22_HEMANT_17.c

```c
/*
Roll no : 22
Batch: A
Author name: Hemant Gupta
Date: 23/08/2024
Description: Program for expression evaluation

*/


#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>

#define MAX 100

double stack[MAX];
int top = -1;

// Push an item onto the stack
void push(double item) {
    if (top < MAX - 1) {
        stack[++top] = item;
    } else {
        printf("Stack overflow!\n");
        exit(EXIT_FAILURE);
    }
}

// Pop an item from the stack
double pop() {
    if (top >= 0) { // Corrected condition
        return stack[top--];
    } else {
        printf("Stack underflow!\n");
        exit(EXIT_FAILURE);
    }
}

// Function to evaluate postfix expression
double evaluatePostfix(char *postfix) {
    char *p = postfix;

    while (*p) {
        // Skip newline characters
        if (*p == '\n') {
            p++;
            continue;
        }

        // Skip spaces
```

```c
        if (*p == ' ') {
            p++;
            continue;
        }

        if (isdigit(*p) || (*p == '-' && isdigit(*(p + 1)))) { // Allow negative numbers
            // Convert number and push to stack
            push(atof(p));           // atof converts string to double
            while (isdigit(*p) || *p == '.') {
                p++;
            }
        } else {
            // Ensure there are enough operands on the stack before performing the operation
            if (top < 1) {                  // We need at least two operands
                printf("Error: Not enough operands for operation '%c'\n", *p);
                exit(EXIT_FAILURE);
            }

            double b = pop();
            double a = pop();
            switch (*p) {
                case '+': push(a + b); break;
                case '-': push(a - b); break;
                case '*': push(a * b); break;
                case '/':
                    if (b != 0) {
                        push(a / b);
                    } else {
                        printf("Error: Division by zero!\n");
                        exit(EXIT_FAILURE);
                    }
                    break;
                default:
                    printf("Error: Unknown operator '%c'\n", *p);
                    exit(EXIT_FAILURE);
            }
            p++;         // Move to the next character
        }
    }

    // Ensure that there is only one result left on the stack
    if (top != 0) {     // Corrected condition
        printf("Error: The expression is invalid (too many operands).\n");
        exit(EXIT_FAILURE);
    }

    return pop(); // Return the final result
}

int main() {
    char postfix[MAX];

    printf("Enter a postfix expression (e.g., '3 5 2 - *'): ");
    fgets(postfix, sizeof(postfix), stdin);
```

```c

    double result = evaluatePostfix(postfix);
    printf("Result: %.2f\n", result);

    return 0;
}

```