**计算机科学与工程学院实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验课程名称** | | **Python编程与数据分析** | | | **实验总成绩** |  |
| **专业** | **物联网工程** | | **班级** |  | **指导教师签字** |  |
| **学号** |  | | **姓名** | **陆晓晓** | **实验报告批改时间** |  |
| **实验报告分项成绩**   |  |  |  | | --- | --- | --- | | **序号** | **实验项目** | **成绩** | | **1** | **Project2** |  | | **2** |  |  | | **3** |  |  | | **4** |  |  | | **5** |  |  | | | | | | | |
| **实验课程总结**  从以下方面总结：1.实验体现知识应用和初步研究能力；2. 反映基本观察、发现问题和分析问题能力；3. 实验项目内容或者实验课程是否存在问题及下一年度改进意见）4.其他方面  这是第二次python实验，和上一次做的project1内容基本有很大的不同，由于对tkinter库不太熟悉设计出来的界面比较丑，所以我改用了PyQT，QT的文档比较多，并且之前在学习C++的时候我已经熟悉过QT的使用。  Python 标准库和第三库众多，功能强大，这一点是c、c++比不了的，代码的封装程度很高，并且相关文档丰富，基本对应的库看一下文档就能会用了。在c、c++中100行的代码在python甚至只需要几行就能实现。  在写project2花费的时间比projec1 的时间要多很多，首先要回忆起来QT的使用方法，并且PyQT的语法规范还是和C++的QT Creator有很多不同之处的，之后便是利用信号和槽机制来完成点击主界面的按钮来跳出子界面，打印表格等等。  同时，我认识到自己对于python语言的练习还是不足的，之后会利用课后时间多加练习，这样也是为期末考试做准备。 | | | | | | |

目录

[1. 实验目的 4](#_Toc26950)

[2. 实验需求分析 4](#_Toc19249)

[3. 实验内容 5](#_Toc2275)

[3.1文件说明 5](#_Toc980)

[3.2 实验运行环境 5](#_Toc28915)

[3.2 程序设计过程思路、遇到问题 7](#_Toc5745)

[3.2.1为什么使用PyQT 7](#_Toc2903)

[3.2.2 在配置PyQT环境的一个问题 7](#_Toc14513)

[3.2.3 爬虫爬取数据、数据预处理 7](#_Toc6127)

[3.2.3.1 爬取城市的名单说明： 7](#_Toc25092)

[3.2.3.2 爬虫的思路： 8](#_Toc13133)

[3.2.3.3 数据预处理 9](#_Toc25128)

[3.2.3.4 处理效果 12](#_Toc14288)

[3.2.4 自动补全、模糊匹配 13](#_Toc11287)

[3.2.5 实现点击按钮，子窗口弹出（利用信号和槽机制） 14](#_Toc19674)

[3.2.6 利用面向对象思想复用获取首字母拼音函数 14](#_Toc26941)

[3.2.7 基于QPalette实现了背景图自适应 15](#_Toc29123)

[3.2.8 写日志 15](#_Toc29201)

[3.2.9 进度条 15](#_Toc19127)

[3.2.10 读文件，显示二手房信息 16](#_Toc32390)

[4.运行结果展示 17](#_Toc2695)

[5.源代码 18](#_Toc3623)

[demo\_main.py：（调用两个界面类） 18](#_Toc29323)

[Ui\_untitled.py：子界面类 21](#_Toc1283)

[Ui\_untitled1.py：主界面类 38](#_Toc9828)

[6.心得体会 42](#_Toc12501)

# Project2

## 实验目的

加深对课上内容的理解，计划做一个关于二手房数据分析项目，共分4个小Project，每2周做一个。

项目需求为基本需求，没有特别的限定死，自由发挥的空间较大，使用的方法和库不设限。

提交报告+代码，报告中对实验步骤做说明，列出并解释关键代码。

## 实验需求分析

1.用向对象思想，按需设计并复Project1中的功能函数(旧志进度条等)(非必选)

2.基于tkinter开发GUI,输入省会城市名,显示手房信息

3.要显示的信息有:位置户型面积/朝向/装修/楼层/结构1价格等，注意访问的频率，可以不全部加载(可在链家网等二手房网站爬取、如用其他方式也可)

**实际实现的内容：（标黄的为一些亮点）**

1.输入省会城市名称，控制台可以获取对应汉语拼音小写首字母，例：北京->bj；

2.支持输入的模糊匹配（搜索“京”可以提示“北京”，输入”北”，可以把”北京”、”台北”都匹配出来）

3.基于面向对象思想，封装了写日志（将日志写到log文件里，记录内容自定）的功能函数

4.基于面向对象思想，实现加载城市文件的进度条

5.爬虫爬取了省会城市信息，可以显示地址/户型/面积/朝向/装修/楼层/建立/结构/价格九项信息

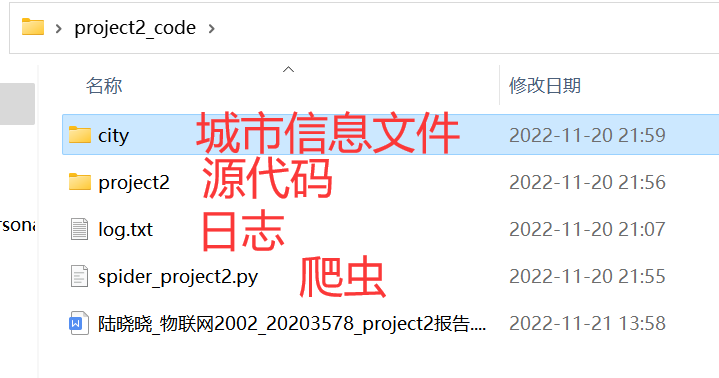
6.基于PyQT开发GUI，输入省会城市名，显示手房信息（使用PyQT进行界面设计，之后在进行主要逻辑的代码编写）

7.添加了背景图，进行了界面的美化

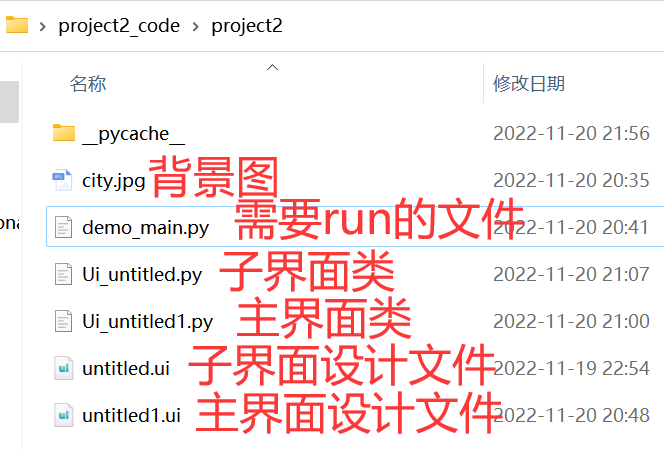
## 实验内容

### 3.1文件说明

整体上分为五个部分：



源代码文件夹里包括：



**运行方法：将project2通过Vscode打开，然后直接在demo\_main.py文件中按F5即可。**

### 3.2 实验运行环境

引入库文件部分如下：

demo\_main.py：



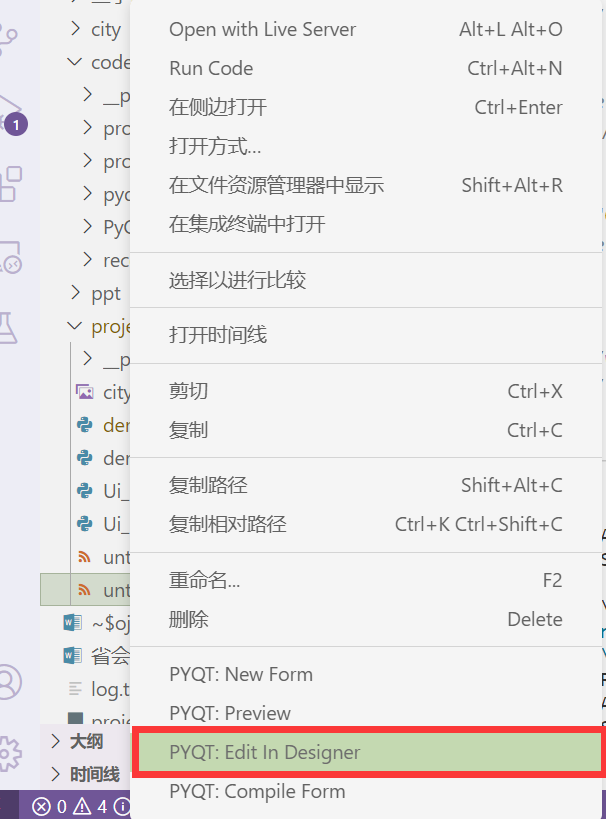
Ui\_untitled.py：



Ui\_untitled1.py：



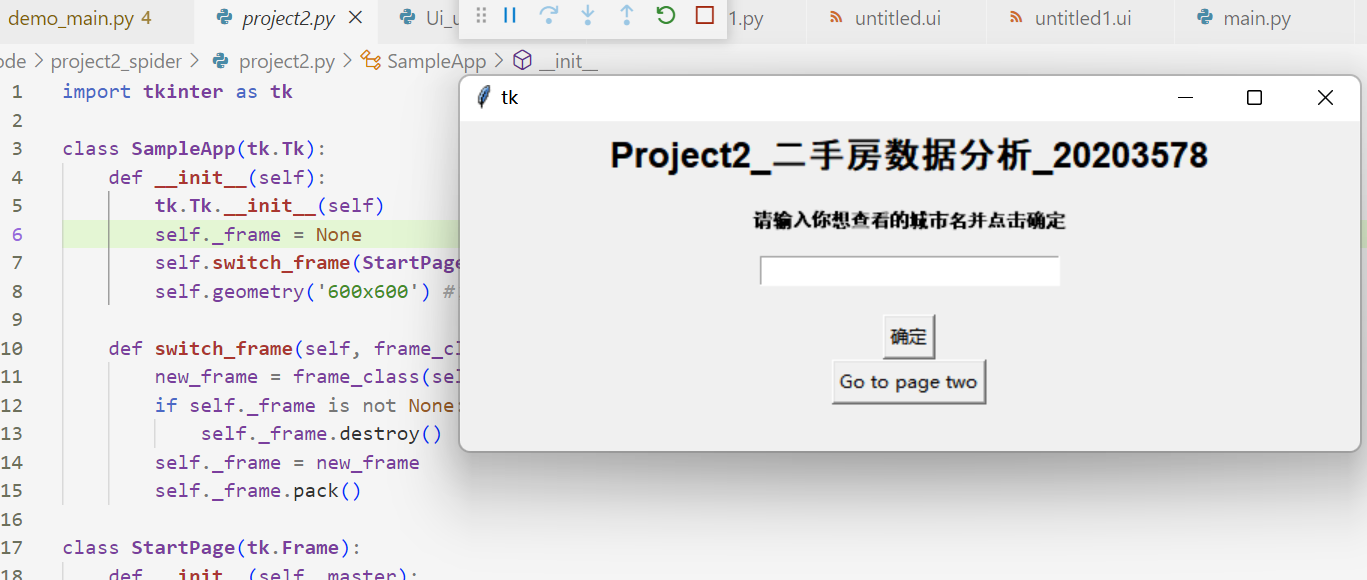
两个.ui文件可以右键进入到QT designer进行编写：



### 3.2 程序设计过程思路、遇到问题

### 3.2.1为什么使用PyQT

一开始使用了老师推荐的tkinter库，由于之前没有用过类似的库导致自己写的东西非常的丑，加上界面设计也非常头疼，听同学说也可以使用其他的工具开发GUI，加上自己之前学习过使用过QT（C++），就决定使用PyQT了，当然PyQT和QT的语法还是整体不一样的，但实现方法、信号和槽机制都是类似的。所以为了更好的完成这次作业，采用了PyQT。



图：尝试使用tkinter编写的界面（太丑了...）

### 3.2.2 在配置PyQT环境的一个问题

我的Python版本是3.10.4，在使用pip命令安装PyQT-tools时候一直安装不上，尝试了很多方法，之后上网搜索发现，PyQT的最新版本不支持Python3.10以上的版本。

解决方法：直接安装QT designer，还是中文版，在VsCode里面配置好了环境用起来很方便~

### 3.2.3 爬虫爬取数据、数据预处理

### 3.2.3.1 爬取城市的名单说明：

爬取的城市名单：标黄的是链家网网站没有的，也就没有爬取，还有一个银川市也没有数据，因为首字母是yc，进去链家网对应的网站是盐城。

北京市 京--北京 ；上海市 --上海 ；天津市 --天津 ；重庆市 --重庆 ；黑龙江省 --哈尔滨 ；吉林省 --长春； 辽宁省 --沈阳 ；内蒙古自治区--呼和浩特 ；河北省 --石家庄 ；新疆维吾尔自治区--乌鲁木齐 ；甘肃省-- 兰州 ；青海省--西宁 ；

陕西省 --西安 ；宁夏回族自治区--银川(是盐城 ；河南省-- 郑州 ；山东省--济南；山西省--太原 ；安徽省--合肥 ；湖南省--长沙；湖北省--武汉 ；江苏省--南京 ；四川省--成都； 贵州省--贵阳 ；云南省--昆明 ；广西壮族自治区--南宁 ；

西藏自治区-- 拉萨(没有 ；浙江省--杭州； 江西省--南昌 ；广东省--广州 ；福建省--福州 ；台湾省--台北 ；海南省 --海口 ；香港特别行政区-- 香港； 澳门特别行政区-- 澳门。

### 3.2.3.2 爬虫的思路：

1.确定链接url：

city='cd'

page=1

url='https://'+city+'.lianjia.com/ershoufang/ty'+**str**(page)

* city替换成每次想爬的城市名字首字母就可以了
* page=1 代表每次都爬取一页就够了，30条信息
* url 观察链家网的名称特点得到的

2.获取url对应页面内容：

import **requests**

import **time**

*#response=requests.get(url)*

response=**requests**.**get**(url, headers={'Connection':'close'})

3.利用lxml解析数据：

from **lxml** import **etree**

data=**etree**.**HTML**(response.text)

**print**(data)

4.用xpath进行数据匹配：

positionInfo=data.xpath('//\*[@id="content"]/div[1]/ul/li/div[1]/div[2]/div/a[1]')

**print**(**type**(positionInfo))

**print**(**len**(positionInfo))

for temp in positionInfo:

**print**(temp.text)

**time**.**sleep**(0.5)

houseInfo=data.xpath('//\*[@id="content"]/div[1]/ul/li/div[1]/div[3]/div/text()')

**print**(**type**(houseInfo))

**print**(**len**(houseInfo))

for temp in houseInfo:

**print**(temp)

**time**.**sleep**(0.5)

*#爬取价格*

priceInfo=data.xpath('//\*[@id="content"]/div[1]/ul/li/div[1]/div[6]/div[1]/span/text()')

*#//\*[@id="content"]/div[1]/ul/li[1]/div[1]/div[6]/div[1]/span*

*#//\*[@id="content"]/div[1]/ul/li[2]/div[1]/div[6]/div[1]/span*

**print**(**type**(priceInfo))

**print**(**len**(priceInfo))

for temp in priceInfo:

**print**(temp)

爬取哪一块的内容，分析对应网页的对应块得到的：

//\*[@id="content"]/div[1]/ul/li/div[1]/div[2]/div/a[1]

5.注意访问的频率：

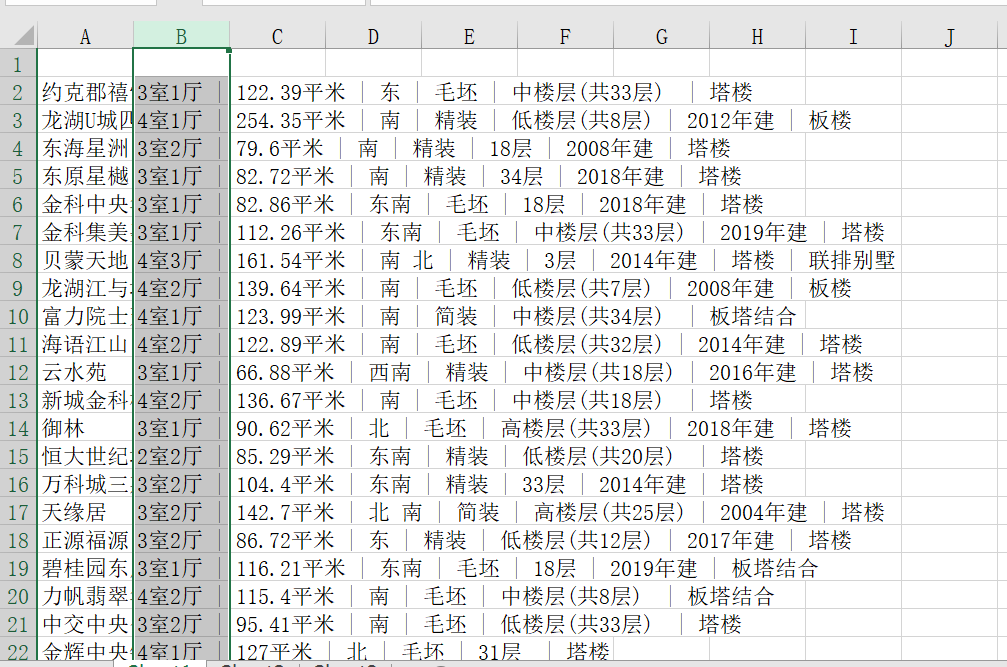
使用time.sleep，（因为有一天爬的太多了就被禁了。。。不过第二天又能用了）

### 3.2.3.3 数据预处理

观察数据，发现房屋的其他信息和位置信息是分开爬取的，所以需要使用|分开，使用excel可以完成基本的数据处理

3室1厅 | 122.39平米 | 东 | 毛坯 | 中楼层(共33层) | 塔楼

第一步，选中这一列



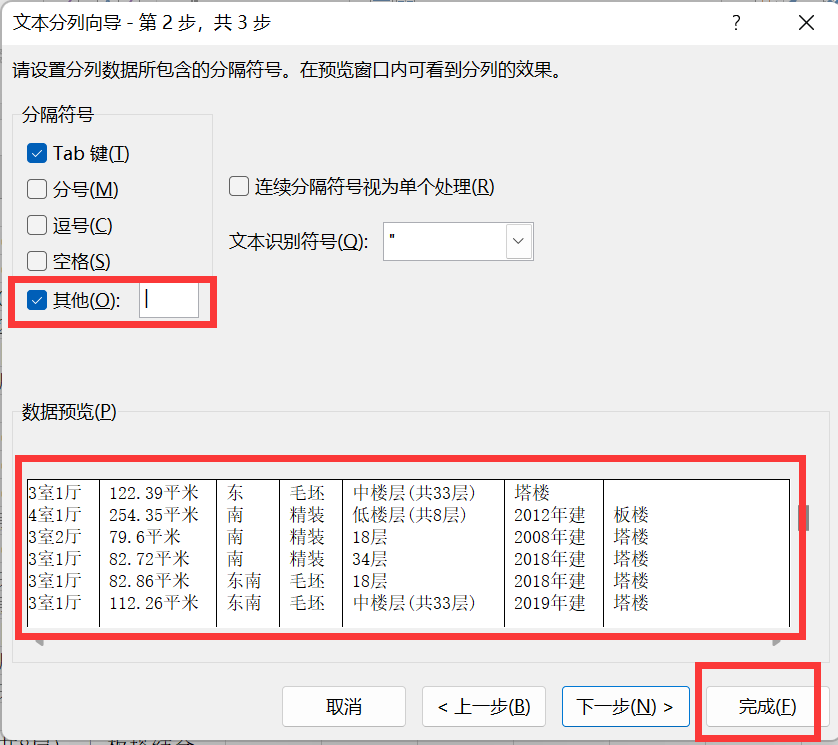
第二步，选中数据，选择分列



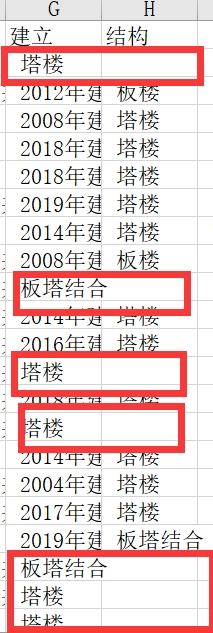
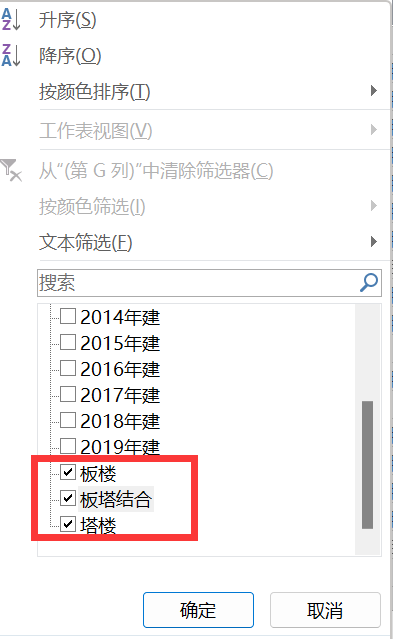
第三步，选择分隔符号



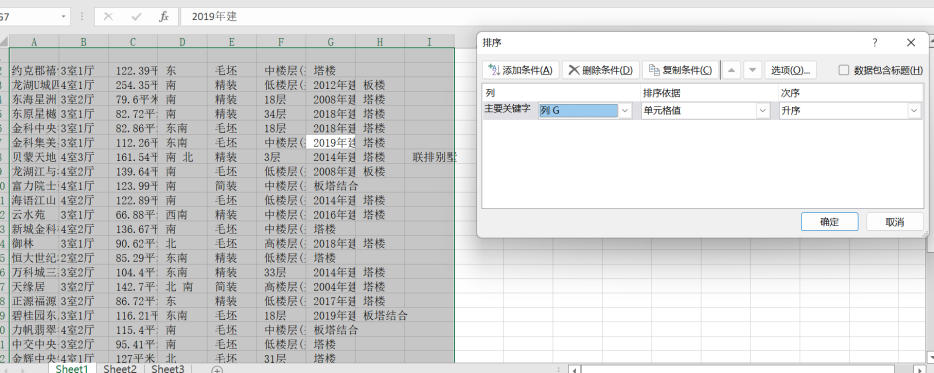
第四步，选择其他符号，选择|分列，点击完成就好了



第五步，有些信息会串列，处理串列的信息

通过excel 筛选，把时间里面的建筑类型勾选上，之后再排序、处理



把空白位置填补上缺失：

整理之后的数据



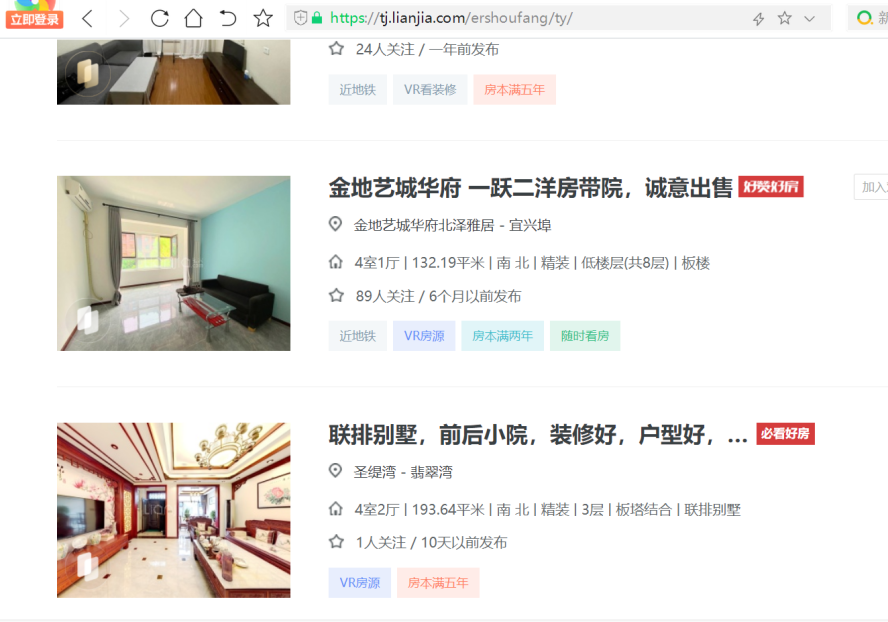
每一个城市信息都进行如上整理。

但还是有一部分城市在链家网是彻底没有建立时间的

比如天津市：

天津市的建立时间是缺失的，回到网站上浏览，发现网站上缺失没有该信息

https://tj.lianjia.com/ershoufang/ty/



像这样的城市直接把这一列标注为缺失即可。

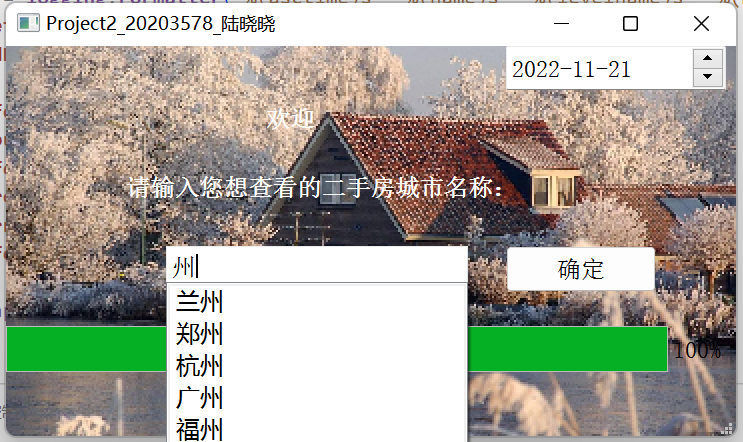
### 3.2.3.4 处理效果

回到网站，对应一下爬取的信息和GUI输出的信息，准确率还可以，大部分都可以对应上。



### 3.2.4 自动补全、模糊匹配

最终成果：输入”州”，能够匹配到带有州的五个城市：兰州、郑州、杭州、广州、福州



实现：

def **auto\_complete**(self):

        items\_list=['北京','上海','天津','重庆','哈尔滨','长春','沈阳','呼和浩特','石家庄','乌鲁木齐','兰州','西宁','西安','银川','郑州','济南','太原','合肥','长沙','武汉','南京','成都','贵阳','昆明','南宁','拉萨','杭州','南昌','广州','福州','台北','海口','香港','澳门']

        self.completer = **QtWidgets**.**QCompleter**(items\_list)

*# 设置匹配模式  Qt.MatchContains 内容匹配*

        self.completer.**setFilterMode**(**QtCore**.**Qt**.MatchContains)

*# 设置补全模式  QCompleter.PopupCompletion 弹出选项补全（默认）*

        self.completer.**setCompletionMode**(**QtWidgets**.**QCompleter**.PopupCompletion)

        self.lineEdit.**setCompleter**(self.completer)

* 定义一个自动补全函数，设置好匹配列表：items\_list
* 使用Qt.MatchContains 内容匹配
* 使用QCompleter.PopupCompletion 弹出选项补全
* 最终设置好lineEdit组件要setCompleter就可以了

### 3.2.5 实现点击按钮，子窗口弹出（利用信号和槽机制）

myWin.pushButton.**clicked**.connect(myWinSecond.**show**)

信号发出者：主界面的pushButton

信号：clicked

接收者：子界面

接收者执行：show

### 3.2.6 利用面向对象思想复用获取首字母拼音函数

设计的思路：将该函数封装到点击按钮之后的槽函数

实现功能：点击确定，会打印你输入的城市首字母

槽函数设计：

myWin.pushButton.**clicked**.connect(myWin.**click\_my\_btn**)

封装设计：

def **click\_my\_btn**(self):

        def **getStrAllAplha**(str):

            return **pinyin**.**get\_initial**(str, delimiter="")

**print**("你要查看的城市的首字母是：")

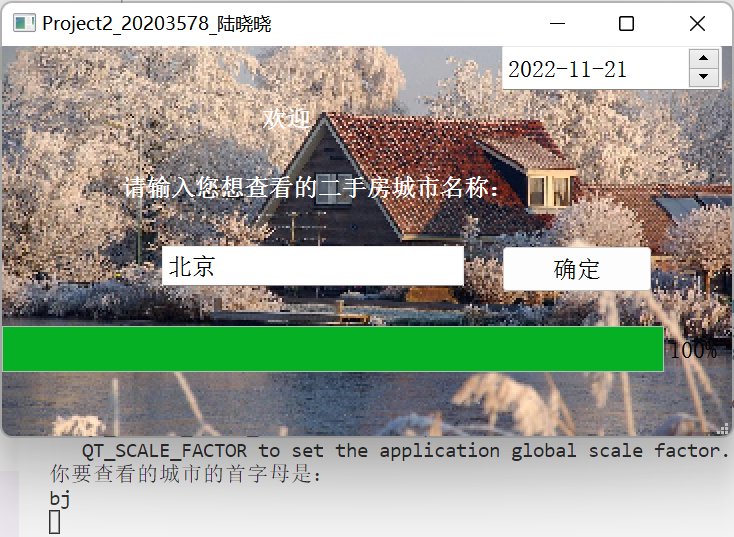
        str\_tmp=self.lineEdit.**text**()

*#print(str\_tmp)*

        self.str=**getStrAllAplha**(str\_tmp)

**print**(self.str)

实现结果：



### 3.2.7 基于QPalette实现了背景图自适应

*#设置背景*

        palette = QPalette()

        pix=QPixmap("project2/city.jpg")

        pix = pix.scaled(self.**width**(),self.**height**())

        palette.setBrush(QPalette.Background,QBrush(pix))

        self.**setPalette**(palette)

这里pix = pix.scaled(self.width(),self.height())这行代码是设置背景图的自适应

其他没什么好说的，这里是相对路径（也可以是绝对路径）

### 3.2.8 写日志

*#写日志文件部分*

    logger = **logging**.**getLogger**(\_\_name\_\_)

    logger.**setLevel**(level = **logging**.INFO)

    handler = **logging**.**FileHandler**("log.txt")

    handler.**setLevel**(**logging**.INFO)

    formatter = **logging**.**Formatter**('%(asctime)s - %(name)s - %(levelname)s - %(message)s')

    handler.**setFormatter**(formatter)

    logger.**addHandler**(handler)

    logger.**info**("Start print log")

    logger.**debug**("Do something")

    logger.**info**("project2\_Information of Second hand house")

    logger.**error**("ERROR message!")

    logger.**warning**("abc")

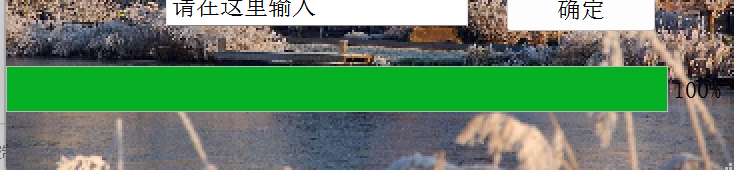
    logger.**info**("Finish your work!Good job!")

具体就不详细解释了，复用了一下Project1这部分的代码。把这部分代码添加到调用主界面类里面了。

最终实现的是：一旦开始调用主界面，就写日志。

### 3.2.9 进度条

进度条没有复用Project1部分的代码，而是使用了pyQT的progressBar方式，用setRange设置一共几步，用setValue设置完成到哪步。



### 3.2.10 读文件，显示二手房信息

读文件：

以北京为例说明：

*#北京'bj'*

data\_bj = **pd**.**read\_excel**(r"C:\Users\86183\Desktop\python-lxx\city\bj.xlsx")

*#print(data)*

Data\_Array\_bj = **np**.**array**(data\_bj).**flatten**()

bj = Data\_Array\_bj.**tolist**()

* flatten()是将一个读出来的表转换为一个一行的数据
* np.array是将数据转化为二维表
* 使用的读excel库是pandas库，功能比较强大

显示二手房信息：

for row in **range**(30):

                for col in **range**(9):

                    item = **QtWidgets**.**QTableWidgetItem**(f'{sh[row\*9+col]}' ) *#这里要注意f输出的方法*

*#item = QtWidgets.QTableWidgetItem(a[row\*7+col] )*

                    self.tableWidget.**setItem**(row,col,item)

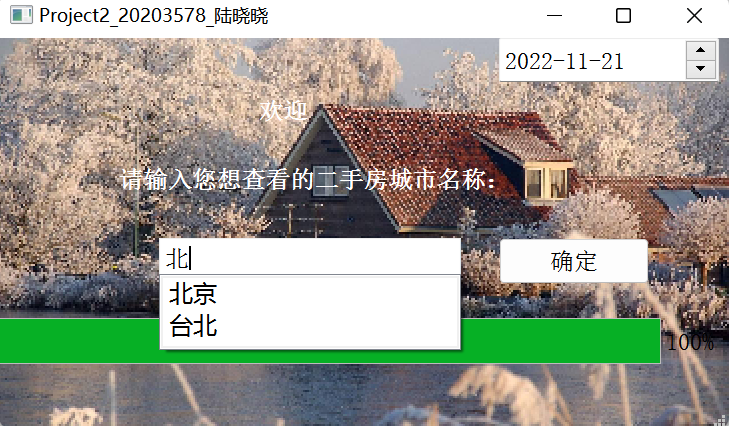
* 这个显示二手房的信息逻辑比较简单，就是将一个一维list算好下标放到table里面。
* 这里要特别注意的是f输出的方法，因为这个QTableWidgetItem只能设置为str类型。

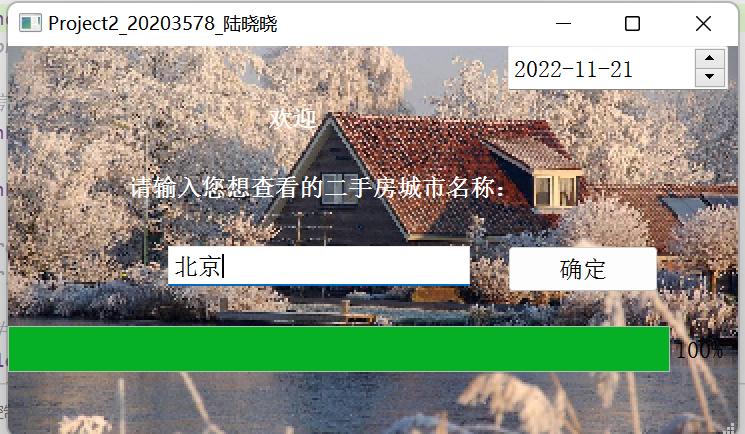
这是成都的二手房信息：



一共能显示30行信息，（因为我只爬了30行）

## 4.**运行结果展示**







也可以点击输入框，换个城市缩写输入： hhht比如呼和浩特



## 5.源代码

### demo\_main.py：（调用两个界面类）

1. **from** ctypes.wintypes **import** HPALETTE
2. **from** PyQt5.Qt **import** \*
3. **import** sys
4. **from** PyQt5.QtWidgets **import** QMainWindow,QApplication,QWidget
5. **from** Ui\_untitled1 **import** Ui\_Project2\_20203578  #导入写的界面类
6. **from** Ui\_untitled **import** Ui\_MainWindow  #导入写的界面类
7. **from** PyQt5 **import** QtCore
8. **import** logging #写日志

11. **class** MyMainWindow(QMainWindow,Ui\_Project2\_20203578):
13. #定义信号
14. \_signal = QtCore.pyqtSignal(str)
16. **def** \_\_init\_\_(self,parent =None):
17. super(MyMainWindow,self).\_\_init\_\_(parent)
18. self.setupUi(self)
19. self.setWindowTitle("Project2\_20203578\_陆晓晓")  #设置窗口标题
20. #初始化子界面
21. #ChildWin=SecondPage()
22. #super(ChildWin, self).\_\_init\_\_()
23. self.SecondPage=SecondPage()
24. #self.pushButton.clicked.connect(self.slot1)
25. #self.pushButton.clicked.connect(SecondPage.getData)
26. #self.setStyleSheet("background-color:blue;")
28. #设置背景
29. palette = QPalette()
30. pix=QPixmap("project2/city.jpg")
32. pix = pix.scaled(self.width(),self.height())
34. palette.setBrush(QPalette.Background,QBrush(pix))
35. self.setPalette(palette)

38. **def** slot1(self):
39. data\_str = self.lineEdit.text()
40. #print(data\_str) 信号发送没问题
41. #发送信号
42. self.\_signal.emit(data\_str)
44. **class** SecondPage(QMainWindow,Ui\_MainWindow): #这里
45. **def** \_\_init\_\_(self,parent =None):
46. super(SecondPage,self).\_\_init\_\_(parent)
47. self.setupUi(self)
48. self.setWindowTitle("二手房详细信息")  #设置窗口标题
50. '''''
51. def onClicked(self): #没有用
52. #连接信号
53. self.MainPage.\_signal.connect(self.getData)
54. print("connect")
56. def getData(self, parameter):
57. self.lineEdit.setText(parameter)
58. '''
60. **if** \_\_name\_\_ == "\_\_main\_\_":
61. app = QApplication(sys.argv)
62. myWin = MyMainWindow()
63. myWin.show()
65. #初始化第二个界面
66. myWinSecond = SecondPage()
67. #myWinSecond.show() 初始化第二个界面时候不能show，要触发了才可以
69. #主界面的信号连接副界面的槽函数
70. myWin.pushButton.clicked.connect(myWinSecond.show)
72. myWin.pushButton.clicked.connect(myWin.click\_my\_btn)
74. #str=Ui\_Project2\_20203578.get\_str()
75. #print(str)
77. #写日志文件部分
78. logger = logging.getLogger(\_\_name\_\_)
79. logger.setLevel(level = logging.INFO)
80. handler = logging.FileHandler("log.txt")
81. handler.setLevel(logging.INFO)
82. formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
83. handler.setFormatter(formatter)
84. logger.addHandler(handler)
86. logger.info("Start print log")
87. logger.debug("Do something")
88. logger.info("project2\_Information of Second hand house")
89. logger.error("ERROR message!")
90. logger.warning("abc")
91. logger.info("Finish your work!Good job!")
93. sys.exit(app.exec\_())

### Ui\_untitled.py：子界面类

1. # -\*- coding: utf-8 -\*-
3. # Form implementation generated from reading ui file 'c:\Users\86183\Desktop\python-lxx\project2\untitled.ui'
4. #
5. # Created by: PyQt5 UI code generator 5.15.7
6. #
7. # WARNING: Any manual changes made to this file will be lost when pyuic5 is
8. # run again.  Do not edit this file unless you know what you are doing.
10. **from** Ui\_untitled1 **import** Ui\_Project2\_20203578  #导入写的界面类
11. **import** time #时间计算
12. **from** rich.progress **import** track #rich库实现进度条
13. **import** numpy as np #构造一个小数的range 引入库实现读文件
14. **from** PyQt5 **import** QtCore, QtGui, QtWidgets
15. **import** pandas as pd
17. #读文件 并且存到一个表格里
18. city='sh'
19. data\_sh = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\sh.xlsx")
20. #print(data)
21. Data\_Array\_sh = np.array(data\_sh).flatten()
22. sh = Data\_Array\_sh.tolist()
23. #print(Data\_List)
25. #北京'bj'
26. data\_bj = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\bj.xlsx")
27. #print(data)
28. Data\_Array\_bj = np.array(data\_bj).flatten()
29. bj = Data\_Array\_bj.tolist()
30. #print(bj)
32. #重庆 cq
33. data\_cq = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\cq.xlsx")
34. #print(data)
35. Data\_Array\_cq = np.array(data\_cq).flatten()
36. cq = Data\_Array\_cq.tolist()
38. #天津 tj
39. data\_tj = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\tj.xlsx")
40. #print(data)
41. Data\_Array\_tj = np.array(data\_tj).flatten()
42. tj = Data\_Array\_tj.tolist()
44. #哈尔滨 heb
45. data\_heb = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\heb.xlsx")
46. #print(data)
47. Data\_Array\_heb = np.array(data\_heb).flatten()
48. heb = Data\_Array\_heb.tolist()
50. #长春 cc
51. data\_cc = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\cc.xlsx")
52. #print(data)
53. Data\_Array\_cc = np.array(data\_cc).flatten()
54. cc = Data\_Array\_cc.tolist()
56. #沈阳 sy
57. data\_sy = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\sy.xlsx")
58. #print(data)
59. Data\_Array\_sy = np.array(data\_sy).flatten()
60. sy = Data\_Array\_sy.tolist()
62. #呼和浩特 hhht
63. data\_hhht = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\hhht.xlsx")
64. #print(data)
65. Data\_Array\_hhht = np.array(data\_hhht).flatten()
66. hhht = Data\_Array\_hhht.tolist()
68. #石家庄 sjz
69. data\_sjz = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\sjz.xlsx")
70. #print(data)
71. Data\_Array\_sjz = np.array(data\_sjz).flatten()
72. sjz = Data\_Array\_sjz.tolist()
74. #乌鲁木齐 wlmq
75. data\_wlmq = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\wlmq.xlsx")
76. #print(data)
77. Data\_Array\_wlmq = np.array(data\_wlmq).flatten()
78. wlmq = Data\_Array\_wlmq.tolist()
80. #兰州 lz
81. data\_lz = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\lz.xlsx")
82. #print(data)
83. Data\_Array\_lz = np.array(data\_lz).flatten()
84. lz = Data\_Array\_lz.tolist()

87. #西宁 xn 链家二手房没有这个城市
88. #data\_xn = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\xn.xlsx")
89. #Data\_Array\_xn = np.array(data\_xn).flatten()
90. #xn = Data\_Array\_xn.tolist()
92. #西安 xa
93. data\_xa = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\xa.xlsx")
94. #print(data)
95. Data\_Array\_xa = np.array(data\_xa).flatten()
96. xa = Data\_Array\_xa.tolist()
98. #银川 yc 盐城
99. data\_yc = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\yc.xlsx")
100. #print(data)
101. Data\_Array\_yc = np.array(data\_yc).flatten()
102. yc = Data\_Array\_yc.tolist()
104. #郑州 zz
105. data\_zz = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\zz.xlsx")
106. #print(data)
107. Data\_Array\_zz = np.array(data\_zz).flatten()
108. zz = Data\_Array\_zz.tolist()
110. #济南 jn
111. data\_jn = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\jn.xlsx")
112. #print(data)
113. Data\_Array\_jn = np.array(data\_jn).flatten()
114. jn = Data\_Array\_jn.tolist()
116. #太原 ty
117. data\_ty = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\ty.xlsx")
118. #print(data)
119. Data\_Array\_ty = np.array(data\_ty).flatten()
120. ty = Data\_Array\_ty.tolist()
122. #合肥 hf
123. data\_hf = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\hf.xlsx")
124. #print(data)
125. Data\_Array\_hf = np.array(data\_hf).flatten()
126. hf = Data\_Array\_hf.tolist()
128. #长沙 cs
129. data\_cs = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\cs.xlsx")
130. #print(data)
131. Data\_Array\_cs = np.array(data\_cs).flatten()
132. cs = Data\_Array\_cs.tolist()
134. #南京 nj
135. data\_nj = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\nj.xlsx")
136. #print(data)
137. Data\_Array\_nj = np.array(data\_nj).flatten()
138. nj = Data\_Array\_nj.tolist()
140. #贵阳 gy
141. data\_gy = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\gy.xlsx")
142. #print(data)
143. Data\_Array\_gy = np.array(data\_gy).flatten()
144. gy = Data\_Array\_gy.tolist()
146. #昆明 km
147. data\_km = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\km.xlsx")
148. #print(data)
149. Data\_Array\_km = np.array(data\_km).flatten()
150. km = Data\_Array\_km.tolist()
152. #南宁 nn
153. data\_nn = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\nn.xlsx")
154. #print(data)
155. Data\_Array\_nn = np.array(data\_nn).flatten()
156. nn = Data\_Array\_nn.tolist()
158. #杭州 hz
159. data\_hz = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\hz.xlsx")
160. #print(data)
161. Data\_Array\_hz = np.array(data\_hz).flatten()
162. hz = Data\_Array\_hz.tolist()
164. #南昌 nc
165. data\_nc = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\nc.xlsx")
166. #print(data)
167. Data\_Array\_nc = np.array(data\_nc).flatten()
168. nc = Data\_Array\_nc.tolist()
170. #福州 fz
171. data\_fz = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\fz.xlsx")
172. #print(data)
173. Data\_Array\_fz = np.array(data\_fz).flatten()
174. fz = Data\_Array\_fz.tolist()
176. #海口 hk
177. data\_hk = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\hk.xlsx")
178. #print(data)
179. Data\_Array\_hk = np.array(data\_hk).flatten()
180. hk = Data\_Array\_hk.tolist()
182. #武汉 wh
183. data\_wh = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\wh.xlsx")
184. #print(data)
185. Data\_Array\_wh = np.array(data\_wh).flatten()
186. wh = Data\_Array\_wh.tolist()
188. #成都 cd
189. data\_cd = pd.read\_excel(r"C:\Users\86183\Desktop\python-lxx\city\cd.xlsx")
190. #print(data)
191. Data\_Array\_cd = np.array(data\_cd).flatten()
192. cd = Data\_Array\_cd.tolist()
194. **class** Ui\_MainWindow(object):
195. **def** setupUi(self, MainWindow):
196. MainWindow.setObjectName("MainWindow")
197. MainWindow.resize(953, 367)
198. self.centralwidget = QtWidgets.QWidget(MainWindow)
199. self.centralwidget.setObjectName("centralwidget")
200. self.label = QtWidgets.QLabel(self.centralwidget)
201. self.label.setGeometry(QtCore.QRect(140, 10, 251, 21))
202. self.label.setObjectName("label")
203. self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)
204. self.lineEdit.setGeometry(QtCore.QRect(380, 10, 61, 20))
205. self.lineEdit.setObjectName("lineEdit")
206. self.pushButton = QtWidgets.QPushButton(self.centralwidget)
207. self.pushButton.setGeometry(QtCore.QRect(450, 10, 91, 23))
208. self.pushButton.setObjectName("pushButton")
209. self.tableWidget = QtWidgets.QTableWidget(self.centralwidget)
210. self.tableWidget.setGeometry(QtCore.QRect(10, 40, 941, 301))
211. self.tableWidget.setObjectName("tableWidget")
212. self.tableWidget.setColumnCount(9)
213. self.tableWidget.setRowCount(30)
214. item = QtWidgets.QTableWidgetItem()
215. self.tableWidget.setVerticalHeaderItem(0, item)
216. item = QtWidgets.QTableWidgetItem()
217. self.tableWidget.setVerticalHeaderItem(1, item)
218. item = QtWidgets.QTableWidgetItem()
219. self.tableWidget.setVerticalHeaderItem(2, item)
220. item = QtWidgets.QTableWidgetItem()
221. self.tableWidget.setVerticalHeaderItem(3, item)
222. item = QtWidgets.QTableWidgetItem()
223. self.tableWidget.setVerticalHeaderItem(4, item)
224. item = QtWidgets.QTableWidgetItem()
225. self.tableWidget.setVerticalHeaderItem(5, item)
226. item = QtWidgets.QTableWidgetItem()
227. self.tableWidget.setVerticalHeaderItem(6, item)
228. item = QtWidgets.QTableWidgetItem()
229. self.tableWidget.setVerticalHeaderItem(7, item)
230. item = QtWidgets.QTableWidgetItem()
231. self.tableWidget.setVerticalHeaderItem(8, item)
232. item = QtWidgets.QTableWidgetItem()
233. self.tableWidget.setVerticalHeaderItem(9, item)
234. item = QtWidgets.QTableWidgetItem()
235. self.tableWidget.setVerticalHeaderItem(10, item)
236. item = QtWidgets.QTableWidgetItem()
237. self.tableWidget.setVerticalHeaderItem(11, item)
238. item = QtWidgets.QTableWidgetItem()
239. self.tableWidget.setVerticalHeaderItem(12, item)
240. item = QtWidgets.QTableWidgetItem()
241. self.tableWidget.setVerticalHeaderItem(13, item)
242. item = QtWidgets.QTableWidgetItem()
243. self.tableWidget.setVerticalHeaderItem(14, item)
244. item = QtWidgets.QTableWidgetItem()
245. self.tableWidget.setVerticalHeaderItem(15, item)
246. item = QtWidgets.QTableWidgetItem()
247. self.tableWidget.setVerticalHeaderItem(16, item)
248. item = QtWidgets.QTableWidgetItem()
249. self.tableWidget.setVerticalHeaderItem(17, item)
250. item = QtWidgets.QTableWidgetItem()
251. self.tableWidget.setVerticalHeaderItem(18, item)
252. item = QtWidgets.QTableWidgetItem()
253. self.tableWidget.setVerticalHeaderItem(19, item)
254. item = QtWidgets.QTableWidgetItem()
255. self.tableWidget.setVerticalHeaderItem(20, item)
256. item = QtWidgets.QTableWidgetItem()
257. self.tableWidget.setVerticalHeaderItem(21, item)
258. item = QtWidgets.QTableWidgetItem()
259. self.tableWidget.setVerticalHeaderItem(22, item)
260. item = QtWidgets.QTableWidgetItem()
261. self.tableWidget.setVerticalHeaderItem(23, item)
262. item = QtWidgets.QTableWidgetItem()
263. self.tableWidget.setVerticalHeaderItem(24, item)
264. item = QtWidgets.QTableWidgetItem()
265. self.tableWidget.setVerticalHeaderItem(25, item)
266. item = QtWidgets.QTableWidgetItem()
267. self.tableWidget.setVerticalHeaderItem(26, item)
268. item = QtWidgets.QTableWidgetItem()
269. self.tableWidget.setVerticalHeaderItem(27, item)
270. item = QtWidgets.QTableWidgetItem()
271. self.tableWidget.setVerticalHeaderItem(28, item)
272. item = QtWidgets.QTableWidgetItem()
273. self.tableWidget.setVerticalHeaderItem(29, item)
274. item = QtWidgets.QTableWidgetItem()
275. self.tableWidget.setHorizontalHeaderItem(0, item)
276. item = QtWidgets.QTableWidgetItem()
277. self.tableWidget.setHorizontalHeaderItem(1, item)
278. item = QtWidgets.QTableWidgetItem()
279. self.tableWidget.setHorizontalHeaderItem(2, item)
280. item = QtWidgets.QTableWidgetItem()
281. self.tableWidget.setHorizontalHeaderItem(3, item)
282. item = QtWidgets.QTableWidgetItem()
283. self.tableWidget.setHorizontalHeaderItem(4, item)
284. item = QtWidgets.QTableWidgetItem()
285. self.tableWidget.setHorizontalHeaderItem(5, item)
286. item = QtWidgets.QTableWidgetItem()
287. self.tableWidget.setHorizontalHeaderItem(6, item)
288. item = QtWidgets.QTableWidgetItem()
289. self.tableWidget.setHorizontalHeaderItem(7, item)
290. item = QtWidgets.QTableWidgetItem()
291. self.tableWidget.setHorizontalHeaderItem(8, item)
292. item = QtWidgets.QTableWidgetItem()
293. brush = QtGui.QBrush(QtGui.QColor(0, 170, 255))
294. brush.setStyle(QtCore.Qt.NoBrush)
295. item.setForeground(brush)
296. self.tableWidget.setItem(0, 0, item)
297. MainWindow.setCentralWidget(self.centralwidget)
298. self.statusbar = QtWidgets.QStatusBar(MainWindow)
299. self.statusbar.setObjectName("statusbar")
300. MainWindow.setStatusBar(self.statusbar)
302. self.retranslateUi(MainWindow)
303. QtCore.QMetaObject.connectSlotsByName(MainWindow)
305. self.pushButton.clicked.connect(self.click\_my\_btn) #点击按钮，能够返回文本
307. **def** click\_my\_btn(self):
308. str\_tmp=self.lineEdit.text()
309. #print(str\_tmp)
310. **if** str\_tmp == 'sh':
311. #city = 'sh'
312. **for** row **in** range(30):
313. **for** col **in** range(9):
314. item = QtWidgets.QTableWidgetItem(f'{sh[row\*9+col]}' ) #这里要注意f输出的方法
315. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
316. self.tableWidget.setItem(row,col,item)
317. **elif** str\_tmp == 'bj':
318. city = 'bj'
319. **for** row **in** range(30):
320. **for** col **in** range(9):
321. item = QtWidgets.QTableWidgetItem(f'{bj[row\*9+col]}' ) #这里要注意f输出的方法
322. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
323. self.tableWidget.setItem(row,col,item)
324. **elif** str\_tmp == 'tj':
325. city = 'tj'
326. **for** row **in** range(30):
327. **for** col **in** range(9):
328. item = QtWidgets.QTableWidgetItem(f'{tj[row\*9+col]}' ) #这里要注意f输出的方法
329. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
330. self.tableWidget.setItem(row,col,item)
331. **elif** str\_tmp == 'cq':
332. city = 'cq'
333. **for** row **in** range(30):
334. **for** col **in** range(9):
335. item = QtWidgets.QTableWidgetItem(f'{cq[row\*9+col]}' ) #这里要注意f输出的方法
336. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
337. self.tableWidget.setItem(row,col,item)
338. **elif** str\_tmp == 'heb':
339. city = 'heb'
340. **for** row **in** range(30):
341. **for** col **in** range(9):
342. item = QtWidgets.QTableWidgetItem(f'{heb[row\*9+col]}' ) #这里要注意f输出的方法
343. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
344. self.tableWidget.setItem(row,col,item)
345. **elif** str\_tmp == 'cc':
346. city = 'cc'
347. **for** row **in** range(30):
348. **for** col **in** range(9):
349. item = QtWidgets.QTableWidgetItem(f'{cc[row\*9+col]}' ) #这里要注意f输出的方法
350. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
351. self.tableWidget.setItem(row,col,item)
352. **elif** str\_tmp == 'sy':
353. city = 'sy'
354. **for** row **in** range(30):
355. **for** col **in** range(9):
356. item = QtWidgets.QTableWidgetItem(f'{sy[row\*9+col]}' ) #这里要注意f输出的方法
357. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
358. self.tableWidget.setItem(row,col,item)
359. **elif** str\_tmp == 'hhht':
360. city = 'hhht'
361. **for** row **in** range(30):
362. **for** col **in** range(9):
363. item = QtWidgets.QTableWidgetItem(f'{hhht[row\*9+col]}' ) #这里要注意f输出的方法
364. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
365. self.tableWidget.setItem(row,col,item)
366. **elif** str\_tmp == 'sjz':
367. city = 'sjz'
368. **for** row **in** range(30):
369. **for** col **in** range(9):
370. item = QtWidgets.QTableWidgetItem(f'{sjz[row\*9+col]}' ) #这里要注意f输出的方法
371. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
372. self.tableWidget.setItem(row,col,item)
373. **elif** str\_tmp == 'wlmq':
374. city = 'wlmq'
375. **for** row **in** range(30):
376. **for** col **in** range(9):
377. item = QtWidgets.QTableWidgetItem(f'{wlmq[row\*9+col]}' ) #这里要注意f输出的方法
378. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
379. self.tableWidget.setItem(row,col,item)
380. **elif** str\_tmp == 'lz':
381. city = 'lz'
382. **for** row **in** range(30):
383. **for** col **in** range(9):
384. item = QtWidgets.QTableWidgetItem(f'{lz[row\*9+col]}' ) #这里要注意f输出的方法
385. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
386. self.tableWidget.setItem(row,col,item)
387. **elif** str\_tmp == 'xa':
388. city = 'xa'
389. **for** row **in** range(30):
390. **for** col **in** range(9):
391. item = QtWidgets.QTableWidgetItem(f'{xa[row\*9+col]}' ) #这里要注意f输出的方法
392. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
393. self.tableWidget.setItem(row,col,item)
394. **elif** str\_tmp == 'yc':
395. city = 'yc'
396. **for** row **in** range(30):
397. **for** col **in** range(9):
398. item = QtWidgets.QTableWidgetItem(f'{yc[row\*9+col]}' ) #这里要注意f输出的方法
399. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
400. self.tableWidget.setItem(row,col,item)
401. **elif** str\_tmp == 'zz':
402. city = 'zz'
403. **for** row **in** range(30):
404. **for** col **in** range(9):
405. item = QtWidgets.QTableWidgetItem(f'{zz[row\*9+col]}' ) #这里要注意f输出的方法
406. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
407. self.tableWidget.setItem(row,col,item)
408. **elif** str\_tmp == 'jn':
409. city = 'jn'
410. **for** row **in** range(30):
411. **for** col **in** range(9):
412. item = QtWidgets.QTableWidgetItem(f'{jn[row\*9+col]}' ) #这里要注意f输出的方法
413. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
414. self.tableWidget.setItem(row,col,item)
415. **elif** str\_tmp == 'ty':
416. city = 'ty'
417. **for** row **in** range(30):
418. **for** col **in** range(9):
419. item = QtWidgets.QTableWidgetItem(f'{ty[row\*9+col]}' ) #这里要注意f输出的方法
420. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
421. self.tableWidget.setItem(row,col,item)
422. **elif** str\_tmp == 'hf':
423. city = 'hf'
424. **for** row **in** range(30):
425. **for** col **in** range(9):
426. item = QtWidgets.QTableWidgetItem(f'{hf[row\*9+col]}' ) #这里要注意f输出的方法
427. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
428. self.tableWidget.setItem(row,col,item)
429. **elif** str\_tmp == 'cs':
430. city = 'cs'
431. **for** row **in** range(30):
432. **for** col **in** range(9):
433. item = QtWidgets.QTableWidgetItem(f'{cs[row\*9+col]}' ) #这里要注意f输出的方法
434. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
435. self.tableWidget.setItem(row,col,item)
436. **elif** str\_tmp == 'nj':
437. city = 'nj'
438. **for** row **in** range(30):
439. **for** col **in** range(9):
440. item = QtWidgets.QTableWidgetItem(f'{nj[row\*9+col]}' ) #这里要注意f输出的方法
441. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
442. self.tableWidget.setItem(row,col,item)
443. **elif** str\_tmp == 'gy':
444. city = 'gy'
445. **for** row **in** range(30):
446. **for** col **in** range(9):
447. item = QtWidgets.QTableWidgetItem(f'{gy[row\*9+col]}' ) #这里要注意f输出的方法
448. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
449. self.tableWidget.setItem(row,col,item)
450. **elif** str\_tmp == 'km':
451. city = 'km'
452. **for** row **in** range(30):
453. **for** col **in** range(9):
454. item = QtWidgets.QTableWidgetItem(f'{km[row\*9+col]}' ) #这里要注意f输出的方法
455. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
456. self.tableWidget.setItem(row,col,item)
457. **elif** str\_tmp == 'nn':
458. city = 'nn'
459. **for** row **in** range(30):
460. **for** col **in** range(9):
461. item = QtWidgets.QTableWidgetItem(f'{nn[row\*9+col]}' ) #这里要注意f输出的方法
462. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
463. self.tableWidget.setItem(row,col,item)
464. **elif** str\_tmp == 'hz':
465. city = 'hz'
466. **for** row **in** range(30):
467. **for** col **in** range(9):
468. item = QtWidgets.QTableWidgetItem(f'{hz[row\*9+col]}' ) #这里要注意f输出的方法
469. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
470. self.tableWidget.setItem(row,col,item)
471. **elif** str\_tmp == 'nc':
472. city = 'nc'
473. **for** row **in** range(30):
474. **for** col **in** range(9):
475. item = QtWidgets.QTableWidgetItem(f'{nc[row\*9+col]}' ) #这里要注意f输出的方法
476. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
477. self.tableWidget.setItem(row,col,item)
478. **elif** str\_tmp == 'fz':
479. city = 'fz'
480. **for** row **in** range(30):
481. **for** col **in** range(9):
482. item = QtWidgets.QTableWidgetItem(f'{fz[row\*9+col]}' ) #这里要注意f输出的方法
483. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
484. self.tableWidget.setItem(row,col,item)
485. **elif** str\_tmp == 'hk':
486. city = 'hk'
487. **for** row **in** range(30):
488. **for** col **in** range(9):
489. item = QtWidgets.QTableWidgetItem(f'{hk[row\*9+col]}' ) #这里要注意f输出的方法
490. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
491. self.tableWidget.setItem(row,col,item)
492. **elif** str\_tmp == 'wh':
493. city = 'wh'
494. **for** row **in** range(30):
495. **for** col **in** range(9):
496. item = QtWidgets.QTableWidgetItem(f'{wh[row\*9+col]}' ) #这里要注意f输出的方法
497. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
498. self.tableWidget.setItem(row,col,item)
499. **elif** str\_tmp == 'cd':
500. city = 'cd'
501. **for** row **in** range(30):
502. **for** col **in** range(9):
503. item = QtWidgets.QTableWidgetItem(f'{cd[row\*9+col]}' ) #这里要注意f输出的方法
504. #item = QtWidgets.QTableWidgetItem(a[row\*7+col] )
505. self.tableWidget.setItem(row,col,item)
507. **def** retranslateUi(self, MainWindow):
508. \_translate = QtCore.QCoreApplication.translate
509. MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))
510. self.label.setText(\_translate("MainWindow", "你要查找的二手房信息的城市名称首字母是："))
511. self.pushButton.setText(\_translate("MainWindow", "更新数据表格"))
512. item = self.tableWidget.verticalHeaderItem(0)
513. item.setText(\_translate("MainWindow", "1"))
514. item = self.tableWidget.verticalHeaderItem(1)
515. item.setText(\_translate("MainWindow", "2"))
516. item = self.tableWidget.verticalHeaderItem(2)
517. item.setText(\_translate("MainWindow", "3"))
518. item = self.tableWidget.verticalHeaderItem(3)
519. item.setText(\_translate("MainWindow", "4"))
520. item = self.tableWidget.verticalHeaderItem(4)
521. item.setText(\_translate("MainWindow", "5"))
522. item = self.tableWidget.verticalHeaderItem(5)
523. item.setText(\_translate("MainWindow", "6"))
524. item = self.tableWidget.verticalHeaderItem(6)
525. item.setText(\_translate("MainWindow", "7"))
526. item = self.tableWidget.verticalHeaderItem(7)
527. item.setText(\_translate("MainWindow", "8"))
528. item = self.tableWidget.verticalHeaderItem(8)
529. item.setText(\_translate("MainWindow", "9"))
530. item = self.tableWidget.verticalHeaderItem(9)
531. item.setText(\_translate("MainWindow", "10"))
532. item = self.tableWidget.verticalHeaderItem(10)
533. item.setText(\_translate("MainWindow", "11"))
534. item = self.tableWidget.verticalHeaderItem(11)
535. item.setText(\_translate("MainWindow", "12"))
536. item = self.tableWidget.verticalHeaderItem(12)
537. item.setText(\_translate("MainWindow", "13"))
538. item = self.tableWidget.verticalHeaderItem(13)
539. item.setText(\_translate("MainWindow", "14"))
540. item = self.tableWidget.verticalHeaderItem(14)
541. item.setText(\_translate("MainWindow", "15"))
542. item = self.tableWidget.verticalHeaderItem(15)
543. item.setText(\_translate("MainWindow", "16"))
544. item = self.tableWidget.verticalHeaderItem(16)
545. item.setText(\_translate("MainWindow", "17"))
546. item = self.tableWidget.verticalHeaderItem(17)
547. item.setText(\_translate("MainWindow", "18"))
548. item = self.tableWidget.verticalHeaderItem(18)
549. item.setText(\_translate("MainWindow", "19"))
550. item = self.tableWidget.verticalHeaderItem(19)
551. item.setText(\_translate("MainWindow", "20"))
552. item = self.tableWidget.verticalHeaderItem(20)
553. item.setText(\_translate("MainWindow", "21"))
554. item = self.tableWidget.verticalHeaderItem(21)
555. item.setText(\_translate("MainWindow", "22"))
556. item = self.tableWidget.verticalHeaderItem(22)
557. item.setText(\_translate("MainWindow", "23"))
558. item = self.tableWidget.verticalHeaderItem(23)
559. item.setText(\_translate("MainWindow", "24"))
560. item = self.tableWidget.verticalHeaderItem(24)
561. item.setText(\_translate("MainWindow", "25"))
562. item = self.tableWidget.verticalHeaderItem(25)
563. item.setText(\_translate("MainWindow", "26"))
564. item = self.tableWidget.verticalHeaderItem(26)
565. item.setText(\_translate("MainWindow", "27"))
566. item = self.tableWidget.verticalHeaderItem(27)
567. item.setText(\_translate("MainWindow", "28"))
568. item = self.tableWidget.verticalHeaderItem(28)
569. item.setText(\_translate("MainWindow", "29"))
570. item = self.tableWidget.verticalHeaderItem(29)
571. item.setText(\_translate("MainWindow", "30"))
572. item = self.tableWidget.horizontalHeaderItem(0)
573. item.setText(\_translate("MainWindow", "地址"))
574. item = self.tableWidget.horizontalHeaderItem(1)
575. item.setText(\_translate("MainWindow", "户型"))
576. item = self.tableWidget.horizontalHeaderItem(2)
577. item.setText(\_translate("MainWindow", "面积"))
578. item = self.tableWidget.horizontalHeaderItem(3)
579. item.setText(\_translate("MainWindow", "朝向"))
580. item = self.tableWidget.horizontalHeaderItem(4)
581. item.setText(\_translate("MainWindow", "装修"))
582. item = self.tableWidget.horizontalHeaderItem(5)
583. item.setText(\_translate("MainWindow", "楼层"))
584. item = self.tableWidget.horizontalHeaderItem(6)
585. item.setText(\_translate("MainWindow", "建立"))
586. item = self.tableWidget.horizontalHeaderItem(7)
587. item.setText(\_translate("MainWindow", "结构"))
588. item = self.tableWidget.horizontalHeaderItem(8)
589. item.setText(\_translate("MainWindow", "价格(万元)"))
590. \_\_sortingEnabled = self.tableWidget.isSortingEnabled()
591. self.tableWidget.setSortingEnabled(False)

594. #a=list(range(0,210)) 测试列表
595. #将文字一步步输入到表格
596. #for i in range(30):
597. #    for j in range(8):
598. #        self.tableWidget.setItem(0, 0, a[(i+1)\*8+j])


602. self.tableWidget.setSortingEnabled(\_\_sortingEnabled)

### Ui\_untitled1.py：主界面类

1. # -\*- coding: utf-8 -\*-
3. # Form implementation generated from reading ui file 'c:\Users\86183\Desktop\python-lxx\project2\untitled1.ui'
4. #
5. # Created by: PyQt5 UI code generator 5.15.7
6. #
7. # WARNING: Any manual changes made to this file will be lost when pyuic5 is
8. # run again.  Do not edit this file unless you know what you are doing.
10. **import** sys
11. **import** pinyin #另外使用pip安装pinyin包
13. **from** PyQt5.QtWidgets **import** QCompleter
14. **from** PyQt5 **import** QtCore, QtGui, QtWidgets
16. **import** time

19. **class** Ui\_Project2\_20203578(object):
20. **def** setupUi(self, Project2\_20203578):
21. Project2\_20203578.setObjectName("Project2\_20203578")
22. Project2\_20203578.resize(365, 195)
23. self.centralwidget = QtWidgets.QWidget(Project2\_20203578)
24. self.centralwidget.setObjectName("centralwidget")
25. self.pushButton = QtWidgets.QPushButton(self.centralwidget)
26. self.pushButton.setGeometry(QtCore.QRect(250, 100, 75, 23))
27. self.pushButton.setObjectName("pushButton")
28. self.label = QtWidgets.QLabel(self.centralwidget)
29. self.label.setGeometry(QtCore.QRect(130, 30, 54, 12))
30. self.label.setObjectName("label")
31. self.label\_2 = QtWidgets.QLabel(self.centralwidget)
32. self.label\_2.setGeometry(QtCore.QRect(60, 60, 191, 21))
33. self.label\_2.setObjectName("label\_2")
34. self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)
35. self.lineEdit.setGeometry(QtCore.QRect(80, 100, 151, 20))
36. self.lineEdit.setObjectName("lineEdit")
37. self.dateEdit = QtWidgets.QDateEdit(self.centralwidget)
38. self.dateEdit.setGeometry(QtCore.QRect(250, 0, 110, 22))
39. self.dateEdit.setDate(QtCore.QDate(2022, 11, 21))
40. self.dateEdit.setObjectName("dateEdit")
41. self.progressBar = QtWidgets.QProgressBar(self.centralwidget)
42. self.progressBar.setGeometry(QtCore.QRect(0, 140, 361, 23))
43. self.progressBar.setProperty("value", 24)
44. self.progressBar.setObjectName("progressBar")
45. Project2\_20203578.setCentralWidget(self.centralwidget)
46. self.statusbar = QtWidgets.QStatusBar(Project2\_20203578)
47. self.statusbar.setObjectName("statusbar")
48. Project2\_20203578.setStatusBar(self.statusbar)
50. self.retranslateUi(Project2\_20203578)
51. QtCore.QMetaObject.connectSlotsByName(Project2\_20203578)
53. #self.combo\_box = QtWidgets.QComboBox(self, minimumWidth=150)
54. #self.combo\_box.setEditable(True)
56. self.progressBar.setRange(0, 3)
57. time.sleep(1)
58. self.progressBar.setValue(1)
59. time.sleep(0.8)
60. self.progressBar.setValue(2)
61. time.sleep(0.5)
62. self.progressBar.setValue(3)
64. self.auto\_complete()
66. #模糊匹配
67. #items\_list=['北京','上海','天津','重庆','哈尔滨','长春','沈阳','呼和浩特','石家庄','乌鲁木齐','兰州','西宁','西安','银川','郑州','济南','太原','合肥','长沙','武汉','南京','成都','贵阳','昆明','南宁','拉萨','杭州','南昌','广州','福州','台北','海口','香港','澳门']
68. '''''
69. completer = QCompleter(items\_list)
70. completer.setFilterMode(QtCore.Qt.MatchContains) #设置匹配模式为内容匹配
71. completer.setCompletionMode(QtWidgets.QCompleter.PopupCompletion) #设置不全模式为全显选项补全
72. '''
74. **def** auto\_complete(self):
75. items\_list=['北京','上海','天津','重庆','哈尔滨','长春','沈阳','呼和浩特','石家庄','乌鲁木齐','兰州','西宁','西安','银川','郑州','济南','太原','合肥','长沙','武汉','南京','成都','贵阳','昆明','南宁','拉萨','杭州','南昌','广州','福州','台北','海口','香港','澳门']
76. self.completer = QtWidgets.QCompleter(items\_list)
78. # 设置匹配模式  Qt.MatchContains 内容匹配
79. self.completer.setFilterMode(QtCore.Qt.MatchContains)
81. # 设置补全模式  QCompleter.PopupCompletion 弹出选项补全（默认）
82. self.completer.setCompletionMode(QtWidgets.QCompleter.PopupCompletion)
84. self.lineEdit.setCompleter(self.completer)
85. #self.combo\_box.setCompleter(self.completer)
87. **def** click\_my\_btn(self):
88. **def** getStrAllAplha(str):
89. **return** pinyin.get\_initial(str, delimiter="")
90. **print**("你要查看的城市的首字母是：")
91. str\_tmp=self.lineEdit.text()
92. #print(str\_tmp)
93. self.str=getStrAllAplha(str\_tmp)
94. **print**(self.str)
96. #def get\_str():
97. #    return str

100. **def** retranslateUi(self, Project2\_20203578):
101. \_translate = QtCore.QCoreApplication.translate
102. Project2\_20203578.setWindowTitle(\_translate("Project2\_20203578", "MainWindow"))
103. self.pushButton.setText(\_translate("Project2\_20203578", "确定"))
104. self.label.setText(\_translate("Project2\_20203578", "欢迎"))
105. self.label\_2.setText(\_translate("Project2\_20203578", "请输入您想查看的二手房城市名称："))
107. self.label.setStyleSheet("font: bold;color:white")
108. self.label\_2.setStyleSheet("font: bold;color:white")
110. self.lineEdit.setText(\_translate("Project2\_20203578", "请在这里输入"))

## 6.心得体会

这是第二次python实验，和上一次做的project1内容基本有很大的不同，由于对tkinter库不太熟悉设计出来的界面比较丑，所以我改用了PyQT，QT的文档比较多，并且之前在学习C++的时候我已经熟悉过QT的使用。

Python 标准库和第三库众多，功能强大，这一点是c、c++比不了的，代码的封装程度很高，并且相关文档丰富，基本对应的库看一下文档就能会用了。在c、c++中100行的代码在python甚至只需要几行就能实现。

在写project2花费的时间比projec1 的时间要多很多，首先要回忆起来QT的使用方法，并且PyQT的语法规范还是和C++的QT Creator有很多不同之处的，之后便是利用信号和槽机制来完成点击主界面的按钮来跳出子界面，打印表格等等。

同时，我认识到自己对于python语言的练习还是不足的，之后会利用课后时间多加练习，这样也是为期末考试做准备。