

The image shows a C++ IDE with a dark theme. On the left is a sidebar with icons for file explorer, search, and other IDE features. The main editor is split into two panes. The left pane, titled 'main.c', contains the following C++ code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define MAX 100
4 typedef struct {
5     int top;
6     char items[MAX];
7 } Stack;
8 void initStack(Stack* s) { s->top = -1; }
9 int isEmpty(Stack* s) { return s->top == -1; }
10 int isFull(Stack* s) { return s->top == MAX - 1; }
11 void push(Stack* s, char c) { if (!isFull(s)) s->items[++s->top] = c; }
12 char pop(Stack* s) { return isEmpty(s) ? '\0' : s->items[s->top--]; }
13 char peek(Stack* s) { return isEmpty(s) ? '\0' : s->items[s->top]; }
14 int isMatchingPair(char opening, char closing) {
15     return (opening == '(' && closing == ')') ||
16            (opening == '{' && closing == '}') ||
17            (opening == '[' && closing == ']');
18 }
19 int areSymbolsBalanced(const char* str) {
20     Stack s;
21     initStack(&s);
22     for (int i = 0; str[i]; i++) {
23         char ch = str[i];
24         if (ch == '(' || ch == '{' || ch == '[') {
25             push(&s, ch);
26         } else if (ch == ')' || ch == '}' || ch == ']') {
27             if (isEmpty(&s) || !isMatchingPair(pop(&s), ch)) {
28                 return 0;
29             }
30         }
31     }
32     return isEmpty(&s);
33 }
```

The right pane, titled 'Output', shows the execution results:

```
/tmp/vcb5xLHJGK.o
Symbols are balanced.

=== Code Execution Successful ===
```

ChatGPT

Online C Compiler - Programiz

(1) WhatsApp

programiz.com/c-programming/online-compiler/

Programiz
C Online Compiler

aws marketplace
Discover generative AI solutions to spark innovation
Learn more >

Programiz PRO >

main.c

Run

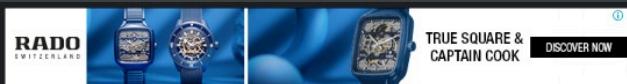
Output

Clear

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4 #define MAX 100
5 typedef struct {
6     int top;
7     char items[MAX];
8 } CharStack;
9 typedef struct {
10     int top;
11     int items[MAX];
12 } IntStack;
13 void initCharStack(CharStack* s) { s->top = -1; }
14 void pushChar(CharStack* s, char c) { s->items[++s->top] = c; }
15 char popChar(CharStack* s) { return s->items[s->top--]; }
16 char peekChar(CharStack* s) { return s->items[s->top]; }
17 int isCharStackEmpty(CharStack* s) { return s->top == -1; }
18 void initIntStack(IntStack* s) { s->top = -1; }
19 void pushInt(IntStack* s, int num) { s->items[++s->top] = num; }
20 int popInt(IntStack* s) { return s->items[s->top--]; }
21 int isIntStackEmpty(IntStack* s) { return s->top == -1; }
22 int precedence(char op) {
23     if (op == '+' || op == '-') return 1;
24     if (op == '*' || op == '/') return 2;
25     return 0;
26 }
27 void infixToPostfix(const char* infix, char* postfix) {
28     CharStack s;
29     initCharStack(&s);
30     int i = 0, j = 0;
31     while (infix[i]) {
32         char ch = infix[i];
```

```
/tmp/ea8ZjXAuYD.o
Postfix Expression: 325*-
Evaluation Result: 13

=== Code Execution Successful ===
```



main.c

Run

Output

Clear

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4 #define MAX 100
5 typedef struct {
6     int top;
7     char items[MAX];
8 } Stack;
9 void initStack(Stack* s) { s->top = -1; }
10 int isEmpty(Stack* s) { return s->top == -1; }
11 void push(Stack* s, char c) { s->items[++s->top] = c; }
12 char pop(Stack* s) { return s->items[s->top--]; }
13 char peek(Stack* s) { return s->items[s->top]; }
14 int precedence(char op) {
15     switch (op) {
16         case '+': case '-': return 1;
17         case '*': case '/': return 2;
18         case '^': return 3;
19         default: return 0;
20     }
21 }
22 void infixToPostfix(const char* infix, char* postfix) {
23     Stack s;
24     initStack(&s);
25     int i = 0, j = 0;
26     while (infix[i]) {
27         char ch = infix[i];
28         if (isalnum(ch)) {
29             postfix[j++] = ch;
30         } else if (ch == '(') {
31             push(&s, ch);
32         } else if (ch == ')') {
```

```
/tmp/pYxh9cF28q.o
Infix Expression: A+B*(C^D-E)
Postfix Expression: ABCD^E-*+

=== Code Execution Successful ===
```