**LIS**

1. 길이 구하기 :

<https://namu.wiki/w/%EC%B5%9C%EC%9E%A5%20%EC%A6%9D%EA%B0%80%20%EB%B6%80%EB%B6%84%20%EC%88%98%EC%97%B4>

1. 수열의 원소를 앞에서부터 검사
2. 각 원소별로 subSequence를 대상으로 lowerBound 실행.
3. lowerBound로 나온 위치 x에 현재 원소를 집어넣음
4. 길이 배열엔 x- subSequence로 길이를 구하여 집어넣음.
5. 만약 위치가 subSequence의 끝이면, subSequence의 길이를 1 증가.
6. 현재 길이가 longest보다 길면 저장과 동시에 longestIndex에 현재 index저장
7. 부분수열 구하기 : LIS의 길이가 저장된 원소 x로부터 가장 가까운 길이가 1씩 작아지는 원소들이, 가장 x로부터 가장 가까운 LIS이다.
   1. 현재 longest엔 LIS의 길이가 저장되어있고, longestIndex엔 가장 뒤에있는 LIS의 가장 뒤 원소의 인덱스가 저장되어있음
   2. longest는 따로 temp에 저장.
   3. longestIndex부터 1까지 이동하며 temp와 같은 것이 있는지 확인
   4. temp와 같은 것이 발견될때마다 temp를 1씩 줄이고, lis에 그 원소를 저장
   5. temp가 0이되면 종료.
8. 코드

int lengthOfSequence, sequence[MAX\_LENGTH], length[MAX\_LENGTH];

int subSequence[MAX\_LENGTH], longest, lis[MAX\_LENGTH];

void LIS() {

subSequence[0] = -2000000000;

int lengthOfSub = 1;

int longestIndex = 0;

for (int i = 1; i <= lengthOfSequence; i++) {

int\* lb = lower\_bound(subSequence, subSequence + lengthOfSub, sequence[i]);

\*lb = sequence[i];

length[i] = lb - subSequence;

if (lb - subSequence == lengthOfSub)

lengthOfSub++;

if (length[i] > longest) {

longest = length[i];

longestIndex = i;

}

}

int temp = longest;

for (int i = longestIndex; i >= 1; i--) {

if (temp == 0)

break;

if (length[i] == temp) {

lis[temp] = sequence[i];

temp--;

}

}

}