## 题目描述

## 给你一根长度为n的绳子，请把绳子剪成整数长的m段（m、n都是整数，n>1并且m>1），每段绳子的长度记为k[0],k[1],...,k[m]。请问k[0]xk[1]x...xk[m]可能的最大乘积是多少？例如，当绳子的长度是8时，我们把它剪成长度分别为2、3、3的三段，此时得到的最大乘积是18。

## 解题思路

/\*\*

 \* 题目分析：

 \* 先举几个例子，可以看出规律来。

 \* 4 ： 2\*2

 \* 5 ： 2\*3

 \* 6 ： 3\*3

 \* 7 ： 2\*2\*3 或者4\*3

 \* 8 ： 2\*3\*3

 \* 9 ： 3\*3\*3

 \* 10：2\*2\*3\*3 或者4\*3\*3

 \* 11：2\*3\*3\*3

 \* 12：3\*3\*3\*3

 \* 13：2\*2\*3\*3\*3 或者4\*3\*3\*3

 \*

 \* 下面是分析：

 \* 首先判断k[0]到k[m]可能有哪些数字，实际上只可能是2或者3。

 \* 当然也可能有4，但是4=2\*2，我们就简单些不考虑了。

 \* 5<2\*3,6<3\*3,比6更大的数字我们就更不用考虑了，肯定要继续分。

 \* 其次看2和3的数量，2的数量肯定小于3个，为什么呢？因为2\*2\*2<3\*3，那么题目就简单了。

 \* 直接用n除以3，根据得到的余数判断是一个2还是两个2还是没有2就行了。Y = number /3

余数为0 则直接就是 pow(3,y)

余数为1 此时拿出一个3 将（3\*1）<（2\*2）

此时应该有两个2 2\*2\*pow（3，y-1）

余数为2 就一个2

 \* 由于题目规定m>1，所以2只能是1\*1，3只能是2\*1，这两个特殊情况直接返回就行了。

算法描述：

class Solution {

public:

int cutRope(int number) {

if(number == 2)

return 1;

if(number == 3)

return 2;

int x = number % 3;

int y = number / 3;

if (x == 0)

return pow(3,y);

else if(x == 1)

return 2\*2\*pow(3,y-1);

else if(x == 2)

return 2\*pow(3,y);

}

};