CSCI204 Assignment 1

Due: by March 19, 14:30. Resubmission is open: till March 26, 14:30

Marks: 8 marks

Objective

This assignment aims to establish a basic familiarity with C++ classes.

General Requirements

- Follow the common principles of OO programming when you design your classes
- Put your name, student number at the beginning of each source file

```
/*-----------------------------------------------------
Student's Name:
Student's email address:
Laboratory group (group code and time):
Purpose of this assignment:
-----------------------------------------------------*/
```

- Add comments to the source code to make your solution easier to follow
- Logical structures and statements are properly used for specific purposes

---

Design requirements:

Write a program to read a list of students from a text file the name of which should be provided from a command line. Create a dynamic array to store the students' records loaded from the file. The format of the data file is as follows:

```
15
Albert Einstein 52 67 63
Steve Abrew 90 86 90 93
David Nagasake 100 85 93 89
Mike Black 81 87 81 85
Andrew Van Den 90 82 95 87
Joanne Dong Nguyen 84 80 95 91
Chris Walljasper 86 100 96 89
Fred Albert 70 68
Dennis Dudley 74 79 77 81
Leo Rice 95
Fred Flinstone 73 81 78 74
Frances Dupre 82 76 79
Dave Light 89 76 91 83
Hua Tran Du 91 81 87 94
Sarah Trapp 83 98
```

The first line has a number equal to the total number of records. Each line shows one student's record with the name and marks for different subjects.

Define a class `Student` in a file **`student.h`** according to the description below:

- The class has a private data member `name` of type `char*`
- The class has a private data member `marks` that store marks of all subjects (up to 4 subjects)
- The class has a private data member `num` that holds the number of subjects that have been chosen
- Define appropriate constructors and a destructor
- Define necessary public member functions that can get values of data members and print out results

Implement member functions of Student in a file **`student.cpp`**

Write a program **`a1main.cpp`** to initialise the array of students by reading the students' records from a text file. After all records have been read, the program shall print the marks for each student along with the total mark and average mark. After all records have been printed, the program shall print the total marks and the average marks for each subject. The program should have appropriate **error detection** and **error recovery in case the data file is corrupted**.

You need to use **dynamic memory allocation** to store student's name for this task. Make sure there is no memory leak. Do not forget to include **namespace** in your program files.

You can use the command `bcheck` on banshee to check if there is any memory leak.

You need to follow good programming practices when you write the source code:
- meaningful identifiers for data members, member functions, classes
- no global variables
- appropriate indentation
- appropriate comments

Testing:

Upload all source code files and the data file to your working directory on banshee.

Compile the program in the working directory on banshee, by

```
CC –o assignment1 a1main.cpp student.cpp
```

Run the program using:

```
task1 students.dat
```

The input test file **`students.dat`** is provided and can be downloaded from the subject web site.

Note: Your program should work (and will be tested) with different data files which follow the same file format. **Do not define the file name as a constant inside your source code.**

The output of your program should be as follows:

Name	Mark1	Mark2	Mark3	Mark4	Total	Average
Albert Einstein	52	67	63		182	61
Steve Abrew	90	86	90	93	359	90
David Nagasake	100	85	93	89	367	92
Mike Black	81	87	81	85	334	84
Andrew Van Den	90	82	95	87	354	88
Joanne Dong Nguyen	84	80	95	91	350	88
Chris Walljasper	86	100	96	89	371	93
Fred Albert	70	68			138	69
Dennis Dudley	74	79	77	81	311	78
Leo Rice	95				95	95
Fred Flinstone	73	81	78	74	306	76
Frances Dupre	82	76	79		237	79
Dave Light	89	76	91	83	339	85
Hua Tran Du	91	81	87	94	353	88
Sarah Trapp	83	98			181	90
Total	1240	1146	1025	866		
Average	82	76	68	57		

Submission:

All assignments must be submitted electronically via the `submit` system. For this assignment you must submit all the files via the command (**in one line**):

```
submit -u your_user_name -c CSCI204 -a 1 a1main.cpp
student.h student.cpp
```

and enter your password.

Make sure that you use the **correct file names. The UNIX system is case sensitive.** You must submit all files **in one *submit* command line.**

After submit your assignment successfully, please check your email of confirmation. You should keep the email for the reference.

NOTES:

1. SUBMIT AS EARLY AS POSSIBLE. YOU CAN RESUBMIT LATER IF NECESSARY. Only the latest submission will be marked.

3**.** SUBMISSION VIA EMAIL IS NOT ACCEPTABLE. YOU HAVE TO USE SUBMIT COMMAND TO SUBMIT YOUR WORK.

4. ASSIGNMENT WITHOUT PROPERLY FILLED ASSIGNMENT HEADERS IN SUBMITTED FILES WILL NOT BE MARKED

4. The submitted file names must be the same as in the submission example. Files with other names will not be tested by the submit system and therefore will not be marked.

5. Enquiries about the marks can only be made within a maximum of 1 week after the assignment results are published. After 1 week the marks cannot be changed.

6. The assignment is an **individual assignment** and it is expected that all its tasks will be solved **individually without any cooperation** with the other students. If you have any doubts, questions, etc. please consult your lecturer. Plagiarism will result in a **FAIL** grade being recorded for that assessment task.