**Submission of Your Work**

You need to prepare and submit ONE SINGLE MS Word document to Canvas (in your lab section) as LastName\_FirstName\_proj2.docx. It must contain:

* Your NAME only on page 1
* Source code. Copy/Paste your final source code. You must include standard “comment header” even if code is provided. *Do Not* paste a snippet of your source code, it must be copy/pasted.
* Initial test plan. After reading the project requirements, but **before** beginning to code, create the test case table, below, completed through column Expected Output. Include in your report *AFTER* source code.
* Final test plan. Write your program then complete the **test table** with actual output results and include in your report *AFTER* your source code.
* UML relationship diagram. Create a UML diagram showing relationship between all classes. Include in your report after final test plan.
* Output results. Paste in a snippet of output showing results for **every listed test case in your final test plan**, labeled with test case #

Test Table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test # | Valid / Invalid Data | Description of test | Input Value | Expected Output | Actual Output | Test Pass / Fail |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |

* Add / delete rows from Test Table as necessary
* Modify column widths as necessary
* Test both valid and invalid input
* Test for every output expected
* If failure is an expected output and it happens then that test Passes
* Any test that fails means the program must be fixed so that it passes the test
  + Failing tests need a new test row, ie 1a, 1b, etc, showing corrections from original

The goal of the program is to calculate and display employee payments for a company. To accomplish this task, you will need to create and derive several classes.

1. Design a base class named Employee. The class should keep the following information in its data members:

* Employee name (separate first and last names)
* Social Security Number, in the format xxx-xx-xxxx, where x is a digit within the range 0 through 9
* Employee number, in the format xxx-L, where x is a digit within the range 0 through 9, and the L is a letter within the range A through M.

Add a constructor and other appropriate member functions to manipulate the class data members (get, set, and print).

Input Validation: Only accept valid Social Security Numbers (with no alphabetic characters) and valid Employee Numbers (as described above).

Display of Social Security Number should be masked to only show the last four numbers, (e.g. xxx-xx-1234)

Note: Input validation will be done in the setter functions and no input statements should be used – i.e. return a methodStatus value indicating success or failure.

2. Design a class named SalaryEmployeePay. This class should be derived from the Employee class. It should store the following information in data members:

* Annual pay
* Weekly pay (to be calculated from annual pay)
* Tax rate code (1 = 25%, 2 = 20%, 3 = 15%)

Add a constructor and other appropriate member functions to manipulate the class data members.

Input Validation: Do not accept negative values for annual pay. Do not accept values outside the range of 1 to 3 for tax rate code.

3. HourlyEmployee. This class should be derived from the Employee class. It should store the following information in data members:

* Hourly pay rate
* Number of hours worked

Add a constructor and other appropriate member functions to manipulate the class data members.

Input Validation: Hourly pay rate cannot be less than $10.00 per hour or more than $75.00 per hour. Do not accept values over 60 for hours worked.

4. HourlyEmployeePay. This class should be derived from the HourlyEmployee class. It should store the following information in data members:

* Overtime pay rate (1.5 times hourly rate if over 40 hours, calculate from HourlyEmployee hourly pay rate)
* Tax rate code (1 = 25%, 2 = 20%, 3 = 15%)
* Work status (F = Full time, P = Part Time)

Add a constructor and other appropriate member functions to manipulate the class data members.

Input Validation: Do not accept values outside the range of 1 to 3 for tax rate code, F or P for work status.

5. AgencyEmployeePay. This class should be derived from the HourlyEmployee class. It should store the following information in data members:

* Company to pay

Add a constructor and other appropriate member functions to manipulate the class data members.

Using the classes you created, write a testbed main program that asks for sample data for the various employee types. Then, for each type calculate the employee’s pay and display her/his pay information on the screen and to an output file ‘pay.dat”.