Rochester Institute of Technology
National Technical Institute for the Deaf
Information and Computing Studies Department

Name: Edward Riley

NACA.161 Programming Fundamentals II
In-class Exercise Day 06 – Accessors, Mutators & Constructors

Overview

These exercises are designed to help expand your knowledge and familiarity with accessor and mutator methods and constructors.

Apartment Class

1) Write a class called **Apartment** that contains two attributes named: **streetAddress** and **monthlyRent**.

2) Set the access modifier for the 2 attributes to **public**.

3) Create a default constructor that sets:
   - **streetAddress** to "10 East Street"
   - **monthlyRent** to $550.00

TestApartment Class

4) Create a class named **TestApartment** that contains a main method.

5) Create an **Apartment** object using the default constructor.

6) Print the values of the two attributes of the **Apartment** object by **directly accessing the attributes.**

   The syntax to access an attribute from another class is **object.attributeName.**

   Why were you able to print the attribute values?

   Because it is inside the scope and thus public.

   There isn't anything really wrong with directly accessing attributes if you are only going to get the values. Now let's see what can go wrong.

7) In the **TestApartment** class, change the value of **monthlyRent** to `-9999.00`. Use the same format as above to refer to the attributes in an assignment statement, namely **object.attributeName.** Now print the two attributes again.

8) Compile and run the **TestApartment** class. If the value of **monthlyRent** did not change to `-9999.00`, correct your program until it does.

What change can you make to prevent the test class from directly changing the value of the attributes?

By altering the method program to private or setting up local variable

9) Change the access modifiers of the attributes in the **Apartment** class to **private.** Compile and run the **TestApartment** class.

Why did it not compile successfully?

Because, it is private to only What is in scope

How do you provide access to **private** attributes?

By changing "private" to "public"

Why is it usually a good idea to set the access modifiers for attributes to private?

When you do not want to leak or show information to public.

10) Create an accessor and a mutator method for each of the attributes, and compile the **Apartment** class.

The mutator **setMonthlyRent** has a return type of void. At this point, we will not validate the monthly rent, which should be non-negative. If the amount of the rent is negative, assign this value to the attribute **monthlyRent**.

Back to the TestApartment Class

11) Change all places where you directly accessed the attributes values to use the appropriate accessor method.

12) Change the line where you directly changed the **monthlyRent** to `-9999.00` to use the appropriate mutator method.

13) Keep compiling and running the **TestApartment** class until it works.

Why did the monthly rent still get set to `-9999.00`?

Because the value was changed to -9999.00

Fix the Apartment Class

14) Modify the code for the mutator for **monthlyRent** to change the attribute's value only if the parameter's value is greater than or equal to zero. But do **not** change the return type of the mutator.

15) Compile the **Apartment** class and run the **TestApartment** class.

Is the monthly rental still equal to `-9999.00`? Yes (No)

If No, why? Number is NOT greater than and therefore false

What value was printed for the monthly rental? ~~[illegible]~~ 555.0

The problem is that the **TestApartment** class was never notified that the **Apartment** class refused to use the invalid value (`-9999.00`).

To correct this problem, the mutator needs to tell the **TestApartment** class that it received an invalid value.

What mechanism can the mutator use to indicate to the **TestApartment** class that it refused to accept a value passed to it?

Either return type or accessor

What data type should the mutator return to indicate that the input was valid or invalid?

Boolean

16) To correct this problem, change the return type of mutator for **monthlyRent** to a boolean. If the value of **monthlyRent** is valid (greater than or equal to 0), change the value of the attribute **monthlyRent** and return a value of true. Otherwise, do not change the value of the attribute and return false.

17) Compile the **Apartment** class and run the **TestApartment** class. Did the test class recognize that an invalid value was passed to the mutator? (Yes) No

Why not? _____

Back to the TestApartment Class

18) If a method returns a value, but the calling class does not store it into a variable, the return value is discarded. Now, modify the **TestApartment** class to save and check the return value from the call to the mutator, and print a message that indicates tha the value passed the mutator caused a return value of false [For example: Invalid monthly rent]. At this point, no input will be entered by the user as the monthly rent is being set to `-9999.00`.

19) Compile and run the **TestApartment** class. If your message does not appear, fix your program until it does.

Saved until Wednesday

20) So far, we have set the value of the parameter to the mutator to `-9999.00`. Now, modify the **TestApartment** class to include a loop to prompt the user to enter the monthly rent using the appropriate method in **Scanner** class.

    Pass the value entered by the user into the mutator. Check the return value from the mutator. If the return value is false, then print a message stating that the input was invalid. If the return value is true, then print a message stating that the input was valid and exit the loop.

21) Compile and run the program. Test your code with a negative value and a positive value.

When you complete all of the steps successfully and answer all of the questions, contact your instructor to check if your program(s) executes correctly and to review your code. We will initial the line below.

_____**Successful execution of code**

If you do not finish the program during the class period, contact your instructor to initial below so that you can complete it before the next class period.

_____**Code not completed during lab time**

You may then have your instructor verify your work at the start of work period in the next class. If you do not have a signature, then you can not receive any points for this assignment.