西安电子科技大学 XIDIAN UNIVERSITY | 软件学院 School of Software

并 行 计 算

课程实验报告

实验名称： Python 基础编程

任课教师： 徐悦甦老师

课程班级： 15 级-云计算方向

学号姓名： 15130120141-孙 晖

提交日期： 2018 年 6 月 7 日

目录

软件学院本科生《并行计算》

课程实验报告

一、实验名称

Python 基础编程（共两道题）

第一题：使用 Python 基础编程统计两个文件中单词“book”出现的总次数；

第二题：使用 Python 基础编程，完成快速排序函数的编写。

二、实验日期

2018 年 06 月 06 日　软件学院实验室 G346

三、实验学生

15130120141 孙晖

四、实验目的

本次实验有两道编程题，老师旨在希望我们通过这两道编程题初步掌握对 Python 的使用，尤其是 Python 在文件操控方面的使用。从老师要求所有结果输出到文件中就可以看出 Python 文件操作的重要性。通过两道题掌握 Python 的基本语法，为下一次实验的 Python 并行做准备。

五、实验内容

本次实验要求学生独自完成，不存在组队的情况。

[1] 第一题：

使用 Python 基础编程，统计两个文件中单词“book”出现的总次数，并将结果写入到文件中（文件名自己决定）。

题目描述：有两个文件，file_1.dat 与 file_2.dat（文件类型也可以是其它文本文件内容类型，如.txt）。

文件内容如下：

file_1.dat 内容

and,with,we,me,university,with,book,computer,country,book

file_2.dat 内容

bag,boy,book,school,teacher,student,book,book

要求：

正确使用 Python 的文件操作，分别统计出 file_1.dat 中出现 3 次，file_2.dat 中出现两次，然后统计出“book”出现的总次数（5 次）

将总次数写入文件中（文件名自己决定）。

[2] 第二题：

使用 Python 基础编程，完成快速排序函数的编写。

输入：3,7,12,5,3,10,11,9,4,2,4

输出：经过快排，正确的排序结果，如

2,3,3,4,4,5,7,9,10,11,12

要求：

使用 Python 实现快速排序的函数；

将排序结果写入到文件中（文件名自己决定）。

六、程序思路、结构

[1] 第一题：

面对题目的第一反应是使用正则表达式，因为方便快捷。re.findall("book",文件)就直接找到文件中的 book，再使用 len 方法就直接返回该文件中“book”的数量。验收时还担心老师会认为我做的不合格，然而老师还夸奖我做的很简单，很开心。

程序思路：就是调用 re.findall 这个方法把“book”这个单词从文件里摘出来，传入“book”和文件就可以了，再使用 len 函数计数，返回数字就可以了。

结构：就是顺序写下来，没有自己 def 函数。

[2] 第二题：

程序思路：快排已经是非常熟练的一个算法了。快排在算法课上我有一个详细而形象化的总结，就是我们以整个数组的最后一个元素为 x 为标致，整个数组从第一个元素到 n-1 个元素都和这个 x 去比较，然后整个数组分为四个区域，分为 ABCD 四个区域，其中：

D 就是元素 X；

C 就是还没有匹配到的元素；

B 就是大于 X 的元素；

A 就是小于 X 的元素。

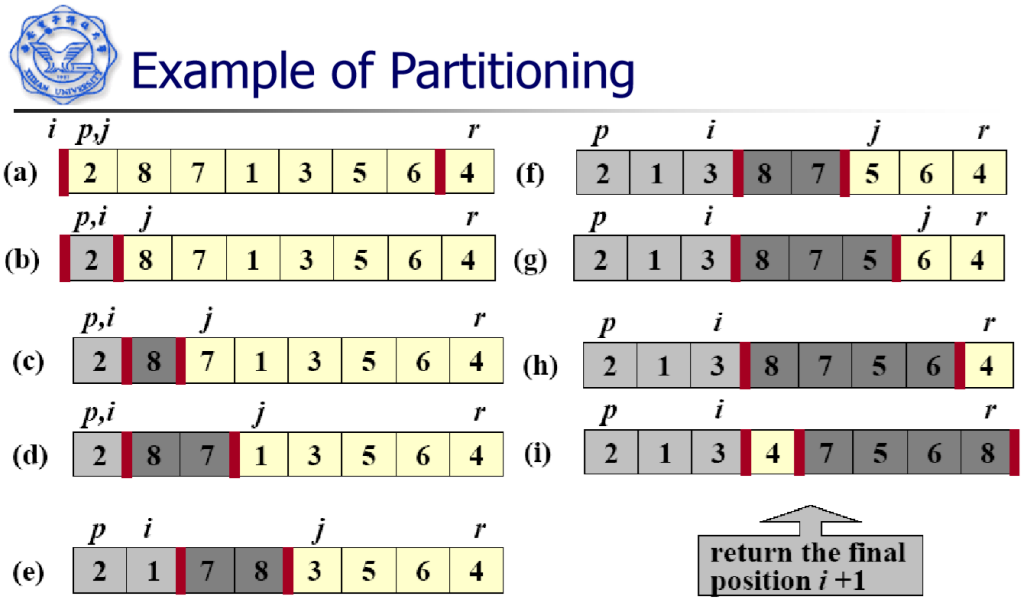
这样看来可以有很多的变种，比如把数组第一个元素选择为 X，把 A 用于存放大于 X 的元素，B 用于存放小于 X 的元素。

然后就是迭代思想。这里我想说明一下迭代和递归的区别：递归就是调用自己，比

如函数 A 一直调用函数 A，迭代就是函数 A 不停调用函数 B。

所以把上一次上述的行为不停的递归就能够做到最后实现排好序。这里需要注意的是要把 X 和 i+1 最后交换一下，还有就是递归的结束标志要写好。快排需要的参数有三个，一个是数组，两个 int 型用于表示数组开始和结束的下标，当然在 python 中就不需要说明类型了。

如图所示：



然后工作就转化为如何使用 python 去实现快排，以此来掌握 python 的一些语言特点，使得自己能够较为熟练的使用 python。

七、程序代码

[1] 第一题代码：

```
#coding=utf-8
import re
import collections

txt1 = open("file_1.txt", "r").read()

txt2 = open("file_2.txt", "r").read()

str1 = "there are %d books in file_1 \n" %(len(re.findall("book", txt1)))
str2 = "there are %d books in file_2\n" %(len(re.findall("book", txt2)))
str3 = "there are %d books in all files\n" %(len(re.findall("book", txt2))+len(re.findall("book", txt1)))
```

```
print (str1)
print (str2)
print (str3)

a = open('D:\\大三下学期\\并行计算\\实验三\\file.txt','a')
a.write(str1)
a.write(str2)
a.write(str3)
a.close()
```

[2] }第二题代码：

```
#coding=utf-8

def QuickSort(arr,firstIndex,lastIndex):
    if firstIndex<lastIndex:
        divIndex=Partition(arr,firstIndex,lastIndex)

        QuickSort(arr,firstIndex,divIndex)
        QuickSort(arr,divIndex+1,lastIndex)
    else:
        return



def Partition(arr,firstIndex,lastIndex):
    i=firstIndex-1
    for j in range(firstIndex,lastIndex):
        if arr[j]<=arr[lastIndex]:
            i=i+1
            arr[i],arr[j]=arr[j],arr[i]
    arr[i+1],arr[lastIndex]=arr[lastIndex],arr[i+1]
    return i
arr=[3,7,12,5,3,10,11,9,4,2,4]

QuickSort(arr,0,len(arr)-1)
b = open('D:\\大三下学期\\并行计算\\实验三\\quicksort.txt','a')
print(arr,file=b)
print(arr)
b.close
```

八、实验结果

[1] 第一题结果：

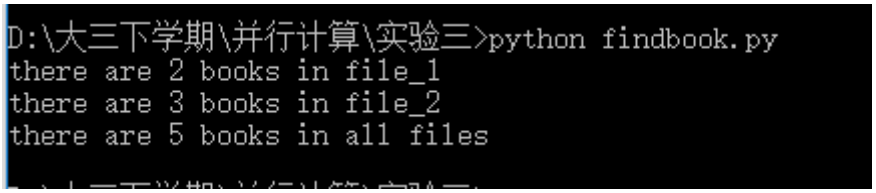


图 1 这是在命令行里显示两个文件中 book 的个数

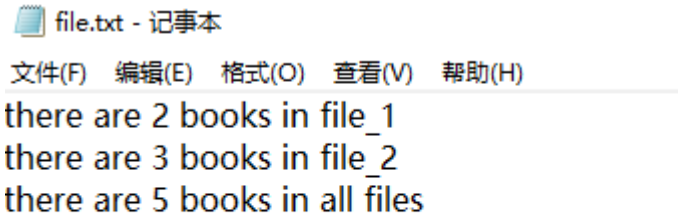
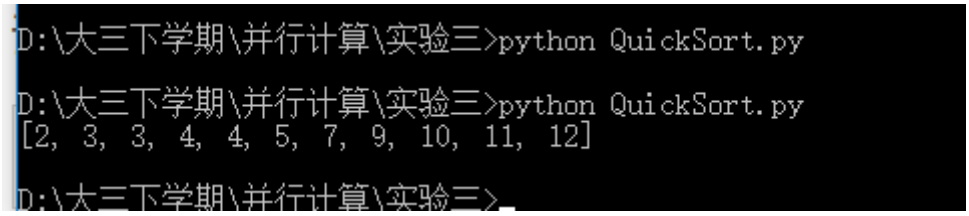


图 2 这是在文件中显示的结果（因为题目要求把结果输出到文件中）

[2] 第二题结果:




```
arr=[3,7,12,5,3,10,11,9,4,2,4]

QuickSort(arr,0,len(arr)-1)
b = open('D:\\大三下学期\\并行计算\\实验三\\quicksort.txt','a')
print(arr,file=b)
print(arr)
b.close
```

图 3 这是在命令行里运行快排后显示的结果

图 4 这是我在程序里规定好的数组

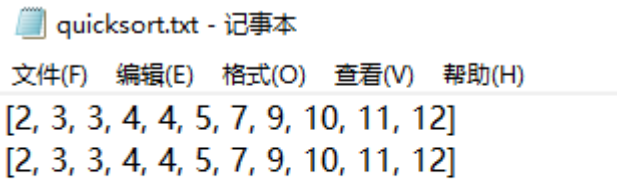


图 5 这是将快排运行结果写入到文件中

九、总结建议

[1] Python 算得上我从大一就接触的语言了，那个时候 15 年 16 年的时候，python2.x 还是比较主流的，短短时间，python3 都成为主流了。那个时候还在为 python2 和

pythonn3 语法的不同而头疼，直到如今，还是有时候写 print 忘记加括号。

[2] 感觉用 python 写过爬虫的话，在面对第一题时首先想到的肯定是正则表达式了。所以我就用正则表达式写了。

[3] 总结整个实验发现，快排这一点于我而言也不存在难点，算法已经是非常熟悉了，这里用到的 python 语法知识也不难，总的来说，可能两道题是远远不够学习或者说回忆 python 语言相关知识的。如果想要只是通过这两道题去掌握 python 语言，那显然是痴心妄想。所以，我趁着这个机会以老师 PPT 为线索，自己再总结一遍，然后上传到 github 用于以后自己学习和回忆。