**ASSIGNMENT 3**

Q1

#include <iostream>

using namespace std;

#define SIZE 100

int stack[SIZE];

int top=-1;

void push(int value){

if(top==SIZE-1){

cout<<"stack is full";

}

else {

top++;

stack[top]=value;

cout<<value<<"pushed into stack\n";

}

}

void pop(){

if(top==-1){

cout<<"stack is empty";

}

else {

cout<<stack[top]<<"popped out";

top--;

}

}

void isfull(){

if(top==SIZE-1){

cout<<"stack is full\n";

}

else

cout<<"stack is not full\n";

}

void peek(){

if(top==-1){

cout<<"stack is empty\n";

}

else

cout<<"top element is:"<<stack[top];

}

void isEmpty() {

if (top == -1)

cout << "Stack is EMPTY.\n";

else

cout << "Stack is NOT EMPTY.\n";

}

void display(){

if(top==-1){

cout<<"stack is empty\n";

}

else {

cout<<"stack element are:";

for(int i=SIZE-1;i>=0;i--){

cout<<stack[i]<<" ";

cout<<endl;

}

}

}

int main()

{ int value;

int choice;

while (true) {

cout << "\n-- Stack Menu --\n";

cout << "1. Push\n2. Pop\n3. isEmpty\n4. isfull\n5. Peek\n6. Display\n7. Exit\n";

cout << "Enter choice: ";

cin >> choice;

switch (choice) {

case 1:

cout << "Enter value to push: ";

cin >> value;

push(value);

break;

case 2:

pop();

break;

case 3:

isEmpty();

break;

case 4:

isfull();

break;

case 5:

peek();

break;

case 6:

display();

break;

case 7:

return 0;

default:

cout << "Invalid choice!\n";

}

}

return 0;

}

Q2

#include <iostream>

#include<cstring>

using namespace std;

#define SIZE 100

char stack[SIZE];

int top=-1;

void push(char c){

top++;

stack[top]=c;

}

void pop(){

if(top==-1){

cout<<"stack is empty";

}

else {

cout<<stack[top]<<"popped out";

top--;

}

}

void isfull(){

if(top==SIZE-1){

cout<<"stack is full\n";

}

else

cout<<"stack is not full\n";

}

void peek(){

if(top==-1){

cout<<"stack is empty\n";

}

else

cout<<"top element is:"<<stack[top];

}

void isEmpty() {

if (top == -1)

cout << "Stack is EMPTY.\n";

else

cout << "Stack is NOT EMPTY.\n";

}

void display(){

if(top==-1){

cout<<"stack is empty\n";

}

else {

cout<<"stack element are:";

for(int i=SIZE-1;i>=0;i--){

cout<<stack[i]<<" ";

cout<<endl;

}

}

}

int main()

{

char str[SIZE];

cout<<"enter the string:";

cin.getline(str,SIZE);

int len=strlen(str);

for(int i=0;i<len;i++){

push(str[i]);

}

for(int i=SIZE-1;i>=0;i--){

cout<<stack[i];

}

return 0;

}

Q3

#include <iostream>

#include<cstring>

using namespace std;

#define SIZE 100

char stack[SIZE];

int top=-1;

void push(char c){

top++;

stack[top]=c;

}

char pop(){

return stack[top--];

}

int isEmpty() {

if (top == -1){

return 1;}

else

return 0;

}

int main()

{

char str[SIZE];

cout<<"enter the string:";

cin.getline(str,SIZE);

int len=strlen(str);

int i=0;

int balanced=1;

while(str[i]!='\0'){

char c=str[i];

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

push(c);

}

else if(c == ')' || c == '}' || c == ']'){

if (isEmpty()) {

balanced=0;

break;

}

char x=pop();

if ((c == ')' && x != '(') ||

(c == '}' && x != '{') ||

(c == ']' && x != '[')){

balanced=0;

break;

}

}

i++;

}

if(balanced==1){

cout<<"balanced";

}

else

cout<<"not balanced";

return 0;

}

Q4

#include <iostream>

#include <cstring>

using namespace std;

#define SIZE 100

char stack[SIZE];

int top = -1;

void push(char c) {

stack[++top] = c;

}

char pop() {

return stack[top--];

}

char peek() {

return stack[top];

}

int isEmpty() {

if (top == -1) return 1; // true

else return 0; // false

}

int precedence(char op) {

if (op == '^') return 3;

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

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

return 0;

}

int isOperator(char c) {

if (c == '+' || c == '-' || c == '\*' || c == '/' || c == '^')

return 1;

return 0;

}

int main() {

char infix[SIZE], postfix[SIZE];

cout << "Enter infix expression: ";

cin.getline(infix, SIZE);

int k = 0; for (int i = 0; infix[i] != '\0'; i++) {

char c = infix[i];

if (isalnum(c)) {

postfix[k++] = c;

}

else if (c == '(') {

push(c);

}

else if (c == ')') {

while (!isEmpty() && peek() != '(') {

postfix[k++] = pop();

}

pop(); // remove '(' from stack

}

else if (isOperator(c)) {

while (!isEmpty() && precedence(peek()) >= precedence(c)) {

postfix[k++] = pop();

}

push(c);

}

}

while (!isEmpty()) {

postfix[k++] = pop();

}

postfix[k] = '\0';

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

return 0;

}

Q5

#include <iostream>

#include <cstring>

using namespace std;

#define SIZE 100

int stack[SIZE];

int top = -1;

void push(int x) {

stack[++top] = x;

}

int pop() {

return stack[top--];

}

int peek() {

return stack[top];

}

int main() {

char postfix[SIZE];

cout << "Enter postfix expression: ";

cin.getline(postfix, SIZE);

int i = 0;

while (postfix[i] != '\0') {

char c = postfix[i];

if (c >= '0' && c <= '9') {

push(c - '0'); // convert char digit to int

}

else {

int b = pop();

int a = pop();

int result = 0;

if (c == '+') result = a + b;

else if (c == '-') result = a - b;

else if (c == '\*') result = a \* b;

else if (c == '/') result = a / b;

push(result);

}

i++;

}

cout << "Result = " << pop() << endl;

return 0;

}