
Instructor: Michael Hackett
Department: Computer Science
Email: mhackett@ccp.edu

Flooring Quote Calculator

Create a program that calculates a price quote for installing different types of flooring in a house.

Your program must allow the user to enter the homeowner's information, the number of rooms where they want new flooring installed, the area of each room, and the type of flooring for each room. Your program will then calculate and print the installation costs.

Your program will need to use a House class that contains information about the home and home owner. An inner class, Room, will maintain the dimension and flooring options selected. The House object will maintain an array of Room objects.

FloorType Class (FloorType.java)

This class contains an enumerator and requires no modification or changes by you. The classes in the Assignment folder will be able to use this FloorType enumerator without the need for any import statements.

Room Class (Inner Class in House.java)

This class must contain only the following:

- Two fields, all private
 - **area** - A double that represents the square footage of the room.
 - **floorType** - A FloorType object that represents the type of flooring chosen for the room.
- One constructor, public
 - Accepts two arguments, parameters named areaIn (double) and floorTypeIn (FloorType)
 - Assigns the parameter's values to their respective fields.
- Two Methods, both public
 - **Getters** for both fields

House Class (House.java)

This class must contain only the following:

- Eight fields, all private
 - **ownerName** - A String for holding the homeowner's name.
 - **phoneNumber** - A String for holding the homeowner's phone number.
 - **streetAddress** - A String for holding the house's street address.
 - **city** - A String for holding the house's city.
 - **state** - A String for holding the house's state.
 - **zipCode** - A String for holding the house's zip code.
 - **rooms** - An array of Room objects.
 - **roomIndex** - An int that keeps track of what index to place new Room objects in the rooms array.
- One constructor, public
 - Accepts one argument- numRooms
 - The numRooms argument is used to instantiate the rooms field with an array of the provided length (numRooms).
 - Assigns empty Strings to the other six fields and assigns zero to roomIndex.
- Fifteen Methods, both public
 - **Setters and getters** for only the ownerName, phoneNumber, streetAddress, city, state, and zipCode fields (12 methods total, all public)
 - **addRoom** (public, returns void)
 - Accepts two arguments, parameters named sqft (double) and fType (FloorType object)
 - "sqft" represents the square footage of the room.
 - fType is an enumerator object. The FloorType class that contains the enumerator is provided. It's in its own class so all classes for the assignment can use it.
 - The method should use the parameters to instantiate a Room object and add it to the next available index in the array (using the roomIndex field).
 - Remember to increment roomIndex after placing the Room object in the array.
 - **getInstallationCost**(public, returns double, no parameters)
 - This method calculates and returns the total cost of installation.
 - The cost is \$10.00 per square foot, regardless of flooring type.
 - You'll need to use the Room objects in the rooms field to complete this.
 - **getFlooringCost**(public, returns double, no parameters)
 - This method calculates and returns the total cost of flooring.
 - Carpet is \$7.00 per square foot
 - Tile is \$5.00 per square foot
 - Hardwood is \$6.00 per square foot
 - You'll need to use the Room objects in the rooms field to complete this.

FlooringCalculator Class (FlooringCalculator.java)

This is the class that will use the House class you created.

In the main method:

- Prompt the user to enter the customer's name, phone number, address, city, state, zip code, and the number of rooms.
- Instantiate a House object. Supply it with the number of rooms entered by the user.
- Provide the House object with the customer's data, using the appropriate setter methods.
- Prompt the user to enter the area (square footage) and flooring type for each room.
 - You can use "Enter 1 for Carpet, 2 for Tile, and 3 for Hardwood: " or something to that effect for selecting the flooring type. No matter how you implement this, be sure to check for invalid entries.
 - Provide the House object's addRoom method with the appropriate arguments, so that it can add a Room object to the array of Rooms.
- Print the customer's information.
 - Use the getter methods you created in the House object.
- Print the installation cost
 - Use the value returned by the House object's getInstallationCost method.
- Print the flooring cost
 - Use the value returned by the House object's getFlooringCost method.
- Print the total cost
 - Sum of the two previously returned values.

UML Diagram

You will also need to create a UML diagram that shows the House, Room, and FlooringType classes. Be sure to use the correct diagrams and symbols to show the relationship between the three classes. You may do this in MS Word, MS PowerPoint, or even MS Paint (or comparable programs). Acceptable file formats will be PDF, JPG, or PNG.

Sample Input/Output

Enter the customer's name: John Smith
Enter the customer's phone number: 215-555-1234
Enter the customer's street address: 123 Any Street
Enter the customer's city: Anytown
Enter the customer's state: PA
Enter the customer's zip code: 12345
Enter the number of rooms: 3

Enter the area (in square feet) of room 1: 20
Select flooring type (1 for Carpet, 2 for Tile, 3 for Hardwood): 1
Enter the area (in square feet) of room 2: 30
Select flooring type (1 for Carpet, 2 for Tile, 3 for Hardwood): 3
Enter the area (in square feet) of room 3: 10
Select flooring type (1 for Carpet, 2 for Tile, 3 for Hardwood): 1

Price Quote For:
John Smith
123 Any Street
Anytown, PA 12345
215-555-1234

Total Installation Cost: \$600.00
Total Flooring Cost: \$390.00
Total Estimated Cost: \$990.00

- Your program's output must exactly match the formatting in the above example.
- Be sure to use comments to document your code. Comments show me that YOU can explain, in plain English) what your program's code is doing.

Grading

See Assignment Rubric in Canvas.