Assignment 1

Due by midnight Sunday, 28th of September, 2014

Read the instructions below carefully. The instructions must be followed.

This assignment is worth 5% of your grade. No late assignments will be accepted.

Put all of your code (for all the questions) in ONE file only, called `a1_xxxxxx.java` (where xxxxxx is replaced with your student number). More specifically, your program must have exactly one class and one method (the `main` method). All your code needs to be inside of that one `main` method. See the last page of the assignment for a possible template you can use for your program.

Within this file, `a1_xxxxxx.java`, separate your answers (i.e. code) to each question with a comment that looks like this:

```
 /*
==================================================
              Question X
==================================================
  */
```

(Replace X with the number of the question you are answering.)

Your program (i.e., your whole `a1_xxxxxx.java` **file) must compile, otherwise the assignment will be graded with mark 0.** For that reason, if you run out of time and/or one of your answers contains code that does not compile, then comment out that section of the code. Submit your assignment via Blackboard Learn (as instructed in the first lab.)

For this assignment, you are only allowed to use the concepts we have seen in the first 4-5 lectures of the semester. In particular,

- `if, for, while` etc. statements are also **not allowed**.
- Variables are only allowed to be declared as primitive data types (such as, `double`, `int`, `char` or `boolean`), except in Question 3, where you can declare some variables to be of type `String` (example, `String s`). In all other questions, strings are only allowed as part of calls to `System.out.println()` and `System.out.print()` methods.

Questions:

1. (1 point) Prompt the user to enter temperature in degrees Fahrenheit and have your program print the temperature in Kelvin.

   The output of your program could look something like this (approximately):

   ```
   Your program:  Enter the current temperature in Fahrenheit:
   User:  90.0
   Your program:  The current temperature is 90.0 degrees Fahrenheit, that is (approximately)
   305.3722222222222 Kelvin.
   ```

2. (1 point) Prompt the user to enter the weight in pounds and ounces and have your program print the weight in kilograms.

   Examples:

   ```
   Your program:  Enter two numbers for the weight (the first in pounds and the second in ounces).
   User:  2 0
   Your program:  2 pounds and 0 ounces is (approximately) 0.9072 kg.

   Your program:  Enter two numbers for the weight (the first in pounds and the second in ounces).
   User:  2 3.2
   Your program:  2 pounds and 3.2 ounces is (approximately) 0.9979 kg.
   ```

3. (1 points:) *Quote formatter:*

   Write a program that asks a user to enter a quote, then asks who the quote is made by, and in what year. It then prints the information as formated below.

   Example:

   ```
   your program:  Give me a quote.
   user:  Everything should be made as simple as possible but not simpler.
   your program:  Who said that?
   user:  Albert Einstein.
   your program:  What year did he say that?
   user:  1933
   your program:

   ===================================================================================
   In 1933, a person called Albert Einstein said: "Everything should
   be made as simple as possible but not simpler."
   ***********************************************************************************
   ```

4. (2 points) Prompt the user to enter a positive integer, n, and have your program print the sum of the first n positive integers.

   Examples:

   ```
   Your program:  Enter a positive integer.
   User:  5
   Your program:  15
   ```

```
Your program:  Enter a positive integer.
User:  101
Your program:  5151
```

5. (2 points) Have your program draw an ASCII art (by using `System.out.println()` and/or `System.out.print()` methods) where in 3 locations in the drawing the program will put the 3 characters that it received from the user.

   For example here is an output of a program that draws an ASCII art that is a cat and where the user entered characters ’A’, ’A’, ’t’ and they were placed in the drawing.

   `http://cwd.co.uk/asciimoo/public/images/1_postcard_small_1232402329.png`

   If the user entered characters ’B’, ’@, ’2’ instead, then ’B’, ’@’, ’2’ should be displayed in the locations of ’A’, ’A’, ’t’ in the above drawing.

   If you want to see more example of ASCII art, google ”small ascii art”. Your art does not have to be artsy or complicated. But, for example, do not make it simpler than two nested rectangles.

6. (2 points)

   Prompt the user to enter a upper case letter. You program should print the same letter but in lower case.

7. (2 points) Prompt the user to enter one non-negative number x and have your program print a pair of numbers (y,z) such that $x = y + z/12$ and y is an integer and z non-negative number smaller than 12 .

   Examples:

   ```
   user:  6.5
   your program:  (6, 6.0)

   user:  5.25
   your program:  (5, 3.0)
   ```

8. (2 points) *Median*

   The *median* of a group of numbers is the number from that group that has the property that at least half of the elements in the group are smaller or equal to it and at least half of the elements in the group are bigger or equal to it.

   In a sorted list of numbers, the median can be found in the middle of the list (well, the middle is only well defined if the list has odd number of elements). For example, the median of this group of numbers: $10, 11, 13, 15, 16, 23, 26$ is 15. Median of this group of numbers: $7, 1, 3$ is 3. The median of this group of numbers: $2, 2, 5$ is 2.

   Write a program that given 3 integers determines which of the given numbers is/are median. Note that the three numbers are not necessarily entered in sorted order.

   Hint: For each of the three given numbers, come up with a boolean expression that tests if that number is a median.

   Example:

   ```
   your program:  Enter three integers
   user:  1 18 8
   your program prints:
   1 is a median.  That is false.
   18 is a median.  That is false.
   ```

```
8 is a median.  That is true.
```

Example:

```
your program:  Enter three integers
user:  1 8 1
your program prints:
1 is a median.  That is true.
8 is a median.  That is false.
1 is a median.  That is true.
```

9. (3 points) *In or out of square?*

   Prompt the user to enter two numbers that represent the x and y coordinates of the bottom left corner of a square. Then promt the user to enter a positive number for the length of the side of the square. (Notice that this completely defines a square and its position in the plane). Finally prompt the user to enter two numbers that represent the x and y coordinates of some query point.

   Your program should test if the given query point is inside of the given square and print a message attesting to that. A point on the boundary of a square is considered to be inside the square.

   Recall, that you are not allowed to use `if` statement.

   Example:

```
your program:  Enter two numbers for two coordinates of the bottom left corner of the square.
user:  0 0
your program:  Enter a positive number for the length of the side of the square.
user:  2.5
your program:  Enter two numbers for the coordiantes of a query point.
user:  0.5.  1.5
your program:
The given query point (0.5, 1.5) is inside of the square.  That is true.
```

   2nd Example:

```
your program:  Enter two numbers for two coordinates of the bottom left corner of the square.
user:  2.5 1
your program:  Enter a positive number for the length of the side of the square.
user:  1
your program:  Enter two numbers for the coordiantes of a query point.
user:  -1.0 1.5
your program:
The given query point (-1.0, 1.5) is inside of the square.  That is false.
```

10. (4 points) *As few coins as possible, please!*
    Suppose that a cashier owes a customer some change and that the cashier only has quarters, dimes, nickels, and pennies. Write a program the computes the minimum number of coins that cashier can return. To solve this problem use the greedy algorithm explained below.

    PROBLEM STATEMENT: Your program should first ask the user for the amount of money he/she is owed (in dollars). You may assume that the user will enter a positive number. It should then print the minimum number of coins with which said amount can be made. Assume that the only coins available are quarters (25 cents), dimes (10 cents), nickels (5 cents), and pennies (1 cent).

EXAMPLES: If cashier owes 56 cents (i.e. $0.56) to the customer, the minimum number of coins the cashier can return is 4 (in particular, 2 quarters, 0 dimes, 1 nickel and 1 penny. It is not possible to return 3 or less coins). If cashier owes $1.42 to the customer, the minimum number of coins the customer can return is 9 (in particular 5 quarters, 1 dime, 1 nickel and 2 cents).

Thus your program runs will look like this:

1st example:

```
your program:  Enter the amount you are owed in $:
use:  0.56
your program:  4
```

2nd example:

```
your program:  Enter the amount you are owed in $:
use:  1.42
your program:  9
```

3rd example:

```
your program:  Enter the amount you are owed in $:
use:  1.00
your program:  4
```

STEPS YOUR PROGRAM SHOULD HAVE:

Your input is a real number (representing the number of dollars). In other words, if a customer is owed $8.25, assume that he/she will input 8.25 and not $8.25 or 825. Or, if a customer is owed $8 exactly, assume that he/she will input will be 8.00 or just 8 but, again, not $8 or 800.

Before doing anything else, you should convert the user's input entirely to cents (i.e., from a double to an int) to avoid errors in dealing with doubles. To do this, think of how many cents is there in one dollar.

HITS for your SOLUTION (ALGORITHM):

To find the minimum number of coins the, so called, greedy strategy (i.e. greedy algorithm) works for this problem. The greedy strategy tries the maximum possible number of the biggest-valued coins first, and then the 2nd biggest and so on. For example if customer is owed $1.42, thus 142 cents, the greedy strategy will first figure out how many quarters can it give to the customer most. In this case, it would be 5 quarters. It cannot be 6 as that equals $1.5 and the cashier would return too much money. Once the cashier returns 5 quarters, he/she still needs to return 17 cents. The next biggest coin after quarter is a dime. So the greedy strategy would try dimes next. Only one dime can fit in 17 cents, so the cashier should next return 1 dime. Then there is 7 cents left. The next biggest coin to consider is a nickel ... etc.

It can be proved that this greedy strategy gives an optimal solution (i.e. the smallest number of coins) for this version of the problem. Thus for this question, you are asked to implement this strategy to find the optimal solution.

Note: in the Canada (and most other) coin systems, a greedy algorithm of picking the largest denomination of coin which is not greater than the remaining amount to be made will always produce the optimal result (i.e. give the smallest number of coins). This is not automatically the case, though: if the coin denominations were 1, 3 and 4 cents then to make 6 cents, the greedy algorithm would choose three coins (4,1,1) whereas the optimal solution is two coins (3,3).

```
// Family name, Given name:
// Student number:
// Course: IT1 1120
// Assignment Number

import java.io.* ;
import java.util.Scanner;
// if you prefer to use the ITI1120 class to read input, skip to 2 lines above,
// but make sure you copy the file ITI1120.java in your current directory

class a1_xxxxxxx  //replace xxxxxxx by your student number
{
   public static void main (String args[ ])
   {

     /*
      ==================================================
                   Question 1
      ==================================================
      */

     // Put your code for question 1 here



     /*
      ==================================================
                    Question 2
      ==================================================
      */

     // Put your code for question 2 here

     //and so on

   }

}
```