```c
#include<stdio.h>
#include<string.h>
#define MAX_ELE 30

// Function to return the precedence of operators
int f(char s) {
    switch(s) {
        case '+':
        case '-': return 2;
        case '*':
        case '/': return 4;
        case '$':
        case '^': return 5;
        case '(': return 0;
        case '#': return -1;
        default: return 8;
    }
}

// Function to return the precedence of operators for incoming operators
int g(char s) {
    switch(s) {
        case '+':
        case '-': return 1;
        case '*':
        case '/': return 3;
        case '$':
        case '^': return 6;
        case '(': return 9;
        case ')': return 0;
        default: return 7;
    }
}

int main() {
    char c, s[MAX_ELE]={'#'}; // Stack initialized with '#'
    char inf[MAX_ELE] = "a/b-(c+d)";  // Example infix expression
    char pf[MAX_ELE];
    int top = 0, i, j = 0; // top = 0 because '#' is already stored in stack

    // Loop through the infix expression
    for(i = 0; i < strlen(inf); i++) {
        c = inf[i];

        // While the precedence of the operator at top of the stack is greater than the current operator
        while(f(s[top]) > g(c)) {
            pf[j] = s[top];
            j++;
            top--;
        }

        // If the precedence is not equal, push the operator onto the stack
```

```
53          if(f(s[top]) ≠ g(c))
54              s[++top] = c;
55          else
56              top--;
57      }
58
59      // Pop the remaining operators from the stack
60      for(; s[top] ≠ '#'; top--)
61          pf[j++] = s[top];
62      pf[j] = '\0';  // Null-terminate the postfix expression
63
64      printf("The postfix expression: %s\n", pf);
65
66      return 0;
67  }
68
```