Application Challenge – Financial Calculation Classes

Purpose: To develop and test Java classes utilizing enumerations, inheritance, and interfaces.

Problem: Your team lead has assigned you to develop two classes that will be added to the firm's Java "finance" package. These classes will perform useful financial calculations that will be used in a variety of applications throughout the firm. You must include full API documentation for your classes. You have also been assigned to unit test the classes you add to the package.

Project Requirements:

1. The classes you develop shall be called "LoanPayment" and "Investment".
2. Both classes shall be added to the "finance" package and shall utilize existing code in the package as follows:
   - Inherit the TVMEngine class
   - Use the CompoundingOptions enumeration class
   - Implement the ReportGenerator interface.
   -
3. ***Loan Payment functions:*** The LoanPayment class shall provide the following capabilities:
   - Calculate the periodic payment required to pay off a loan given the amount purchased, down payment, interest rate (APR), compounding, and loan duration.
   - The periodic payment shall be provided by the getValue method (required of all classes that inherit TVMEngine).
   - The periodic payment shall be produced as text formatted as US currency.
   - Produce a text summary report of the loan as shown in the example below:

     ```
     Loan Payment Summary
     Purchase Amount: $120,000.00
     Down Payment: $20,000.00
     Amount Financed: $100,000.00
     APR: 5.0%
     Compounding: monthly
     Loan Duration (years): 30.0
     Payment (monthly): $536.82
     ```

   - The summary report shall be provided by the generateReport method required by the ReportGenerator interface.
4. ***Investment functions:*** The Investment class shall provide the following capabilities:
   - Calculate the future value of an investment given an initial investment amount, periodic investment amount, annual interest rate, compounding, and length of the investment in years.
   - The investment future value shall be provided by the getValue method (required of all classes that inherit TVMEngine).
   - The investment future value shall be produced as text formatted as US currency.

- Produce a text summary report of the investment as shown in the example below:

  ```
  Investment Value Summary
  Initial Investment: $10,000.00
  Periodic Investment (monthly): $1,000.00
  Annual Return: 10.0%
  Investment Value after 20.0 years: $832,649.57
  ```

- The summary report shall be provided by the generateReport method required by the ReportGenerator interface.

5. Detailed specifications of the classes and methods required for this project are provided in Appendices A and B.

6. A full API must be provided for all classes developed. The API must be embedded in the software such that a documentation package can be generated using the Javadoc tool.

Design Goals:
Application software will be written to be easily maintained (i.e., easily understood and modified). Application software will make efficient use of the code modules provided with this assignment as well as the Java libraries.

Task List:
1. Implement the Java modules that meet the project requirements and design goals listed above. Use the NetBeans IDE to develop the code.
   With the exception of the TVMEngine class, CompountingOption class, and ReportGenerator interface, all code submitted must be your group's original work.

2. Test your modules using the test vectors provided in Appendix C. Write a test report using the test report template supplied on Blackboard. Your test report must include the following:
   - Test vectors showing test results for each test case (i.e., indication that your code either did or did not produce the expected results)
   - A screen shot of the test output for one of the test cases specified in the test vectors.
   - Include the test report in the "doc-files" folder of your project (in PDF format).

Grading
The grading policy for this project is as follows:

- Zero points will be awarded for Javadocs if the Javadoc tool cannot successfully run on your project.
- Your project will be evaluated using the instructor's test software. Therefore, strict adherence to the API specified in the assignment is mandatory, or your code will not run on the test platform. Zero points will be awarded for both performance and the test report if your code doesn't run on the instructor's test platform.

Project Grading Rubric

Item	Points
LoanPayment Class	
Design Goals	5
Javadocs	
Constructor	5
getValue ()	3
generateReport ()	3
Performance	
getValue ()	12
generateReport ()	12
Investment Class	
Design Goals	5
Javadocs	
Constructor	5
getValue ()	3
generateReport ()	3
Performance	
getValue ()	12
generateReport ()	12
Test Report	
Screen Shot	4
Test Cases	16
Project Total	**100**

Appendix A – LoanPayment Class Detailed Specification

The LoanPayment class calculates the periodic payment needed to pay off a loan. It also provides a summary report of the loan parameters. The class interface is based on a purchase amount and a down payment. Internally, the class calculates the amount to be financed by the loan as the purchase amount less the down payment. This class inherits the TVMEngine class and implements the ReportGenerator interface.

Exhibit A-1. LoanPayment Class Diagram

LoanPayment
+ LoanPayment (double, double, double, CompoundingOption, double)
+ getValue (): String
+ generateReport (): String

Constructor Requirements

The class constructor creates a LoanPayment object configured to calculate a loan payment. The loan parameters are provided to the object through the constructor.

Constructor declaration and parameter definitions shall be as follows:

```
public LoanPayment (double purchaseAmount,
                    double downPayment,
                    double interestRate,
                    CompoundingOption compounding,
                    double loanDuration)
```

`purchaseAmount` - the amount of the purchase to be made by the borrower
`downPayment` – the down payment to be made by the borrower against the purchase amount
`interestRate` – the annual percentage rate (APR) to be applied to the loan.
`compounding` – indicates how often interest is added to the loan principal.
`loanDuration` - the duration of the loan in years

The CompundingOption class is part of the finance.enum package.

Method Requirements: getValue

Provides the periodic payment required to pay off the amount financed. The payment is provided as text formatted as US currency.

Method declaration, parameter definitions, and return value shall be as follows:

```
public String getValue ()
```

`return` - A String object representing the loan payment formatted as US currency rounded to two decimal places.

Method Requirements: generateReport

Provides a text summary report of the loan. The report includes the amount purchased, down payment, interest rate (APR), compounding, loan duration, and periodic payment.

Method declaration, parameter definitions, and return value shall be as follows:

```
public String generateReport ()
```

`return` - A String object containing a summary of the loan parameters

Appendix B – Investment Class Detailed Specification

The Investment class calculates the future value of an investment after a specified number of years. It also provides a summary report of the investment parameters. This class inherits the TVMEngine class and implements the ReportGenerator interface.

Exhibit B-1. Investment Class Diagram

Investment
+ Investment (double, double, double, CompoundingOption, double)
+ getValue (): String
+ generateReport (): String

Constructor Requirements

The class constructor creates an Investment object configured to calculate future value of an investment. The investment parameters are provided to the object through the constructor.

```
public Investment (double initialInvestment,
                   double periodicPayment,
                   double interestRate,
                   CompoundingOption compounding,
                   double yearsInvested)
```

`initialInvestment` - the amount invested at the beginning of the investment term
`periodicPayment` – the amount invested at regular intervals over the length of the investment
`interestRate` – the return on the investment. This parameter is expressed as an annual percentage rate (APR).
`compounding` – indicates how often interest is added to the value of the investment.
`yearsInvested` - the length of time the investment is made in years

The CompundingOption class is part of the finance.enum package.

Method Requirements: getValue

Provides the future value of an investment as text formatted as US currency.

Method declaration, parameter definitions, and return value shall be as follows:

`public String getValue ()`

`return` - A String object representing the investment future value formatted as US currency rounded to two decimal places.

Method Requirements: generateReport

Provides a text summary report of the investment. The report includes the initial investment amount, periodic investment amount, annual return, compounding, length of the investment in years, and the future value of the investment.

Method declaration, parameter definitions, and return value shall be as follows:

`public String generateReport ()`

`return` - A String object containing a summary of the investment parameters

Appendix C – Class Test Vectors

Exhibit C-1. LoanPayment Class getValue () Test Vector

Test Case	Purchase Amount	Down Payment	APR (%)	Duration (years)	Compounding	Periodic Payment
1	350000	0	10	15	Annual	$46,015.82
2	350000	50000	10	15	Annaul	$39,442.13
3	350000	0	10	15	Semiannual	$22,768.00
4	350000	0	10	15	Quarterly	$11,323.69
5	350000	0	10	15	Monthly	$3,761.12
6	350000	0	10	15	Weekly	$866.75

Exhibit C-2. LoanPayment Class generateReport () Test Vector

Test Case	Inputs	Output
1	Pruchase Amount: 350000 Down Payment: 50000 APR: 10 Duration: 15 Compounding: Annual	Loan Payment Summary Purchase Amount: $350,000.00 Down Payment: $50,000.00 Amount Financed: $300,000.00 APR: 10.0% Compounding: annual Loan Duration (years): 15.0 Payment (annual): $39,442.13
2	Pruchase Amount: 350000 Down Payment: 50000 APR: 10 Duration: 15 Compounding: Monthly	Loan Payment Summary Purchase Amount: $350,000.00 Down Payment: $50,000.00 Amount Financed: $300,000.00 APR: 10.0% Compounding: monthly Loan Duration (years): 15.0 Payment (monthly): $3,223.82

Exhibit C-3. Investment Class getValue () Test Vector

Test Case	Initial Investment	Periodic Investment	APR (%)	Duration (years)	Compounding	Future Value
1	0	100	10	20	Annual	$5,727.50
2	100	100	10	20	Annaul	$6,400.25
3	0	100	10	20	Semiannual	$12,079.98
4	0	100	10	20	Quarterly	$24,838.27
5	0	100	10	20	Monthly	$75,936.88
6	0	100	10	20	Weekly	$331,493.67

Exhibit C-4. Investment Class generateReport () Test Vector

Test Case	Inputs	Output
1	Initial Investment: 100 Periodic Investment: 100 APR: 10 Duration: 20 Compounding: Annual	Investment Value Summary Initial Investment: $100.00 Periodic Investment (annual): $100.00 Annual Return: 10.0% Investment Value after 20.0 years: $6,400.25
2	Initial Investment: 100 Periodic Investment: 100 APR: 10 Duration: 20 Compounding: Monthly	Investment Value Summary Initial Investment: $100.00 Periodic Investment (monthly): $100.00 Annual Return: 10.0% Investment Value after 20.0 years: $76,669.69