1. 소스 코드

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define MAX\_STACK\_SIZE 10 // 스택의 최대 크기

typedef int element; // 데이터의 자료형

element stack[MAX\_STACK\_SIZE]; // 1차원 배열

int top = -1;

// 공백 상태 검출 함수

int is\_empty()

{

return (top == -1);

}

// 포화 상태 검출 함수

int is\_full()

{

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

}

// 삽입 함수

void push(element item)

{

if (is\_full()) {

fprintf(stderr, "Stack Full\n");

return NULL;

}

else stack[++top] = item;

}

// 삭제 함수

element pop()

{

if (is\_empty()) {

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

return NULL;

}

else return stack[top--];

}

// 피크 함수

element peek()

{

if (is\_empty()) {

fprintf(stderr, "스택 공백 에러\n");

return;

}

else return stack[top];

}

int main(void)

{

srand(time(NULL));

int rand\_num = rand() % 100 + 1;

for (int i = 0; i < 30; i++)

{

printf("%d 번 ", i);

rand\_num = rand() % 100 + 1;

if (rand\_num % 2 == 0)

{

printf("push %d\n", rand\_num);

push(rand\_num);

}

else {

printf("pop %d\n", pop());

}

}

}

2. 실행 화면

텍스트이(가) 표시된 사진

자동 생성된 설명

3. 느낀점

난수를 생성해서 직접 push와 pop을 하니 스택이 어떻게 돌아가는지가 보여 이해가 더 잘되었다.

Top을 잘 관리하여야 스택을 오류가 없도록 하는데 중요하다고 생각하였다. 그리고 그를 위해 is\_empty와 is\_full을 잘 만들어 둔뒤 활용해야 하는구나를 느끼게 되었다.