Try 2 (구글링)

nums = [10, 20, 10, 30, 20, 50]
1 2 1 3 2 4

⇒ 각 숫자 중 앞쪽 수중 작은 숫자의 개수 = 배열의 길이 (겹치지X, 개념+1)

```python
import sys

N = int(input())
numbers = list(map(int, sys.stdin.readline().split()))
LIS = [1] * N
for i in range(1, N):
    for j in range(i):
        if numbers[i] > numbers[j]:
            LIS[i] = max(LIS[i], LIS[j]+1)

print(max(LIS))
```

LIS = [1, 2, 1, 3 ...]

i=1 for j in range(1):

20 > 10 ⇒ LIS[1] = max(1, 1+1)

i=2 for j in range(2):

10 > 10 (X)

10 > 20 (X)

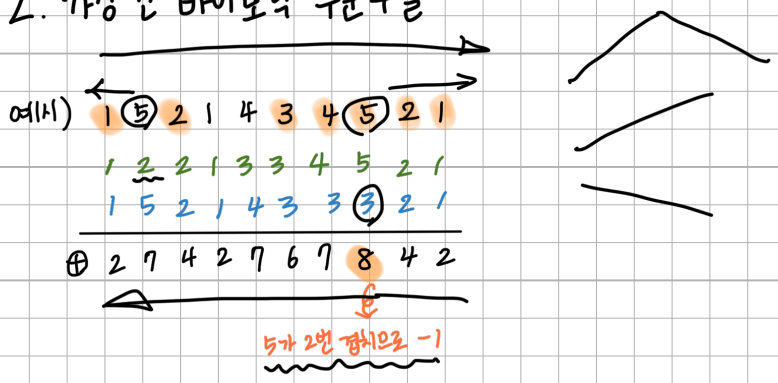
i=3 for j in range(3):

30 > 10 ⇒ LIS[3] = max (1, 1+1)

30 > 20 ⇒ = max (1, 2+1)

30 > 10 ⇒ = max (1, 1+1)

2. 가장 긴 바이토닉 부분수열

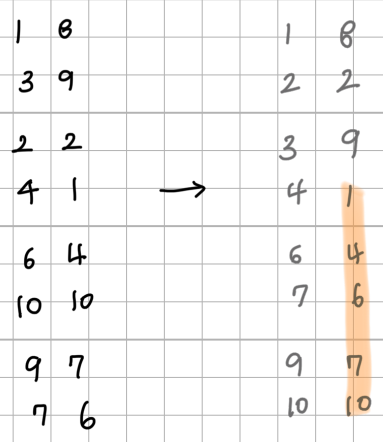



```python
import sys

N = int(input())
numbers = list(map(int, sys.stdin.readline().split()))
increase = [1] * N
decrease = [1] * N
sums = [0] * N

for i in range(N):
    for j in range(i):
        if numbers[i] > numbers[j]:
            increase[i] = max(increase[i], increase[j]+1)
for i in range(N-1, -1, -1):
    for j in range(N-1, i, -1):
        if numbers[i] > numbers[j]:
            decrease[i] = max(decrease[i], decrease[j]+1)
for i in range(N):
    sums[i] = increase[i] + decrease[i]

print(max(sums)-1)
```

3. 전깃줄

① 첫번째 정보대의 전깃줄 연결 위치로 오름차순으로 정렬

② 두번째 전봇대 연결위치 중 가장 긴 수열찾기




```python
import sys

N = int(input())
numbers = []
for i in range(N):
  a, b = map(int, sys.stdin.readline().split())
  numbers.append((a,b))

numbers.sort()

increase = [1] * N

for i in range(N):
    for j in range(i):
        if numbers[i][1] > numbers[j][1]:
            increase[i] = max(increase[i], increase[j]+1)

print(N-max(increase))
```

4. LCS

반례

		A	C	A	Y	K	P
	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1
A	0	1	1	2	2	2	2
P	0	1	1	2	2	2	3
C	0	1	2	2	2	2	2
A	0	1	2	3	3	3	3
K	0	1	2	3	3	4	4

다른경우
[i][j-1]
[i-1][j]
중 큰거

같은경우
[i-1][j-1]+1
j==0 이면 그냥 1

		A	A	A	A
	0	0	0	0	0
A	0	1	1	1	1
A	0	1	2	3	4
A	0	1	2	3	5

		A	A	A	A
	0	0	0	0	0
A	0	1	1	1	1
A	0	1	2	2	2
B	0	1	2	2	2

같으면
[i-1][j-1]+1

```python
import sys

A = sys.stdin.readline().strip()
B = sys.stdin.readline().strip()

lcs = [[0] * (len(A)+1) for _ in range(len(B)+1)]

for i in range(1,len(B)+1) :
  for j in range(1,len(A)+1) :
    if B[i-1] == A[j-1] :
      lcs[i][j] = lcs[i-1][j-1] + 1
    else :
      lcs[i][j] = max(lcs[i][j-1], lcs[i-1][j])

print(lcs[-1][-1])

```

5. 연속합

10 -4 3 1 5 6 -25 12 21 -1

dp = [10 10 10]

arr = [10 6 9 10]

처음까지 탐색

dp[i-1] > num[i] → dp[i] = dp[i-1]

a+3 > 3

a-1 > -1

은채는 귀요미

〈 동적계획법 〉

1. 가장 긴 증가하는 부분수열

A = { 10 20 10 30 20 50 }

5 4 3 6 7 9 6 8 9 10 2 3

[5,1]

[5,1] [4,1]

[5,1] [4,1] [3,1]

[6,2] [6,2] [6,2]

[7,3] [7,3] [7,3]

[9,4] [9,4] [9,4]

[9,4] [9,4] [9,4] [6,1]

32 83 21 23 54 69 26 27 35 22

70 7 15 60 85 99 42 -88 69 59

Try 1 (self)

arr = [[num, cnt], [ ] ···]

nums = [10, 20, 10, 30, 20, 50]

기본) arr = [[10, 1], [ ] ···]

↓

20 > 10 → [20, 2] 업데이트

10 < 20 → [10, 1] 다음 idx에 새롭게 추가

} 하문이후 arr[k][1] 중 max 값으로 찾기

문제점) { 2, 3, 7, 5, 6 }

→ 돌아온 답: 2 3 5 6 ⇒ 4

→ 정답 : { 2 3 7 } { 5 6 } ⇒ 3으로 판단