Assignment 2 Solutions - Yash Awasthi 102109029 T5

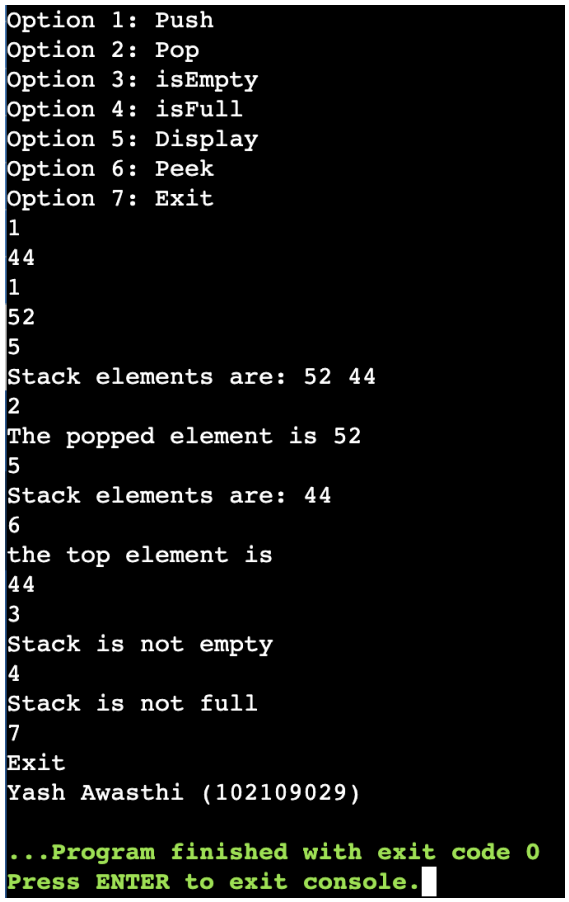
1.Develop a menu driven program demonstrating the following operations on a Stack:
push(), pop(), isEmpty(), isFull(), display(), and peek().

```
#include <iostream>
using namespace std;
int stack[100], n=100, top=-1;

void isfull(){
   if(top>=n-1)
      cout<<"Stack is Full"<<endl;
   else
      cout<<"Stack is not full"<<endl;
}
void push(int val){
   top++;
   stack[top]=val;
}
void isempty(){
   if(top<=-1)
      cout<<"Stack is empty"<<endl;
   else
      cout<<"Stack is not empty"<<endl;
}
void pop(){
   cout<<"The popped element is "<< stack[top]<<endl;
   top--;
}
void peek(){
   cout<<"the top element is "<<endl;
   cout<<stack[top]<<endl;
}
void display(){
   if(top>=0){
      cout<<"Stack elements are: ";
      for(int i=top; i>=0; i--){
         cout<<stack[i]<<" ";
      }
   }
   cout<<endl;
}
```

```
int main()
{
    int n,x;
    cout<<"Option 1: Push"<<endl;
    cout<<"Option 2: Pop"<<endl;
    cout<<"Option 3: isEmpty"<<endl;
    cout<<"Option 4: isFull"<<endl;
    cout<<"Option 5: Display"<<endl;
    cout<<"Option 6: Peek"<<endl;
    cout<<"Option 7: Exit"<<endl;
    do{
        cin>>n;
        switch(n){
            case 1:
                cin>>x;
                push(x);
                break;
            case 2:
                pop();
                break;
            case 3:
                isempty();
                break;
            case 4:
                isfull();
                break;
            case 5:
                display();
                break;
            case 6:
                peek();
                break;
            case 7:
                cout<<"Exit"<<endl;
                break;
            default:{
                cout<<"Invalid Choice"<<endl;
            }
        }
    }while(n!=7);
    return 0;
}
```

Output:


Option 1: Push
Option 2: Pop
Option 3: isEmpty
Option 4: isFull
Option 5: Display
Option 6: Peek
Option 7: Exit
1
44
1
52
5
Stack elements are: 52 44
2
The popped element is 52
5
Stack elements are: 44
6
the top element is
44
3
Stack is not empty
4
Stack is not full
7
Exit
Yash Awasthi (102109029)
...Program finished with exit code 0
Press ENTER to exit console.

2. Given a String, Reverse it using STACK. For example, “data structure” should be output as “erutcurtsatad.”

```
#include<iostream>
#include<string.h>
using namespace std;
int stack[100], n=100, top=-1;

char push(char val){
   if(top>=n-1)
      cout<<"Stack is full"<<endl;
   else
      top++;
      stack[top]=val;
}
char pop(){
   if(top<=-1){
      cout<<"Stack is empty"<<endl;
   }else{
      return stack[top--];
   }
}

int main(){
   string s;
   getline(cin,s);
   for(int i=0;i<s.length();i++){
      push(s[i]);
   }
   for(int i=0;i<s.length();i++){
      s[i]=pop();
   }
   cout<<"Reverse String is "<<s;
   cout<<endl<<"Yash Awasthi (102109029)";
   return 0;
}
```

```
datastructure
Reverse String is erutcurtsatad
Yash Awasthi (102109029)

...Program finished with exit code 0
Press ENTER to exit console.
```

3. Write a program that checks if an expression has balanced parentheses.

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

bool areBracketsBalanced(string expr)
{
	stack<char> temp;
	for (int i = 0; i < expr.length(); i++) {
		if (temp.empty()) {
			temp.push(expr[i]);
		}
		else if ((temp.top() == '(' && expr[i] == ')')
				|| (temp.top() == '{' && expr[i] == '}')
				|| (temp.top() == '[' && expr[i] == ']')) {
			temp.pop();
		}
		else {
			temp.push(expr[i]);
		}
	}
	if (temp.empty()) {
		return true;
	}
	return false;
}

int main()
{
	string expr = "{()}[]";
	if (areBracketsBalanced(expr))
		cout << "Balanced";
	else
		cout << "Not Balanced";
	cout<<endl<<"Yash Awasthi (102109029)";
	return 0;
}
```

```
Balanced
Yash Awasthi (102109029)

...Program finished with exit code 0
Press ENTER to exit console.
```

4. Write a program to convert an Infix expression into a Postfix expression.

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

int prec(char c)
{
	if (c == '^')
		return 3;
	else if (c == '/' || c == '*')
		return 2;
	else if (c == '+' || c == '-')
		return 1;
	else
		return -1;
}

void infixToPostfix(string s)
{
	stack<char> st;
	string result;
	for (int i = 0; i < s.length(); i++) {
		char c = s[i];
		if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')
			|| (c >= '0' && c <= '9'))
			result += c;
		else if (c == '(')
			st.push('(');
		else if (c == ')') {
			while (st.top() != '(') {
				result += st.top();
				st.pop();
			}
			st.pop();
		}
		else {
			while (!st.empty()
				&& prec(s[i]) <= prec(st.top())) {
				if (c == '^' && st.top() != '^')
					break;
				else {
					result += st.top();
					st.pop();
				}
			}
```

```
                st.push(c);
            }
        }
        while (!st.empty()) {
            result += st.top();
            st.pop();
        }
        cout << result << endl;
}

int main()
{
        string exp = "a+b*(c^d-e)^(f+g*h)-i";
        infixToPostfix(exp);
        cout<<endl<<"Yash Awasthi (102109029)";
        return 0;
}
```

```
abcd^e-fgh*+^*+i-

Yash Awasthi (102109029)

...Program finished with exit code 0
Press ENTER to exit console.
```

5. Write a program for the evaluation of a Postfix expression.

```
#include <iostream>
#include <string.h>

using namespace std;
struct Stack
{
	int top;
	unsigned capacity;
	int* array;
};

struct Stack* createStack( unsigned capacity )
{
	struct Stack* stack = (struct Stack*) malloc(sizeof(struct Stack));
	if (!stack) return NULL;
	stack->top = -1;
	stack->capacity = capacity;
	stack->array = (int*) malloc(stack->capacity * sizeof(int));
	if (!stack->array) return NULL;
	return stack;
}

int isEmpty(struct Stack* stack)
{
	return stack->top == -1 ;
}

char peek(struct Stack* stack)
{
	return stack->array[stack->top];
}

char pop(struct Stack* stack)
{
	if (!isEmpty(stack))
		return stack->array[stack->top--] ;
	return '$';
}

void push(struct Stack* stack, char op)
{
	stack->array[++stack->top] = op;
}
```

```
int evaluatePostfix(char* exp)
{
        struct Stack* stack = createStack(strlen(exp));
        int i;
        if (!stack) return -1;
        for (i = 0; exp[i]; ++i)
        {
                if (isdigit(exp[i]))
                        push(stack, exp[i] - '0');

                else
                {
                        int val1 = pop(stack);
                        int val2 = pop(stack);
                        switch (exp[i])
                        {
                        case '+': push(stack, val2 + val1); break;
                        case '-': push(stack, val2 - val1); break;
                        case '*': push(stack, val2 * val1); break;
                        case '/': push(stack, val2/val1); break;
                        }
                }
        }
        return pop(stack);
}

int main()
{
        char exp[] = "231*+9-";
        cout<<"postfix evaluation: "<< evaluatePostfix(exp);
        cout<<endl<<"Yash Awasthi (102109029)";
        return 0;
}
```

```
postfix evaluation: -4
Yash Awasthi (102109029)

...Program finished with exit code 0
Press ENTER to exit console.
```