Vanier College

Computer Science Department

Data Structures and OOP

Assignment 1

Due Date: By 11:59pm Friday March 8, 2024
Evaluation: 10% of final mark (see marking rubric at the end of handout)
Late Submission: none accepted.

General Guidelines When Writing Programs:

Include the following comments at the top of your source codes :

```
// -------------------------------------------------------

// Assignment 1

// Written by: (include your name and student id)

// For "Data Structures and OOP" Section (include number)- Winter 2024

// --------------------------------------------------------
```

- In a comment, give a general explanation of what your program does. As the programming questions get more complex, the explanations will get lengthier.
- Include comments in your program describing the main steps in your program.
- Display a welcome message.
- Display clear prompts for users when you are expecting the user to enter data from the keyboard.
- All output should be displayed with clear messages and in an easy to read format.
- End your program with a closing message so that the user knows that the program has terminated.

What to Submit:

- Make **one ZIP** file that includes all of the **.java files.**
- Submit the ZIP file via Omnivox.
    - Do not use the RAR (or some other) format!

Assignments not submitted to the correct location or not in the requested format will not be graded.

Q1:

ABC Company helps its clients to find a quick and easy analysis for better investment decisions. It would like to include the ways to calculate the evaluated price and investment analysis for the two types of property, 'Condo' and 'single-family home (SFHome).

Implement a base class `property` which is abstract and two derived classes `Condo` and `SFHome` based on the following specification.

Note: You need to decide which members to be defined in base class and which members in derived classes

- Instance variables:
    - `address` that holds the address of the property and is of type string.
    - `ZoneCode` which is an integer indicating the zone of the property. (possible values `1, 2,` and `3`)
    - `NoOfBedrooms` that holds the number of bedrooms in the property.
    - `YearofConstruction` an integer to hold the year of construction
    - `RiskFactor` which is a float number in the range(0.0,1.0) for the risks related to investments in real estate market.

- At least three `constructors`, a default constructor, a parametrized constructor (which will accept enough parameters to initialize all the attributes of the created object from this class), and a copy constructor.
  Note (for implementing derived classes): An object creation using the default constructor must trigger the default constructor of its ancestor classes, while creation using parametrized constructors must trigger the parametrized constructors of the ancestors.

- `Accessor` and `Mutator` methods
    - In the mutator methods to set the `riskFactor` and `zonecode`, check the value/range, if they are not valid, display appropriate message and exit the system.

- `toString` and `equals` methods:
    - Note: `toString` method must return clear description and information of the object (display each attribute of the object in a separate line)
    - Note: You are always overriding these methods in the derived classes.

- `AnalyzeInvestment` that returns as a double the value computed for the property investment analysis.

- `EvaluatePrice` that returns as a double the price calculated based on the given input data.

`AnalyzeInvestment` and `EvaluatePrice` are calculated according to the following formulas:

Type: Condo

ZoneCode	Base_price
1	300000
2	200000
3	100000

Type: SFHome

ZoneCode	Base_price
1	500000
2	400000
3	300000

```
EvaluatePrice =
             Base_price + (5% * Base_price * NoOfBedrooms) +
            (YearofConstruction * 100)/2
```

```
AnalyzeInvestment= RiskFactor * 50
```

- Test the classes from your main method by creating an array of type Property and adding two instances of Condo and two instances of SFHome to the array. Iterate through the array and output the evaluated price and investment analysis for each property. Ensure that the program operates as expected.

Figure 1 indicates the sample output:

```
***********************
Welcome to ABC Analyzer
***********************

Property 1:
Type: Condo.
Address: 7 Main St.
Zone: 1
No. of Bedrooms:2
Year of Construction: 2012
R Factor:0.1

Investment analysis: 5.0
Evaluated Price: $430600.0

======================
Property 2:
Type: Condo.
Address: 20 Square St.
Zone: 2
No. of Bedrooms:1
Year of Construction: 2020
R Factor:0.02

Investment analysis: 1.0
Evaluated Price: $311000.0

======================
Property 3:
Type: Single-Family Home.
Address: 65 rue College.
Zone: 3
No. of Bedrooms:3
Year of Construction: 1988
R Factor:0.2

Investment analysis: 10.0
Evaluated Price: $444400.0

======================
Property 4:
Type: Single-Family Home.
Address: 110 Flowers rd.
Zone: 1
No. of Bedrooms:4
Year of Construction: 1992
R Factor:0.15

Investment analysis: 7.5
Evaluated Price: $699600.0

======================
Thanks for using our software!
```

Figure 1: Sample output

Q2:

ABC Group is doing research on the Vehicle Fuel Consumption Value (VFCV). In the first step, VFCV data of 20 vehicles is collected (which are integers between 5 and 25 inclusive). In the second step, statistical information is generated based on the following criteria:

a) The number of vehicles which have the VFCV ***equals to*** each integer in the range of 5 and 25 (Each output reflects the total number of vehicles with the same VFCV).

b) The number of vehicles which have the VFCV ***above*** each integer in the range of 5 and 25.

c) The VFCV collected by the ***maximum*** number of vehicles.

d) The ***highest*** VFCV which belongs to ***no*** vehicle.

Note 1: You need to define a separate method for each part (a,b,c, and d).

Here is the name of the functions:

- `getEachVFCV` (for part a)
- `getAboveVFCV` (for part b)
- `maxNumVehiclesVFCV` (for part c)
- `highestVFCVnoVehicle` (for part d)

Note 2: You can define and use other methods as well if you wish.

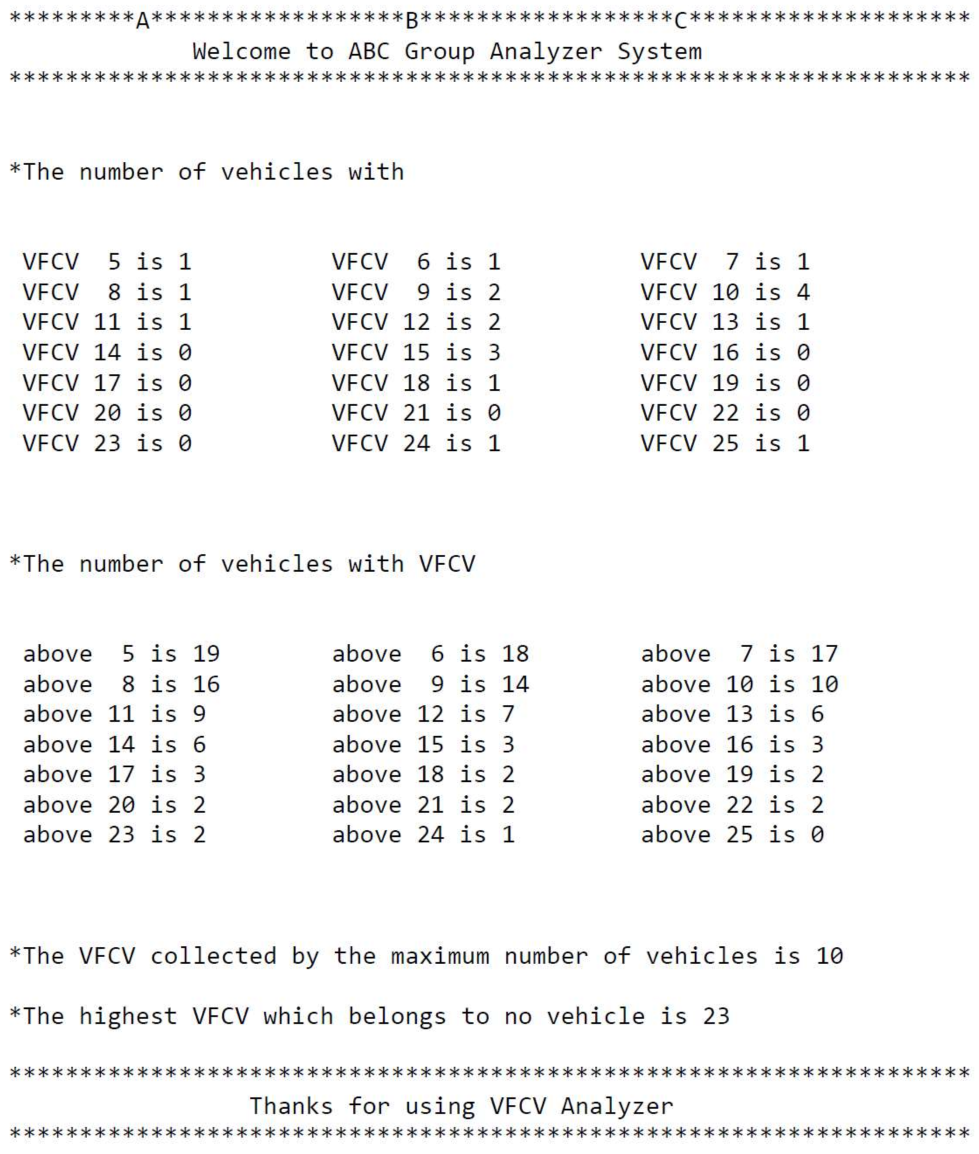
Note 3: You need to use an array data structure to store the VFCVs.

Here is a sample array:

(It can be initialized inside the body of your source code without prompting user)

```
int[] VFCV={8,12,13,15,15,15,10,9,9,10,25,18,7,6,5,10,24,10,11,12};
```

Here is a sample output screen to illustrate the expected behaviour of your program.



```
*********A******************B******************C********************
             Welcome to ABC Group Analyzer System
********************************************************************


*The number of vehicles with


 VFCV  5 is 1         VFCV  6 is 1         VFCV  7 is 1
 VFCV  8 is 1         VFCV  9 is 2         VFCV 10 is 4
 VFCV 11 is 1         VFCV 12 is 2         VFCV 13 is 1
 VFCV 14 is 0         VFCV 15 is 3         VFCV 16 is 0
 VFCV 17 is 0         VFCV 18 is 1         VFCV 19 is 0
 VFCV 20 is 0         VFCV 21 is 0         VFCV 22 is 0
 VFCV 23 is 0         VFCV 24 is 1         VFCV 25 is 1



*The number of vehicles with VFCV


 above  5 is 19       above  6 is 18       above  7 is 17
 above  8 is 16       above  9 is 14       above 10 is 10
 above 11 is 9        above 12 is 7        above 13 is 6
 above 14 is 6        above 15 is 3        above 16 is 3
 above 17 is 3        above 18 is 2        above 19 is 2
 above 20 is 2        above 21 is 2        above 22 is 2
 above 23 is 2        above 24 is 1        above 25 is 0



*The VFCV collected by the maximum number of vehicles is 10

*The highest VFCV which belongs to no vehicle is 23

********************************************************************
                 Thanks for using VFCV Analyzer
********************************************************************
```

Evaluation Criteria for Assignment 1 (60 points)

Programming Style for the assignment	**10 pts**
Comments/description of variables/description of the program (authors, date, purpose)	5 pts
Clear prompts to user & clear message with output	2 pts
Use of significant names for identifiers	1 pts
Indentation and readability	2 pts
Question 1	**25 pts**
Format/clarity/completeness/accuracy of output	13 pts
Proper use of required concepts	12 pts
Question 2	**25 pts**
Format/clarity/completeness/accuracy of output	13pts
Proper use of required concepts	12 pts
Total	**60 pts**