**DATA STRUCTURE ASSIGNMENT -4**

**Name-shubham**

**Roll no. - 1024030357**

QUESTION 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().

ANSWER: CODE

#include <bits/stdc++.h>

using namespace std;

#define size 50

class Stack

{

    int element[size];

    int top;

public:

    Stack()

    {

        top = -1;

    }

    void push(int x)

    {

        if(isFull()){

            cout << "stack is full" <<endl;

            return;

        }

        else{

            element[++top]=x;

            cout << x << " is pushed" << endl;

        }

    }

    void pop()

    {

        if(isEmpty())

        {

            cout << "stack is empty" << endl;

            return ;

        }

        else{

            cout << element[top--] << "is popped "<< endl;

        }

    }

    bool isEmpty()

    {

        return (top == -1);

    }

    bool isFull()

    {

        return (top == size -1);

    }

    void peek()

    {

        if(isEmpty())

        {

            cout << "stack is empty" << endl;

            return ;

        }

        else{

            cout << element[top] << " is at the top of stack "<< endl;

        }

    }

    void display()

    {

        if (isEmpty())

        {

            cout << "stack is empty" << endl;

            return ;

        }

        else{

            cout << "elements in stack from top to bottom :" << endl;

            for(int i=top;i>=0;i--)

            {

                cout << element[i] << endl;

            }

        }

    }

};

int main()

{

    Stack s;

    int choice, x;

    while(true)

    {

        cout << "1. push, 2. pop, 3. isEmpty, 4. isFull, 5. peek, 6. display, 7. exit" << endl;

        cout << "Enter your choice: ";

        cin >> choice;

        switch(choice)

        {

            case 1:

            cout << "Enter value to push: ";

            cin >> x;

            s.push(x);

            break;

            case 2:

            cout << "element popped" << endl;

            s.pop();

            break;

            case 3:

            cout << (s.isEmpty() ? "yes" : "no") << endl;

            return 0;

            case 4:

            cout << (s.isFull() ? "yes" : "no") << endl;

            return 0;

            case 5:

            s.peek();

            break;

            case 6:

            s.display();

            break;

            case 7:

            exit(0);

            default:

            cout << "invalid choice" << endl;

        }

    }

    return 0;

}

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

ANSWER : CODE

#include <bits/stdc++.h>

using namespace std;

#define size 50

int main()

{

    char stack[size], input[size];

    int top =-1;

    cout << "enter a string: " <<endl;

    cin.getline(input, size);

    for(int i=0; input[i] !='\0';i++)

    {

        if(top < size-1)

        {

            stack[++top]=input[i];

        }

        else{

            cout << "stack is full " << endl;

            return 1;

        }

    }

    cout << "stack after reversing " << endl;

    for (int i=top; i>=0;i--)

    {

        cout << stack[i];

    }

    return 0;

}

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

ANSWER:CODE

#include <bits/stdc++.h>

using namespace std;

#define size 100

class Solution {

public:

bool isValid(string s) {

stack<char> st;

for(char c : s){

if(c == '(' || c == '{' || c == '['){

st.push(c);

}

else {

if(st.empty()) return false;

char ch = st.top();

if((c == ')' && ch == '(') ||

(c == '}' && ch == '{') ||

(c == ']' && ch == '[')){

st.pop();

} else {

return false;

}

}

}

return st.empty();

}

};

}

int main() {

string s; cin >> s;

cout << (isBalanced(s) ? "Yes" : "No");

return 0;

}

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

ANSWER: CODE

#include <bits/stdc++.h>

using namespace std;

int precedence(char op) {

    if (op == '+' || op == '-') { return 1;}

    if (op == '\*' || op == '/') {return 2;}

    if (op == '^') {return 3;

    return -1

}

int main() {

string s; cin >> s;

stack<char> st; string res;

for (char c : s) {

if (isalnum(c)) res += c;

else if (c=='(') st.push(c);

else if (c==')') {

while (!st.empty() && st.top()!='(') { res+=st.top(); st.pop(); }

st.pop();

} else {

while (!st.empty() && prec(st.top())>=prec(c)) { res+=st.top(); st.pop(); }

st.push(c);

}

}

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

cout << res;

return 0;

}

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

ANSWER:CODE

#include <bits/stdc++.h>

using namespace std;

int main() {

    string postfix\_expression = "231\*+9-";

    stack<int> s;

    for (char c : postfix\_expression) {

        if (isdigit(c)) {

            s.push(c - '0');

        } else {

            int val2 = s.top();

            s.pop();

            int val1 = s.top();

            s.pop();

            switch (c) {

                case '+':

                    s.push(val1 + val2);

                    break;

                case '-':

                    s.push(val1 - val2);

                    break;

                case '\*':

                    s.push(val1 \* val2);

                    break;

                case '/':

                    s.push(val1 / val2);

                    break;

            }

        }

    }

    cout << "Postfix expression: " << postfix\_expression << endl;

    cout << "Result: " << s.top() << endl;

    return 0;

}