2020년도 15회 E-PPER 대비 기출 풀이 - 8번

E-PPER(Ewha Power ProgrammER) 프로그래밍 경진대회
2019년도 제 13회 EPEER C부문 기출 풀이
담당: 컴공 17권지현

- 8번 발표자료 링크 (dropbox paper로 연결됩니다.)
- 8번 구름 테스트 링크 (구름 ide에서 직접 코드를 작성하고 테스트케이스를 확인할 수 있습니다.)

🎯 8번 후위표기법 (스택)

[문제 설명]

- 후위표기법의 수식을 입력받아 계산하는 프로그램을 작성하시오.
- A + B 와 같이 피연산자 A와 B 중간에 연산자 +가 위치하 는 계산방식을 중위표기법(Infix Notation)이라 부른다.
- 후위 표기법(Postfix Notation)은 A B + 와 같이 피연산자 A와 B 의 뒤에 연산자 +가 위치한 표기법을 말한다. 수식 (5 + 8) * 2 를 후위표기법으로 바꾸면 5 8 + 2 * 와 같이 되어, 괄호가 없이도 연산자의 우선 순위를 명확히 할 수 있다는 장점이 있다.

[입력 형식]

- 첫 번째 줄에 총 입력되는 연산자와 피연산자의 개수의 합 m을 입력한다. ($3 \le m \le 11$)
- 그 다음 줄에 m개의 연산자와 피연산자를 공백을 구분 으로 입력한다. 피연산자 x는 $0 \le x \le 9$ 의 법위를 갖는 정수이며, 연산자는 사칙연산인 * / + - 을 입력한다. (0으로 나누는 경우는 입력하지 않고, 피연산자나 연산자 가 부족한 경우와 같이 완전하지 않은 수식을 입력하지 않는다. 나눗셈 연산의 경우, 몫이 반환된다.)

[출력 형식]

- 계산 결과를 출력한다.

🔍 풀이

후위 표기법은 대표적으로 stack을 사용하는 문제입니다.
문제에서 주어진 식을 저장하는 char형 배열(*input[]*)과, 연산 결과를 저장하는 int 형 스택(*stack[]*)을 선언한 후 값을 계산하면 쉽게 풀 수 있습니다.

첫번째 테스트케이스의 연산 순서는 다음과 같습니다.

#1	*TEST CASE*
input	3

	2 3 +
output	5

1. 입력 데이터 저장하기 char input[]

index	0	1	2
value	2	3	+

위 표는 주어진 식이 input에 저장된 형태입니다.

2. input 탐색하기

input[i]를 탐색하며 만약 숫자인 경우, stack[]에 push연산으로 삽입됩니다. i=2일때 stack에 저장된 형태는 다음과 같습니다.

index	0	1	2
value	2	3	

만약 연산 기호인 경우, stack[]에 pop연산으로 두 숫자를 꺼낸 후 값을 계산합니다. 계산한 값은 다시 stack에 저장됩니다. 이 경우, 3과 2가 차례로 pop되어 '+'연산을 거친 후 5가 저장됩니다. i=3일 때, stack에 저장된 형태는 다음과 같습니다.

index	0	1	2
value	5		

3. 결과 출력

input[] 탐색이 종료되면 stack[0]이 결과값으로 리턴됩니다. 이 경우, 결과값은 5가 출력됩니다.

[수도코드]

```
// 필요한 변수 정의
int stack[n];
int top=-1; //스택 포인터 역할

// 스택 연산에서 필요한 함수 정의
int pop();
void push(int i);

//계산한 후 결과값을 출력하는 함수 정의
```

```
int cal(int n);

int main(){
    int n 사용자 입력 받기;
    char input[n] 사용자 입력 받기;

    int answer = cal(n);
    answer 출력하기;

    return 0;
}

int cal(int n){
    input[i]가 숫자라면
        stack push 연산하기;

    input[i]가 연산 기호라면
        '+'인 경우
                stack[top] = stack[top] + stack[top-1];
                stack pop 연산하기;
        '-'인 경우
                stack[top] = stack[top] - stack[top-1];
                    stack pop 연산하기;
        '*'인 경우
                stack[top] = stack[top] * stack[top-1];
            stack pop 연산하기;
            '/'인 경우
                stack[top] = stack[top] / stack[top-1];
            stack pop 연산하기;
    stack[0] 리턴;
    }
}
```