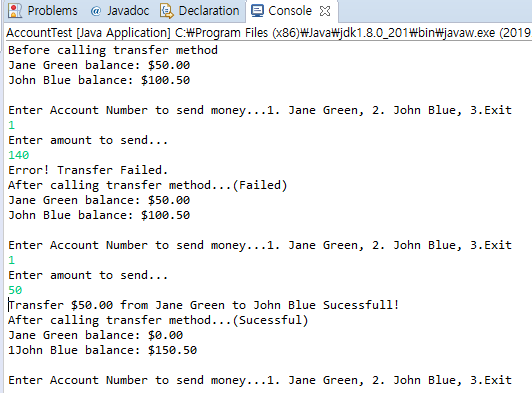
**II. Exercises (15 points)**

1. Answer the questions about **Fig 3.8** (**Account.java**).
2. 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**  **}** |

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



* Screenshot 첨부

1. 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** |

1. 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.
2. Answer the following questions about Fig **6.3** **(MaximumFinder.java).**

**A**. **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**)

**B**. **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**).

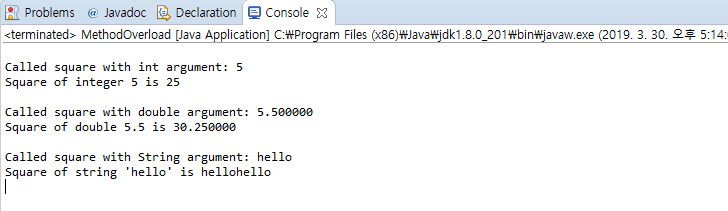
Answer

A : 컴파일을 할 수 없다. 그 이유는 non-static 메소드인 maximum 메소드를 메인에서는 사용할 수 없기 때문이다. 해결방안은 static 조건을 붙여줘야한다.

B : exercise05에 첨부하였다.

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

A. 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**)



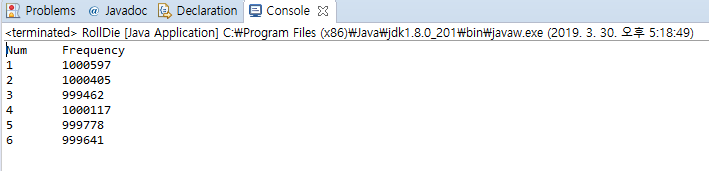
* Exercise06에 소스를 첨부하였따.

|  |
| --- |
| 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**  {  .....  } |

1. 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)

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

1. 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**).



->소스는 exercise07에 첨부하였다