电 子 科 技 大 学 实 验 报 告

课程名称： Python语言程序设计及其应用

实验地点： 科A229

指导教师： 张勇

评 分：

完成实验学生信息：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序** | **姓名** | **学号** | **选课**  **序号** | **贡献**  **百分比(%)** | **备注**  **（主要工作**） |
| 1 | 付文亮 | 2022080401004 | 69 |  | 1.xx  2.xx |
| 2 | 傅若山 | 2022080401023 | 71 |  | 1.  2. |
| 3 | 苏徐涛 | 2022080401011 | 70 |  | 1.  2. |
| 4 |  |  |  |  | 1.  2. |

1. 学生人数按照任课教师要求限定；
2. 对于“评价、改进、总结和体会”都要认真填写，和其他内容是评价实验成绩的重要参考。

**实验题目名称: 科学计算实验（数值分析+最优化方法）**

1. **实验内容**
2. 编程实现牛顿法求解非线性方程

请先设计接口，编程实现，并求解下列方程的解。



1. **实验目的**

实现牛顿法求解非线性方程

1. **实验过程**

三人分别思考，根据网上相关资料分别编写代码，提出各自的方案。

目 录

[1. 实验1：科学计算实验（数值分析+最优化方法） 5](#_Toc5577)

[1.1. 问题分析 5](#_Toc17416)

[1.2. 系统设计与算法设计 5](#_Toc6345)

[1.3. 编写程序 5](#_Toc3642)

[1.4. 运行结果 5](#_Toc23611)

[1.5. 实验结果分析 5](#_Toc2792)

[1.6. 优缺点及改进方向 5](#_Toc2495)

[1.7. 心得体会与总结 6](#_Toc11727)

[2. 实验2：XXXX 6](#_Toc23636)

[3. 问题重述 6](#_Toc19292)

[4. 对本次实验的设计提出改进意见 6](#_Toc28269)

[5. 附件 6](#_Toc21740)

[附件1.XX的程序 6](#_Toc4555)

[附件2.YY的程序 6](#_Toc11208)

[附件3.ZZ程序的输出结果 7](#_Toc18163)

# **实验1：**科学计算实验（数值分析+最优化方法）

## 问题分析

该问题从题面来看主要是个数学问题，编程实现的难度并不大。其中先理解牛顿迭代法的数学原理，牛顿迭代法又称牛顿-拉夫逊方法，是牛顿在17世纪提出的一种在实数域和复数域上近似求方程的方法。该方法的基础是利用泰勒展开式。方法使用函数f(x)的泰勒级数的前几项寻找方程f(x) = 0 的根。最大优点是在方程f(x)=0的单根附近具有平方收敛，该方法可以用来求方程的重根、复根。

## 系统设计与算法设计

功能设计：设计一个函数Newton()，提供一个接口，返回解的情况以及解的个数。该函数的参数包括解的预测值、原函数、原函数的导数、精确度以及最大迭代次数。

算法设计：函数内部按照牛顿迭代法的思路来写。

## 编写程序

import numpy  
  
def Newton(x\_front,f,df,accuracy=1e-6,max\_iter=100):  
 '''  
 牛顿法求解非线性方程  
 :param x\_front: x的初始值  
 :param f: 函数  
 :param df: 导数  
 :param accuracy:精确度  
 :param max\_iter: 最大迭代次数  
 :return: 无解或解的值  
 '''  
 for i in range(max\_iter):  
 x\_back=x\_front-f(x\_front)/df(x\_front)  
 if abs(x\_back-x\_front)<accuracy:  
 return x\_back,i  
 x\_front=x\_back  
 return '该方程无解！'  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 x0=int(input("请输入预测值："))  
 f=lambda x:2\*x\*x+10/numpy.exp(x)-5  
 df=lambda x:4\*x-10/numpy.exp(x)  
  
 ans=Newton(x0,f,df)  
 if type(ans)==str:  
 print(ans)  
 else:  
 print("近似解为{:6f}，迭代次数为{:d}".format(ans[0],ans[1]))

## 运行结果

输入：

函数：

导数：

精确度：1e-6

预测值：2

输出：

该方程无解！

输入：

函数：

导数：

精确度：1e-6

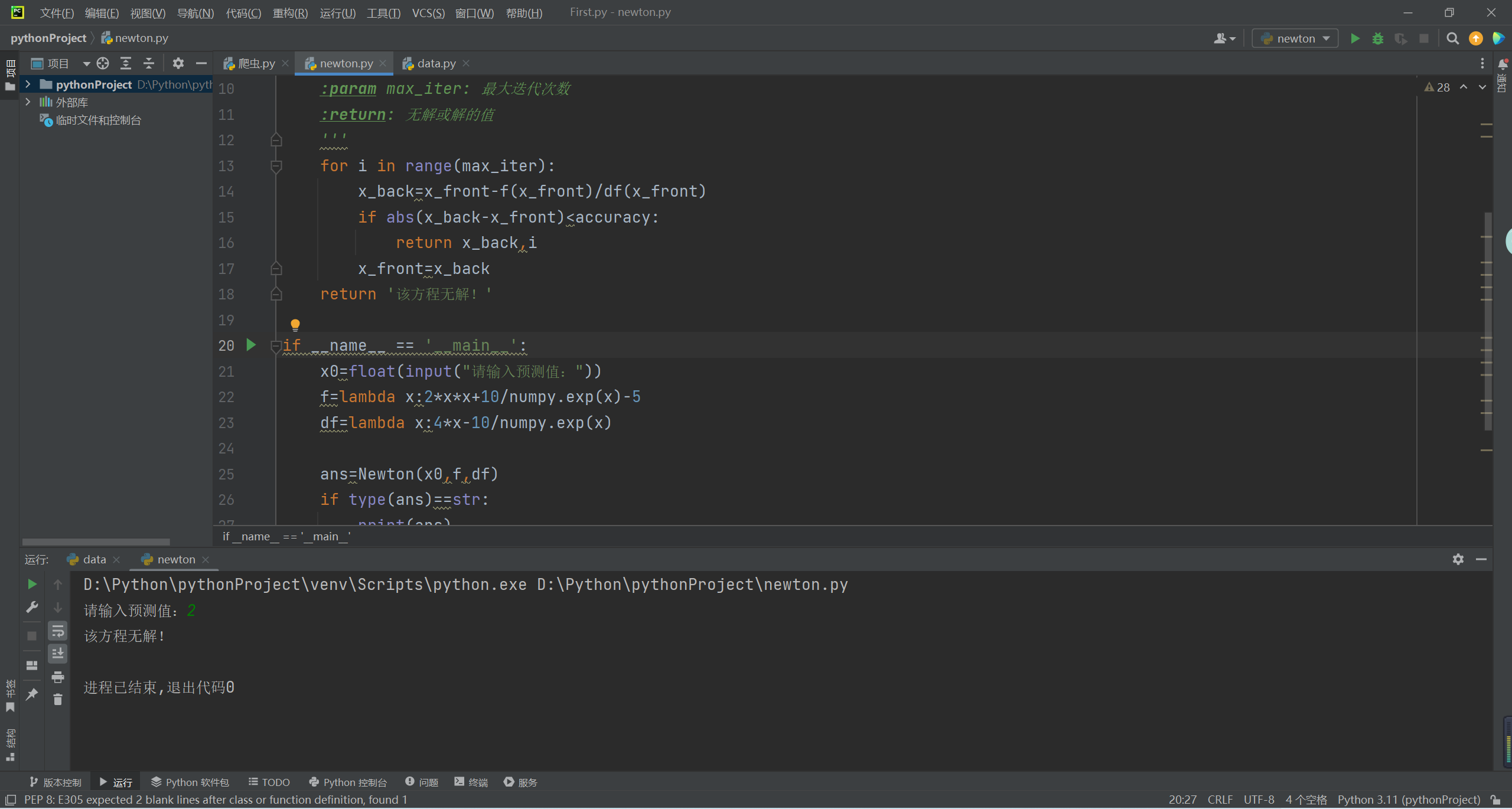
预测值：2、0

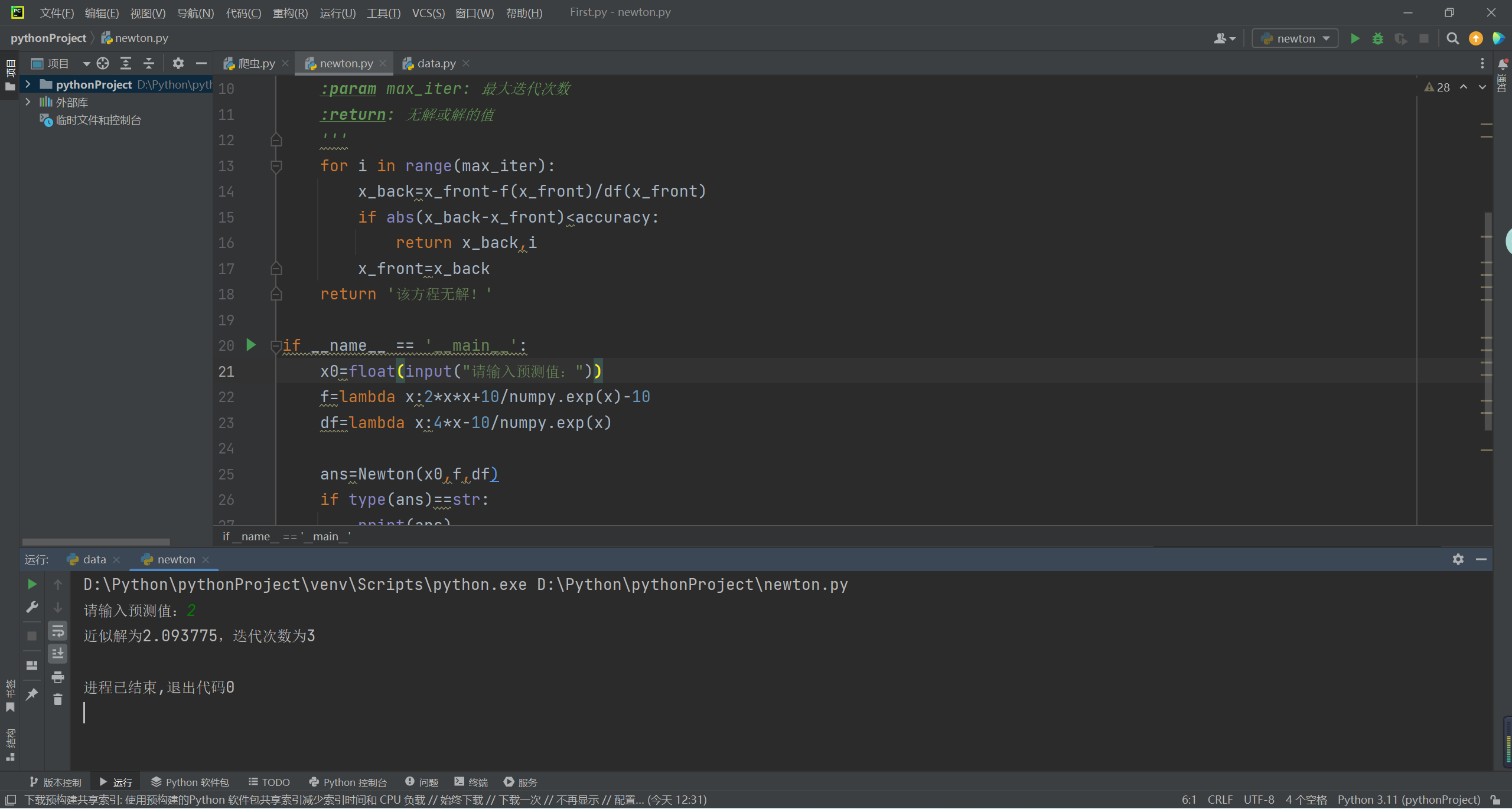
输出：

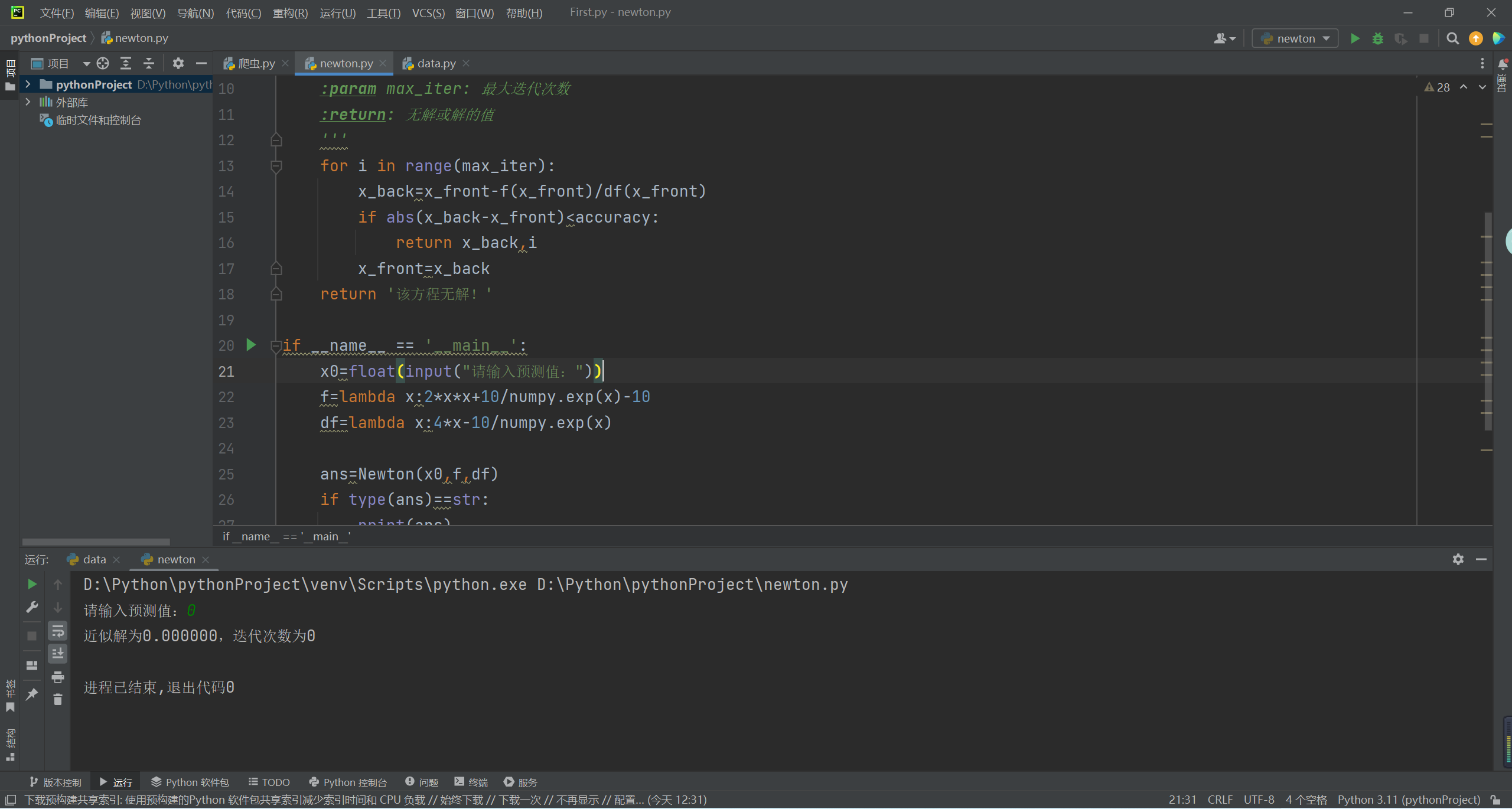
近似解为2.093775，迭代次数为2

近似解为0.000000，迭代次数为0

## 实验结果分析







可以从结果看出：牛顿迭代法的准确度很高，返回的结果也清晰明了。当求方程的精确解很困难或不可能时，用牛顿迭代法求近似解也是很重要的。

## 优缺点及改进方向

优点：能控制精确度和最大迭代次数，可以计算出非常精确的近似解，且操作方便，容 易理解。

不足：需要人为给出一个预测值，如果预测值给的太偏或恰好在导数值等于0处则失去 作用。此外，该方法仅能给出一个解，具体得到哪个解则取决于预测值。

改进：进一步识别导数值等于0的特殊情况，以及给出多解。

## 心得体会与总结

对具体数学方法在程序实现上的理解再一次加深，在学习算法的道路上又前进了一步。这个实验只是一个开端，它为以后我们学习更难的算法与程序提供了经验。

# **实验2：XXXX**

# **问题重述**

# **对本次实验的设计提出改进意见**

(向教师反馈实验的设计，例如，修改实验任务使得更能体现能力、水平；增加实验任务等等；所有围绕实验的建议、意见。)

# **附件**

## 附件1.牛顿法的程序

import numpy  
  
def Newton(x\_front,f,df,accuracy=1e-6,max\_iter=100):  
 '''  
 牛顿法求解非线性方程  
 :param x\_front: x的初始值  
 :param f: 函数  
 :param df: 导数  
 :param accuracy:精确度  
 :param max\_iter: 最大迭代次数  
 :return: 无解或解的值  
 '''  
 for i in range(max\_iter):  
 x\_back=x\_front-f(x\_front)/df(x\_front)  
 if abs(x\_back-x\_front)<accuracy:  
 return x\_back,i  
 x\_front=x\_back  
 return '该方程无解！'  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 x0=float(input("请输入预测值："))  
 f=lambda x:2\*x\*x+10/numpy.exp(x)-5  
 df=lambda x:4\*x-10/numpy.exp(x)  
  
 ans=Newton(x0,f,df)  
 if type(ans)==str:  
 print(ans)  
 else:  
 print("近似解为{:6f}，迭代次数为{:d}".format(ans[0],ans[1]))

## 附件2.YY的程序

## 附件3.ZZ程序的输出结果