Lesson – 3

Problem : 1

Create your own Customer class. A Customer has a firstName, lastName, socSecurityNum (which you can represent as a String), also it has billingAddress and a shippingAddress(which you can represent as a type of Address. Initialize billingAddress and shippingAddress through its setter from Customer class.

Create a constructor for your Customer class to initialize firstName, lastName and socSecurityNum. Create getter, and setter methods for all five attributes.

Create an Address class with the attributes such as street, city, state and zip(which you can represent as a String). Create a constructor to initialize fields of Address class.

Your Customer class should have a toString() method that provides a string representation of the customer. A typical toString() output would be "[Joe, Smith, ssn: 332-221-4444]". Just copy this code in your Customer class.

```
public String toString() {

        return "[" + firstName + ", " + lastName + ", " + "ssn: " + socSecurityNum
+ "]";

    }
```

In the main method of a Main class, create three instances of Customer (be sure to create instances of Address to populate their billingAddress and shippingAddress fields using setters). Add these instances to an array. Then loop through the array and print to the console all those Customers whose billingAddress is located in the city of Chicago (when you create instances of Customer initially, be sure to create at least one Customer whose billing address is in Chicago).

Reference: see objectdemo package from the Democode from \\CS5 or Sakai

Problem – 2

(Target-Heart-Rate Calculator) While exercising, you can use a heart-rate monitor to see that your heart rate stays within a safe range suggested by your trainers and doctors. According to the American Heart Association (AHA), the formula for calculating your Maximum Heart Rate (MHR) in beats per minute is 220 minus your age in years. Your target heart rate is a range that's 50–85% of your maximum heart rate.

[Formula: To Calculate Target heart rate range is mentioned below]

a. Assign Resting Heart Rate (RHR) = 70.
b. Assign MHR = 220 – age.
c. Calculate Average Heart Rate (AHR) = MHR – RHR.
d. Assign Lower Boundary(LB) = 0.5
e. Assign Upper Boundary(UB) = 0.85.
f. Calculate Lower Boundary Target Heart Rate (LBTHR)= (AHR*LB) + RHR,
g. Calculate Upper Boundary Target Heart Rate (UBTHR) = (AHR*UB) + RHR
h. The Result of Target Heart Rate Range is between LBTHR and UBTHR

Create a class called HeartRates. The class attributes should include the person's first name, last name and date of birth. Your class should have a constructor that receives this data as parameters. For each attribute provide set and get methods. The class also should include

a. A method that calculates and returns the person's age (in years), (Refer AgeCalculator.java)
b. A method that calculates and returns the person's maximum heart rate and
c. A method that calculates and prints the person's target heart rate range.
d. Override the toString() method to display the person's first name, last name and date of birth, person's age in (years), maximum heart rate(MHR) and target-heart-rate range.

Write a Java app that prompts for the person's information, instantiates an object of class HeartRates and prints the required information.

Problem : 3

3. (MyDate Class) Create class MyDate with the following capabilities:
   [Use Java 8 LocalDate API to solve this problem]

   a) Output the date in multiple formats, such as
   MM/DD/YYYY
   June 14, 2016
   DDD YYYY

   b) Use overloaded constructors to create MyDate objects initialized with dates of the formats in part (a).
   1. In the first case the constructor should receive three integer values.
   2. In the second case it should receive a String and two integer values.
   3. In the third case it should receive two integer values, the first of which represents the day number in the year.

   c) Get the choices from the user as follows and supply input through console.

      Enter 1 for format: MM/DD/YYYY

      Enter 2 for format: Month DD, YYYY

      Enter 3 for format: DDD YYYY

      Enter 4 to exit

Sample output :

Enter 1 for format: MM/DD/YYYY
Enter 2 for format: Month DD, YYYY
Enter 3 for format: DDD YYYY
Enter 4 to exit
Choose your Choice :
1
Choice: 1
Enter Month:
12
Enter Day:
28
Enter Year:

2015

MM/dd/yyyy: 12/28/2015
MMMM dd,yyyy:December 28,2015
DDD yyyy:362 2015
Do you want to Continue :
y

Enter 1 for format: MM/DD/YYYY
Enter 2 for format: Month DD, YYYY
Enter 3 for format: DDD YYYY
Enter 4 to exit
Choose your Choice :
2
Choice: 2
Enter Month in text:
January
Enter Day:
26
Enter Year:
2004
MM/dd/yyyy: 01/26/2004
MMMM dd,yyyy:January 26,2004
DDD yyyy:026 2004
Do you want to Continue :
Y

Enter 1 for format: MM/DD/YYYY
Enter 2 for format: Month DD, YYYY
Enter 3 for format: DDD YYYY
Enter 4 to exit
Choose your Choice :
3
Choice: 3
Enter Day of Year:
221
Enter Year:
2016
MM/dd/yyyy: 08/08/2016
MMMM dd,yyyy:August 08,2016
DDD yyyy:221 2016
Do you want to Continue :
n
Terminated

Problem – 4

Reference : ImmutableDemo.java

Create Java classes for Triangle, Rectangle, and Circle. Provide each class with a method

public double computeArea()

Make all of these classes immutable. (Follow the guidelines in the slides for creating this type of class.) Provide one constructor for each class; the constructor should accept the data necessary to specify the figure, and to compute its area. The values accepted by the constructor should be stored in (private) instance fields of the class. For example, Rectangle should have instance fields width and length, and the constructor should look like this

public Rectangle(double width, double length)

For Triangle, you may use arguments base and height. And for Circle, use radius as the constructor argument.

Whenever you create instance fields for one of these classes, provide public accessors for them (but do not provide mutators since the class is supposed to be immutable – for instance, the dimensions of a Rectangle should be read-only). For example, you will have in the Rectangle class:

```
private double width;

public double getWidth() {
        return width;
}
```

Create a fourth class Main that will, in its main method, test these three figure classes as follows: It will create one instance of each (you can make your own choice for the dimensions of your figures and get the input from the console) and then print to the console the area of each. Typical output would be:

A Sample Output looks like this:

Enter C for Circle

Enter R for Rectangle

Enter T for Triangle

R

Enter the width of the Rectangle

120

Enter the height of the Rectangle

200

The area of Rectangle is : 24000.00

Here are some area formulas, in case you do not remember them:

Area of a rectangle = width * height

Area of a triangle = 1/2 * base * height

Area of a circle = PI * radius * radius