Java의 가변인자 메소드

```
public static void testVarargs(String...strings)
{
      System.out.println(strings.length + ” “);
      for(String str : strings) {
            System.out.print(str + "\t");
      }
      System.out.println("\n");
}
```

호출은 어떻게 할까?

```
testVarargs("MS", "Apple", "IBM");
testVarargs("Red", "Blue", "Green", "White",
"Orange");
```

```
3 MS Apple IBM
5 Red Blue Green White Orange
```

Static variables vs. Recursion

- 지역변수가 static 변수인 경우
  - 모든 언어에서 recursion이 지원 안 되는 것은 아니다.. (eg. C)
- Recursion을 허용하는 경우
  - 모든 언어가 static 지역변수를 허용하지 않는 것은 아니다. (eg. C)

그렇다면 무슨 얘기?

related concept : reentrancy, thread-safety, idempotency … ➔ "point는 recursion 안에서 static 변수 쓰지 마라"

참고) 원론: re-entrant, recursive-safe , idempotent

- http://lucyjava.blogspot.kr/2007/06/re-entrant-recursive-and-idempotent.html
- A method in stack is re-entrant allowing multiple concurrent invocations that do not interfere with each other. To be reentrant, a function must hold no static data, must not return a pointer to static data, must work only on the data provided to it by the caller, and must not call non-reentrant functions. http://en.wikipedia.org/wiki/Reentrant
- A function is recursive if it calls itself. Given enough stack space, recursive method calls are perfectly valid in Java though it is tough to debug. Recursive functions are useful in removing iterations from many sorts of algorithms. All recursive –safe functions are re-entrant but not all re-entrant functions are recursive.
- Idempotent methods are methods that repeated calls to the same method with the same arguments yield same results. For example clustered EJBs, which are written with idempotent methods, can automatically recover from a server failure as long as it can reach another server.

Call by Result

- call by value와 call by result는 전달 방향만 다를 뿐 방법이 비슷하다.
- 하지만, input parameter 전달은 call-by-value 가 좋다면서, ouput parameter 전달은 call by result로 할 때 문제가 생긴다.
- 왜?
  - (8-2 p10 참고)

  생각해보기…