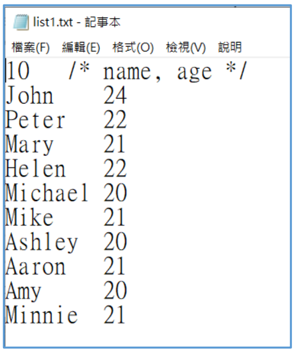
Data Structure Program Assignment #1
(Due: PM: 6:00, March 7, 2025)

Introduction

This homework assignment is designed to help you understand how to create a **struct data type** that can store multiple simple data types within a single record. A demonstration program is provided to guide you in designing your own programs. Your task is to modify the demonstration program to extend its functionality by handling **student grades**, calculating their **average**, sorting the average grades, and displaying all the recorded data.

Steps

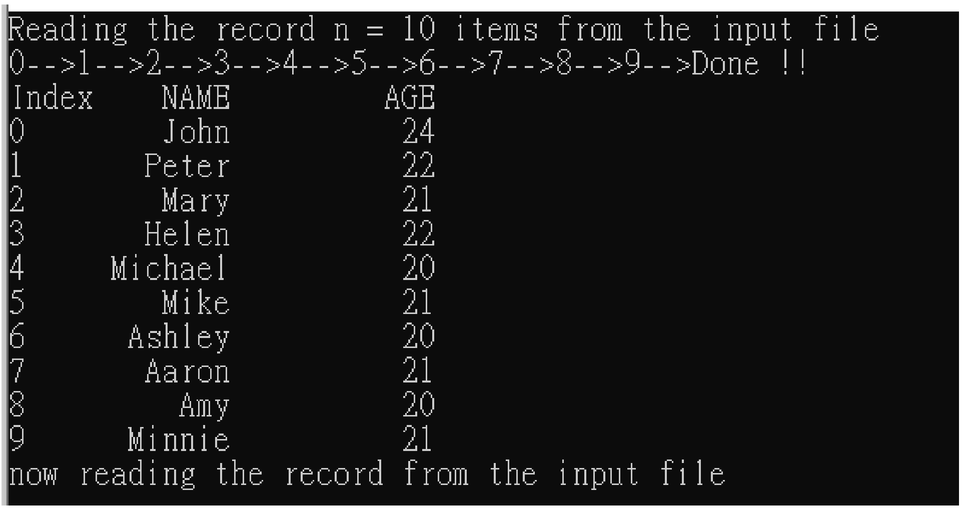
1. In class, we demonstrated a program that reads **student data** from a file named **"list1.txt"**, as shown on the right.



```
10    /* name, age */
John     24
Peter    22
Mary     21
Helen    22
Michael  20
Mike     21
Ashley   20
Aaron    21
Amy      20
Minnie   21
```

2. A demonstration program is provided to help you quickly become familiar with designing **C++ programs** using `typedef struct {}`. The program reads **student records** from the file **"list1.txt"**, stores them in memory, and then outputs the student records **one by one**, as shown below.

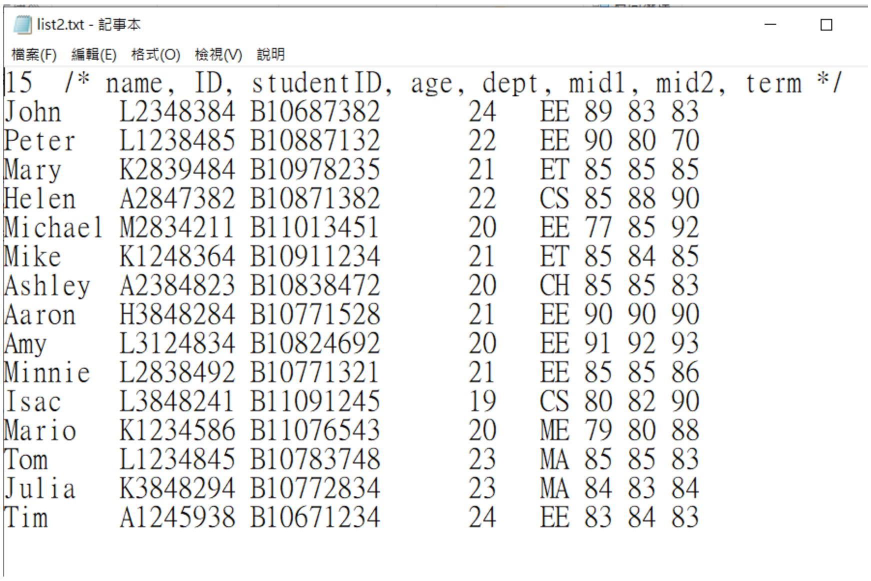
```
typedef struct {
    char name[80];
    int age;
} student;
```



```
Reading the record n = 10 items from the input file
0-->1-->2-->3-->4-->5-->6-->7-->8-->9-->Done !!
Index    NAME        AGE
0         John        24
1        Peter        22
2         Mary        21
3        Helen        22
4      Michael        20
5         Mike        21
6       Ashley        20
7        Aaron        21
8          Amy        20
9       Minnie        21
now reading the record from the input file
```

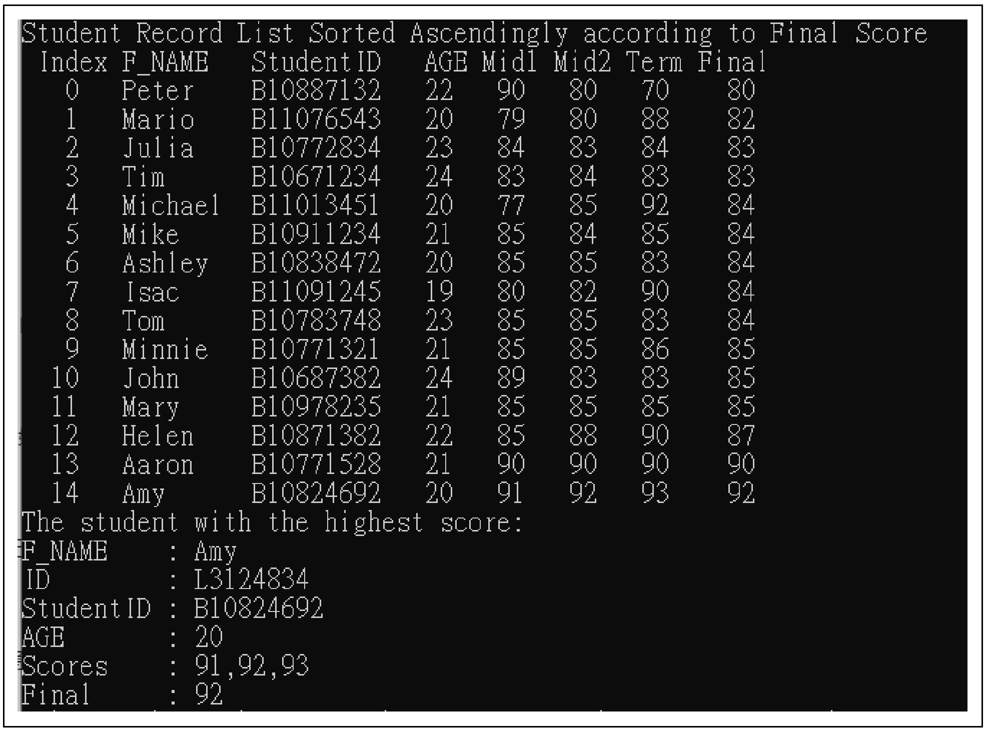
3. For this homework, another file, “list2.txt”, with more data is provided. It additionally comprises personal ID, student ID, department, and scores of

midterm1, midterm2, and term examinations, as compared to the list1.txt.



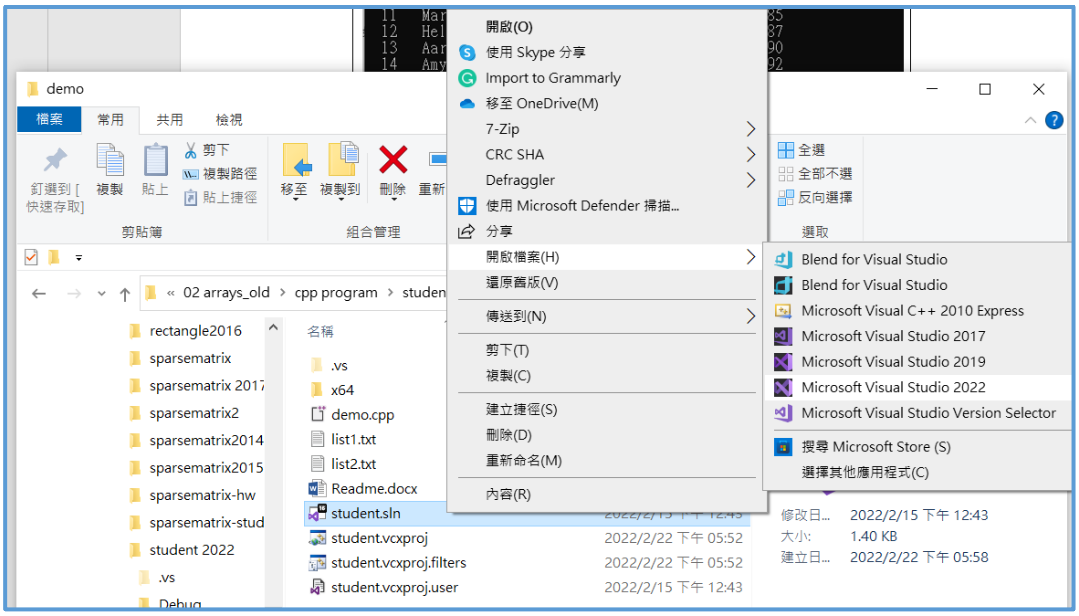
```
15  /* name, ID, studentID, age, dept, mid1, mid2, term */
John     L2348384 B10687382      24    EE 89 83 83
Peter    L1238485 B10887132      22    EE 90 80 70
Mary     K2839484 B10978235      21    ET 85 85 85
Helen    A2847382 B10871382      22    CS 85 88 90
Michael  M2834211 B11013451      20    EE 77 85 92
Mike     K1248364 B10911234      21    ET 85 84 85
Ashley   A2384823 B10838472      20    CH 85 85 83
Aaron    H3848284 B10771528      21    EE 90 90 90
Amy      L3124834 B10824692      20    EE 91 92 93
Minnie   L2838492 B10771321      21    EE 85 85 86
Isac     L3848241 B11091245      19    CS 80 82 90
Mario    K1234586 B11076543      20    ME 79 80 88
Tom      L1234845 B10783748      23    MA 85 85 83
Julia    K3848294 B10772834      23    MA 84 83 84
Tim      A1245938 B10671234      24    EE 83 84 83
```

4. Modify the demo program such that when the input file is changed to list2.txt, your new program can read and store them in computer memory.
5. Sort the average score ascendingly (see lecture1 notes basic-53) and print out the student list data. (shown below)
6. You had to output these data in the following format. (Note that the output **DATA FORMAT SHOULD STRICTLY BE IDENTICAL** to the one shown below.) You had to print out the full student record with the highest final score. For example, the figures



```
Student Record List Sorted Ascendingly according to Final Score
 Index F_NAME   StudentID  AGE Mid1 Mid2 Term Final
   0   Peter    B10887132  22   90   80   70    80
   1   Mario    B11076543  20   79   80   88    82
   2   Julia    B10772834  23   84   83   84    83
   3   Tim      B10671234  24   83   84   83    83
   4   Michael  B11013451  20   77   85   92    84
   5   Mike     B10911234  21   85   84   85    84
   6   Ashley   B10838472  20   85   85   83    84
   7   Isac     B11091245  19   80   82   90    84
   8   Tom      B10783748  23   85   85   83    84
   9   Minnie   B10771321  21   85   85   86    85
  10   John     B10687382  24   89   83   83    85
  11   Mary     B10978235  21   85   85   85    85
  12   Helen    B10871382  22   85   88   90    87
  13   Aaron    B10771528  21   90   90   90    90
  14   Amy      B10824692  20   91   92   93    92
The student with the highest score:
F_NAME    : Amy
ID        : L3124834
StudentID : B10824692
AGE       : 20
Scores    : 91,92,93
Final     : 92
```

7. For you to quickly start your programming, a reference project with a demo program is provided for you. After decompressing the zip file, click and open the “student.sln” with the visual studio 2022 to edit and design your program, as shown below:



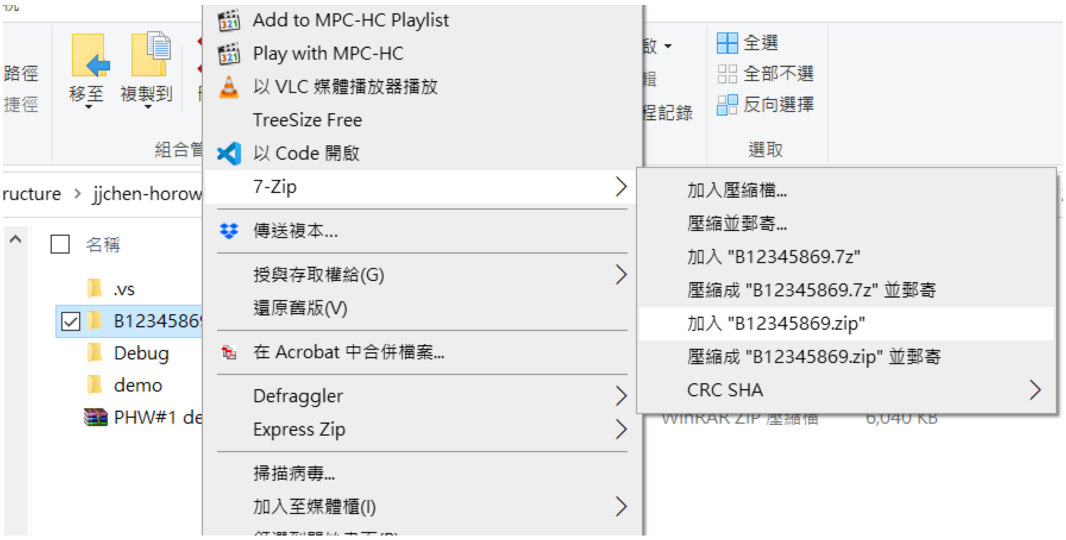
8. You are asked to:

[1] Provide detailed notes explaining each instruction in your program. Determine the minimum storage size required for a single student record.

[2] Summarize your work and experience in a Word document to demonstrate to the TA and teacher that you completed the assignment independently.

[3] Compress the whole project programs you have finished and use your student ID as the file name. For example, if your student ID is b12345869, your zip file should be b12345869.zip.

[4] Upload the compressed file to the Moodle website before the due date.


路徑
捷徑
移至
複製到
組合管
全選
全部不選
反向選擇
選取
程記錄
ructure > jjchen-horow
名稱
.vs
B1234586
Debug
demo
PHW#1 de
Add to MPC-HC Playlist
Play with MPC-HC
以 VLC 媒體播放器播放
TreeSize Free
以 Code 開啟
7-Zip
傳送複本...
授與存取權給(G)
還原舊版(V)
在 Acrobat 中合併檔案...
Defraggler
Express Zip
掃描病毒...
加入至媒體櫃(I)
加入壓縮檔...
壓縮並郵寄...
加入 "B12345869.7z"
壓縮成 "B12345869.7z" 並郵寄
加入 "B12345869.zip"
壓縮成 "B12345869.zip" 並郵寄
CRC SHA
WinRAR ZIP 壓縮檔
6,040 KB