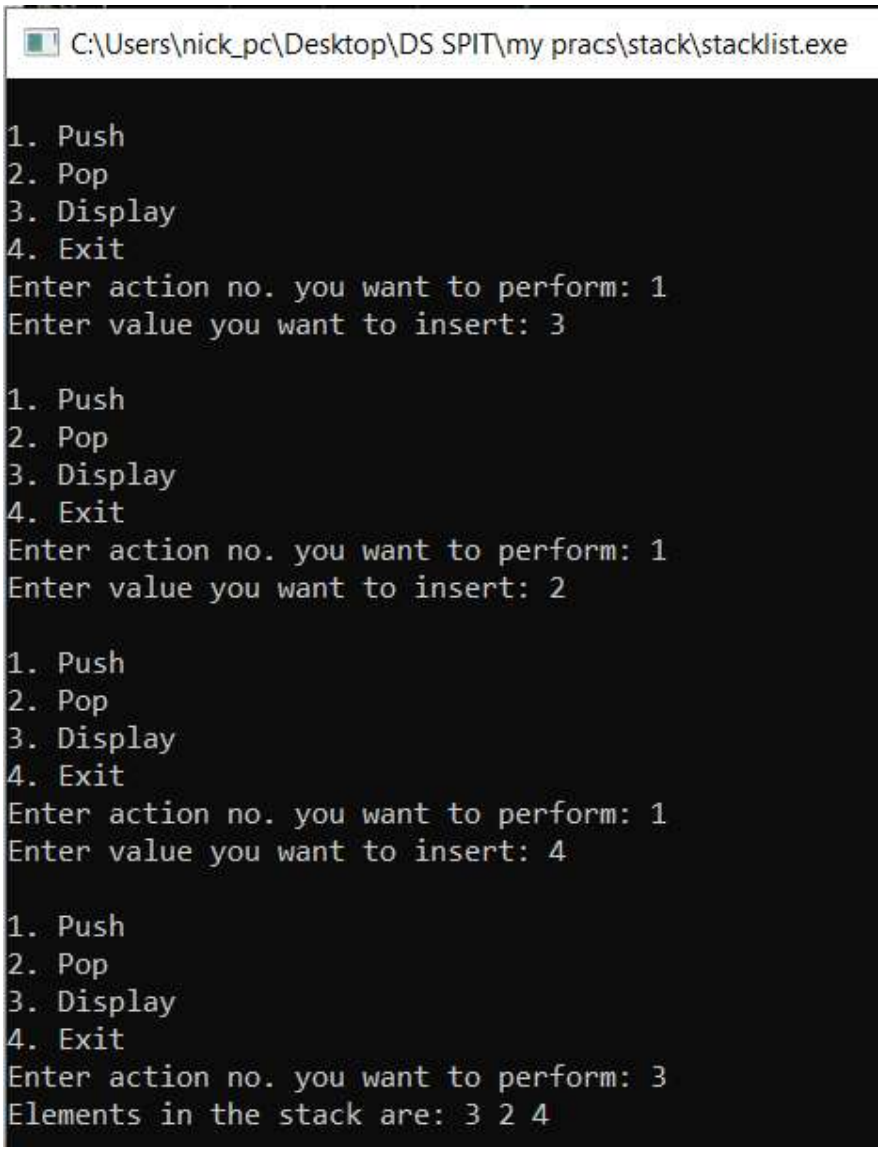
```
   struct node *next;
}*list = NULL, *top = NULL, *p, *q;

class nick{
   int action, value;
   string str;
   public:
      void mytech(){
         do{
            cout << "\n1. Push\n2. Pop\n3. Display\n4. Exit\nEnter action no. you want to perform: ";
            cin >> action;
            switch (action)
            {
               case 1:
                  push();
                  break;
               case 2:
                  pop();
                  break;
               case 3:
                  display();
                  break;
               case 4:
                  break;
               default:
                  cout << "Invalid input.";
            }
         }while(action != 4);
      }

      void push(){
```

```
        cout << "Enter value you want to insert: ";
        cin >> value;
        p = (struct node*)malloc(sizeof(node));
        p->data = value;
        p->next = NULL;
        top = p;
        if(list==NULL)
            list = p;
        else{
            q = list;
            while(q->next!=NULL)
                q = q->next;
            q->next = p;
        }
    }

    void pop(){
        if(list==NULL)
            cout << "Underflow.\n";
        else{
            cout << top->data << " has been popped." << endl;
            q = list;
            if(q->next==NULL)
                list = top = NULL;
            else{
                while(q->next != top && q->next!=NULL)
                    q = q->next;
                free(top);
                top = q;
                q->next=NULL;
            }
```

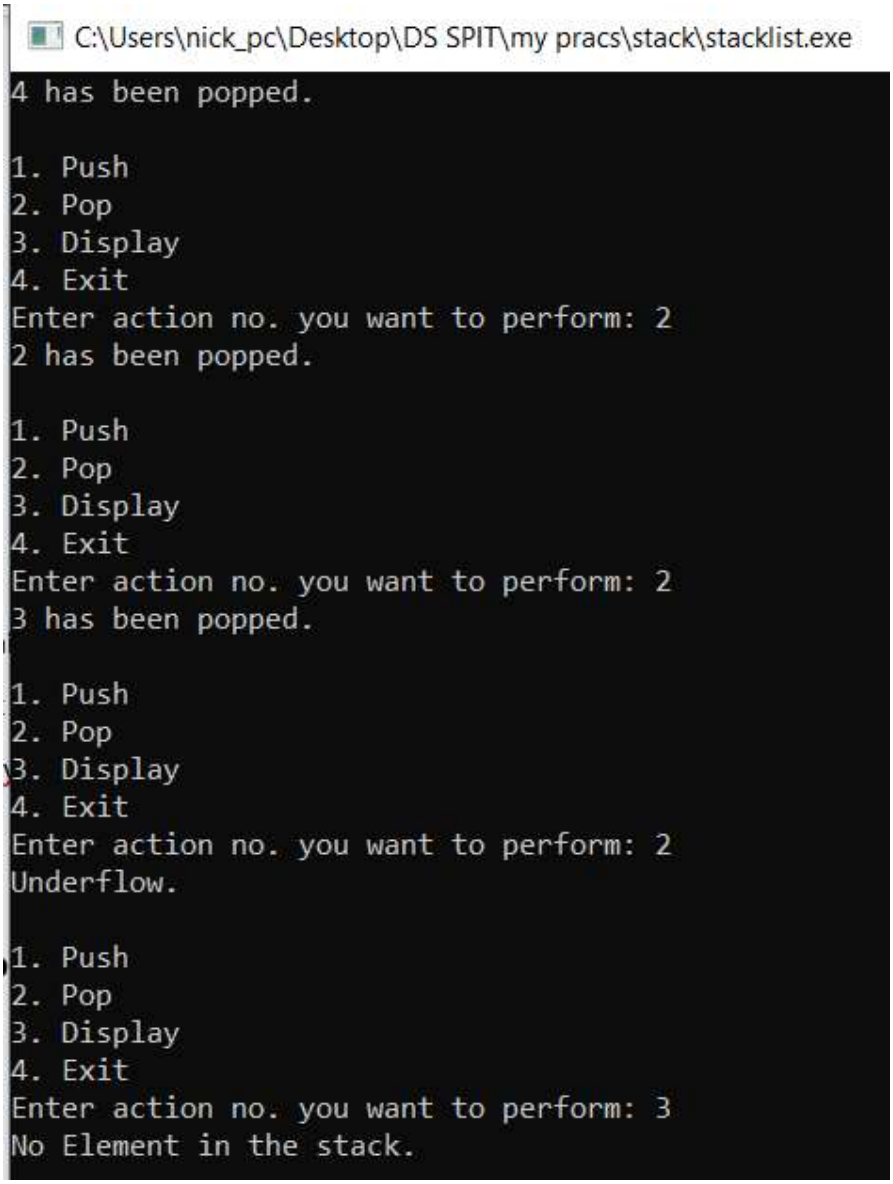
Stack operations

```
        }
    }

    void display(){
    if(list==NULL)
        cout << "No Element in the stack." << endl;
    else{
        p = list;
        cout << "Elements in the stack are: ";
        while(p!=NULL){
            cout << p->data << " ";
            p = p->next;
        }
        cout << endl;
    }
  }
};

int main(){
  nick o;
  o.mytech();

}
```

Output:-



Stack operations



Stack operations