SouthWest University

Lab report

Couse name Principle of programming

Semester 2021 - 2022 - 1

Grade 2021 Class 34

Name ChengYi Student No. 222021321102110

Tutor Li Ya

School of Computer and Information Science

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Lab 2 | | | **Practicing on File.** | | | | |
| Issue Date | | 2020年12月22日 | | | experimental types | □validation experiment,  □comprehensive experiment  ☑design experiment | |
| Goal  • You will practice the reading, writing, and printing of file.  Assignment  Given a csv file(score.csv) which contain scores of some students(see figure 1).    Figure 1  Write a program based on the sample code to  calculate the total and average score of all students, average score of each subject and write the results into a new score file(newscore.csv) as in the following figure:    Write another program to provide an inquiry function which can prompt user to enter student’s name, and then prints the results of the query. | | | | | | | |
| * Experimental contents and process   1.The execution of the code requires a CSV file. I learned how to use the for loop to print and calculate each line and create a new list to print the last line.  2.The topic also requires a new CSV file. I learned to use the while loop to make sure the name is correct and to use the if statement to find out if the name is in the file. I use the dictionary “dict” for easy indexing and return the correct results. | | | | | | | |
| * Screen shots of the Python IDLE showing the output results of running your Lab code.   Program 1:  score-database.csv      score-database-new.csv      Program 2: | | | | | | | |
| * Lab Code   Program 1:  import os  import csv  def fileWrite(fileRead, newFileRead):      '''Pass in the original and new files, complete the contents, and write to the file'''      lastLine = ['', '', 0, 0, 0, 0, 0]      count = 0      for line in fileRead:          if line[0].isdigit():              total = float(line[2]) + float(line[3]) + float(line[4])              line[5], line[6] = total, total/3              for i in range(2, 7):                  lastLine[i] += float(line[i])              count += 1          elif line[0] == 'Averange Score':              lastLine[0] = line[0]              for i in range(2, 7):                  lastLine[i] /= count              line = lastLine          newFileRead.writerow(line)  # Open the file  path = os.path.dirname(\_\_file\_\_)  # Confirm the path to avoid relative path errors  file = open(os.path.join(path, 'score-database.csv'), 'r')  newFile = open(os.path.join(path, 'score-database-new.csv'), 'w', newline='')  fileRead = csv.reader(file)  newFileRead = csv.writer(newFile)  # Complete and write to the file  fileWrite(fileRead, newFileRead)  # Close the file  file.close()  newFile.close()  Program 2:  import os  import csv  def dictProcess(fileRead):      '''Read the content and convert it into a dictionary'''      dict = {}      for line in fileRead:          if line[0].isdigit():              for i in range(2, 7):                  line[i] = float(line[i])              dict[line[1]] = (tuple(line))      return dict  def searchData(dict):      '''Import the dictionary contents and index them according to your needs to return the specified results'''      while True:          data = dict.get(input('Please input the name to search: '), 0)          if not data:              print("Sorry, we can't find data for that name.")              if input('Do you want to quit?(Y) ') == 'Y':                  print('Thanks for using.')                  break          else:              print('''\      Class: {0[0]:3}         | Name: {0[1]:11} |      Homework 1:  {0[2]:>5.1f} | Hmoework 2:    {0[3]:>5.1f} | Final Exam: {0[4]:>5.1f}      Total Score: {0[5]:>5.1f} | Average Score: {0[6]:>5.1f} |'''.format(data))  # Open the file  path = os.path.dirname(\_\_file\_\_)  # Confirm the path to avoid relative path errors  file = open(os.path.join(path, 'score-database-new.csv'), 'r')  fileRead = csv.reader(file)  # Establish a dictionary and provide an index  searchData(dictProcess(fileRead))  # Close the file  file.close() | | | | | | | |
| * Experimental summary/ Analysis   This time, under the guidance of my teacher, I learned the basic programming operation and definition of CSV file, and learned how to read and write files and print corresponding rows and columns. After the CSV file is converted into test text, separate the elements with "," and each line in the CSV file is a list. Therefore, we can use the corresponding code to quickly query the information in the file. In a word, I gained a lot from learning CSV files this time, and through repeated practice and continuous analysis of various problems to deepen my mastery of CSV file programming. As the saying goes, practice makes perfect, only by constantly | | | | | | | |
|  | Criteria | | | | | | scale |
| Goal | | | | | | A B C D E |
| Process | | | | | |
| Design | | | | | |
| Algorithm | | | | | |
| Code | | | | | |
| Data/Results | | | | | |
| summary | | | | | |
| written | | | | | |
| Score | | | : | | | |
| * Lab Evaluation Criteria   A: This lab is exceptional, working and meeting all of the specifications. The code is exceptionally well organized and very easy to follow. The code could be reused as a whole or each routine could be reused. The documentation is well written and clearly explains what the code is accomplishing and how. The program was delivered on time. The code is extremely efficient without sacrificing readability and understanding.  B: This lab is very good-- works and produces the correct results and displays them correctly. It also meets most of the other specifications. The code is fairly easy to read. Most of the code could be reused in other programs. The documentation consists of embedded comment and some simple header documentation that is somewhat useful in understanding the code. The program was delivered within a week of the due date. The code is fairly efficient without sacrificing readability and understanding.  C: This lab is adequate, with only minor deficiencies. The program produces correct results but does not display them correctly. The code is readable only by someone who knows what it is supposed to be doing. Some parts of the code could be reused in other programs. The documentation is simply comments embedded in the code with some simple header comments separating routines. The code was within 2 weeks of the due date. The code is brute force and unnecessarily long.  D: This lab shows some effort but has at least one major deficiency. The program is producing incorrect results. The code is poorly organized and very difficult to read. The code is not organized for reusability. The documentation is simply comments embedded in the code and does not help the reader understand the code. The code was more than 2 weeks overdue. The code is huge and appears to be patched together.  E: This lab is poorly written and shows very little effort or understanding. | | | | | | | |