Homework 2: More on Strings, Classes and UML

Part 1: The Stock class (edu.albany.hw2.stock)

Design a class named Stock that contains the following:

1. A Symbol enum field named **symbol** for the stock's symbol
2. A double data field name previousClosingPrice that stores the stock price for the previous day
3. A double data field named currentPrice that stores the stock price for the current time
4. A constructor that creates a stock with specified **symbol** and name
5. The accessor functions for all data fields
6. The mutator functions for previousClosingPrice and currentPrice
7. A function named getChangePercent() that returns the percentage changed from previousClosingPrice to currentPrice
8. The Enum "Symbol" – will be used to denote stock for companies
    a. Declare the stock symbols for the following: Microsoft, Apple, Google, Amazon, AT&T
    b. The symbol should have a string value which is the company name
    c. Example Symbol.MSFT represents Microsoft stock.

Draw the UML diagram(s). Implement the classes. Write a test program that creates a Stock object with the stock symbol MSFT and the previous closing stock price of 58.9. Set a new current price to 59 and display the price change percentage. Repeat for one other company.

Part 2: Morse code generator (edu.albany.hw2.morse)

Morse code is a code where each letter of the English alphabet, each digit, and various punctuation characters are represented by a series of dots and dashes. Figure 1 shows part of the code. Write a program that asks the user to enter a string and then converts that string into Morse code. Use hyphens for dashes and periods for dots.

Some points to consider:

1. Design: Figure out what classes and methods you need to develop. Ex. Utility classes, class to represent Morse code generator, etc
2. Architecture: Create a UML of your architecture idea. Make it re-usable and configurable by someone other than yourself
3. The input can be given in two ways
    a. Command line args
    b. Passed into a function
4. Use string buffers when you are (or are going to) manipulating large Strings
5. Use appropriate accessors for your variables and methods
6. **EXPLAIN: What strategy you used to store the morse code encoding table and WHY? (-10 points if this is missing). Can add it as comment on the class.**

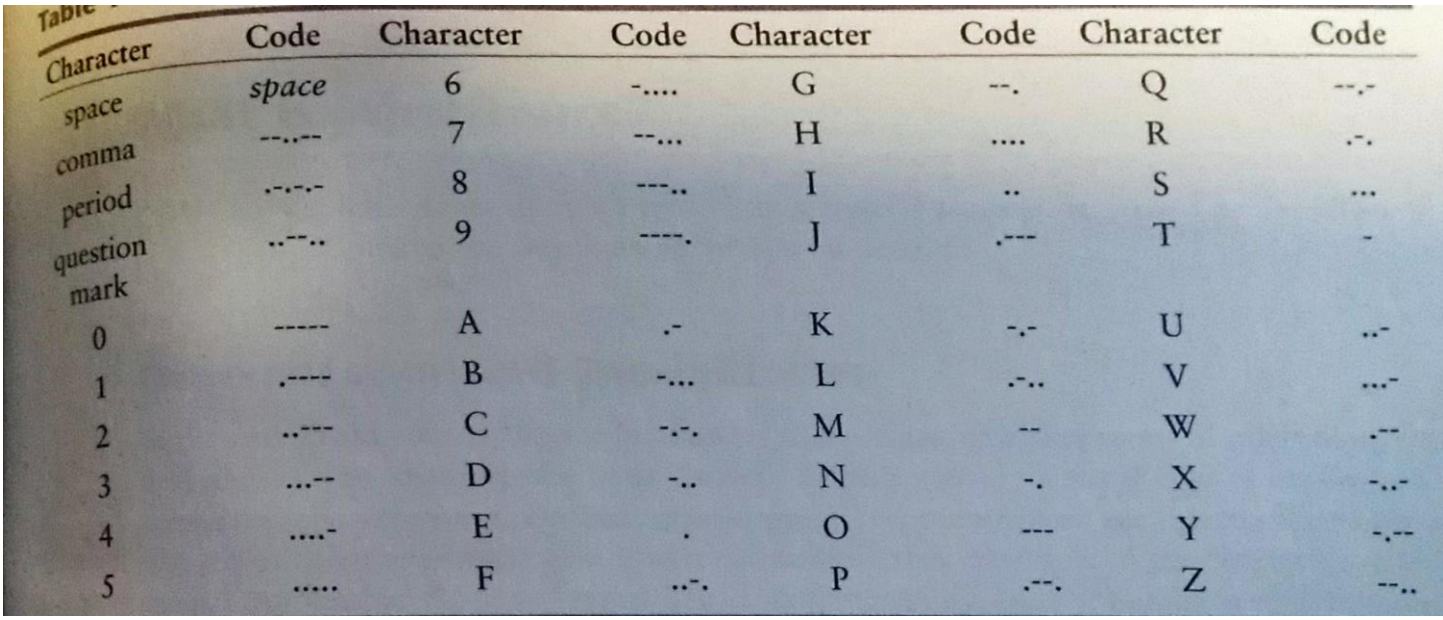
Part 3: Car Instrument Simulator (edu.albany.hw2.car)

For this part, you will design a set of classes that work together to simulate a car's fuel gauge and odometer. The classes you will design are the following:

- The FuelGauge class: This class will simulate a fuel gauge. Its responsibilities are as follows:
    - To know the car's current amount of fuel, in gallons
    - To report the car's current amount of fuel in gallons
    - To be able to increment he amount of fuel by 1 gallon. This simulates putting fuel in the car (The car can hold max 15 gallons)
    - To be able to decrement the amount of fuel by 1 gallon, if the amount of fuel is greater than 0 gallons. This simulates burning fuel as the car runs.
- The Odometer class: This class will simulate the car's odometer. Its responsibilities are as follows:
    - To know the car's current mileage
    - To report the car's current mileage
    - To be able to increment the current mileage by 1 mile. The maximum mileage the odometer can store is 999,999 miles. When this amount is exceeded, the odometer resets the current mileage to 0
    - To be able to work with a FuelGauge object. It should decrease the FuelGuage object's current amount of fuel by 1 gallon for every 22 miles travelled. (The car's fuel economy is 22 miles per gallon)

Demonstrate the classes by creating instances of each. Simulate filling the car up with fuel, and the run a loop that increments the odometer until the car runs out of fuel. During each loop iteration, print the car's current mileage and amount of fuel.

- Each part should have its code files in its respective package mentioned above
- The homework should be submitted as 1 eclipse project zipped
- UML diagrams can be submitted separately from the project zip file (but in the same submission on blackboard)
- Deadlines mentioned on blackboard are final. ONLY consider those deadlines
- You only need to use what has been taught to date in the class to solve this homework. Do not over-think it
- You will be graded based on the understanding of your concepts and the implementation (see Grading)



Character	Code	Character	Code	Character	Code	Character	Code
space	*space*	6	-....	G	--.	Q	--.-
comma	--..--	7	--...	H	....	R	.-.
period	.-.-.-	8	---..	I	..	S	...
question mark	..--..	9	----.	J	.---	T	-
0	-----	A	.-	K	-.-	U	..-
1	.----	B	-...	L	.-..	V	...-
2	..---	C	-.-.	M	--	W	.--
3	...--	D	-..	N	-.	X	-..-
4	....-	E	.	O	---	Y	-.--
5	.....	F	..-.	P	.--.	Z	--..

Figure 1: Morse code

Grading

Implementation – Considering the use of OO principles (taught till date)	40%
Execution	40%
UML diagrams	10%
Code clarity / organization / Error handling / encapsulation	10%