

CSCI 2010 Assignment 8

OBJECTIVES

This assignment has you work with aggregation and inheritance.

Q1: Aggregation [50 points]

Create a project in Visual C++ Express named `<my name>Pass8Q1` (lastname_firstname_Pass8Q1) and add three classes described below.

- The first class represents an author. In the header file `<my name>Author.h` create a class named `<my name>Author` with the following **private** member variables:
 - name: a string
 - number of books written: an integerThe class should define the following **public** member functions:
 - A constructor that takes one parameter: initial value for name.
 - A getter method for name.
 - Setter and getter methods for number of books written.Implement the constructor, the setter and getter methods in the `<my name>Author.cpp` file.
- The second class represents a publisher. In the header file `<my name>Publisher.h` create a class named `<my name>Publisher` with the following **private** member variables:
 - name: a string
 - address: a string
 - phone number: a stringThe class should also define the following **public** member functions:
 - A constructor that takes one parameter: initial value for name.
 - A getter method for name.
 - Setter and getter methods for address and phone number.Implement the constructor, setter and getter methods in the `<my name>Publisher.cpp` file.
- The third class represents a book. In the header file `<my name>Book.h` create a class named `<my name>Book`. The class should have the following **private** member variables:
 - title: a string
 - author: an Author object
 - publisher: a Publisher object
 - number of pages: an integer
 - edition: a string
 - copyright year: an integerThe class should also define the following **public** member functions:
 - A constructor that takes three parameters: title, author, and publisher.
 - Setter and getter methods for all member variables and member objects.Implement the constructor, setter and getter methods in the `<my name>Book.cpp` file.
- Write a main function in `<my name>Pass8Q1_Demo.cpp` after all the classes are defined.
 - Declare at least two Author objects.
 - Set appropriate values of other member variables in each Author object.
 - Declare at least two Publisher objects.
 - Set appropriate values of other member variables in each Publisher object.
 - Declare at least four Book objects.
 - Set appropriate values of other member variables in each Book object.
 - Check if two books are written by the same author.

- Check if two books are from the same publisher.
- You need to submit the following:
 1. You need to **zip the entire project folder** which should contains project files and the following source files:
 - <my name>Author.h
 - <my name>Author.cpp
 - <my name>Publisher.h
 - <my name>Publisher.cpp
 - <my name>Book.h
 - <my name>Book.cpp
 - <my name>Pass8Q1_Demo.cpp
 2. Name the ZIP file <my name>Pass8Q1.zip. Upload the ZIP file into the DropBox in D2L.

Important: If you do not put <my name> to the above mentioned fields (**class name** and **filename**), you will get 0 point for the question.

Estimated time: 3 hours

Q2: Inheritance [50 points]

Create a project in Visual C++ Express named <my name>Pass8Q2 (lastname_firstname_Pass8Q1) and add the files described below.

Define two classes which will represent different ways of display clocks. Specifications for classes should be in the **.h** files. Constructors and member function implementations should be in the **.cpp** file.

- A class named <my name>Clock
 - Create two member variables. One for hour, and one for minutes.
 - Legal values for hours are 0 to 23
 - Legal values for minutes are 0 to 59
 - Define a constructor with 2 arguments for the hour and minutes
 - Define a method named **printTime()**
 - Has the keyword **virtual** as part of its definition.
 - Print the time in the format **0:0**. For example, ten minutes after midnight would be 0:10.
 - The printed hour should be in the range to 0 to 23.
 - Define getter and setter methods for each data member.
- A class named <my name>AccurateClock represents time for a 12-hour clock with seconds.
 - This class is a subclass of <my name>Clock
 - It has an additional member variable seconds
 - Legal values for seconds is 0 to 59
 - Define getter and setter methods for seconds
 - Define a constructor with 3 arguments for the hour, minutes, and seconds
 - Define a method named **getAMPM()**
 - Returns “AM” if the hours is less that 12, otherwise returns “PM”
 - Override the **printTime()** method in the parent class
 - Print the time in the format **0:0:0 AM/PM**. For example, ten minutes thirty seconds after midnight would be **12:10:30 AM**
 - The AM and PM should be part of the string.
 - The printed hour should be in the range to 1 – 12.

- Add the file `<my name>Pass8Q2_Demo.cpp` to your project. It should contain the following functions
 - In `main()`
 - Create a **Clock** object that represents midnight.
 - Print the time.
 - Change it to represent 20:56.
 - Print it again
 - Create an **AccurateClock** that represents 7:45:30 PM
 - Print the time.
 - Change it to represent 8:40:25 AM
 - Print it again
 - Create an **AccurateClock** that represents 12:59:58 AM
 - Print the time.
 - Change it to represent 12:20:58 PM
 - Print it again
- Wherever you are asked to print the time in `main()`, it should print it with the appropriate format. For example, if you have a `Clock` that represents 11:20, it should print
The current time is 11:20
- On the other hand, if you have an `AccurateClock` object that represents 11:20:50 PM, it should print
The current time is 11:20:50 PM
- You need to submit the following:
 1. You need to **zip the entire project folder** which should contains project files and the following source files:
 - `<my name>Clock.h`
 - `<my name>Clock.cpp`
 - `<my name>AccurateClock.h`
 - `<my name>AccurateClock.cpp`
 - `<my name>Pass8Q2_Demo.cpp`
 2. Name the ZIP folder `<my name>Pass8Q2.zip`. Upload the ZIP file into the DropBox in D2L.

Important: If you do not put `<my name>` to the above mentioned fields (**class name** and **filename**), you will get 0 point for the question.

Estimated time: 3 hours

Submission instructions:

You need to compile the above programs (questions) separately, and provide **two test cases** (if applicable) for each program (question). Do a screen capture of the input and related output for each test case. Use any graphic editing software (e.g. Microsoft Paint, Adobe Fireworks) to cut out the program input and output (from the screen capture), paste them into a word document under a related question number, save the document as a pdf file. A sample input/output (screen capture) can be found at the end of this document.

You need to submit the following:

1. A pdf file containing the screen captures of program input and output of all test cases, name the file **lastname_firstname_assignment08.pdf**.
2. Two zip files
 - **lastname_firstname_Pass8Q1.zip** (`<my name>Pass8Q1.zip`)
 - **lastname_firstname_Pass8Q2.zip** (`<my name>Pass8Q2.zip`)

Please submit electronic copy (the above mentioned files) to D2L digital dropbox.
If you cannot follow the above instructions, points would be deducted.

Grading guidelines (programming questions):

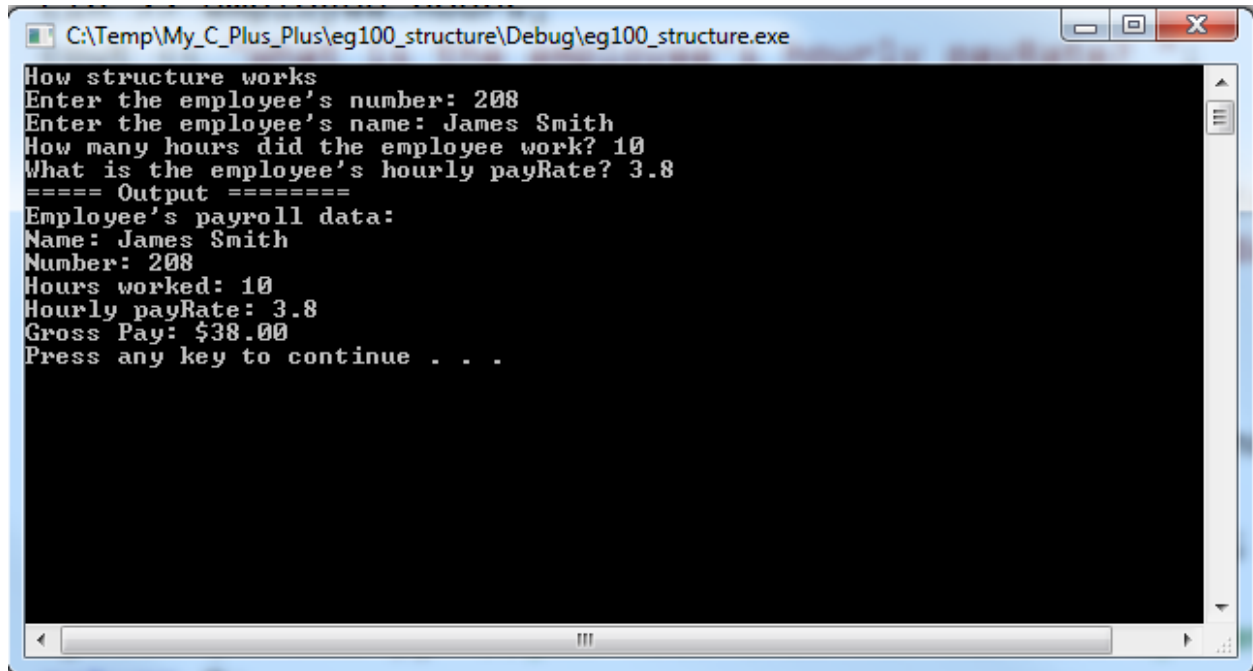
Your programs will be judged on several criteria, which are shown below.

- Correctness (50%): Does the program compile correctly? Does the program do what it's supposed to do?
- Design (20%): Are operations broken down in a reasonable way (e.g. classes and methods)?
- Style (10%): Is the program **indented** properly? Do variables have **meaningful names**?
- Robustness (10%): Does the program handle erroneous or unexpected input gracefully?
- Documentation (10%): Do all program files begin with a **comment** that identifies the author, the course code, and the program date? Are all the classes, methods and data fields clearly **documented (comments)**? Are unclear parts of code **documented (comments)**? (Some items mentioned may not apply to some languages)

A program that does not compile will get at most **50% of the possible points**.

Sample input/output (screen captures)

Assignment X, test case 1, input/output:



```
C:\Temp\My_C_Plus_Plus\eg100_structure\Debug\eg100_structure.exe
How structure works
Enter the employee's number: 208
Enter the employee's name: James Smith
How many hours did the employee work? 10
What is the employee's hourly payRate? 3.8
===== Output =====
Employee's payroll data:
Name: James Smith
Number: 208
Hours worked: 10
Hourly payRate: 3.8
Gross Pay: $38.00
Press any key to continue . . .
```

Screen capture must be readable by the instructor, or 0 point will be given.

Please note that you can use more than one screen captures for each test case.