Loop invariant (루프 증명)

loop invariant 란 루프를 돌 동안 유지되어야 하는 statement 들이다. Loop invariant 를 체크해 봄으로써 루프가 완성적인지 아닌지 체크해 볼 수 있으므로,알고리즘을 짤 때, 특히 루프를 사용할 때 루프 점검에 아주 탁월한 방법이다. Loop invariant 에는 세 가지의 요구 조건이 있다.

1. Initialization : (precondition)
루프에 처음 들어갈 때 변수의 변화 및 statement(조건)의 변화가 맞는지 판단해서, 루프에 정확하게 들어가는지 확인한다.

2. Maintenance :
루프가 시작하기 전부터 해서, 다음 루프로 넘어가기 전까지 만족해야 되는 조건으로 이 조건이 알고리즘의 목적에 부합하는지를 판단한다.

3. Termination : (postcondition)
루프가 끝났을때, 우리는 사용가능하고 유용한 결과를 얻어야 하며, 이것이 문제해결(알고리즘의 목표)에 도움이 되는 결과가 나오는지 확인한다.
위 세 가지를 모두 만족했을 때, 이 루프는 제대로 짜여진 것이며, 이상이 없다라고 한다.
(Optimization (최적화) 와는 상관없음, 제대로 짜여졌는지만 확인할 수 있다)
간단하게 Insertion sort(삽입정렬)로 예를 들어보자.

Pseudocode of Insertion-Sort(A)

```
Insertion-Sort(A)
1     for j<-2 to length[A]
2          do key <- A[j]
3               i <- j-1
4               while i>0 and A[i] >key
5                    do A[i+1] <- A[j]
6                    i <- i-1
7               A[i+1] <- key
```

<Loop-invariant> : A[1...j-1] 이 소팅되는 과정

Initialization : j=2 에 대해서, A[1...j-1] = A[1] 이 정렬된다.
Maintenance : j가 증가하고 그에따라 A[1...j-1]가 정렬된다.
Termination : j = length[A] + 1 일때 루프는 끝이나며, 루프가 끝이 났을때는 A[1...j-1] 은 정렬이 끝나있다.
이런식으로 조건과 결과에 대해서 모두 확인이 끝나면 이 루프는 (for 문) 제대로 짜여졌다. 라고 한다.