Stack/stack.c

```c
/*-----------------------------------------------------------------------------------------------
File Name: stack.c
Programmed by: Hangyeol Lee
Affiliation: Chungbuk University
Functions: push(), pop(), print() in stack
Copyright (c) 2025 Hangyeol Lee. All rights reserved.
-----------------------------------------------------------------------------------------------*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define STACK_OVERFLOW -1
#define STACK_UNDERFLOW -2
#define OK 0
#define STACK_MAXSIZE 3
#define ZERO_SIZE -1

#define PUSH 1
#define POP 2
#define PRINT 3
#define QUIT 4
#define INITIALIZATION 5

typedef struct Node // the Node structure of Stack
{
    int number;    // 학번
    char name[10]; // 이름
} STACK_NODE;

STACK_NODE Stack[STACK_MAXSIZE]; // Stack
int top = -1;                    // the top pointer's definition and initilization in Stack

/*-----------------------------------------------------------------------------------------------
 Function: 스택 top인덱스 리턴
 Interface: pop()
 Paramete: None
 return: int = index of a node to be poped
-----------------------------------------------------------------------------------------------*/
int pop()
{
    if (top == ZERO_SIZE) // 스택의 사이즈가 0이면 언더플로우
        return STACK_UNDERFLOW;
    else
        return top; // top 리턴 즉, 맨 위의 스택 index 리턴
}

/*-----------------------------------------------------------------------------------------------
 Function: 스택에 값을 넣음
 Interface: push()
 Parameter: int number = a student number to push
            char name[] = a name of student to push
 return: OK = if the pushing is complete
        STACK_OVERFLOW = if the stack is full
-----------------------------------------------------------------------------------------------*/
int push(int number, char name[])
{
    if (top == STACK_MAXSIZE - 1) // 스택이 꽉찼으면 오버플로우
        return STACK_OVERFLOW;
    else
    {
        top++; // top + 1
        Stack[top].number = number;
        strcpy(Stack[top].name, name); // push할 변수 초기화
        return OK;
    }
}

/*---------------------------------------------------------------------------------------------
 Function: 스택의 값들을 출력
 Interface: print_stach()
 Parameter: None
 return: void
-----------------------------------------------------------------------------------------*/
void print_stack()
{
    if (top == ZERO_SIZE) // 스택이 비어있을때
        printf("Stack is Empty!\n");
    else
    {
        for (int i = 0; i <= top; i++)
        {
            printf("%d\t%s\n", Stack[i].number, Stack[i].name); // 학번 먼저 출력
        }
    }
}

/*---------------------------------------------------------------------------------------
 Function: 스택 초기화
 Interface: initialize_stack()
 Parameter: None
 return: void
-----------------------------------------------------------------------------------------*/
void initialize_stack()
{
    for (int i = 0; i < STACK_MAXSIZE; i++)
    {
        Stack[i].number = 0;
        strcpy(Stack[i].name, "");
    }
    top = -1;
}

/* ---------------------------------------------------------------------------------------
 Stack이 잘 구현됐는지 확인하는 인터페이스
 command에 숫자를 입력받아서 원하는 명령 실행
-----------------------------------------------------------------------------------------*/
int main()
{
```

```c
    int command;
    do
    {
        printf("----------------------------------------------------------------\n");
        printf("                     stack                         \n");
        printf("----------------------------------------------------------------\n");
        printf(" Push            = 1           Pop           = 2 \n");
        printf(" Print           = 3           Quit          = 4 \n");
        printf(" Initialization  = 5 \n");
        printf("----------------------------------------------------------------\n");

        printf("Command = ");
        scanf("%d", &command);

        switch (command)
        {
        case PUSH: // 1
        {
            int number = 0;
            char name[10];
            int ret = 0;

            printf("Input number = ");
            ret = scanf("%d", &number);
            if (ret != 1)
            {
                printf("Wrong Input!\n");
                while (getchar() != '\n')
                    ; // 입력 버퍼 비우기
                break;
            }

            printf("Input name = ");
            ret = scanf("%s", name);
            if (ret != 1)
            {
                printf("Wrong Input!\n");
                while (getchar() != '\n')
                    ; // 입력 버퍼 비우기
                break;
            }

            if (push(number, name) == STACK_OVERFLOW)
            {
                printf("Stack OverFlow!\n");
            }
            break;
        }
        case POP: // 2
        {
            int result = pop(); // 리턴 받은 인덱스 result에 저장
            if (result == STACK_UNDERFLOW)
            {
                printf("Stack UnderFlow!\n");
            }
            else
            {
                printf("%d\n", Stack[result].number);
                printf("%s\n", Stack[result].name); // 인덱스에 해당하는 값 출력
                top--;                              // top - 1
            }
            break;
        }
        case PRINT: // 3
            print_stack();
            break;
        case INITIALIZATION: // 5
            initialize_stack();
        case QUIT: // 4
            break;
        default:
            printf("\n       >>>>>   Concentration!!   <<<<<      \n");
            break;
        }
```

```c
    } while (command != QUIT);
}ㅂ
```