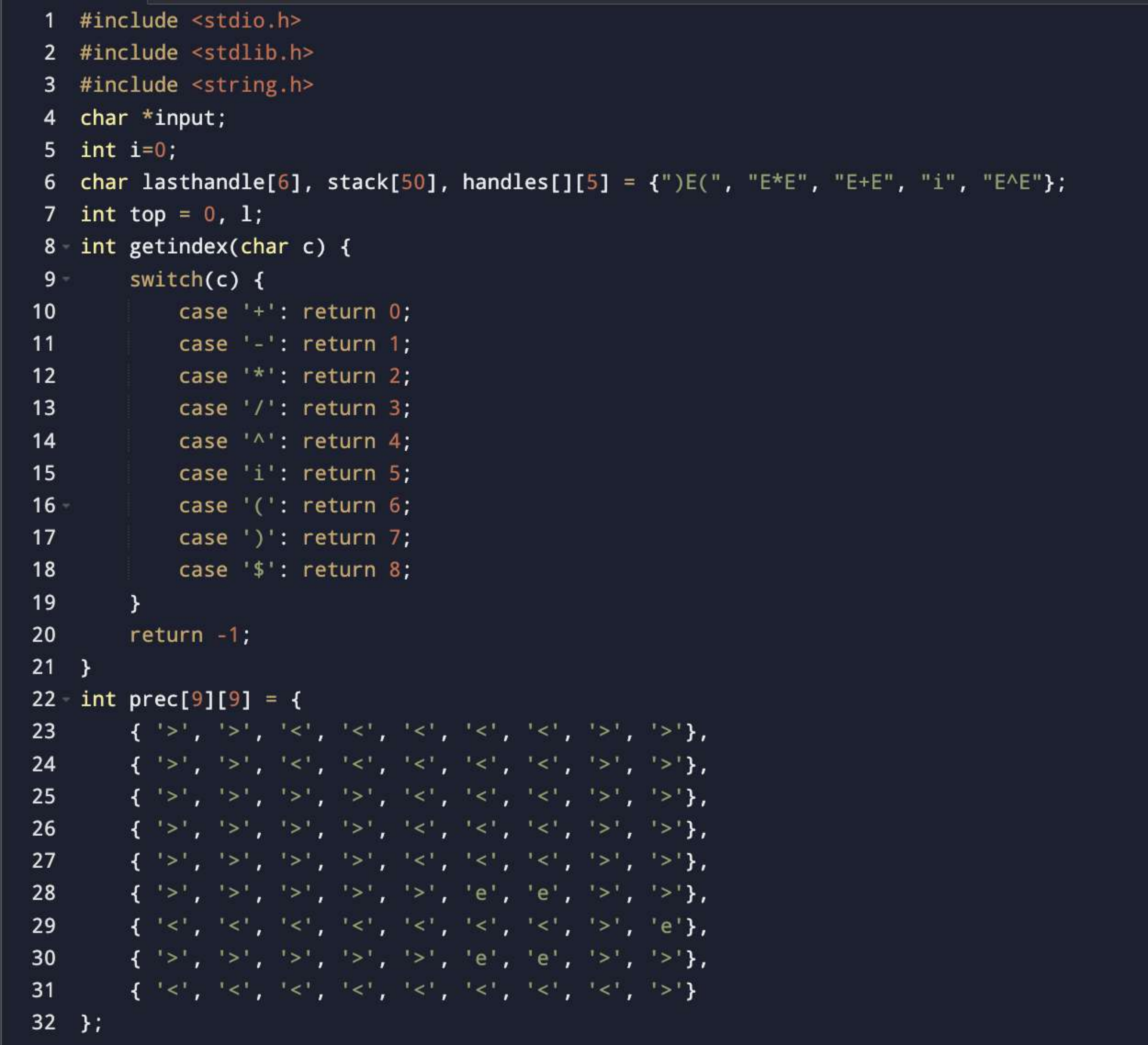
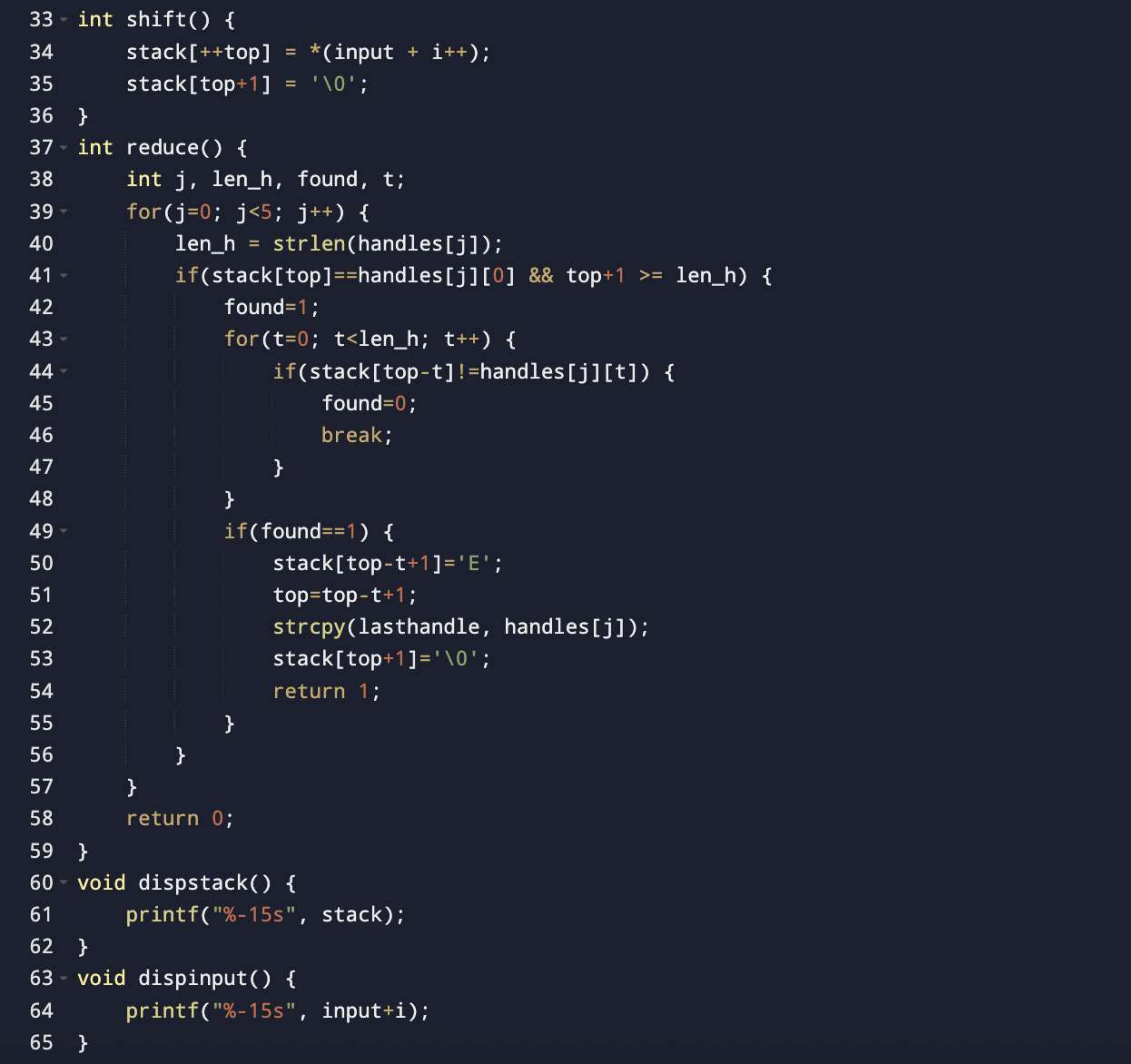
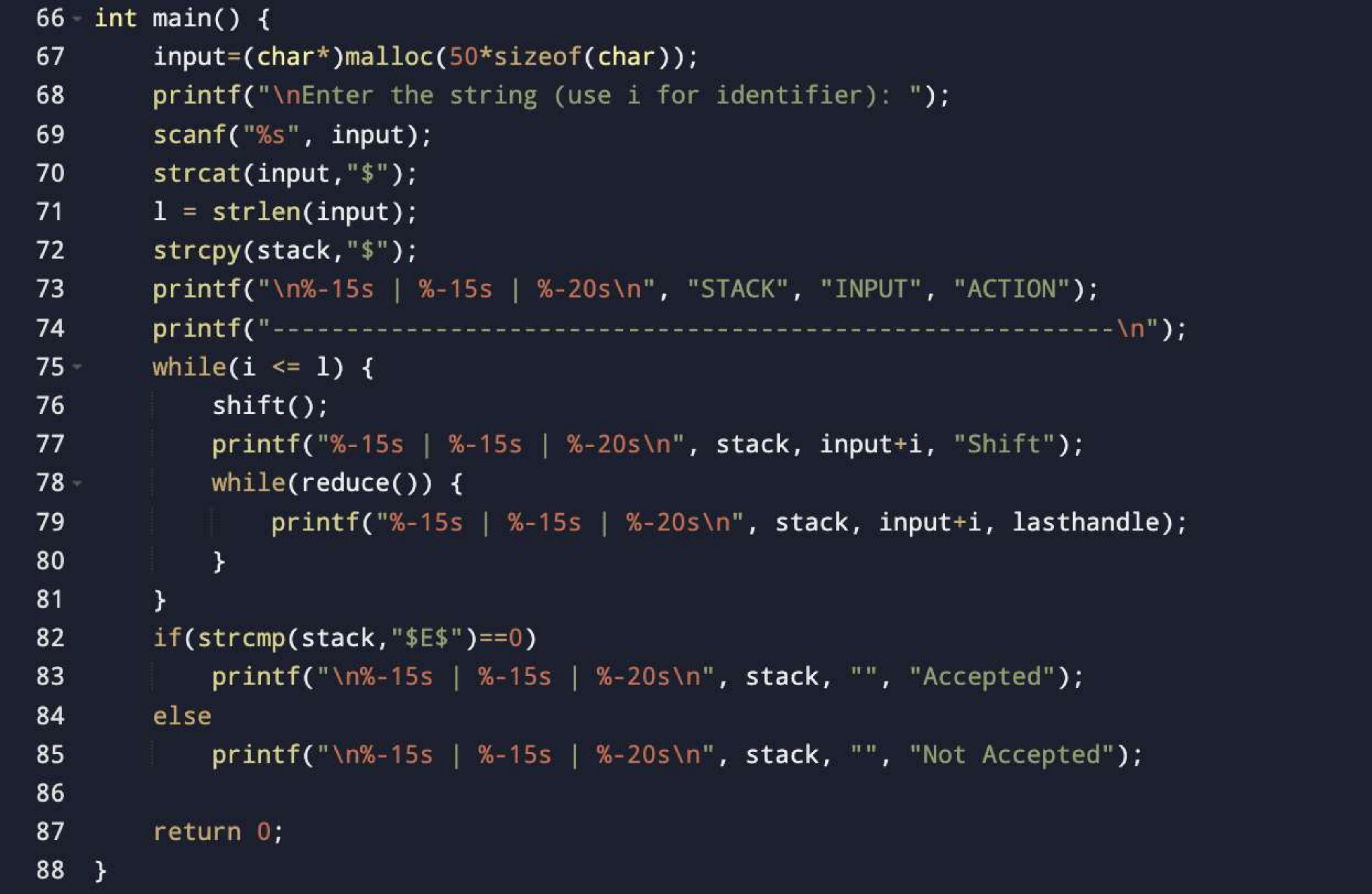


```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char *input;
int i=0;
char lasthandle[6], stack[50], handles[][5] = {")E(", "E*E", "E+E", "i", "E^E"};
int top = 0, l;
int getindex(char c) {
    switch(c) {
        case '+': return 0;
        case '-': return 1;
        case '*': return 2;
        case '/': return 3;
        case '^': return 4;
        case 'i': return 5;
        case '(': return 6;
        case ')': return 7;
        case '$': return 8;
    }
    return -1;
}
int prec[9][9] = {
    { '>', '>', '<', '<', '<', '<', '<', '>', '>'},
    { '>', '>', '<', '<', '<', '<', '<', '>', '>'},
    { '>', '>', '>', '>', '<', '<', '<', '>', '>'},
    { '>', '>', '>', '>', '<', '<', '<', '>', '>'},
    { '>', '>', '>', '>', '<', '<', '<', '>', '>'},
    { '>', '>', '>', '>', '>', 'e', 'e', '>', '>'},
    { '<', '<', '<', '<', '<', '<', '<', '>', 'e'},
    { '>', '>', '>', '>', '>', 'e', 'e', '>', '>'},
    { '<', '<', '<', '<', '<', '<', '<', '<', '>'}
};
```

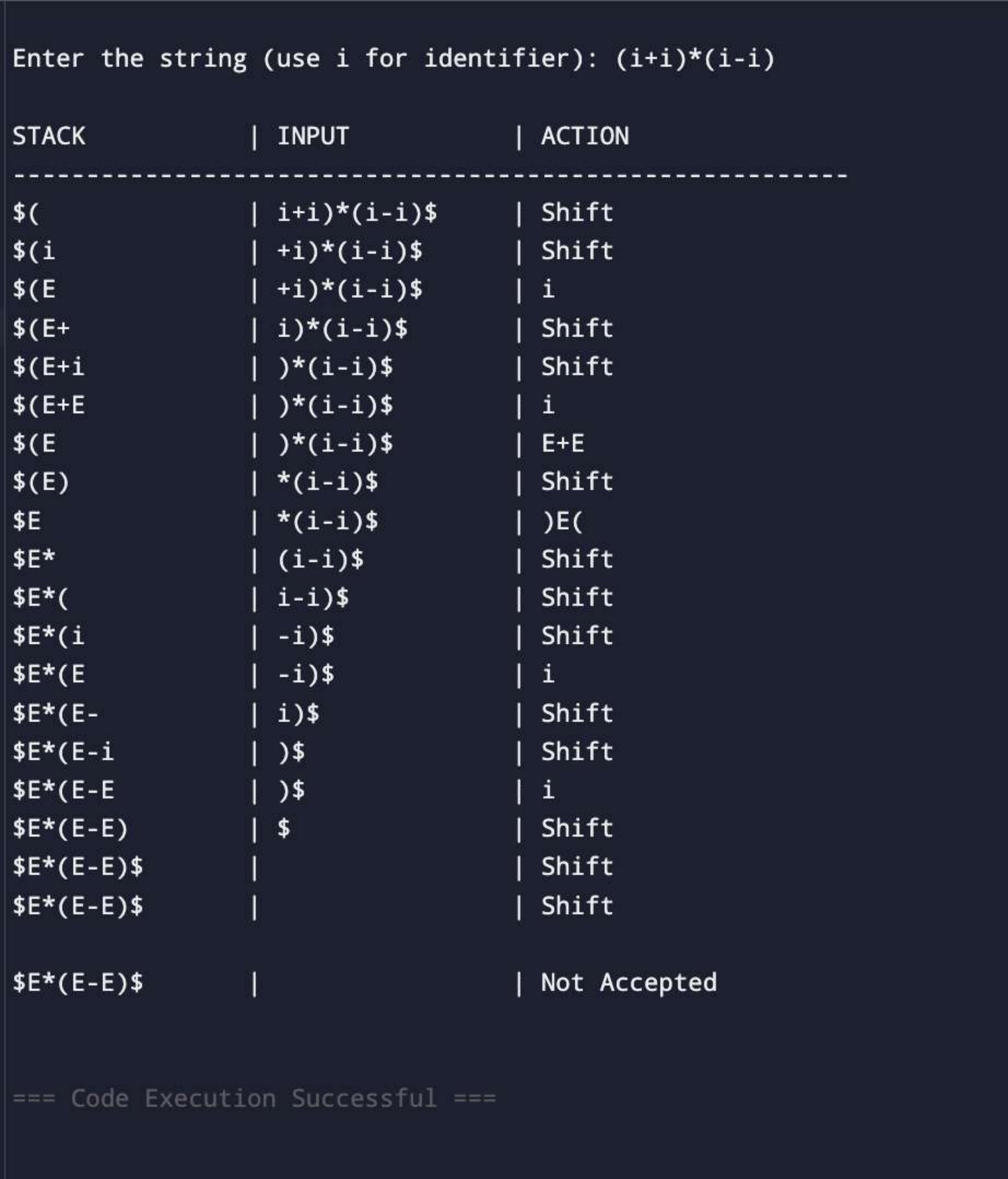


```c
int shift() {
    stack[++top] = *(input + i++);
    stack[top+1] = '\0';
}
int reduce() {
    int j, len_h, found, t;
    for(j=0; j<5; j++) {
        len_h = strlen(handles[j]);
        if(stack[top]==handles[j][0] && top+1 >= len_h) {
            found=1;
            for(t=0; t<len_h; t++) {
                if(stack[top-t]!=handles[j][t]) {
                    found=0;
                    break;
                }
            }
            if(found==1) {
                stack[top-t+1]='E';
                top=top-t+1;
                strcpy(lasthandle, handles[j]);
                stack[top+1]='\0';
                return 1;
            }
        }
    }
    return 0;
}
void dispstack() {
    printf("%-15s", stack);
}
void dispinput() {
    printf("%-15s", input+i);
}
```



```c
int main() {
    input=(char*)malloc(50*sizeof(char));
    printf("\nEnter the string (use i for identifier): ");
    scanf("%s", input);
    strcat(input,"$");
    l = strlen(input);
    strcpy(stack,"$");
    printf("\n%-15s | %-15s | %-20s\n", "STACK", "INPUT", "ACTION");
    printf("--------------------------------------------------------\n");
    while(i <= l) {
        shift();
        printf("%-15s | %-15s | %-20s\n", stack, input+i, "Shift");
        while(reduce()) {
            printf("%-15s | %-15s | %-20s\n", stack, input+i, lasthandle);
        }
    }
    if(strcmp(stack,"$E$")==0)
        printf("\n%-15s | %-15s | %-20s\n", stack, "", "Accepted");
    else
        printf("\n%-15s | %-15s | %-20s\n", stack, "", "Not Accepted");

    return 0;
}
```



```
Enter the string (use i for identifier): (i+i)*(i-i)

STACK           | INPUT           | ACTION
---------------------------------------------------------
$(              | i+i)*(i-i)$     | Shift
$(i             | +i)*(i-i)$      | Shift
$(E             | +i)*(i-i)$      | i
$(E+            | i)*(i-i)$       | Shift
$(E+i           | )*(i-i)$        | Shift
$(E+E           | )*(i-i)$        | i
$(E             | )*(i-i)$        | E+E
$(E)            | *(i-i)$         | Shift
$E              | *(i-i)$         | )E(
$E*             | (i-i)$          | Shift
$E*(            | i-i)$           | Shift
$E*(i           | -i)$            | Shift
$E*(E           | -i)$            | i
$E*(E-          | i)$             | Shift
$E*(E-i         | )$              | Shift
$E*(E-E         | )$              | i
$E*(E-E)        | $               | Shift
$E*(E-E)$       |                 | Shift
$E*(E-E)$       |                 | Shift

$E*(E-E)$       |                 | Not Accepted



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