CSCI 1130A/B Introduction to Computing Using Java 2019-2020 First Term

Department of Computer Science and Engineering
The Chinese University of Hong Kong

Assignment 2: Future Value

Due date: 9 October 2019 (Wed) Full mark: 100

Expected normal time spent: 6 hours

Aims:

1. To implement future value calculation for growing asset by writing a Java console application.
2. To practise the use of variables/expression, looping/branching statements and basic Math functions.

Background:

Future value (FV) is the value of an asset at a specific date. It measures the future sum of money at a specified time in the future assuming a certain interest rate; it is the present value, or *principal*, *multiplied by the accumulation function*. Compound interest is commonly applied where the interest shall be added to the principal sum of deposit for accumulation, or in other words, interest on interest. For example, given the principal amount of money ***P***, annual interest rate ***r***, the ***FV*** being accumulated after ***T*** years is given by:

$$\mathrm{FV} = \mathrm{P} \times (1 + \mathrm{r})^{T}$$

In this assignment, we are going to deal with principal between $10,000.00 to $109,700.00, fixed annual interest rate from 1.0% to 10.0%, timespan of 2 to 10 years, while compounding period is assumed to be either monthly, half-yearly or annually where it could be accounted with a slight update:

$$\mathrm{FV} = \mathrm{P} \times (1 + \frac{r}{m})^{Tm}$$

where *m* is the compounding frequency per year

In the output, we shall print out the first 2 (at most) and the last future value (see examples shown later).

Another common inquiry related to future value is that given the same annual interest rate and compounding period, how long it shall take to double or triple the asset? [Hint: To determine the required duration, one could take a natural logarithm to both sides and obtain the result by re-arranging terms.]. For example, in order to double an asset, we can start with the following:

$$2\mathrm{P} = \mathrm{P} \times (1 + \frac{r}{m})^{Tm}$$

$$log_e 2 = T \times m \times log_e(1 + \frac{r}{m})$$

…

In our assignment, we are going to determine the time, in years, to double (×2), triple (×3) and quadruple (×4) a given asset. You can leave the result as approximate number of years, up to two decimal places, with no need to round up to the nearest compounding period.

Problem Definition:

In this assignment, you are required to write a program to perform future value calculation. You are required to obtain the input from the console standard input using ***Scanner*** and ***System.in***. A sample run of the program is shown below:

```
Input Principal [$10000.00 - $109700.00]: 50000.0
Input Annual Interest Rate [1.0% - 10.0%]: 10.0
Input Timespan [2 - 10 years]: 10
Input Compounding Period [1, 6 or 12 months]: 6
Future Value after 6 months: 52500.00
Future Value after 12 months: 55125.00
...
Future Value at the end: 132664.89
Time to obtain asset x2: 7.10 years
Time to obtain asset x3: 11.26 years
Time to obtain asset x4: 14.21 years
```

You are **NOT** required to validate the input numbers types; meaning that you can assume all input numbers are of valid number type. **However**, you are **REQUIRED** to validate the input range as indicated above, e.g. timespan should be in-between 1 to 10, warning should be given to user on invalid input and request input again until the entered value is within the range e.g.:

```
Input Timespan [2 - 10 years]: 12
Invalid Timespan, please enter again.
Input Timespan [2 - 10 years]: 8
Input Compounding Period [1, 6 or 12 months]: 6
```

The number of printed future value should depend on the total number of compounding periods. For 2 compounding periods only, the output should be:

```
Input Timespan [2 - 10 years]: 2
Input Compounding Period [1, 6 or 12 months]: 12
Future Value after 12 months: 10100.00
Future Value at the end: 10201.00
```

For 3 compounding periods, we shall print:

```
Input Timespan [2 - 10 years]: 3
Input Compounding Period [1, 6 or 12 months]: 12
Future Value after 12 months: 10100.00
Future Value after 24 months: 10201.00
Future Value at the end: 10303.01
```

For more than 3 compounding periods, we shall replace some of the items with "…":

```
Input Timespan [2 - 10 years]: 2
Input Compounding Period [1, 6 or 12 months]: 6
Future Value after 6 months: 10500.00
Future Value after 12 months: 11025.00
...
Future Value at the end: 12155.06
```

You shall define one package **`exercise`** and one class called **`FutureValue`** in a new NetBeans project named **`FutureValue`**. The class shall contain a main method that performs all the operations including input, conversion and output. Optionally, you may define more than one method in addition to the main method to carry out the calculation.

Procedure:

1. Create a new NetBeans project with a source file named **`FutureValue.java`** that contains the class **`FutureValue`**, under package **`exercise`**.
2. Define the main method, together with optionally other fields and methods (if any), to perform I/O as well as future value calculation in the **`FutureValue`** class.
3. If you have completed writing the **`FutureValue`** class, try build the project (press the function key ***[F11]*** on the keyboard). If there are errors, don't panic. Double-click on the first error message in the Output window. Check the error, correct it and re-compile. Feel tired? Take a rest.
4. If you have many opened projects, close others or click menu ***[Run] [Set Main Project]*****.**
5. You may insert `println()` statements in your work to inspect variables and intermediate results.
6. When you finish and there is no more error, you are ready to try out the program by pressing the function key ***[F6]*** on the keyboard. Then you can type the input in the standard input. Enjoy!

Submission:

1. **Locate** your NetBeans project folder, e.g. **H:\FutureValue**.
2. ZIP the project folder **FutureValue** and Submit the file **FutureValue.zip** via our Online Assignment Collection Box on Blackboard **https://blackboard.cuhk.edu.hk**

Marking Scheme and Notes:

1. Name your project, package and file correctly.
2. Corresponding student name, student ID and date should be filled into the specified position in the annotation.
3. The indentation of the code in the .java file should make the code easy to read. The .java file should contain comments to indicate the computational logic.
4. The program should be free of any compilation errors such as missing semicolon, type mismatch, etc. *Deduct 5 marks* for each kind of *correctable* compilation error (max deduction: 10.) If your code suffers from only one or two minor and correctable error, marker will try to apply adequate correction and attempt grading the functionalities of the work, i.e., checking output format and applying the test cases.
5. The program should be runnable. *Deduct 5 marks* for each kind of runtime exception or error such as NullPointerException, DivisionByZero, etc. (max deduction: 10.)
6. Input and output format will be checked, for examples, spelling mistakes, user input handling, spacing, etc.
7. To evaluate the accuracy of the program, the program has to pass 4 test cases (15 marks each). Such test cases shall not be released in advance, so test your work using different sets of inputs.

Score distribution is as follows:

Category	Sub-category	Score
1. Naming	Project naming	5
	Package naming	5
	File naming	5
2. Personal Information		5
3. Coding Style	Indentation	5
	Comment	5
4. Compilation	Compilation error	-5 each; -10 max
5. Running	Runtime error	-5 each; -10 max
6. Functionality: I/O Format	User input prompt and output	10
7. Functionality: Result	Test Case 1	15
	Test Case 2	15
	Test Case 3	15
	Test Case 4	15
Total		**100**

Remember to do your submission before 6:00pm of the due date. **NO late submission would be accepted.**

If you submit multiple times, **ONLY** the content and timestamp of the **latest** one would be counted. You may delete (i.e. take back) your attached file and re-submit. We shall take into account the last submission **ONLY**.

University Guideline for Plagiarism:

Attention is drawn to University policy and regulations on honesty in academic work, and to the disciplinary guidelines and procedures applicable to breaches of such policy and regulations. Details may be found at http://www.cuhk.edu.hk/policy/academichonesty/.

With each assignment, students are required to submit a statement that they are aware of these policies, regulations, guidelines and procedures, in a header comment block.

Faculty of Engineering Guidelines to Academic Honesty:

MUST read: https://www.erg.cuhk.edu.hk/erg/AcademicHonesty
(you may need to access via CUHK campus network/ CUHK1x/ CUHK VPN)