单调递增的数字

给定一个非负整数 N，找出小于或等于 N 的最大的整数，同时这个整数需要满足其各个位数上的数字是单调递增。

（当且仅当每个相邻位数上的数字 x 和 y 满足 x <= y 时，我们称这个整数是单调递增的。）

示例 1:

输入: N = 10

输出: 9

示例 2:

输入: N = 1234

输出: 1234

示例 3:

输入: N = 332

输出: 299

说明: N 是在 [0, 10^9] 范围内的一个整数。

```
/*
（1）先把 N 拆成单个数字存入 buf
（2）然后比较前一个元素和后一个元素的大小，如果后一个元素大，则需要--
（3）从没调整的那一位之后的一位开始，一直到最后一位，全部写成 9
（4）再把 buf 换算成 N 返回
*/
int monotoneIncreasingDigits(int N){
    int buf[10]={0};
    int cout=0;
    int flag=0;
    while(N)
    {
        buf[cout++]=N%10;
        N=N/10;
    }
    for(int i=1;i<cout;i++)
    {
        if(buf[i]>buf[i-1])
        {
            buf[i]--;
            flag=i;//记录不再调整那一位
```

```
            }
        }
        for(int j=flag-1;j>=0;j--)
        {
            buf[j]=9;
        }
        for(int i=cout-1;i>=0;i--)
        {
            N*=10;
            N+=buf[i];
        }
        return N;
}
```