# COP 3330 - Programming Assignment #6

**Due: Mon, March 31**

## Objective

**To gain experience with base and derived classes, virtual functions, and using applications of polymorphism. Also, to gain further practice with file I/O.**

## Task

**You will design a set of classes for storing student information, along with a main program that will read student information from a file, store the data, compute final grades, and then print a summary report to an output file.**

---

### Details

1. **Design a set of classes that store student grade information. There should be one base class to store common data, and three derived classes that divide the set of students into three categories: English students, History students, and Math students. All data stored in these classes should be private or protected. Any access to class data from outside should be done through public member functions. The base class should allocate storage for the following data (and only this data):**
   - **student's first name (you may assume 20 characters or less)**
   - **student's last name (you may assume 20 characters or less)**
   - **Which course the student is in (English, History, or Math)**

2. **Each class should have a function that will compute and return the student's final average, based on the stored grades. All grades are based on a 100 point scale. Here are the grades that need storing for each subject, along with the breakdown for computing each final grade:**

   **English -- Attendance = 10%, Project = 30%, Midterm = 30%, Final Exam = 30%**

   **History -- Term Paper = 25%, Midterm = 35%, Final Exam = 40%**

   **Math -- There are 5 quizzes, to be averaged into one Quiz Average (which can be a decimal number). Final grade computed as follows:**
   *** Quiz Average = 15%, Test 1 = 25%, Test 2 = 25%, Final Exam = 35%**

3. **Write a main program (in a separate file) that does the following (in this order):**

   **a) Ask the user for input and output file names. This is the only input and output that should be done from keyboard and to the screen. All other input and output will be to and from files. (See the sample run below).**

   **b) Read the student data from the input file and store it using an array of appropriate type. You should use just one array for all students, not a separate array for each subject (i.e. this will be a heterogeneous**

**list). You will need to allocate this list dynamically, since the size is stored in the input file. (Note that this also means each list item will need to be created dynamically). Each student's data should be stored in a separate object. (Any dynamically allocated space should be cleaned up appropriately with `delete` when you are finished with it).**

**Hint: Remember that a heterogeneous list is implemented using an array of pointers to the base class type. And as stated above, this must be created dynamically in this situation. i.e. you will need to use the `new` operator. If you declare your array like this:**

```
  Student* list[size];
```

**then it is WRONG.**

**c) Print a summary report to the output file, as specified below. You'll need to use the function that computes the final average when you do this, since the final averages will be included in this summary report.**

4. **File formats**

   **Input File -- The first line of the input file will contain the number of students listed in the file. This will tell you how big a list you need. After the first lines, every set of two lines constitutes a student entry. The first line of a student entry is the name, in the format *lastName, firstName*. Note that a name could include spaces -- the comma will delineate last name from first name. The second line will contain the subject ("English", "History", or "Math"), followed by a list of grades (all integers), all separated by spaces. The order of the grades for each class type is as follows:**

   **English -- Attendance, Project, Midterm, Final Exam**
   **History -- Term Paper, Midterm, Final Exam**
   **Math -- Quiz 1, Quiz 2, Quiz 3, Quiz 4, Quiz 5, Test 1, Test 2, Final Exam**

   **Output File -- The output file that you print should list each student's name (*`firstName lastName`* - no extra punctuation between), Final Exam grade, final average (printed to 2 decimal places), and letter grade (based on 10 point scale, i.e. 90-100 A, 80-89 B, etc). Output should be separated by subject, with an appropriate heading before each section, and each student's information listed on a separate line, in an organized fashion. (See example below).**

5. **General Requirements**
   - **No global variables, other than constants!**
   - **All member data of your classes must be private or protected**
   - **Use the `const` qualifier on member functions wherever it is appropriate.**
   - **The code for this program should be portable. Test with g++ compiler commands before submitting**
   - **You may use any of the standard libraries that have been discussed in class (`iostream`, `iomanip`, `fstream`, as well as C libraries, like `cstring`, `cctype`, etc). You may also use the `string` class library if you wish.**
   - **You may not use any of the other STL (Standard Template Libraries) besides `string`**
   - **You should have already seen basic C++ file I/O techniques and I/O formatting in your pre-requisite course. If you need a refresher, see the following notes sets from COP 3014, Fall 2006:**
     - **File I/O and Stream Objects**
     - **Output Stream Formatting**

## Extra Credit

**Within each subject in the output file, list the students in alphabetic order, sorted by *last name*. Do not change the given case (upper/lower case) of the names that were read from the input file when you *print* the output file, and do not change the output file format. Just print the records in order by last name. This sort needs to be true alphabetical (not just the "lexicographical" sort).**

---

## Sample run

### Screen input and output: (keyboard input is underlined)

```
Please enter the name of the input file.
Filename: test.txt
Please enter the name of the output file.
Filename: outfile.txt
Processing Complete
```

---

### Sample input file: (Get a copy here)

```
6
Bunny, Bugs
Math 90 86 80 95 100 99 96 93
Schmuckatelli, Joe
History 88 75 90
Dipwart, Marvin
English 95 76 72 88
Crack Corn, Jimmy
Math 44 58 23 76 50 59 77 68
Kirk, James T.
English 40 100 68 88
Lewinsky, Monica
History 60 72 78
```

---

### Corresponding output file:

```
Student Grade Summary
---------------------

ENGLISH CLASS

Student                                   Final   Final   Letter
Name                                      Exam    Avg     Grade
----------------------------------------------------------------
Marvin Dipwart                            88      80.30   B
James T. Kirk                             88      80.80   B


HISTORY CLASS

Student                                   Final   Final   Letter
Name                                      Exam    Avg     Grade
----------------------------------------------------------------
Joe Schmuckatelli                         90      84.25   B
Monica Lewinsky                           78      71.40   C


MATH CLASS

Student                                   Final   Final   Letter
Name                                      Exam    Avg     Grade
----------------------------------------------------------------
Bugs Bunny                                93      94.83   A
Jimmy Crack Corn                          68      65.33   D
```

---

**Submit your files in the usual way, through the web submission page. Submit all source code files that you write**

**for this program. This should include a set of classes (base and derived), along with the main program described above.**