Topics Covered

- Java variables and methods
- Java control statements
- Java operators

Java Language Basics

Variables

- **Local Variables**
- A method will often store its temporary state in *local variables*.
- The syntax for declaring a local variable is simple (for example, int count = 0;).
- Variable visibility is where variable is declared — which is between the opening and closing braces of a method.
- Local variables are only visible to the methods in which they are declared; they are not accessible from the rest of the class.

Java Language Basics
Variables

- **Local Variables**
- The syntax for declaring a local variable is simple (for example, int count = 0;).
- Variables has two parts:
- The data type of the variable: which types of data values can be stored.
- Name of the variable: name used for the variable in the program

Java Language Basics
Naming of Variables

- Variable names are case-sensitive.
- A variable's name can be any legal identifier — an unlimited-length sequence of Unicode letters and digits, beginning with a letter, the dollar sign "$", or the underscore character "_". White space is not permitted.

Java Language Basics

Data Types of Variables

- A variable's data type determines the values it may contain, plus the operations that may be performed on it.
- A primitive type is predefined by the language and is named by a reserved keyword.
- Java programming language supports eight *primitive data types*.

Java Language Basics

Data Types of Variables

Eight *primitive data types*

- For storing Integer: Byte, Short, Int, Long
- For storing real: Float, Double
- For storing characters: char
- For true/false: Boolean

Java Language Basics
Data Types of Variables

int

- The int data type is a 32-bit signed two's complement integer. It has a minimum value of -2,147,483,648 and a maximum value of 2,147,483,647 (inclusive).
- For integral values, this data type is generally the default choice.

Java Language Basics

Data Types of Variables

double

- The double data type is a double-precision 64-bit IEEE 754 floating point.
- For decimal values, this data type is generally the default choice.

Java Language Basics
Data Types of Variables

boolean

- The boolean data type has only two possible values: true and false.
- Use this data type for simple flags that track true/false conditions.
- This data type represents one bit of information, but its "size" isn't something that's precisely defined.

Java Language Basics
Data Types of Variables

char

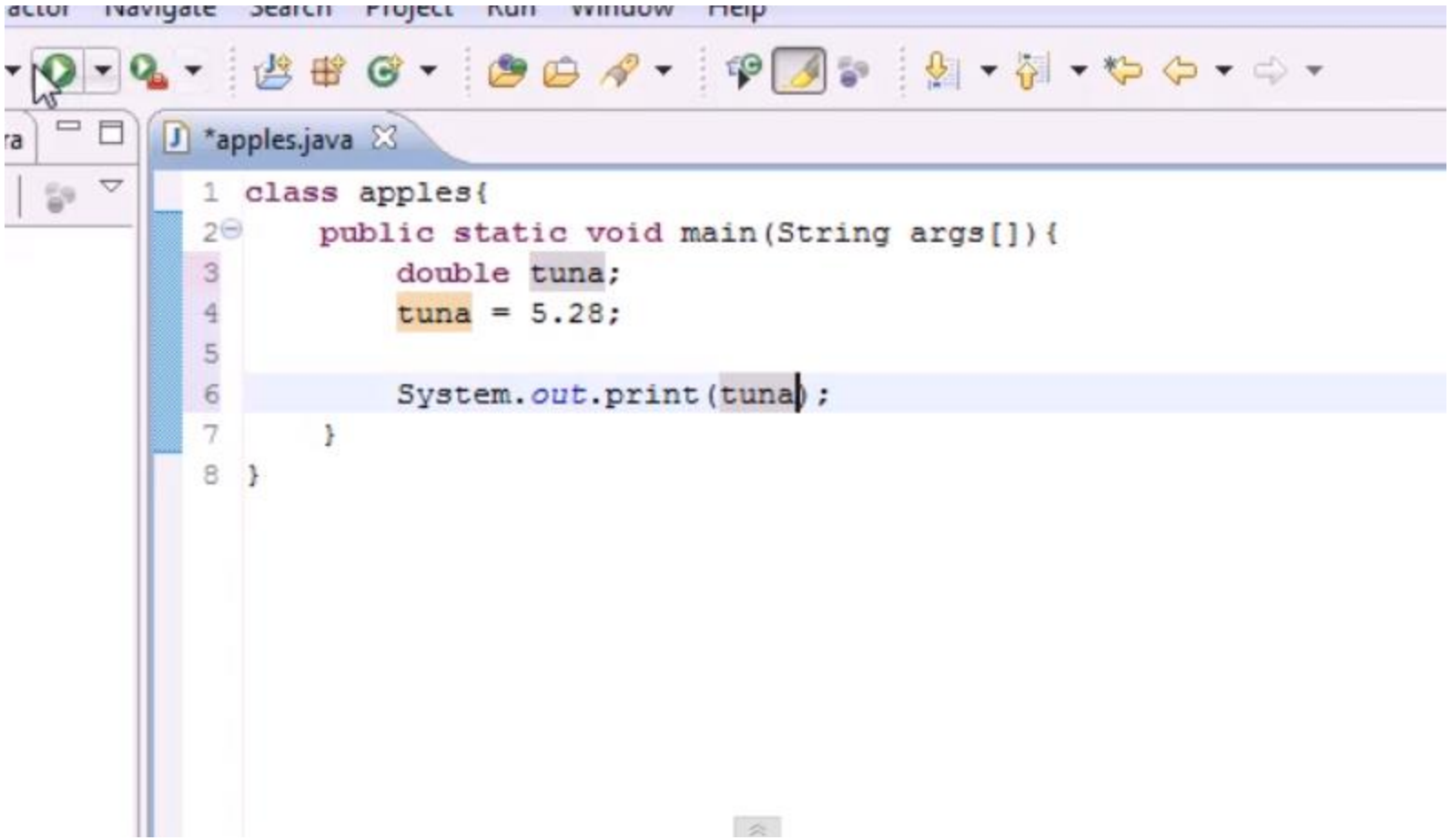
- The char data type is a single 16-bit Unicode character. It has a minimum value of '\u0000' (or 0) and a maximum value of '\uffff' (or 65,535 inclusive).
- Used for storing characters, e.g., ‘A’, ‘E’, ‘2’, etc.

Introduction to variables in Java

*apples.java

```
1  public class apples
2  {
3      public static void main(String args[])
4      {
5          int val;
6          val = 30;
7          System.out.println(val);
8
9      }
10
11 }
12
```

Introduction to variables in Java



Introduction to variables in Java

apples.java

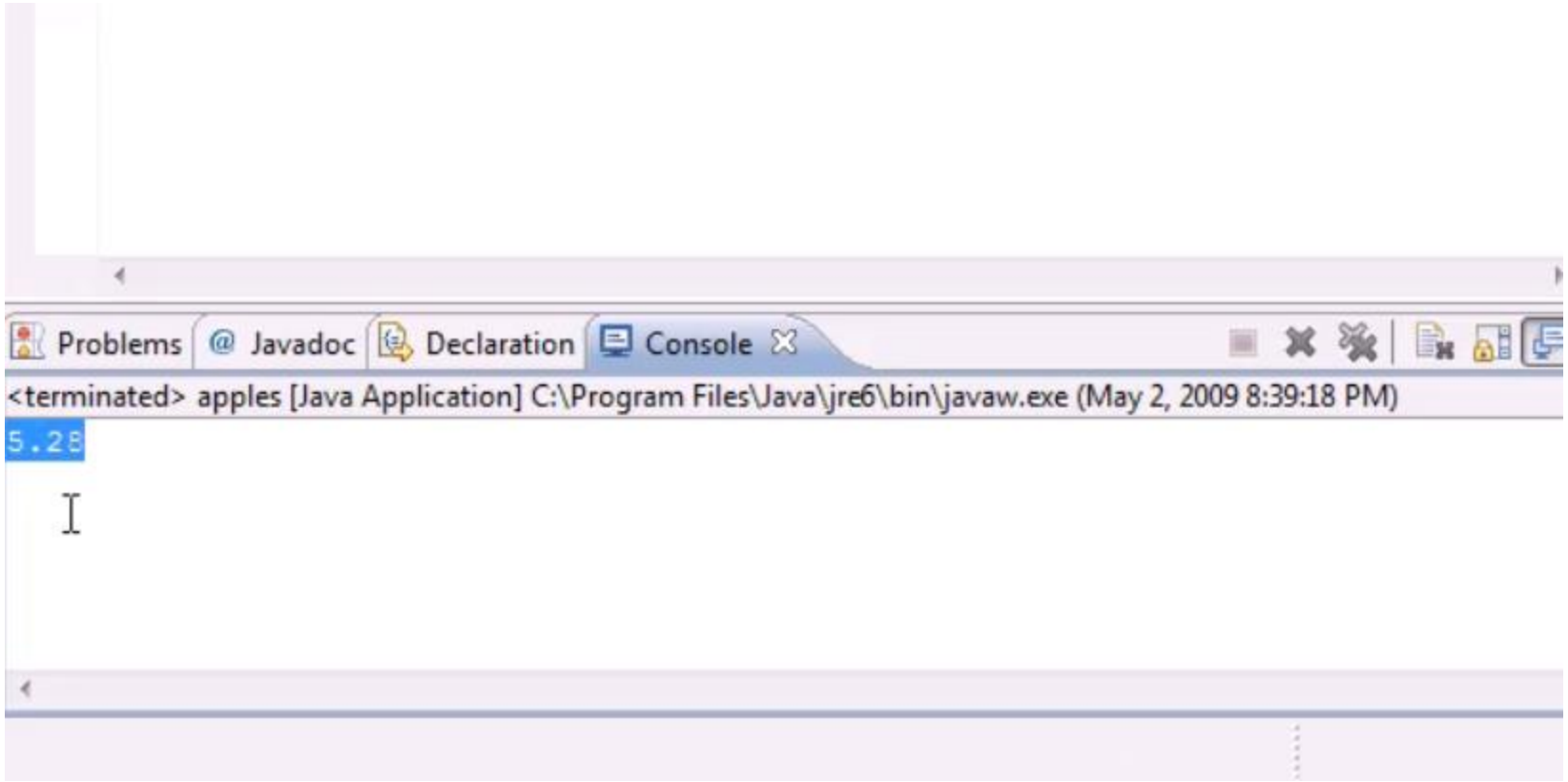
```java
public class apples
{
    public static void main(String args[])
    {
        char ch;
        ch = 'A';
        System.out.println(ch);

    }

}
```

Introduction to variables in Java

apples.java

```java
public class apples
{
    public static void main(String args[])
    {
        boolean b;
        b = true;
        System.out.println(b);

    }

}
```

View output

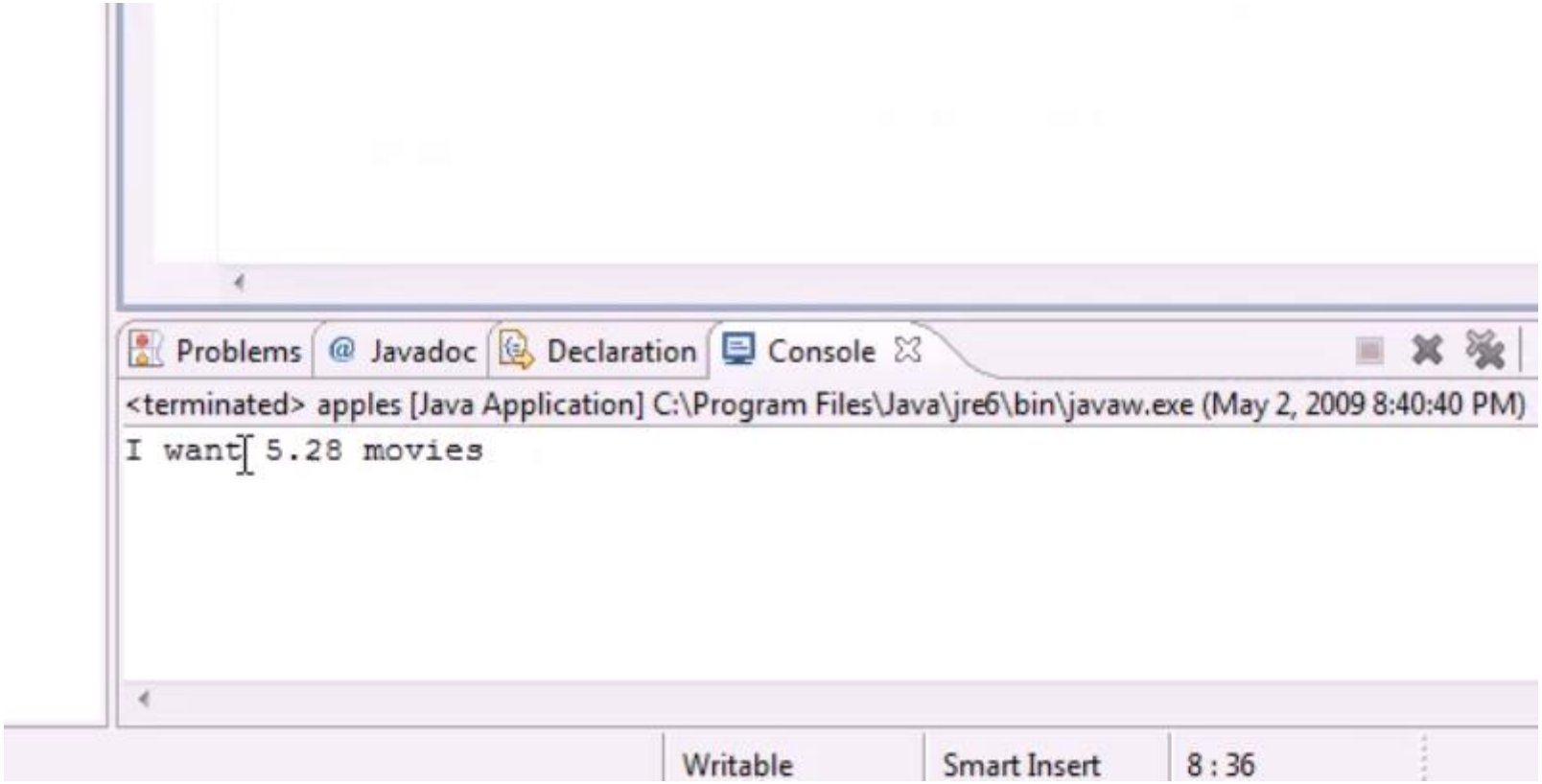

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 2, 2009 8:39:18 PM)
5.28

Print and println in Java

*apples.java

```
1  class apples{
2      public static void main(String args[]){
3          double tuna;
4          tuna = 5.28;
5
6          System.out.print("I want ");
7          System.out.print(tuna);
8          System.out.println(" movies");
9      }
10 }
```

View output changes



Add another print

*apples.java

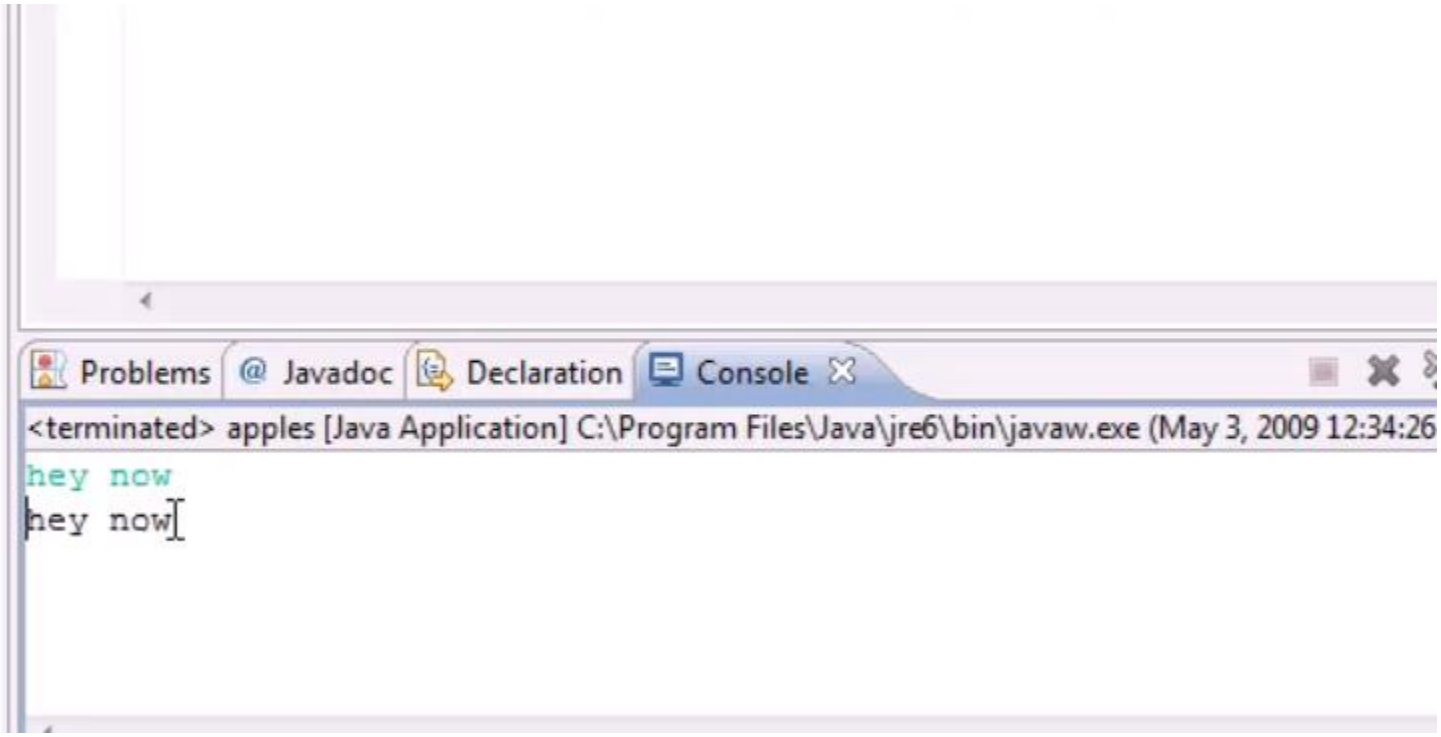
```
1  class apples{
2      public static void main(String args[]){
3          double tuna;
4          tuna = 5.28;
5
6          System.out.print("I want ");
7          System.out.print(tuna);
8          System.out.println(" movies");
9          System.out.print("apples");
10     }
11 }
```

Getting user input in Java

*apples.java

```
1 import java.util.Scanner;
2
3 class apples{
4     public static void main(String args[]){
5         Scanner bucky = new Scanner(System.in);
6         System.out.println(bucky.nextLine());
7     }
8 }
```

View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 12:34:26
hey now
hey now

A basic calculator in Java

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);
        double fnum, snum, answer;
        System.out.println("Enter first num: ");
        fnum = bucky.nextDouble();
        System.out.println("Enter second num: ");
        snum = bucky.nextDouble();
        answer = fnum + snum;
        System.out.println(answer);
    }
}
```

View output

Problems | Javadoc | Declaration | Console

```
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 2:22:43 AM)
34.6
Enter second num:
54.2
88.80000000000001
```

Java Language Basics

Operators

- Operators are special symbols that perform specific operations on one, two, or three *operands*, and then return a result.

Java Language Basics

Operators

The Arithmetic Operators

- + additive operator, e.g, 3 + 9
- - subtraction operator, e.g, 3 - 9
- * multiplication operator, e.g, 3 * 9
- / division operator, e.g, 9 / 3
- % remainder operator, e.g, 9 % 3

Java Language Basics

Operators

The Simple Assignment Operator

- One of the most common operators that you'll encounter is the simple assignment operator "=".
- it assigns the value on its right to the operand on its left:
  - int cadence = 0;
  - int speed = 0;
  - int gear = 1;

Java Language Basics

Operators

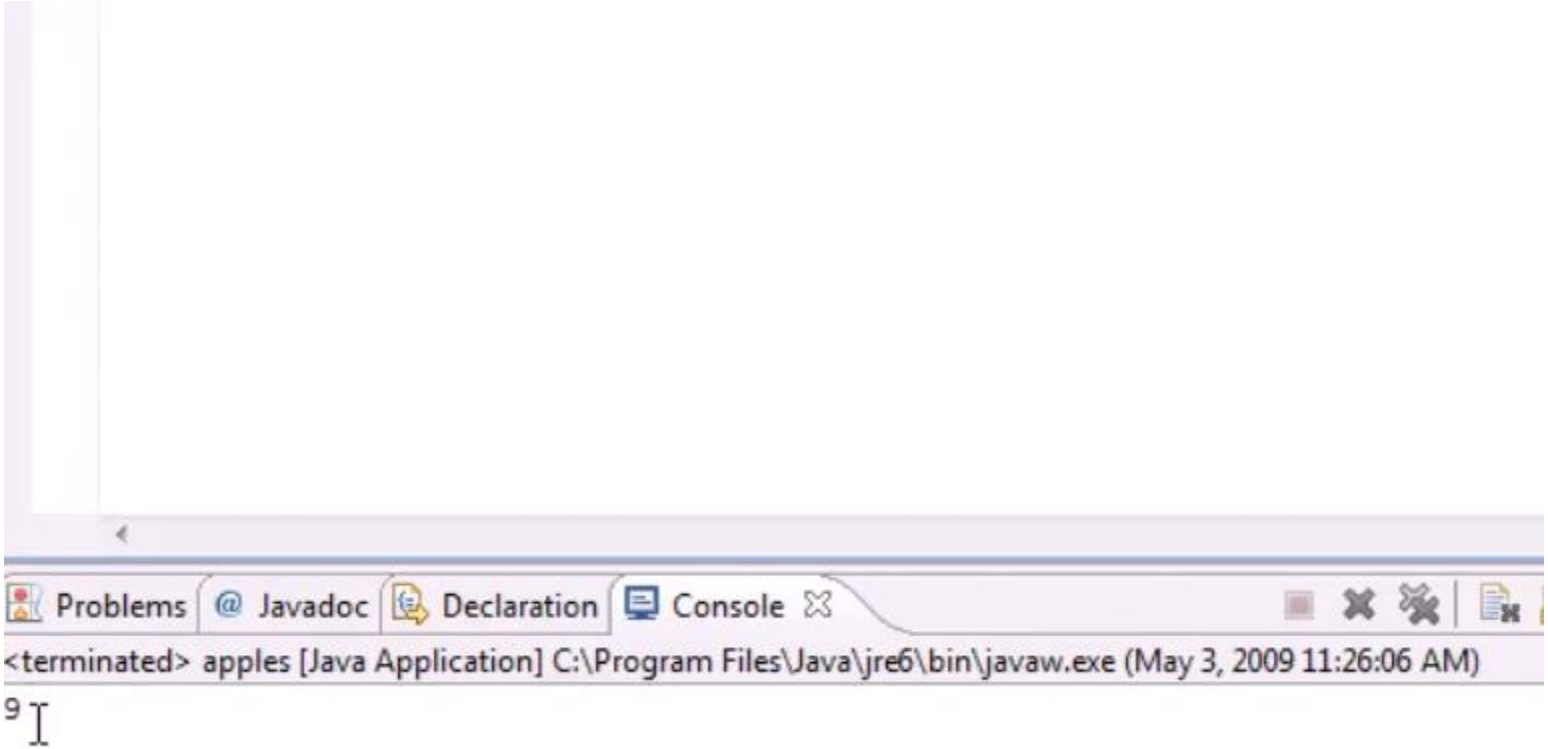
- **The Increment/Decrement Operator**
- ++ Increments value by one
  - int speed = 0;
  - ++x
- -- Decrements value by one
  - int speed = 0;
  - --x

Basic math operators in Java: plus

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int girls, boys, people;
        girls = 6;
        boys = 3;
        people = girls + boys;
        System.out.println(people);
    }
}
```

View output

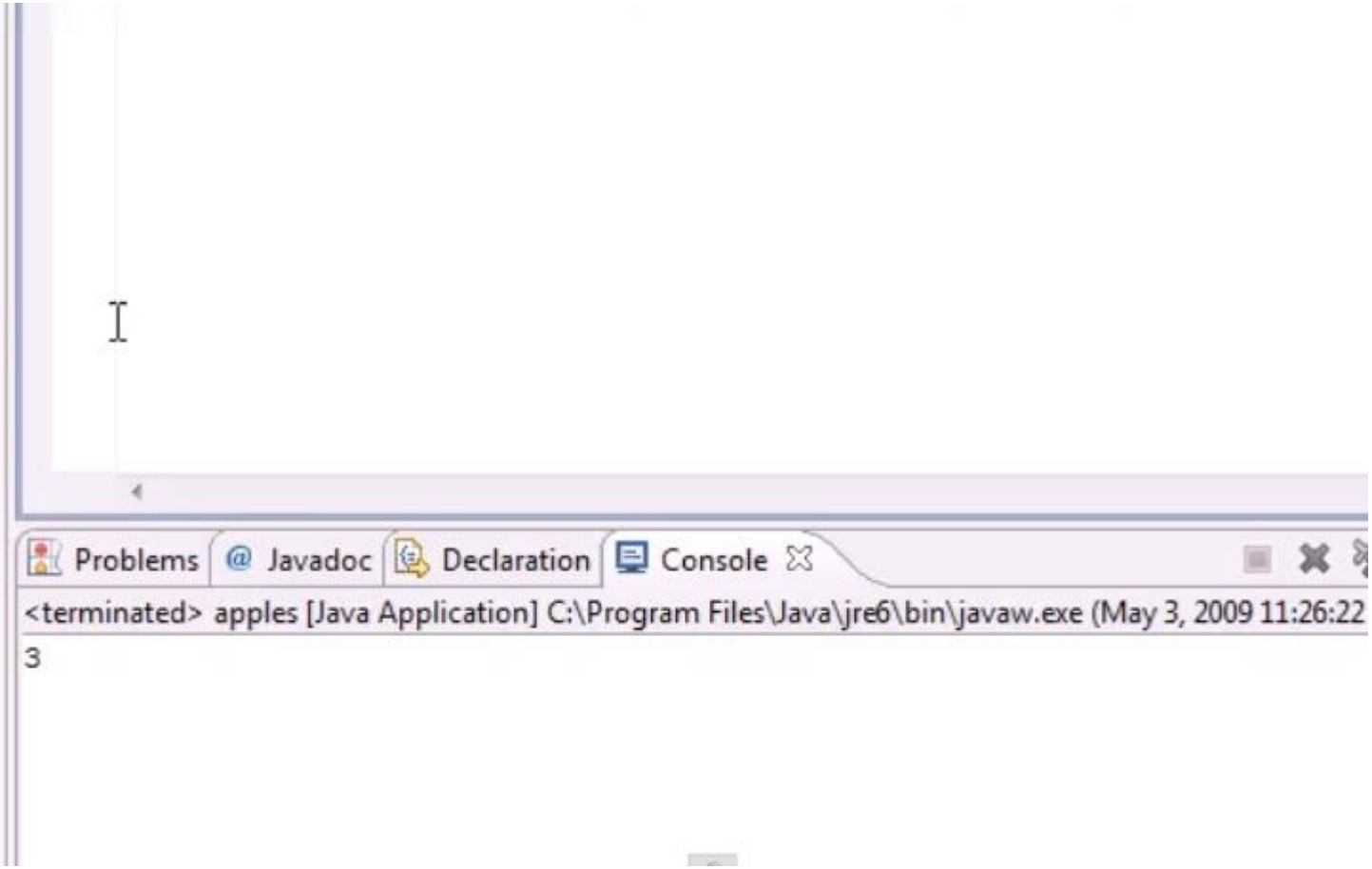

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:26:06 AM)
9

Basic math operators in Java: minus

*apples.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String args[]){
5          Scanner bucky = new Scanner(System.in);
6
7          int girls, boys, people;
8          girls = 6;
9          boys = 3;
10         people = girls - boys;
11         System.out.println(people);
12     }
13 }
```

View output

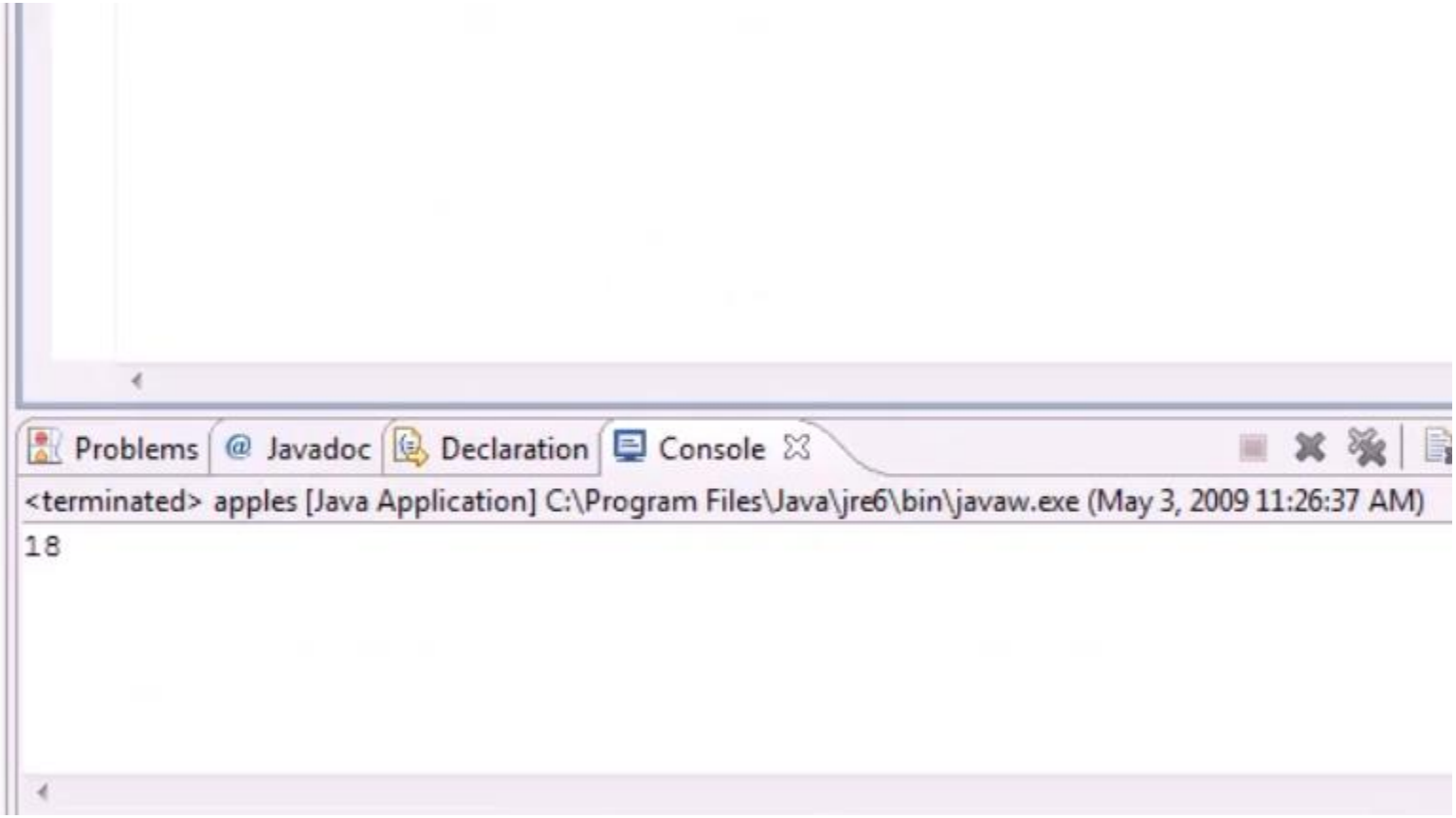

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:26:22
3

Basic math operators in Java: multiply

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int girls, boys, people;
        girls = 6;
        boys = 3;
        people = girls * boys;
        System.out.println(people);
    }
}
```

View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:26:37 AM)
18

Basic math operators in Java: division

*apples.java

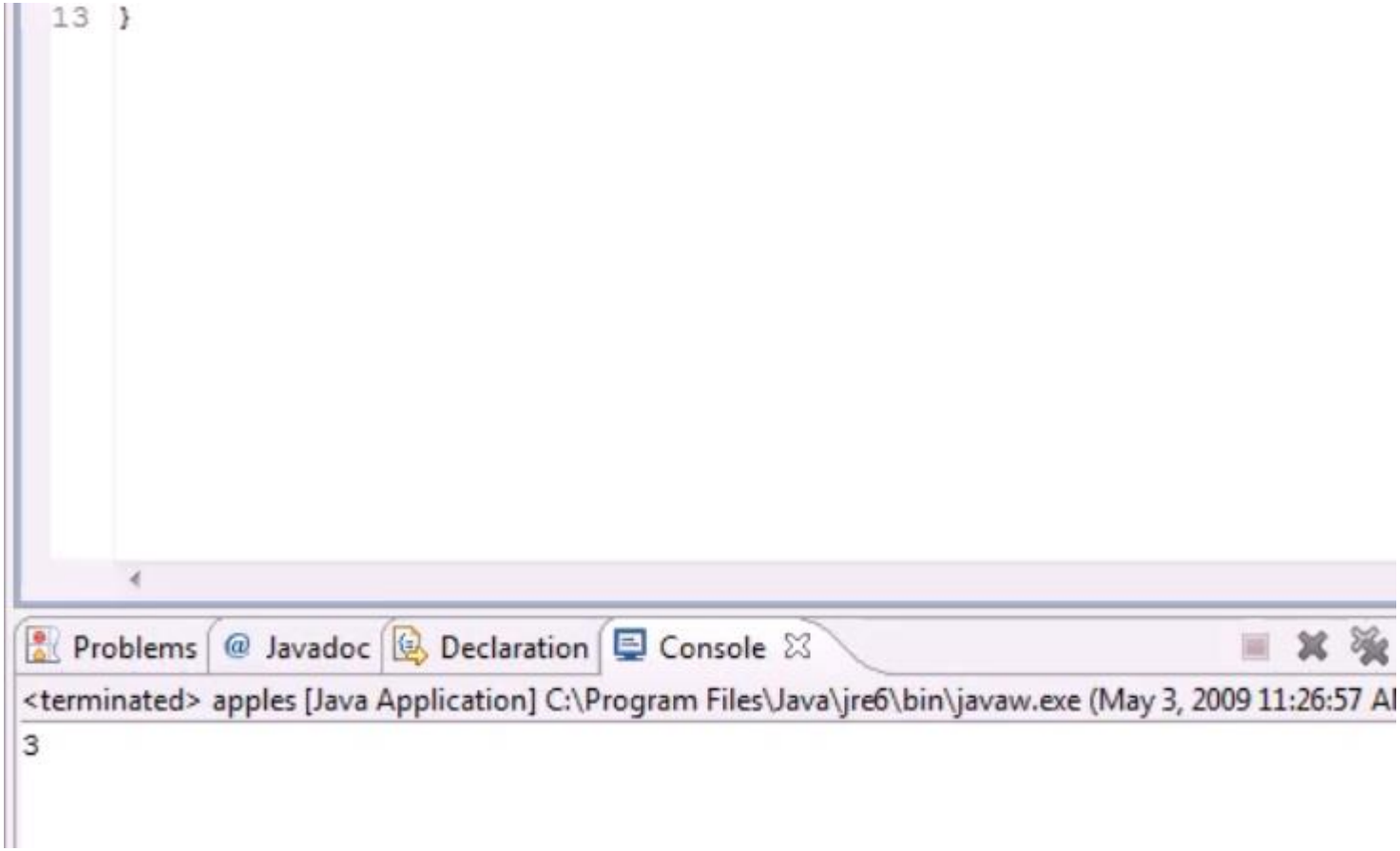
```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int girls, boys, people;
        girls = 6;
        boys = 3;
        people = girls / boys;
        System.out.println(people);
    }
}
```

Basic math operators in Java: division

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int girls, boys, people;
        girls = 11;
        boys = 3;
        people = girls / boys;
        System.out.println(people);
    }
}
```

View output


13 }
Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:26:57 A
3

Basic math operators in Java

apples.java

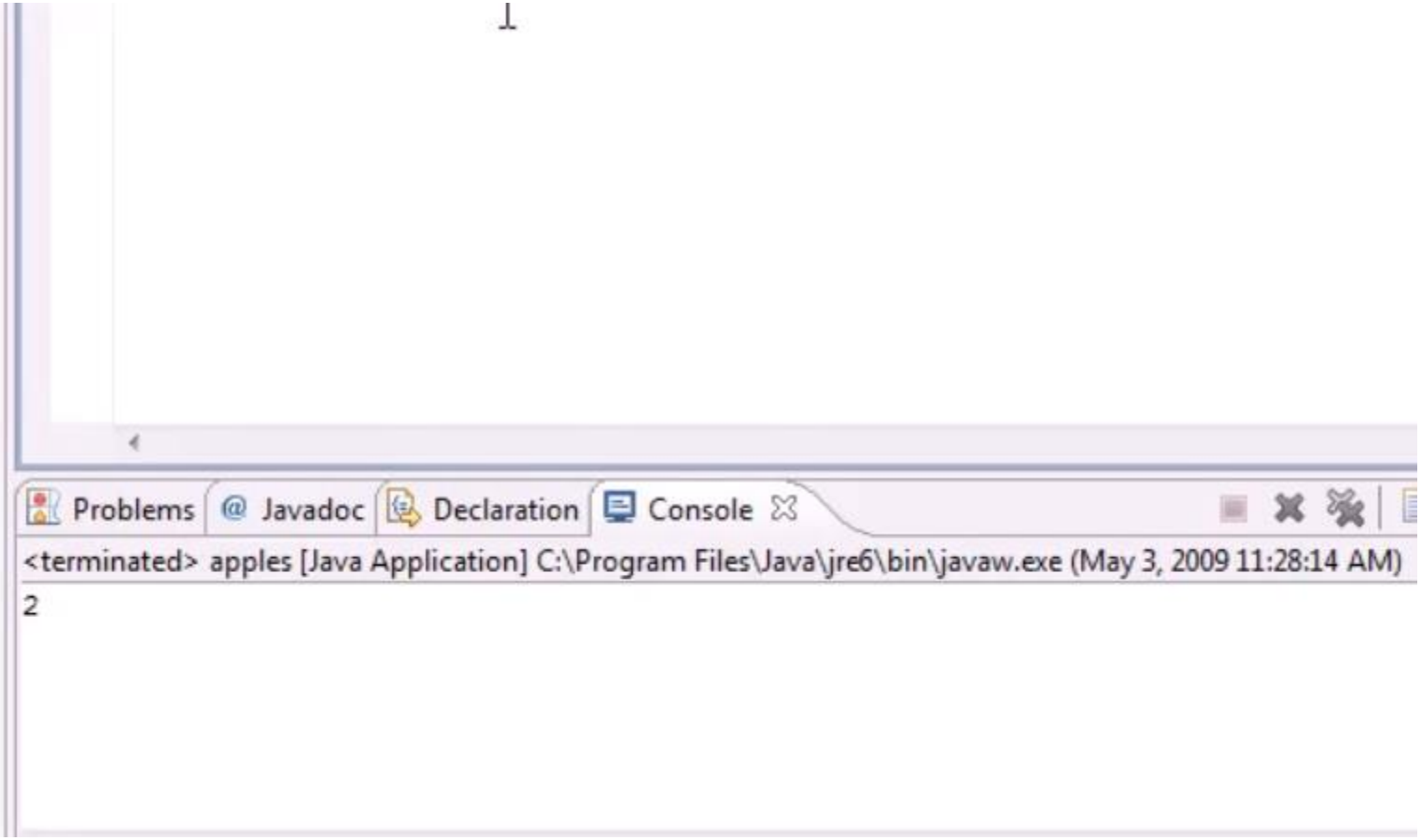
```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        double girls, boys, people;
        girls = 11;
        boys = 3;
        people = girls / boys;
        System.out.println(people);
    }
}
```

Basic math operators in Java: remainder

*apples.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String args[]){
5          Scanner bucky = new Scanner(System.in);
6
7          int girls, boys, people;
8          girls = 11;
9          boys = 3;
10         people = girls % boys;
11         System.out.println(people);
12     }
13 }
```

View output

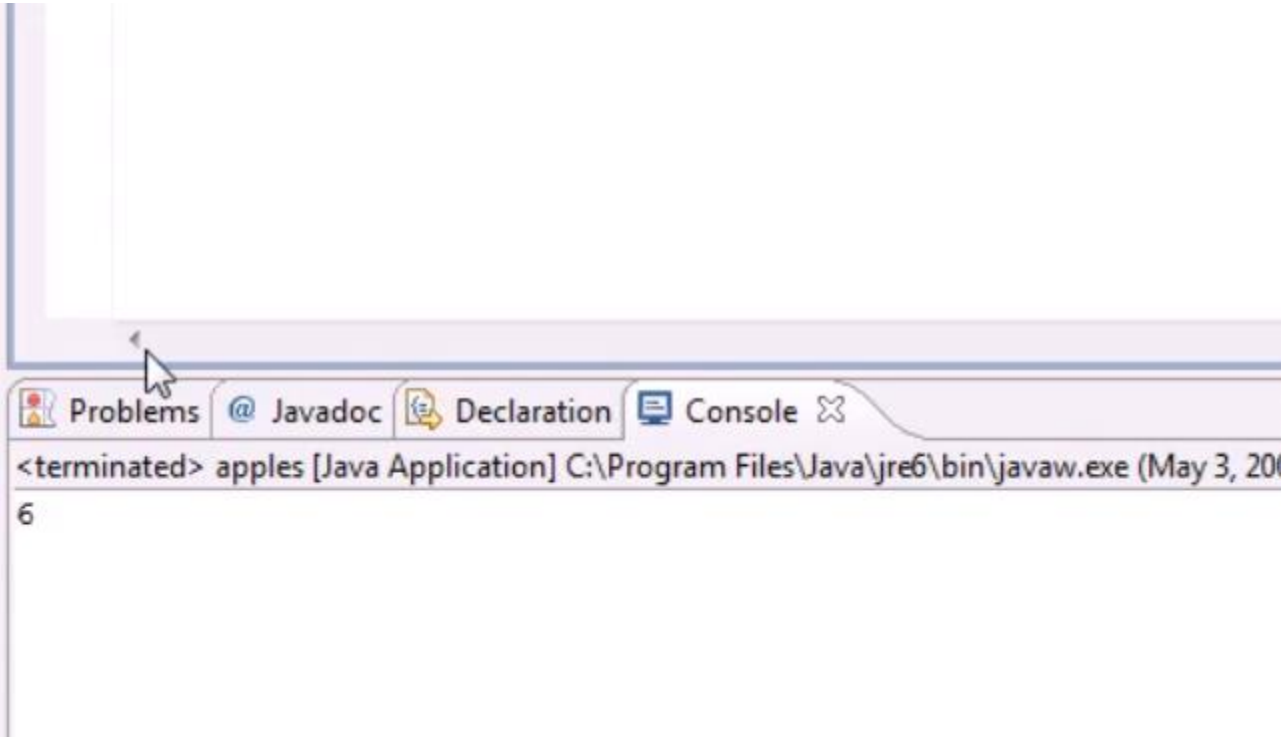

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:28:14 AM)
2

Incr/Decr operators in Java

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int tuna = 5;
        int bass = 18;
        ++tuna;
        System.out.println(tuna);
    }
}
```

View output

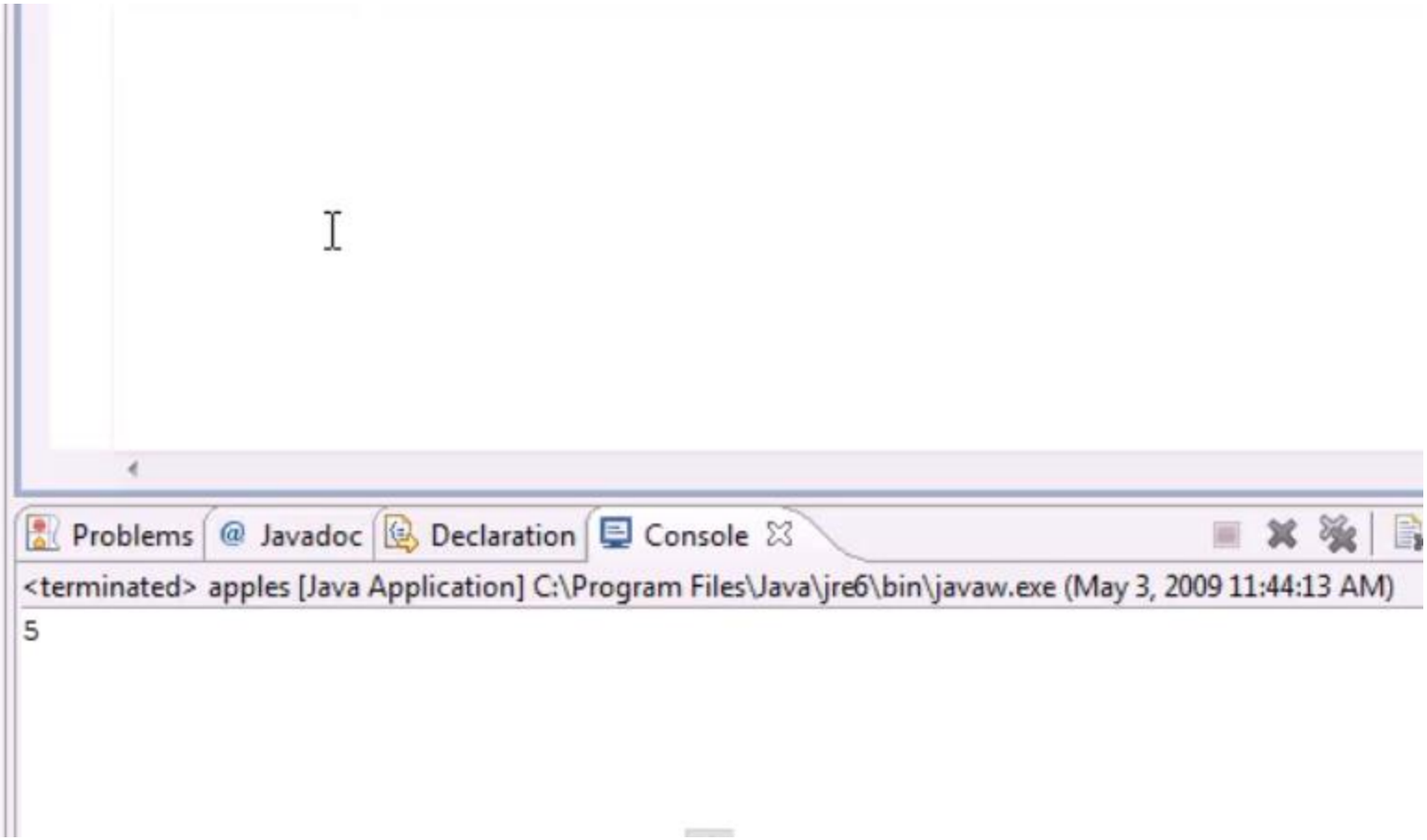

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 20
6

Post increment operator

*apples.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String args[]){
5          Scanner bucky = new Scanner(System.in);
6
7          int tuna = 5;
8          int bass = 18;
9          System.out.println(tuna++);
10     }
11 }
```

View output

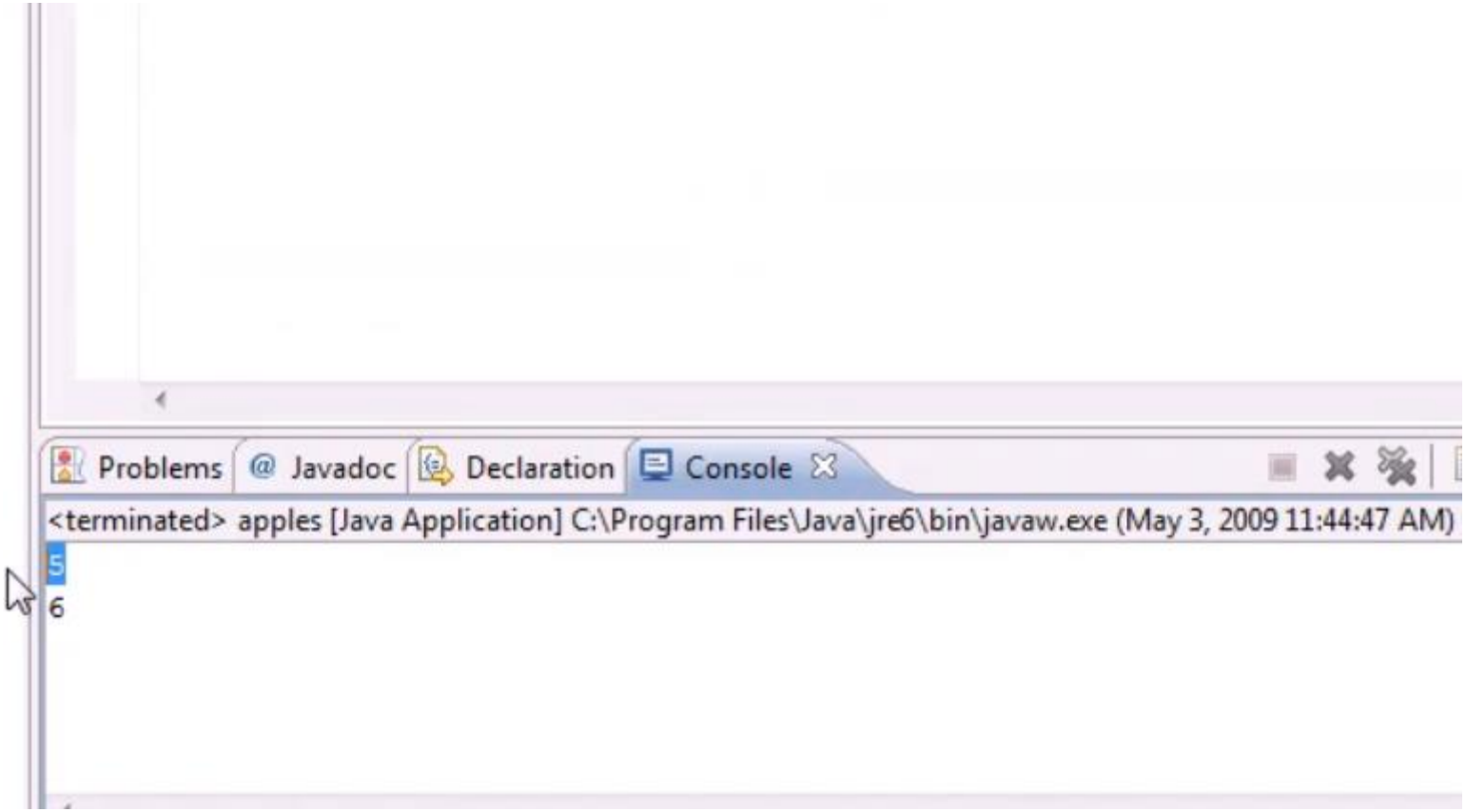

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:44:13 AM)
5

Post increment operator (more)

*apples.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String args[]){
5          Scanner bucky = new Scanner(System.in);
6
7          int tuna = 5;
8          int bass = 18;
9          System.out.println(tuna++);
10         System.out.println(tuna);
11     }
12 }
```

View output

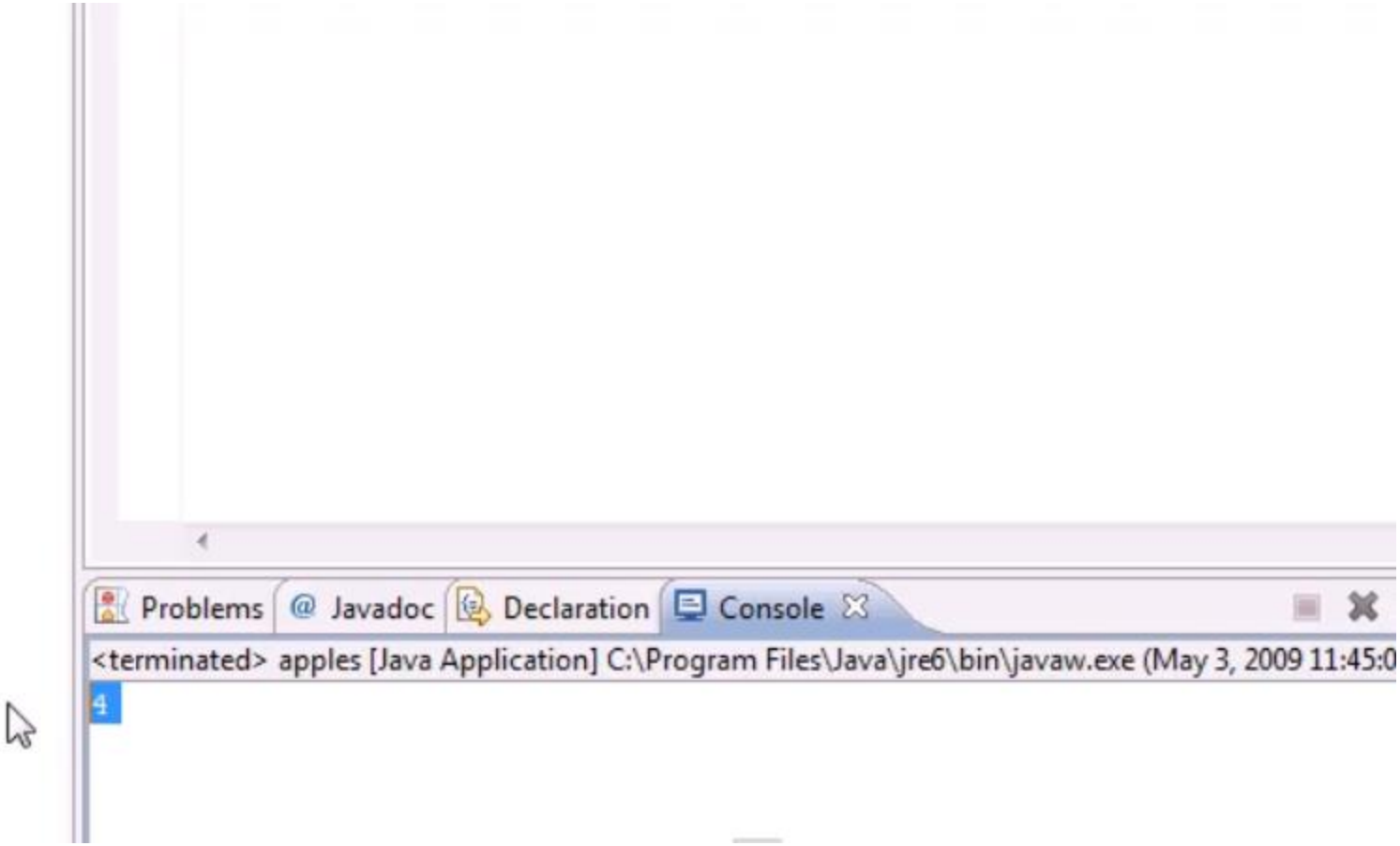

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:44:47 AM)
5
6

Pre decrement operator

*apples.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String args[]){
5          Scanner bucky = new Scanner(System.in);
6
7          int tuna = 5;
8          int bass = 18;
9          System.out.println(--tuna);
10
11     }
12 }
```

View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:45:0
4

Assignment operator in Java

apples.java

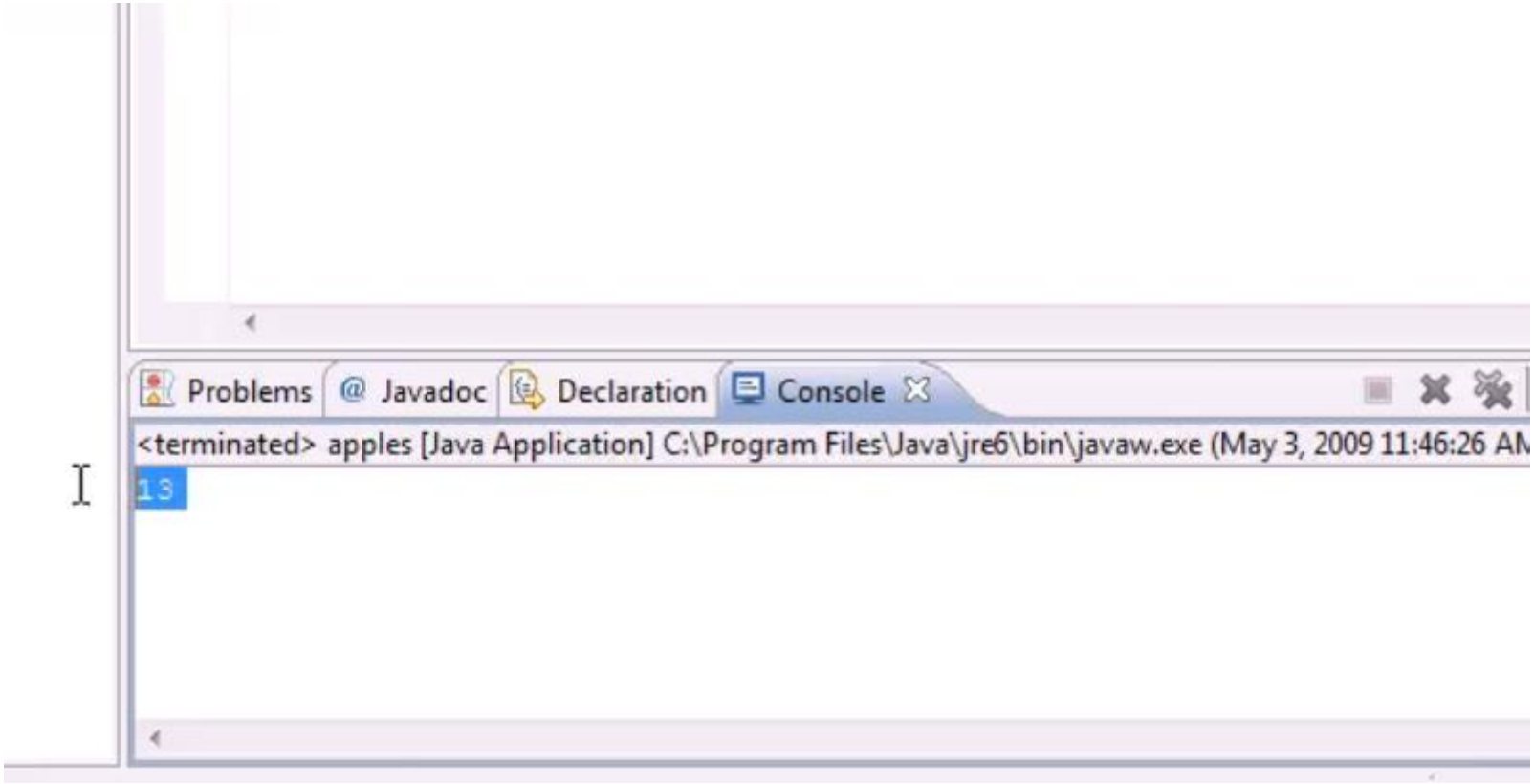
```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String args[]){
5          Scanner bucky = new Scanner(System.in);
6
7          int tuna = 5;
8          int bass = 18;
9
10         tuna = tuna + 5;
11
12         System.out.println(tuna);
13
14     }
15 }
```

Composite assignment operator: +=

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int tuna = 5;
        int bass = 18;

        tuna += 8;

        System.out.println(tuna);

    }
}
```

View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:46:26 AM
13

Composite assignment operators: *=

*apples.java

```java
import java.util.Scanner;

class apples{
    public static void main(String args[]){
        Scanner bucky = new Scanner(System.in);

        int tuna = 5;
        int bass = 18;

        tuna *= 8;

        System.out.println(tuna);

    }
}
```

View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 11:46:54 AM
40

Control Flow Statements

- The statements inside your source files are generally executed from top to bottom, in the order that they appear.
- *Control flow statements*, however, break up the flow of execution by employing decision making, looping, and branching, enabling your program to *conditionally* execute particular blocks of code.

Control Flow Statements

Control flow statements include-

- decision-making statements (if-then, if-then-else, switch)
- the looping statements (for, while, do-while),
- the branching statements (break, continue, return)

The if-then and if-then-else Statements

The if-then Statement

- It tells your program to execute a certain section of code *only if* a particular test evaluates to true.

```
void applyBrakes() {
    // the "if" clause: bicycle must be moving
    if (isMoving){
        // the "then" clause: decrease current speed
        currentSpeed--;
    }
}
```

The if-then and if-then-else Statements

The if-then-else Statement

- The if-then-else statement provides a secondary path of execution when an "if" clause evaluates to false.

```
void applyBrakes() {
    if (isMoving) {
        currentSpeed--;
    } else {
        System.err.println("The bicycle has " + "already stopped!");
    }
}
```

The if-then and if-then-else Statements

The if-then-else Statement

- The if-then-else statement provides a secondary path of execution when an "if" clause evaluates to false.

```
void applyBrakes() {
    if (isMoving) {
        currentSpeed--;
    } else {
        System.err.println("The bicycle has " + "already stopped!");
    }
}
```

The if-then and if-then-else Statements

The switch Statement

- Unlike if-then and if-then-else statements, the switch statement can have a number of possible execution paths.
- A switch works with the byte, short, char, and int primitive data types.

The if-then and if-then-else Statements

The Equality and Relational Operators

- The equality and relational operators determine if one operand is greater than, less than, equal to, or not equal to another operand.
- They are used in testing conditions.

```
==        equal to
!=        not equal to
>         greater than
>=        greater than or equal to
<         less than
<=        less than or equal to
```

If-else statement in Java

*apples.java

```
1
2  class apples{
3      public static void main(String args[]){
4          int test = 6;
5
6          if (test == 9){
7              System.out.println("yes");
8          }else{
9              System.out.println("this is else");
10         }
11     }
12 }
```

If-else statement in Java

apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int test = 6;
5
6         if (test < 9){
7             System.out.println("yes");
8         }else{
9             System.out.println("this is else");
10        }
11    }
12 }
```

If-else statement in Java

*apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int test = 6;
5
6         if (test != 9)
7     }
8 }
```

If-else statement in Java

*apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int test = 6;
5
6         if (test < 9)
7     }
8 }
```

If-else statement in Java

*apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int test = 6;
5
6         if (test > 9)
7     }
8 }
```

If-else statement in Java

*apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int test = 6;
5
6         if (test <= 9)
7     }
8 }
```

If-else statement in Java

*apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int test = 6;
5
6         if (test >= 9)
7     }
8 }
```

Logical operators in Java

*apples.java

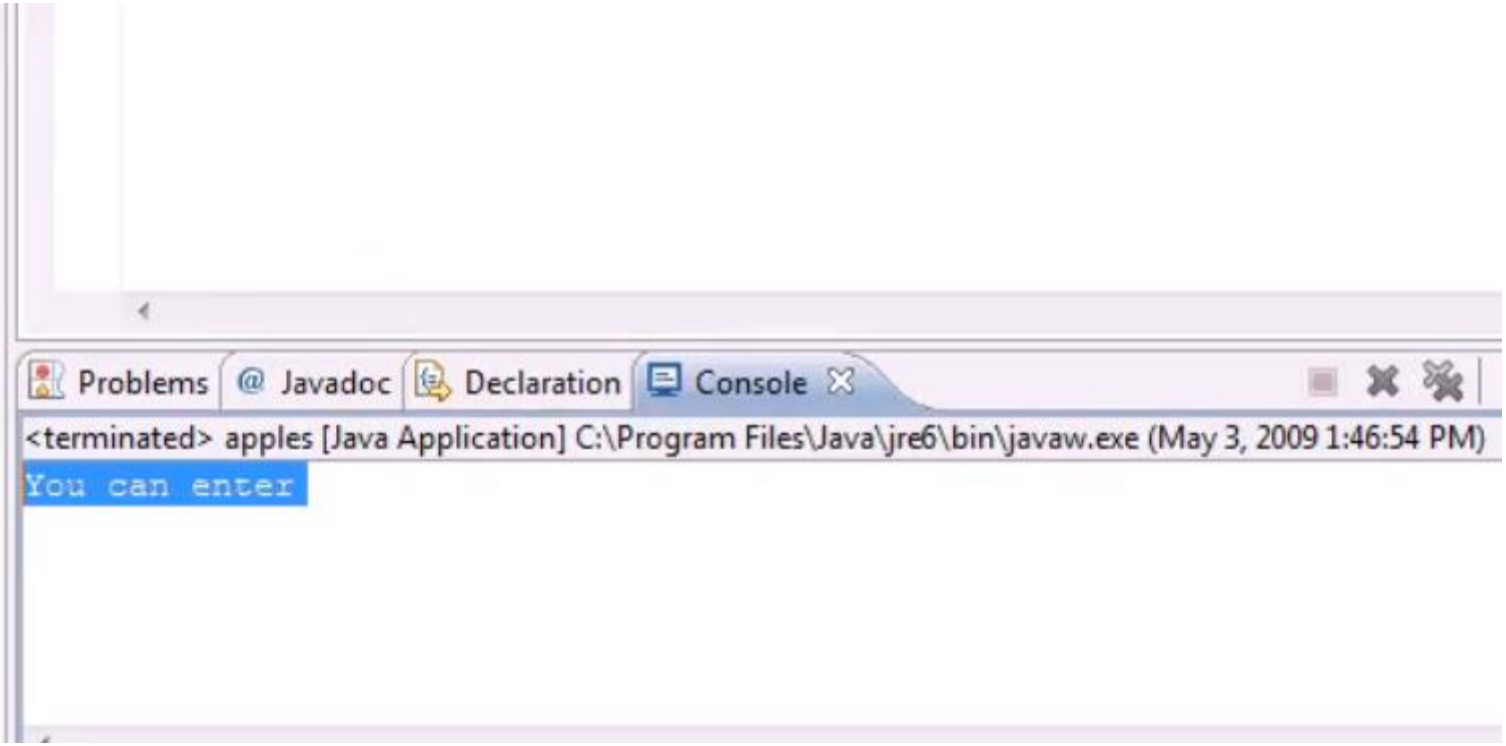
```
1
2 class apples{
3     public static void main(String args[]){
4         int boy, girl;
5         boy = 18;
6         girl = 68;
7
8         if(boy > 10){
9             System.out.println("You can enter");
10        }else{
11            System.out.println("You are too young");
12        }
13    }
14 }
```

Logical operators in Java: &&

*apples.java

```
1
2  class apples{
3      public static void main(String args[]){
4          int boy, girl;
5          boy = 18;
6          girl = 40;
7
8          if(boy > 10 &&  girl < 60){
9              System.out.println("You can enter");
10         }else{
11             System.out.println("You can not enter");
12         }
13     }
14 }
```

View output

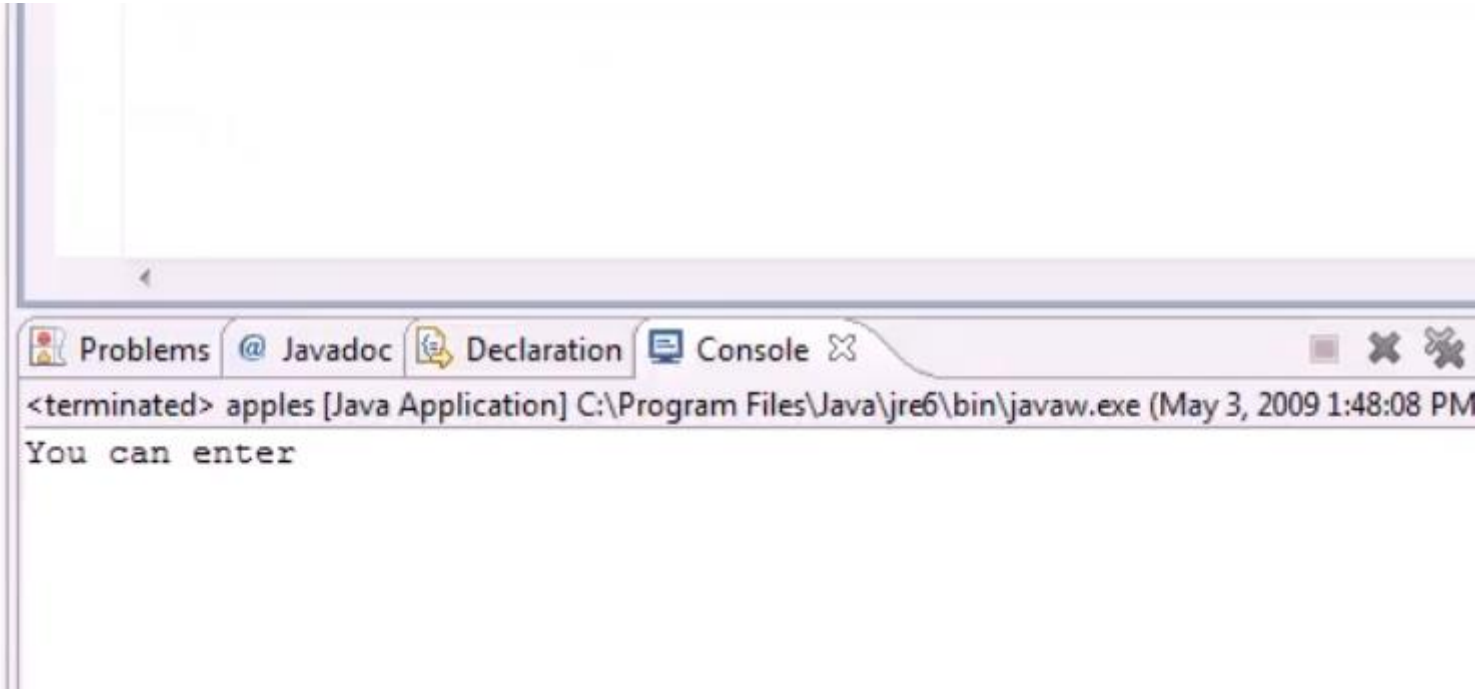

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 1:46:54 PM)
You can enter

Logical operators in Java: ||

*apples.java

```
1
2  class apples{
3      public static void main(String args[]){
4          int boy, girl;
5          boy = 18;
6          girl = 99;
7
8          if(boy > 10 ||  girl < 60){
9              System.out.println("You can enter");
10         }else{
11             System.out.println("You can not enter");
12         }
13     }
14 }
```

View output

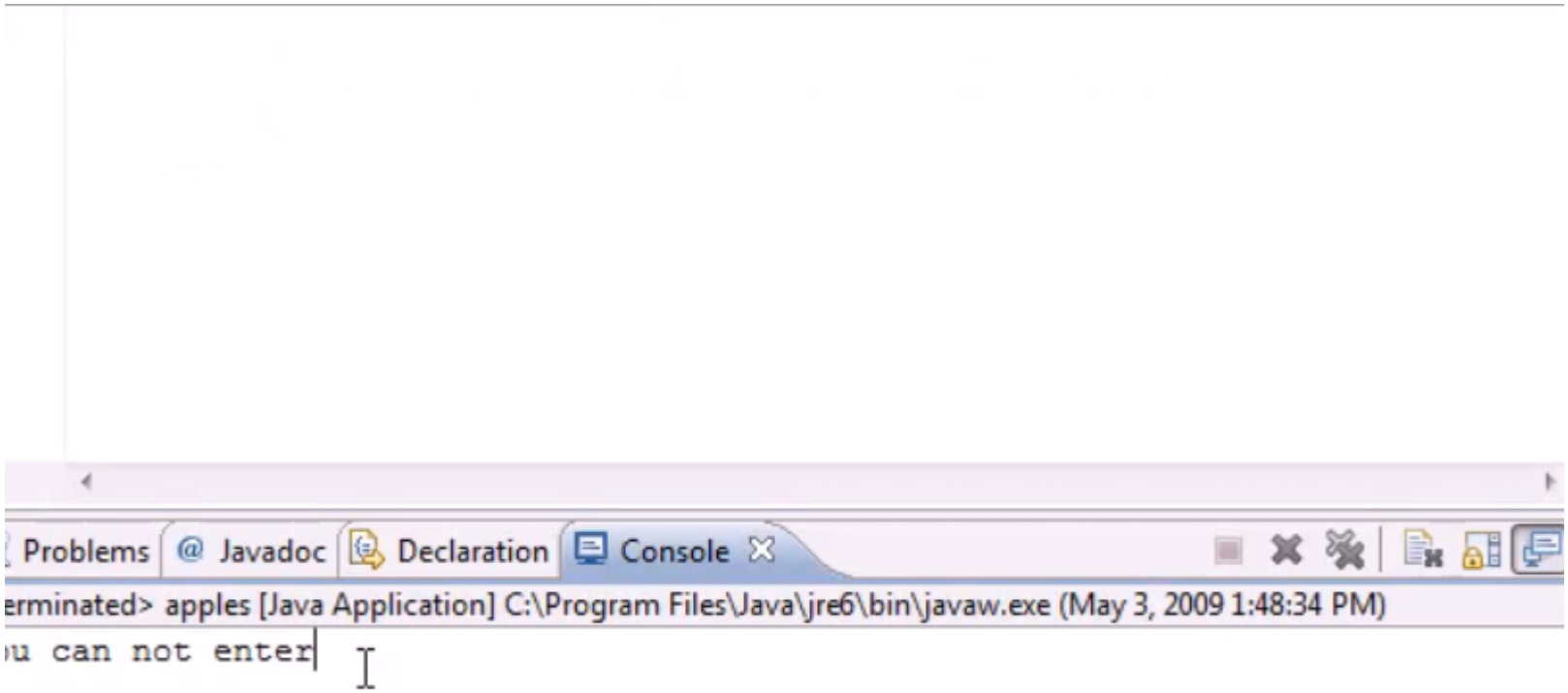

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 1:48:08 PM
You can enter

Logical operators in Java: ||

*apples.java

```
1
2 class apples{
3     public static void main(String args[]){
4         int boy, girl;
5         boy = 8;
6         girl = 99;
7
8         if(boy > 10 ||  girl < 60){
9             System.out.println("You can enter");
10        }else{
11            System.out.println("You can not enter");
12        }
13    }
14 }
```

View output


Problems
Javadoc
Declaration
Console
erminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 1:48:34 PM)
u can not enter

Switch statement in Java

*apples.java

```
1
2  class apples{
3      public static void main(String args[]){
4          int age;
5          age = 3;
6
7          if(age == 1){
8              System.outy.()
9          }
10         if (age ==2){}
11
12     }
13 }
```

Switch statement in Java

*apples.java

```
1
2  class apples{
3      public static void main(String args[]){
4          int age;
5          age = 3;
6
7          switch (age){
8          case 1:
9              System.out.println("You can crawl");
10             break;
11         case 2:
12             System.out.println("You can talk");
13             break;
14         case 3:
15             System.out.println("You can get in toruble");
16             break;
17         default:
18
19
```

Switch statement in Java

```
        switch (age){
        case 1:
            System.out.println("You can crawl");
            break;
        case 2:
            System.out.println("You can talk");
            break;
        case 3:
            System.out.println("You can get in toruble");
            break;
        default:
            System.out.println("I dont know how old you are");
        break;

        }
    }
}
```

View output

```
21            }
22        }
23    }
```

Problems | @ Javadoc | Declaration | Console

```
terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 2:09:57 PM)
You can get in toruble
```

Switch statement in Java

```
class apples{
    public static void main(String args[]){
        int age;
        age = 7;

        switch (age){
        case 1:
            System.out.println("You can crawl");
            break;
        case 2:
            System.out.println("You can talk");
            break;
        case 3:
            System.out.println("You can get in toruble");
            break;
        default:
            System.out.println("I dont know how old you are");
        break;

        }
```

While loop in Java

apples.java

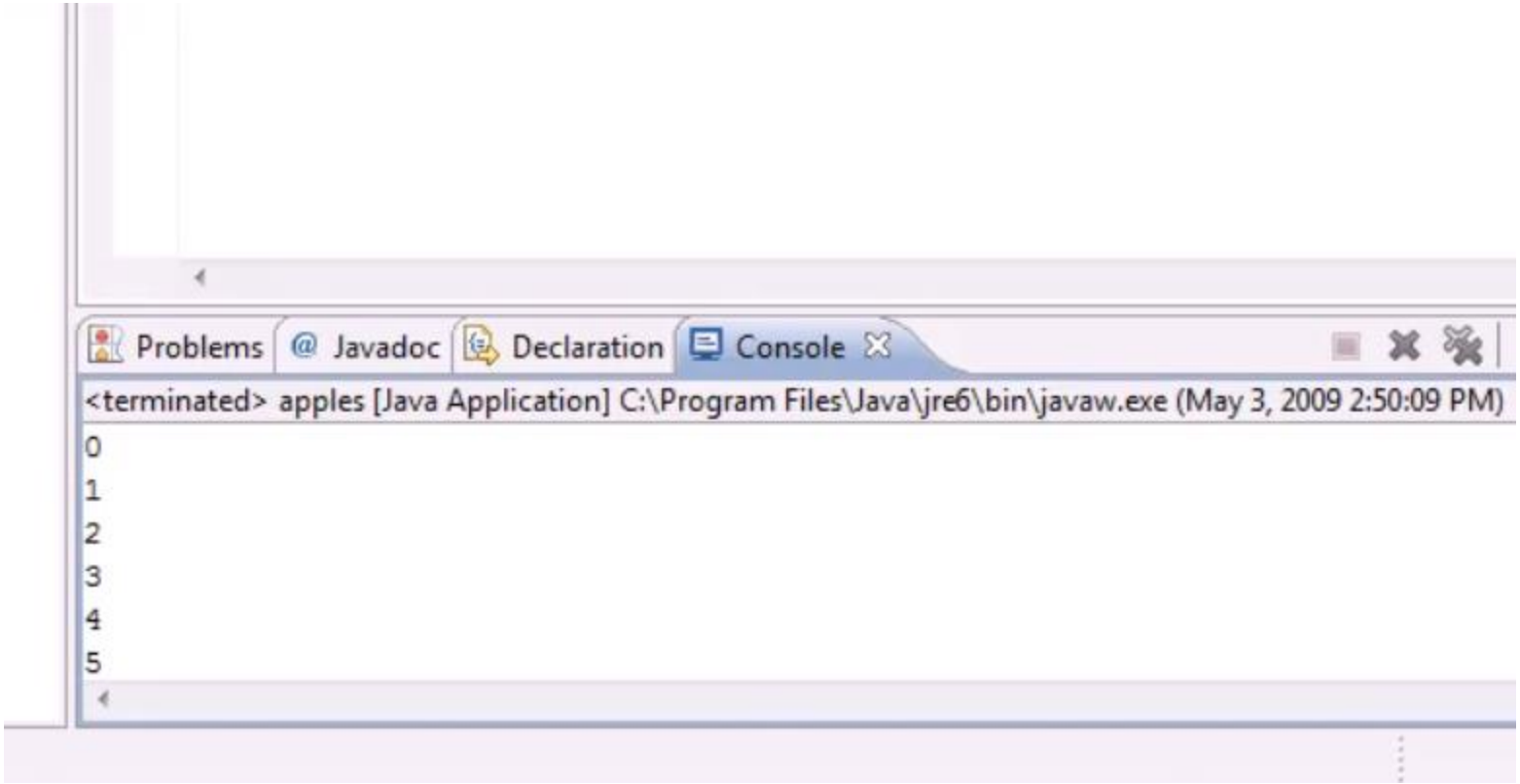
```
class apples{
    public static void main(String args[]){
        int counter = 0;
        while (counter < 10){
            System.out.println(counter);
        }
    }
}
```

While loop in Java

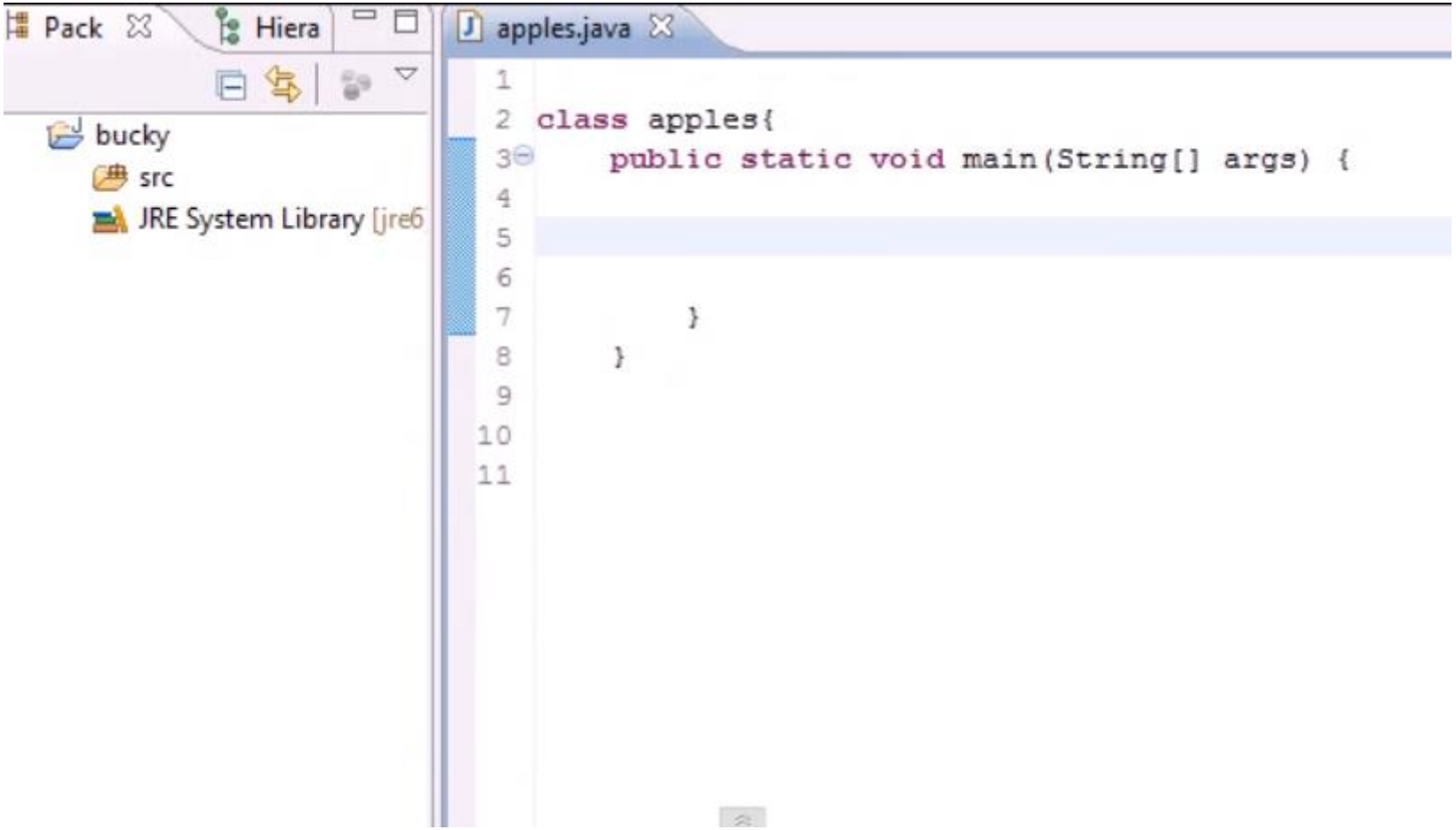
*apples.java

```
1
2  class apples{
3      public static void main(String args[]){
4          int counter = 0;
5          while (counter < 10){
6              System.out.println(counter);
7              counter++;
8          }
9      }
10 }
```

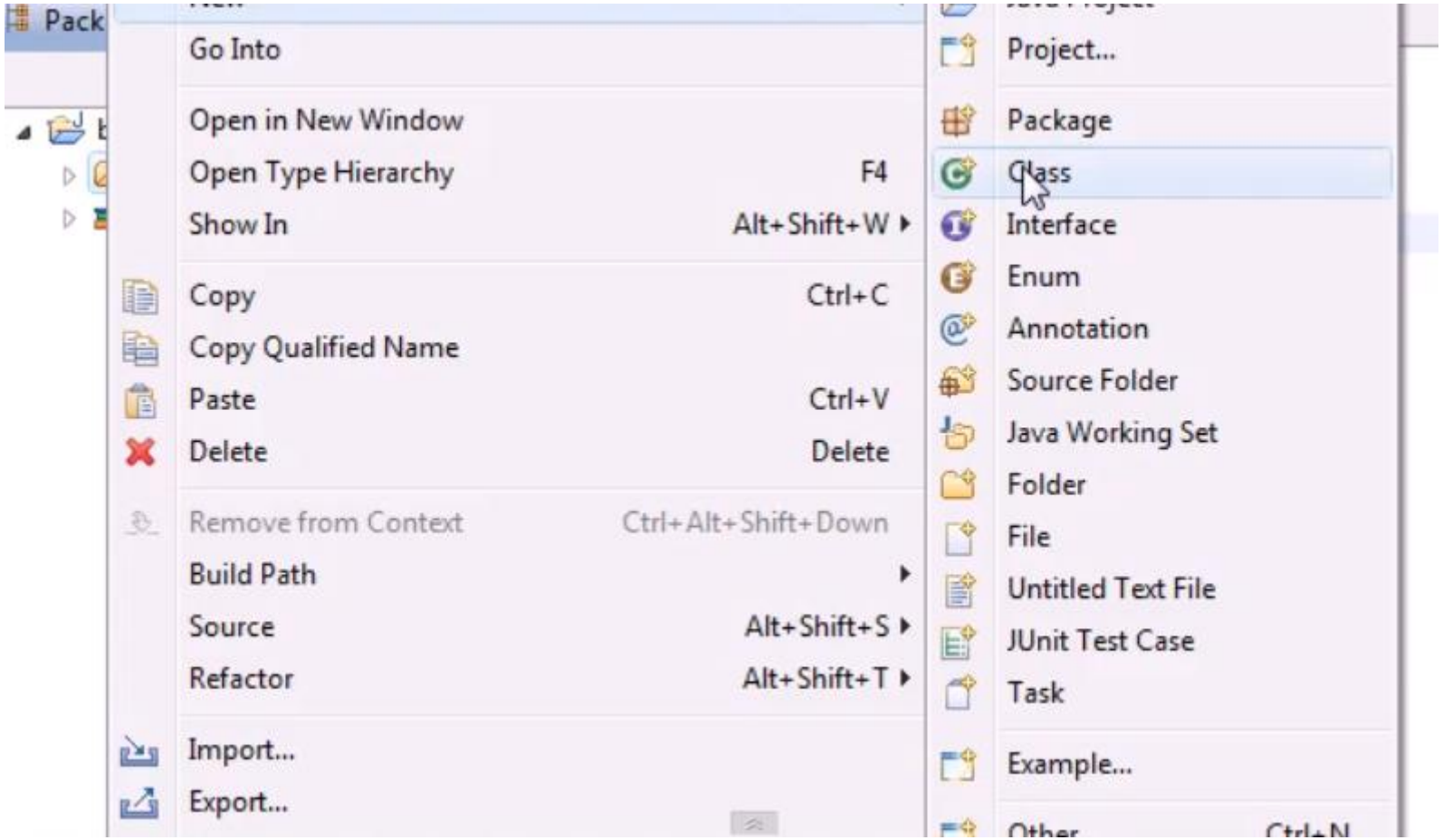
View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 3, 2009 2:50:09 PM)
0
1
2
3
4
5

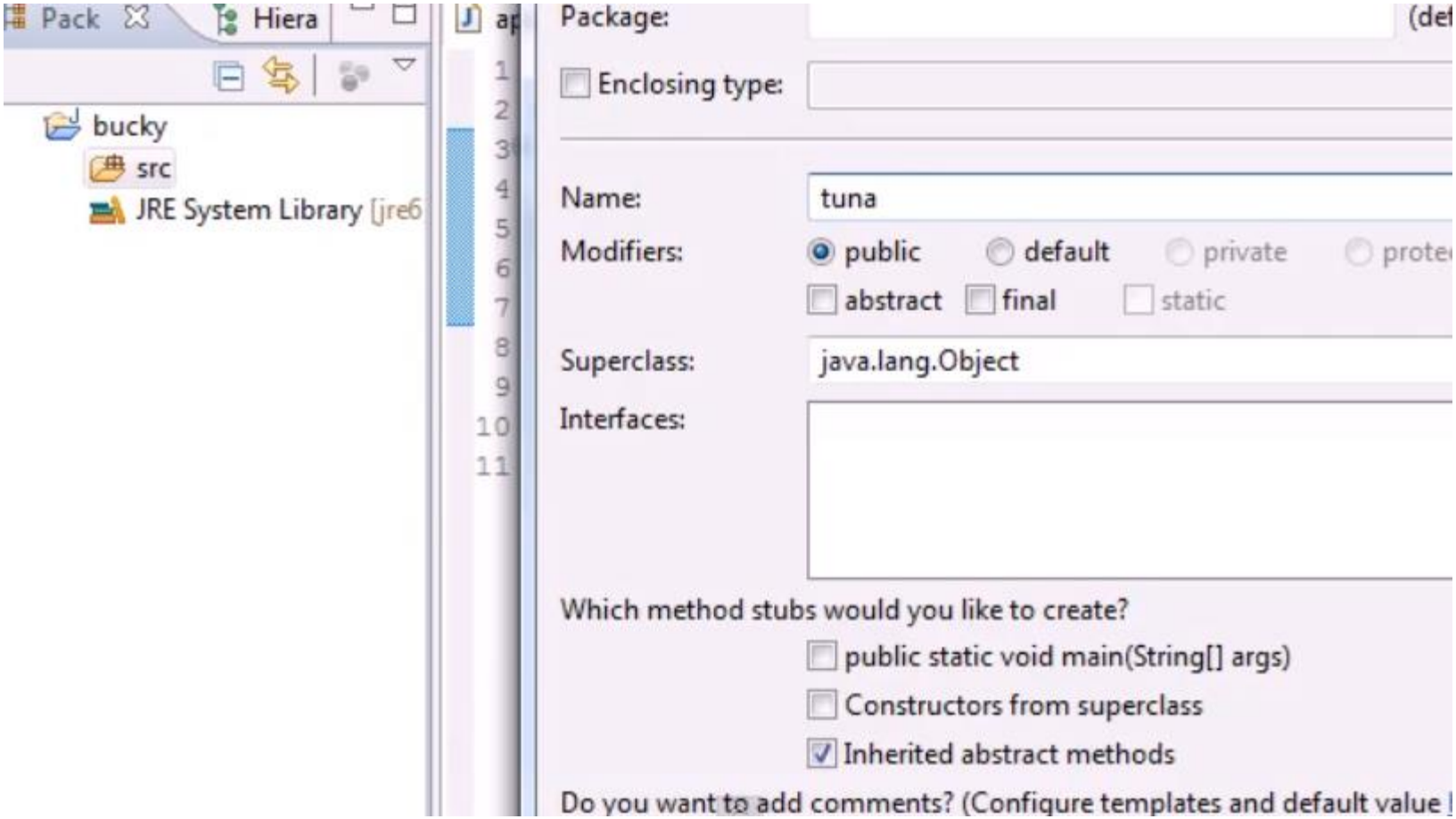
Using objects in Java



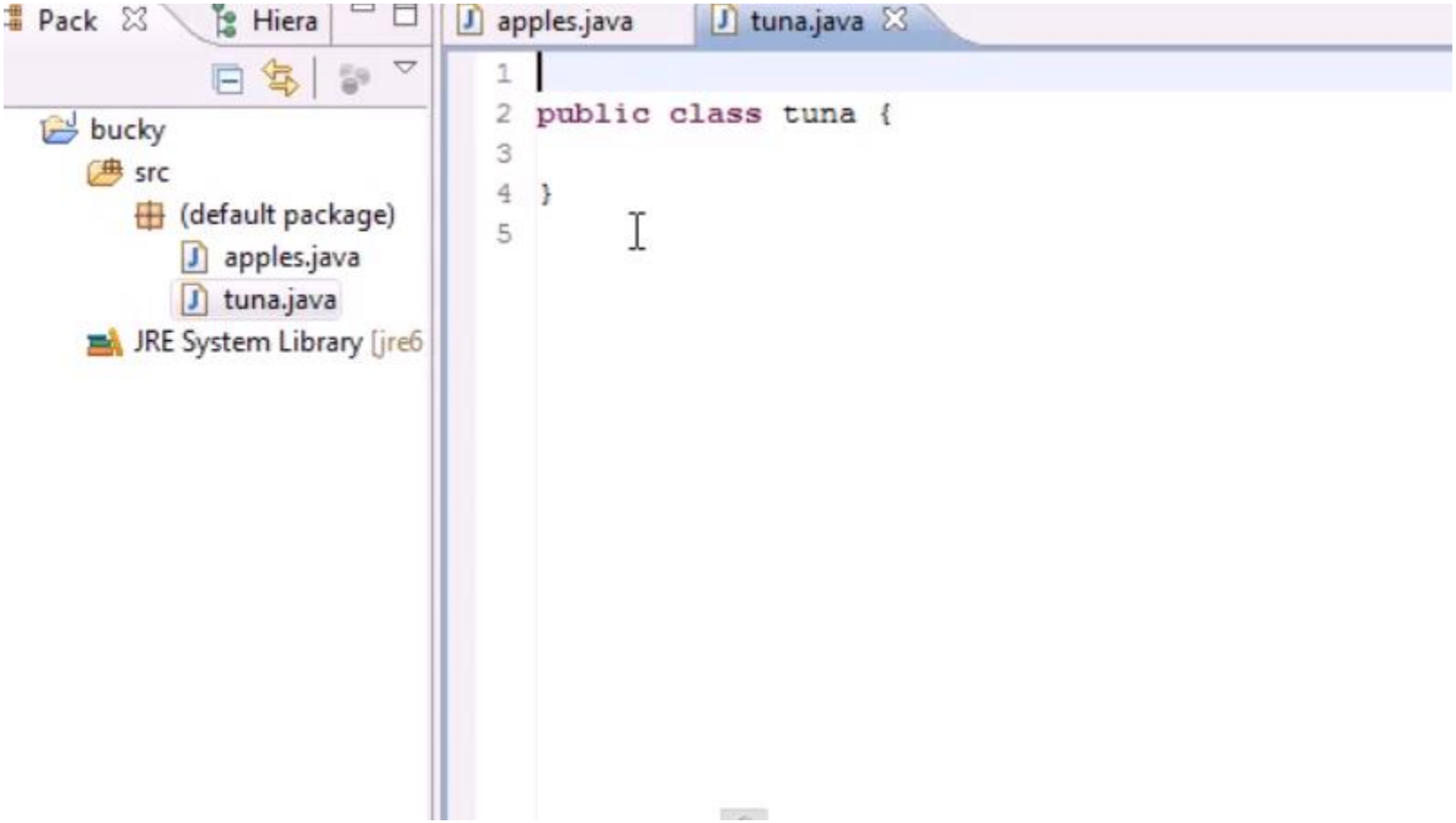
Create a new java class



Name the new class



Write the java class



Add a method in tuna

apples.java | *tuna.java

```
1
2 public class tuna {
3     public void simpleMessage(){
4         System.out.println("This is another class");
5     }
6 }
7
```

Use objects of tuna

*apples.java tuna.java

```
1
2 class apples{
3     public static void main(String[] args) {
4
5             tuna tunaObject = new tuna();
6             tunaObject.simpleMessage();
7
8         }
9     }
10
11
12
```

Add parameter for tuna method

*apples.java | *tuna.java

```
1
2 public class tuna {
3     public void simpleMessage(String name){
4         System.out.println("Hello " + name);
5     }
6 }
7
```

Use parameterized tuna method

*apples.java *tuna.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String[] args) {
5
6              Scanner input = new Scanner(System.in);
7              tuna tunaObject = new tuna ();
8
9              System.out.println("Enter your name here: ");
10             String name = input.nextLine();
11
12             tunaObject.simpleMessage(name);
13
14         }
15     }
16
17
18
```

String name - apples.main(String[])

Press 'F2' for focus

View output

```
17
18
```

Problems | @ Javadoc | Declaration | Console

```
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 9, 2009 5:05:08 PM)
Enter your name here:
bucky roberts
Hello bucky roberts
```

Many methods/instances in Java

*apples.java | *tuna.java

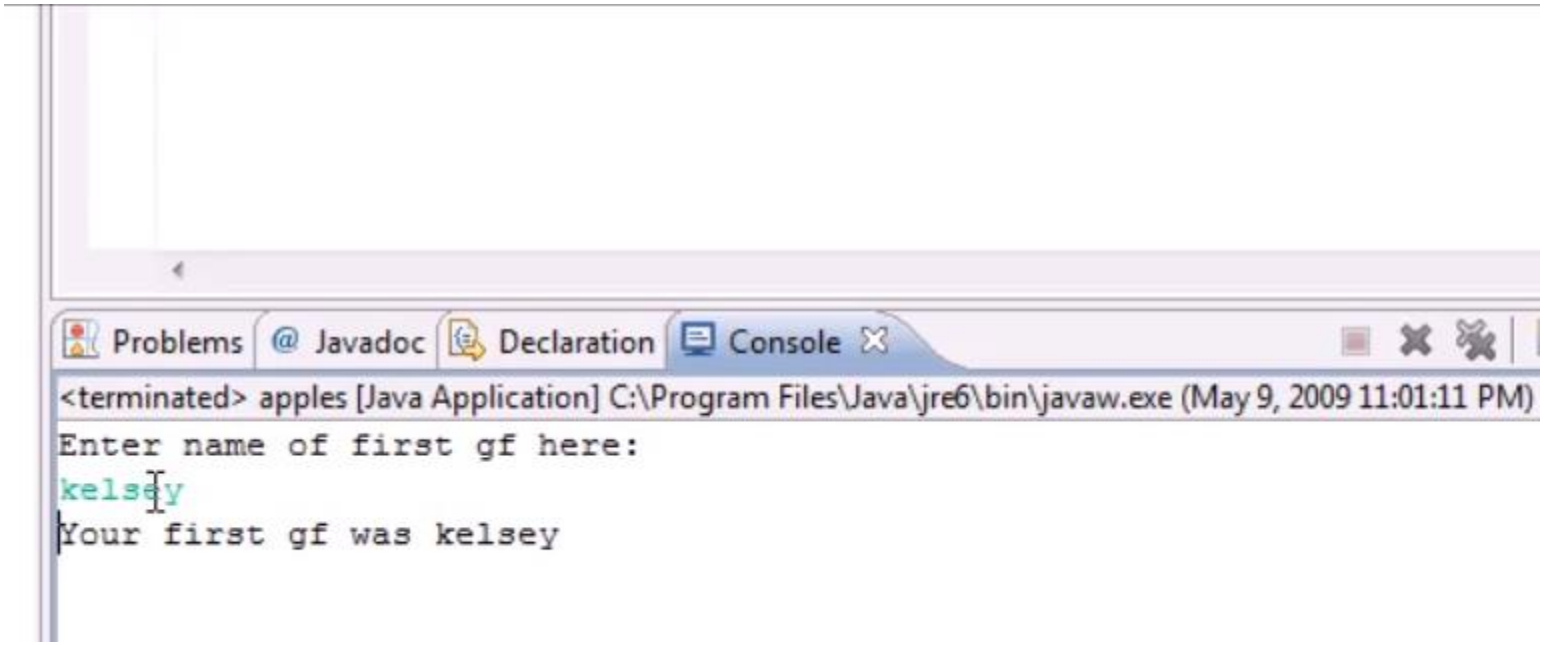
```java
public class tuna {
    private String girlName;
    public void setName(String name){
        girlName=name;
    }
    public String getName(){
        return girlName;
    }
    public void saying(){
        System.out.printf("Your first gf was %s", getName());
    }
}
```

Many methods/instances in Java

*apples.java *tuna.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String[] args) {
5              Scanner input = new Scanner(System.in);
6              tuna tunaObject = new tuna();
7              System.out.println("Enter name of first gf here: ");
8              String temp = input.nextLine();
9              tunaObject.setName(temp);
10             tunaObject.saying();
11         }
12     }
13
14
15
```

View output

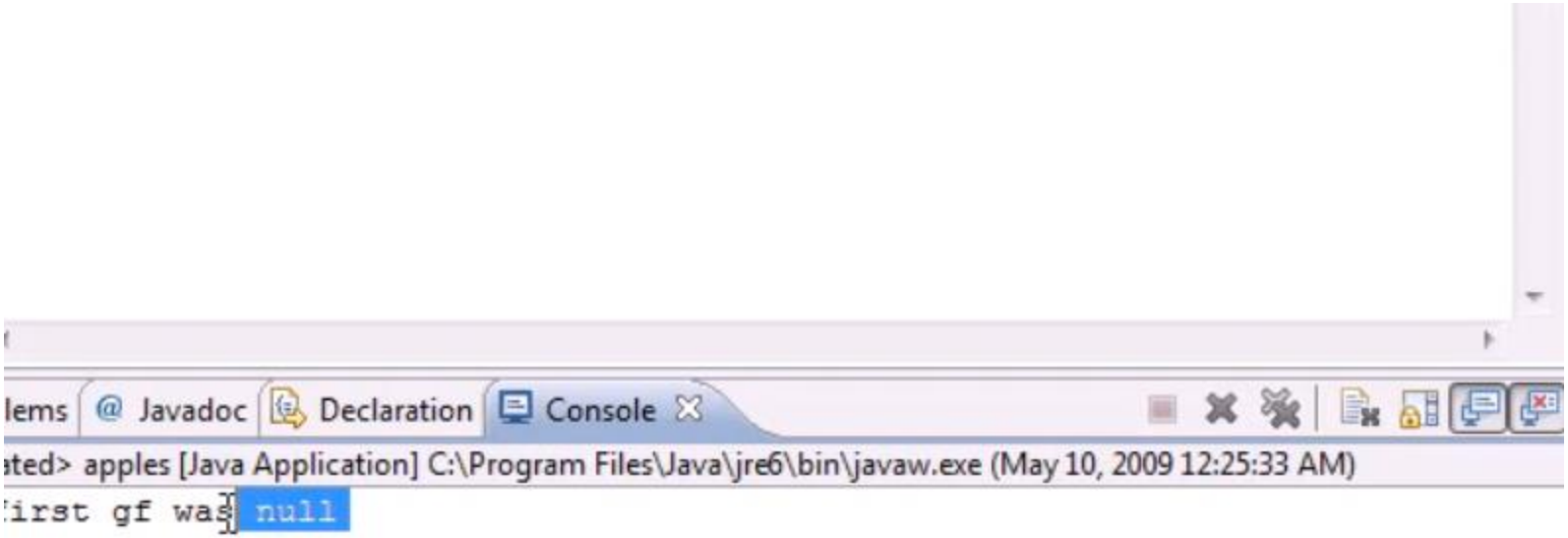

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 9, 2009 11:01:11 PM)
Enter name of first gf here:
kelsey
Your first gf was kelsey

Constructors in Java

*apples.java *tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             tuna tunaObject = new tuna();
4             tunaObject.saying();
5         }
6     }
7
8
9
```

Constructors in Java



Add a constructors in tuna

apples.java | *tuna.java

```java
public class tuna {
    private String girlName;

    public tuna(String name){
        girlName=name;
    }

    public void setName(String name){
        girlName=name;
    }
    public String getName(){
        return girlName;
    }
    public void saying(){
        System.out.printf("Your first gf was %s\n", getName());
    }
}
```

Create tuna objects using constructor

apples.java tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             tuna tunaObject = new tuna("Kelsey");
4             tunaObject.saying();
5         }
6     }
7
8
9
```

Call tuna methods

apples.java | tuna.java

```
1  class apples{
2      public static void main(String[] args) {
3              tuna tunaObject = new tuna("Kelsey");
4              tuna tunaObject2 = new tuna("Nicole");
5              tunaObject.saying();
6              tunaObject2.saying();
7          }
8      }
9
10
11
```

Constructors in Java

Problems | @ Javadoc | Declaration | Console

rminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 12:29:08 AM)

```
ur first gf was Kelsey
ur first gf was Nicole
```

What Is a Class?

- A *class* is the blueprint from which individual objects are created.
- Classes encapsulates states and behaviors of single entity
- Class includes variables which are called instance variables
- Class have methods which are called instance methods

Declaring Classes

Classes are defined in the following way:

```
class MyClass {
    // field, constructor, and
    // method declarations
}
```

Declaring Classes

Class definition consists of two parts:

- The *class declaration*.
- The *class body* (the area between the braces) contains all the code that provides for the life cycle of the objects created from the class
- Class body can have constructors for initializing new objects, declarations for the fields that provide the state of the class, and methods to implement the behavior of the class.

Declaring Classes

Class definition consists of two parts:

- The *class declaration*.
- The *class body* (the area between the braces) contains all the code that provides for the life cycle of the objects created from the class
- Class body can have constructors for initializing new objects, declarations for the fields that provide the state of the class, and methods to implement the behavior of the class.

Declaring Classes

Class declaration can include these components-

- Modifiers such as *public*, *private*
- The class name, with the initial letter capitalized by convention.
- The name of the class's parent (superclass), if any, preceded by the keyword *extends*. A class can only *extend* (subclass) one parent.
- Followed by The class body, surrounded by braces, {}.

Declaring Classes

We can also add modifiers like *public* or *private*

- The modifiers *public* and *private*, which determine what other classes can access the declared class
- We usually write public which means any other class can access

What is an object

- An object stores its state in *fields* (instance variables)
- An object exposes its behavior through *methods*
- Methods operate on an object's internal state and serve as the primary mechanism for object-to-object communication.

What is an object

- Hiding internal state and requiring all interaction to be performed through an object's methods is known as *data encapsulation* — a fundamental principle of object-oriented programming

Declaring Member Variables

There are several kinds of variables:

- Member variables in a class—these are called *fields*.
- Variables in a method or block of code—these are called *local variables*.
- Variables in method declarations—these are called *parameters*.

Declaring Member Variables

Field declarations are composed of three components, in order:

- Zero or more modifiers, such as public or private.
- The field's type.
- The field's name.

Access Modifiers

The first (left-most) modifier used lets you control what other classes have access to a member field.

- public modifier—the field is accessible from all classes.
- private modifier—the field is accessible only within its own class.
- In the spirit of encapsulation, it is common to make fields private.

Variable Types

All variables must have a type.

- You can use primitive types such as int, float, boolean, etc. Or
- you can use reference types, such as strings, arrays, or objects.

Variable Names

All variables must have a type.

- You can use primitive types such as int, float, boolean, etc. Or
- you can use reference types, such as strings, arrays, or objects.

Defining Methods

Method declarations have six components, in order:

- Modifiers—such as public, private, and others you will learn about later.
- The return type—the data type of the value returned by the method, or void if the method does not return a value.
- The method name—the rules for field names apply to method names as well, but the convention is a little different.

Defining Methods

Method declarations have six components, in order:

- The method body, enclosed between braces—the method's code, including the declaration of local variables, goes here.

Nested if in Java

*apples.java tuna.java

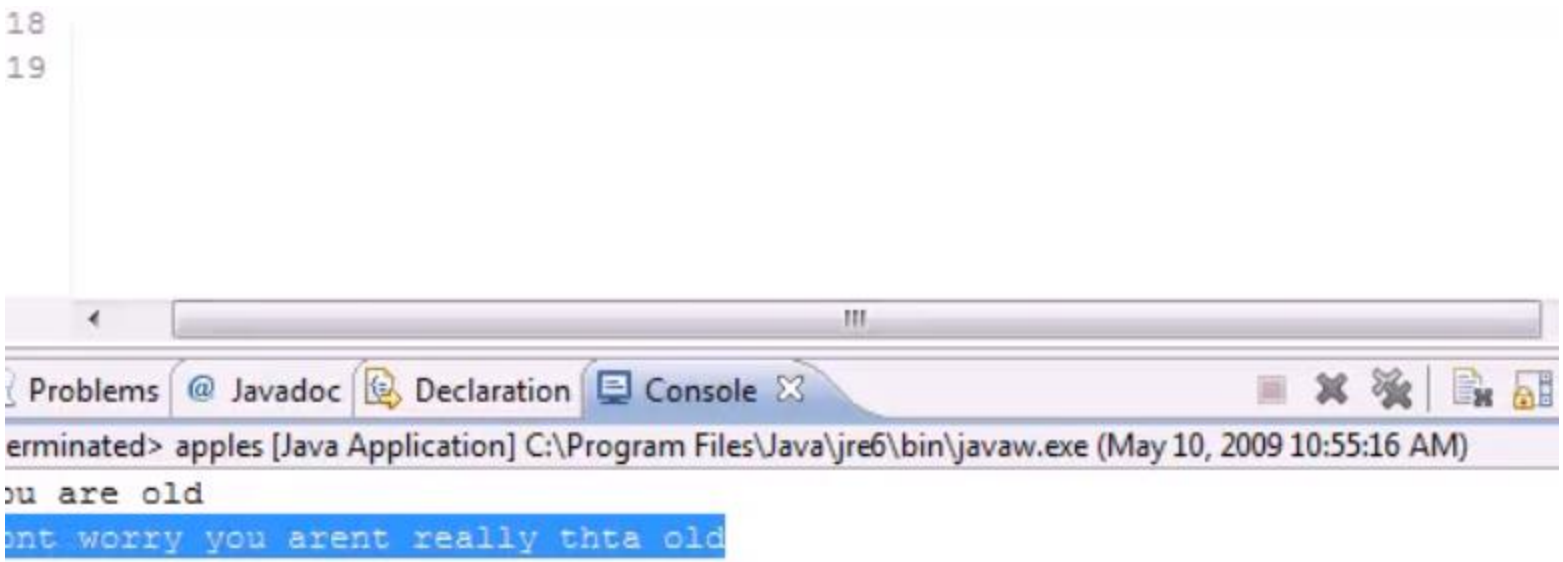
```
1  class apples{
2      public static void main(String[] args) {
3              int age = 60;
4
5              if (age < 50){
6                  System.out.println("You are young");
7              }else{
8                  System.out.println("You are old");
9                  if (age > 75){
10                     System.out.println("You are REALLY old!");
11                 }
12             }
13         }
14     }
15
16
17
```

Nested if in Java

*apples.java tuna.java

```
1  ss apples{
2   public static void main(String[] args) {
3           int age = 60;
4
5           if (age < 50){
6               System.out.println("You are young");
7           }else{
8               System.out.println("You are old");
9               if (age > 75){
10                  System.out.println("You are REALLY old!");
11              }else{
12                  System.out.println("dont worry you arent really thta old"
13              }
14          }
15      }
16  }
17
18
19
```

View output

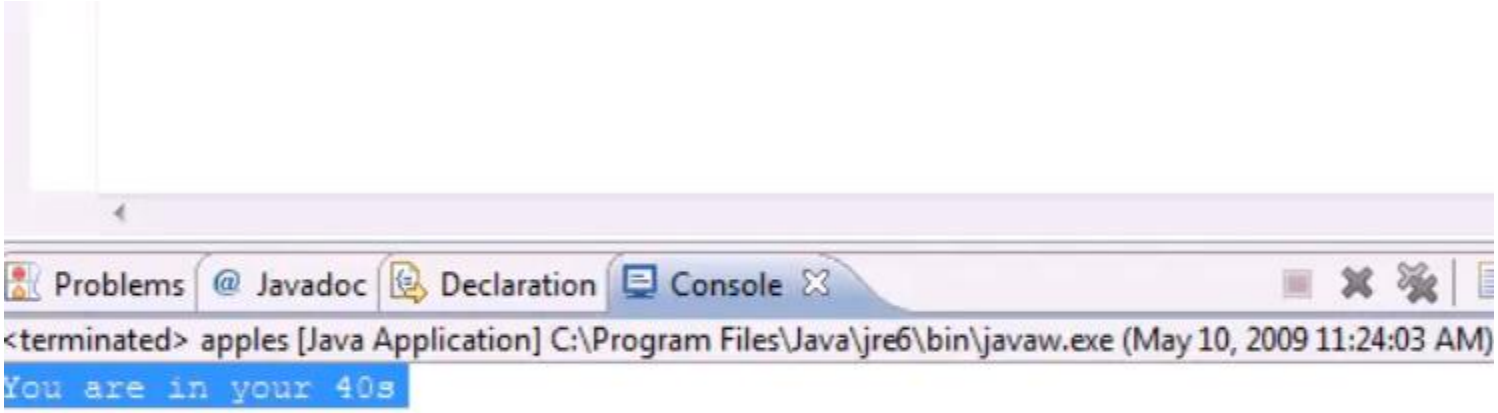

18
19
Problems
Javadoc
Declaration
Console
erminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 10:55:16 AM)
ou are old
ont worry you arent really thta old

Nested if in Java

*apples.java tuna.java

```java
class apples{
    public static void main(String[] args) {
            int age = 45;

            if (age >= 60)
                System.out.println("You are a senior citizen");
            else if (age >=50)
                System.out.println("You are in your 50s");
            else if (age >=40)
                System.out.println("You are in your 40s");
            else
                System.out.println("You are a young buck");

        }
    }
```

View output

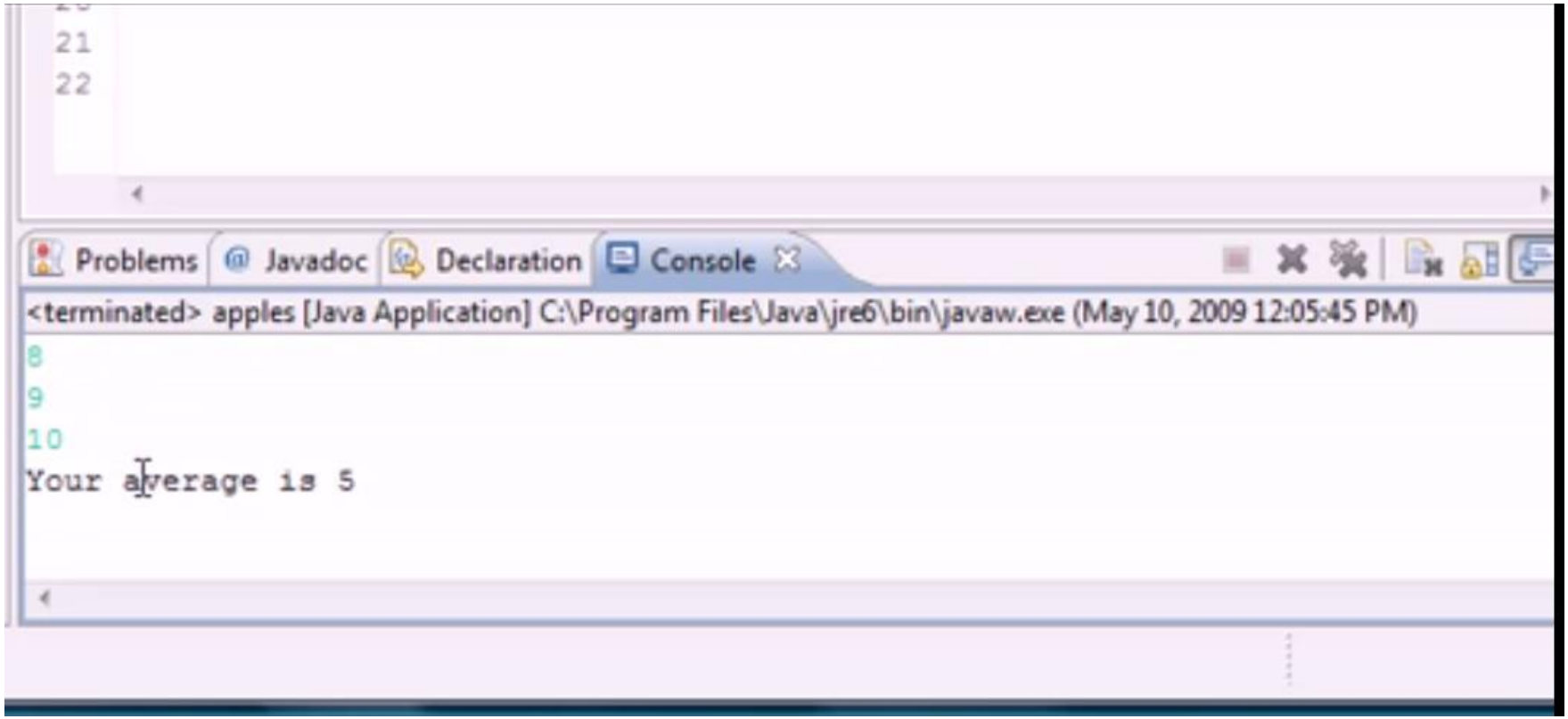

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 11:24:03 AM)
You are in your 40s

Simple Average Program in Java

*apples.java tuna.java

```
1  import java.util.Scanner;
2
3  class apples{
4      public static void main(String[] args) {
5              Scanner input = new Scanner(System.in);
6              int total = 0;
7              int grade;
8              int average;
9              int counter = 0;
10
11             while (counter < 10){
12                 grade = input.nextInt();
13                 total = total + grade;
14                 counter++;
15             }
16             average = total/10;
17             System.out.println("Your average is "+ average);
18         }
```

View output


21
22
Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 12:05:45 PM)
8
9
10
Your average is 5

For loop in Java

Java_Programming_Tutorial_-_22_-_for_Loops.mp4 - VLC media player

Media Playback Audio Video Tools View Help

*apples.java tuna.java

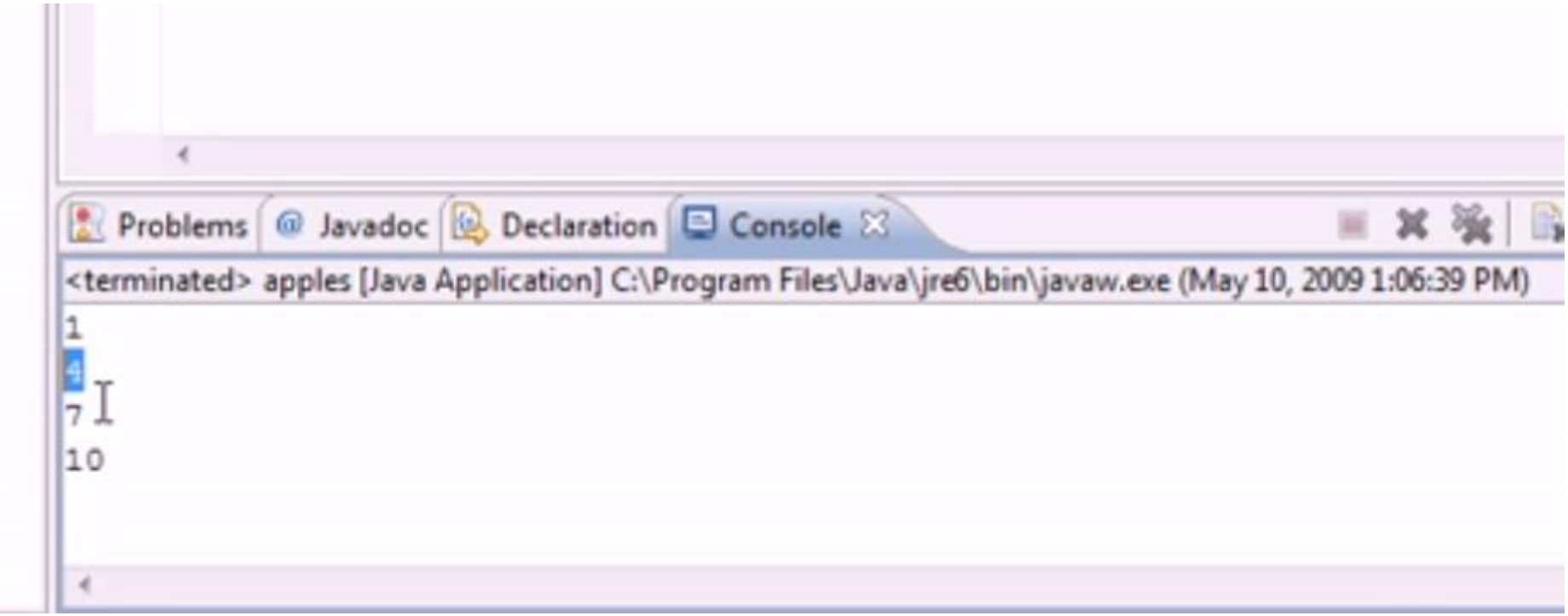
```
1  class apples{
2      public static void main(String[] args) {
3              for(int counter=1;counter<=10;counter++){
4                  System.out.println(counter);
5              }
6          }
7      }
8
9
10
```

For loop in Java

*apples.java tuna.java

```
1  class apples{
2      public static void main(String[] args) {
3              for(int counter=1;counter<=10;counter+=3){
4                  System.out.println(counter);
5              }
6          }
7      }
8
9
10
```

View output

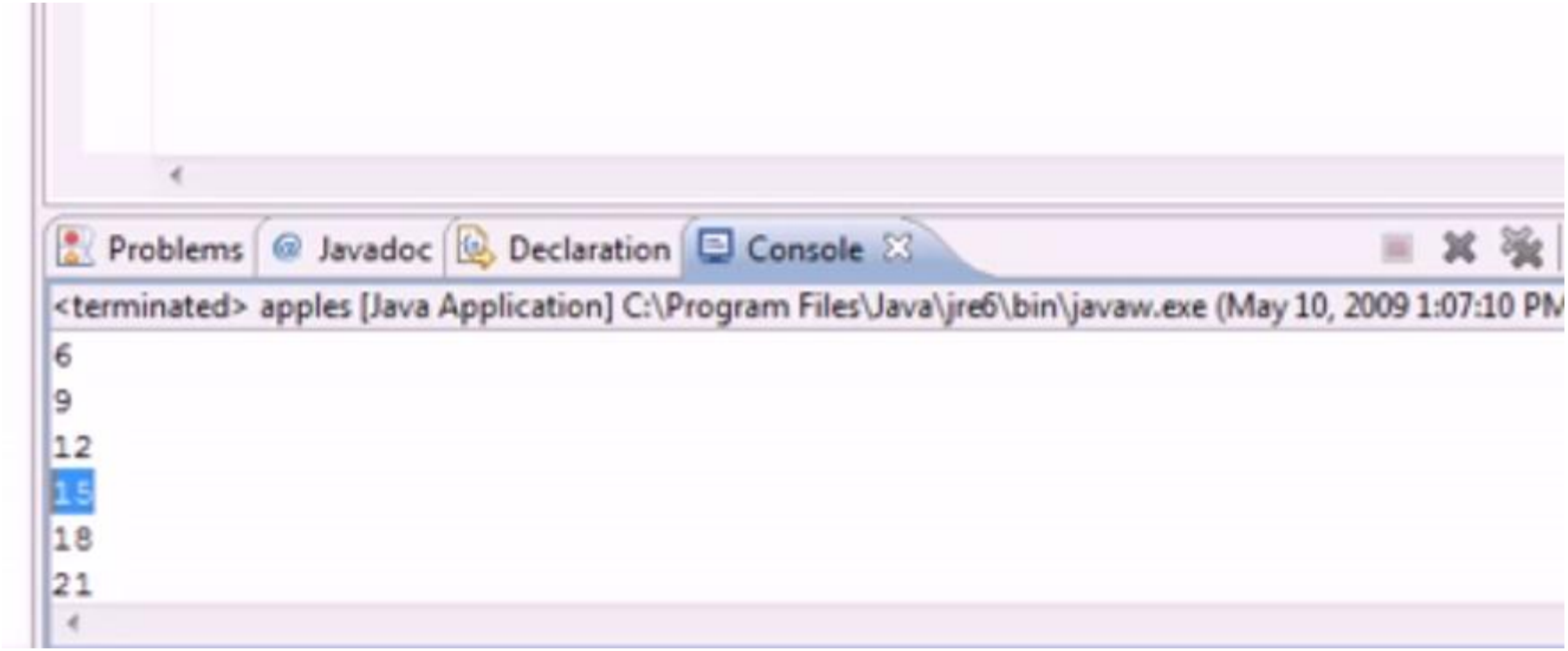

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 1:06:39 PM)
1
4
7
10

Change increment section

*apples.java tuna.java

```
1  class apples{
2      public static void main(String[] args) {
3              for(int counter=6;counter<=21;counter+=3){
4                  System.out.println(counter);
5              }
6          }
7      }
8
9
10
```

For loop in Java



Compound Interest in Java

*apples.java tuna.java

```java
class apples{
    public static void main(String[] args) {
            double amount;
            double principal = 10000;
            double rate = .01;

            for(int day=1;day<=20;day++){
                amount=principal*Math.pow(1 + rate, day);
                System.out.println(day + "    "+ amount);
            }
        }
    }
```

Compound Interest in Java

Problems | Javadoc | Declaration | Console

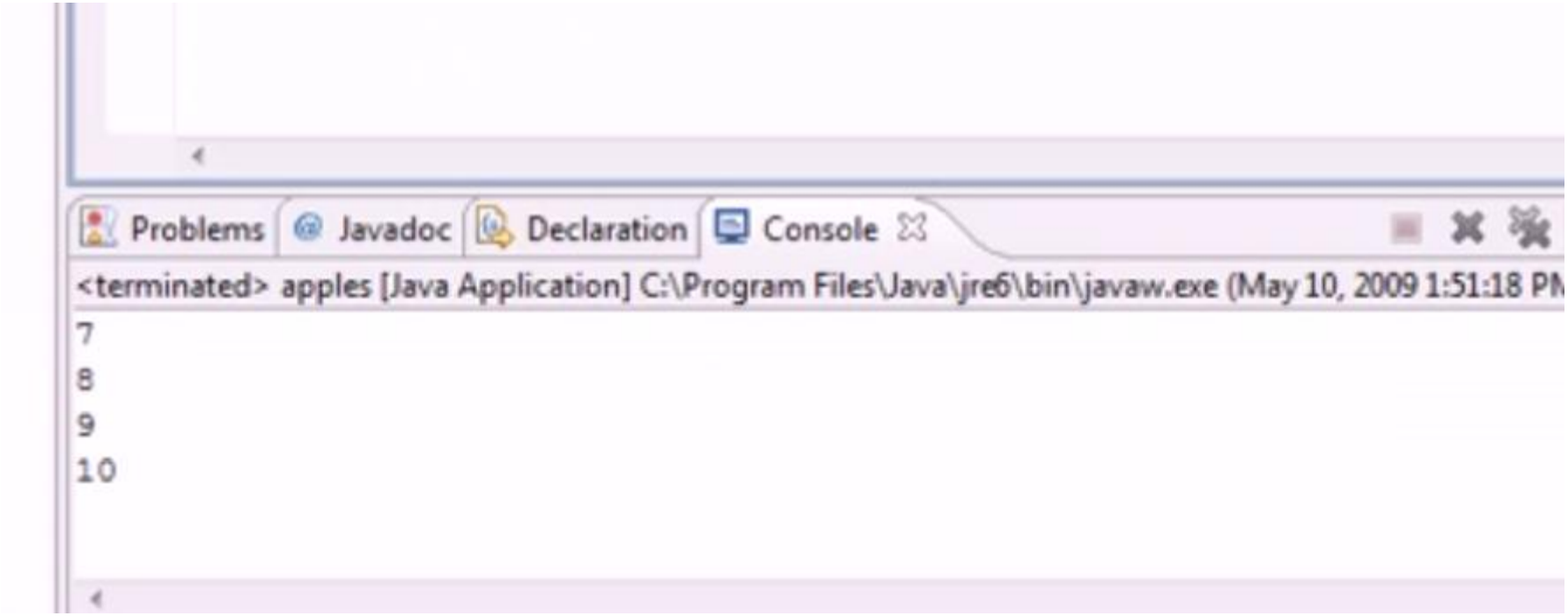
```
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 1:24:52 PM)
1   10100.0
2   10201.0
3   10303.010000000002
4   10406.0401
5   10510.100501
6   10615.201506010002
```

Do-while in Java

*apples.java tuna.java

```
1  class apples{
2      public static void main(String[] args) {
3              int counter = 0;
4
5              do{
6                  System.out.println(counter);
7                  counter++;
8              }while(counter <=10);
9          }
10     }
11
12
13
```

View output

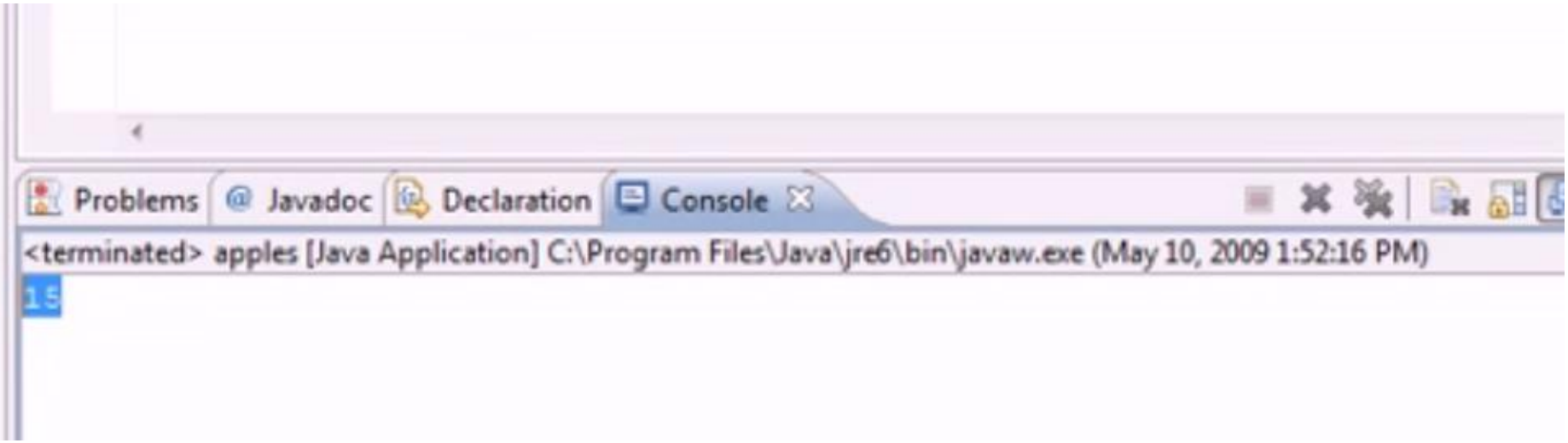

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 1:51:18 PM
7
8
9
10

Change counter value

*apples.java tuna.java

```
1  class apples{
2      public static void main(String[] args) {
3              int counter = 15;
4
5              do{
6                  System.out.println(counter);
7                  counter++;
8              }while(counter <=10);
9          }
10     }
11
12
13
```

View output

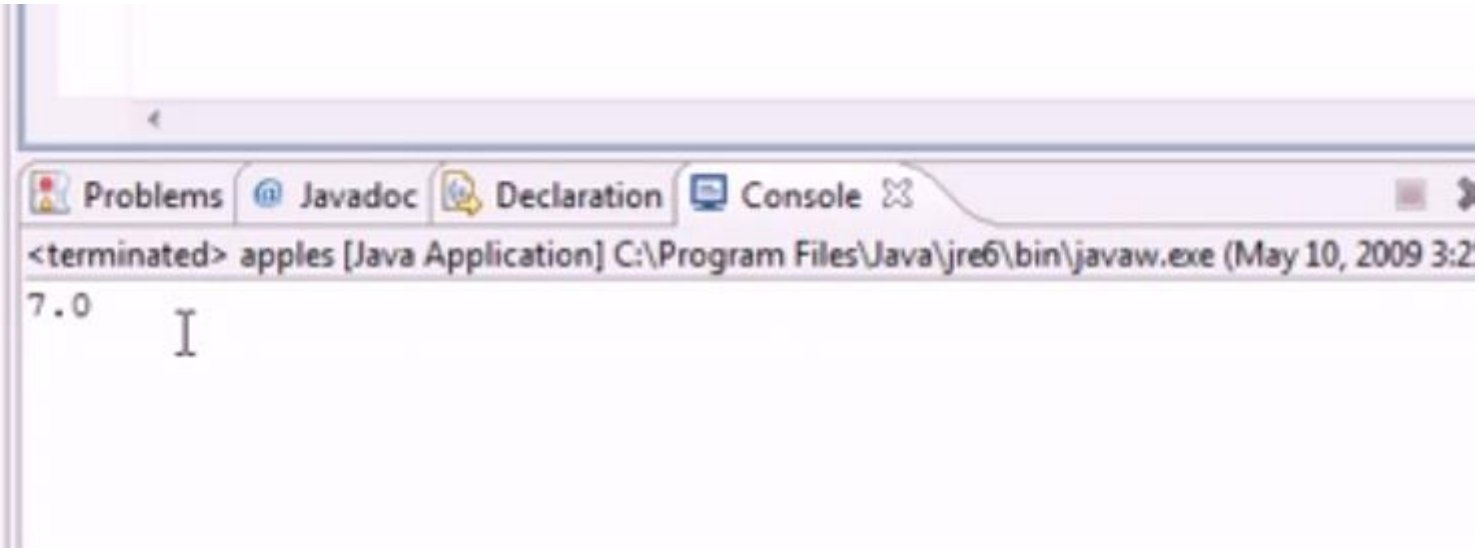

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 1:52:16 PM)
15

Math Class Methods in java: ceil

apples.java tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             System.out.println(Math.ceil(7.4));
4         }
5     }
6
7
8
```

View output

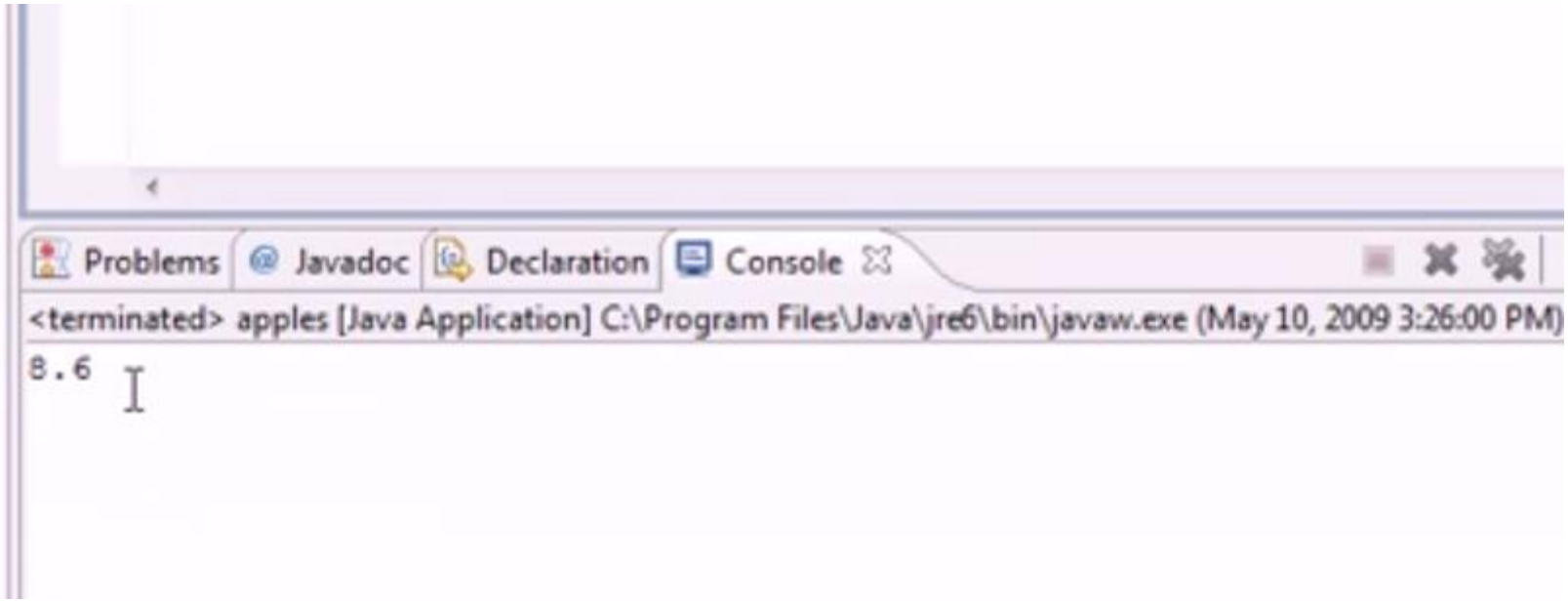

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 3:2
7.0

Math Class Methods in Java: max

*apples.java tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             System.out.println(Math.max(8.6,5.2));
4         }
5     }
6
7
8
```

View output

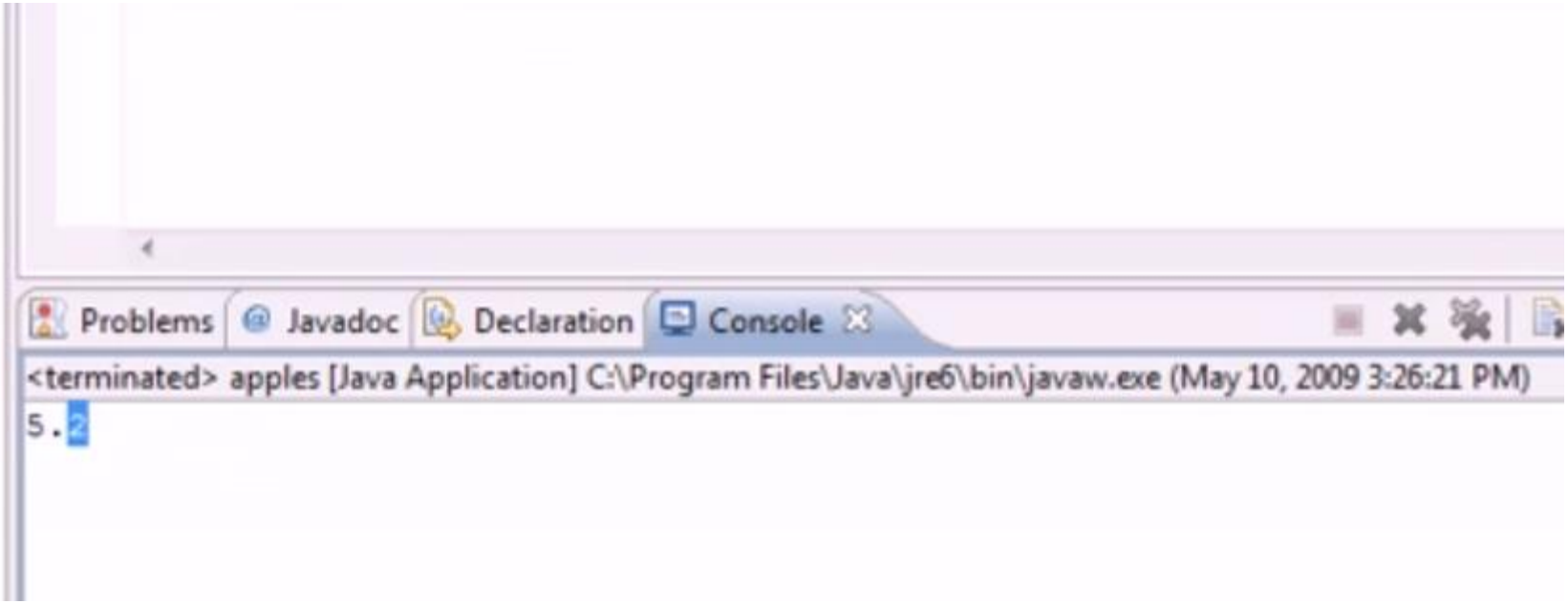

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 3:26:00 PM)
8.6

Math Class Methods in Java: min

*apples.java tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             System.out.println(Math.min(8.6,5.2));
4         }
5     }
6
7
8
```

View output

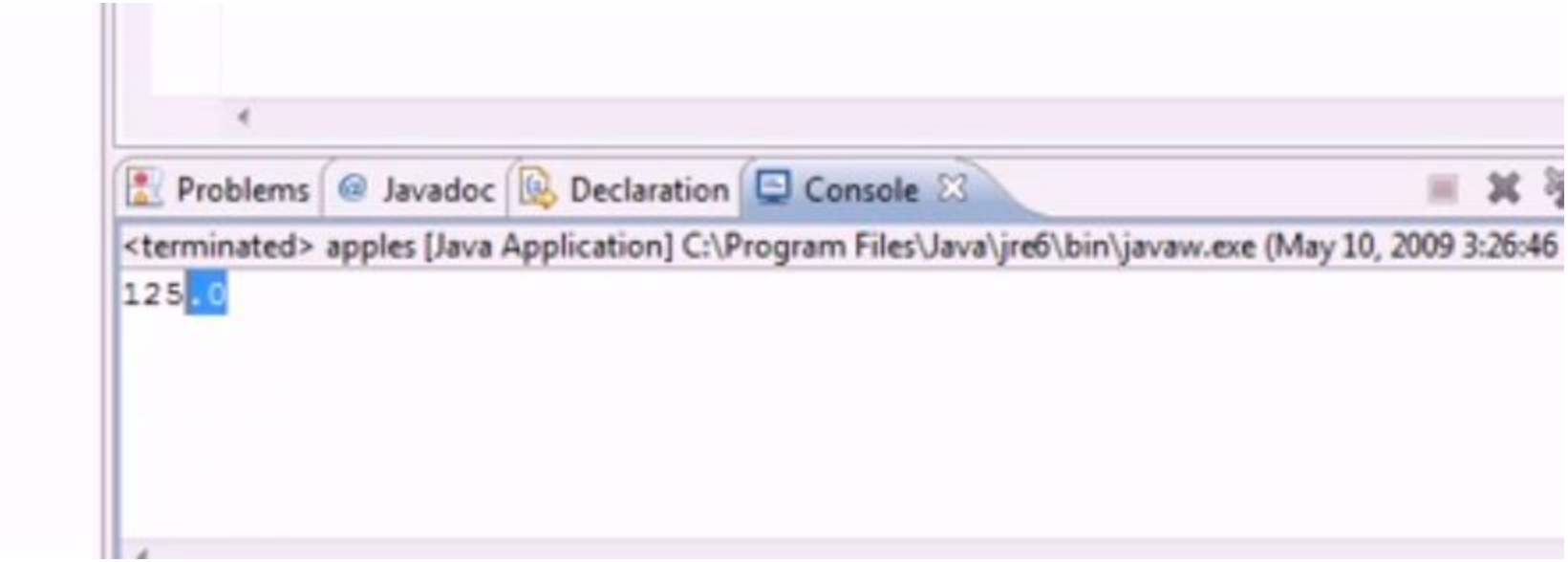

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 3:26:21 PM)
5.2

Math Class Methods in Java: pow

*apples.java tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             System.out.println(Math.pow(5,3));
4         }
5     }
6
7
8
```

View output

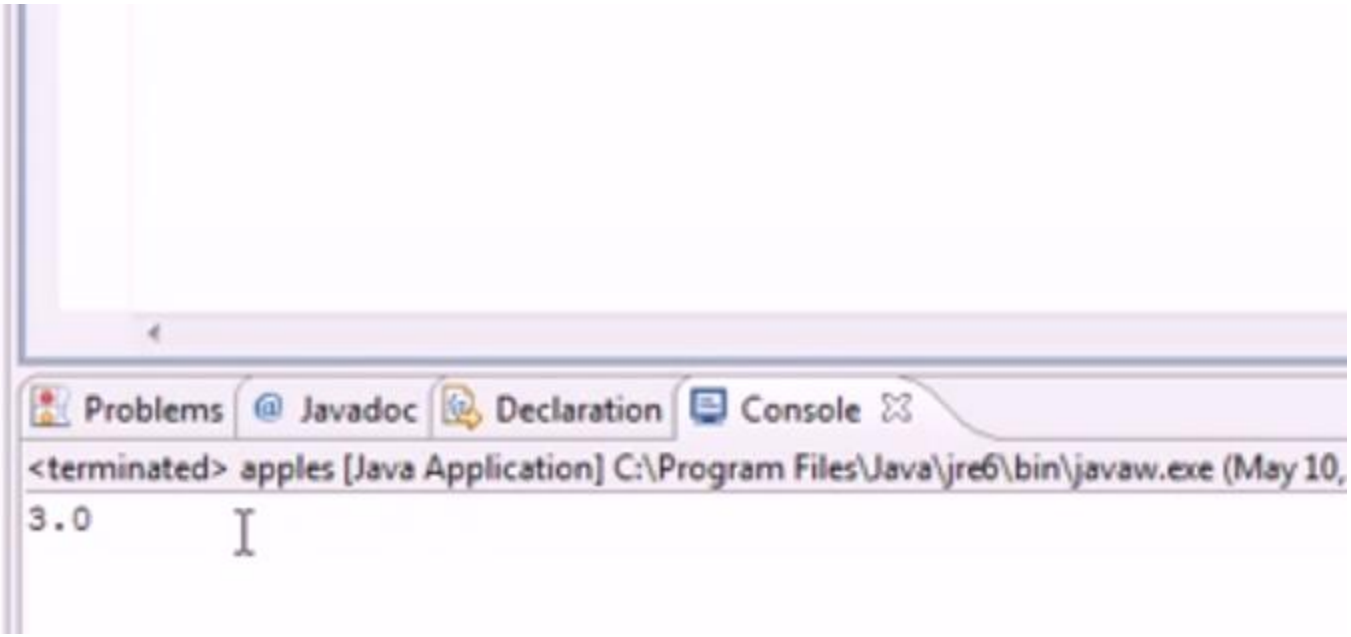

Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10, 2009 3:26:46
125.0

Math Class Methods in Java: sqrt

*apples.java tuna.java

```
1 class apples{
2     public static void main(String[] args) {
3             System.out.println(Math.sqrt(9));
4         }
5     }
6
7
8
```

View output


Problems
Javadoc
Declaration
Console
<terminated> apples [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 10,
3.0