Assignment 3 by Gavisht Singh(1024030358)  
1. Develop a menu driven program demonstrating the following operations on a Stack using array:

1. push(), (ii) pop(), (iii) isEmpty(), (iv) isFull(), (v) display(), and (vi) peek().

Ans  
#include <iostream>

using namespace std;

#define MAX 5

class stack{

     private:

     int arr[MAX];

     int top;

     public:

     stack(){

        top=-1;

     }

     void isFull(){

        if( top==MAX-1){

            cout<<"Yes its full"<<endl;

        }

        else{

            cout<<"Its not full";

        }

     }

     void isEmpty() {

        if(top==-1){

            cout<<"Its empty"<<endl;

        }

        else{

            cout<<"Its not empty"<<endl;

        }

    }

    void push(){

        if(top==MAX-1){

            cout<<"Its full"<<endl;

        }

        else{

            int item;

        cout << "Enter the element to push: ";

        cin >> item;

        top++;

        arr[top] = item;

        }

    }

    void pop(){

        if(top==-1){

            cout<<"Its empty"<<endl;

        }

        else{

            int data=arr[top];

            top--;

            cout<<data<<" Is removed"<<endl;

        }

    }

    void peek(){

        if(top==-1){

            cout<<"Its empty"<<endl;

        }

        else{

            cout<<arr[top]<<endl;

        }

    }

    void display(){

        if(top==-1){

            cout<<"Its empty"<<endl;

        }

        for(int i=top;i>=0;i--){

            cout<<arr[i]<<" ";

        }

        cout<<endl;

    }

};

int main(){

stack s1;

int choice;

int element;

do{

        cout << "1. Push (Add an element)"<<endl;

        cout << "2. Pop (Remove an element)"<<endl;

        cout << "3. Display the stack"<<endl;

        cout << "4. Peek (View the top element)"<<endl;

        cout << "5. Check if Empty"<<endl;

        cout << "6. Check if Full"<<endl;

        cout << "7. Exit"<<endl;

        cout << "Enter your choice: "<<endl;

        cin >> choice;

        switch(choice){

            case 1:

            s1.push();

            break;

            case 2:

            s1.pop();

            break;

            case 3:

            s1.display();

            break;

            case 4:

            s1.peek();

            break;

            case 5:

            s1.isEmpty();

            break;

            case 6:

            s1.isFull();

            break;

            case 7:

            cout<<"Exiting program"<<endl;

            break;

        }

}

while(choice!=7);

*return* 0;

}

2. Given a string, reverse it using STACK. For example “DataStructure” should be output as

“erutcurtSataD.”

Ans:

#include <iostream>

#include <stack>

#include <string>

using namespace std;

int main(){

string s="DataStructures";

stack <char> charStack;

for (char ch : s) {

        charStack.push(ch);

    }

    string reversedString = "";

while(!charStack.empty()){

    char topChar=charStack.top();

    reversedString = reversedString+ topChar;

    charStack.pop();

}

cout<<"Reversed is "<<reversedString<<endl;

*return* 0;

}

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

Ans  
#include <iostream>

#include <stack>

using namespace std;

int main() {

    string expr;

    cout << "Enter an expression: ";

    cin >> expr;

    stack<char> s;

    int is\_balanced = 1;

    for (char ch : expr) {

        switch (ch) {

            case '(':

                s.push(ch);

                break;

            case '{':

                s.push(ch);

                break;

            case '[':

                s.push(ch);

                break;

            case ')':

                if (s.empty() || s.top() != '(') {

                    is\_balanced = 0;

                } else {

                    s.pop();

                }

                break;

            case '}':

                if (s.empty() || s.top() != '{') {

                    is\_balanced = 0;

                } else {

                    s.pop();

                }

                break;

            case ']':

                if (s.empty() || s.top() != '[') {

                    is\_balanced = 0;

                } else {

                    s.pop();

                }

                break;

        }

        if (is\_balanced == 0) {

            break;

        }

    }

    if (is\_balanced == 1 && s.empty()) {

        cout << "Balanced" <<endl;

    } else {

        cout << "Not Balanced" << endl;

    }

*return* 0;

}

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

Ans

#include <iostream>

#include <stack>

#include <cctype>

using namespace std;

int main() {

    string infix;

    string postfix = "";

    stack<char> s;

    cout << "Enter an Infix expression: ";

    getline(cin, infix);

    for (char ch : infix) {

        if (isalnum(ch)) {

            postfix += ch;

        }

        else if (ch == '(') {

            s.push(ch);

        }

        else if (ch == ')') {

            while (!s.empty() && s.top() != '(') {

                postfix += s.top();

                s.pop();

            }

            if (!s.empty()) {

                s.pop();

            }

        }

        else {

            int currentPrecedence = 0;

            if (ch == '^') currentPrecedence = 3;

            else if (ch == '\*' || ch == '/') currentPrecedence = 2;

            else if (ch == '+' || ch == '-') currentPrecedence = 1;

            while (!s.empty() && s.top() != '(') {

                char topOp = s.top();

                int topPrecedence = 0;

                if (topOp == '^') topPrecedence = 3;

                else if (topOp == '\*' || topOp == '/') topPrecedence = 2;

                else if (topOp == '+' || topOp == '-') topPrecedence = 1;

                if (currentPrecedence <= topPrecedence) {

                    postfix += s.top();

                    s.pop();

                } else {

                    break;

                }

            }

            s.push(ch);

        }

    }

    while (!s.empty()) {

        postfix += s.top();

        s.pop();

    }

    cout << "Infix Expression: " << infix << endl;

    cout << "Postfix Expression: " << postfix << endl;

*return* 0;

}

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

Ans  
#include <iostream>

#include <stack>

#include <string>

#include <cctype>

using namespace std;

int main() {

    string postfixExpression;

    stack<int> s;

    cout << "Enter a Postfix expression (single digits only, no spaces): ";

    cin >> postfixExpression;

    for (char ch : postfixExpression) {

        if (isdigit(ch)) {

            s.push(ch - '0');

        }

        else {

            int operand2 = s.top();

            s.pop();

            int operand1 = s.top();

            s.pop();

            switch (ch) {

                case '+':

                    s.push(operand1 + operand2);

                    break;

                case '-':

                    s.push(operand1 - operand2);

                    break;

                case '\*':

                    s.push(operand1 \* operand2);

                    break;

                case '/':

                    s.push(operand1 / operand2);

                    break;

            }

        }

    }

    int result = s.top();

    cout << "Expression: " << postfixExpression << endl;

    cout << "Result: " << result << endl;

*return* 0;

}