CSCI 200 – Programming Project 2

Due 3/4/16 by 5:00pm

This assignment is designed to strengthen your understanding of some basic Java programming concepts from some of Chapter 2 and Chapter 3. This assignment is to be done **individually**. It is due by the beginning of class on that day – no exceptions. Make sure that all the work you submit to me is professional looking. Both hard and soft copies of this assignment will be turned into me by the due date and time or you will receive a zero! Soft copies will be submitted to blackboard by the due date above. Hard copies will be turned in at the beginning of class on the due date to the instructor.

What I expect for the softcopy of this assignment:
You will submit your code for Program Project on Blackboard under Programming Project 2 when complete (just attach the .java file and the screen shot of the output) and include the code discussion (described below) in the comment section of the submission page.

What I expect for the hardcopy of this assignment:
There is no hard copy requirement for this assignment.

Programming Project

(You will submit the code for this program and screenshot to blackboard). Use jGrasp and create a Java file named *YourName*_Java_Assignment2.java. Once you complete the project you will take a screenshot of the output and name that *YourName*_screenshot_Assignment2.jpg which should show the output console with the results of your running code for credit. Both files (screenshot and java) should be attached and submitted through blackboard (Do not zip these files together!).

Code Discussion: In the comments section on blackboard (where/when you submit the code), please include a brief paragraph discussing any issues that you encountered and measures that you took to correct them. Also, discuss anything that you learned from this assignment and confusion (if any) that you may still have on the material that we have addressed. This is also *required* for full credit on the programming portion of the assignment.

You will be required to have a comment block header and use comments throughout your code to explain your reasoning.

The comment header should include the following information: program name, course name and section number, author (this is your name), due date, and a brief description of the program.

Program Assignment Description

For this programming assignment, you will write one main method (one program) to solve three sub-problems. You should solve Part (a) then print blank lines and a “spacer” line of all dashes, then code for Part (b), then another “spacer” line of all dashes followed by Part (c) afterwards.

Part (a)
Ask the user to enter (in this order) a lucky number on one line, followed by his/her first name, followed by his/her last name on one line. Then, print the following to the screen, followed by a blank line.

Welcome <u>user's last name</u>, <u>user's first name</u>! Your lucky number was <u>number.</u>

Part (b)
Using a new class from the Java library: The GregorianCalendar Class. You can find all of the information that you need from this class in the API since this one is not in your textbook.

The GregorianCalendar class describes a point in time, as measured by the Gregorian calendar, the standard calendar that is commonly used throughout the world today.

You construct a GregorianCalendar object from a year, month, and day of the month, like this:
```
      GregorianCalendar example = new GregorianCalendar(); //Today's date
      GregorianCalendar EckertsBirthday = new GregorianCalendar(1919, Calendar.APRIL, 9);
```

Use the values Calendar.JANUARY . . . Calendar.DECEMBER to specify the month. The add method can be used to add a number of days to a GregorianCalendar object:
```
      example.add(Calendar.DAY_OF_MONTH, 10); // Now example is ten days from today
```

This is a mutator method—it changes the example object. The get method can be used to query a given GregorianCalendar object:
```
      int dayOfMonth = example.get(Calendar.DAY_OF_MONTH);
      int month = example.get(Calendar.MONTH);
       // 0 is January, 1 is February, ... and so on
      int year = example.get(Calendar.YEAR);
      int weekday = example.get(Calendar.DAY_OF_WEEK);
       // 1 is Sunday, 2 is Monday, . . . , 7 is Saturday
```

Create two GregorianCalendar objects. One should use the default constructor (the one with no parameters) and one should use the constructor that allows you to set the default date to be Alan Turing's birthday: June 23, 1912.

Create a String variable that will store "SunMonTueWedThuFriSat" and use your string methods to display the weekday with the appropriate string of three characters rather than with a number. Also, display the months correctly so that January is 1 and February is 2, and so on. Date should display in the format: month-day-year.
Your task is to write a program that prints the following information:

1. Today's Date (Hint: Print Month, day, and year separately as suggested above.)
2. Current Weekday - The day of the week it currently is (Hint: think the substring method…)
3. The date that is 100 days from today
4. Is the current date within a leap year?
5. The date of Alan Turing's birthday
6. The weekday of Alan Turing's birthday
7. The date that is 35,502 days from Alan Turing's birthday

Part (c)
Using a new class from the Java library. For this problem you will be using the Point Class from the java library. (Do not use Point2D) You can find all of the information that you need from this class in the API since this one is not in your textbook.

A Boy Scout is traveling to the Scout Headquarters but gets lost in the woods. He starts at location (-4, 5) and the HQ is at location (10, 10). Create two Point objects to store this info and an additional one to store the final resulting location. He travels as follows:

The scout follows his compass in direction 45 degrees for 15 miles, then turns and follows his compass in the direction 135 degrees for 10 miles. Turning again, following the compass in direction 270 for 12 miles. Finally, turning again and following the compass in direction 180 degrees for 13 miles.

Calculate how far the scout traveled from his original point and the point that he ends up at by the end of his travels. Display both to the screen as well as the distance they are from each other once his travels are completed and his distance to the headquarters. Assume one unit is one mile. All calculations must be accomplished in the code and blanks must use variable concatenation to display calculated distance and the Point toString method to display the points in the output (only print out the portion of the string that conatins the point values). Display results as follows (Include tabs and Do NOT use the Point2D class):

The Boy Scout stated at location _______ and,
after traveling a distance of _______ miles,
ended up at location ________.
The scout is a distance of __________ miles from the Headquarters.

Display both distance calculations to exactly 4 decimal places.

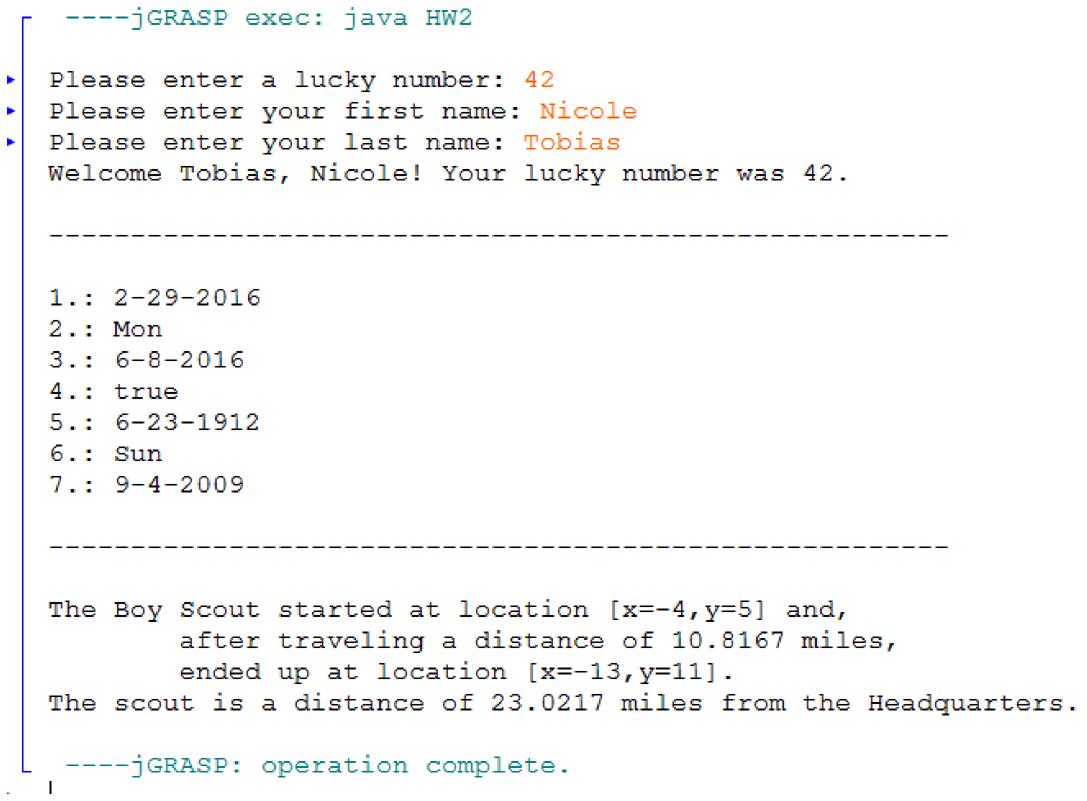
Formulas that you may find useful:
distance = $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

To travel from one point to another given direction and distance you can calculate the new location can be calculated as follows:
new_X = currect_X + dist * cos(direction_in_radians)
new_Y = current_Y + dist * sin(direction_in_radians)

So at the end of the day your output should look like the following: (things in red are user input)

```
Please enter a lucky number: user's lucky number
Please enter your first name: user's first name
Please enter your last name: user's last name
Welcome __lastname__, __firstname__! Your lucky number was __luckynum__.

---------------------------------------------------------------------------

1.: _________
2.: _________
3.: _________
4.: _________
5.: _________
6.: _________
7.: _________

---------------------------------------------------------------------------

The Boy Scout stated at location __startingPT__ and,
      after traveling a distance of __dist(startingPT,endingPT)__ miles,
      ended up at location __endingPT____.
The scout is a distance of __dist(endingPT,HQlocation)__ miles from the Headquarters.
```

Your output should look like mine:



```
 ----jGRASP exec: java HW2

Please enter a lucky number: 42
Please enter your first name: Nicole
Please enter your last name: Tobias
Welcome Tobias, Nicole! Your lucky number was 42.

-------------------------------------------------------

1.: 2-29-2016
2.: Mon
3.: 6-8-2016
4.: true
5.: 6-23-1912
6.: Sun
7.: 9-4-2009

-------------------------------------------------------

The Boy Scout started at location [x=-4,y=5] and,
        after traveling a distance of 10.8167 miles,
        ended up at location [x=-13,y=11].
The scout is a distance of 23.0217 miles from the Headquarters.

 ----jGRASP: operation complete.
```

Suggested layout:

```
/*Place header comment block here*/
public class YourName_Java_Assignment2
{
      public static void main(String[] args)
      {
            //Statements to solve Part (a)
            //Your code here

            System.out.println("\n------------------------\n");

            //Statements to solve Part (b)
            //Your code here

            System.out.println("\n------------------------\n");

            //Statements to solve Part (c)
            //Your code here

      }
}
```

Code will be graded on its readability, style, and clarity as well as its ability to display the proper results.

Do not turn in useless/wasteful code! Make sure that what you turn into me is final presentation quality. Good luck!! -Ms. Tobias