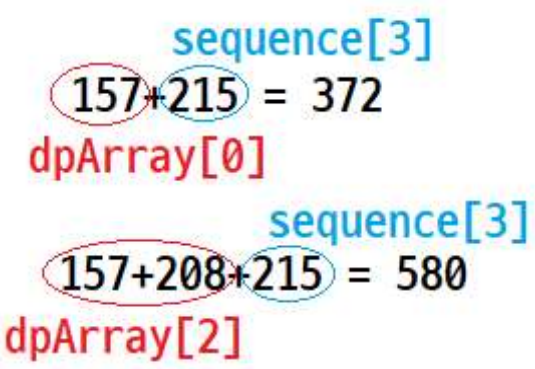
#11055 가장 큰 증가하는 부분 수열

sianke1991

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 157 | 241 | 208 | 215 | 59 | 224 | 96 | 39 | 167 | 144 | 126 | 119 |
| dpArray | 157 | 398 | 365 | 580 | 59 | ??? | | | | | | |

입력된 수열 A에 대해 dpArray[i]는 A[i]를 최대치로 갖는 증가하는 부분 수열의 합의 최대치이다.
예를 들어 dpArray[3]은 A[3]=215를 최대치로 갖는 증가하는 부분 수열의 합의 최대치이다.
A[3]=215를 최대치로 갖는 증가하는 부분 수열은
215
157, 215
208, 215
157, 208, 215
가 있으며 이들 수열의 합은
215 = 215
157+215 = 372
208+215 = 423
157+208+215 = 580
이다. 580이 이들 중 가장 크므로 dpArray[3] = 580이 된다. 이 값은 어떻게 구할 수 있을까. 위의 덧셈에서 밑줄이 들어간 셈은 다음과 같이 해석할 수 있다. (밑줄이 들어가지 않은 식은 밑줄이 들어간 식보다 작은 값이 된다.)




*dpArray를 구성하면서 이전에 구해 놓은 값을 다시 사용할 수 있다.*

즉, dpArray[3]은 dpArray[0]+sequence[3], dpArray[2]+sequence[3] 중 최대치가 된다. (dpArray[1]+sequence[3] 은 고려 대상이 아님에 주의한다. dpArray[1]은 A[1]=241을 최대치로 갖는 증가하는 수열의 합의 최대치이다. 증가하는 부분 수열에 대해 A[1]=241 뒤에 A[3]=215가 올 수 없다.)

dpArray[4]는 A[4]와 동일한 59인데 이는 A[4]가 A[i] (where i ∈ [0, 4]) 중에서 가장 작기 때문이다. 이 경우를 쉽게 처리하기 위해서 dpArray[i]의 초기값은 A[i]와 동일하게 두는 방법을 사용할 수 있다.

dpArray[5]를 구해보자. dpArray[5]가 될 수 있는 후보는
dpArray[0]+A[5] = 157+224 = 381
dpArray[2]+A[5] = 365+224 = 589
dpArray[3]+A[5] = 580+224 = 804
dpArray[4]+A[5] = 59+224 = 283
이 있고 이 중 가장 큰 804가 dpArray[5]가 된다.

이를 일반화해서 코드를 다음과 같이 작성할 수 있다.

```
// dpArray 초기화
for (int i=0; i<N; i++)
    dpArray[i] = A[i];

// 점화식
for (int i=0; i<N; i++) {
    for (int j=0; j<i; j++) {
        if (A[j]>=A[i] continue;
        if (dpArray[j]+A[i]>dpArray[i]) dpArray[i] = dpArray[j]+A[i];
    }
}
```

입력된 수열에 대해 그 수열의 증가하는 부분 수열 중에서 합이 가장 큰 것을 구하는 것이 목표였으므로 dpArray의 최대치가 구하던 값이 된다.