Data Structure

Siwon Yun

March 27, 2025

Contents

Theme 1
Stack

DEFINITION 1.1 (스택). 쌓아놓은 더미를 스택[stack]이라고 한다.

책이 쌓여있는 것을 생각하자. 제일 아래에 깔려있는 책을 ㄱ보기 위해서는 위에 있는 모든 책을 걷어내야 한다.

1.0.1 LIFO

이러한 스택의 특징을 후입선출[LIFO, Last In First Out]이라고 한다. 즉, 가장 마지막에 들어간 데이터가 가장 먼저 나온다.

1.0.2 스택의 구조

스택의 데이터는 요소[element]라고 부른다. 가장 위, 아래에 있는 요소는 각각 스택의 상단(top), 하단(bottom)이라고 부른다.

EXAMPLE 1.1. 재귀함수 또는 함수 안에서 호출되는 함수는 스택의 예시이다. 함수 안에서 새로운 함수가 호출되면 호출된 함수가 끝나기까지 다른 명령어가 실행되지 아니한다. 나중에 호출된 함수가 가장 먼저 실행되는 것이 스택과 같다.

1.1 ADT

스택의 간단한 ADT에 대하여 알아본다. 스택 객체는 0개 이상의 원소를 가지는 유한 선형 리스트이다. 유한 선형 리스트이기에 — 동적으로 스택 최대 크기를 끊임없이 늘려준다고 한들, 최대 크기가 존재하나 해당 크기를 늘려주는 것이니 유한하다고 표현한다. — 스택은 최대 크기를 가진다. 스택은 기본적으로 자료를 저장하는 도구이기 때문에 자료를 추가하고 조회하는 연산을 가진다. 주요 연산은 다음과 같다:

- `create(size)`: 최대 크기가 size인 스택을 생성한다.
- `isEmpty(s)`: 스택이 비어있는지 확인한다.

- `isFull(s)`: 스택이 가득 차있는지 확인한다.
- `push(s, item)`: 스택에 원소를 추가한다.
- `pop(s)`: 스택에서 원소를 제거하고 제거한 값을 반환한다.
- `peek(s)`: 스택의 상단에 있는 원소를 반환한다.(제거하지 않음)

push, pop, peek 모두 명령을 실행하기 사전에 스택이 가득 차있는지 고려해야 한다.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

#define MAX_STACK_SIZE 100
typedef int element;
element stack[MAX_STACK_SIZE];
int top = -1;

bool is_empty() {
  return (top == -1);
}

bool is_full() {
  return (top == (MAX_STACK_SIZE - 1));
}

void push(element item) {
  if (is_full()) {
    fprintf(stderr, "스택 포화 에러\n");
    return;
  }
  else stack[++top] = item;
}

element pop() {
  if (is_empty()) {
    fprintf(stderr, "스택 공백 에러\n");
    exit(1);
  }
  else return stack[top--];
}

int main() {
  push(1);
  push(2);
  push(3);
```

```
  printf("%d\n", pop());
  printf("%d\n", pop());
  printf("%d\n", pop());
  return 0;
}
```

비단, 이와같이 스택을 구현하게 되면 스택은 변수에 종속적이다. 각 함수가 하나의 스택만을 위한 것이기 때문에 좋은 코드가 아니다.

ToDo... 구조체 배열을 활용한 스택, 동적 스택

INDEX