5645640 제갈건 0504

1. 실행코드

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include<time.h>

#define MAX\_STACK\_SIZE 2 //임시로 최대 크기 2로 수정

typedef struct {

int top;// top이자 현재 스택의 크기

int\* array; // array는 실질적인 스택을 구현한다.

int capacity; //스택의 전체 크기, size

}ArrayStack;

ArrayStack\* createStack() {

ArrayStack\* S = (ArrayStack\*)malloc(sizeof(ArrayStack));

if (!S) {

return NULL;

}

S->capacity = MAX\_STACK\_SIZE;

S->top = -1;

S->array = (int\*)malloc(S->capacity \* sizeof(int));

if (!S->array) {

return NULL;

}

return S;

}

int is\_empty(ArrayStack\* s) {

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

}

int is\_full(ArrayStack\* s) {

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

}

void push(ArrayStack\* s, int item) {

if (is\_full(s)) {

s->capacity \*= 2;

s->array = (int\*)realloc(s->array, s->capacity \* sizeof(int));

printf("Let's extend the array by double\n");

printf("size of array: %d\n", s->capacity);

}

printf("push: %d\n", item);

s->array[++(s->top)] = item;

}

int pop(ArrayStack\* s) {

if (is\_empty(s)) {

fprintf(stderr, "스택 공백 에러\n");

return -1;

}

else return s->array[(s->top)--]; // top의 위치만 이동 한 것이고 배열 안의 데이터는 그대로 남아있음

}

// 피크함수

int peek(ArrayStack\* S)

{

if (is\_empty(S)) {

fprintf(stderr, "Stack is Empty\n");

exit(1);

}

else return S->array[S->top]; //top에 있는 값 출력

}

void deleteStack(ArrayStack\* S) {

if (S) {

if (S->array) free(S->array);

free(S);

}

}

int Stacksize(ArrayStack\* S) {

if (is\_empty(S)) {

fprintf(stderr, "스택 공백 에러\n");

return -1;

}

return S->top + 1;

}

int main(void) {

ArrayStack\* mys;

mys = createStack();

for (int i = 1; i <= 10; i++) {

push(mys, i);

}

printf("TOP에 있는 데이터: %d\n", peek(mys));

printf("Stack Size: %d\n", Stacksize(mys));

/\*

for (int j = 1; j <= 10; j++) {

int tmp = pop(mys);

printf("POP: %d\n", tmp);

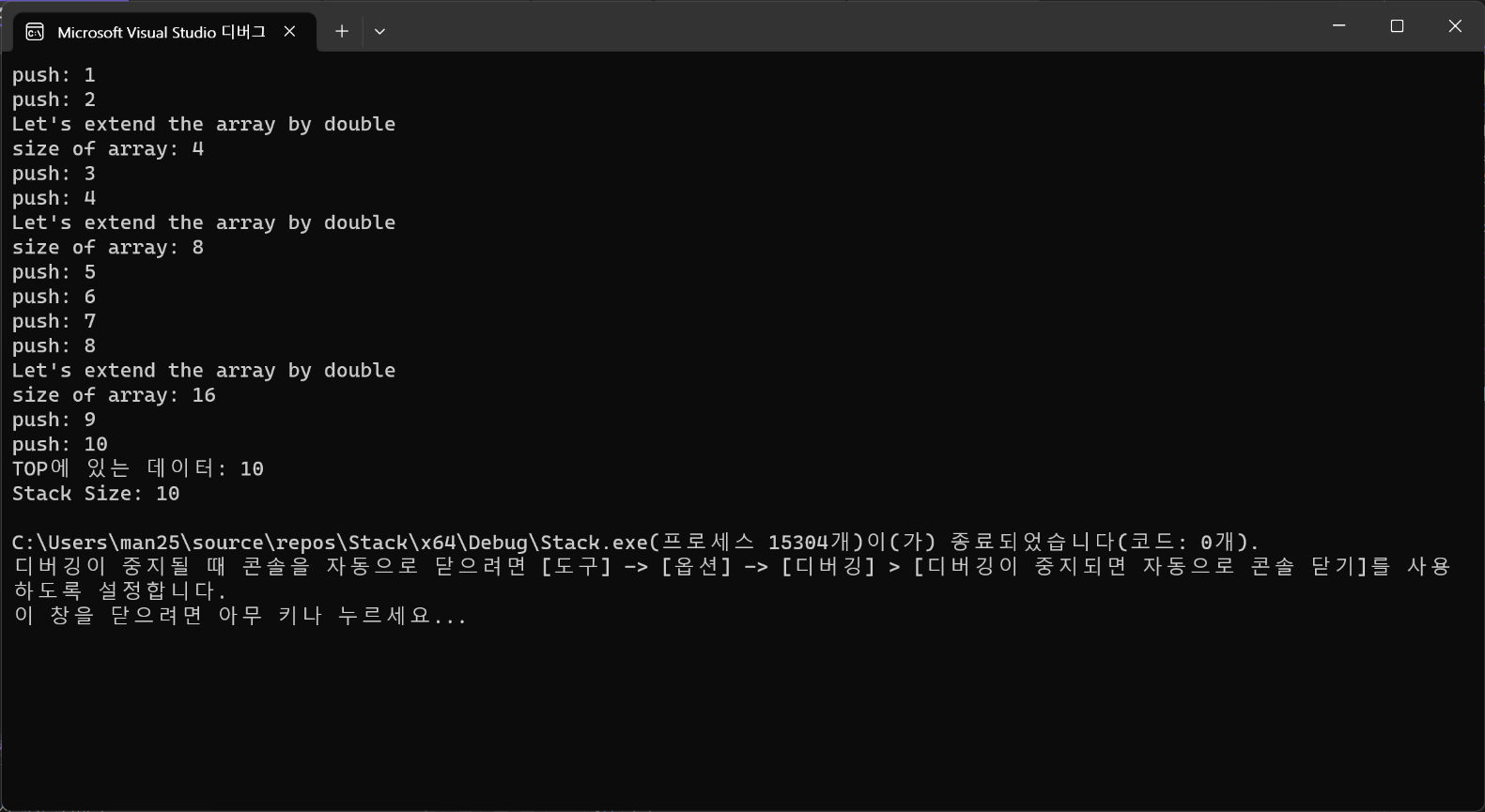
}\*/

deleteStack(mys);

return 0;

}

1. 실행화면



1. 고찰

push에서 배열이 full이 되었을 때 배열의 크기를 확장하는 것을 구현하는 것에 애를 먹었습니다. Realloc()함수를 몰라 배열을 복사하고 다시 동적 배열을 생성하여 복사하는 방식으로 구현하였습니다. 하지만 강의를 듣고 Realloc()를 이용하여 구현하였습니다.