chapter3. 재귀

- Hello Coding 그림으로 개념을 이해하는 알고리즘 -

TABLE OF CONTENTS

1. 재귀함수

재귀 함수는 자기 자신을 호출하는 함수입니다.

그래서 실수로 무한 반복을 하는 함수를 만들기 쉽습니다.

다음과 같이 카운트다운을 하는 함수를 만들어봅시다.

3…2…1

```
def countdown(i):
    print(i)
    countdown(i-1)
```

위 코드는 끝없이 실행되는 문제가 있습니다.

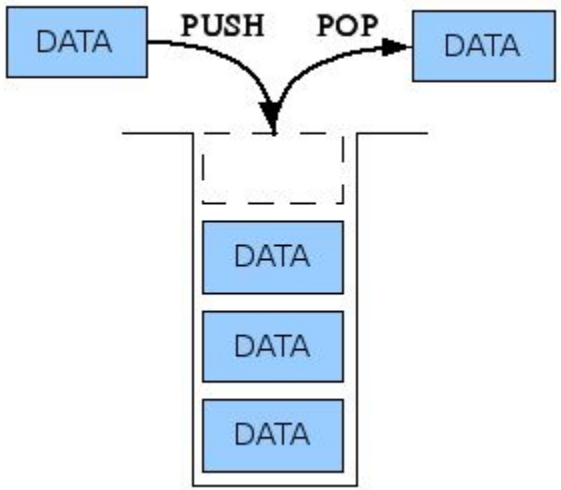
재귀 함수를 만들 때는 언제 멈출지 알려줘야 합니다. 그래서 모든 재귀 함수는 기본단계와 재귀단계라는 두 부분으로 나누어져 있습니다.

재귀 단계는 함수가 자기 자신을 호출하는 부분이며 기본 단계는 무한 반복으로 빠져들지 않게 하는 부분입니다.

```
def  countdown(i):
  print(i)
  #기본 단계
  if i <= 1:
    return
  #재귀 단계
  else:
    countdown(i-1)

countdown(5)
```

2. 스택



스택은 가장 나중에 들어온 자료가 가장 먼저 처리되는
LIFO(Last-In-First-Out) 자료구조입니다.
구멍이 하나밖에 없는 병이라고 생각하면 이해하기 쉽습니다.
스택은 아주 단순한 자료구조입니다.
push(삽입) pop(떼어내서 읽기) 두가지 밖에 없습니다.

3. 호출스택

```
def 인사(name):
    print("안녕" + name)
    인사2(name)
    인사3()

def 인사2(name):
    print("잘지내니?" + name)

def 인사3():
    print("잘가")

인사("세종대왕")
```

인사(“세종대왕”)을 명령했다고 가정해봅시다.

그러면 우선 컴퓨터는 이 함수 호출을 위해 메모리 상자를 하나 할당합니다.

메모리 상자에 name이라는 변수 값이 “세종대왕” 이라는 정보를 저장합니다.

print(“안녕” + name)을 프린트 한 후, 인사2 함수를 호출합니다.

인사2 함수를 호출할때도 똑같이 메모리상자 하나를 할당하고 값을 저장합니다.

그럼 현재 인사 메모리상자 위에 인사2 메모리상자가 올려져있겠죠?

```
def 인사(name):
    print("안녕" + name)
    인사2(name)
    인사3()

def 인사2(name):
    print("잘지내니?" + name)

def 인사3():
    print("잘가")

인사("세종대왕")
```

이제 “잘지내니” + name을 프린트 한 후 인사2 함수는 반한됩니다. 함수가 반환되면 가장 위에 있는 상자는 pop연산으로 인해 없어집니다.

이제 스택에서 가장 위에 있는 상자는 인사 함수가 되었습니다. 즉 인사 함수로 되돌아 온 것입니다.

인사2를 호출했을 때 인사 함수는 완전히 실행되지 않은 상태입니다.

어떤 함수를 호출하여 완전히 실행을 완료하기 전이라도 그 함수를 잠시 멈추고 다른 함수를 호출할 수 있습니다.

중지된 함수의 변수 값들은 모두 메모리에 저장되어 있습니다.

여러개의 함수를 호출하면서 함수에 사용되는 변수를 저장하는 스택을 호출스택이라고 합니다.

〈요약〉

재귀는 함수가 스스로 호출하는 것을 말한다.

모든 재귀 함수는 기본 단계와 재귀 단계라는 두 부분으로 나누어져 있다.

스택에는 푸시와 팝이라는 두가지 연산이 있다.

모든 함수 호출은 호출 스택을 사용한다.

호출 스택은 너무 커져서 메모리를 엄청나게 소비할수도 있다.