**Exercise 1**

Write a Java class representing a Circle (x, y, r). Inside the class, define a static attribute representing PI. Create two objects, c1 and c2, of class Circle.

Verify what happens when PI is accessed as a class attribute or, instead, as an instance attribute (using c1 or c2). What happens in c2 when c1 modifies the value of PI? What happens when PI is defined final?

**Exercise 2**

Write a Java class MyMath implementing a simple math library. In particular, MyMath must have 4 static methods implementing the 4 operations on doubles (add, subtract, multiply, divide). Implement another class, MyMathNS, with the same methods but NOT static. What changes?

**Exercise 3**

Use the Java wrapper types for converting a double variable into a string and back. Repeat the same exercise for an int variable. See Java API for classes *Double* and *Integer*.

**Exercise 4**

Create a class Employee including three private instance variables:

firstName (String)

lastName (String)

salary (double).

The class must have a constructor for initializing all three attributes. If salary is not positive, set it to 0. Furthermore, provide setters and getters for each attribute.

The class must have a second constructor (overloading) setting only first name and last name. In this case, the salary must be set to a default value (2500), taken from a *public static final* variable.

Write a test application showing the class Employee’s capabilities. Create two Employee objects and display each object’s salary. Then give each Employee a 10% raise and display each Employee’s salary again.

*Remember! Getters, Setters, Constructors and toString() methods can be automatically generated with Eclipse (see Source Menu).*

**Exercise 5**

Create a class *SavingsAccount*. Use a static variable *annualInterestRate* to store the annual interest rate for all account holders. The class contains only a protected attribute *balance* indicating the amount of the deposit.

Provide a method *calculateMonthlyInterest* to calculate the monthly interest by multiplying *balance* by *annualInterestRate* divided by 12.

Write a program to test the class *SavingsAccount*. Instantiate two *savingsAccount* objects, saver1 and saver2, with balances of $2000.00 and $3000.00, respectively. Set *annualInterestRate* to 4%, then calculate the monthly interest and update *balance* accordingly. Then set *annualInterestRate* to 5%, calculate the next month’s interest and update *balance* again.

**Exercise 6**

Implement a simple Bingo (Tombola) application using two classes: *Dealer* and *Player*.

***Class Dealer***

Constructors:

*Dealer():* create a new Dealer object

Methods:

*int exctractNumber():* returns the next number between [1..90]

***Class Player***

Constructors:

*Player(String name):* create a new player with a given name. Each player holds 5 numbers (randomly assigned by the constructor).

Methods:

*void checkNumber(int n):* verify if the player holds the number n and eventually marks it as extracted

*boolean bingo():* returns true if all 5 numbers have been extracted