Stack

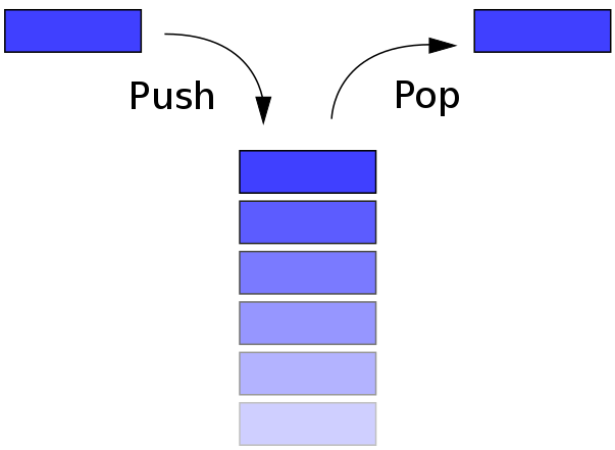
스택

목차

(1). 스택

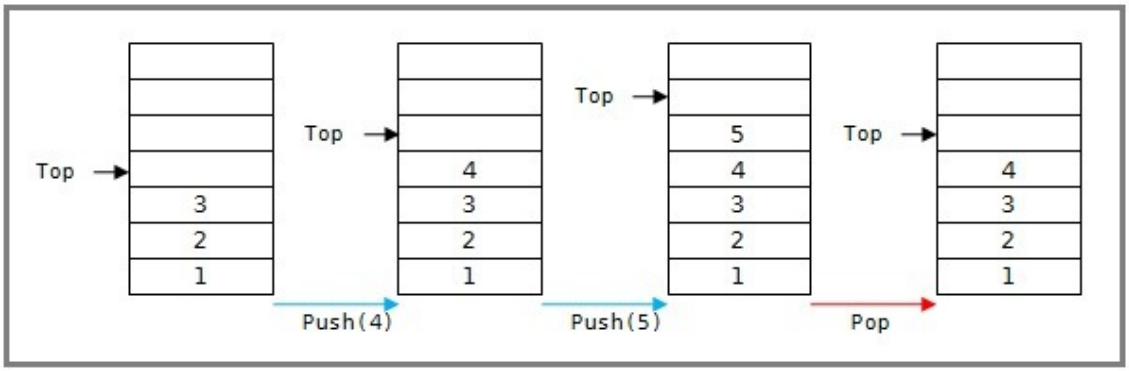
✓ 스택(Stack)이란 쌓아 올린다는 의미이다. 따라서 스택 자료구조라는 것은 접시를 쌓듯이 자료를 차곡차곡 쌓아 올린 형태의 구조를 말한다.

✓ 스택은 한 쪽 끝에서만 자료를 넣거나 뺄 수 있는 선형 구조(LIFO - Last In First Out)으로 되어 있다. 자료를 넣는 것을 '밀어넣는다' 하여 Push 라고 하고 반대로 넣어둔 자료를 꺼내는 것을 Pop 이라고 하는데, 이때 꺼내지는 자료는 가장 최근에 보관한 자료부터 나오게 된다. 이처럼 나중에 넣은 값이 먼저 나오는 것을 LIFO 구조라고 한다.



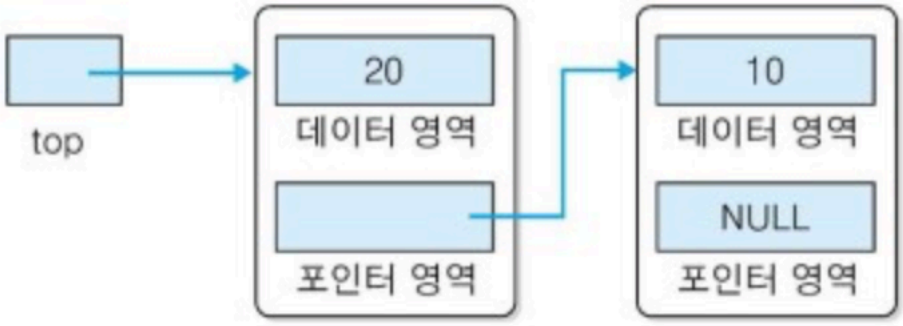
(2). 순차 자료구조 방식을 이용한 스택의 구현

✓ 순차 자료구조 방식을 이용하여 스택을 구현하기 위해서 배영을 사용한다. stack[n]에서 n 은 배열의 크기로서 배열 데이터의 개수를 나타내는데, 이것은 스택의 크기가 되고, 배열의 인덱스는 스택에 데이터가 쌓이는 순서를 표현한다. 스택의 첫 번째 데이터는 stack[0]에 저장되고, 스택의 두 번째 데이터는 stack[1]에 저장되고, 스택의 i 번째 데이터는 stack[i-1]에 저장된다. Top 은 마지막 데이터의 인덱스를 저장하는 변수를 사용하고 공백 스택에서의 top 은 -1 값을 갖는다.



(3). 연결 자료구조 방식을 이용한 스택의 구현

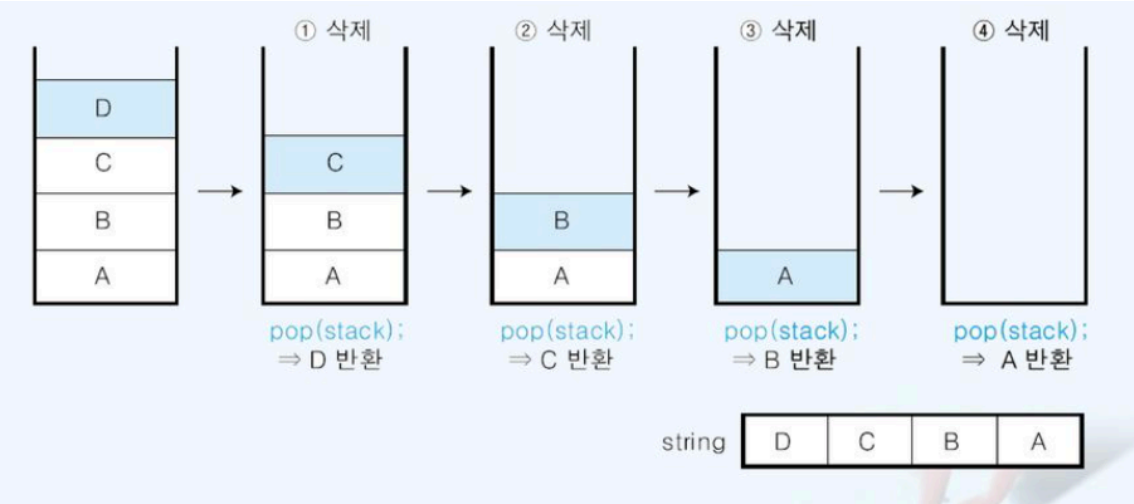
✓ 연결 자료구조 방식을 이용하여 스택을 구현하면 스택의 데이터는 연결 리스트의 노드가 된다. 스택에 데이터를 삽입할 때마다 연결 리스트에 노드를 하나씩 할당한다. 스택 데이터의 순서는 연결 리스트 노드의 링크를 사용하여 표현하고, 스택의 top 은 참조 변수 top 을 사용한다.



[그림 7-26] 연결 리스트로 구현한 스택 구조

(4). 역순 문자열 만들기

✓ 스택의 LIFO 설질을 이용하여 문자열에 대한 역순 문자열을 간단히 만들 수 있다. 문자열을 처음부터 순서대로 스택에 Push 한다. 다음, 스택에 있는 원소들을 공백 스택이 될 때까지 삭제하면서 반환된 문자를 나열하면 원래 문자열의 역순 문자열이 된다.



(5). 시스템 스택

✓ 따라서 함수의 호출과 복귀 순서는 스택의 LIFO 구조를 응용하여 관리하는데, 이때 사용하는 스택을 시스템 스택이라 한다.

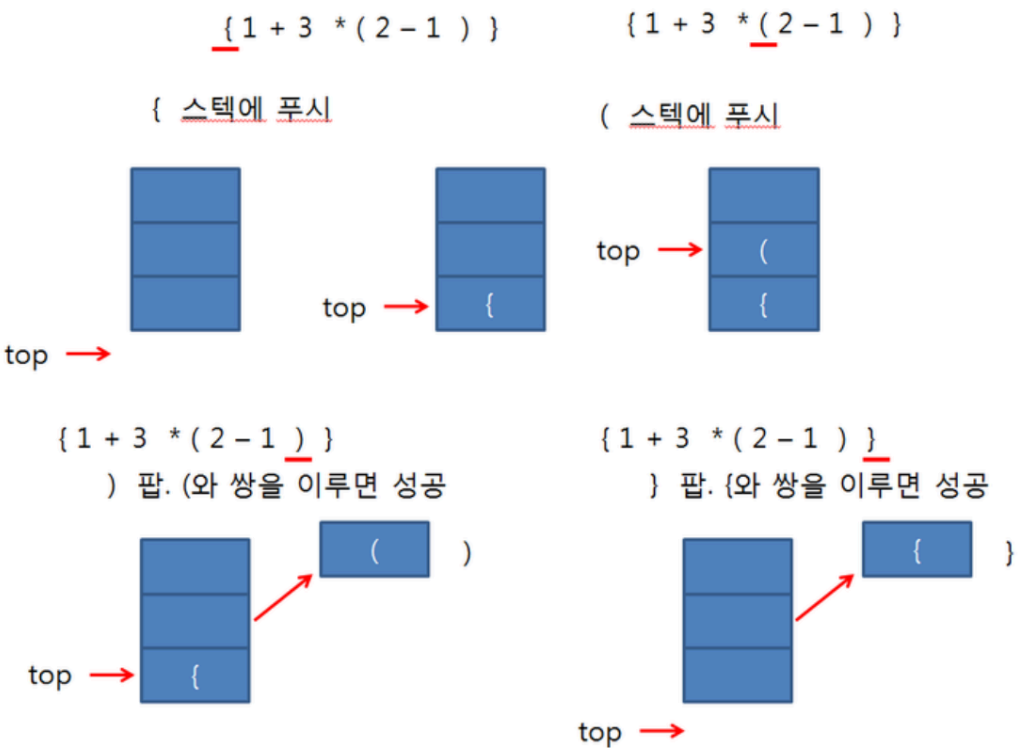
① 주 프로그램인 main( ) 함수 실행 시작

② main( ) 함수의 method1( ) 함수 호출

③ method1( ) 함수 호출 중에 method2( ) 함수 호출
④ method2( ) 함수를 실행
⑤ method2( ) 함수 실행을 마치고 호출 위치로 복귀
⑥ method1( ) 함수로 복귀하여 나머지 부분 실행
⑦ method1( ) 함수 실행을 마치고 호출 위치로 복귀
⑧ main( ) 함수로 복귀하여 나머지 부분을 실행하고 종료

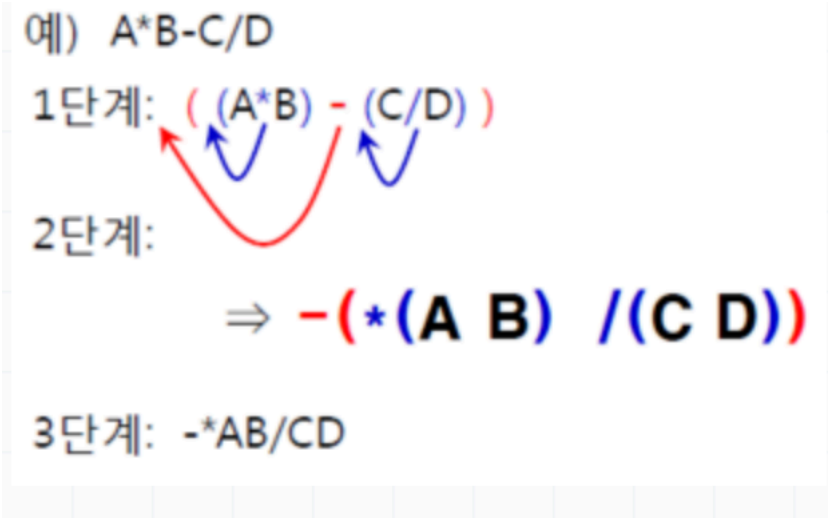
(6). 수식의 괄호 검사

✓ 괄호가 포함된 수식에서 괄호의 쌍이 제대로 사용되었는지를 검사하기 위해서 스택을 사용할 수 있다. 수식을 읽으면서 왼쪽 괄호를 만나면 스택에 Push 하고, 오른쪽 괄호를 만나면 스택을 Pop 한다. 현재 검사 중인 오른쪽 괄호와 Pop 하여 구한 왼쪽 괄호가 같은 종류의 괄호면 괄호의 쌍이 맞는 것이고, 수식의 처리가 모두 끝났을 때 스택이 공백 스택이 되면 왼쪽 괄호와 오른쪽 괄호의 개수가 맞는 것이다.

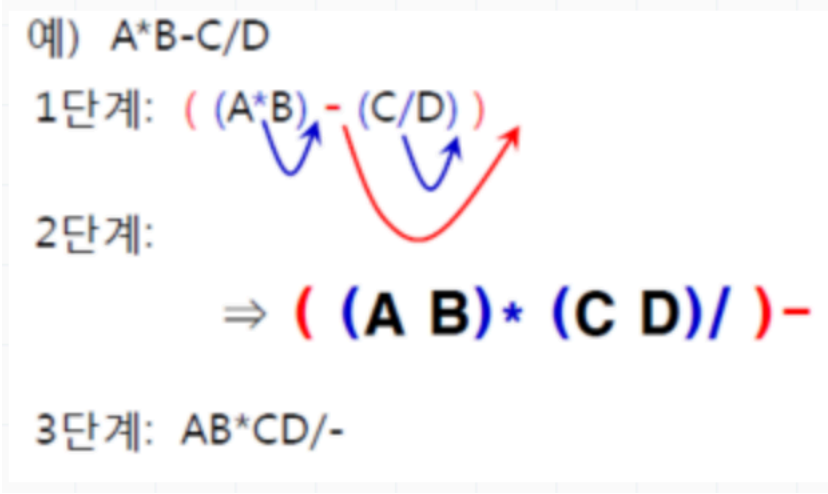


(7). 수식의 후위 표기법 변환

- ✓ **수식의 표기법 종류**
  - ① 전위 표기법: 연산자를 앞에 표기하고 그 다음에 피연산자를 표기하는 방법 Ex) +AB
  - ② 중위 표기법: 연산자를 피연산자의 가운데에 표기하는 방법 Ex) A+B
  - ③ 후위 표기법: 연산자를 피연산자 뒤에 표기하는 방법 Ex) AB+
- ✓ **중위 표기법 -> 전위 표기법**
  - ① 수식의 각 연산자에 대해서 우선순위에 따라 괄호를 사용하여 다시 표현한다.
  - ② 각 연산자를 그에 대응하는 왼쪽 괄호의 앞으로 이동시킨다.
  - ③ 괄호를 제거한다.



- ✓ **중위 표기법 -> 후위 표기법**
  - ① 수식의 각 연산자에 대해서 우선순위에 따라 괄호를 사용하여 다시 표현한다.
  - ② 각 연산자를 그에 대응하는 오른쪽 괄호의 뒤로 이동시킨다.
  - ③ 괄호를 제거한다.



✓ **스택을 이용하여 수식을 후위 표기식으로 변환하는 방법**

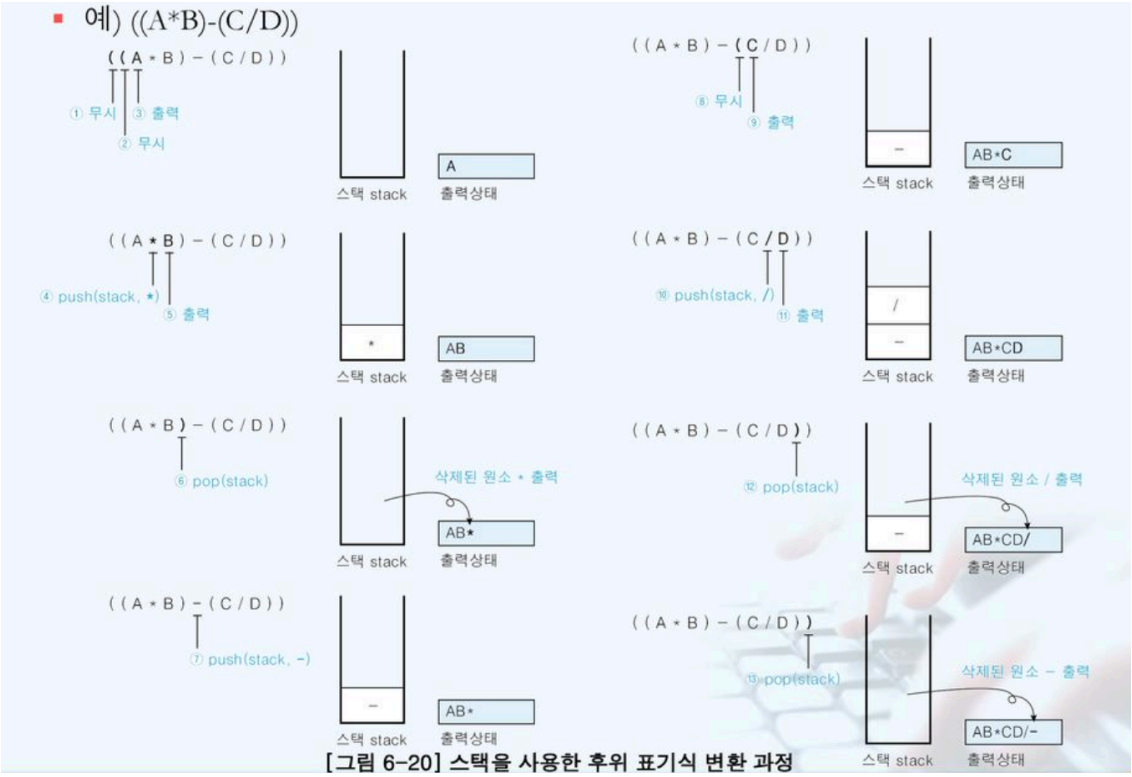
① 왼쪽 괄호를 만나면 무시하고 다음 문자를 읽는다.

② 피연산자를 만나면 출력한다.

③ 연산자를 만나면 스택에 삽입한다.

④ 오른쪽 괄호를 만나면 스택을 pop 하여 출력한다.

⑤ 수식이 끝나면 스택이 공백이 될 때까지 pop 하여 출력한다.



[그림 6-20] 스택을 사용한 후위 표기식 변환 과정

(8). 후위 표기 수식의 연산

✓ 컴퓨터 내부에서 후위 표기법의 수식을 연산할 때에도 스택을 사용할 수 있다.