Assignment-3

Problem-1 (4 points):

State whether each of the following is *true* or *false*. If *false*, explain why.

a) Every class definition contains the keyword *private*.
b) Variables declared in the body of a particular member function are known as data members and can be used in all member functions of the class.
c) Empty parentheses following a function name in a function prototype indicate that the function can accept any number of parameters to perform its task.
d) Data members or member functions declared with access specifier *private* are accessible to member functions of the class in which they're declared.

Problem-2 (4 points):

Write a detailed answer for each of the following questions:

a) What's a header? What's a source-code file? Discuss the purpose of each.
b) Explain why a class might provide a *set* function and a *get* function for a data member.

Problem-3 (7 points):

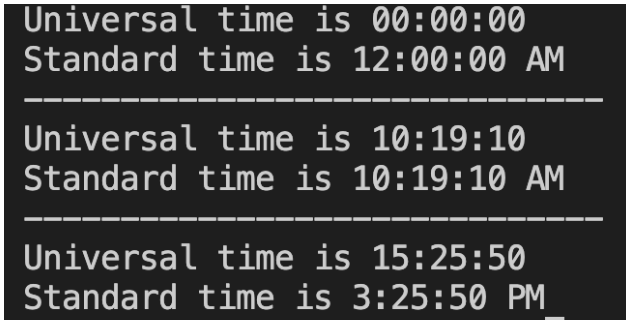
Design a class named Time. The class contains:

- Three private unsigned int data members hour, minute, and second that represent a time.
- A no-arg constructor that creates a Time object and initializes the data members to 0.
- A constructor that constructs a Time object with a specified elapse time in seconds since the middle of night. For example, if the elapse time is 555550 seconds (or 154 hours, 19 minutes, and 10 seconds), the hour is 10, the minute is 19, and the second is 10.
- A constant function printUniversal() that prints time in universal-time format.
- A constant function printStandard() that prints time in standard-time format.

Draw the UML diagram for the class. Write a test program that creates three Time objects, one using a no-arg constructor, the second one using Time (555550), the third one using

Time(55550) and display their hour, minute, and second in standard and universal time formats.

Here is a sample run:



```
Universal time is 00:00:00
Standard time is 12:00:00 AM
------------------------------
Universal time is 10:19:10
Standard time is 10:19:10 AM
------------------------------
Universal time is 15:25:50
Standard time is 3:25:50 PM
```

Hint: use function *setfill* to set the appropriate number of zeros in the output.

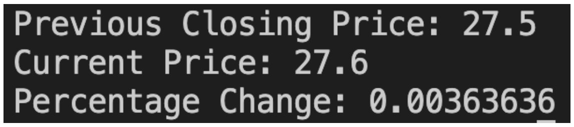
Problem-4 (5 points):

Design a class named Stock that contains the following:

- A string data field named symbol for the stock's symbol.
- A string data field named name for the stock's name.
- A double data field named previousClosingPrice that stores the stock price for the previous day.
- A double data field named currentPrice that stores the stock price for the current time.
- A constructor that creates a stock with specified symbol and name.
- The constant accessor functions for all data fields.
- The mutator functions for previousClosingPrice and currentPrice.
- A constant function named getChangePercent() that returns the percentage changed from previousClosingPrice to currentPrice.

Write a test program that creates a Stock object with the stock symbol MSFT, the name Microsoft Corporation, and the previous closing price of 27.5. Set a new current price to 27.6 and display the price-change percentage.

Here is a sample run:

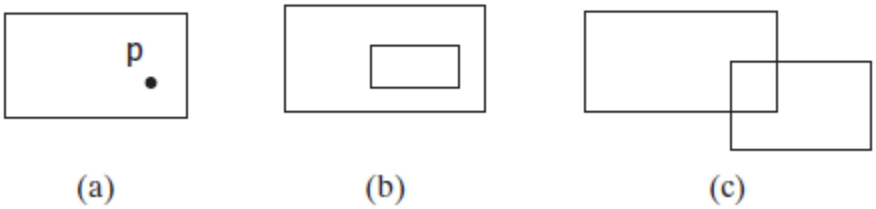


```
Previous Closing Price: 27.5
Current Price: 27.6
Percentage Change: 0.00363636
```

Problem-5 (10 points):

Define the Rectangle2D class that contains:

- Two double data fields named x and y that specify the center of the rectangle with constant get functions and set functions. (Assume that the rectangle sides are parallel to x- or y-axes.)
- The double data fields width and height with constant get functions and set functions.
- A no-arg constructor that creates a default rectangle with (0, 0) for (x, y) and 1 for both width and height.
- A constructor that creates a rectangle with the specified x, y, width, and height.
- A constant function contains(double x, double y) that returns true if the specified point (x, y) is inside this rectangle. See Figure a.
- A constant function contains(const Rectangle2D &r) that returns true if the specified rectangle is inside this rectangle. See Figure b.
- A constant function overlaps(const Rectangle2D &r) that returns true if the specified rectangle overlaps with this rectangle. See Figure c.



- Draw the UML diagram for the class. Implement the class. **Separate a class definition from a class implementation.** Write a test program that creates three Rectangle2D objects r1(2, 2, 5.5, 4.9), r2(4, 5, 10.5, 3.2), and r3(3, 5, 2.3, 5.4), and displays the result of r1.contains(3, 3), r1.contains(r2), and r1.overlaps(r3).

Rubric for Implementation Problems

0%	25%	50%	75%	100%
Source code files were not provided. Problem solution was	Significant assignment requirements were ignored or violated.	The output of the program was not shown. Lack of comments. Pour code readability	Choosing a poorly approach to solve a problem, for example, solving a	Program works correctly and meets the requirements of the assignment.

not submitted.	Program doesn’t compile.	(inconsistent indentation, variable naming, general organization)	problem with hard coding instead of using a loop. Minor details of the program specifications were violated.	Code is clean, well-organized, and well commented.

What to Hand In

Save your .cpp files for Problem 3 and 4 as FirstName_LastName_ProbX_Assignment3.cpp (for example, Carina_Winters_Prob3_Assignment3.cpp). Please submit (upload) your source codes (two .cpp files) to Canvas. For Problem 5 submit (upload) three separate files Rectangle2D.h, Rectangle2D.cpp, testRectangle2D.cpp. **Please provide snapshots of all your results after running your code.**