Research Project 2: Student II

Kuan Lu

Date: 2015-3-31

Chapter 1: Introduction

Write a CLI program that reads scores and name of students, and prints out a summary sheet. The user can input as many students as possible. One students can have as many courses as possible. One course consists the name of the course and the marks the student got.

Input specifics

Each row consists of the student's name at the beginning, followed by the name of a course and the mark of the course. One can **add as many** course **in whatever order** he/she wants in the row, to indicate the end of inputting, add a semicolon **;** at the end of the row. After inputting every student's information, type "end" in the next line to inform the end of input.

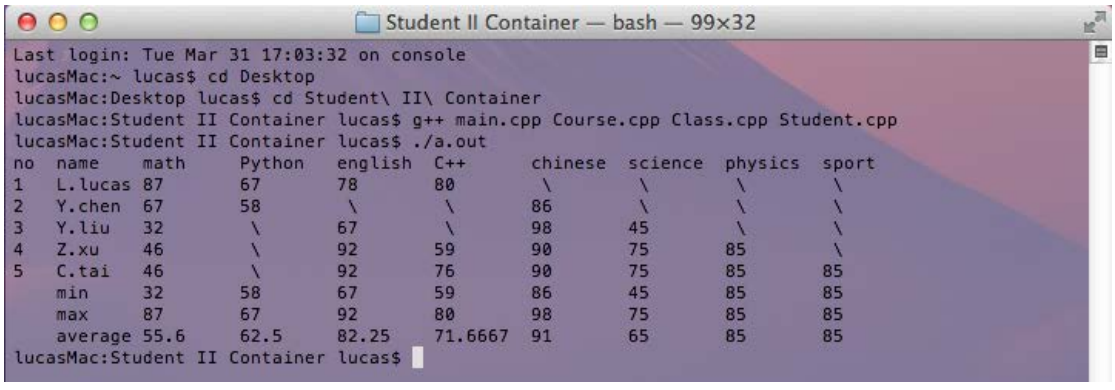
Input example:

```
L.lucas math 87 Python 67 english 78 C++ 80;
Y.chen chinese 86 math 67 Python 58;
Y.liu science 45 english 67 chinese 98 math 32;
Z.xu physics 85 chinese 90 english 92 science 75 math 46 C++ 59;
C.tai english 92 math 46 chinese 90 science 75 physics 85 sport 85 C++ 76;
end
```

Output specifics:

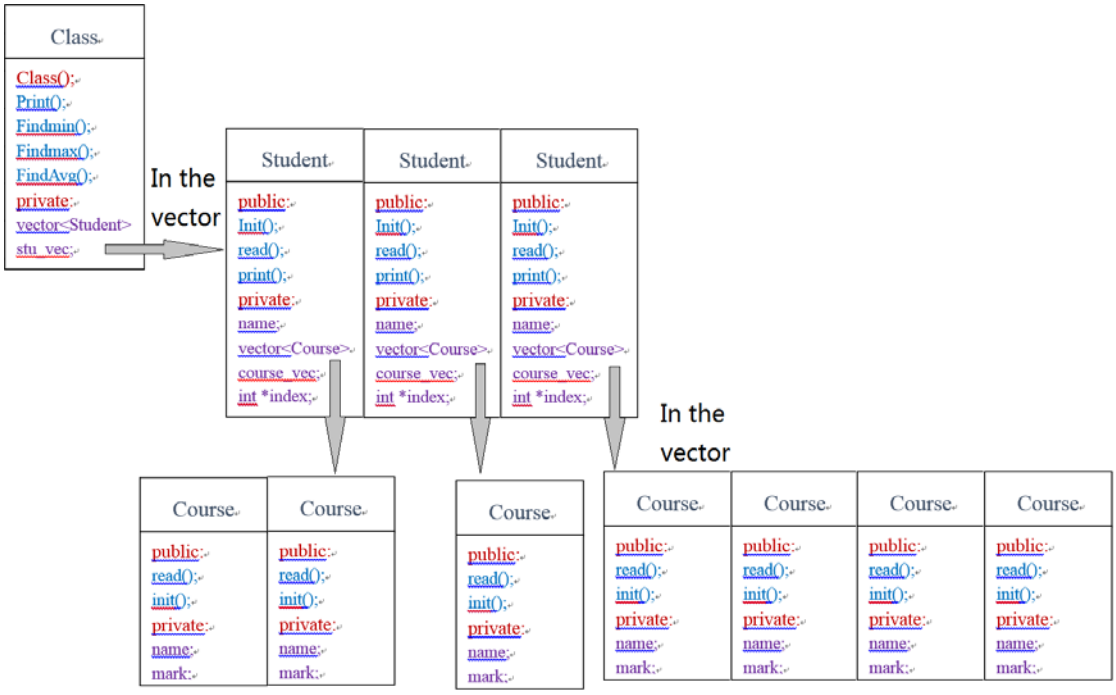
The first row lists all the courses, if a student doesn't have mark on this course the corresponding column will show a slash \ otherwise would be the mark.

Output example:



Chapter 2: Coding Specification

I. Class in this project:



II. Source Code

(1) Course.h

```
#ifndef __COURSE_H__
#define __COURSE_H__
#include<string>
using namespace std;

class Course{
public:
  double read();     /*read the mark of the course*/
  string read(string course_name); /*read the name of the course*/
  int init();       /*write the name and mark in the object*/
private:
  string  name;  /*name of the course*/
  double mark;   /*mark of the course*/
};

#endif
```

(2) Course.cpp

```
#include<iostream>
```

```
#include<vector>
#include<string>
#include"Course.h"

using namespace std;

double Course::read() {return mark;} //return the mark
string Course::read(string course_name){ return name;}
                            //overloading, return the course name

int Course::init(){  /*read from the input and write it to mark*/
    extern vector<string> course_name_vec;
    cin>>name;
    if(name==";" )   /*check if it gets to the end*/
        return 0;
    else
        cin>>mark;
    vector<string>::iterator  p;       //if  the  course  was  in  the
                                       //course_name_vec, do nothing
    for(p=course_name_vec.begin();p<course_name_vec.end();p++)
        if(name==*p)
        return 1;
    course_name_vec.push_back(name); //else put the course name in the
                                     //container
    return 1;
}
```

(3) Student.h

```
#ifndef __STUDENT_H__
#define __STUDENT_H__
#include<string>
#include<vector>
#include"Course.h"
using namespace std;

class Student{
public:
  int init();    /*Initialize the Student*/
  double read(int option); /*read the Student's mark of a course*/
  void print();  /*print the student's information*/
private:
  string name;
  vector<Course>  course_vec;  /*the vector that contain's every course*/
```

```
  int * index;      /*index stores the position of the student's course
                            in all the courses for uniform output*/
};

#endif
```

(4) Student.cpp

```
#include<iostream>
#include<iomanip>
#include<vector>
#include"Student.h"

using namespace std;

double Student::read(int option)
{
    if(index[option]!=-1) //if the student have the course
       return course_vec[index[option]].read(); //return the mark
    else
       return -1; //if a student don't have the course return -1
}

int Student::init() {
    cin>>name;    //write the name of the student
    Course* Ctmp;
    if(name=="end") //check if reaches the end of input
      return 0;
    else
     {
        Ctmp=new Course; //alloc space for a sourse
     while(Ctmp->init())  //initiate the course
        {
           course_vec.push_back(*Ctmp);  //push the course into the
                                         //vector
           Ctmp=new Course; //assign space for a new course
        }
     }
     delete Ctmp; //delete the temp pointer
      return 1;
}

void Student::print() {
     extern vector<string> course_name_vec;
```

```
    unsigned int size_total_course=course_name_vec.size(); //record
                                        //the total number of courses
    index=new int[size_total_course]; //allocate space for array index
    unsigned int i,j;

    for(i=0;i<size_total_course;i++) //initialize the array by setting
                                      //all the value -1
      index[i]=-1;

    for(i=0;i<course_vec.size();i++)
    {
       for(j=0;j<size_total_course;j++) //find the place of each course
                                        //of the stu in the total
        {                                          //course vector
           if(course_vec[i].read(" ")==course_name_vec[j])
           index[j]=i;
        }
    }                  /*finally we get the of the Xst(1st,2nd,3rd)
                         course of the student
                        is in the Xst column in the summary sheet*/
    cout<<left<<setw(8)<<name;
    for(i=0;i<size_total_course;i++)  //outputing the mark of each
                                      //course
    {
       if(index[i]!=-1)           //if a student have this course
       cout<<left<<setw(9)<<course_vec[index[i]].read();
       else                      //if a student don't have this course
                                  //output "\"
       cout<<left<<setw(9)<<" \\";
    }
    cout<<endl;
}
```

(5) Class.h

```
#ifndef __CLASS_H__
#define __CLASS_H__
#include"Student.h"

class Class{
public:
  Class();      /*Initialize the class*/
  void print();  /*Print the mark of all the student in the class
                   and the max, min and average of each course*/
```

```
  int Findmin(int option); /*Find the minimum mark of each subject*/
  int Findmax(int option); /*Find the maximum mark of each subject*/
  double FindAvg(int option);      /*calculate the average of each
subject*/
private:
  vector<Student>  stu_vec;  /*vector that contains the every student*/
};

#endif
```

(6) Class.cpp

```
#include<iostream>
#include<iomanip>
#include<cstdio>
#include"Class.h"
using namespace std;

Class::Class() //constructer that initialize the Class
{
    Student* Stmp;
    Stmp=new Student;
    while(Stmp->init()) //if doesn't reach the end of input
    {
      stu_vec.push_back(*Stmp);// put it in the stuent vector
      Stmp=new Student;
    }
    delete Stmp; //delete the temp pointer
}

void Class::print()
{    extern vector<string> course_name_vec;
     unsigned int i=1;
     cout<<"no  name   ";
     vector<string>::iterator p;
     for(p=course_name_vec.begin();p!=course_name_vec.end();p++)
     cout<<left<<setw(8)<<*p<<" ";   //output name of all the course in
the 1st row

     cout<<endl;
     vector<Student>::iterator q;
     for(q=stu_vec.begin();q!=stu_vec.end();q++,i++)
      {    cout<<left<<setw(4)<<i;    // the index of each student
           q->print();             //output each student's mark
```

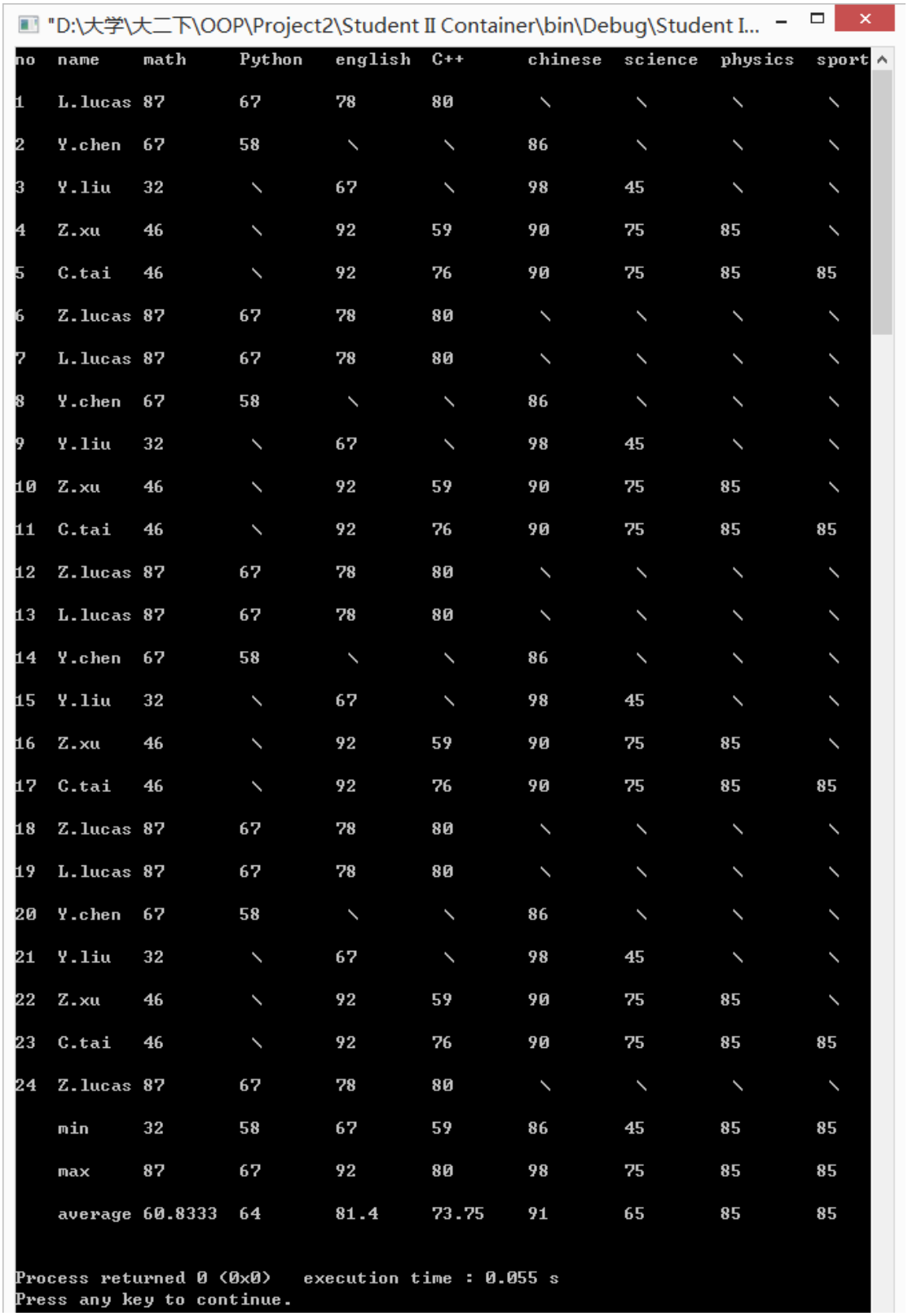
```
      }
      cout<<"   "<<left<<setw(8)<<"min";
      for(i=0;i<course_name_vec.size();i++)
        cout<<left<<setw(9)<<Findmin(i);   //output the min of each
                                                  //course
      cout<<endl;
      cout<<"   "<<left<<setw(8)<<"max";
      for(i=0;i<course_name_vec.size();i++)
        cout<<left<<setw(9)<<Findmax(i);   //output the max of each
                                                  //course
       cout<<endl;
       cout<<"   "<<left<<setw(8)<<"average";
      for(i=0;i<course_name_vec.size();i++)
        cout<<left<<setw(9)<<FindAvg(i);   //output the avg of each
                                                  //course
       cout<<endl;
}

int Class::Findmin(int option)
{
    double tmp;
    int mark;
    vector<Student>::iterator p;
    for(p=stu_vec.begin();p!=stu_vec.end();p++)//find the first student
       if(p->read(option)!=-1)  //that has mark on this course as tmp
       {
       tmp=p->read(option);
       break;
        }
    for(p=stu_vec.begin();p!=stu_vec.end();p++)
    {
    mark=p->read(option);    //searching through every stu to find the min
   if(mark==-1);          //omitting those who doesnt have this course
       else if (mark<tmp)
           tmp=mark;
    }
    return tmp;
}

int Class::Findmax(int option)
{
    double tmp=stu_vec[0].read(option);
    vector<Student>::iterator p;
    for(p=stu_vec.begin();p!=stu_vec.end();p++)
```

```
   if(p->read(option)>tmp)   //searching through every stu to find the max
          tmp=p->read(option);
    return tmp;
}

double Class::FindAvg(int option)
{
    double tmp=0;
    int counter=0;
    int mark;
    vector<Student>::iterator p;
    for(p=stu_vec.begin();p!=stu_vec.end();p++)
     {
          mark=p->read(option);
          if(mark==-1)  //omitting those who doesnt take the course
              tmp+=0;
          else
          {
     tmp+=mark; //if the student have taken the course add it to total
              counter++; //count how many stu have taken the course
          }
     }
    return tmp/counter;
}
```

(7) main.cpp

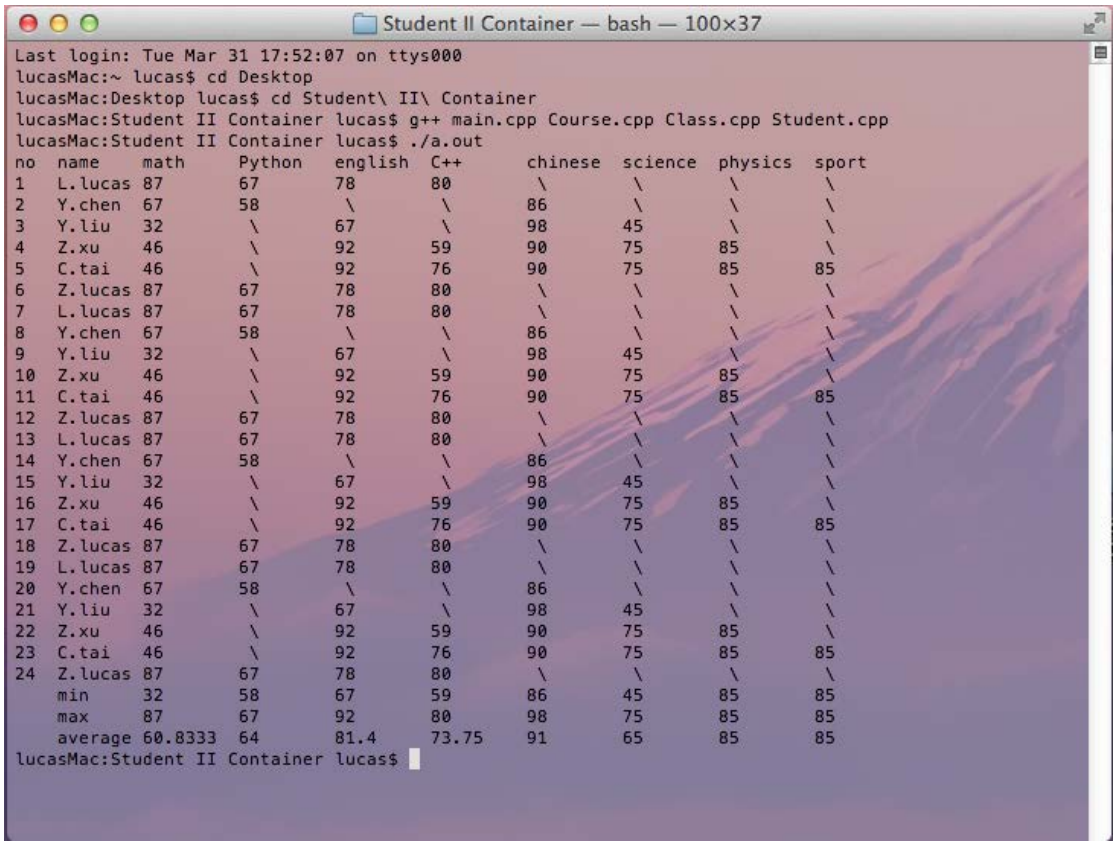
```
#include<iostream>
#include<vector>
#include<cstdio>
#include<cstdlib>
#include"Class.h"

using namespace std;
vector <string> course_name_vec; /*global variable, store the
                                name of different courses*/
int main()
{
  Class C;      /*declaration of Class C*/
  C.print();    /*print the summary sheet*/
  return 0;
}
```

Chapter 3: Testing Results



no	name	math	Python	english	C++	chinese	science	physics	sport
1	L.lucas	87	67	78	80	\	\	\	\
2	Y.chen	67	58	\	\	86	\	\	\
3	Y.liu	32	\	67	\	98	45	\	\
4	Z.xu	46	\	92	59	90	75	85	\
5	C.tai	46	\	92	76	90	75	85	85
6	Z.lucas	87	67	78	80	\	\	\	\
7	L.lucas	87	67	78	80	\	\	\	\
8	Y.chen	67	58	\	\	86	\	\	\
9	Y.liu	32	\	67	\	98	45	\	\
10	Z.xu	46	\	92	59	90	75	85	\
11	C.tai	46	\	92	76	90	75	85	85
12	Z.lucas	87	67	78	80	\	\	\	\
13	L.lucas	87	67	78	80	\	\	\	\
14	Y.chen	67	58	\	\	86	\	\	\
15	Y.liu	32	\	67	\	98	45	\	\
16	Z.xu	46	\	92	59	90	75	85	\
17	C.tai	46	\	92	76	90	75	85	85
18	Z.lucas	87	67	78	80	\	\	\	\
19	L.lucas	87	67	78	80	\	\	\	\
20	Y.chen	67	58	\	\	86	\	\	\
21	Y.liu	32	\	67	\	98	45	\	\
22	Z.xu	46	\	92	59	90	75	85	\
23	C.tai	46	\	92	76	90	75	85	85
24	Z.lucas	87	67	78	80	\	\	\	\
	min	32	58	67	59	86	45	85	85
	max	87	67	92	80	98	75	85	85
	average	60.8333	64	81.4	73.75	91	65	85	85

```
Process returned 0 (0x0)   execution time : 0.055 s
Press any key to continue.
```



Declaration

We hereby declare that all the work done in this project titled "World's Richest" is of my independent effort.