# 44-542 Lab Activity

# Control Structures

1. All classes in this lab should be in a package named **motelrental**. For your project name, use your last name, followed by **Lab03ControlStructures**.
2. Create a class named **MotelRental**. This class provides methods for calculating rental prices for motel room. It is described below.
3. Attributes are given in this table. You must use the names given here, with the same capitalization as shown here. All attributes are private.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Identifier |
| standard motel rental amount | **double** | **standardMotelRentalAmount** |
| % seasonal rate | **double** | **seasonalRate** |
| % yearly discount | **double** | **yearlyDiscount** |

1. Constructor: There will be only one constructor. It has the following signature:

**MotelRental(double standardMotelRentalAmount, double seasonalRate, double yearlyDiscount).**

1. Methods: Methods are described below. All methods are public.
   1. Write getters for the private instance variables. Remember that getters are named starting with **get** and followed by the name of the instance variable, with the first character in upper case – **getStandardMotelRentalAmount**, for example.
   2. **getMonthlyRental(int month)**: Returns rental amount (as type **double**) for the specified month. The monthly rental amount is the same as the standard rental amount except for December and January. In December and January the standard rental amount is increased by the percentage stored in **seasonalRate**. Months are represented by the integer values 0-11, with 0 representing January, 1 representing February, and so forth.
   3. **getYearlyRental()**: Returns the total charged (as a **double**) for a year-long lease. For a year-long lease, the amount charged each month is the standard motel rental amount, decreased by the percent value stored in **yearlyDiscount**. Note that seasonal rates do not apply to a year-long lease – that is, there is no increase in the monthly rental amount for December and January. Also, note that the amount returned by this method is the *total* for the year.
   4. **getMonthlyRental(int month, int numberOfMonths)**: This method returns the total charged (as a **double**) for a lease of several months. The month the lease starts is represented by the first parameter, with a value of 0 representing January, 1 representing February, and so forth. The number of months that the motel room will be leased is given by the second parameter. The yearly discount applies to all full years, but not to partial years.

*Example: Suppose the standard motel rental is $2,000 per month, the seasonal percent rate is 0.20, and the yearly discount percent is 0.10. Assume the method is invoked with month = 9 and numberOfMonths = 43. Month 9 = October, and 43 months = 3 years and 7 months. For the first three years (36 months), the yearly discount applies, and no seasonal increases occur, for a total cost of $64,800 (36 \* $2000 = $72,000, minus a discount of $7,200). The seven months at the end of the lease begin with month 9 and continue through month 3 of the following year. Five of these months are at the standard rate of $2,000 for a total cost of $10,000. The additional two months (month 11 and 0) incur the season rate increase and are charged at the rate of $2,400 per month for a total of $4,800. Total cost for the 43 months is $64,800 + $10,000 + $4,800 = $79,600.*

* 1. **toString() method:** This method returns a string representation of the motel object.

*Example: For the motel described above, the* ***toString*** *method would return the following string:*

**Motel rental fee is $2000.0; the seasonal rate is 0.2; the yearly discount is 0.1**

1. Include Javadoc comments, using the @author, @param, and @return annotations when appropriate.
2. Generate documentation for your project by clicking on **Run** from the NetBeans menu bar and then selecting **Generate Javadoc**. The documentation will be placed in a **javadoc** subfolder of the **dist** subfolder inside your project folder. To view the documentation, open the **index.html** file that is created.
3. Create a class named **MotelRentalDriver:** This class has a method **main** and uses two **Scanner** objects to get input from files named **motelData.txt** and **monthData.txt**.

The file **motelData.txt** has information for each motel, with a separate line for each motel room. The information for a single motel room is on one line with the standard rental fee listed first, followed, on the same line, by the seasonal rate, followed, on the same line by the yearly discount. White space separates the values on each line.

The file **monthData.txt** is used to test the method **getMonthlyRental(int month, int numberOfMonths)**. Each line in this file consists of two **int** values, which will be passed as parameters to this method. White space separates the values on each line.

***Both files must be placed in the root folder of your Java project.***

Here is a pseudocode version of this class. You can use different variable names than those shown here. You do not have to follow these steps exactly, but you must produce the same output, *including labels*.

*Declare and initialize a Scanner object named inMotelData to read from the file motelData.txt*

*While motelData.txt has more data {*

*Read in the data from a single line and use it to create a new motel room named room1.*

*Print room1.*

*Print each instance variable using the getters.*

*Use a loop to test getMonthlyRental (int month) for each month of the year.*

*Test getYearlyRental by finding the yearly rental for room1.*

*Declare and initialize a Scanner object named inMonthData to read from the file monthData.txt.*

*While inMonthData has more data to read {*

*Read in the data from a single line, and use the values as arguments for a call to getMonthlyRental(int month, int numberOfMonths).*

*Print the result.*

*}*

*Close inMonthData.*

*}*

*Close inMotelData.*

A sample run is shown below, along with the input files used for testing. Your program must work for any valid data. We may use a different driver to test your methods, and we may use different data files for testing. If we use your driver, with the files given below, your output should match exactly that given here.

**motelData.txt (there is no line feed at the end of the second line)**

**2000 0.20 0.10**

**5000 0.15 0.05**

**monthData.txt (there is no line feed at the end of the second line)**

**7 3**

**9 43**

**3 9**

**8 5**

**10 48**

**Sample Run (using data files above)**

**NEW MOTEL ROOM CREATED:**

**Motel room rental fee is 2000.0; the seasonal rate is 0.2 the yearly discount is 0.1**

**TESTING GETTERS**

**Standard rental fee = 2000.0**

**Seasonal rate = 0.2**

**Yearly discount = 0.1**

**TESTING getMonthlyRental(int month)**

**Rental fee for month 0 is 2400.0**

**Rental fee for month 1 is 2000.0**

**Rental fee for month 2 is 2000.0**

**Rental fee for month 3 is 2000.0**

**Rental fee for month 4 is 2000.0**

**Rental fee for month 5 is 2000.0**

**Rental fee for month 6 is 2000.0**

**Rental fee for month 7 is 2000.0**

**Rental fee for month 8 is 2000.0**

**Rental fee for month 9 is 2000.0**

**Rental fee for month 10 is 2000.0**

**Rental fee for month 11 is 2400.0**

**TESTING getYearlyRental()**

**Yearly rental fee = 21600.0**

**Rental fee for 3 months, beginning with month 7 is 6000.0**

**Rental fee for 43 months, beginning with month 9 is 79600.0**

**Rental fee for 9 months, beginning with month 3 is 18400.0**

**Rental fee for 5 months, beginning with month 8 is 10800.0**

**Rental fee for 48 months, beginning with month 10 is 86400.0**

**NEW MOTEL ROOM CREATED:**

**Motel room rental fee is 5000.0; the seasonal rate is 0.15 the yearly discount is 0.05**

**TESTING GETTERS**

**Standard rental fee = 5000.0**

**Seasonal rate = 0.15**

**Yearly discount = 0.05**

**TESTING getMonthlyRental(int month)**

**Rental fee for month 0 is 5750.0**

**Rental fee for month 1 is 5000.0**

**Rental fee for month 2 is 5000.0**

**Rental fee for month 3 is 5000.0**

**Rental fee for month 4 is 5000.0**

**Rental fee for month 5 is 5000.0**

**Rental fee for month 6 is 5000.0**

**Rental fee for month 7 is 5000.0**

**Rental fee for month 8 is 5000.0**

**Rental fee for month 9 is 5000.0**

**Rental fee for month 10 is 5000.0**

**Rental fee for month 11 is 5750.0**

**TESTING getYearlyRental()**

**Yearly rental fee = 57000.0**

**Rental fee for 3 months, beginning with month 7 is 15000.0**

**Rental fee for 43 months, beginning with month 9 is 207500.0**

**Rental fee for 9 months, beginning with month 3 is 45750.0**

**Rental fee for 5 months, beginning with month 8 is 26500.0**

**Rental fee for 48 months, beginning with month 10 is 228000.0**

**Exercise 2 (Optional):** Want some more practice? Try this problem.

In this exercise, you will define two classes **Town** and **TownDriver**. These classes should be placed in a package named **towns**.

**The Town class**: This class predicts the population density of a given town for the next ten years, assuming that the population of the town will increase by 0.8% every year. The class is described in detail below.

1. **Attributes and Constants**:

A **Town** object has three attributes and two constants.

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Identifier |
| the name | **String** | **name** |
| the population | **long** | **population** |
| the area (in sq miles) | **double** | **area** |

The identifier of the first constant is **POPULATION** and is of type **long** with the value **10581**. The identifier of the second constant is **AREA** and is of type **double** with the value **13**.

1. **Constructors:**

There are two constructors, the first is with three arguments and the second is with no arguments.

The first constructor has the following signature

**Town(String name, long population, double area).**

The second constructor has the following signature

**Town().**

This constructor initializes the **name** to “Maryville”, the population to **POPULATION** (the constant), and the area to **AREA** (the other constant).

1. **Methods:**

There are two methods, **getDensity()** and **getFirstHighestYear().**

The first method **getDensity()** returns a **String**. The **String** returned by this method contains 10 lines. You can use a variable **temp** of type **String** and concatenate it with the labels described below and return it.

This method calculates the **population** and density ( = **population/area**) for the next ten years. For each year, the **temp** string is concatenated with a second string that provides information about the density for that year. The population for any given year is 0.8% more than the previous year. Use a **for** loop to calculate the population and density for the next ten years. The initial year is 2006. The first time through the loop the string constructed should contain density information for 2007. For each year check the density of the population and build a string using appropriate labels as given below.

If density is 830or less then concatenate **temp** with the string

**The population density will be under control in: *year***

Where **year** is the value for the current year.

For a density greater than 830 and up to 840 concatenate **temp** with the same string as above replacing the **under control** part with **moderately high**.

For a density greater than 840 and up to 860 concatenate **temp** with the same string replacing the label **under control** with **high**.

For the density greater than 860 concatenate **temp** with the same string replacing the label **under control** with **extremely high**.

The second method **getFirstHighestYear()** returns an **int.** The integer returned by this method will be value of yearfor which the population density was greater than 860 for the first time. Use a **while** loop in this method to calculate the population and density. Repeat the loop for each year until the population density exceeds 860. *(If the density is 840 in 2007, 858 in 2008, and 863 in 2009 then this method should return 2009).*

1. **Java Documentation**:

Add javadoc comments for the class, each constructor, and each method.

**The TownDriver class:** This class has a method **main** that creates two instances of **Town.** The first **Town** object is instantiated by calling the no-arg **Town** constructor. The second **Town** object is instantiated by calling the **Town** constructor that takes three arguments. The three arguments may be any values of your choice with appropriate data types.Finally call the **getDensity()** and **getFirstHighestYear()**methods for both **Town** objects and print the string returned by each.

*Sample Run:*

For Town with no arg constructor:

The population density will be under control in:2007

The population density will be under control in:2008

The population density will be moderately high in:2009

The population density will be high in:2010

The population density will be high in:2011

The population density will be high in:2012

The population density will be extremely high in:2013

The population density will be extremely high in:2014

The population density will be extremely high in:2015

The population density will be extremely high in:2016

The first year when the density is extremely high:2013

For Hyderabad with 8.1 million population and an area of 260.0 sq miles

The population density will be extremely high in:2007

The population density will be extremely high in:2008

The population density will be extremely high in:2009

The population density will be extremely high in:2010

The population density will be extremely high in:2011

The population density will be extremely high in:2012

The population density will be extremely high in:2013

The population density will be extremely high in:2014

The population density will be extremely high in:2015

The population density will be extremely high in:2016

The first year when the density is extremely high:2006