

Assignment 3

Handout: **Monday, 17 October 2021**
Due: **23:59, Friday, 28 October 2021**

Goals:

- To understand better the importance of information hiding, inheritance, polymorphism, and dynamic binding;
- To design and implement Java classes;
- To get more familiar with the IntelliJ Idea IDE.

Attention

You MUST NOT change the names of the Java files or the signatures/return types of the methods in those files. We use tests to grade your assignments and the tests refer to the classes and methods defined in the given Java files, as is done in the released sample tests. Failed tests due to changes to the file names or method signatures/return types will be treated the same as failed tests due to incorrect implementations.

1. Employee and Manager (10 points)

Read files `SalaryLevel.java`, `Employee.java`, and `Manager.java` in the assignment project, then add the missing code so that 1) the tests in `EmployeeTest.java` and `ManagerTest.java` will all pass and 2) the classes meet all the requirements specified in the Java files.

What to Do:

[Task 1] Add necessary code in `SalaryLevel.java`, `Employee.java`, and `Manager.java`.

2. Base-N Integer (28 points)

In base-N numbering, totally N different digits are used to represent integers and the actual value of a digit is determined by the position of that digit in integers. For example, in the widely used base-10 numbering, ten digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 are used, and digit 5 would have values 5×10^0 , 5×10^1 , 5×10^2 , etc., when used in positions 0 (i.e., the right most position), 1 (i.e., the position to the immediate left of position 0), 2 (i.e., the position to the immediate left of position 1), etc.

Class `BaseNIntegerUnsigned` abstracts and manipulates unsigned (i.e., non-negative) base-N integers ($1 < N < 27$). Each unsigned base-N integer is represented in the class using a `base` and a `magnitude`, where the `base` is a regular `int` value in decimal, and the `magnitude` is the string of its digits. Note that the class is using the first N uppercase English letters (starting with A for 0) for the digits required by base-N numbering, and that the 26 uppercase letters are enough for the encoding here since N is smaller than 27. For example, base-5 integers will use digits A, B, C, D, and E when abstracted with `BaseNIntegerUnsigned`. Given a `BaseNIntegerUnsigned` object with `base` being 5 and `magnitude` being CBA, its digits C, B, and A are in positions 2, 1, and 0, respectively. Correspondingly, the value of the object can be calculated as $2 \times 5^2 + 1 \times 5^1 + 0 \times 5^0 = 55$ in decimal.

Please read the existing code and comments in class `BaseNIntegerUnsigned` carefully, and then complete the methods in the class to satisfy their corresponding requirements.

Note

1. You should not convert `BaseNIntegerUnsigned` objects to 'int' values in operations like compare, add, and subtract since they may not be representable as 'int' values. Instead, you need to mimic how we would compare, add, and subtract those integers manually. For example, given two base-3 numbers CCC (222) and CC (22), to add the two numbers, you start from the digits in position 0. $C + C$ ($2 + 2$) gives you B (1) in position 0 and a carry 1 to position 1. Then, $C + C + 1$ ($2 + 2 + 1$) in position 1 gives you C (2) in position 1 and a carry 1 to position 2. In position 2, we have $C + A + 1$ ($2 + 0 + 1$), which leads to A (0) in position 2 and a carry 1 to position 3. Therefore, in the end, the result should be BACB (1021).
2. In Class `BaseNIntegerUnsigned`, we throw an `IllegalArgumentException` object to indicate a fatal error has occurred. Therefore, you do not need to consider those error conditions when implementing the methods.

What to Do:

[Task 2] Add necessary code in `BaseNIntegerUnsigned.java`.

3. Inheritance (12 points)

Read the file `InheritanceRelation.java` in the assignment project, then complete Tasks 3 through 8 specified in the comments so that 1) all the classes in `InheritanceRelation.java` compile successfully and 2) all the tests in `InheritanceRelationTest.java` pass.

What to Do:

[Task 3] Read and complete all the tasks in the file `InheritanceRelation.java`. Note that you are NOT allowed to add new fields or methods to the classes.

What to Hand in

The whole Assignment3 folder with the completed methods in a ZIP file. The recommended procedures to obtain the ZIP file using IntelliJ IDEA: File -> Export -> Project to ZIP file.