Steps to install java

1. Step 1: Download JDK. ...
2. Step 2: Install JDK and JRE. ...
3. Step 3: Include JDK's "bin" Directory in the PATH. ...
4. Step 4: Verify the JDK Installation

Steps to install eclipse

1. Click Eclipse. ...
2. Click the Windows 32 or 64 Bit Operating System for your machine, under the heading Eclipse Standard 4.4 (right under Package Solutions). ...
3. Click the orange DOWNLOAD button. ...
4. Move this file to a more permanent location, so that you can install Eclipse (and reinstall it later, if necessary).

Steps to create workspace

create project

File -> Project

Project -> src-> class name ->finish

Steps to create project

Open eclipse

File -.>project

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

select file->project

select project, right click and select package

what is main method will do?

Main method is starting point of program

creating property/data members :

int salary

what is data type and different data types`

a particular kind of data item, as defined by the values it can take, the programming language used, or the operations that can be performed on it.

Int

Double

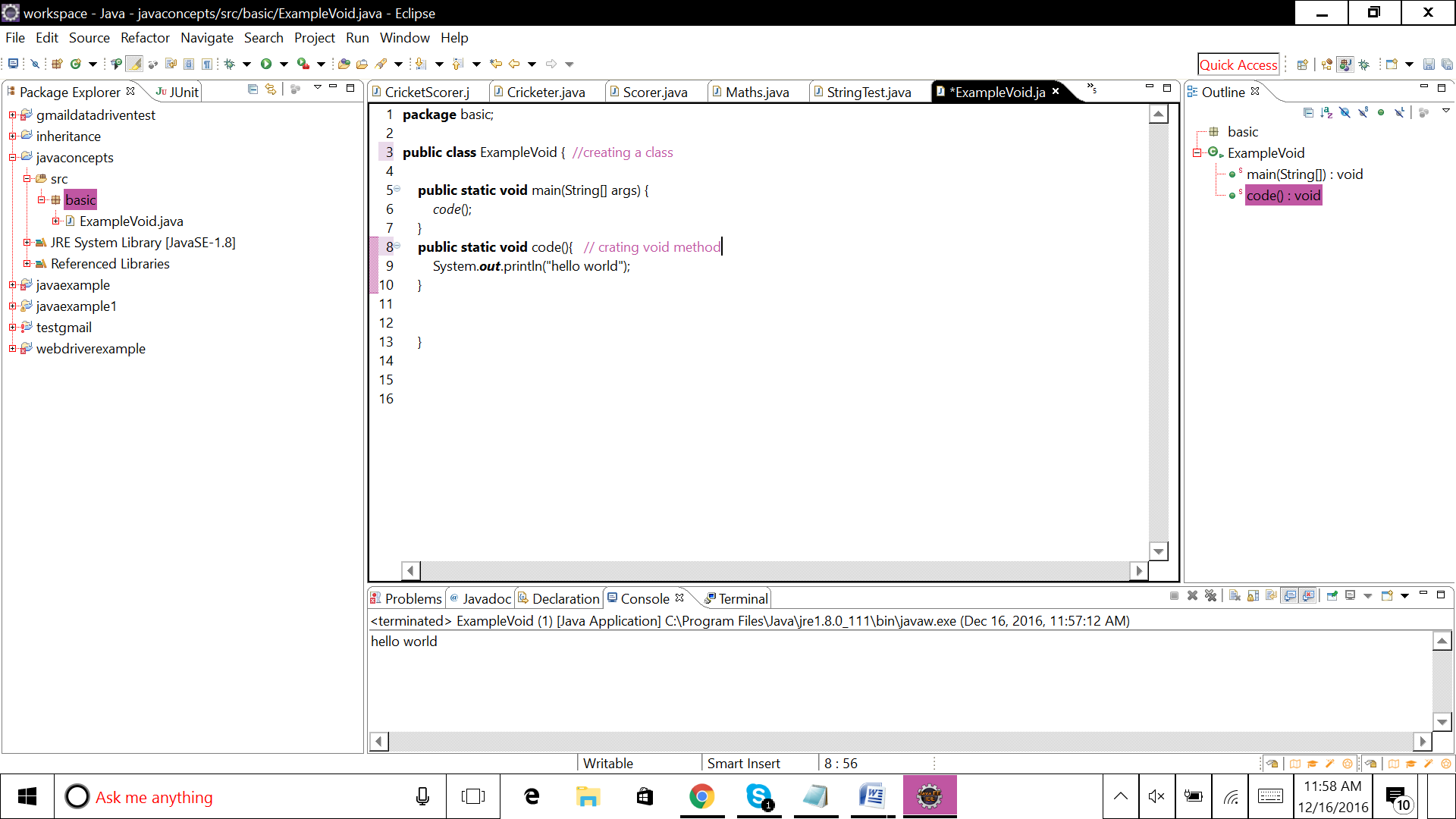
Float

Char

What is variable?

A Java variable is a piece of memory that can contain a data value. A variable thus has a data **type**.. Variables are typically used to store information .

creating method with void:



**package** basic;

**public** **class** ExampleVoid { //creating a class

**public** **static** **void** main(String[] args) {

*code*();

}

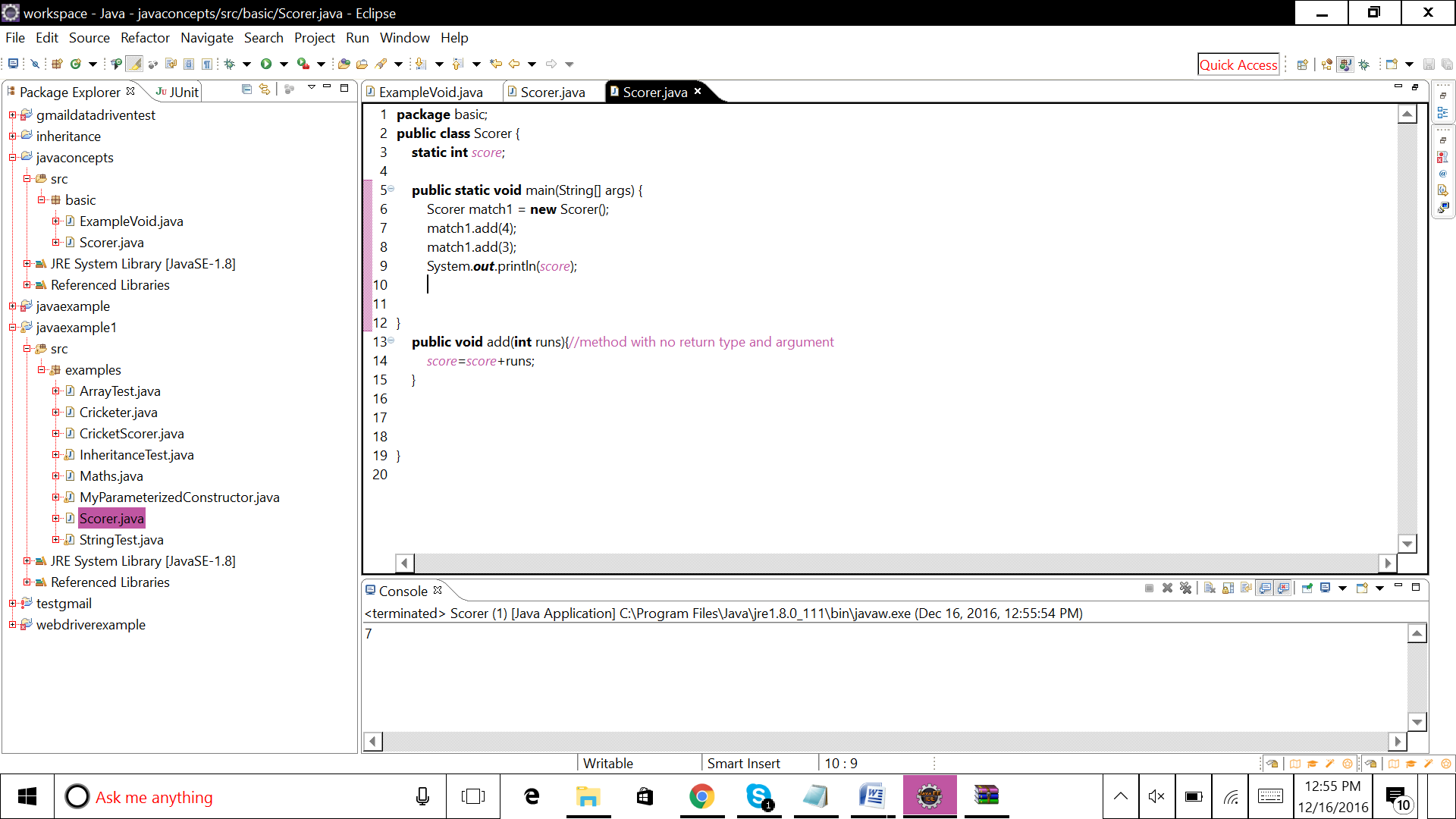
**public** **static** **void** code(){ // crating void method

System.***out***.println("hello world");

}

}

creating method with void and parameter :



**package** basic;

**public** **class** Scorer {

**static** **int** *score*;

**public** **static** **void** main(String[] args) {

Scorer match1 = **new** Scorer();

match1.add(4);

match1.add(3);

System.***out***.println(*score*);

}

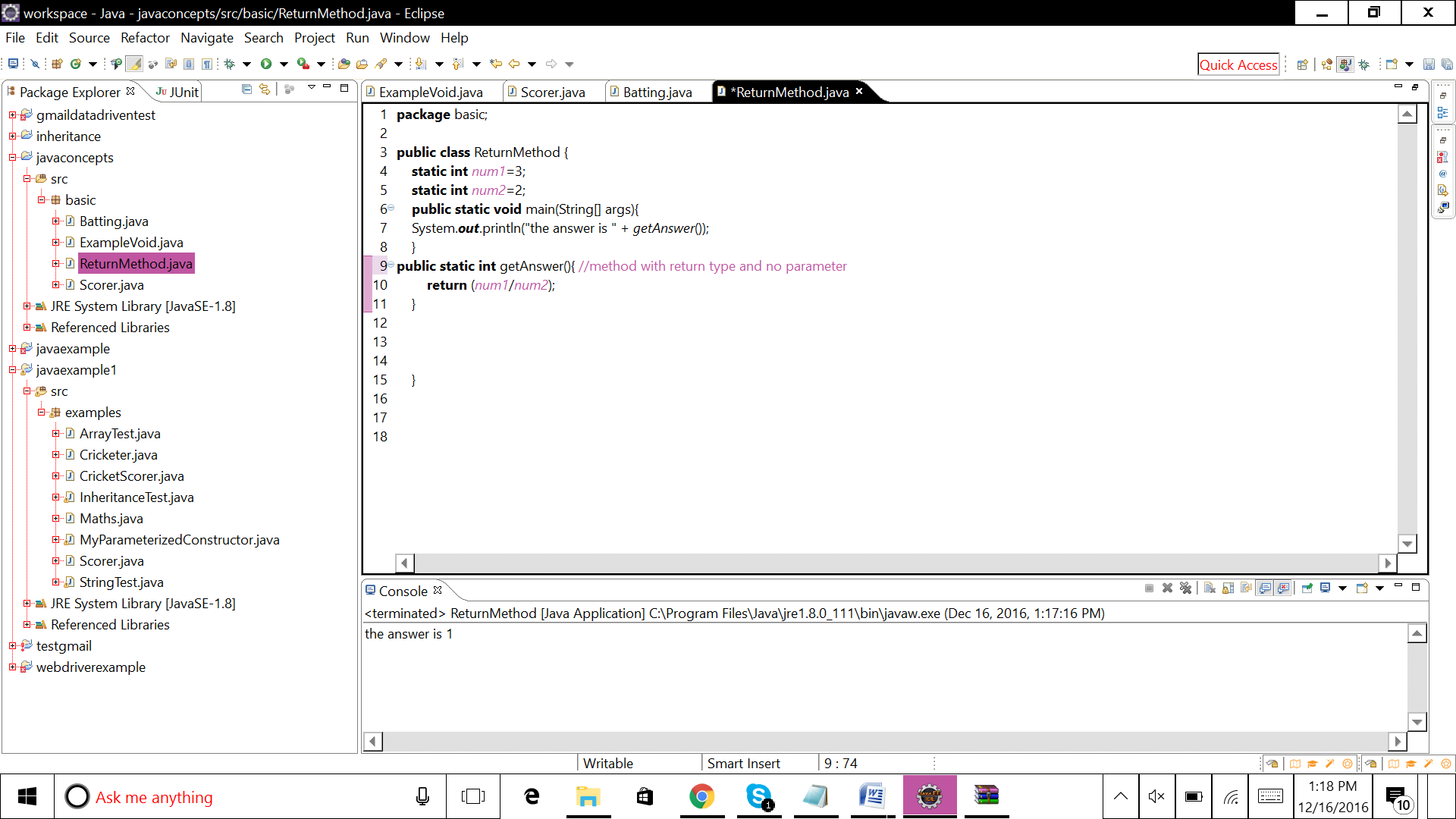
**public** **void** add(**int** runs){//method with void and parameter

*score*=*score*+runs;

}

}

creating method with return data type



**package** basic;

**public** **class** ReturnMethod {

**static** **int** *num1*=3;

**static** **int** *num2*=2;

**public** **static** **void** main(String[] args){

System.***out***.println("the answer is " + *getAnswer*());

}

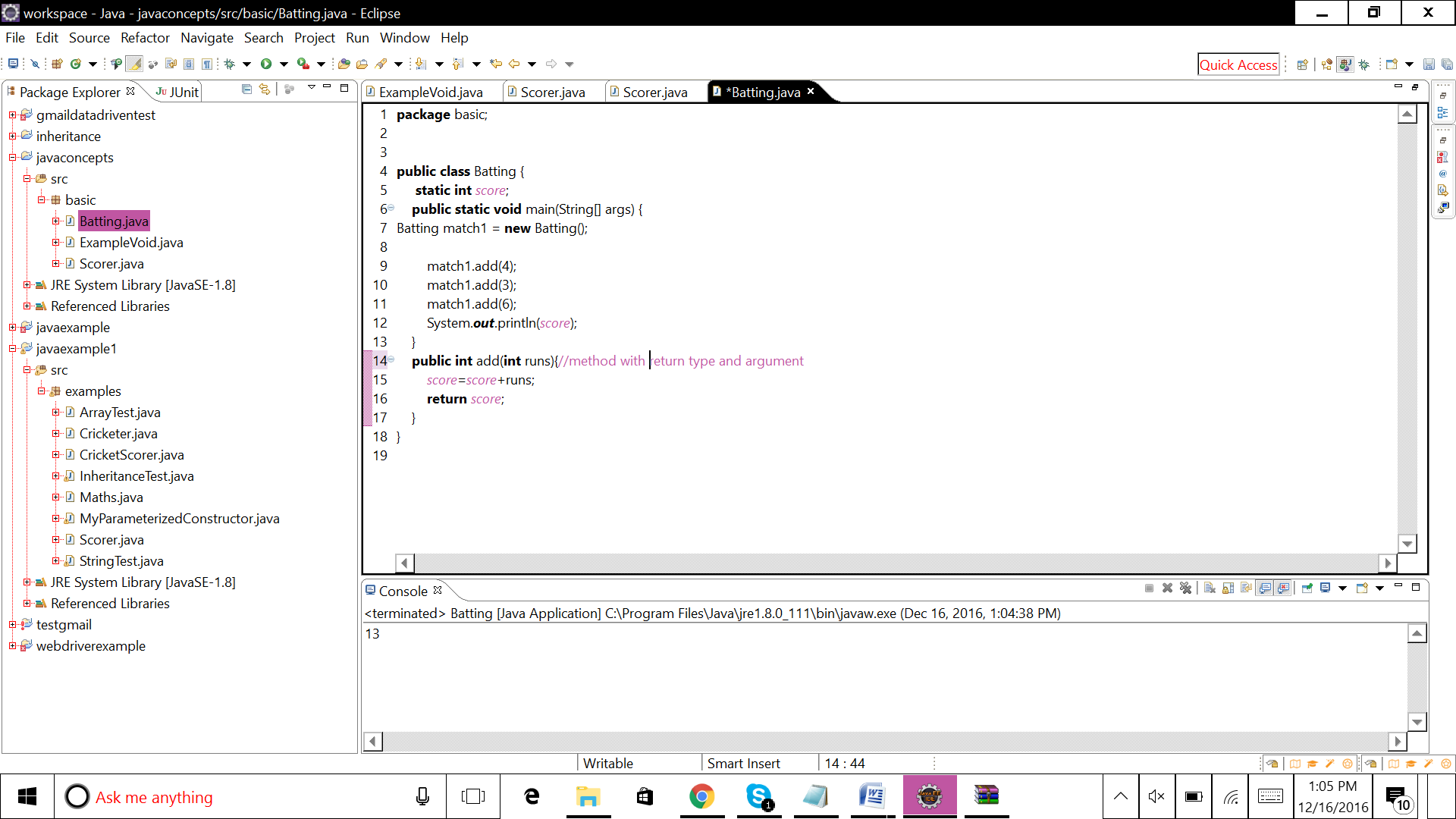
**public** **static** **int** getAnswer(){ //method with return type and no parameter

**return** (*num1*/*num2*);

}

}

creating method with return data type and parameter:



**package** basic;

**public** **class** Batting {

**static** **int** *score*;

**public** **static** **void** main(String[] args) {

Batting match1 = **new** Batting();

match1.add(4);

match1.add(3);

match1.add(6);

System.***out***.println(*score*);

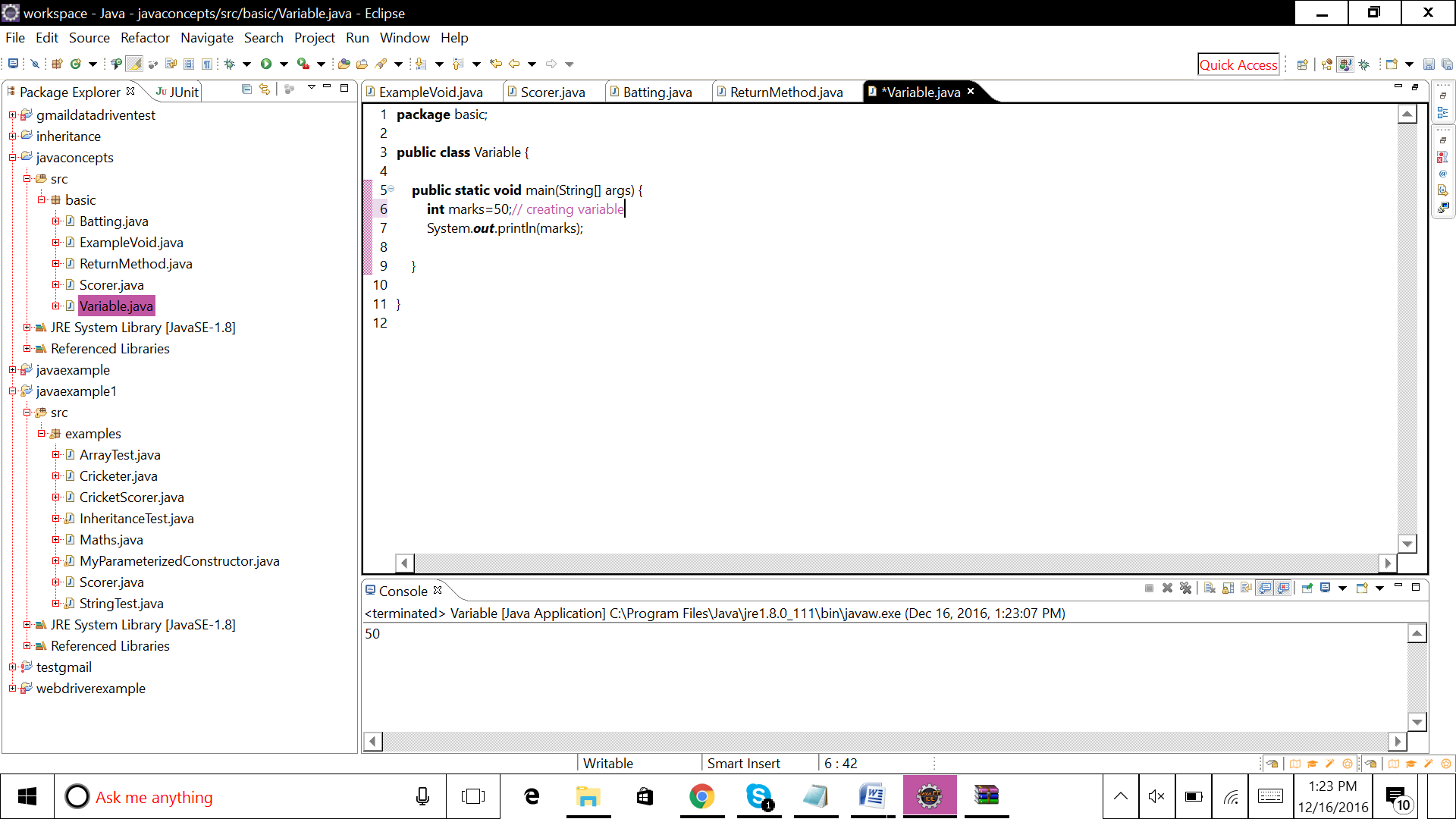
}

**public** **int** add(**int** runs){//method with return type and argument

*score*=*score*+runs;

**return** *score*;

creating variable



**package** basic;

**public** **class** Variable {

**public** **static** **void** main(String[] args) {

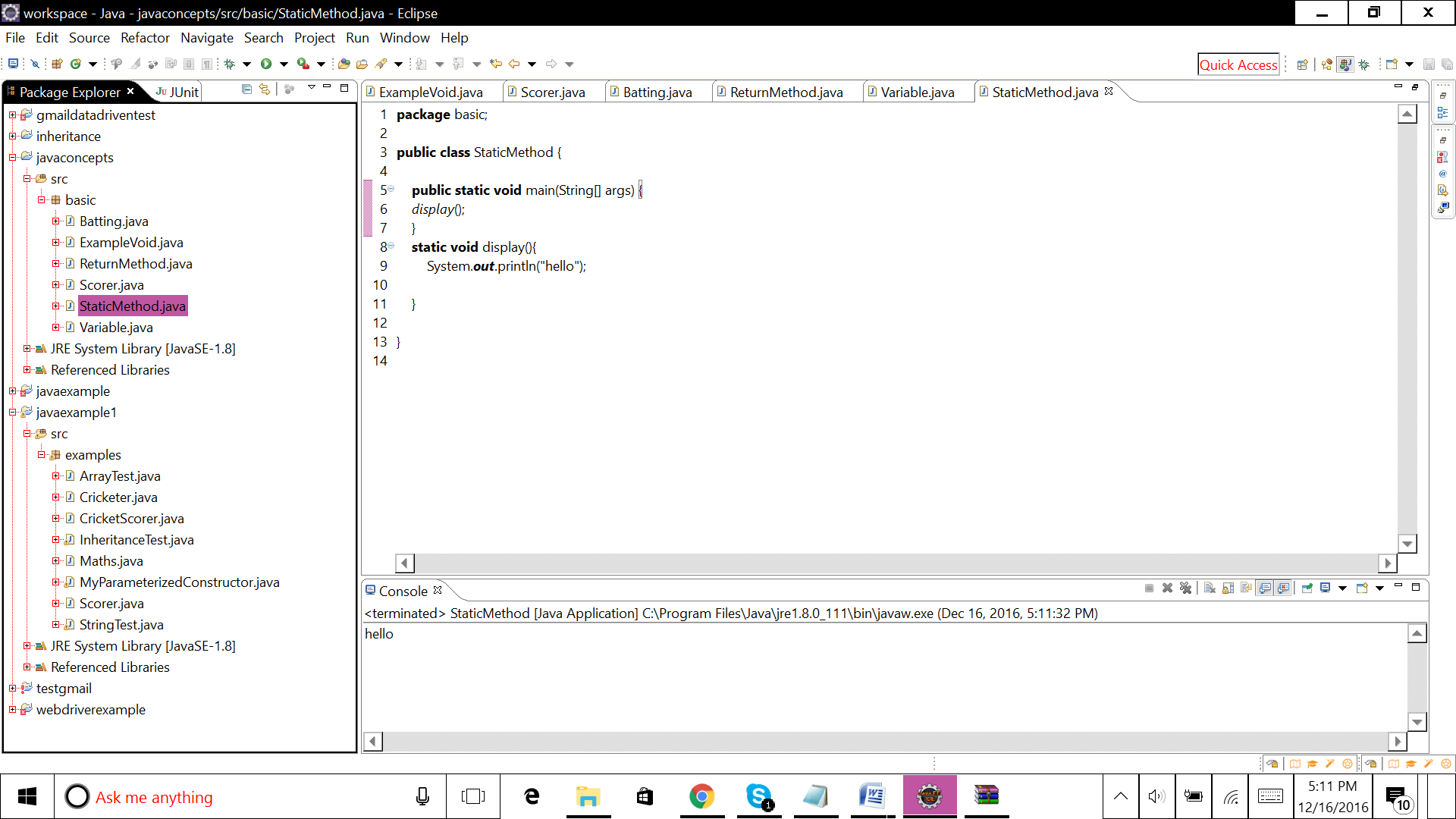
**int** marks=50;// creating variable

System.***out***.println(marks);

}

}

creating static method



**package** basic;

**public** **class** StaticMethod {

**public** **static** **void** main(String[] args) {

*display*();

}

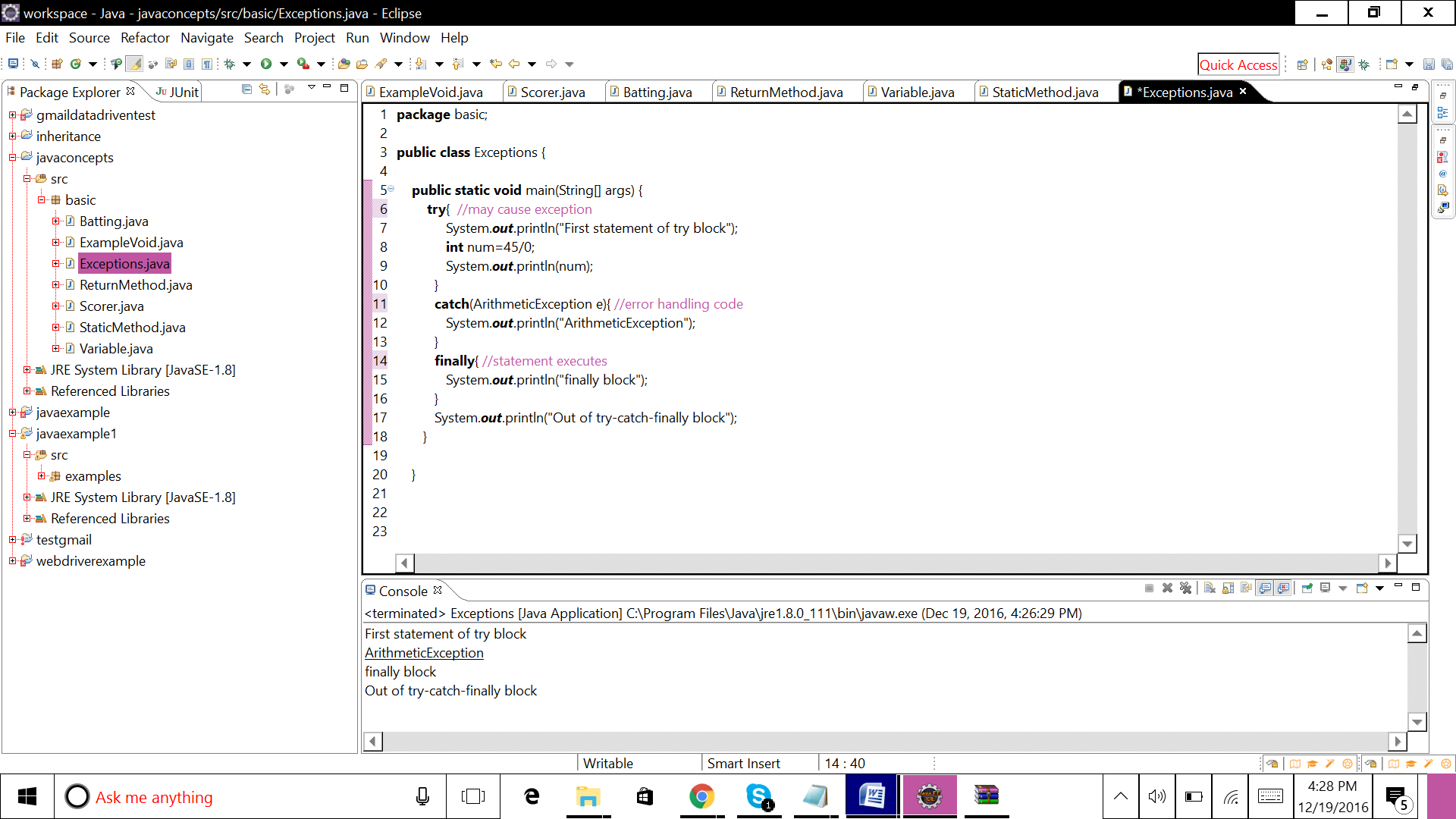
**static** **void** display(){

System.***out***.println("hello");

}

}

write code to handle exceptions with try/catch/finally



**package** basic;

**public** **class** Exceptions {

**public** **static** **void** main(String[] args) {

**try**{ //may cause exception

System.***out***.println("First statement of try block");

**int** num=45/0;

System.***out***.println(num);

}

**catch**(ArithmeticException e){ //error handling code

System.***out***.println("ArithmeticException");

}

**finally**{ //statement executes

System.***out***.println("finally block");

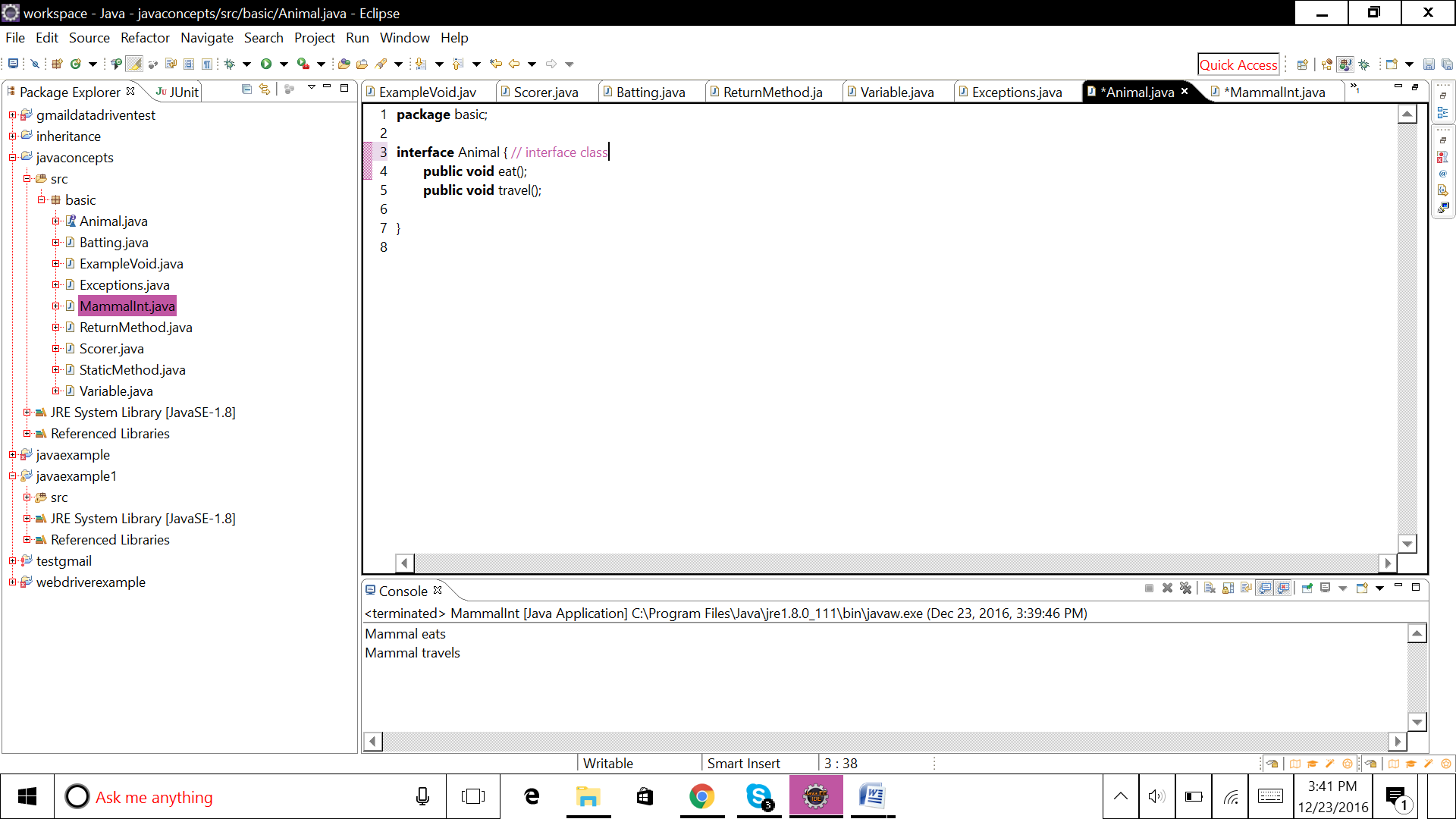
}

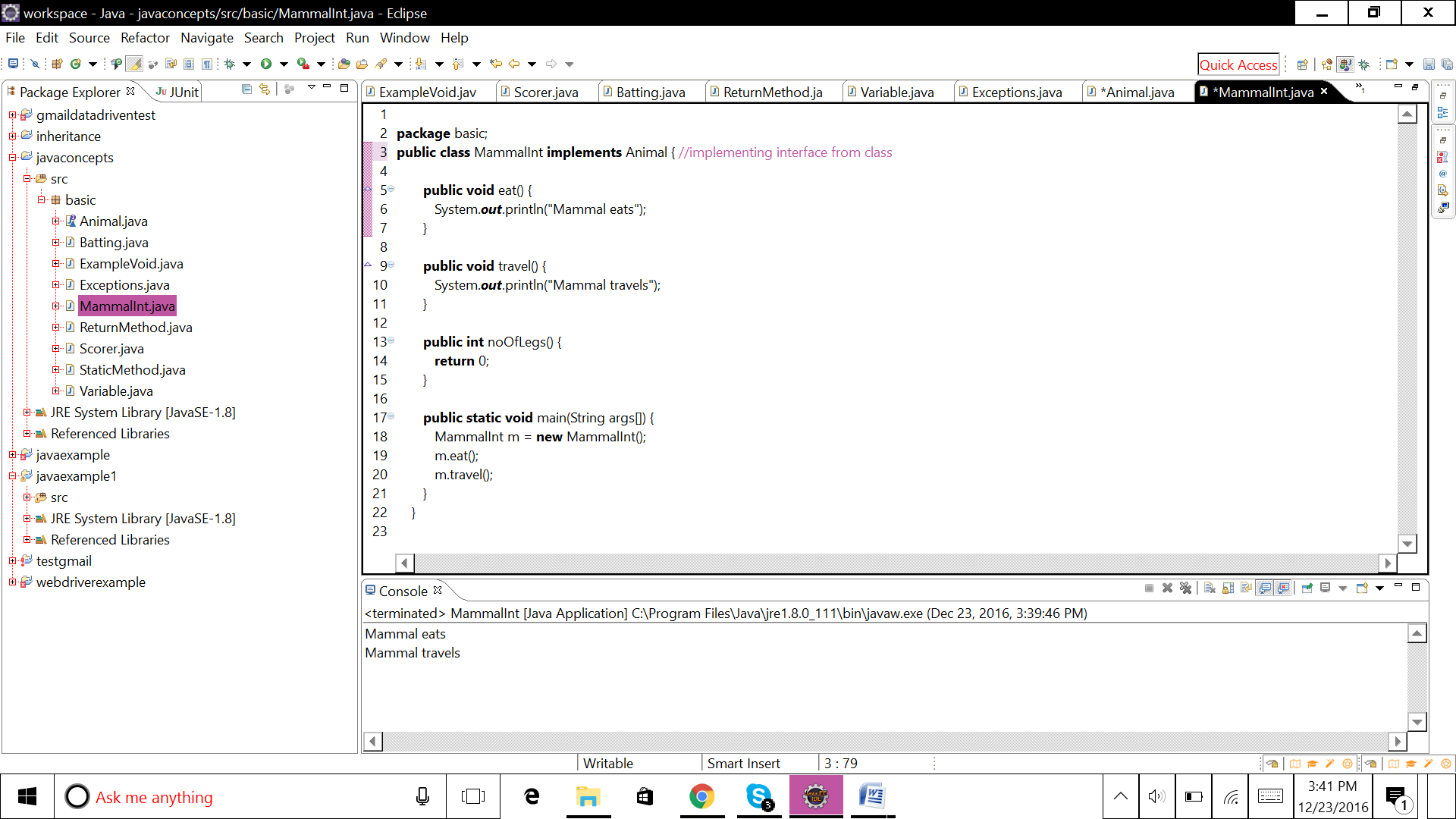
System.***out***.println("Out of try-catch-finally block");

}

}

write code for interface and create class to implement that interface





**package** basic;

**interface** Animal { // interface class

**public** **void** eat();

**public** **void** travel();

}

**package** basic;

**public** **class** MammalInt **implements** Animal { //implementing interface from class

**public** **void** eat() {

System.***out***.println("Mammal eats");

}

**public** **void** travel() {

System.***out***.println("Mammal travels");

}

**public** **int** noOfLegs() {

**return** 0;

}

**public** **static** **void** main(String args[]) {

MammalInt m = **new** MammalInt();

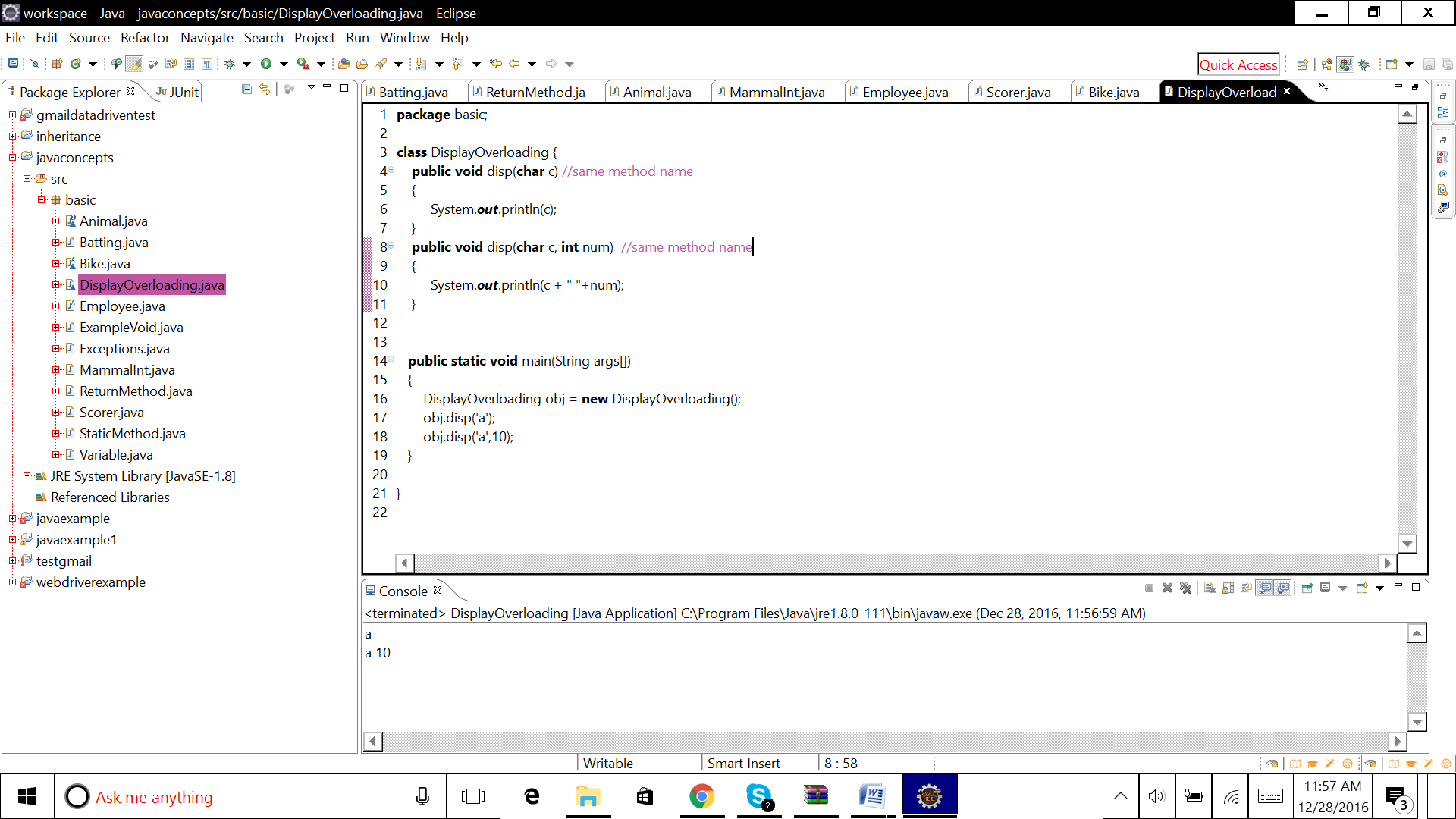
m.eat();

m.travel();

}

}

implement method overloading



**package** basic;

**class** DisplayOverloading {

**public** **void** disp(**char** c) //same method name

{

System.***out***.println(c);

}

**public** **void** disp(**char** c, **int** num) //same method name

{

System.***out***.println(c + " "+num);

}

**public** **static** **void** main(String args[])

{

DisplayOverloading obj = **new** DisplayOverloading();

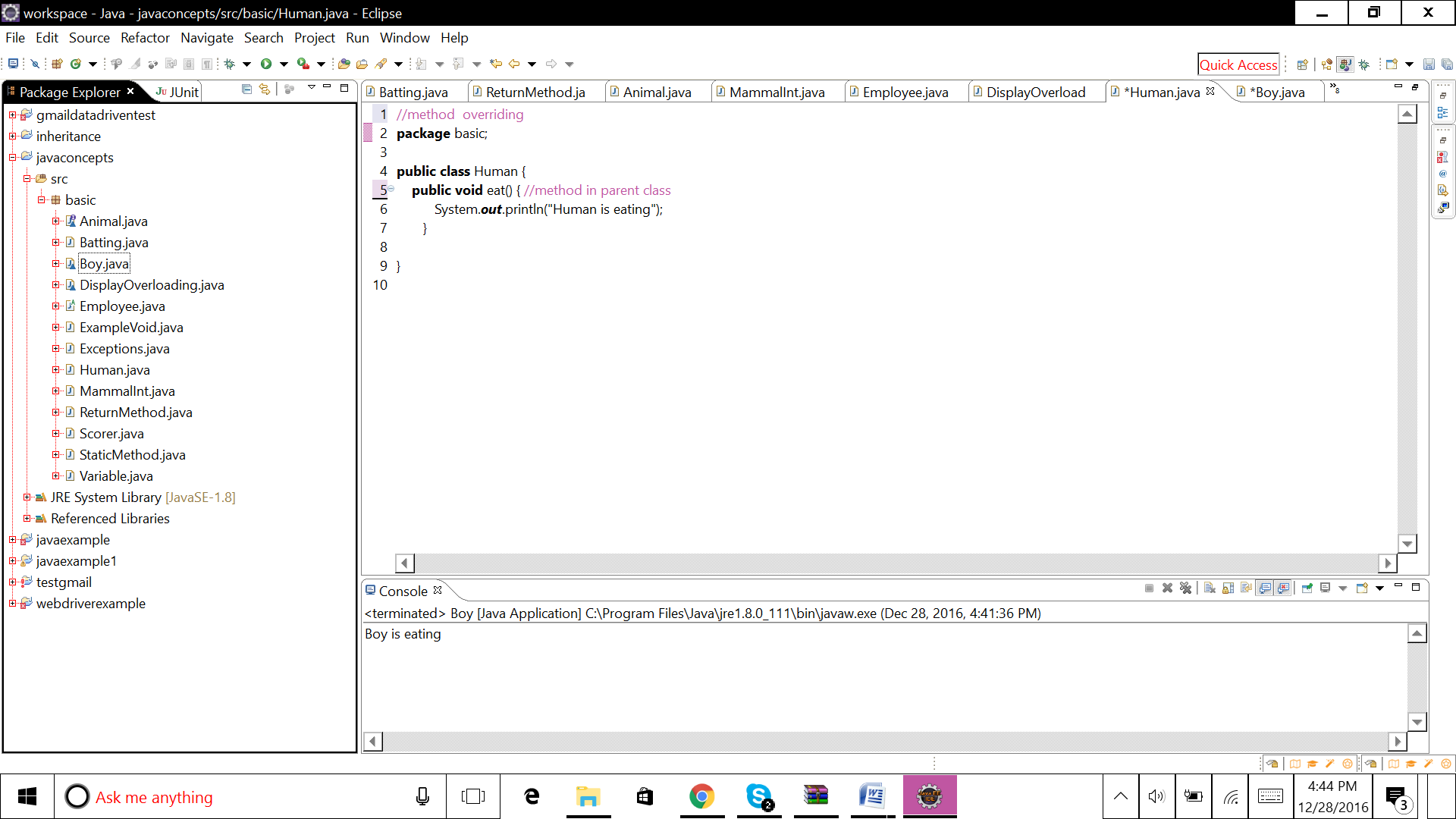
obj.disp('a');

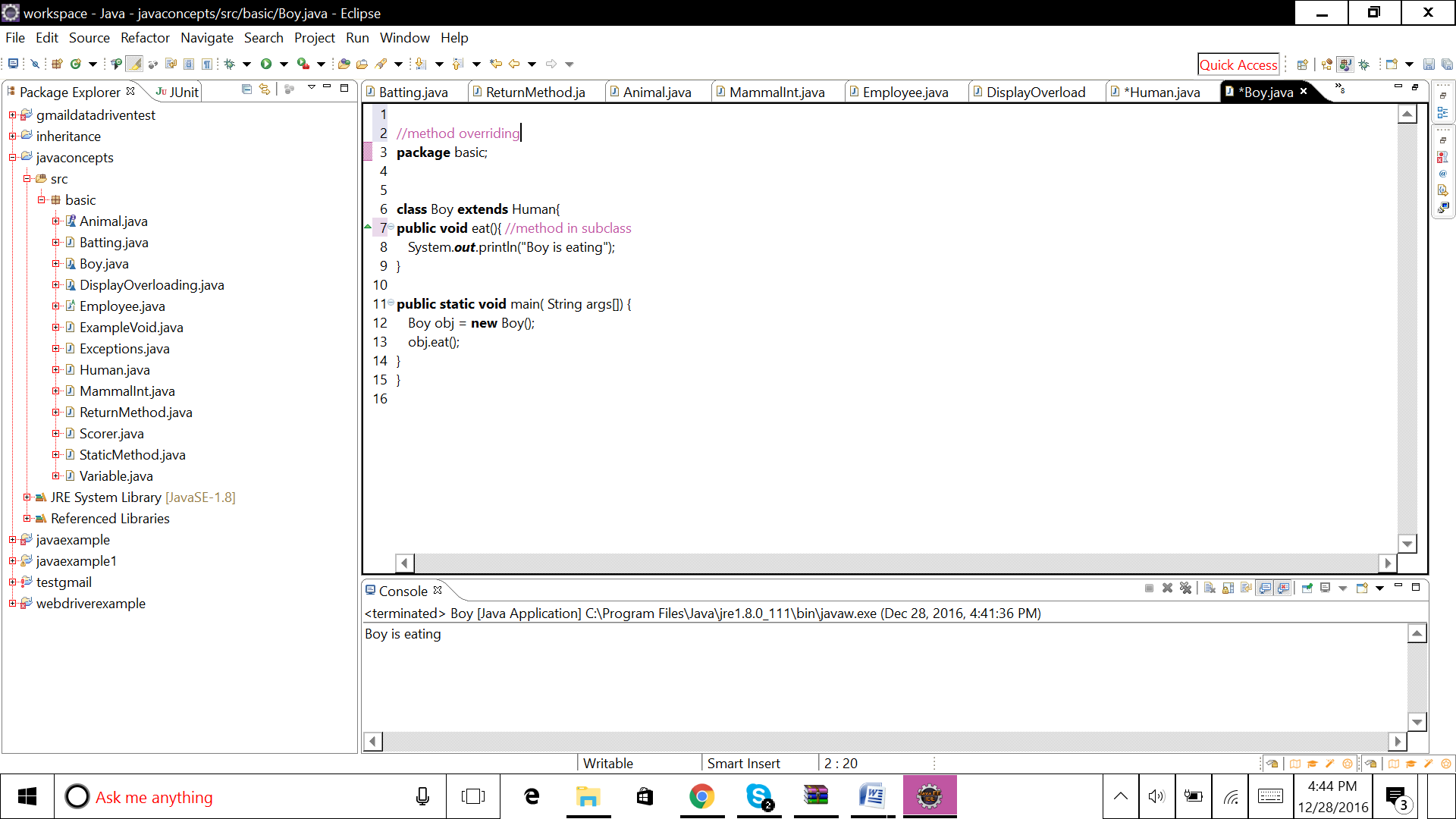
obj.disp('a',10);

}

}

implement method overriding





//method overriding

**package** basic;

**public** **class** Human {

**public** **void** eat() { //method in parent class

System.***out***.println("Human is eating");

}

}

//method overriding

**package** basic;

**class** Boy **extends** Human{

**public** **void** eat(){ //method in subclass

System.***out***.println("Boy is eating");

}

**public** **static** **void** main( String args[]) {

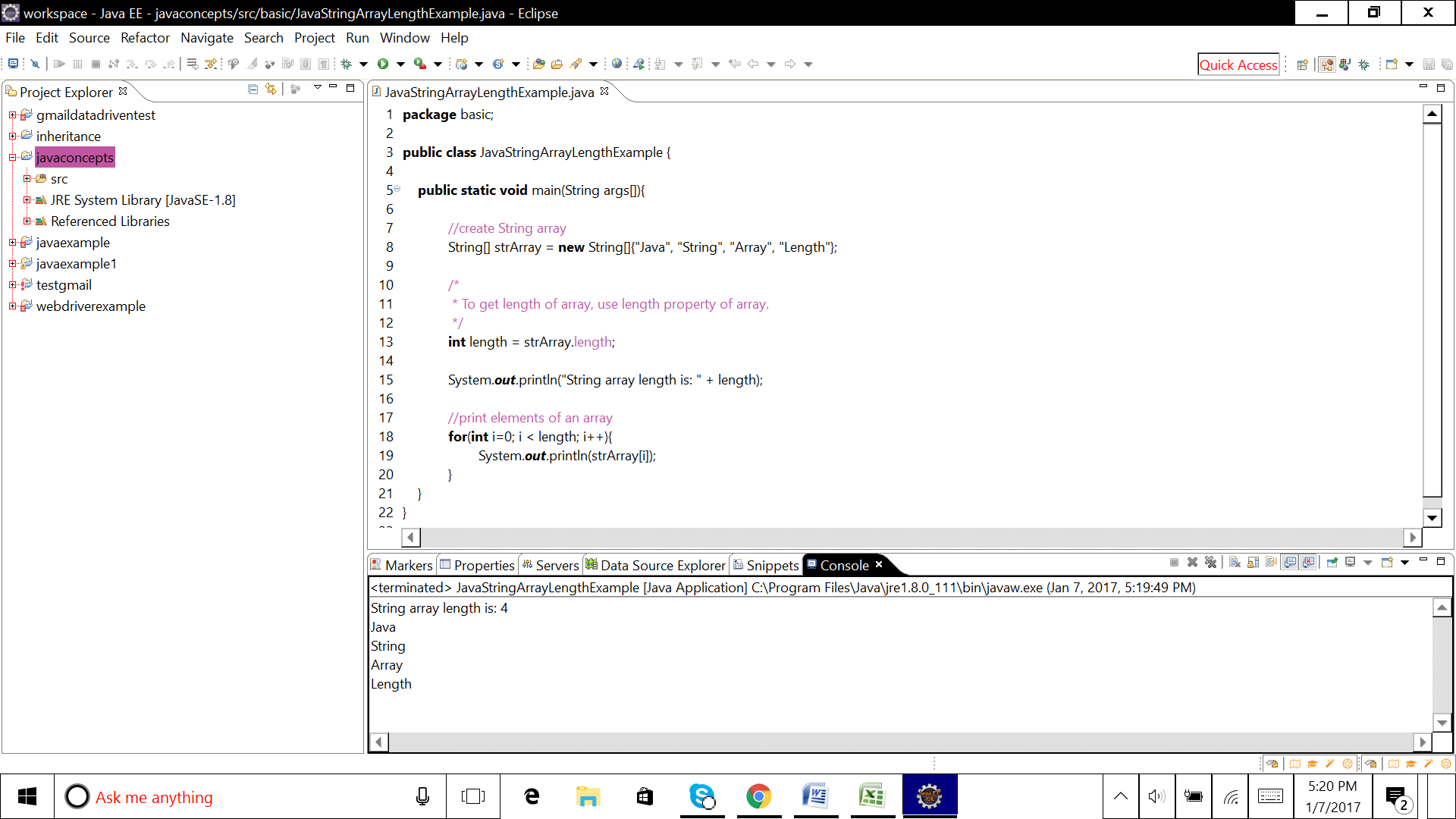
Boy obj = **new** Boy();

obj.eat();

}

}

**Java String Array Length Example**

****

**package** basic;

**public** **class** JavaStringArrayLengthExample {

**public** **static** **void** main(String args[]){

//create String array

String[] strArray = **new** String[]{"Java", "String", "Array", "Length"};

/\*

\* To get length of array, use length property of array.

\*/

**int** length = strArray.length;

System.***out***.println("String array length is: " + length);

//print elements of an array

**for**(**int** i=0; i < length; i++){

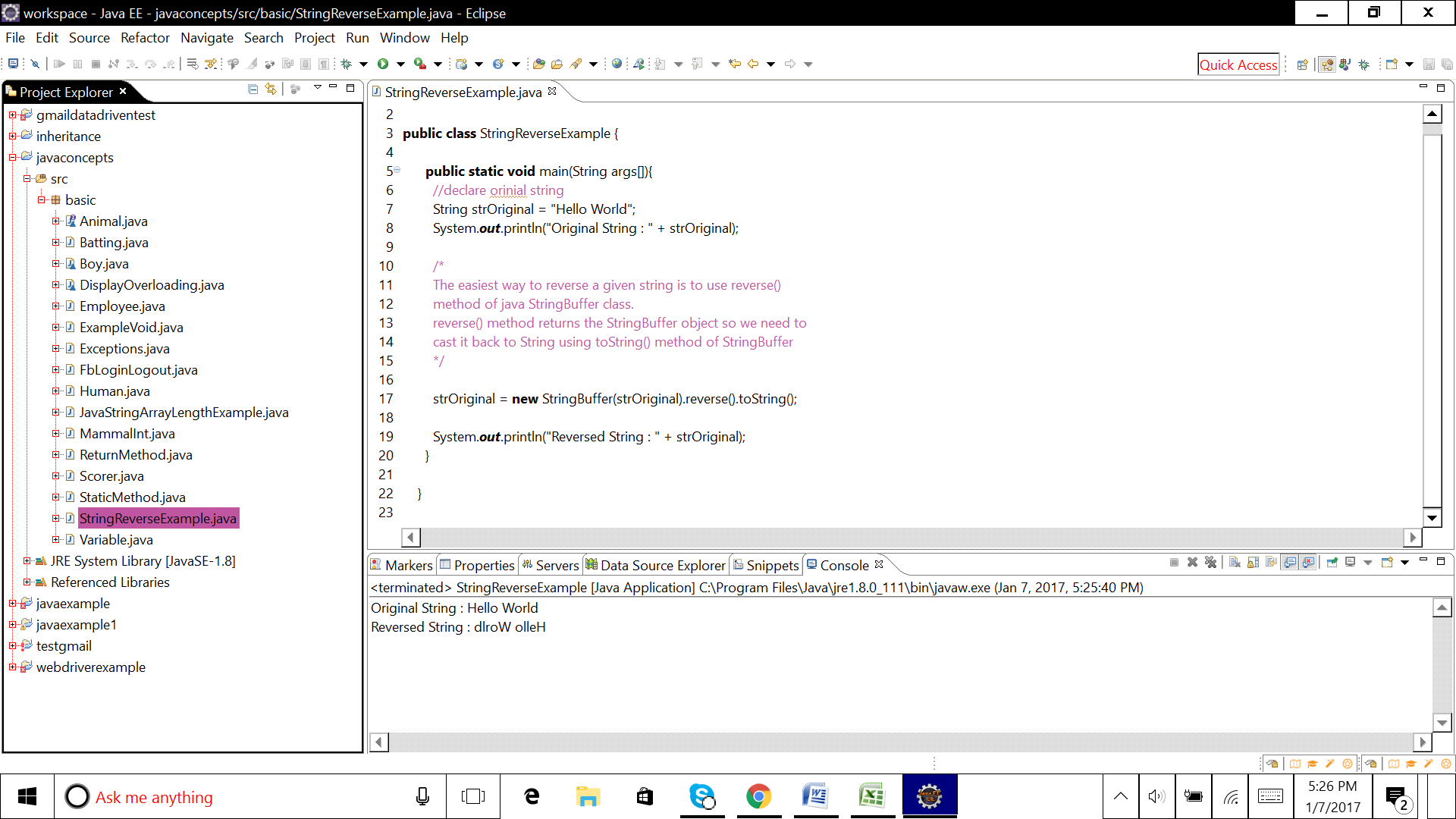
System.***out***.println(strArray[i]);

}

}

}

**Java String Reverse Example**

****

**public** **class** StringReverseExample {

**public** **static** **void** main(String args[]){

//declare orinial string

String strOriginal = "Hello World";

System.***out***.println("Original String : " + strOriginal);

/\*

The easiest way to reverse a given string is to use reverse()

method of java StringBuffer class.

reverse() method returns the StringBuffer object so we need to

cast it back to String using toString() method of StringBuffer

\*/

strOriginal = **new** StringBuffer(strOriginal).reverse().toString();

System.***out***.println("Reversed String : " + strOriginal);

}

}