# Assignment 3

# Q1

# #include <iostream>

# using namespace std;

# class Stack {

# int arr[200];

# int top;

# int size;

# public:

# Stack(int n) { size = n; top = -1; }

# bool isEmpty() { return top == -1; }

# bool isFull() { return top == size - 1; }

# void push(int x) {

# if (isFull()) cout << "Stack Overflow\n";

# else arr[++top] = x;

# }

# void pop() {

# if (isEmpty()) cout << "Stack Underflow\n";

# else top--;

# }

# void peek() {

# if (isEmpty()) cout << "Stack empty\n";

# else cout << "Top: " << arr[top] << "\n";

# }

# void display() {

# if (isEmpty()) cout << "Stack empty\n";

# else {

# cout << "Stack (top to bottom): ";

# for (int i = top; i >= 0; i--) cout << arr[i] << " ";

# cout << "\n";

# }

# }

# };

# int main() {

# int n;

# cout << "Enter stack size (<=200): ";

# cin >> n;

# if (n < 1 || n > 200) { cout << "Invalid size\n"; return 0; }

# Stack s(n);

# int ch, val;

# do {

# cout << "1.Push 2.Pop 3.Peek 4.Display 5.Exit : ";

# cin >> ch;

# if (ch == 1) { cout << "Value: "; cin >> val; s.push(val); }

# else if (ch == 2) s.pop();

# else if (ch == 3) s.peek();

# else if (ch == 4) s.display();

# } while (ch != 5);

# return 0;

# }

# Q2.

# #include <iostream>

# #include <cstring>

# using namespace std;

# int main() {

# char str[200];

# cout << "Enter string (no spaces): ";

# cin >> str;

# int n = strlen(str);

# char st[200];

# int top = -1;

# for (int i = 0; i < n; i++) st[++top] = str[i];

# for (int i = 0; i < n; i++) str[i] = st[top--];

# cout << "Reversed: " << str << "\n";

# return 0;

# }

# Q3

# #include <iostream>

# #include <cstring>

# using namespace std;

# int main() {

# char exp[300];

# cout << "Enter expression (no spaces): ";

# cin >> exp;

# char st[300];

# int top = -1;

# bool ok = true;

# for (int i = 0; i < (int)strlen(exp); i++) {

# char c = exp[i];

# if (c == '(' || c == '[' || c == '{') st[++top] = c;

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

# if (top == -1) { ok = false; break; }

# char t = st[top--];

# if ((c == ')' && t != '(') || (c == ']' && t != '[') || (c == '}' && t != '{')) {

# ok = false; break;

# }

# }

# }

# if (ok && top == -1) cout << "Balanced\n"; else cout << "Not Balanced\n";

# return 0;

# }

# Q4

# #include <iostream>

# #include <cstring>

# using namespace std;

# int prec(char c) {

# if (c == '^') return 3;

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

# if (c == '+' || c == '-') return 1;

# return 0;

# }

# int main() {

# char in[300], post[300];

# cout << "Enter infix (no spaces): ";

# cin >> in;

# char st[300]; int top = -1, k = 0;

# for (int i = 0; in[i] != '\0'; i++) {

# char c = in[i];

# if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') || (c >= '0' && c <= '9'))

# post[k++] = c;

# else if (c == '(') {

# st[++top] = c;

# } else if (c == ')') {

# while (top != -1 && st[top] != '(') post[k++] = st[top--];

# if (top != -1) top--;

# } else {

# while (top != -1 && prec(st[top]) >= prec(c)) post[k++] = st[top--];

# st[++top] = c;

# }

# }

# while (top != -1) post[k++] = st[top--];

# post[k] = '\0';

# cout << "Postfix: " << post << "\n";

# return 0; }

# Q5.

# #include <iostream>

# #include <cstring>

# #include <cctype>

# #include <cmath>

# using namespace std;

# int main() {

# char exp[300];

# cout << "Enter postfix (single-digit operands, no spaces): ";

# cin >> exp;

# int st[300]; int top = -1;

# for (int i = 0; exp[i] != '\0'; i++) {

# char c = exp[i];

# if (isdigit(c)) st[++top] = c - '0';

# else {

# int b = st[top--];

# int a = st[top--];

# int r = 0;

# if (c == '+') r = a + b;

# else if (c == '-') r = a - b;

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

# else if (c == '/') r = a / b;

# else if (c == '^') r = (int)pow(a, b);

# st[++top] = r;

# }

# }

# cout << "Result: " << st[top] << "\n";

# return 0;

# }