D Power up!!——张潮鹏 & 渣诚

题目描述

由于上一次上机的逗比情况，激发了渣诚的慢性心理阴影，所以这一次，他决定出一道很简单的题目：求 a 的 π 次方。
然而这时 Thor 发话了——怎么能这么水呢？于是机智的 Thor 决定附加有个规则：
不准使用 pow 函数！
So……good luck fellows……

输入

多组测试数据。
每组测试数据为一行，只有一个正实数，表示 a。
保证输入数据合法

输出

对于每组输入数据，输出一行，包含一个实数，为计算结果，结果保留两位小数。

输入样例

1
1

输出样例

1.00
1.00

HINT

我、我的草稿纸呢？
(ノ ' □ ')ノ ┻━┻
Thor 表示他什么都不知道……

Think:

D 题要计算 a^π 次方？刚开始以为是宋友老师在课堂上要我们自己 a^b 函数的翻版。。稍稍一想，略觉不同，这明明不是整数次方，貌似无法用 for 循环计算涅。再转弯一想，是否可以 π 取 3.14，然后计算 $a^{\frac{157}{50}}$，这样就可以解决幂不为整数的麻烦了。But。。敢想却不敢做，因为直觉告诉我，a^157 会溢出。高精度幂又没掌握。又只能作罢。

也许为了高数的期中考，临时抱了泰勒的脚，发现它跟（1+x）^α 长得挺像，无非令 x+1=a,α=π 。便可以用泰勒展开式解决了。。。原来泰勒公式（别名：太乐公式）不仅可以用来求极限，求阶，求近似值，竟然还可以用来过 OJ。。难怪魏光美老师会说它是微分学的皇冠了。膜拜泰勒 ORZ　ORZ　ORZ.....

Code：

```
#include<iostream>//呵呵，助教竟然考我们泰勒公式。
#include<iomanip>
#include<cmath>
using namespace std;//(1+x)^a=1+ax+……
double  fn(double x,double a)//计算(1+x)^a
{
    double tx=a*x;//tx 为展开式中的通项
    double fn=tx;//除了1以后的第一项
  for(int  k=2;tx>0.00000000000000000000000001;k++)// 当 最 后 一 项 比
//0.00000……01还小的时候我就不要了。这个可随意，仁亲选择，别 WA 就 OK。
  {
      tx=tx*((a-k+1)/k)*x;//通项之间的递推关系
      fn+=tx;//把每个通项都累加起来
  }
```

```
    return fn+1;//别忘了最前面的那个不带 x 的“1”
}
int main()
{   double n;
    while(cin>>n)
    {   double a=3.1415926535898;//刚开始 a 取值小了，结果 WA 了，so,适
//当取大能精确些
        if(0<n&&n<2)cout<<setprecision(2)<<fixed<<fn(n-1,a)<<endl;
        else
        {
            n=1/n;
            a=-a;
            cout<<setprecision(2)<<fixed<<fn(n-1,a)<<endl;
        }
    }
}//上完机同学说助教告诉他们不用这么复杂，顿时泪奔。。会错意了。
```

官方解答

上机前发给大家的文档其实是有原因的……原因就是——这道题的答案就在里面哦—v—Y

首先考虑不使用 pow 函数，但是其它函数都可以使用。这样的话，相信大家都学过一个叫做指对互化的东西，于是我们就用到了一个叫 exp 的和另一个叫 log 的库函数

$$a^b = \exp\left(\ln\left(a^b\right)\right) = \exp(\mathrm{b} * \ln\mathrm{a})$$

而 cmath 库中的 ln 函数即为 log(double)

这样一来，问题就基本解决了。

至于 π 的话……有一个东西叫做反三角函数……

总之本题的目的就是想给大家拓宽一下视野，cmath 库中还有许多有趣的函数，大家有兴趣可以查一查相关的文档。

参考代码：

```
#include<iostream>
#include<cmath>
#include<iomanip>
#include<fstream>
#define calc(x) exp(acos(-1)*log(x))
using namespace std;
int main()
{
    int n;
    while(cin>>n)
       cout<<setprecision(2)<<fixed<<calc(n)<<endl;
    return 0;
}
```