**Lab4**

|  |  |
| --- | --- |
| Due Date: | **Sep 27, 11:30 )** |

* **Submit your assignment using the following file format:**
* LabNumber \_RoomNumber\_StudentName\_ID.zip

(eg. 4\_328\_Hongkildong\_201620505.zip

* The zip file will contain **two types of** files, namely:

1. **report file** with file format **“Report\_Lab number**” (eg. report\_4) to write the answer of theory questions and the screen shots that display the output of your program.
2. Source code file that contains codes of classes to answer programming questions.

.

**I. Objectives**

1. Learning the Java method and **paramete**r passing.

2. Learning the concept of **static variables** and **static** methods.

3. Utilizing a **Math** library classes.

4. Utilizing a **Random** number generator.

The source code is attached to the file without taking a screenshot.

201520908

Yoo seong min

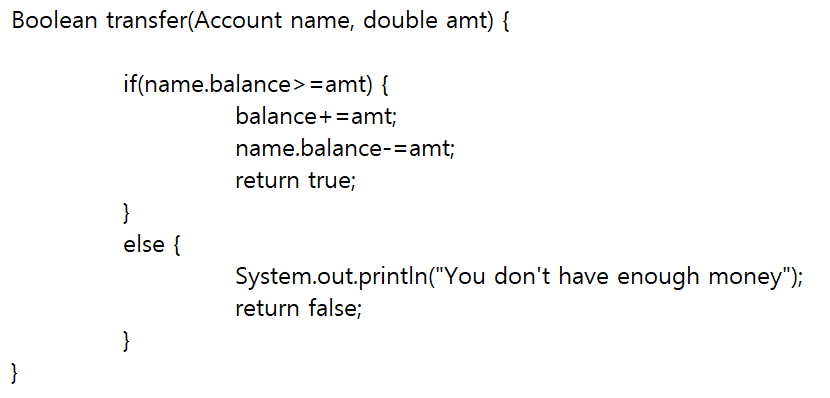
software

**II. Exercises (15 points)**

1. Answer the questions about **Fig 3.8** (**Account.java**).

A. Add **transfer** () method to the “**Account**” class as shown below**.** Implement this method that transfer an “**amt**” from **the given reference** to another object reference of the “**Account Class”.** If the money is transferred normally, the method return **true**. Otherwise, return **false** (**2** points)

|  |
| --- |
| **Example: Boolean transfer (**Account **b**, double **amt**)  **{**  **// Body of this method**  **}** |

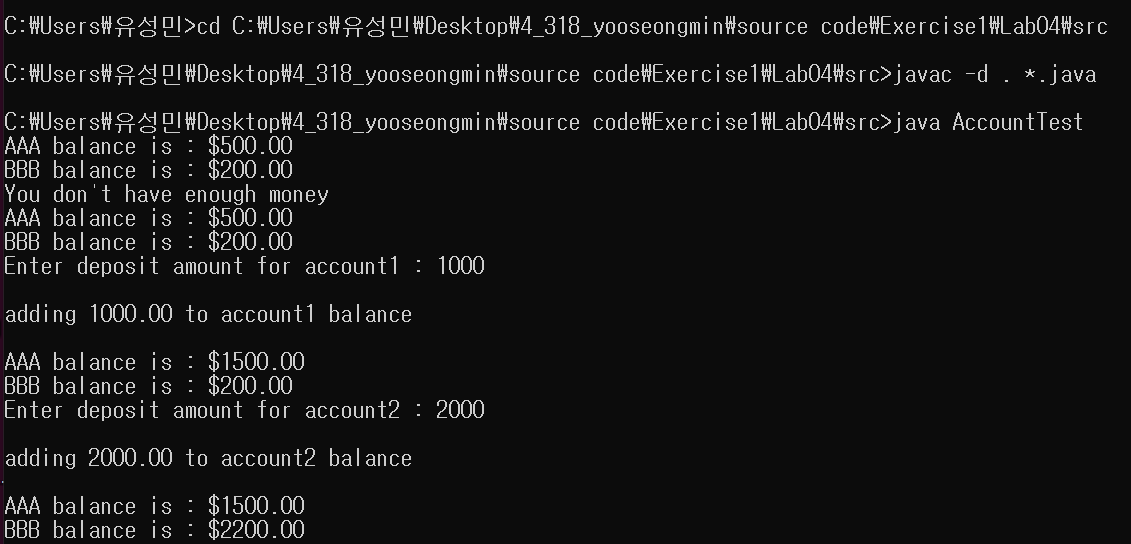
source code

B. Call the **transfer (**) method after making “**AccountTest** Class” such as **Fig 3.9**. Compare the Balance value of Account **before** calling the transfer method with the Balance value of Account and **after calling** the transfer method (including codes and captured screen) (**2** points)

|  |
| --- |
| Example: AccountTest(Fig 3.9)  Account a = new Account(“AAA”, 500);  Account b = new Account(“BBB”, 200);  a.transfer(b, 300); // function call |

텍스트이(가) 표시된 사진

자동 생성된 설명case1 . balance >amt

case2. Balance < amt

2. Modify the **Account class** (**Fig 3.8**) by adding **account number** and initial number (**no**) as its member variables.

* + Initialize the **account number** as indicated below.
  + Modify the constructor as indicated blew.
  + Add  **getNo() and getAccountNo() methods as follows**
  + **However, the program has logical error. Identify the error and give explanation**
  + **Correct the code by showing c**aptured screen(**3 points**)

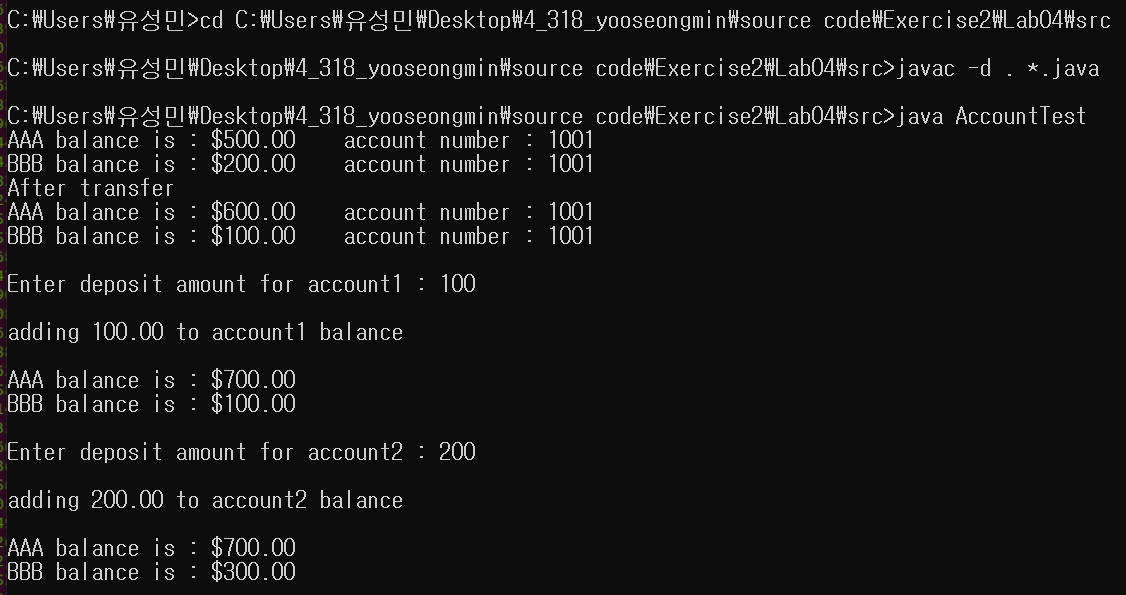
[Hint] Print out Account number after creating two or more Account objects using “static”

|  |
| --- |
| **private int accountNo; //** Account number  **private int no = 1000;** // Variable to creating account numbers sequentially such as //1001, 1002, 1003, 104,105,106,...  public Account(String **name**, double **balance**) **// constructor**  {  this.name = name;  if (balance > 0.0)  {  this.balance = balance;  }  **acountNo = getNo(); // Initialize account number**  **}**  **private int getNo()**  **{**  return ++no;  **} // added method to create account number.**  **public int getAccountNo()**  **{**  return accountNo;  **} // added method to know account number** |

Not use static keyword => private into no = 1000; not attached this code

Account numer doesn’t change

The no variable is an instance variable, which has a unique value for each instance. Therefore, the variable of Account1 and the variable of Account2 are set to 1000 by explicit initialization, and the getNo method is executed to change the value to 1001.

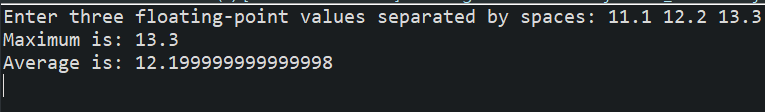


Use static keyword => private static int no = 1000; attached this code

The static keyword adds a class variable that is shared by all instances.

텍스트이(가) 표시된 사진

자동 생성된 설명Therefore, 1000 is entered into the no variable of Account1 and becomes 1001 after the getNo method is executed. This is a value shared by all instances, so the no variable for Account2 will be 1001. This will also change the value to 1002 through the getNo method.

3. In **Fig.6.3, t**he class has two methods, namely main () and maximum (). Modify this program by adding a third method called **average ()** that computes the average of three numbers. Modify also the body of main () method so that it will display the average value on the screen.

4. Answer the following questions about Fig **6.3** **(MaximumFinder.java).**

1. **Re-write** the program by removing the **“static”** modifier of the **maximum** () method, and then compile the source code. Do you get the same result? Explain your reason (**1 point**)

The same result did not come out.

This program produces an error because you removed the static modifier. This behavior changes from a static method to an instance method. To use the maximum function, you must instantiate a new object.

1. **Re-write** the program by implementing the body of the maximum () method using the **max**() method of the **Math class** from the java.lang package. You have to get similar result (**1pt**)

The modified code is attached to the source code.

5. Answer the questions using the **MethodOverload**.**java (Fig 6.10)** codes.

검은색이(가) 표시된 사진

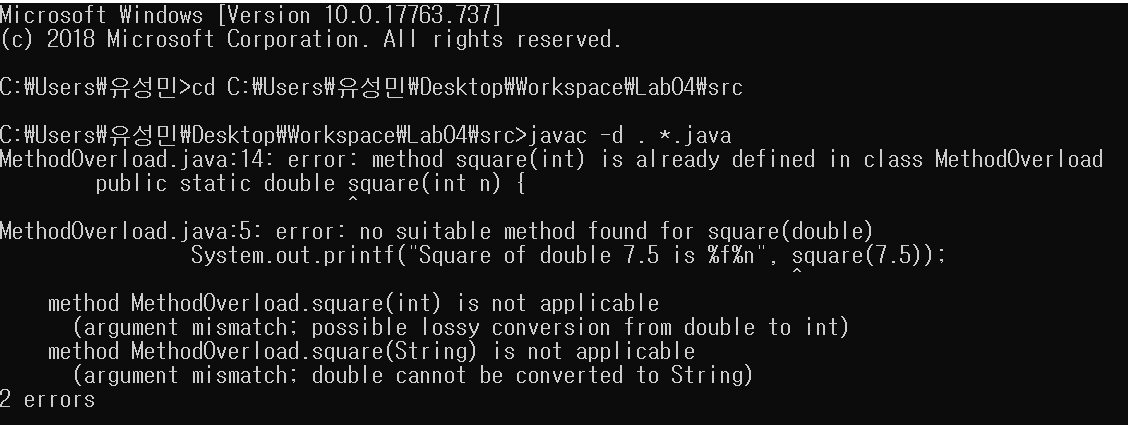
자동 생성된 설명A. Modify the code by adding the following **square(String stringValue)**  method as a form of method overloading. This method returns a **new string** by concatenating the given string. Example, if the string “**hello”** is an argument, then it returns the string “h**ellohello**”. Print out the result after calling the **square(String stringValue)**  method from main method (including codes and captured screen). (**2 points**)

|  |
| --- |
| System.out.printf("Square of String 'hello' is %s%n", square("hello")); **// call from main() method**  public **static** **String** square(String stringValue) **// header of the method**  {  .....  } |

|  |
| --- |
| public static **double** square(int **n**)  {  ....  } |

B. Modify the program by adding the following **square (int n)** method as method overloading. This method takes **an int value** as an argument and return **double type** value as a square of an int value. The program will **have an error**. Explain what problem occur? (Including both the codes and explanation). (2 points)

As you can see the name of method is same(square). But two square methods have different parameter types and different return type at first (int type parameter and double type parameter). But after modifying the codes, two method’s parameter type are same type(int). So when the method square is called, compiler can’t distinguish which method you want to use.



6. Answer the following question using **Fig 6.7** (**RollDie.java**) codes.

Re-write the program to store the “**frequency**” using array data structure (including codes and captured screen). (**2 points**).

