

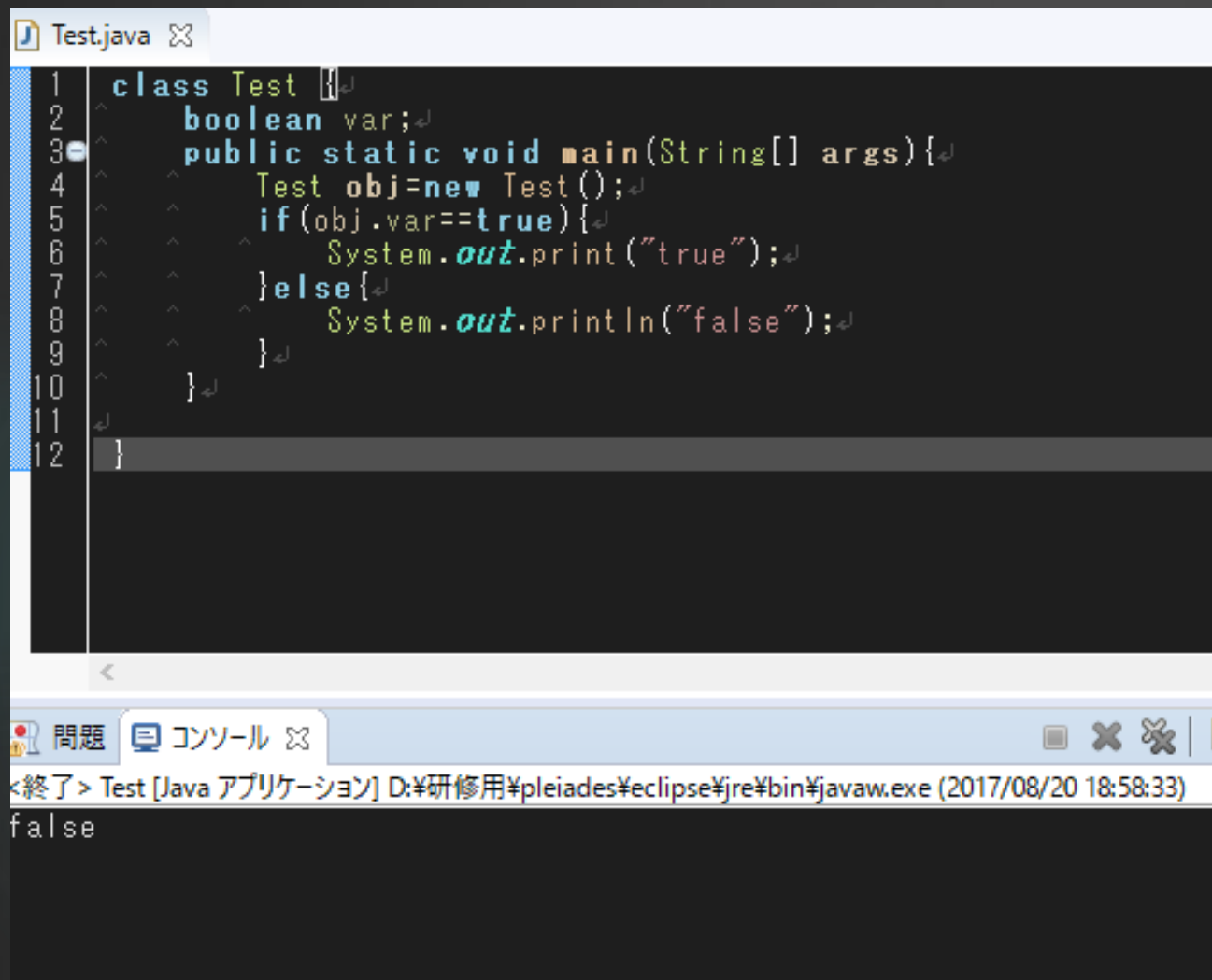
第6章

クラス定義とオブジェクトの生成・使用

6-1

▶ 省略

6-2



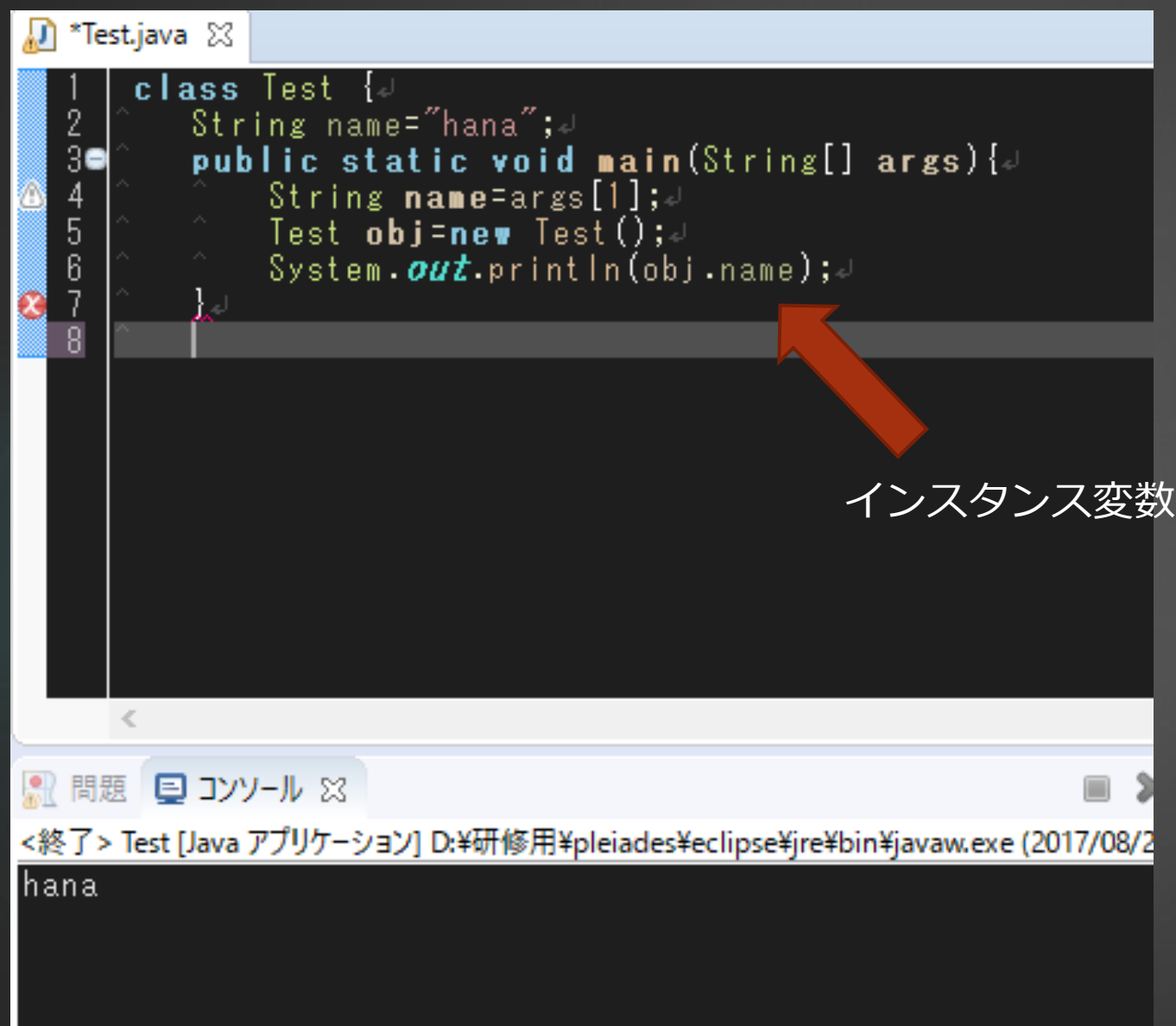
The screenshot shows the Eclipse IDE with a Java file named `Test.java` open. The code defines a `Test` class with a `boolean` variable `var` and a `main` method. The `main` method creates a new `Test` object and checks if `obj.var` is `true`. If it is, it prints `true`; otherwise, it prints `false`. The IDE's console window at the bottom shows the output `false`.

```
1 class Test {  
2     boolean var;  
3     public static void main(String[] args) {  
4         Test obj = new Test();  
5         if (obj.var == true) {  
6             System.out.print("true");  
7         } else {  
8             System.out.println("false");  
9         }  
10    }  
11 }  
12 }
```

問題 コンソール

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/20 18:58:33)
false

6-3



The screenshot shows the Eclipse IDE with a Java file named `*Test.java`. The code defines a `Test` class with a `main` method. A red arrow points to the `Test obj = new Test();` line, with the text "インスタンス変数" (Instance Variable) written below it. The console at the bottom shows the output "hana".

```
1 class Test {  
2     String name="hana";  
3     public static void main(String[] args){  
4         String name=args[1];  
5         Test obj=new Test();  
6         System.out.println(obj.name);  
7     }  
8 }
```

インスタンス変数

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/2
hana

名前(N): Test

メイン (x)= 引数 JRE クラスパス ソース

プログラムの引数(A):

kei bill

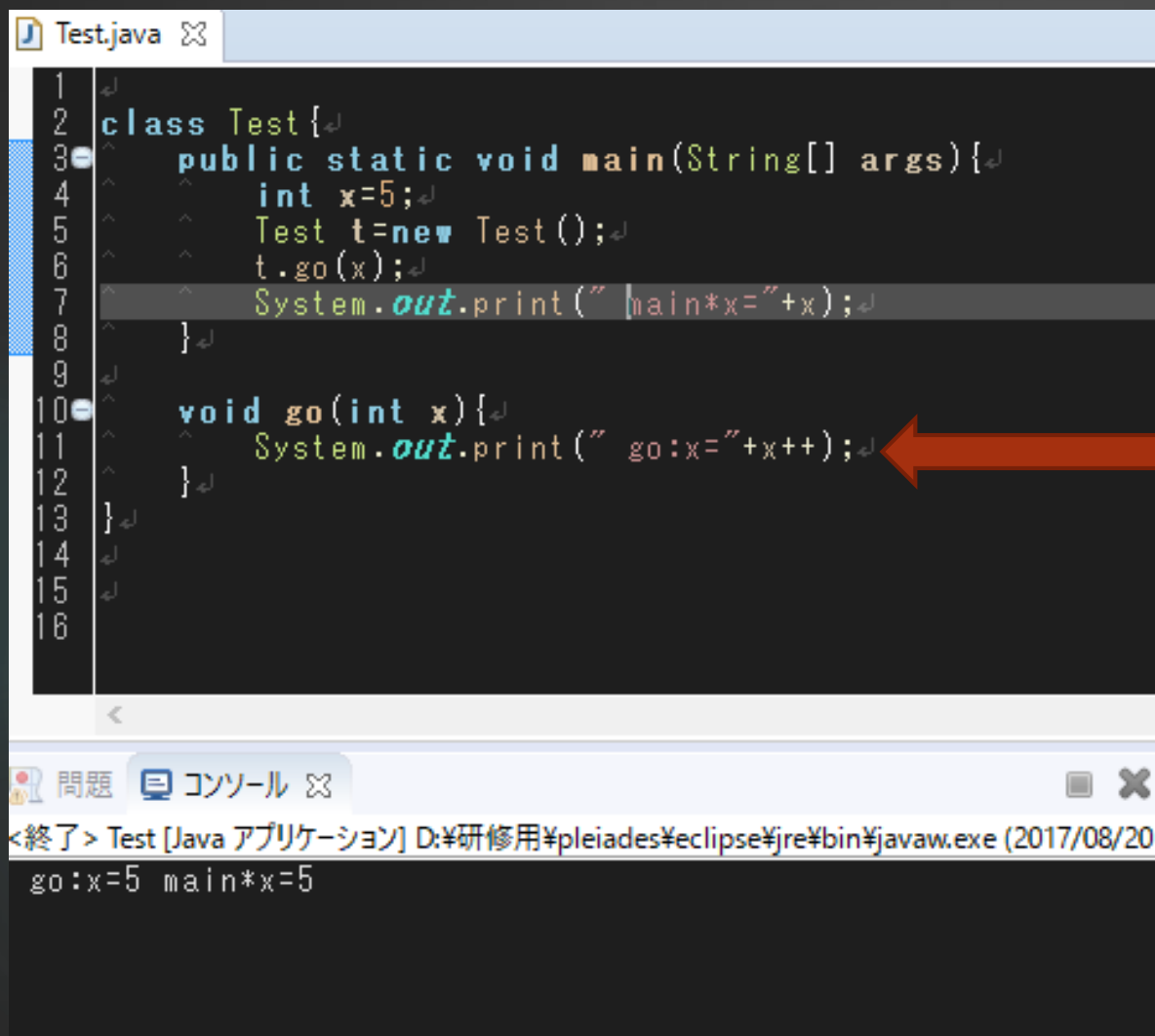
6-4

```
Test.java
1 class Foo {
2     int[] method(int a,int b){
3         int[] data={a,b};
4         return data;
5     }
6
7 }
8
9 class Test{
10    public static void main(String[] args){
11        Foo obj= new Foo();
12        int []array=obj.method(100,200);
13        System.out.println(array[0]+array[1]);
14    }
15 }
16
17
18
```

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/30)

- メソッドmethodの条件：
- ・ int型の引数二つを返す
 - ・ 配列

6-5



The screenshot shows the Eclipse IDE with a Java file named `Test.java` open. The code is as follows:

```
1 class Test {  
2     public static void main(String[] args) {  
3         int x=5;  
4         Test t=new Test();  
5         t.go(x);  
6         System.out.print(" main*x="+x);  
7     }  
8  
9     void go(int x){  
10        System.out.print(" go:x="+x++);  
11    }  
12 }  
13  
14  
15  
16
```

A red arrow points to the line `System.out.print(" go:x="+x++);` in the `go` method, highlighting the increment operation `x++`.

Below the code editor, the console window is visible, showing the output:

```
<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/20  
go:x=5 main*x=5
```

インクリメント：8行目～10行目まで有効

6-6

```
Test.java
1 class Test {
2     int x=50;
3     int y=100;
4     public static void main(String[] args) {
5         int x=0, y=10;
6         Test t=new Test();
7         while(x<3) {
8             x++;
9             y--;
10        }
11
12        System.out.println("x="+x+", y="+y);
13    }
14 }
```

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/0

x=3, y=7

	x	y
x=0	1	9
x=1	2	8
x=2	3	7

6-7(1)

Test.java

```
1 class MyClass{  
2     public int myMethod(double a,int i){return 0;}  
3  
4     public int myMethod(int i,double a){return 0;}  
5     //public double myMethod(double b,int j){return 0;}  
6     //public int myMethod(double a,double b,int i){return 0;}  
7     //public int yourMethod(double a,int i){return 0;}  
8     //int myMethod(double a,int i){return 0;}  
9     //int myMethod(double a){return 0;}  
10    //public int myMethod(double x,int y){return 0;}  
11 }
```

戻りの順番が変わった！

Test.java

```
1 class MyClass{  
2     public int myMethod(double a,int i){return 0;}  
3  
4     //public int myMethod(int i,double a){return 0;}  
5     //public double myMethod(double b,int j){return 0;}  
6     //public int myMethod(double a,double b,int i){return 0;}  
7     //public int yourMethod(double a,int i){return 0;}  
8     //int myMethod(double a,int i){return 0;}  
9     int myMethod(double a){return 0;}  
10    //public int myMethod(double x,int y){return 0;}  
11 }
```

戻りの数、型、順番が変わった！

Test.java

```
1 class MyClass{  
2     public int myMethod(double a,int i){return 0;}  
3  
4     //public int myMethod(int i,double a){return 0;}  
5     //public double myMethod(double b,int j){return 0;}  
6     public int myMethod(double a,double b,int i){return 0;}  
7     //public int yourMethod(double a,int i){return 0;}  
8     //int myMethod(double a,int i){return 0;}  
9     //int myMethod(double a){return 0;}  
10    //public int myMethod(double x,int y){return 0;}  
11 }
```

戻りの数、型、順番が変わった！

Test.java

```
1 class MyClass{  
2     public int myMethod(double a,int i){return 0;}  
3  
4     //public int myMethod(int i,double a){return 0;}  
5     //public double myMethod(double b,int j){return 0;}  
6     //public int myMethod(double a,double b,int i){return 0;}  
7     public int yourMethod(double a,int i){return 0;}  
8     //int myMethod(double a,int i){return 0;}  
9     //int myMethod(double a){return 0;}  
10    //public int myMethod(double x,int y){return 0;}  
11 }
```

別メソッド！

6-7(2)

```
*Test.java
1 class MyClass{
2     public int myMethod(double a,int i){return 0;}
3
4     //public int myMethod(int i,double a){return 0;}
5     public double myMethod(double b,int i){return 0;}
6     //public int myMethod(double a,double b,int i){return 0;}
7     //public int yourMethod(double a,int i){return 0;}
8     //int myMethod(double a,int i){return 0;}
9     //int myMethod(double a){return 0;}
10    //public int myMethod(double x,int y){return 0;}
11 }
```

戻り値として、double型を指定できない

```
Test.java
1 class MyClass{
2     public int myMethod(double a,int i){return 0;}
3
4     //public int myMethod(int i,double a){return 0;}
5     //public double myMethod(double b,int i){return 0;}
6     //public int myMethod(double a,double b,int i){return 0;}
7     //public int yourMethod(double a,int i){return 0;}
8     int myMethod(double a,int i){return 0;}
9     //int myMethod(double a){return 0;}
10    //public int myMethod(double x,int y){return 0;}
11 }
```

引数の型、数、順番が変わっていない

```
Test.java
1 class MyClass{
2     public int myMethod(double a,int i){return 0;}
3
4     //public int myMethod(int i,double a){return 0;}
5     //public double myMethod(double b,int i){return 0;}
6     //public int myMethod(double a,double b,int i){return 0;}
7     //public int yourMethod(double a,int i){return 0;}
8     //int myMethod(double a,int i){return 0;}
9     //int myMethod(double a){return 0;}
10    public int myMethod(double x,int y){return 0;}
11 }
```

引数の型、数、順番が変わっていない

6-8

```
Test.java
1 class Test {
2     static void func(float a, float b) {
3         System.out.print(a+b);
4     }
5     static void func(String a, String b) {
6         System.out.print(a+b);
7     }
8     public static void main(String[] args) {
9         func(10, 20);
10        func("a", "b");
11    }
12 }
```

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/

30.0ab

9行目 : 10.0+20.0=30.0
10行目 : 文字列と文字列

6-9

```
1 class Foo{  
2     void Foo(){  
3         System.out.println("Hello");  
4     }  
5     void Foo(String str){  
6         System.out.println("Bye");  
7     }  
8 }  
9 class Test{  
10    public static void main(String[] args){  
11        Foo f=new Foo();  
12    }  
13 }  
14  
15
```

2行目：voidがついているので、コンストラクタではなく、メソッドである！

f.Foo()を追加すると、
"Hello"が出力される

問題 コンソール

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/20 19:45:59)

6-10

```
Foo.java
1 class Foo{
2     Foo(){
3         System.out.println("Hello");
4     }
5     private Foo(String str){
6         System.out.println("Bye");
7     }
8 }
9 class Test{
10    public static void main(String[] args){
11        Foo f=new Foo();
12    }
13 }
14
15
```

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/2

Hello

コンストラクタである！

6-11、6-12、6-13、6-14

▶ 省略

6-15

```
Foo.java
1 class Test {
2     private static int a;
3     private int b;
4     public static int methodA() {
5         return ++a;
6     }
7     public int methodB() {
8         return methodA();
9     }
10    public static void main(String[] args) {
11        Test obj = new Test();
12        System.out.print(obj.methodB() + " ");
13        System.out.print(obj.methodA());
14    }
15 }
16
```

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017

1 2

初期化しない場合、自動的に0になる。

12行目：
methodA()の結果がobj.methodB()の結果になる→インクリメントによって、1が出力される。

13行目：
インクリメントによって、2が出力される。

6-16

```
Test.java
1 class Test {
2     public static void main(String[] args) {
3         int num;
4         num=10;
5         calc(num);
6         System.out.println("num="+num);
7     }
8
9     static void calc(int num) {
10        num+=100;
11    }
12 }
13
```

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/20 20:04)

num=10

戻り値は返さない！
(110という結果が返さない)

6-17

```
Test.java
1 class Test {
2     ^ static int num;
3     ^ void method() {
4         ^ num++;
5     }
6     ^ static void methodB() {
7         ^ num++;
8     }
9     ^ public static void main(String[] args) {
10        ^ methodA();
11        ^ methodB();
12        ^ System.out.println(num);
13    }
14 }
15
```

methodAは非staticメソッド
なので、生成せずに呼び出す
ことができない！

6-18

```
Test.java
1 class Test{
2     static String lang="C";
3     public String operation="Unix";
4     Test(){
5     }
6     Test(String str){
7         operation=str;
8     }
9     public static void main(String args[]){
10        Test obj1=new Test();
11        Test obj2=new Test("Solaris");
12        obj2.lang="Java";
13        System.out.println(obj1.lang+"%t"+obj1.operation);
14        System.out.println(obj2.lang+"%t"+obj2.operation);
15    }
16 }
17 }
```

問題 コンソール

<終了> Test [Java アプリケーション] D:\研修用\pleiades\eclipse\jre\bin\javaw.exe (2017/08/20 21:38:29)

Java Unix
Java Solaris

langはstatic変数なので、"Java"をlang変数に代入することによって、obj1も同じデータ("Java")を参照しているので、obj1.langの結果も"Java"になる。