

PROBLEM STATEMENT:

The creation of a simple hierarchy of classes along with implementation and use of inheritance and use of an array of objects.

Notes:

This exercise is a continuation of Lab Exercise 6A.

Extend your Lab 6A program. You will modify and extend your Lab 6A program as described below.

CODE :

Modify the Main driver class from the previous lab exercise as described below.

Within the main method, declare and create an array of the Loan data type. This array will be able to hold instances of the Loan class and any of its subclasses. This is true because, in general, every instance of a subclass is an instance of its parent class.

Declare the array so that it consists of 50 cells, so that it can hold up to 50 Loan instances.

Eliminate the variables named ln1, ln2, ln3, ln4, ln5, ln6, ln7. Totally eliminate the use of any of these variables from the program.

You will need to keep and continue to use the single Loan variable ln.

Eliminate the two switch statements that deal with selecting among the variables ln1, ln2, ln3, ln4, ln5, ln6, ln7. Replace each of the switch statements with a one-line statement that uses the appropriate element of the array.

This Lab Exercise assumes that, as part of Lab 6A, you already added code within the while loop to prompt the user for the type of loan and input the type from the user. Be sure to present the list of available types to the user, namely (1) Residential Loan, (2) Commercial Loan, (3) Agricultural Loan and (4) Unspecified Loan. Tell the user what to enter to specify the loan type. For example, you could show the numbers for the list loans, and tell the user to enter the number of their choice. An alternative would be to have the user enter the first letter of the type name. An unspecified loan should also be the default if an incorrect loan type is entered.

This Lab Exercise assumes that you eliminated the following statement that was in the previously existing driver program to create the loan object (instance).

ln = new Loan(borrowerNameStr, loanId, propertyCost, downPayment, numOf years);

You replaced the statement with code that creates the correct type of loan object (instance) based on the loan type that the user entered. The program should create instances of the UnspecifiedLoan, AgriculturalLoan, CommercialLoan, and ResidentialLoan classes, which are described below. The program should not create instances of the Loan class.

Modify the for-loop that creates the final display list of all the loans of all types. This for-loop is after the while(true) loop and iterates though only the actual (non-null) elements of the array. In order to accomplish this, do the following:

Replace the switch statement with a one-line statement that selects the correct element of the array.

In the body of the for-loop, in the statement that concatenates to the variable outputStr2, you should already have added code (as part of Lab 6-A) to also display the loan type on a separate line. Use the following code to concatenate the loan type into your outputStr2 string:

... + "Loan Type: " + ln.getClass().getName() + ...

Be sure that your display includes all the data on each loan. Some information will be displayed together on one line. See the screen capture displays below.
Keep the statement that appears after the for-loop and uses a JOptionPane dialog to display the contents of outputStr2.

Create a new public class, called CommercialLoan, that is a subclass of Loan. This new class must be in a separate file with the same name as the class. You can create the new class in the same package (folder) that contains the Main class (file) and Loan class.

The CommercialLoan class must inherit all the member variables of the Loan class.
Do not declare any member variables in the CommercialLoan class.

The CommercialLoan class must have a public constructor that takes a value for each of the inherited member variables as parameter. The inherited variables are: name, loan ID, property cost, down payment, number of years and annual interest rate(9.2%). The constructor must invoke the constructor of the parent class to initialize the variables. The CommercialLoan constructor should not initialize the variables within its own body. The current commercial rate is 9.2%.

The CommercialLoan class must have a public computeMonthlyPayment method that calculates the monthly payment by simply invoking the computeMonthlyPayment method in the parent class after implementing the following specifications (1) for interest rates specified below and (2) for a new specification: the commercial loan must not exceed 15 years. If it does, reset the number of years to 15. That is, CommercialLoans yield their typical cost but be limited to a 15 year maximum. The current rate is 9.2%. However, if the down payment is less than 30% of the property cost, the commercial interest rate is then 9.6%.

The CommercialLoan class must have a public toString method that concatenates together and returns the following:

The string returned by invoking the parent class toString method to get a string that includes the values of all the member variables for a CommercialLoan instance.

Note: The type of the loan should be included in the string returned by the parent class toString method.

The result of formatting the return value from invoking the computeMonthlyPayment method that is within the CommercialLoan class.

Create a new public class, called AgriculturalLoan, that is a subclass of Loan. This new class must be in a separate file with the same name as the class. You can create the new class in the same package (folder) that contains the Main class (file) and Loan class.

The Agriculturalloan class must inherit all the member variables of the Loan class. Do not declare any member variables in the AgriculturalLoan class.

The AgriculturalLoan class must have a public constructor that takes a value for each of the inherited member variables as parameter. The inherited variables are: name, loan ID, property cost, down payment, number of years and annual interest rate(5.25%). The constructor must invoke the constructor of the parent class to initialize the variables. The AgriculturalLoan constructor should not initialize the variables within its own body. The current agricultural rate is 5.25%.

The AgriculturalLoan class must have a public computeMonthlyPayment method that calculates the monthly payment by simply invoking the computeMonthlyPayment method in the parent class after implementing the following specifications (1) for interest rates specified below and (2) for a new specification: the agricultural loan must not exceed 20 years. If it does, reset the number of years to 20. That is, AgriculturalLoans yield their typical cost but be limited to a 20 year maximum. The current rate is 5.25%. However, if the down payment is less than 10% of the property cost, the agricultural interest rate is then 6.25%.

The AgriculturalLoan class must have a public toString method that concatenates together and returns the following:
The string returned by invoking the parent class toString method to get a string that includes the values of all the member variables for a AgriculturalLoan instance.

Note: The type of the loan should be included in the string returned by the parent class toString method.

The result of formatting the return value from invoking the computeMonthlyPayment method that is within the AgriculturalLoan class.

Create a new public class, called UnspecifiedLoan, that is a subclass of Loan. This new class must be in a separate file with the same name as the class. You can create the new class in the same package (folder) that contains the Main class (file) and Loan class.

The UnspecifiedLoan class must inherit all the member variables of the Loan class. Do not declare any member variables in the UnspecifiedLoan class.

The UnspecifiedLoan class must have a public constructor that takes a value for each of the inherited member variables as parameter. The inherited variables are: name, loan ID, property cost, down payment, number of years and annual interest rate(9.8%). The constructor must invoke the constructor of the parent class to initialize the variables. The UnspecifiedLoan constructor should not initialize the variables within its own body. The current unspecified rate is 9.8%.

The UnspecifiedLoan class must have a public computeMonthlyPayment method that calculates the monthly payment by simply invoking the computeMonthlyPayment method in the parent class after implementing the following specifications (1) for interest rate of 9.8% and (2) for a new specification: the unspecified loan must not exceed 5 years. If it does, reset the number of years to 5. That is, UnspecifiedLoans are limited to a 5 year maximum. The current rate is 9.8%. However, if the down payment is less than one third of the property cost, the unspecified interest rate is then 10%.

The UnspecifiedLoan class must have a public toString method that concatenates together and returns the following:

The string returned by invoking the parent class toString method to get a string that includes the values of all the member variables for a UnspecifiedLoan instance.

Note: The type of the loan should be included in the string returned by the parent class toString method.

The result of formatting the return value from invoking the computeMonthlyPayment method that is within the UnspecifiedLoan class.

Compile, execute, and debug the program. Test and debug your program using a variety of input data that includes the use of all four subclasses. Run the program and make window captures of the dialogs displaying input / output.

Answer the Question Items using the Java JDK Documentation. You downloaded and installed the documentation on your computer as per the instructions in the syllabus. Determine the answers to the questions listed below and report your answers in your ReadMe for this project.

Do the following three question items using the Java Platform API Specification. As stated above, report your results in your ReadMe doc file and label your results for each in the ReadMe with the numbers indicated below (namely 1, 2, 3 or i, ii, iii).

i. Significance of the Object class. Examine the documentation on the Object class and read the brief paragraph that describes the Object class. This paragraph states what is unique about this class. Copy and paste this paragraph it into your ReadMe as the answer to question item 1, the importance of the Object class. Note that an object of the class named Class is returned by the getClass method in the Object class. Note that the getClass method is used in Step 2 of this exercise.

ii. Return data type of getName method. Examine the documentation on the Class class and look at the information on the getName method. State what data type is returned by the getName method.

iii. Purpose of the getName method. Report what the method does by copying the description/explanation of the method and pasting it into your ReadMe

Create batch file.

Create a batch file as you did for the previous assignments so that your program can be executed without the use of NetBeans.
Be sure that the batch file is within your top level NetBeans project folder.

As usual create ReadMe.pdf file.

Create a file named ReadMe.pdf

In this document, insert your name at the top, and on the next line insert the assignment number

Then enter any comments regarding the assignment and your program.

Then insert several window captures of windows showing the inputs and outputs from the execution of the program.

Be sure the ReadMe file is within your top level project folder.

Zip the project folder and all its contents.

Change the name the zip file so that its name consists of your name along with the assignment number, as follows: " LastName_ Lab_06A_cs209.zip".

Do not use spaces in the name of the file, use underscores or hyphens instead.

Deliverables:

Send to duttat@ecc.edu an email this the exact subject

cs209_ Lab_06A

In this email attached the above named zip file

LastName_ Lab_06A_cs209.zip

Due Date : 5:00pm 16 October 2014

Confirm

?

1. Loan Type: lab06_v3.ResidentialLoan, Name: John Smith ID: 1111, Cost: \$100000.00 Down Payment: \$20000.00, Years: 30, Rate: 5.75% , Monthly Payment \$466.86
Max number of allowed Loans is: 50
Continue?

Yes

No

Confirm


?

1. Loan Type: lab06_v3.ResidentialLoan, Name: John Smith ID: 1111, Cost: \$100000.00 Down Payment: \$20000.00, Years: 30, Rate: 5.75% , Monthly Payment \$466.86
2. Loan Type: lab06_v3.ResidentialLoan, Name: David Smith ID: 2222, Cost: \$100000.00 Down Payment: \$10000.00, Years: 30, Rate: 6.75% , Monthly Payment \$583.74
3. Loan Type: lab06_v3.CommercialLoan, Name: Ellen Andrews ID: 3333, Cost: \$100000.00 Down Payment: \$30000.00, Years: 15, Rate: 9.20% , Monthly Payment \$718.34
4. Loan Type: lab06_v3.CommercialLoan, Name: Tom Andrews ID: 4444, Cost: \$100000.00 Down Payment: \$20000.00, Years: 15, Rate: 9.60% , Monthly Payment \$840.21
5. Loan Type: lab06_v3.AgriculturalLoan, Name: Max Zaccharia ID: 5555, Cost: \$100000.00 Down Payment: \$15000.00, Years: 20, Rate: 5.25% , Monthly Payment \$572.77
6. Loan Type: lab06_v3.AgriculturalLoan, Name: Mickey Jones ID: 6666, Cost: \$100000.00 Down Payment: \$9000.00, Years: 20, Rate: 6.25% , Monthly Payment \$665.14
7. Loan Type: lab06_v3.UnspecifiedLoan, Name: Jane Jones ID: 7777, Cost: \$100000.00 Down Payment: \$34000.00, Years: 5, Rate: 9.80% , Monthly Payment \$1395.82
Max number of allowed Loans is: 50
Continue?

Yes

No

Loan Summary



Loan list:

1. John Smith ID No: 1111

Loan Type: lab06_v3.ResidentialLoan

Property Cost: \$100000.00 Loan Amount: \$80000.00

Number Of Years: 30 Annual Interest Rate: 5.75%

Monthly Loan Payment: \$466.86 Total Amount Paid for loan: \$168068.98

2. David Smith ID No: 2222

Loan Type: lab06_v3.ResidentialLoan

Property Cost: \$100000.00 Loan Amount: \$90000.00

Number Of Years: 30 Annual Interest Rate: 6.75%

Monthly Loan Payment: \$583.74 Total Amount Paid for loan: \$210145.78

3. Ellen Andrews ID No: 3333

Loan Type: lab06_v3.CommercialLoan

Property Cost: \$100000.00 Loan Amount: \$70000.00

Number Of Years: 15 Annual Interest Rate: 9.2%

Monthly Loan Payment: \$718.34 Total Amount Paid for loan: \$129301.02

4. Tom Andrews ID No: 4444

Loan Type: lab06_v3.CommercialLoan

Property Cost: \$100000.00 Loan Amount: \$80000.00

Number Of Years: 15 Annual Interest Rate: 9.6%

Monthly Loan Payment: \$840.21 Total Amount Paid for loan: \$151238.49

5. Max Zaccharia ID No: 5555

Loan Type: lab06_v3.AgriculturalLoan

Property Cost: \$100000.00 Loan Amount: \$85000.00

Number Of Years: 20 Annual Interest Rate: 5.25%

Monthly Loan Payment: \$572.77 Total Amount Paid for loan: \$137464.21

6. Mickey Jones ID No: 6666

Loan Type: lab06_v3.AgriculturalLoan

Property Cost: \$100000.00 Loan Amount: \$91000.00

Number Of Years: 20 Annual Interest Rate: 6.25%

Monthly Loan Payment: \$665.14 Total Amount Paid for loan: \$159634.72

7. Jane Jones ID No: 7777

Loan Type: lab06_v3.UnspecifiedLoan

Property Cost: \$100000.00 Loan Amount: \$66000.00

Number Of Years: 5 Annual Interest Rate: 9.8%

Monthly Loan Payment: \$1395.82 Total Amount Paid for loan: \$83749.13

OK

6