一、题目

给定一个非负整数num，你**至多**可以交换一次数字中的任意两位。返回你能得到的最大值。

1. 测试样例

样例 1：

```
输入：2736
输出：7236
解释：交换数字2和数字7。
```

样例 2：

```
输入：9973
输出：9973
解释：不需要交换。
```

2. 数据范围

$0 \leq num \leq 10^8$

二、解答

1. 整体思路

先分解输入num，得到num的每一位上的具体数字以及总位数。

再一一枚举数字中的两位进行交换，返回交换过程中能得到的最大的值

2. 代码

```
class Solution {
public:
    int maximumSwap(int num) {
        /*初始化*/
        //int 大小为-2147483648 ~ 2147483647(2*10^9)
        //我们的输入num在10^8(int)以内，同理输出ans也应该在int范围内
        int cnt=0,ans=num,x=num;
        //cnt:统计num的总位数，ans是答案，x是临时变量
        int a[10],pow[10];
        //a[i]为从后往前第i位的数字，pow[i]是对应位所代表的基数10^i(1,10,100,1000……)
        pow[1]=1;//初始化第一位的基数是1
        memset(a,0,sizeof(a));//初始化数组a全为0，此处可不写，因为cnt同样是a数组的大小

        /*分解x(也就是分解num,初始化x=num)*/
        //x=a[1]*pow[1]+a[2]*pow[2]+a[3]*pow[3]……  2736=6*1+3*10+7*100+2*1000
        while(x)
        {
            a[++cnt]=x%10;
            pow[cnt+1]=pow[cnt]*10;
            x/=10;
```

```
        }


        /*一一枚举num的两位进行交换得到最大值*/
        for(int i=1;i<cnt;i++)
            for(int j=i+1;j<=cnt;j++)
            {
                //交换第i位和第j位
                //x=num-a[i]*pow[i]-a[j]*pow[j]+a[i]*pow[j]+a[j]*pow[i]
                //合并同类项得x=num+(a[j]-a[i])*(pow[i]-pow[j])
                x=num+(a[j]-a[i])*(pow[i]-pow[j]);
                if(x>ans)ans=x;//如果当前交换结果大于已知的最大值，则更新最大值
            }

        return ans;
        //返回结果ans
    }
};
```

三、总结

本题难度较低，根据其数据范围大小可知直接枚举的暴力方法也可以轻松通过，对于常见于此类题型的

- 前导零问题（特殊讨论处理）
- 溢出风险处理（将所有相关数据类型转化为long long类型，或者使用高精度处理 ）

也无需多做处理

时间复杂度为$O(log^2num)$，主要是暴力枚举的二重循环

空间复杂度为$O(lognum)$

当然，也可以进一步使用贪心来进行算法的优化，可以做到线性处理（当数据范围很大比如超过10^10000次方时，推荐使用高精度+贪心）

大家感兴趣的话可以直接去翻翻后面的题解，此处不再赘述。

时间复杂度：$O(lognum)$

空间复杂度：$O(lognum)$