

Chapter 4 Methods

1. At least three benefits: (1) Reuse code; (2) Reduce complexity; (3) Easy to maintain. See the Sections 4.2 and 4.3 on how to declare and invoke methods.
2. `void`
3. Yes. `return (num1 > num2) ? num1 : num2;`
4. True: a call to a method with a void return type is always a statement itself.
False: a call to a method with a nonvoid return type is always a component of an expression.
5. A syntax error occurs if a return statement does not appear in a non-void method. You can have a return statement in a void method, which simply exits the method. But a return statement cannot return a value such as `return x + y` in a void method.
6. No.
7. Computing a sales commission given the sales amount and the commission rate
Answer: non-void method with return type double

Printing a calendar for a month

Answer: void method

Computing a square root

Answer: non-void method with return type double

Testing whether a number is even and return true if it is

Answer: non-void method with return type boolean

Printing a message for a specified number of times

Answer: void method

Computing the monthly payment, given the loan amount, number of years, and annual interest rate.

Answer: non-void method with return type double.

Finding the corresponding uppercase letter given a lowercase letter.

Answer: non-void method with return type char.

8. Line 2: `method1` is not defined correctly. It does not have a return type or void.
Line 2: type `int` should be declared for parameter `m`.
Line 8: parameter type for `n` should be `double` to match `xMethod(3.4)`.

Line 11: `if (n<0)` should be removed in `xMethod`, otherwise the a compilation error is reported.

9.

```
public class Test {  
    public static double xMethod(double i, double j) {  
        while (i<j) {  
            j--;  
        }  
        return j;  
    }  
}
```

10. You pass actual parameters by passing the right type of value in the right order. The actual parameter can have the same name as its formal parameter.

11. "Pass by value" is to pass a copy of the value to the method.

(A) The output of the program is 0, because the variable `max` is not changed by invoking the method `max`.

(B)

Before the call, variable `times` is 3

`n = 3`

Welcome to Java!

`n = 2`

Welcome to Java!

`n = 1`

Welcome to Java!

After the call, variable `times` is 3

(C)

2

2 4

2 4 8

2 4 8 16

2 4 8 16 32

2 4 8 16 32 64

(D)

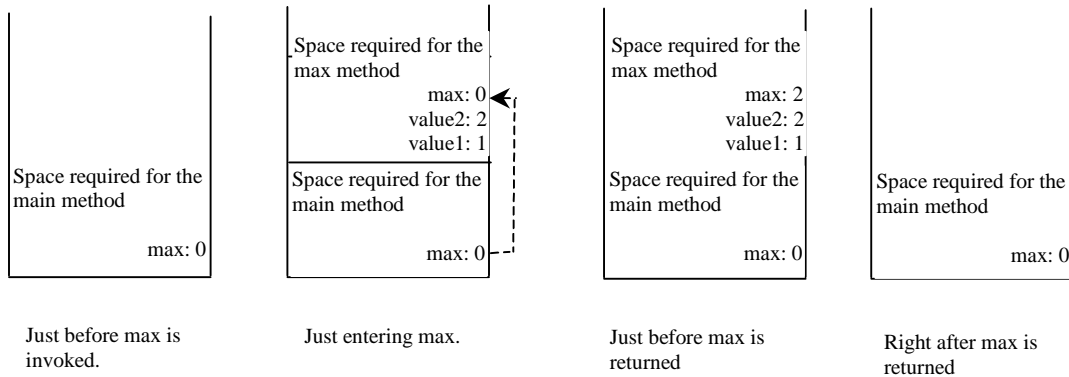
1

2 1

2 1

4 2 1

12.



13. Two methods with the same name, defined in the same class, is called method overloading. It is fine to have same method name, but different parameter types. You cannot overload methods based on return type, or modifiers.

14. Methods `public static void method(int x)` and `public static int method(int y)` have the same signature `method(int)`.

15. Line 8: `int n = 1` is wrong since `n` is already declared in the method signature.

16. 0.5, 0.0, 0.234

17. (a) `34 + (int)(Math.random() * (55 - 34))`
 (b) `(int)(Math.random() * 1000)`
 (c) `5.5 + (Math.random() * (55.5 - 5.5))`
 (d) `(char)('a' + (Math.random() * ('z' - 'a' + 1)))`

18.

`Math.sqrt(4) = 2.0`

`Math.sin(2*Math.PI) = 0`

`Math.cos(2*Math.PI) = 1`

`Math.pow(2, 2) = 4.0`

`Math.log(Math.E) = 1`

`Math.exp(1) = 2.718`

```
Math.max(2, Math.min(3, 4)) = 3
```

```
Math rint(-2.5) = -2.0  
Math.ceil(-2.5) = -2.0  
Math.floor(-2.5) = -3.0  
Math.round(-2.5f) = -2  
Math.round(-2.5) = -2  
Math rint(2.5) = 2.0  
Math.ceil(2.5) = 3.0  
Math.floor(2.5) = 2.0  
Math.round(2.5f) = 3  
Math.round(-2.5) = -2
```

```
Math.round(Math.abs(-2.5)) = 3
```

19. A method that calls itself. One or more base cases (the simplest case) are used to stop recursion. Every recursive call reduces the original problem, bringing it increasingly close to a base case until it becomes that case.
20. The output is 15 ($5 + 4 + 3 + 2 + 1 = 15$)
21. The first program on the left prints 5 4 3 2 1 and the second on the right prints 1 2 3 4 5.
22. Three benefits: (1) to avoid naming conflicts. (2) to distribute software conveniently. (3) To protect classes.
23. The source code .java file should be stored in anyDir\java\chapter4, and the class file may be stored in anyOtherDir\java\chapter4. The .java file and .class may be in the same directory or a different directory. So anyDir and anyOtherDir may be same or different. To make the class available, add anyOtherDir in the classpath using the command
set classpath=%classpath%;anyOtherDir
24. The Math class is in the java.lang package. Any class in the java.lang package is automatically imported. So there is no need to import it explicitly.