

CC LAB 07 | Postfix Evaluation

Aim: Lex program to evaluate a postfix expression

Implementation:

```
%{
#include <stdio.h>
#include <stdlib.h>

#define MAX_STACK_SIZE 100

int stack[MAX_STACK_SIZE];
int top = -1;

void push(int value) {
    if (top >= MAX_STACK_SIZE - 1) {
        fprintf(stderr, "Stack overflow\n");
        exit(1);
    }
    stack[++top] = value;
}

int pop() {
    if (top < 0) {
        fprintf(stderr, "Stack underflow\n");
        exit(1);
    }
    return stack[top--];
}
}%

%option noyywrap

%%
[0-9]+    { int value = atoi(yytext); push(value); }
[-+*/]    {
            int operand2 = pop();
            int operand1 = pop();
            int result;
            switch (yytext[0]) {
                case '+': result = operand1 + operand2; break;
                case '-': result = operand1 - operand2; break;
                case '*': result = operand1 * operand2; break;
                case '/': result = operand1 / operand2; break;
            }
        }
```

```
        }
        push(result);
    }
    [ \t\n] ; // Ignore whitespace
    . {
        fprintf(stderr, "Invalid character: %s\n", yytext);
        exit(1);
    }
}

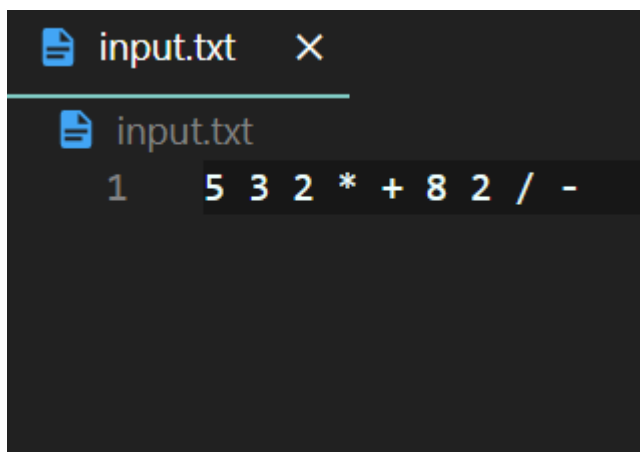
%%

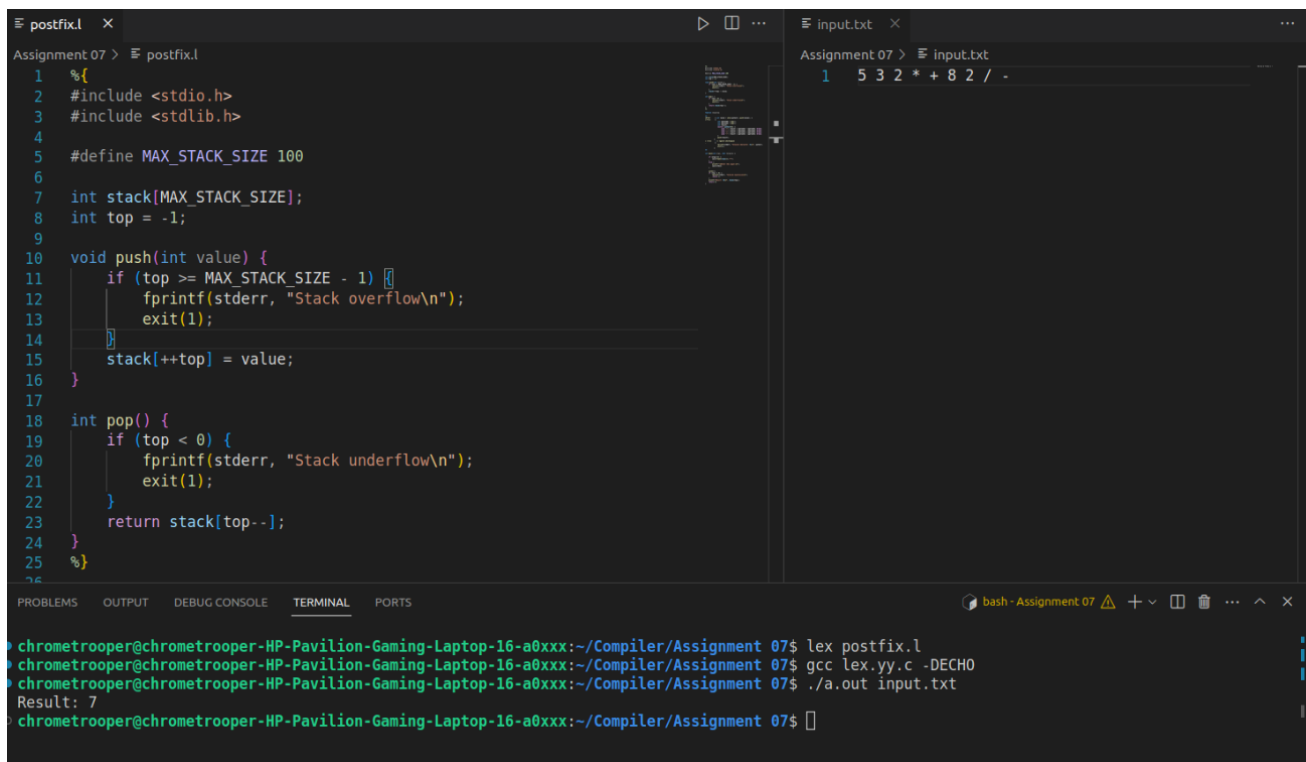
int main(int argc, char *argv[]) {

    if (argc==2) {
        yyin=fopen(argv[1], "r");
    }
    else {
        printf("\nEnter the input:\n");
        yyin=stdin;
    }

    yylex();
    if (top != 0) {
        fprintf(stderr, "Invalid expression\n");
        return 1;
    }
    printf("Result: %d\n", stack[top]);
    return 0;
}
```

Input:



Output:

The screenshot displays a code editor with two files: `postfix.l` and `input.txt`. The `postfix.l` file contains a C program for a postfix calculator using a stack. The `input.txt` file contains the expression `1 5 3 2 * + 8 2 / -`. The terminal at the bottom shows the compilation and execution process, resulting in the output `Result: 7`.

```
1  %{
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  #define MAX_STACK_SIZE 100
6
7  int stack[MAX_STACK_SIZE];
8  int top = -1;
9
10 void push(int value) {
11     if (top >= MAX_STACK_SIZE - 1) {
12         fprintf(stderr, "Stack overflow\n");
13         exit(1);
14     }
15     stack[++top] = value;
16 }
17
18 int pop() {
19     if (top < 0) {
20         fprintf(stderr, "Stack underflow\n");
21         exit(1);
22     }
23     return stack[top--];
24 }
25 %}
```

```
Assignment 07 > input.txt
1  1 5 3 2 * + 8 2 / -
```

```
bash - Assignment 07
chrometrooper@chrometrooper-HP-Pavilion-Gaming-Laptop-16-a0xxx:~/Compiler/Assignment 07$ lex postfix.l
chrometrooper@chrometrooper-HP-Pavilion-Gaming-Laptop-16-a0xxx:~/Compiler/Assignment 07$ gcc lex.yy.c -DECHO
chrometrooper@chrometrooper-HP-Pavilion-Gaming-Laptop-16-a0xxx:~/Compiler/Assignment 07$ ./a.out input.txt
Result: 7
chrometrooper@chrometrooper-HP-Pavilion-Gaming-Laptop-16-a0xxx:~/Compiler/Assignment 07$
```