

**자료구조 및 알로리즘 과제**

**항공우주및소프트웨어공학과**

**강경석**

**2020080164**



**1. 스택 (10828)**

## 문제

정수를 저장하는 스택을 구현한 다음, 입력으로 주어지는 명령을 처리하는 프로그램을 작성하시오.

명령은 총 다섯 가지이다.

* push X: 정수 X를 스택에 넣는 연산이다.
* pop: 스택에서 가장 위에 있는 정수를 빼고, 그 수를 출력한다. 만약 스택에 들어있는 정수가 없는 경우에는 -1을 출력한다.
* size: 스택에 들어있는 정수의 개수를 출력한다.
* empty: 스택이 비어있으면 1, 아니면 0을 출력한다.
* top: 스택의 가장 위에 있는 정수를 출력한다. 만약 스택에 들어있는 정수가 없는 경우에는 -1을 출력한다.

## 입력

첫째 줄에 주어지는 명령의 수 N (1 ≤ N ≤ 10,000)이 주어진다. 둘째 줄부터 N개의 줄에는 명령이 하나씩 주어진다. 주어지는 정수는 1보다 크거나 같고, 100,000보다 작거나 같다. 문제에 나와있지 않은 명령이 주어지는 경우는 없다.

## 출력

출력해야하는 명령이 주어질 때마다, 한 줄에 하나씩 출력한다.

예제 입력 1 복사

14

push 1

push 2

top

size

empty

pop

pop

pop

size

empty

pop

push 3

empty

top

예제 출력 1 복사

2

2

0

2

1

-1

0

1

-1

0

3

예제 입력 2 복사

7

pop

top

push 123

top

pop

top

pop

예제 출력 2 복사

-1

-1

123

123

-1

-1

**(2) Source Code(소스코드) 및 설명**

// 자료구조 및 알고리즘

//21-1 스택(10828)

//헤더파일

#include <stdio.h>

#include <string.h> //strcmp 함수가 내장된 헤더선언

int stack[1000001]; // 횟수 1000001(으)로 제한

int cnt = 0; // 스택사이즈 저장

void push(int n) { // stack에 넣을 값 입력(입력시마다 증가)

stack[cnt] = n;

cnt++;

}

void pop() { // stack[cnt] - 가장 최근값; printf출력후 stack[cnt]에 0을 넣어 초기화

if (cnt != 0) {

cnt--;

printf("%d\n", stack[cnt]);

stack[cnt] = 0;

}

else {

printf("%d\n", -1); // 더이상 꺼낼 스택이 없을시 -1 출력

}

}

void size() {

printf("%d\n", cnt); // 스택에 저장된 값의 갯수 출력

}

void empty() {

if (cnt != 0) {

printf("0\n"); // 스택에 저장된 값이 있을시 0 출력

}

else {

printf("1\n"); // 스택에 저장된 값이 없을시 -1 출력

}

}

void top() {

if (cnt != 0) {

printf("%d\n", stack[cnt - 1]);

}

else {

printf("%d\n", -1); // 스택에 저장된 값이 없을시 -1 출력

}

}

int main()

{

int stack\_size;

int i; //쌓인 스택수

scanf("%d", &stack\_size);

char order[100]; // 입력 글자크기 최대 100글자로 설정

for (i = 0; i < stack\_size; i++) {

scanf("%s", &order); // 명령어 입력

if (!strcmp(order, "push")) {

int n;

scanf("%d", &n);

push(n);

}

else if (!strcmp(order, "top")) {

top();

}

else if (!strcmp(order, "size")) {

size();

}

else if (!strcmp(order, "empty")) {

empty();

}

else {

pop();

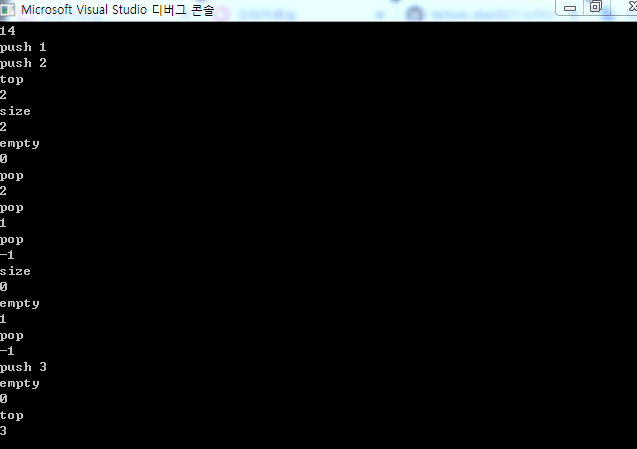
}

}

return 0;

}

**(3) Screenshot of the Results**



**(4) 문제를 해결하면서 느낀점**

수업시간에 학습한 스택을 가지고 처음으로 문제사이트에서 코드를 짜서 풀어보았다. 5개의 함수로 구성이 된 코드를 짜면서 스택의 각 기능을 이해할수 있는 좋은 기회였던 것 같다. 하지만 아직은 스택이 어려운 것 같다…많이 코딩해보면서 그냥 보고 타이핑 하는 것이 아니라 이해하고 변형할수 있도록 노력해야 겠다

문제해결에 있어서 주어지는 조건의 중요성을 알게 되었다.

**백준에서 채점결과 성공으로 판별됨.**

**2. 제로(10773)**

**(1) 소개**

## 문제

나코더 기장 재민이는 동아리 회식을 준비하기 위해서 장부를 관리하는 중이다.

재현이는 재민이를 도와서 돈을 관리하는 중인데, 애석하게도 항상 정신없는 재현이는 돈을 실수로 잘못 부르는 사고를 치기 일쑤였다.

재현이는 잘못된 수를 부를 때마다 0을 외쳐서, 가장 최근에 재민이가 쓴 수를 지우게 시킨다.

재민이는 이렇게 모든 수를 받아 적은 후 그 수의 합을 알고 싶어 한다. 재민이를 도와주자!

## 입력

첫 번째 줄에 정수 K가 주어진다. (1 ≤ K ≤ 100,000)

이후 K개의 줄에 정수가 1개씩 주어진다. 정수는 0에서 1,000,000 사이의 값을 가지며, 정수가 "0" 일 경우에는 가장 최근에 쓴 수를 지우고, 아닐 경우 해당 수를 쓴다.

정수가 "0"일 경우에 지울 수 있는 수가 있음을 보장할 수 있다.

## 출력

재민이가 최종적으로 적어 낸 수의 합을 출력한다. 최종적으로 적어낸 수의 합은 231-1보다 작거나 같은 정수이다.

## 예제 입력 1 복사

4

3

0

4

0

## 예제 출력 1 복사

0

## 예제 입력 2 복사

10

1

3

5

4

0

0

7

0

0

6

## 예제 출력 2 복사

7

**(2) Source Code(소스코드) 및 설명**

// 자료구조 및 알고리즘

//21-1 제로(10773)

#include <stdio.h>

int stack[100001];

int size = 0;

void push(int a) {

stack[size++] = a;

}

int pop() {

return stack[--size];

}

int main(void) {

int n;

int input;

int res = 0;

int i;

scanf("%d", &n); //n입력받음

// 스택에 입력(push)와 출력(pop)작업 수행

for (i = 0; i < n; i++) {

scanf("%d", &input); //입력받음

if (input == 0)

pop(); // 0이면 지운다

else

push(input); // 0이 아니면 스택에 쌓는다.

}

//printf("%d\n", size);

// 스택 값 합구하기

for (i = 0; i < size; i++) {

res += stack[i];

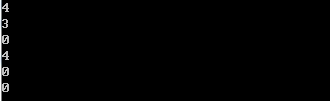
}

printf("%d", res); // 결과값 출력

return 0;

}

**(3) Screenshot of the Results**



**(4) 문제를 해결하면서 느낀점**

스택의 push, pop의 함수를 통해서 비교적 간단하게 문제를 해결할수 있었다.

초반에 size변수를 0으로 초기화 하지 않아서 결과값에 오류가 발생을 하였다.

결과와 관련된 변수들은 한번더 확인하고 결과값에 영향이 가지 않도록 초기화를 확인하는 습관을 가져야 겠다.

백준 채점결과 성공처리 되었다.

**3. 괄호 (9012)**

**(1) 소개**

## 문제

괄호 문자열(Parenthesis String, PS)은 두 개의 괄호 기호인 ‘(’ 와 ‘)’ 만으로 구성되어 있는 문자열이다. 그 중에서 괄호의 모양이 바르게 구성된 문자열을 올바른 괄호 문자열(Valid PS, VPS)이라고 부른다. 한 쌍의 괄호 기호로 된 “( )” 문자열은 기본 VPS 이라고 부른다. 만일 x 가 VPS 라면 이것을 하나의 괄호에 넣은 새로운 문자열 “(x)”도 VPS 가 된다. 그리고 두 VPS x 와 y를 접합(concatenation)시킨 새로운 문자열 xy도 VPS 가 된다. 예를 들어 “(())()”와 “((()))” 는 VPS 이지만 “(()(”, “(())()))” , 그리고 “(()” 는 모두 VPS 가 아닌 문자열이다.

여러분은 입력으로 주어진 괄호 문자열이 VPS 인지 아닌지를 판단해서 그 결과를 YES 와 NO 로 나타내어야 한다.

## 입력

입력 데이터는 표준 입력을 사용한다. 입력은 T개의 테스트 데이터로 주어진다. 입력의 첫 번째 줄에는 입력 데이터의 수를 나타내는 정수 T가 주어진다. 각 테스트 데이터의 첫째 줄에는 괄호 문자열이 한 줄에 주어진다. 하나의 괄호 문자열의 길이는 2 이상 50 이하이다.

## 출력

출력은 표준 출력을 사용한다. 만일 입력 괄호 문자열이 올바른 괄호 문자열(VPS)이면 “YES”, 아니면 “NO”를 한 줄에 하나씩 차례대로 출력해야 한다.

## 예제 입력 1 복사

6

(())())

(((()())()

(()())((()))

((()()(()))(((())))()

()()()()(()()())()

(()((())()(

## 예제 출력 1 복사

NO

NO

YES

NO

YES

NO

## 예제 입력 2 복사

3

((

))

())(()

## 예제 출력 2 복사

NO

NO

NO

**(2) Source Code(소스코드) 및 설명**

// 자료구조 및 알고리즘

//21-1 괄호(9012)

#include <stdio.h>

#include <string.h>

#define MAX\_STACK\_SIZE 52

typedef struct

{

char data[MAX\_STACK\_SIZE];

int top;

}StackType;

//스택초기화

void init\_stack(StackType\* s)

{

s->top = -1;

}

//공백상테 검출 함수

int is\_empty(StackType\* s)

{

return (s->top == -1);

}

// 포화 상태 검출 함수

int is\_full(StackType\* s)

{

return (s->top == MAX\_STACK\_SIZE - 1);

}

// 삽입함수

void push(StackType\* s, char item)

{

if (is\_full(s)) {

return;

}

else s->data[++s->top] = item;

}

// 삭제함수

char pop(StackType\* s)

{

if (is\_empty(s)) {

return 1;

}

else return s->data[s->top--];

}

int main()

{

StackType s;

init\_stack(&s);

char text[MAX\_STACK\_SIZE], item;

int i, j, n;

int flag = 1; //괄호 짝 판단값 저장 변수 if) 안맞는 경우 0으로 바뀜

scanf("%d", &n);

getchar(); // 버퍼비우기

for (i = 0; i < n; i++)

{

fgets(text, sizeof(text), stdin);// 개행문자까지 포함됨. fgets를 사용한 이유: get은 메모리의 침범으로 프로그램에 오류를 발생을 시킨다. 되도록이면 메모리의 주소를 지정해주는 fgets사용

for (j = 0; j < strlen(text) - 1; j++)

{

if (text[j] == '(') {

push(&s, text[j]);

}

else if (text[j] == ')') {

item = pop(&s);

if (item != '(') {

flag = 0;

break;

}

}

}

if (s.top != -1)

flag = 0;

//printf("%d\n", flag);

if (flag == 0)

printf("NO\n"); // flag==0 => 괄호의 짝이 안맞음 => no출력

else

printf("YES\n");

flag = 1; //flag재 초기화, 만약 0으로 바꼈는데 초기화 안해주면 정상작동이 안됨.

s.top = -1; // top초기화, 다음 항목작업을 수행하기 위해서 top을 초기의 -1로 재 초기화

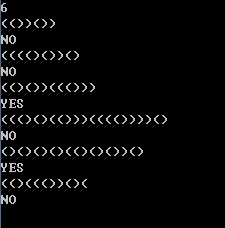
}

return 0;

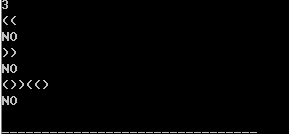
}

**(3) Screenshot of the Results**

(1)



(2)



**(4) 문제를 해결하면서 느낀점**

본 문제를 해결하기 위해서 일주일이라는 시간을 걸렸던 것 같다.

문제의 맞춘인원의 비율또한 42.126%로서 쉽지 않은 문제인 것 같았는데 스택

에 대한 각 함수의 기능과 C언어의 지식이 없이는 풀기 힘든 문제였던 것 같다.

수많은 시도와 고민끝에 괄호의 짝이 유무를 파악하여 flag 변수를 통해 표현하였다. 초기에 flag를 1로 초기화 시켰다.

이후 만약에 괄호의 짝이 맞지 않는다면 flag를 0으로 변경하여 NO를 출력할수 있도록 하였다.

Flag의 값이 초기와 같이 1이면 짝이 맞는 경우이므로 YES를 출력하였다.

마지막에는 다시 falg를 1로 초기화 시켜 다음 반복문 순서에서 오류가 나지 않도록 유의 하였다.

또한 괄호를 사용자로부터 입력을 받을 때 gets은 메모리의 침범과 같은 오류가 발생할수 있어 fgets을 통해 입력을 받는다. 실제로 gets를 사용하여 프로그램 실행에 오류가 발생이 되어 고생하였다.

이때 fgets은 개행문자까지 포함된다. fgets를 사용한 이유는 get은 메모리의 침범으로 프로그램에 오류를 발생을 시킨다. 되도록이면 메모리의 주소를 지정해주는 fgets사용하였다.

백준 채점결과 성공처리 되었다.

**(2) 프로젝트를 위한 GIthub 저장소의 웹 주소**

<https://github.com/KANGKYEONGSEOK/Data-Structures-and-Algorithms.git>