App Project #1

Create an Android app that helps a home health and wellness store manage a **list** of saunas available to its customers. Each sauna should be represented as an object with the following data members:

- make (string)
- product code #—used to uniquely identify the specific sauna (string)
- wood type (string)
- capacity (# persons) (int)
- kind of heating (string) (for example, electric or infrared)
- price (double)
- image URL: location of an image of the sauna on the Web (string)

In addition to the standard set of accessors for the above data members, define the method "getShipping" (as part of the class Sauna) to return the shipping cost for the given sauna. The shipping cost is based on the sauna's price and capacity. For a capacity of up to three persons, the shipping cost is 5% of the price. For a capacity of four or more persons, it is 8% of the price. Do not store the shipping cost as a separate data member. Calculate it on request.

Your app will use a singleton class SaunaList to create and hold the list of sauna objects.

Your app should have a MainActivity app bar that provides the following options for accessing and manipulating the data on the list. (You can use shorter phrases in the actual app bar.)

1. Display list
2. Load list from file
3. Add a new sauna
4. Show details of a sauna
5. Remove a sauna
6. Adjust price of sauna (by a %)
7. Show all saunas with a given capacity
8. Show the median price of the saunas
9. Save list to file

Details of each option:

– **Option 1:** Display (on the screen in a TextView) the make, product code #, capacity, and price of each sauna in order starting from the first one. Display the information for one sauna per line; something like:

Saunacore, D15F4, 4 persons, $3155.99

– **Option 2:** Load a collection of saunas (that is, sauna data) into the list from a web text (data) file given by the user. The new data from the file ***completely replaces*** the current contents of the list. Make certain to test for the file's existence. If the file does not exist, then indicate this with a message.

The data file will contain sauna information in the following format. Each data item will appear on a separate line. For a single sauna, the make will appear first, followed by the product code #, wood type, etc. I will provide you with an example of such a file. Note that you do **not** need to modify the file to reflect changes in the list as the app runs. See Option 9.

As another example, you should create a file of your own and place it in your "public_html" directory on your AFS account. Include at least three saunas and make certain to have corresponding image (jpeg) files for each.

NOTE: input from an assets file is not an option in this app.

– **Option 3:** Add a new sauna to the *end* of the list. **All** information about the new sauna (including make, product code #, etc.) is to be requested (input) from the user using appropriate EditTexts (on one screen). That is, you will need to ask for 7 pieces of data from the user. You'll, of course, need to create a new sauna object to hold this data.

– **Option 4:** Display in a neat format (using a TextView and SimpleDraweeView in combination) all the information pertaining to the sauna whose product code # is given by the user. Display the following information:

- make

- wood type
- capacity
- kind of heating
- price
- shipping cost
- total cost (= price + shipping)
- image of the sauna

NOTE: If the user inputs a product code # that does *not* exist, then the app should display an error message (using a toast, for example).

– **Option 5:** Remove the sauna whose product code # is given by the user. If the sauna is not on the list, display an error message.

– **Option 6:** Change the price (up or down) of the sauna whose product code # is given by the user by a given percentage. Get the percentage from the user. A positive percentage indicates an increase in price. A negative percentage indicates a reduction in price.

– **Option 7:** Show all saunas with a given capacity. Get that capacity value from the user. Display the make, product code #, wood type, kind of heating, and price for each such sauna; something like:

Saunacore, D15F4, Canadian hemlock, electric, $3155.99

– **Option 8:** Show the median price of the saunas. ***Note:*** in whatever manner you choose to calculate this value, make certain that you do not alter the ordering of the saunas on the list as a result.

– **Option 9:** Save (write) the data pertaining to *all* saunas on the list to a file given by the user. (This file can only be placed in the app's external files directory.) The output for each sauna should be in the same format as in the input file. Make certain to maintain the order of the saunas in the output file as they appear on the list.

NOTE AGAIN: Assume that all options must have some error checking to determine if an input (e.g., file name, product code #) does or does not exist.