**Notes: First you prepare how to write a code for syntax and write a code**

**Java: (day one)**

* **Steps to install java**

Step1: Download JDK

Step2: Install JDK and JRE

Step3: Include JDK’s bin directory in the path

* **Steps to install eclipse**

1. To use Eclipse for Java programming, you need to first install Java Development Kit.
2. Download Eclipse from [http://www.eclipse.org/downloads](http://www.eclipse.org/downloads/).
3. To install Eclipse, simply unzip the downloaded file into a directory of your choice

**Launch Eclipse**

1. Start Eclipse by running "eclipse.exe" in the Eclipse installed directory.
2. Choose an appropriate directory for your *workspace*.
3. If the "Welcome" screen shows up, close it by clicking the "close" button.

##### Create a new Java Project

1. Choose "File" menu ⇒ "New" ⇒ "Java project".
2. The "New Java Project" dialog pops up.
   1. In the "Project name" field, enter "First Project".
   2. Check "Use default location".
   3. In the "JRE" box, select Use default JRE But check the JDK version, you should be using JDK and above.
   4. Click "Finish".

* **Steps to create workspace**

You can even switch workspaces via File→Switch workspace.

File→Import→Existing project in to workspace→select project.

* **Steps to create project**

File -> Project

We call Project is program

* **How to create .java file/class**

Project -> new class and give extension .java

Class Employee {

}

* **how to create packages and what is best way to give name**

From solution explorer, select project, right click and select package

Ex: companyname.projectname.foldername (this is common naming standard)

* **what is main method will do?**

Main method is starting point of program

* **What is variable?**

It will store the value in memory

To create variable we specify

Variablename datatype;

* **what is data type and different data types`**

It will represents what type of data

Int

Double

Float

* **creating property/data members :**

we create properties at class level

int salary

String firstname

creating method with void :

we write methods in void

* **creating method with void and parameter**

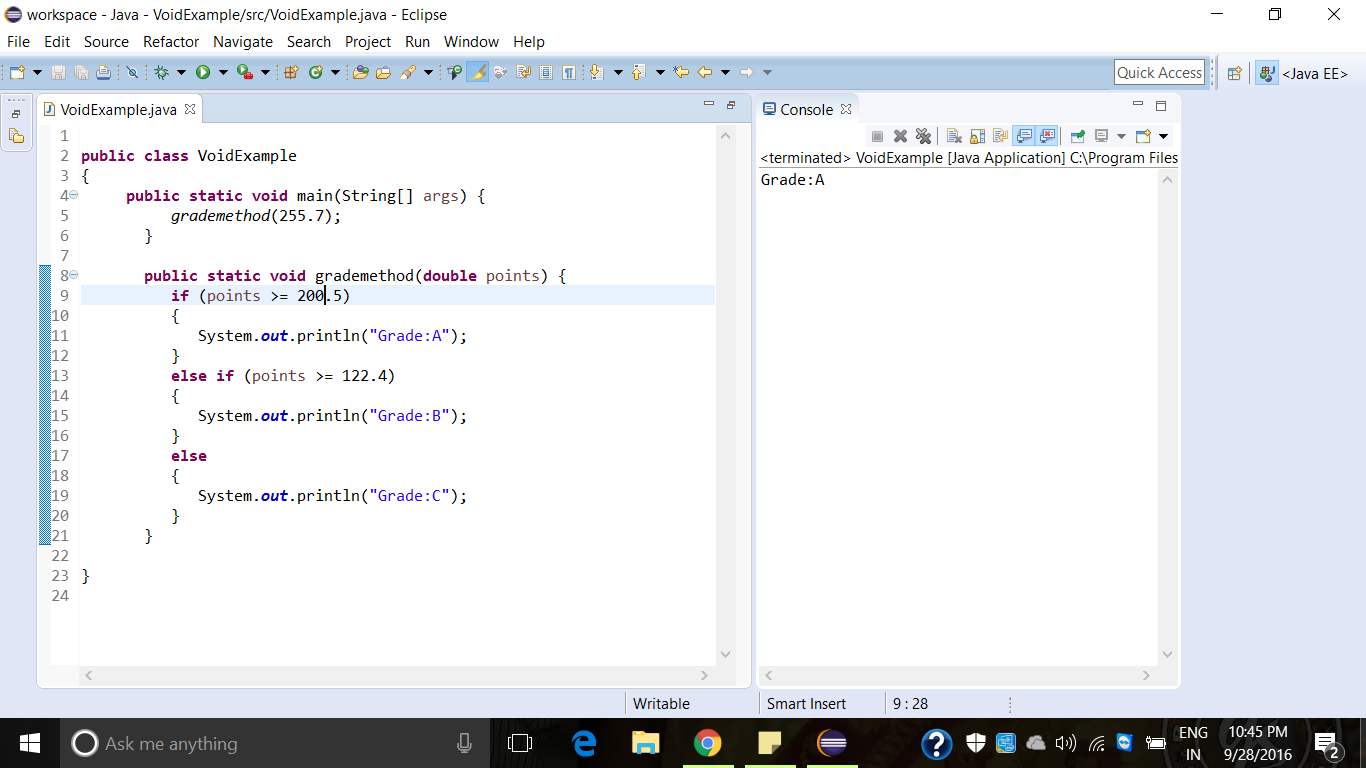
For creating a method, it should be called.

When a program invokes a method, the program control gets transferred to the called method.

**VOID:**

The void keyword allows us to create methods which do not return a value.

EX:

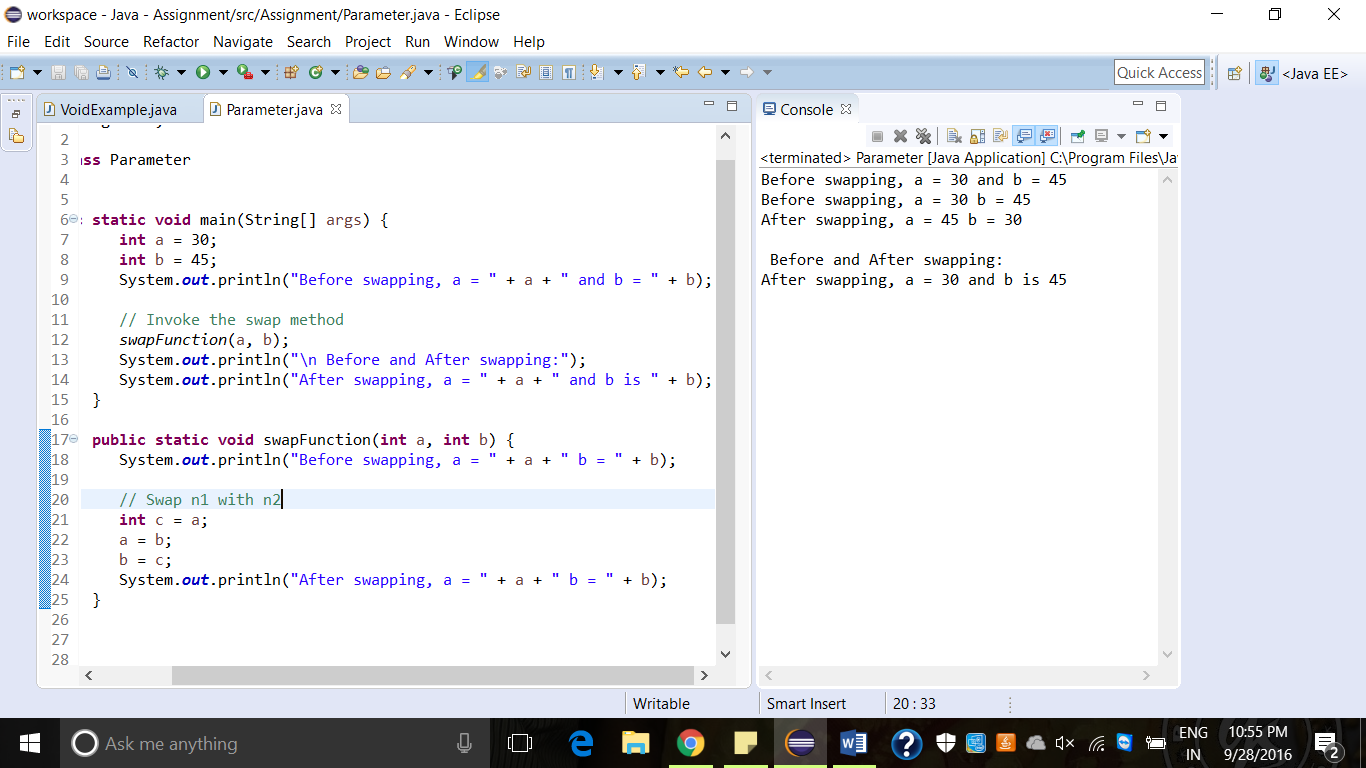


**PARAMETER:**

Parameters can be passed by value or by reference.

Passing parameter by value means calling a method with a parameter.

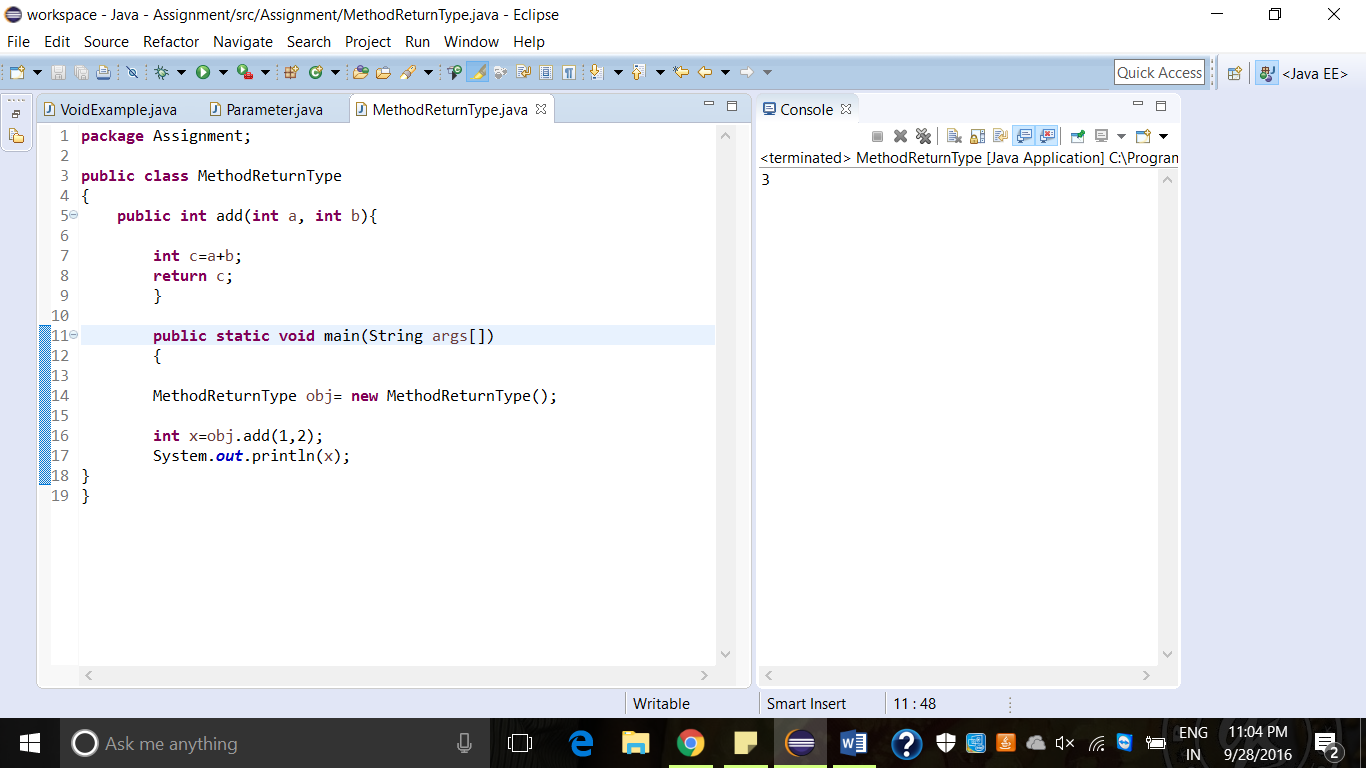
EX:



* **creating method with return data type**

A method returns to the code that invoked it when it

* Completes all the statements in the method
* Reaches a return statement, or
* Throws an exception



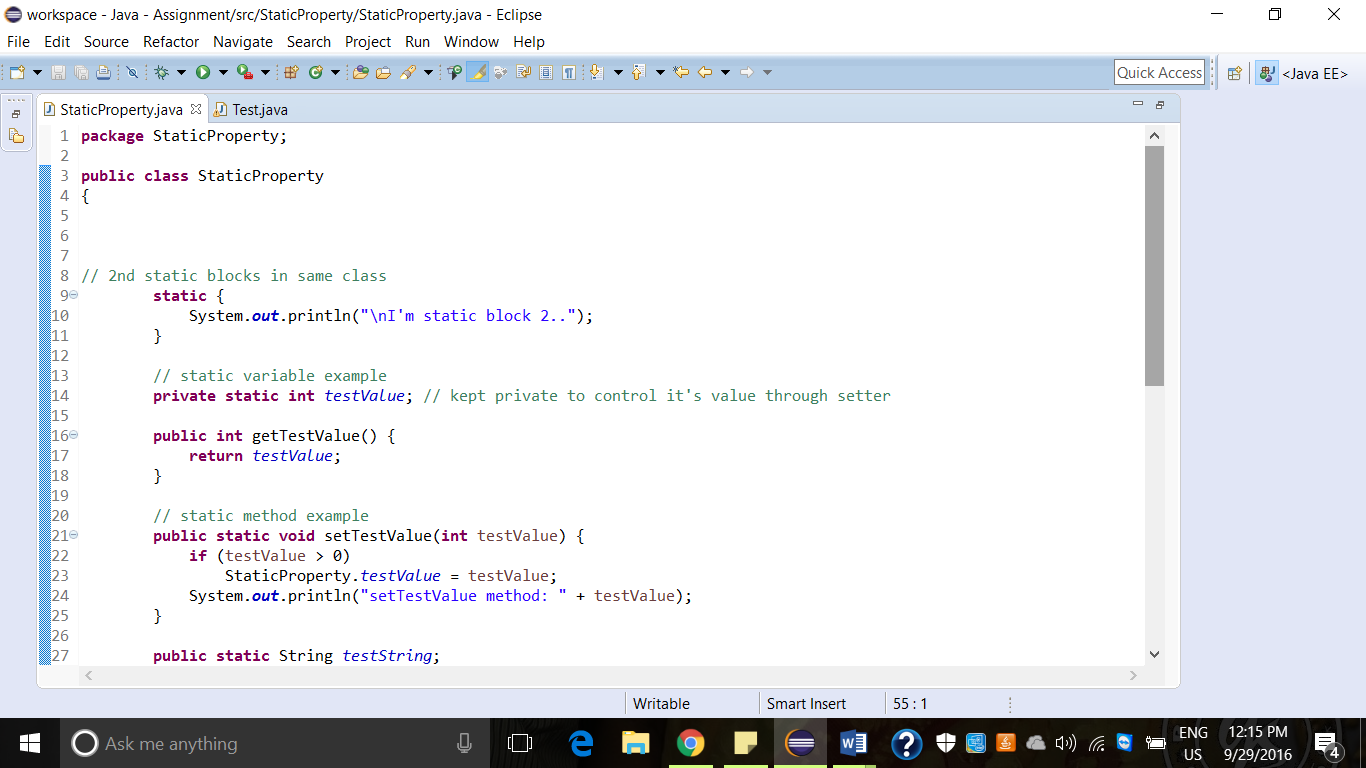
* **creating method with return data type and parameter**
* **creating variable**

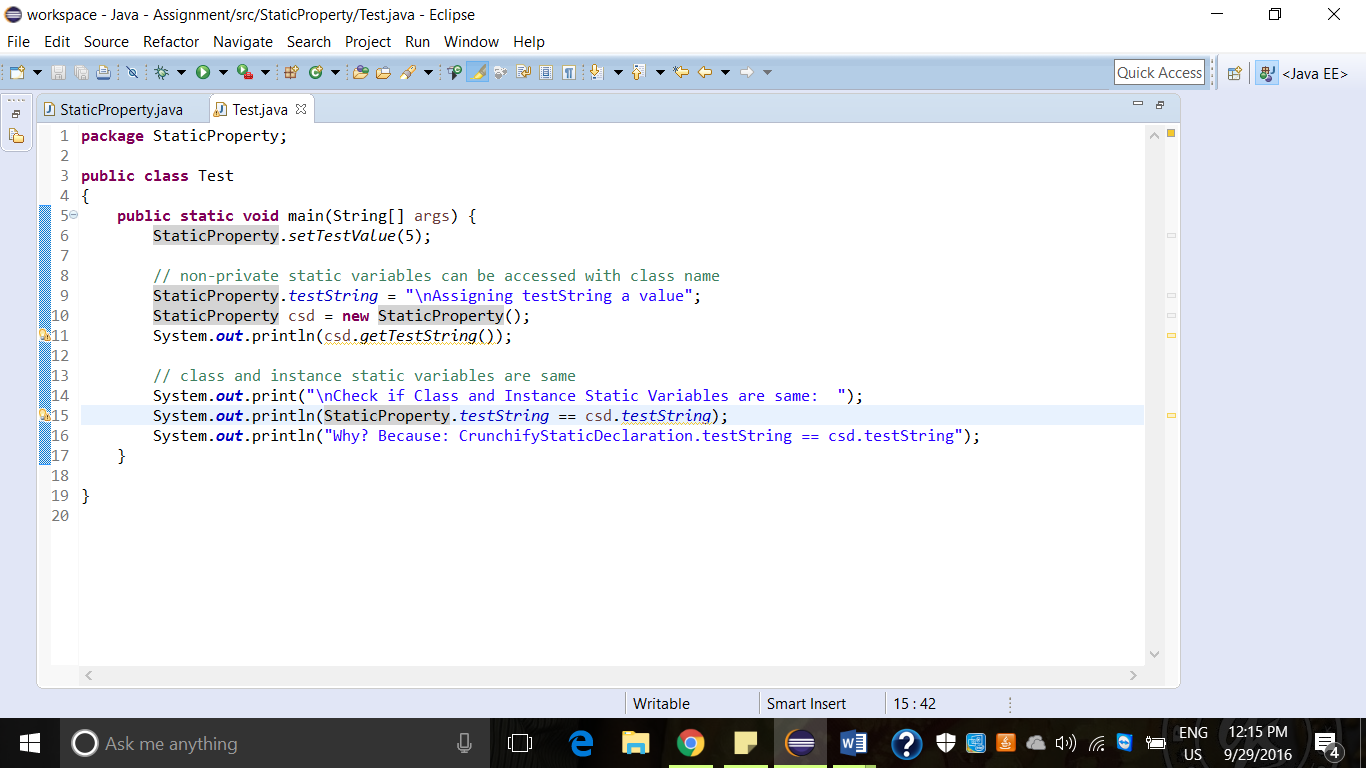
Variables are most important concepts. They store information such as numbers, words, sentences and more. This will give us an introduction into using a variable.

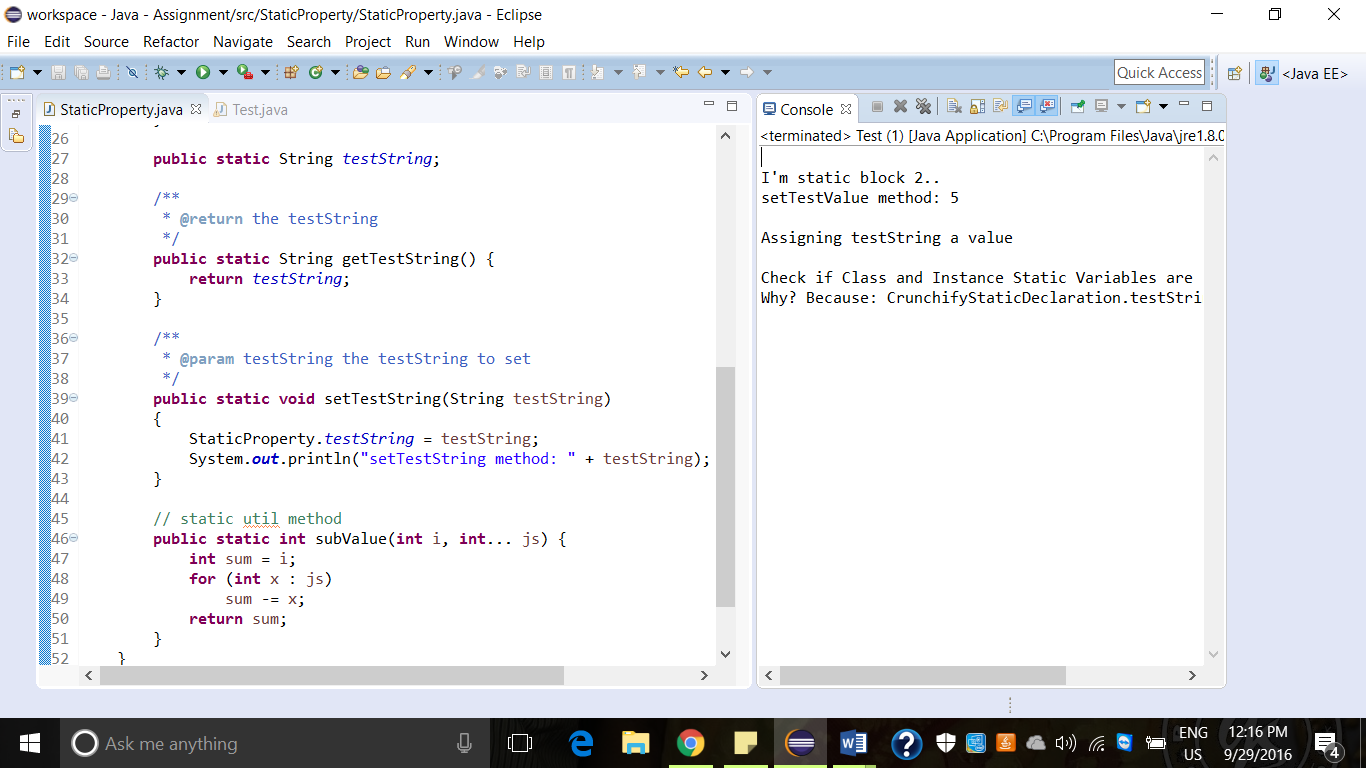


creating static property All instances shared the value

<http://crunchify.com/java-static-methods-variables-static-block-and-class-with-example/>







* **creating static method**

<http://crunchify.com/java-static-methods-variables-static-block-and-class-with-example/>

It is a method which belongs to the class and not the object

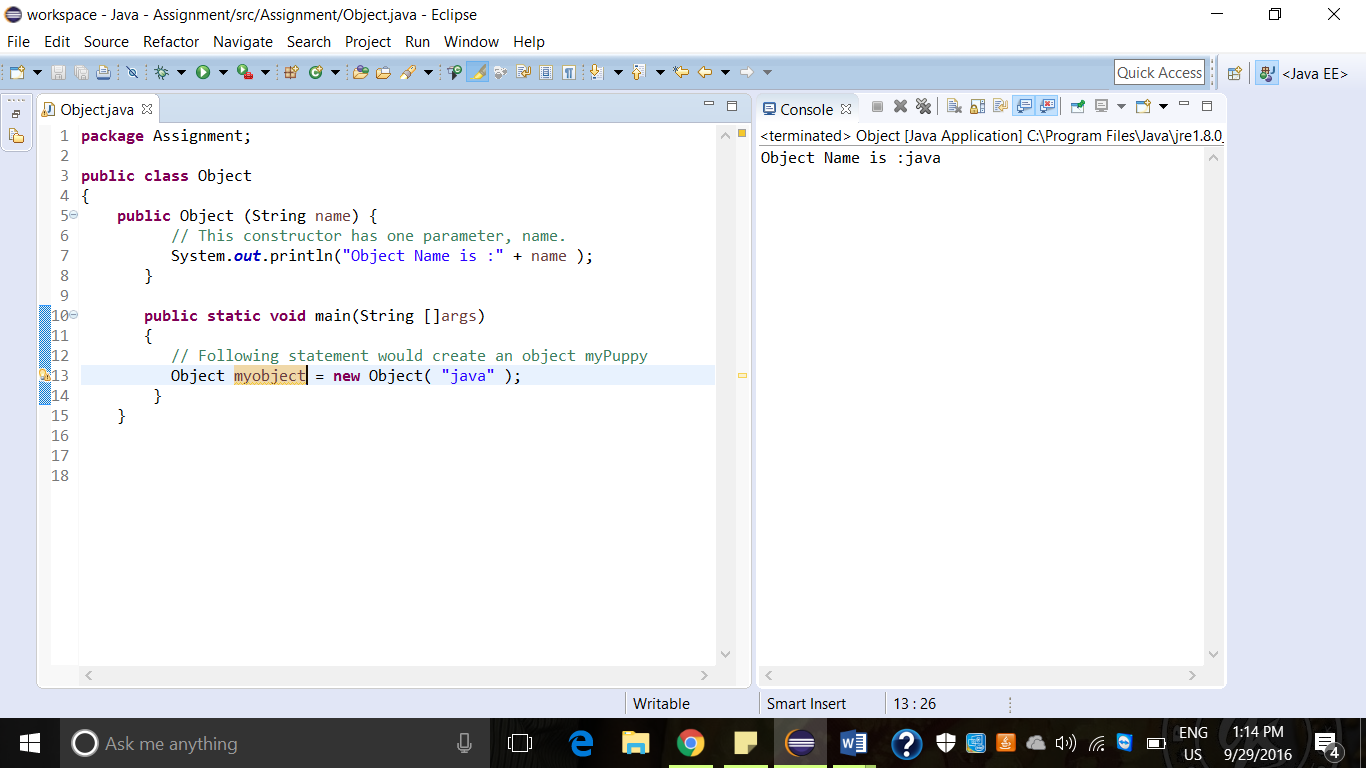
A static method can access only static data. It cannot access non-static data.

This method can call only other static methods and cannot call a non-static method.

Syntax: <class-name>. <method-name>

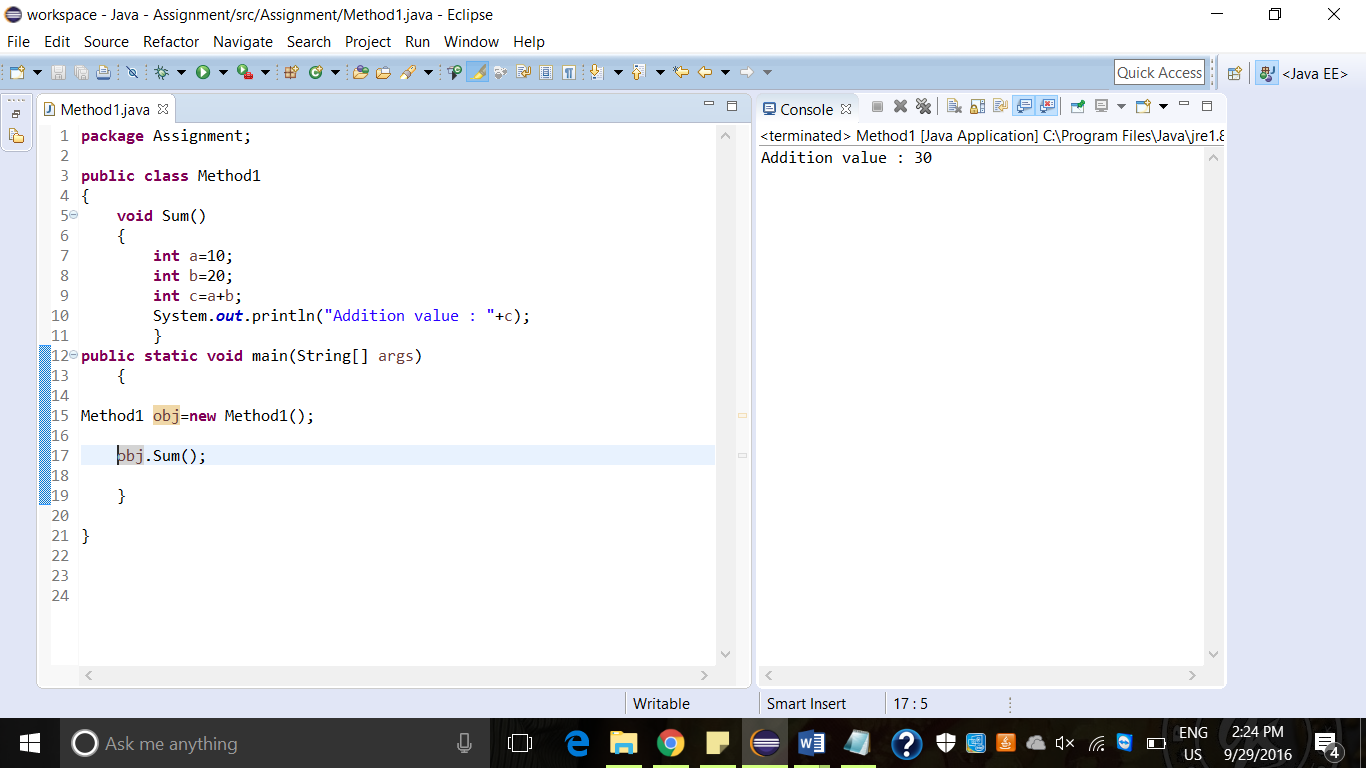
* **creating object**

As we already know class provides a blueprint for objects, you can create an object from a class. Program creates an object and assigns it to a variable.

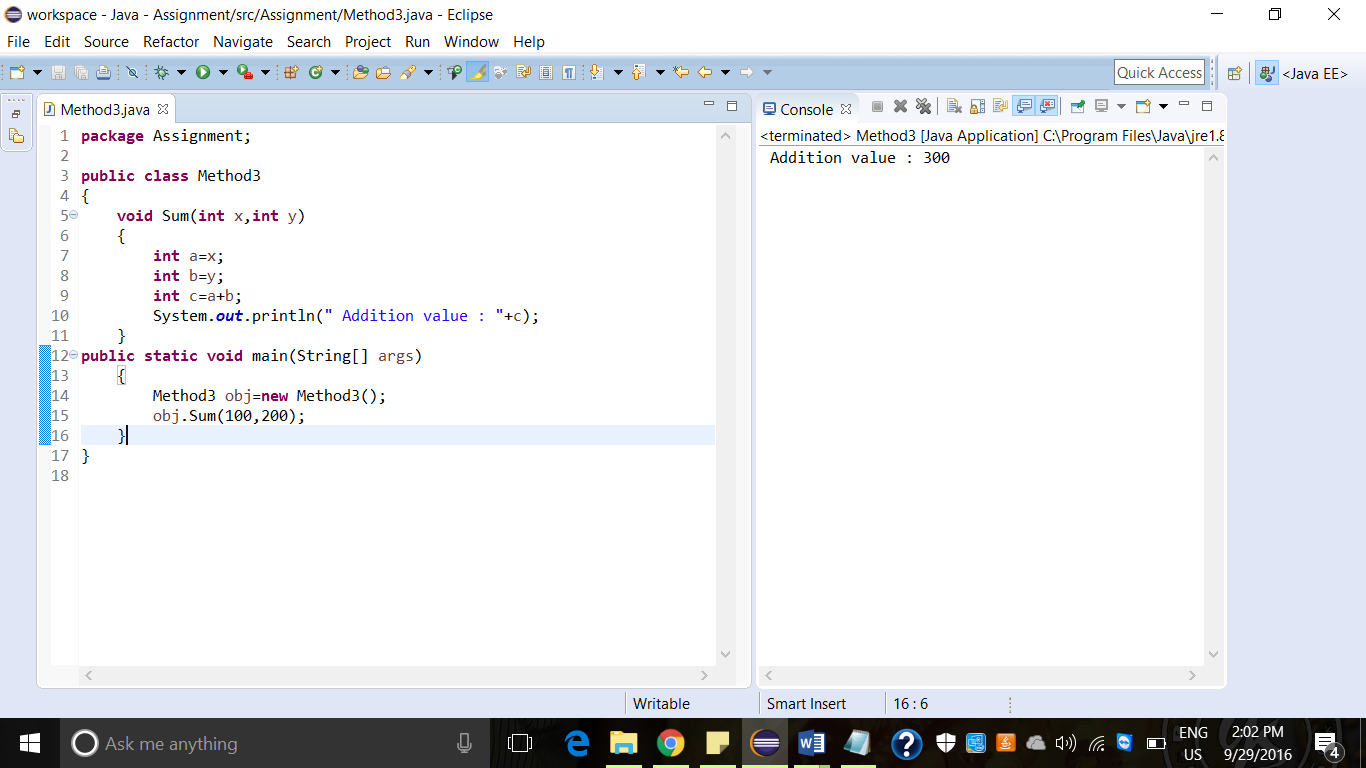


* **calling method with no return**

In this function containing to define the input in function and also define in function.

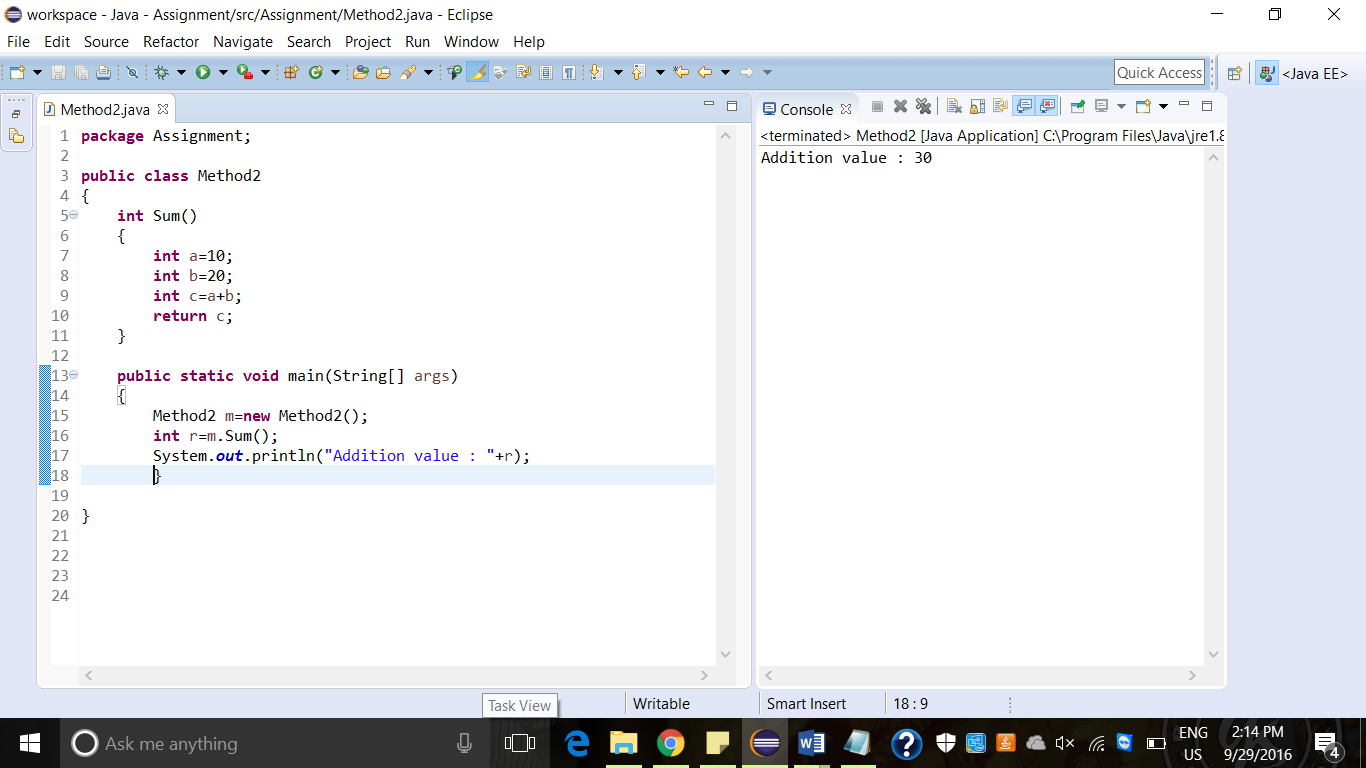


* **calling method with no return and parameter**

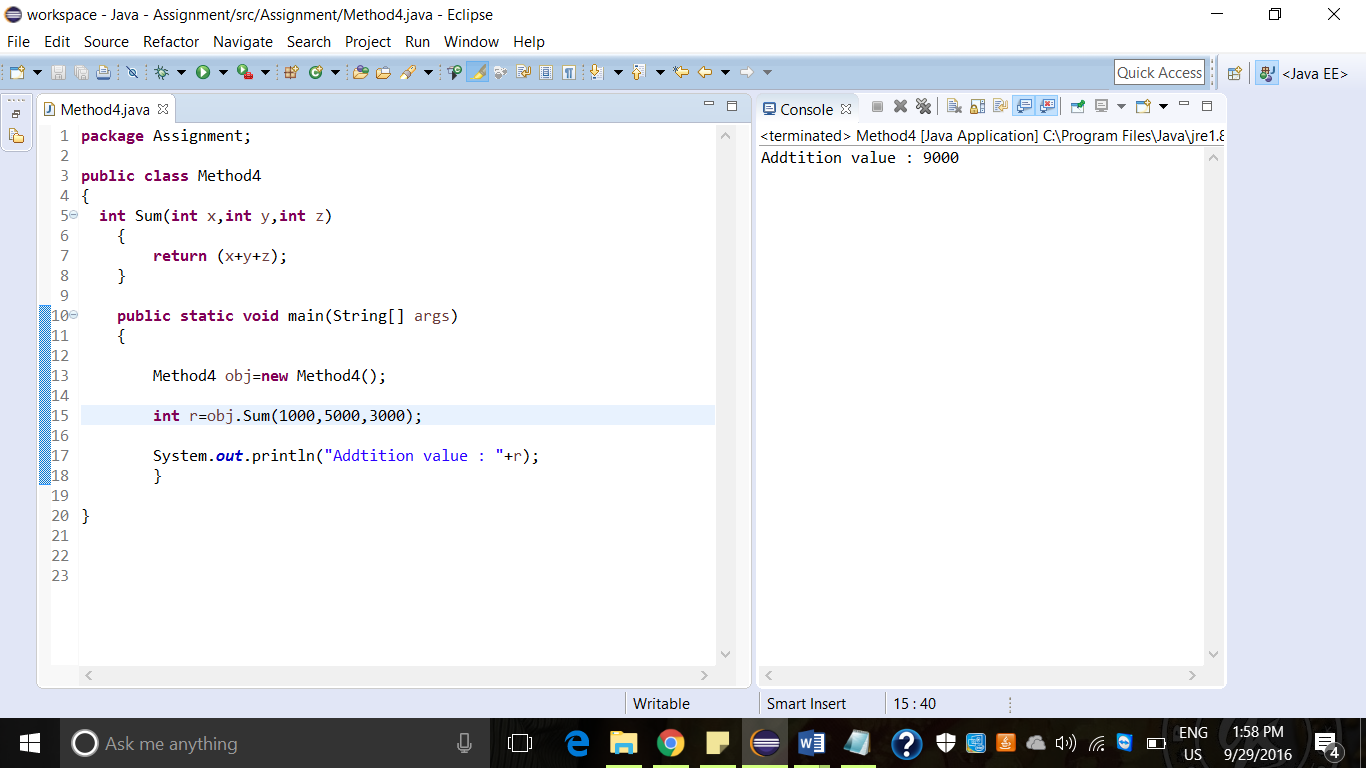
In this function we send the values from function calling to function build and we are accessing the output from function build to function calling.

* **calling method with return and no parameter**

In this function containing the inputs in function but output we return to function build to function calling.

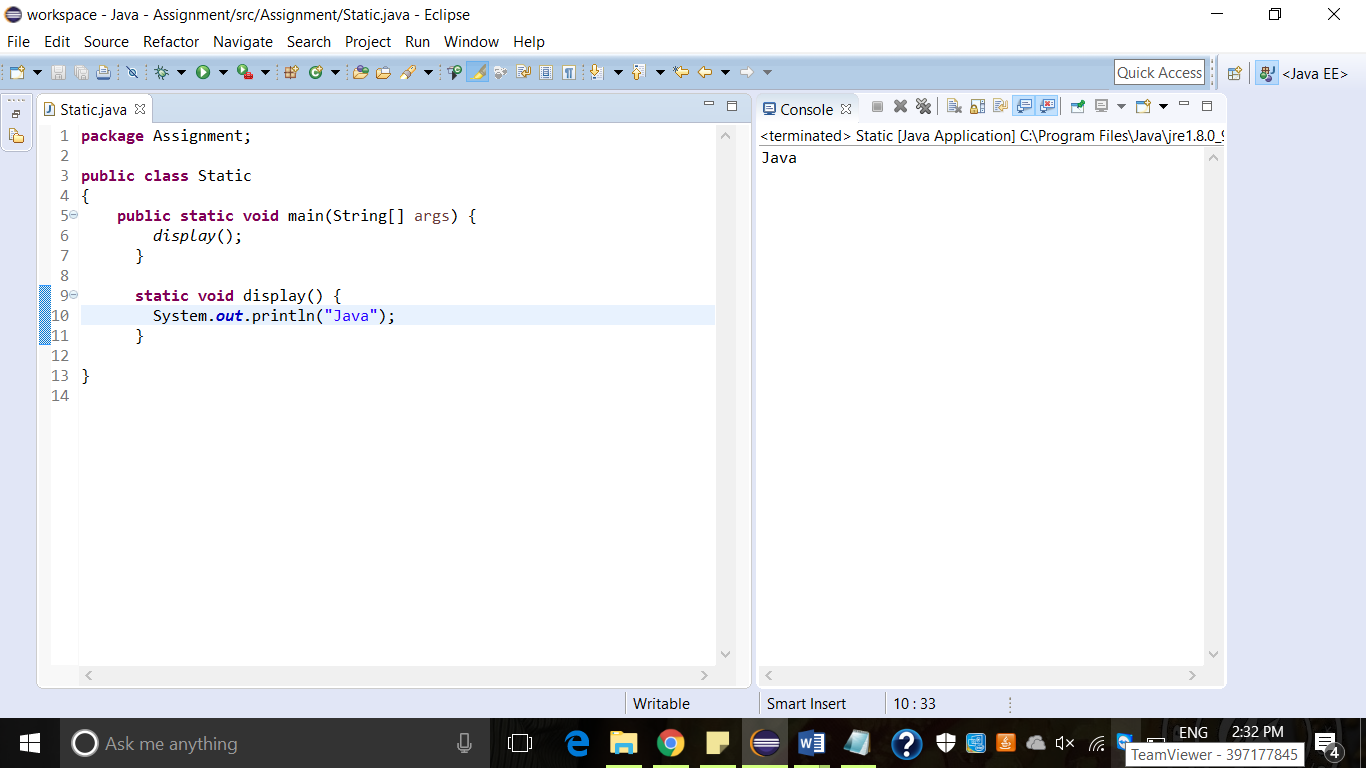
EX:

* **calling method with return and parameter**



* **calling static method**

Static methods in Java can be called without creating an object of class.



using static property

**Java: Day2**

create classes under multiple packages

calling classes under different packages

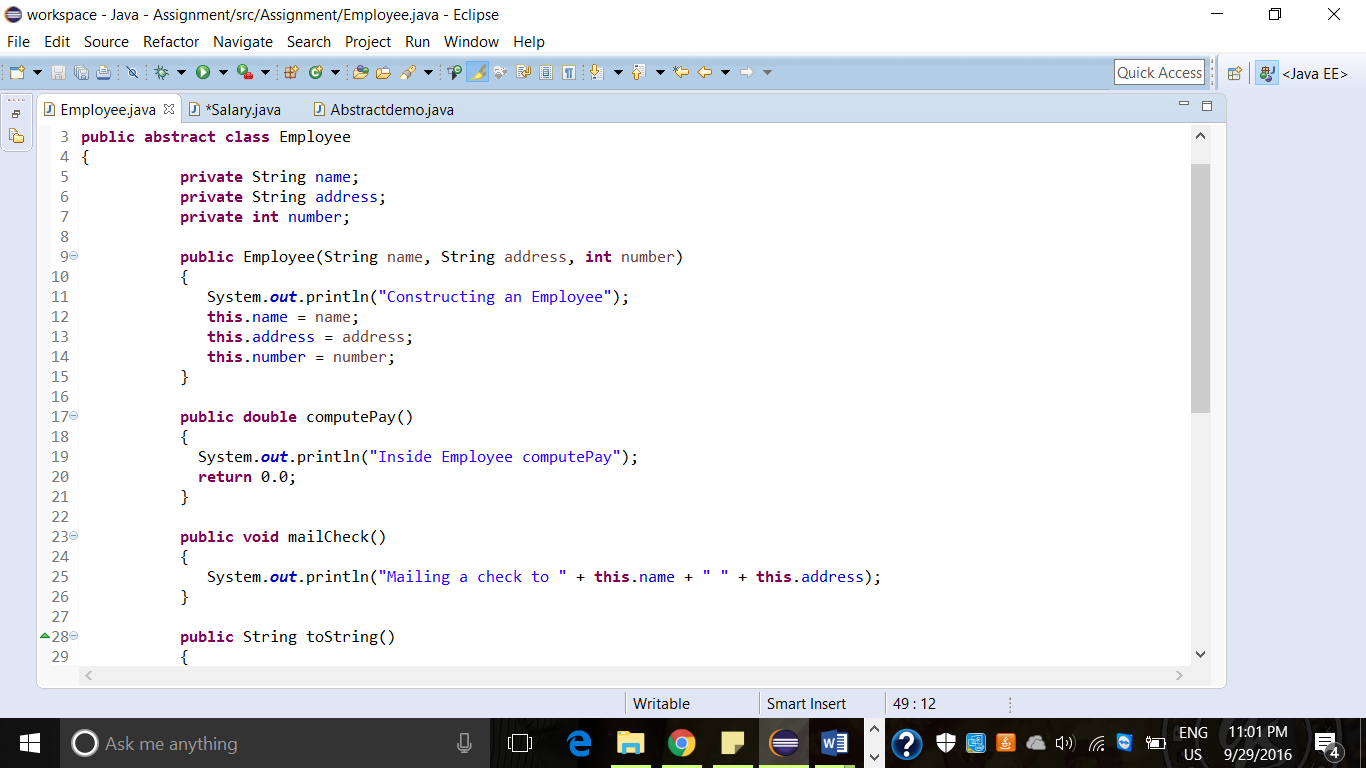
* **write code to handle exceptions with try/catch/finally**

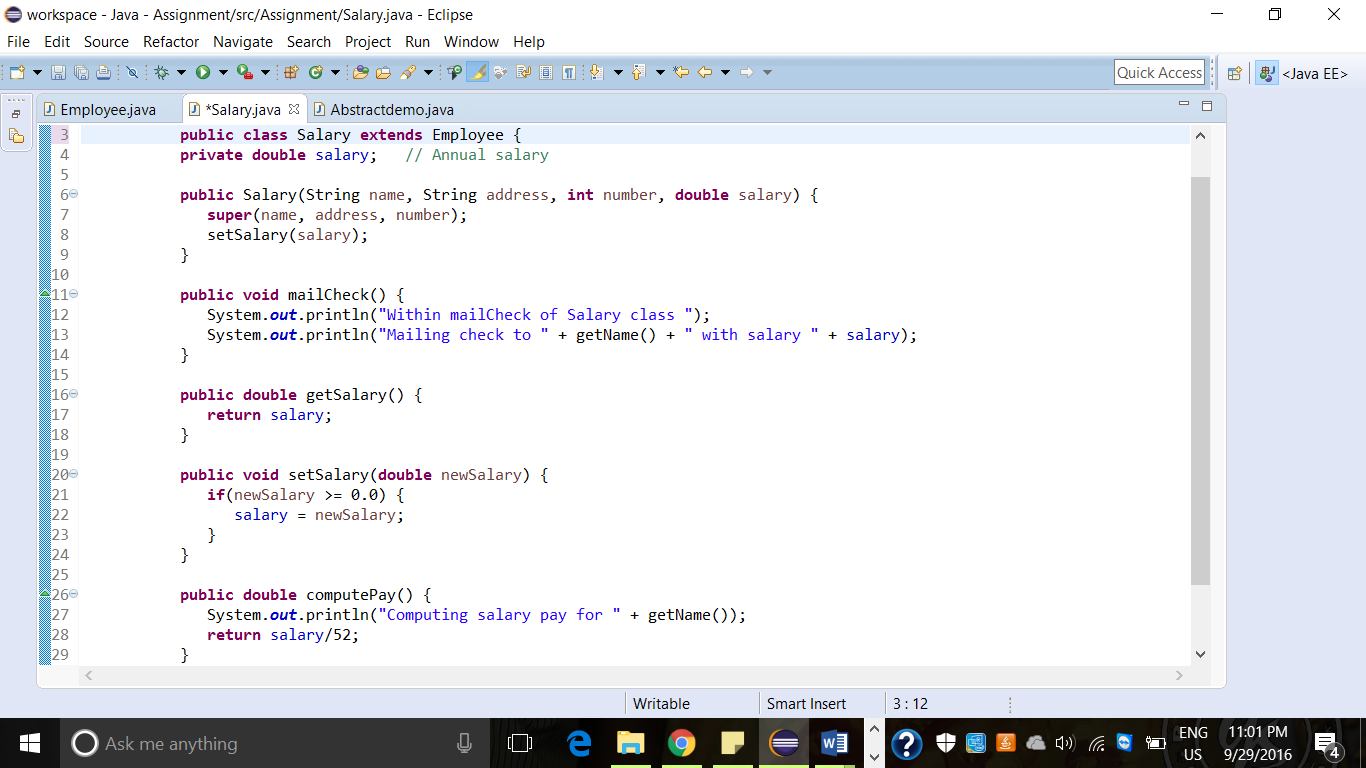


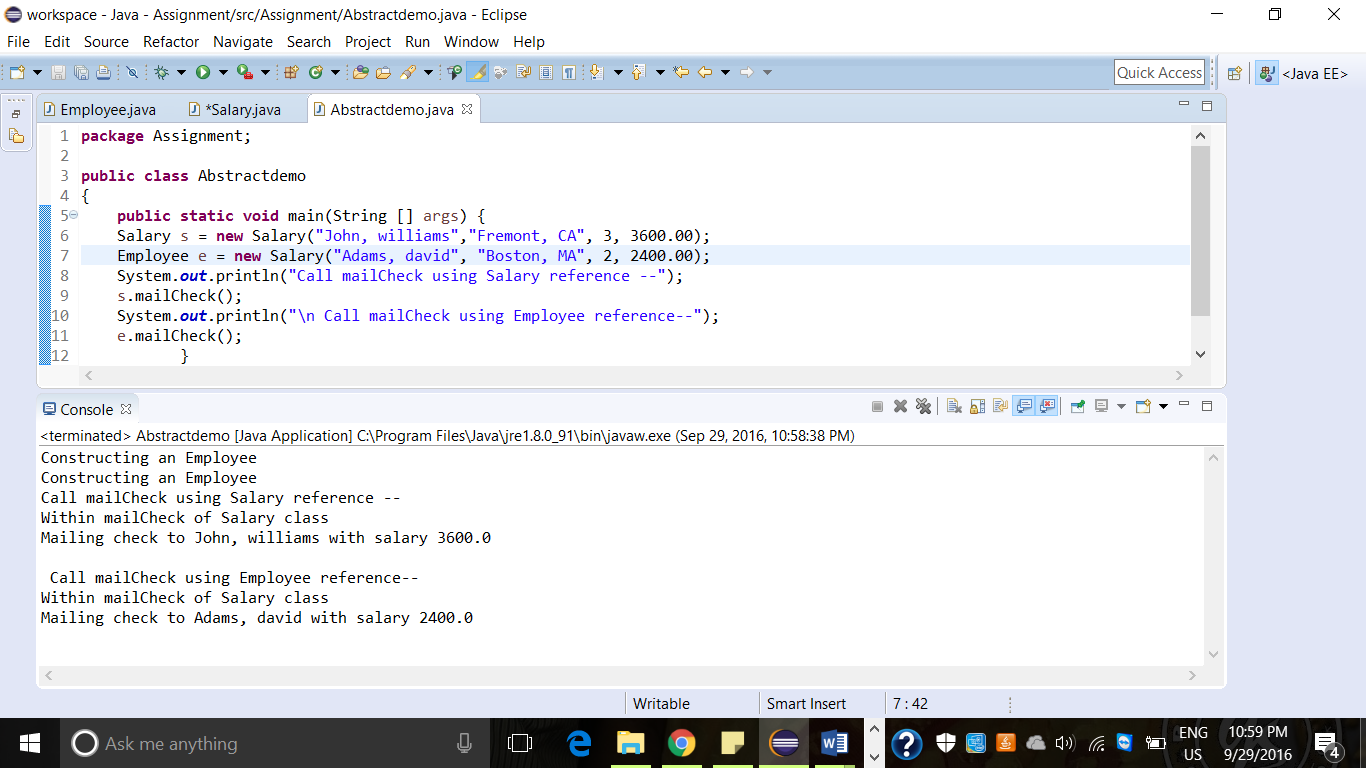
* **what is final keyword**

The final keyword in java is used to restrict the user. The java final keyword can be used in many contexts. Final can be: variable, method, class.

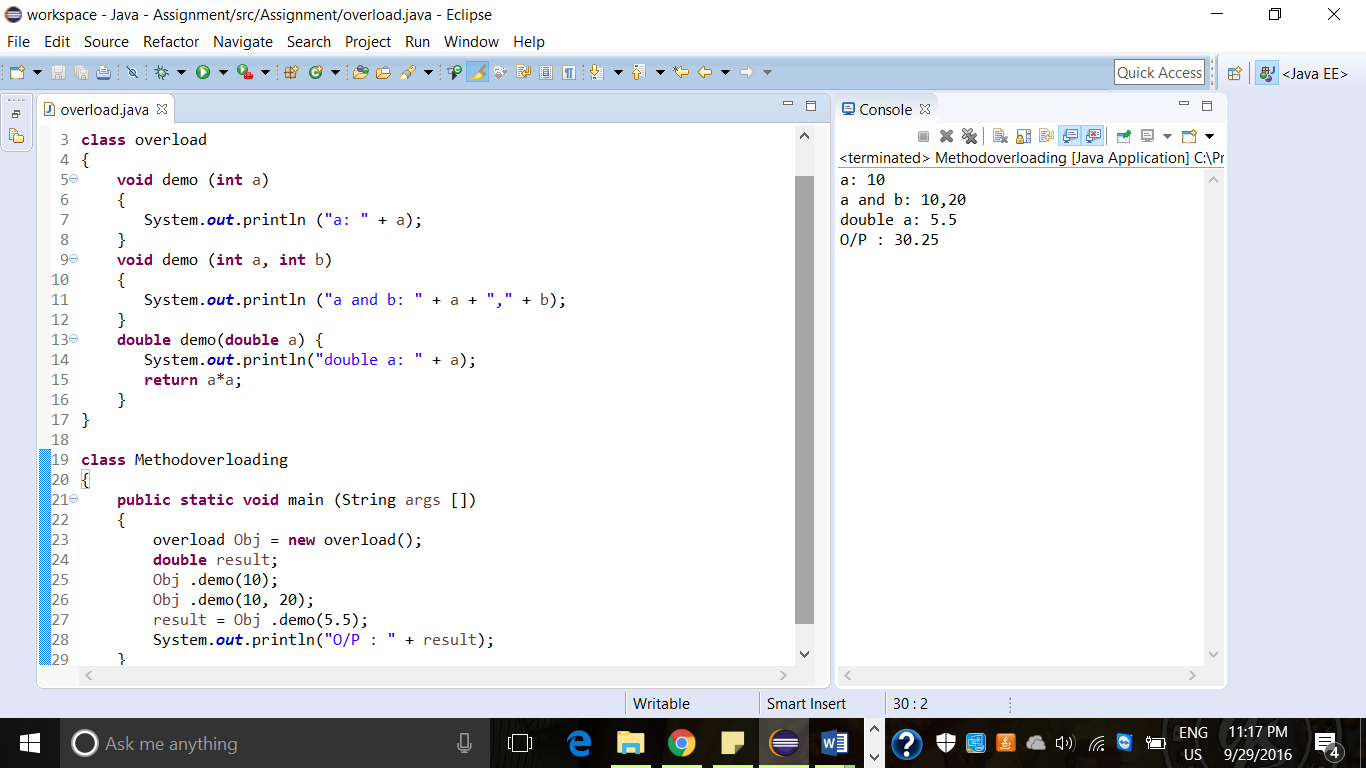
* **write code for creating abstract class**



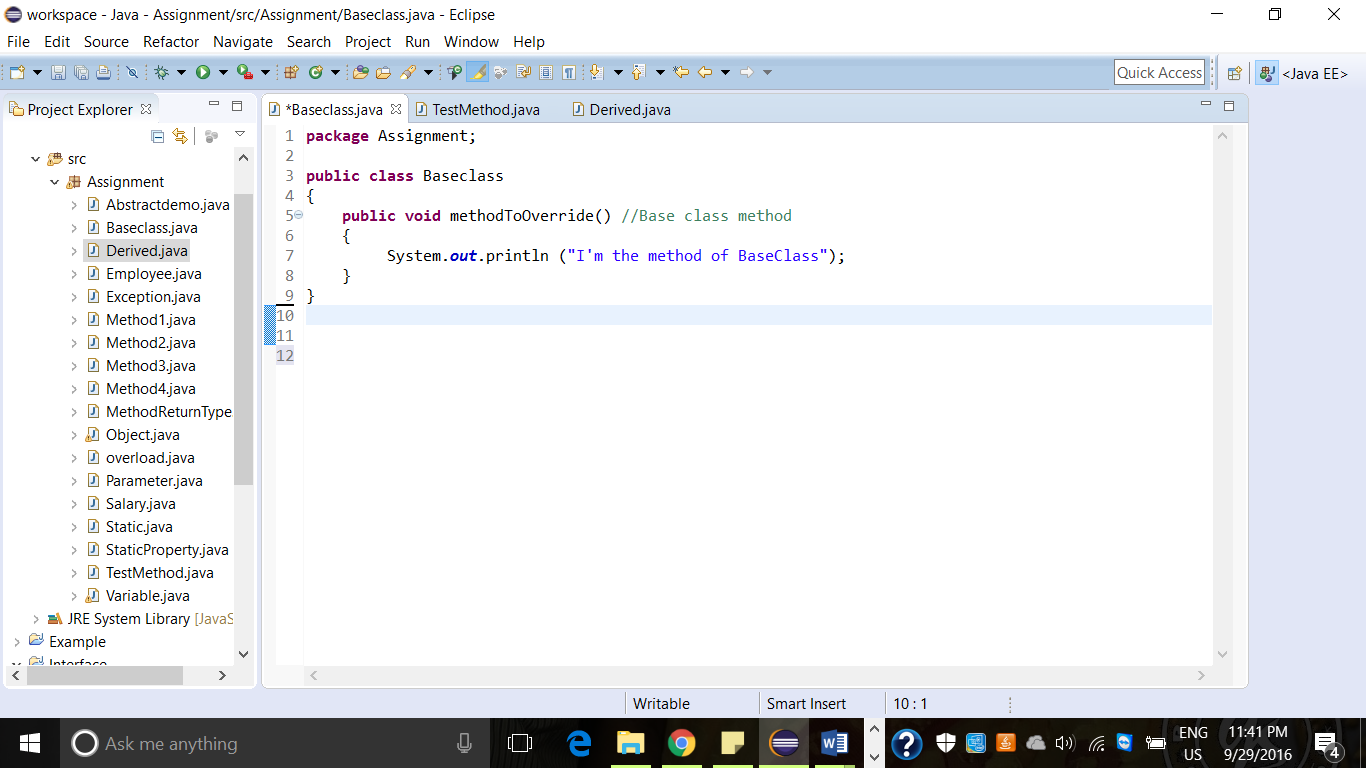


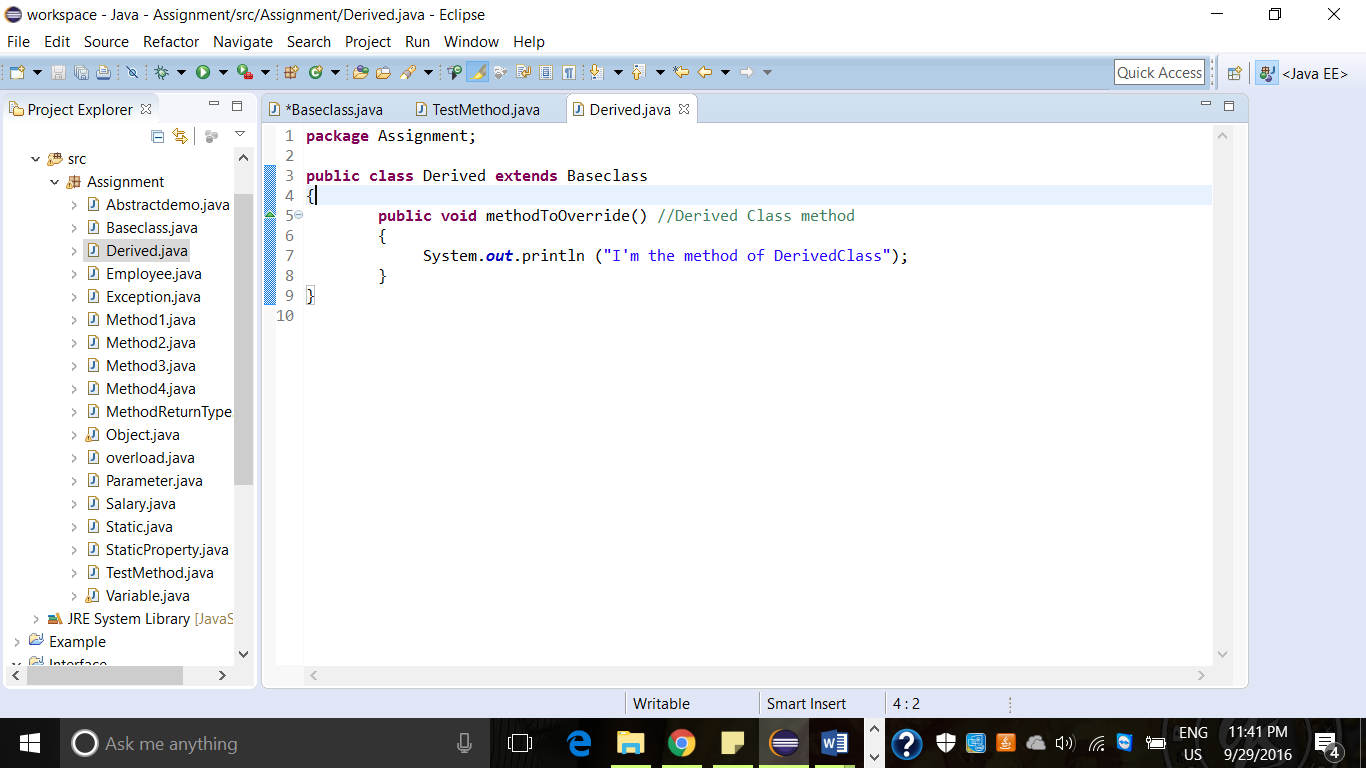


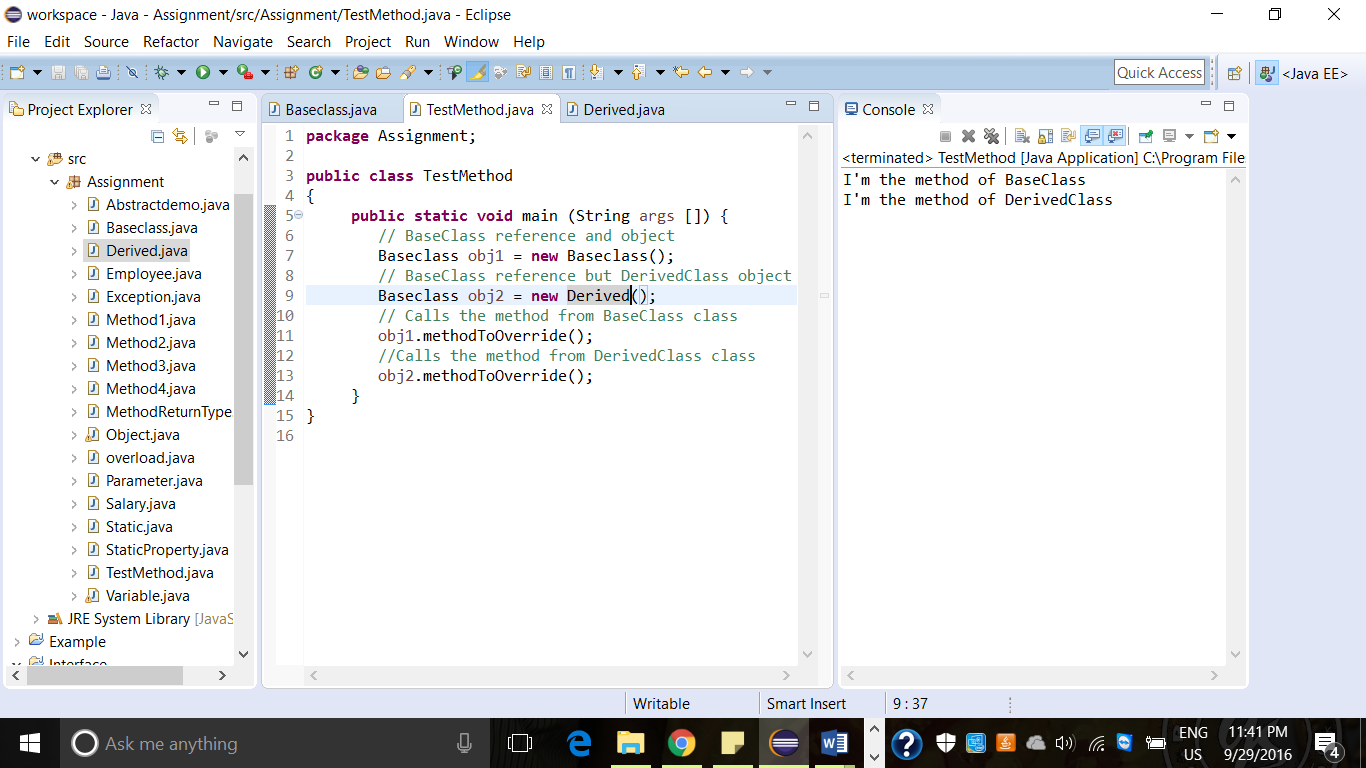
* **implement method overloading**



* **implement method overriding**



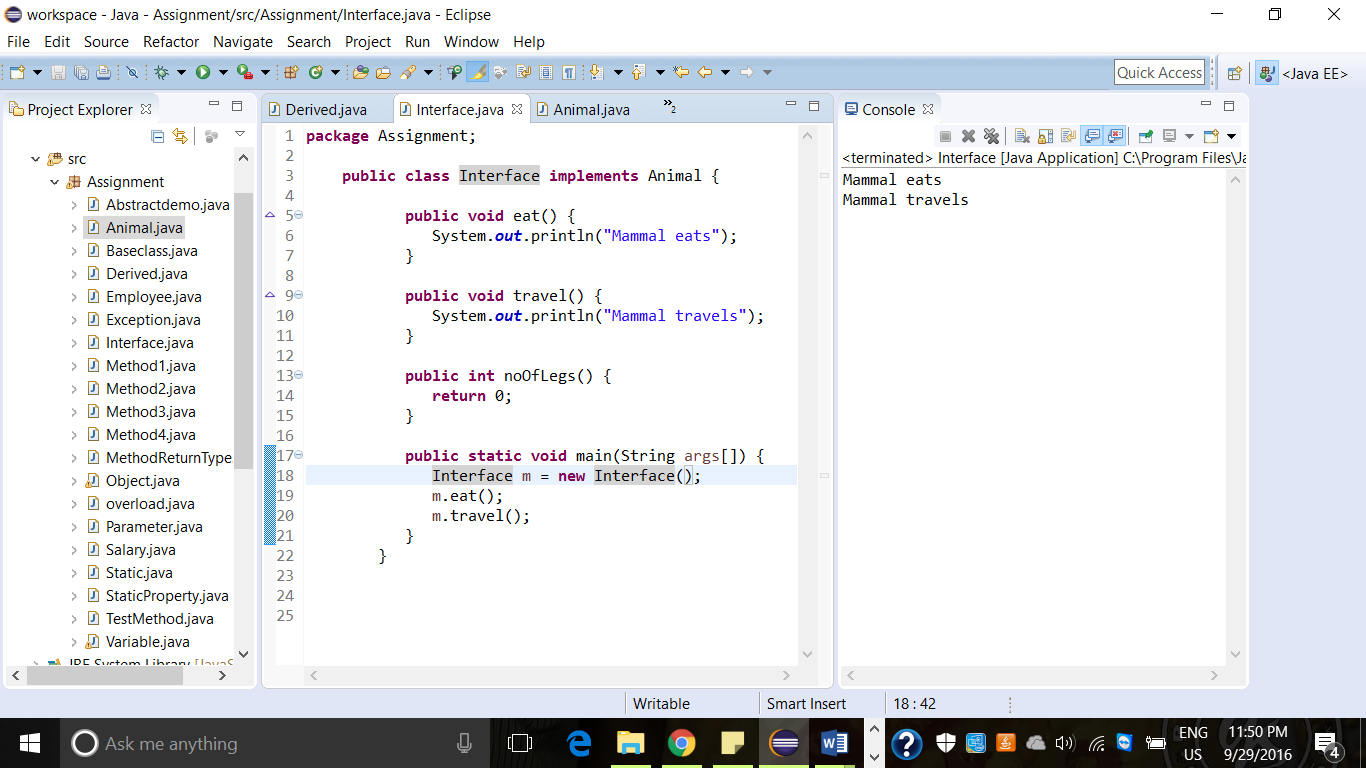


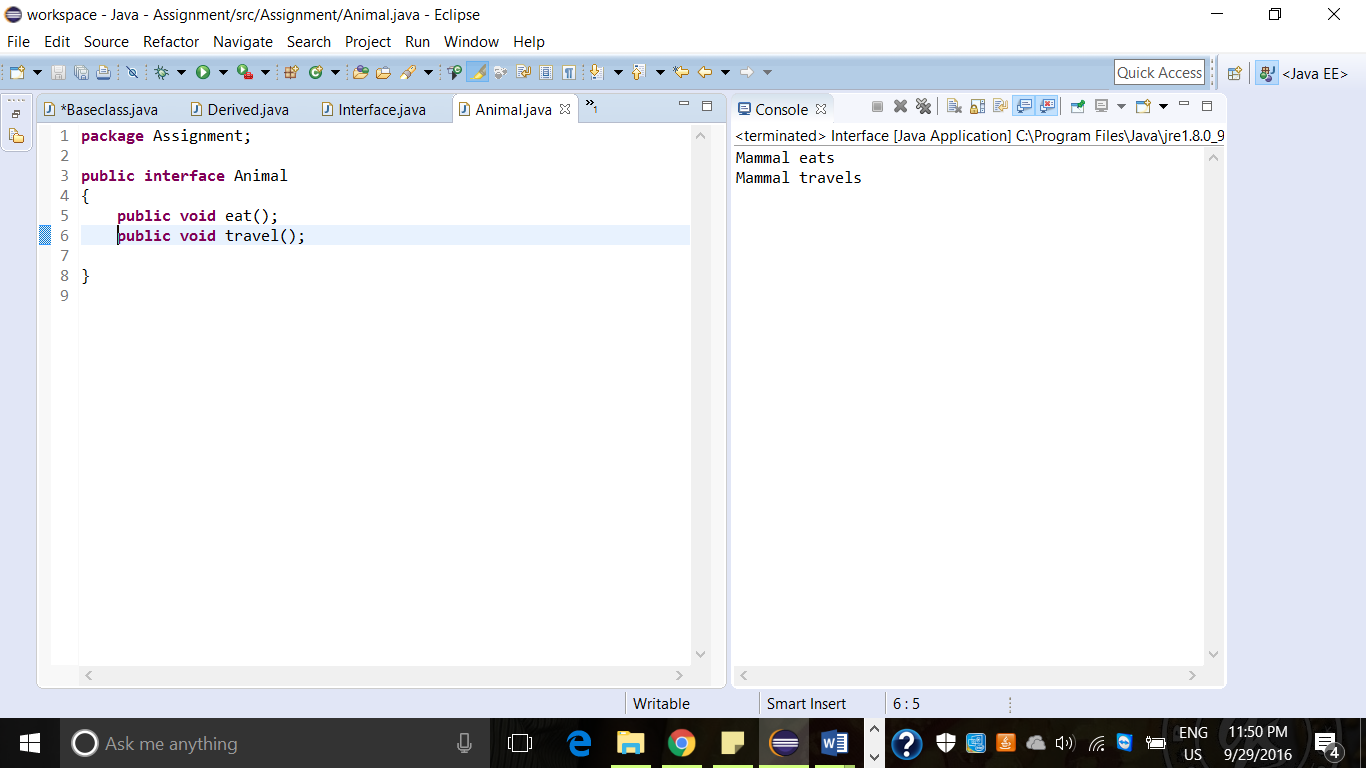


* **implementing polymorphism**

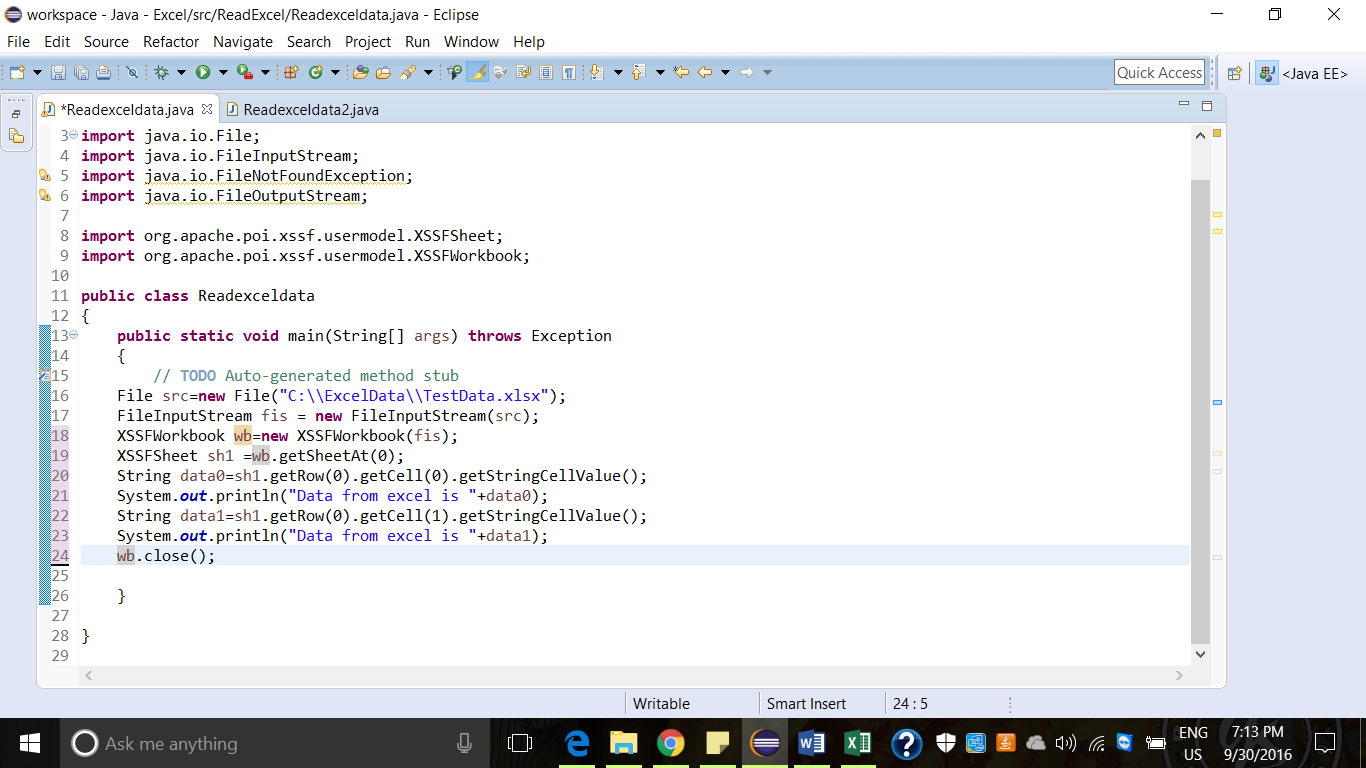
It is nothing but the ability to take more than one form. This can be applied to both operations as well as objects. It is tightly coupled inheritance. It means one interface, many possible implementations.

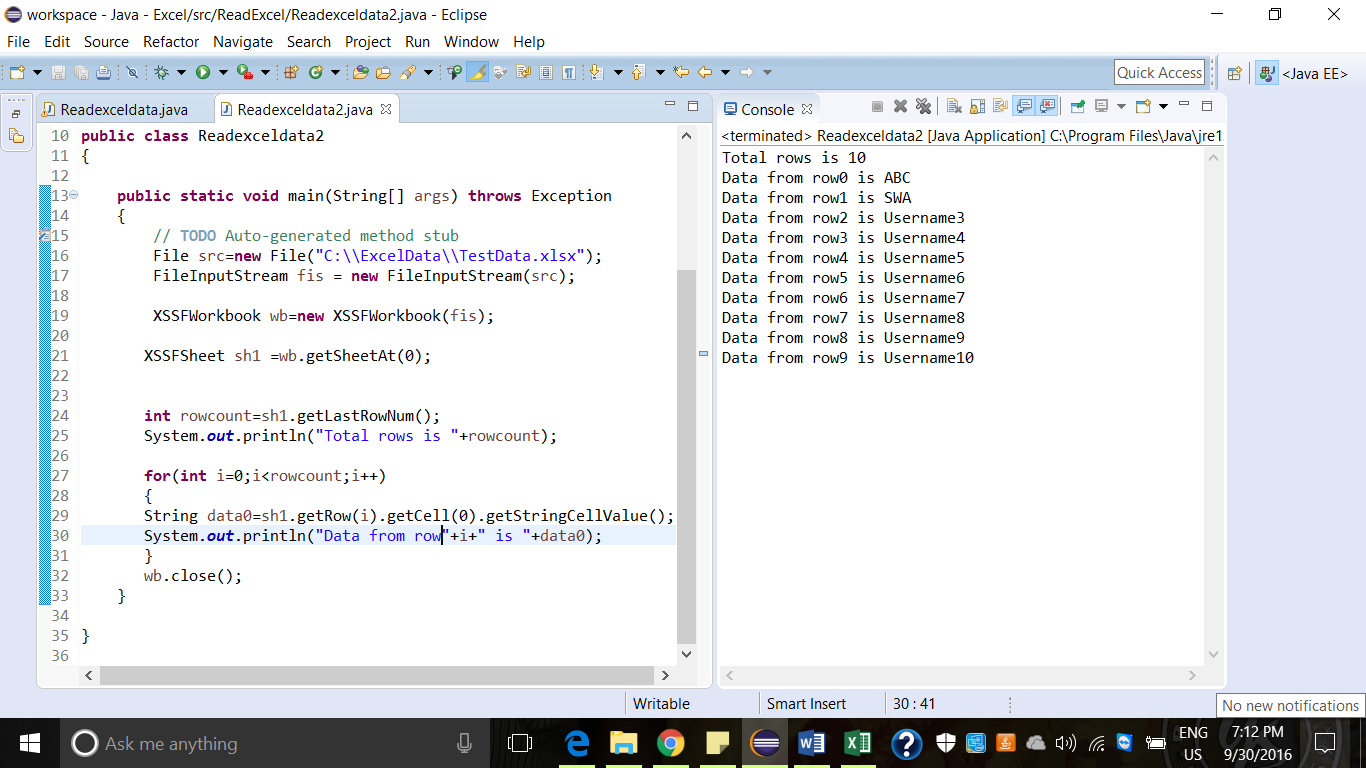
* **implementing interface**

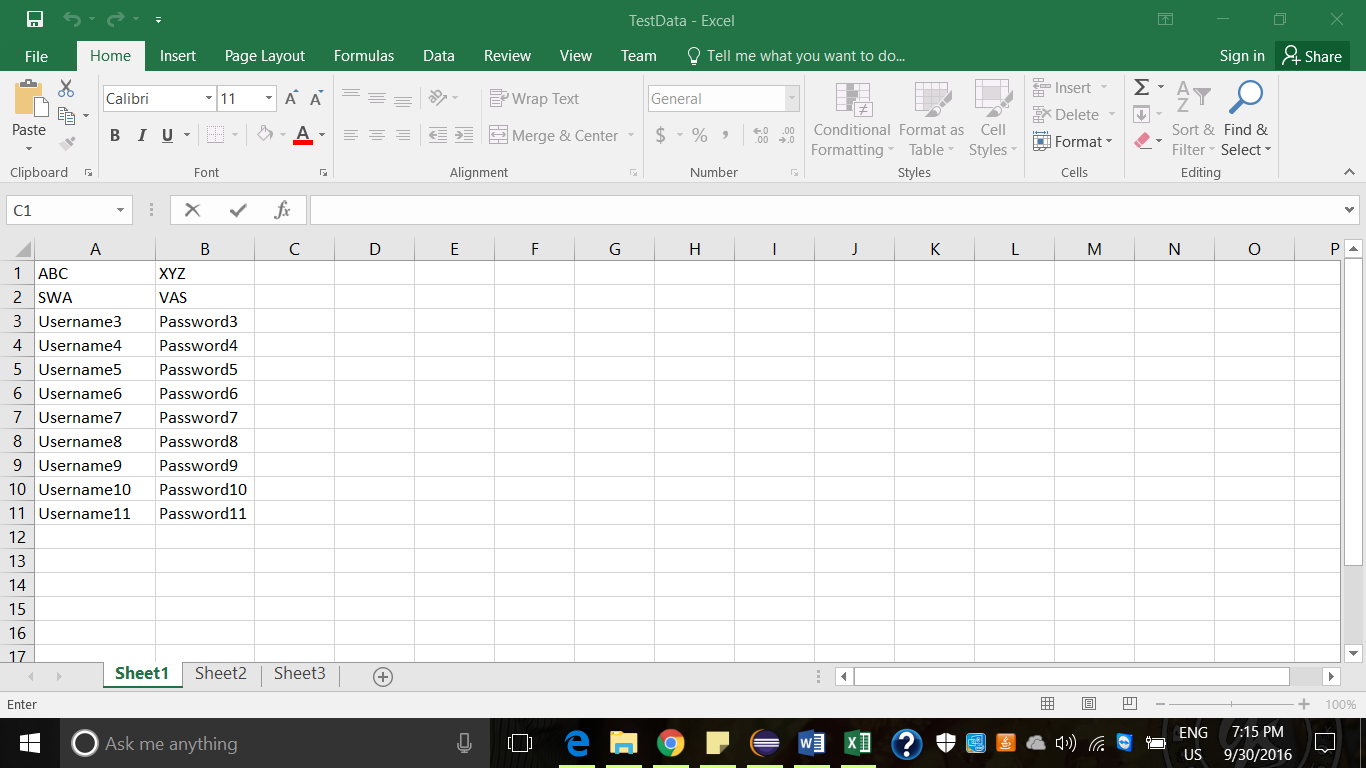




* **write a code to save data into excel file and read from excel file (POI and jexcel API)**



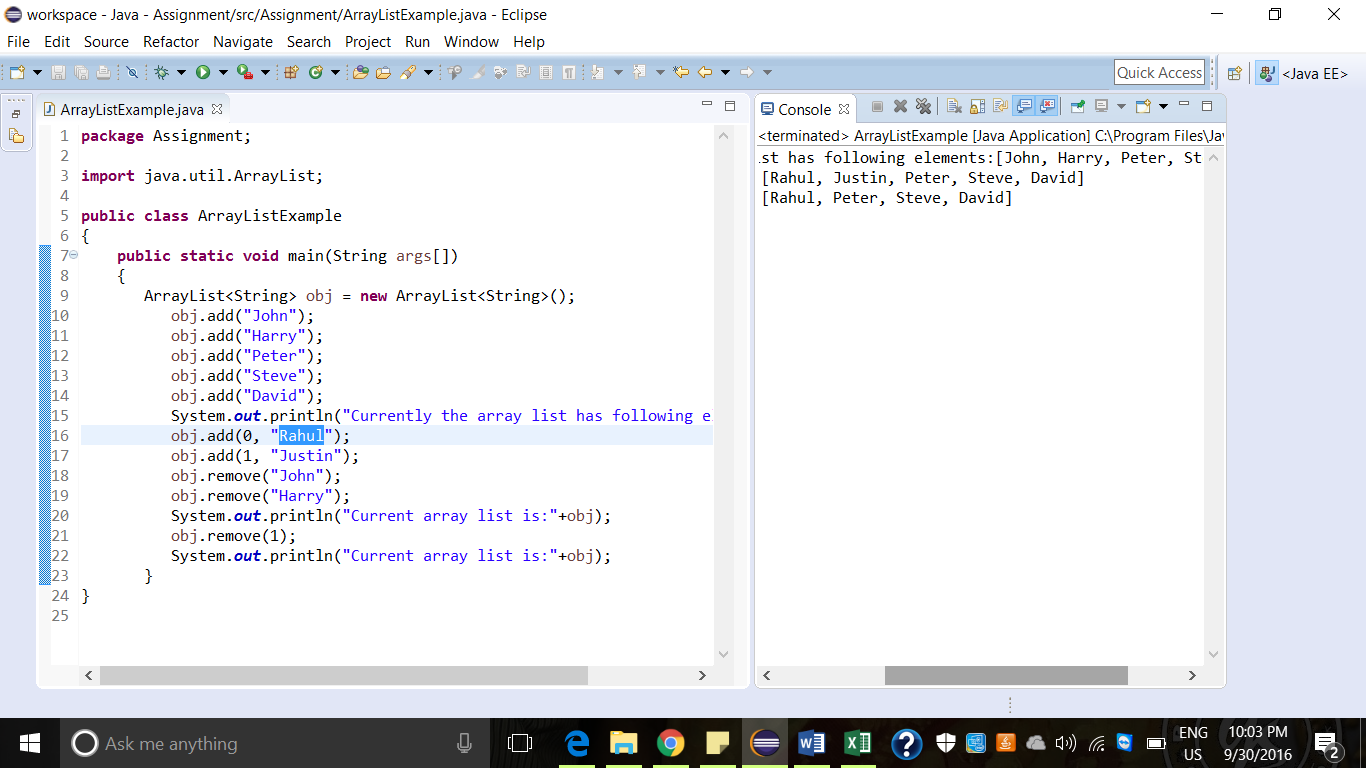




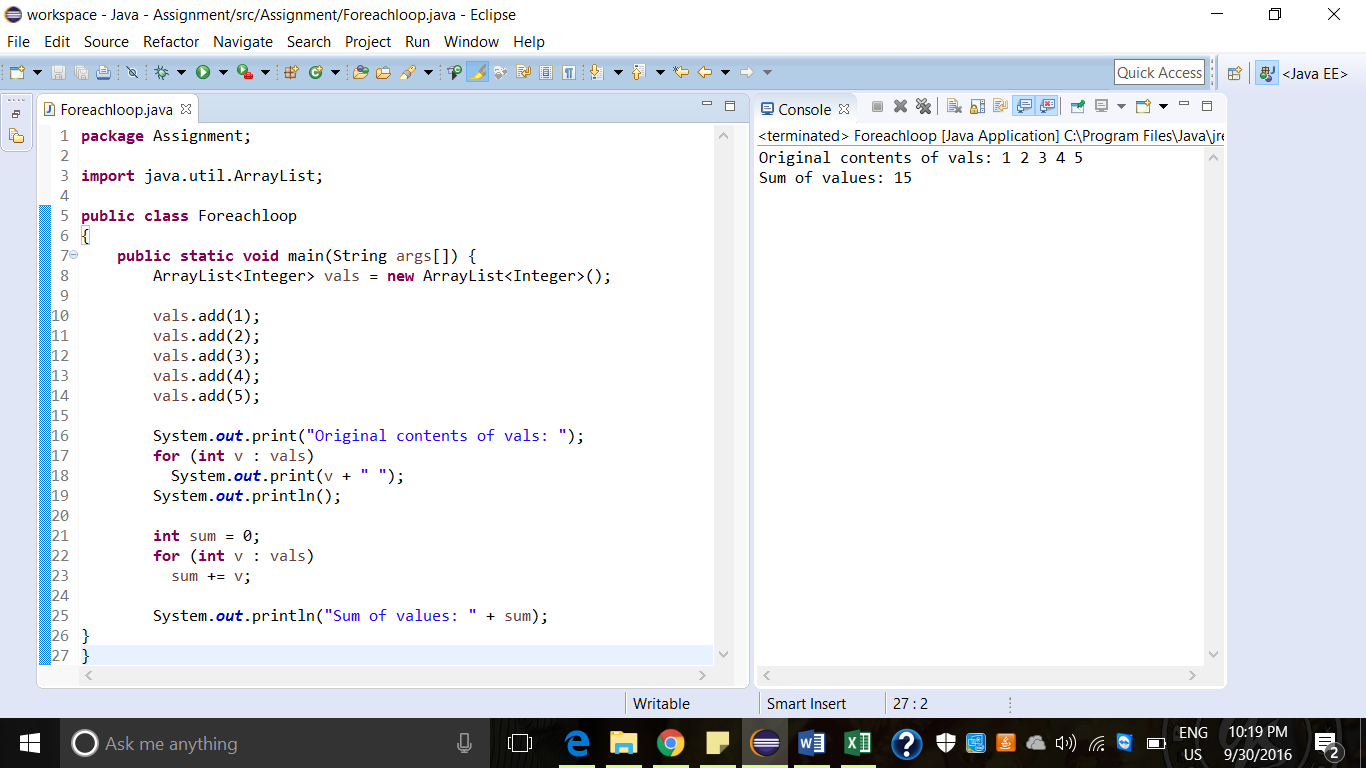
* **how to update the data into XML file and read data from XML file**
* **write code to add items to integer, string array**

write code to retrieve items from integer, string **array**

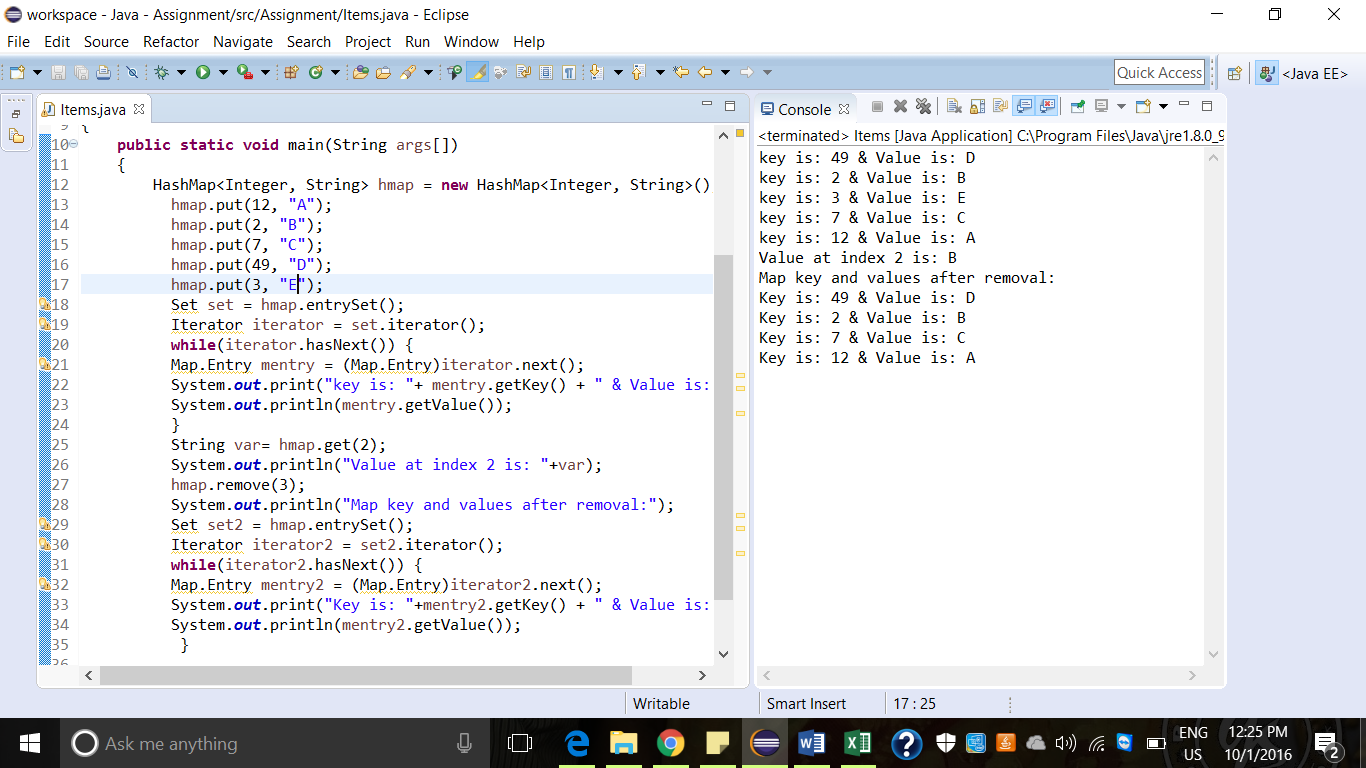
* **write code to add items to ArrayList collection**



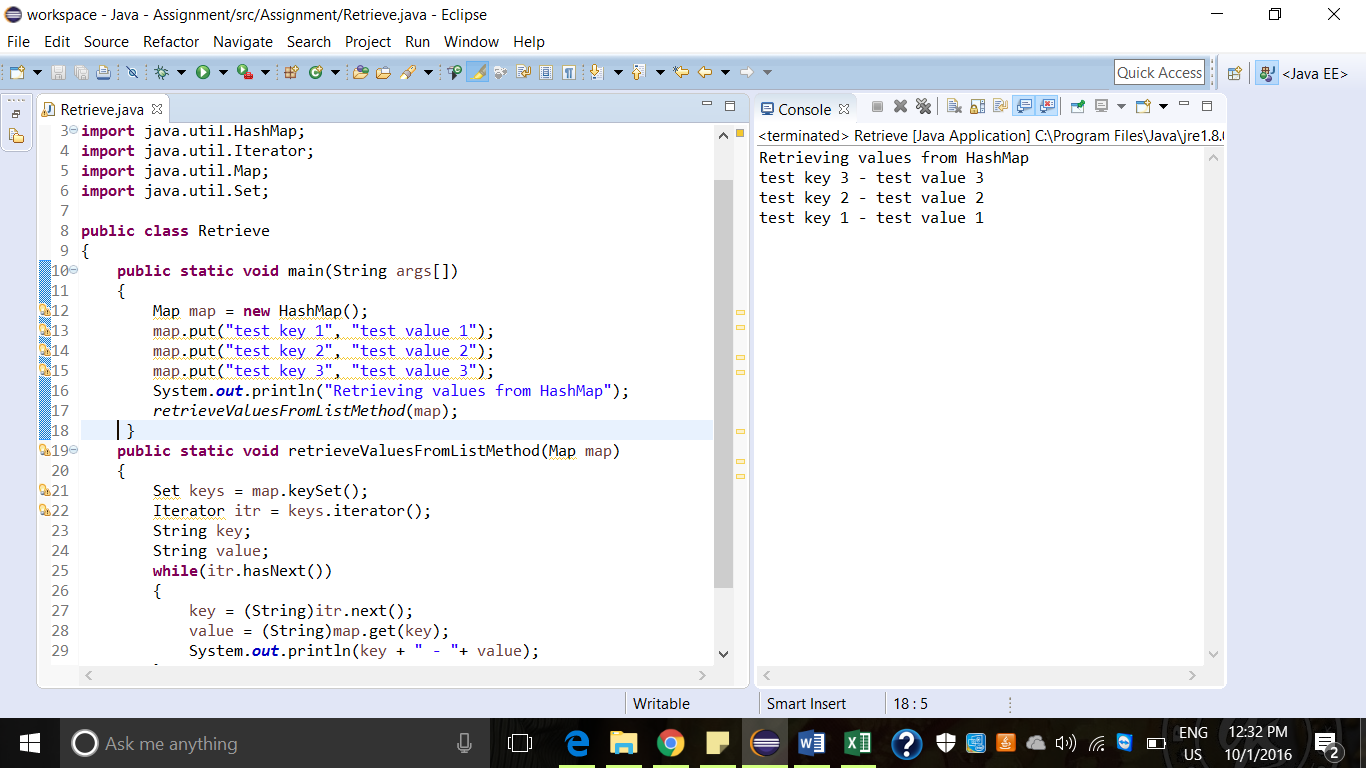
* **write code to retrieve items from array list (using for each loop)**



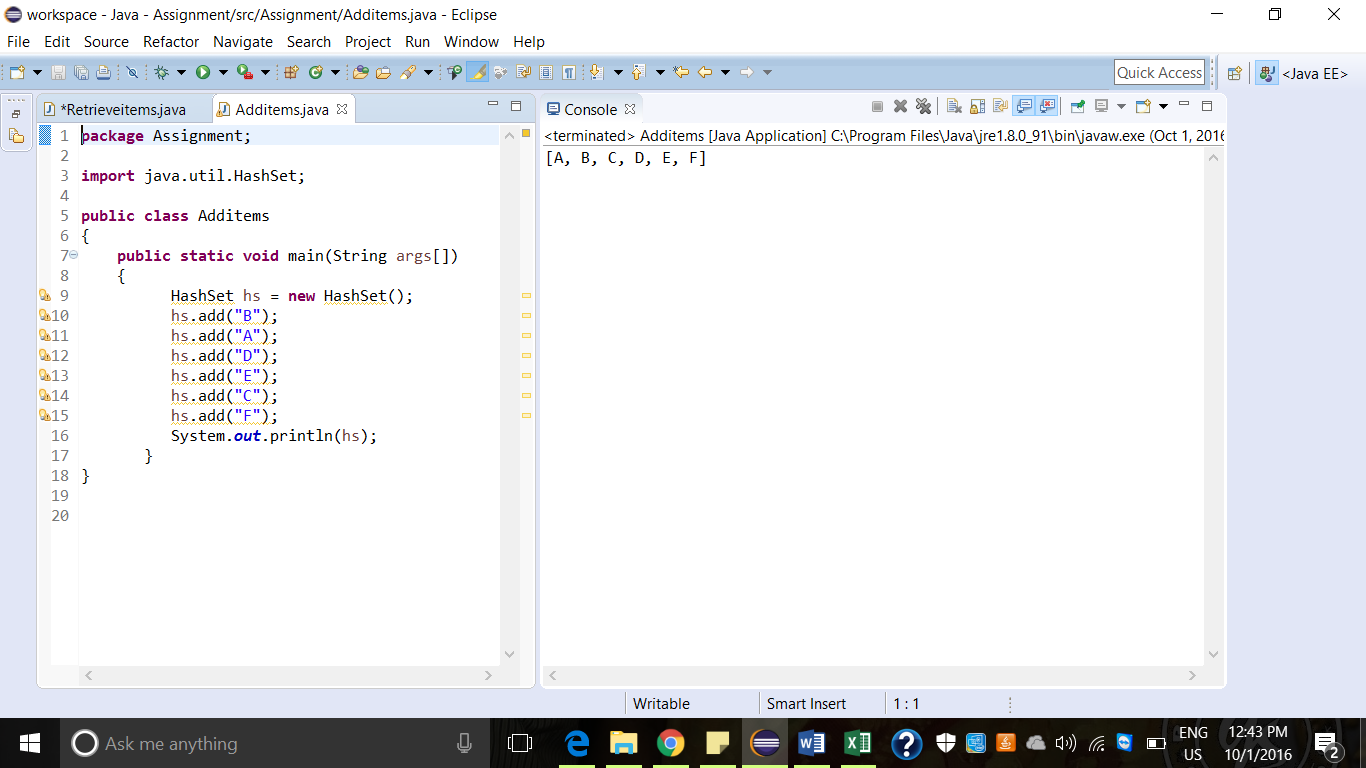
* **write code to add items HashMap**



* **write code to retrieve items HashMap**



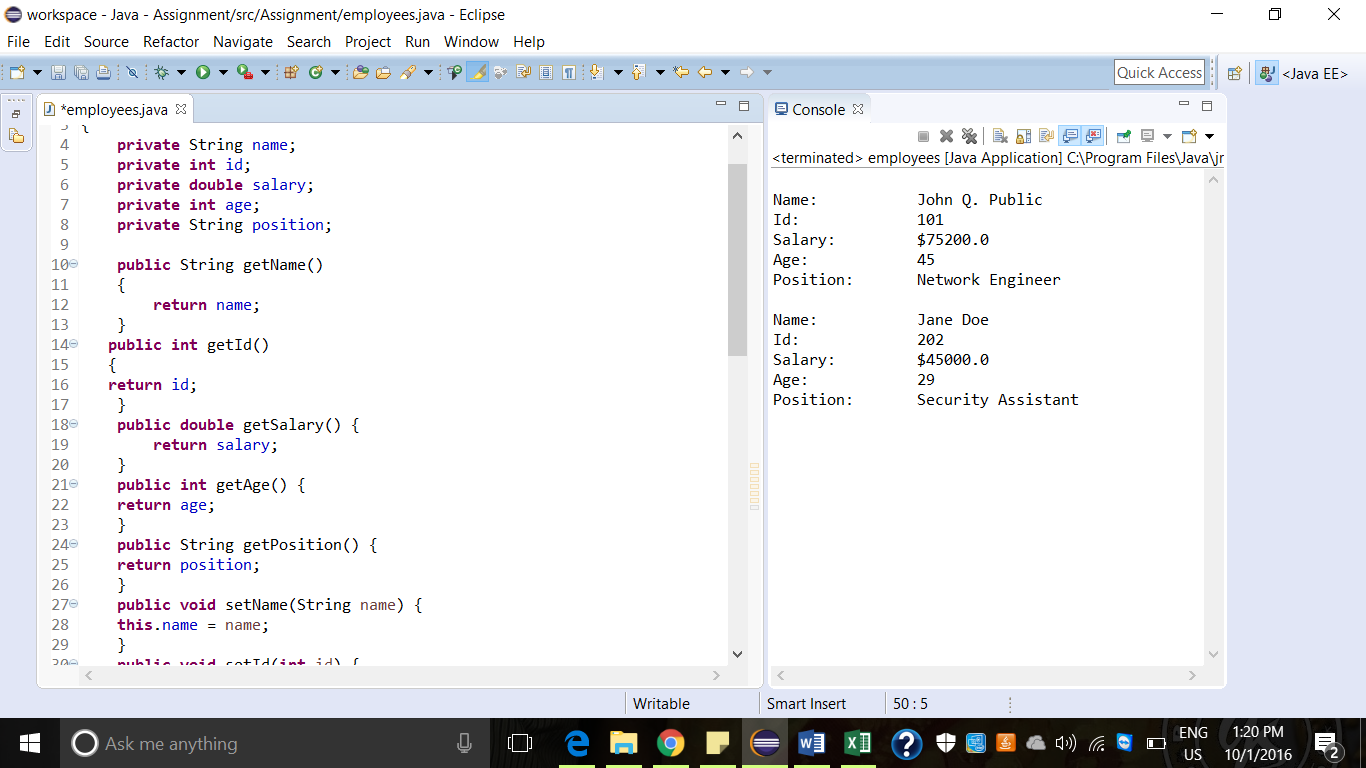
* **Write code to add items to hashset**

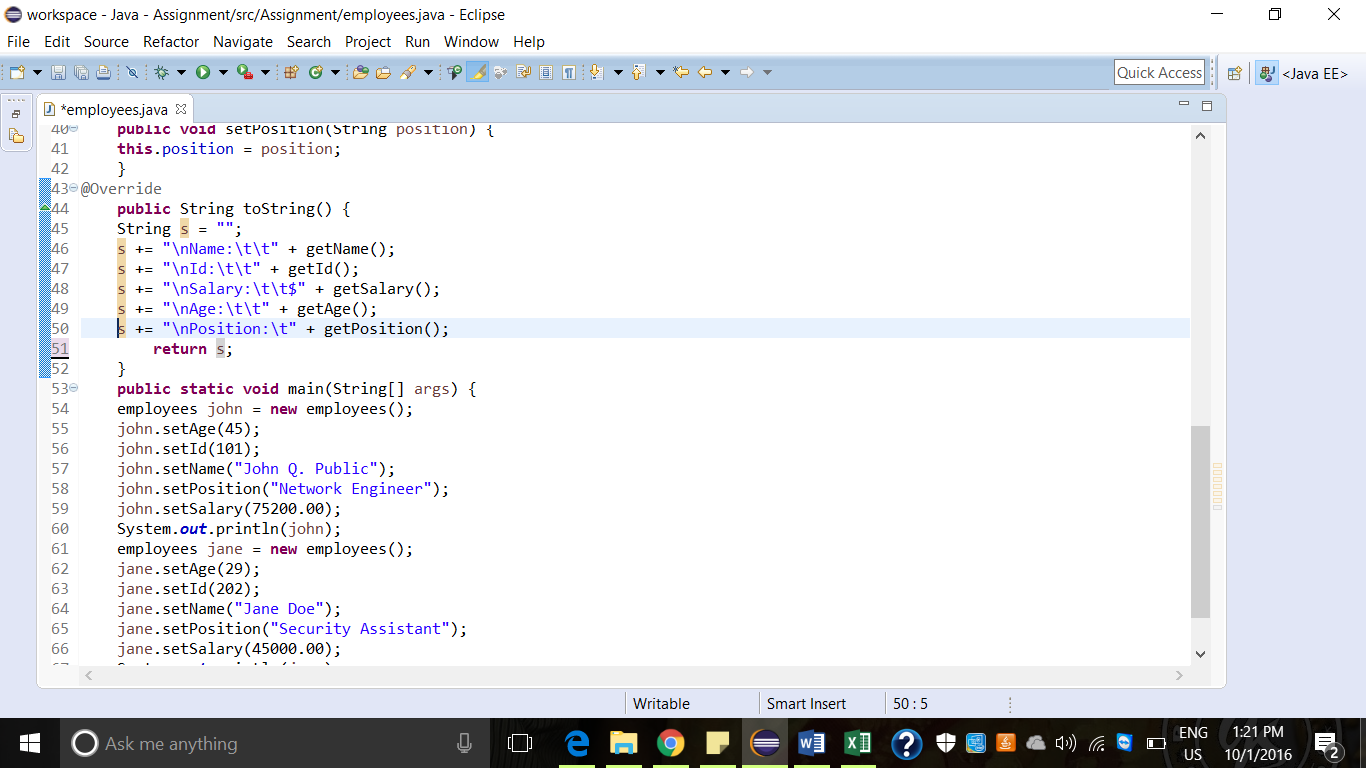


* **Write code to retrieve items to hasset**



* **write code to connect to JDBC to get rows from employee table**
* **create Employee class**





Add employee class to list collection

create method that return list of employee collection

* **Difference between string, string buffer, string builder with example**

**String:**

String is immutable object. The object created as a string is stored in the constant string pool. String cannot be used by two threads simultaneously.  
String  once assigned cannot be changed.

String demo = “hello”;

**String Buffer:**  
String Bufferis mutable means one can change the value of the object . The object created through String Buffer is stored in the heap.

String Buffer demo1 = new String Buffer("Hello");

demo1=new String Buffer("Bye");

**String Builder**:  
  
String Builder  is same as the String Buffer , that is it stores the object in heap and it can also be modified . The main difference between the String Buffer and String Builder is thatString Builder is also not thread safe.

String Builder demo2= new String Builder("Hello");

demo2=new String Builder("Bye");