给定a和n，计算a+aa+aaa+a...a(n个a)的和（大数据处理）

题目描述：
给定a和n，计算a+aa+aaa+a...a(n个a)的和。
输入：
测试数据有多组，输入a，n（1<=a<=9,1<=n<=100）。
输出：
对于每组输入,请输出结果。
样例输入：
1 10
样例输出：
1234567900
从题中就可以看出，当a = 9, n = 100的时候，一个int类型的数是存不下100位的数，所以不能运用平常的方法来求，下面介绍我的解法，我声明一个向量v用来存储a+aa+aaa+a...a(n个a)的和，temp是用来存储a...a(n个a)的，从个位向高位分别相加，hight用来存储进位的。

```
#include <iostream>
#include <vector>
using namespace std;

int main(){
 int a, n;
 int sum = 0;
 vector<int> v;
 vector<int> temp;
 vector<int>::iterator it;
 int hight = 0; //存储进位
 while(cin >> a >> n){
  v.clear();
  temp.clear();
  v.push_back(a);
  temp.push_back(a);
  for(int i = 2; i <= n; i++){
   temp.push_back(a);
   int j = temp.size() - 1;
   int k = v.size() - 1;
   hight = 0;
   sum = 0;
   while(k >= 0 && j >= 0){ //从低位向高位相加
    sum = temp[j] + v[k] + hight;
    hight = 0;
    if(sum > 9){
     hight = sum / 10; //求进位
```

```
   }
   v[k] = sum % 10;
   k--;
   j--;
  }

  //if(hight > 0){
   while(j >= 0){ //可能要加的数比总的位数还要多，比如 9  + 99;
    sum = temp[j] + hight;
    hight = 0;
    if(sum > 9){
     hight = sum / 10;
    }
    v.insert(v.begin(), sum % 10);
    j--;
   }

   if(hight > 0){//如果还有进位，那就放到最高位
    v.insert(v.begin(), hight);
   }
  //}
 }

 for(it = v.begin(); it != v.end(); it++){
  cout << *it;
 }
 cout << endl;
 }
 return 0;
}
```

结果：a = 9 n = 200,a+aa+aaa+a...a = 1111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111091
0

下面有个人给出了更简单的解法：直接模拟小学加法从个位数开始加，该进位的进位，然后存到一个栈里面，最后出栈输出就完事了，代码：

```
#include <cstdio>
#include <stack>
using namespace std;
```

```
int main()
{
//    freopen("1.txt", "r", stdin);
    int a, n, i, t, c;
    while(~scanf("%d %d", &a, &n))
    {
        stack<int> S;
        for(c=0,i=1; i<=n; i++)
        {
            t = (n-i+1)*a;
            S.push((t+c)%10);
            c = (t+c)/10;
        }
        if(c>0)
            S.push(c);
        while(!S.empty())
        {
            printf("%d", S.top());
            S.pop();
        }
        printf("₩n");
    }
    return 0;
}
```

但是这个有个缺点，就是当n好大，t = (n - i + 1) * a 会溢出，这个程序的优点是运行速度很快，我上面的代码当n比较大的时候，运行速度很慢。但不会溢出。