

"Nuts and Bolts" (*)

EXTRA EXAMPLES

** variables

** primitive types

** assignment

** basic operations

** keywords

** control structures



Chapter 3 (sections 3.1–3.5; 3.8) – "Core Java" book Chapters 1+3 – "Head First Java" book Chapters 2–4 – "Introduction to Java Programming" book Chapter 2 (sections 2.1 – 2.5) – "Java in a Nutshell" book

> (*) Basic, practical details of Java.





These slides are left as practice and self-study.

Operator Precedence: Examples 1+2

```
int a = 4;
int b = 1;
int c = 3;

int result = a - b + c;

System.out.println(result);

both operators are of equal precedence in our table

: evaluate left to right!
    result = 6 (not 0)
```

```
int a = 4;
int b = 1;
int c = 3;
int result = a = b = c;
System.out.println(result);
```

both operators are of equal precedence in our table

∴ evaluate right to left!
result = 3



Operator Precedence: Examples 3 – 6

Post incrementing

```
int a = 4;
int result = a++ + a;
System.out.println("result = " + result + ", a = " + a);

a = 4;
result = a + a++;
System.out.println("result = " + result + ", a = " + a);
```

Pre incrementing

```
int a = 4;
int result = ++a + a;
System.out.println("result = " + result + ", a = " + a);

a = 4;
result = a + ++a;
System.out.println("result = " + result + ", a = " + a);
```



Examples: Moving the i++ and resulting behaviour

```
int i = 0;
do {
   i++;
   System.out.println("i = " + i);
} while (i < 3);
int i = 0;
while (i < 3) {
   i++;
   System.out.println("i = " + i);
}
int i = 3;</pre>
```

```
Generates: i = 1

i = 2

i = 3
```

```
Generates: i = 1

i = 2

i = 3
```

```
int i = 3;
do {
    System.out.println("i = " + i);
    i++;
} while (i < 3);

int i = 3;
while (i < 3) {
    System.out.println("i = " + i);
    i++;</pre>
```

Generates:

$$i = 3$$

Generates:

nothing

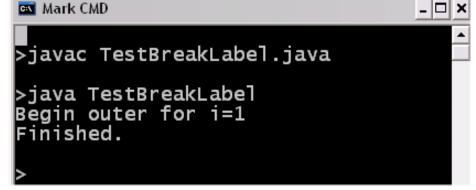


Using <u>labelled statement blocks</u> with break (1/2)

```
class TestBreakLabel {
  public static void main(String[] args) {
                                               This is just a line
    outer:
                                               marker, NOT a block.
      for (int i = 1; i < 5; i++) {
        System.out.println("Begin outer for i=" + i);
        inner:
          for (int j = 1; j < 5; j++) {
            if (j == i) break outer;
              System.out.println(" inner: i=" + i + " j= " + j);
          System.out.println("End outer for i=" + i);
      System.out.println("Finished.");
```



Use carefully as it can result in 'messy' code!





Using labelled statement blocks with break (2/2)

```
class TestBreakNoLabel {
  public static void main(String[] args) {
    outer:
      for (int i = 1; i < 5; i++) {
        System.out.println("Begin outer for i=" + i);
        inner:
                                             CMD CMD
           for (int j = 1; j < 5; j++) {
             if (j == i) break;
               System.out.println(
                         inner: i=" +
                     i + " j = " + j );
                                             Begin outer for
           System.out.println(
                "End outer for i=" + i);
                                             Begin outer for
      System.out.println("Finished.");
                                                 inner:
                Use carefully as it can
                                             Finished.
                result in 'messy' code!
```

Using <u>labelled statement blocks</u> with continue (1/2)

```
class TestContinueLabel {
  public static void main(String[] args) {
    outer:
      for (int i = 1; i < 5; i++) {
        System.out.println("Begin outer for i="+i);
         inner:
           for (int j = 1; j < 5; j++) {
                                              Mark CMD
             if (j == i) continue outer;
                                              >javac TestContinueLabel.java
               System.out.println(
                                              >java TestContinueLabel
                            inner: i=" +
                                              Begin outer for i=1
                       i + " j = " + j);
                                              Begin outer for i=2
                                              Begin outer for
           System.out.println(
                 "End outer for i=" + i);
                                              Begin outer
                                                  inner: i=4 i=2
      System.out.println("Finished.");
                                                  inner: i=4 j=3
                                              Finished.
                 Use carefully as it can
                 result in 'messy' code!
```

Using labelled statement blocks with continue (2/2)

```
class TestContinueNoLabel {
 public static void main(String[] args) {
    outer:
      for (int i = 1; i < 5; i++) {
        System.out.println(
              "Begin outer for i=" + i);
        inner:
          for (int j = 1; j < 5; j++) {
            if (j == i) continue;
              System.out.println(
                         inner: i=" +
                       i + " j = " + j );
          System.out.println(
               "End outer for i=" + i);
      System.out.println("Finished.");
                     Use carefully as it can
```

```
CMD CMD
>java TestContinueNoLabel
Begin outer for i=1
     inner: i=1 j=2
     inner: i=1 i=3
     inner: i=1 i=4
End outer for i=1
Begin outer for
     inner: i=2 i=3
End outer for i=2
Begin outer for
     inner:
            i=3
End outer for i=3
Begin outer for
End outer for i=4
```