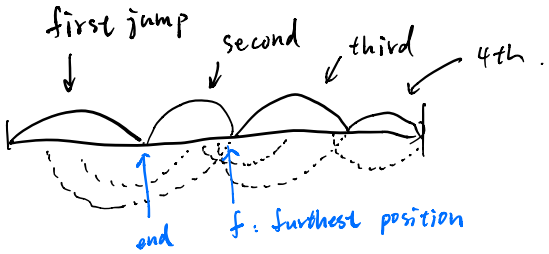
[2. 3. 1. 1. 4].



for i in range of 1th

⇒ only 1 step.

for i in range of 2th.

⇒ only 2 step

⋮

- iterate over nums.
  - when at end of range.
    ⇒ steps ++
    update end of range!

```
const fn = (nums) => {
    let steps = 0;
    let end = 0;
    let f = 0;
    for(let i=0; i<nums.length; i++) {
        f = Math.max(f, i+nums[i]);
        if(i === end){
            steps += 1;
            end = f;
        }
    }
    return steps;
}
```

we don't need to include last one.

```
  0  1  2  3  4  5  6
[3, 3, 3, 3, 0, 0, 1]
```

```
const fn = (nums) => {
    const n = nums.length;
    const dp = new Array(n).fill(Infinity);
    dp[0] = 0;
    for(let i=0; i<n; i++) {
      for(let j=1; j<=nums[i]; j++) {
        dp[i+j] = Math.min(dp[i+j], dp[i]+1);
      }
    }
    return dp[n-1];
}
```

time : O(N·K)

space : O(N)