求导后会发现这个函数在[0,100]的区间要么先单调递减后单调递增，要么一直单调递增，符合三分算法条件

直接套个实数三分去判就行了。裸题，不解释，不会三分的看博客去学。(先学二分，此题二分也可做)

也可以求导后，再用二分算法去求导数函数的零点。

此处只放三分的标程，二分去找过了的同学要代码。

C语言标程:

#include<stdio.h>

#include<math.h>

#define ll long long

**const** **double** eps=1e-10;

**double** y;

**double** cal(**double** x)

{

**return** 6\*pow(x,7)+8\*pow(x,6)+7\*pow(x,3)+5\*pow(x,2)-y\*x;

}

**int** main()

{

**int** t;

    scanf("%d",&t);

**while**(t--)

    {

        scanf("%lf",&y);

**double** l=0,r=100;

**while**(r-l>eps)

        {

**double** lmid=l+(r-l)/3;

**double** rmid=r-(r-l)/3;

**double** lans=cal(lmid),rans=cal(rmid);

**if**(lans<=rans) r=rmid;

**else** l=lmid;

        }

        printf("%.4lf\n",cal(l));

    }

}

C++标程:(交G++)

#include<bits/stdc++.h>

#define ll long long

#define IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

**using** **namespace** std;

**const** **double** eps=1e-10;

**double** y;

**double** cal(**double** x)

{

**return** 6\*pow(x,7)+8\*pow(x,6)+7\*pow(x,3)+5\*pow(x,2)-y\*x;

}

**int** main()

{

    IOS

**int** t;cin>>t;

**while**(t--)

    {

        cin>>y;

**double** l=0,r=100;

**while**(r-l>eps)

        {

**double** lmid=l+(r-l)/3;

**double** rmid=r-(r-l)/3;

**double** lans=cal(lmid),rans=cal(rmid);

**if**(lans<=rans) r=rmid;

**else** l=lmid;

        }

        cout<<fixed<<setprecision(4)<<cal(l)<<endl;

    }

}