

# Computer Science and Elements of Programming

---

## Java Lecture 5

Dr Giovanna Di Marzo Serugendo

Department of Computer Science  
and Information Systems  
Birkbeck College, University of London

Email: [dimarzo@dcs.bbk.ac.uk](mailto:dimarzo@dcs.bbk.ac.uk)

Web Page: <http://www.dcs.bbk.ac.uk/~dimarzo>

# Java Lecture 4: Review

- Control statements
- Selection (choices) statements
  - `if`
  - `if - else`
  - `switch`
- Nested `if`

# Java Lecture 5: Overview

- Control Structures
- Loops
- Loops in Java
  - `for ()`
  - `while ()`
  - `do .. while`
- Nested Loops
- `y++` and `++y`
- To learn more on Java

# Control Statements

- Instruction flow

- Sequential

```
public static void main(String[] args)
{
    String s;
    s = "Hello World";
    System.out.println(s);
}
```



- Control statement

- **Disruption** to the sequential flow

# Control Statements

- Selection statements (choices) (Lecture 4)
  - `if`, `if-else`, `switch`
- Iteration statements (loops)
  - `while`, `do-while`, `for`
- Transfer statements
  - `break`, `continue`, `return`,  
`try-catch-finally`

# Iteration Statements

- Loops
  - for executing a block of statements **repeatedly**
- **Loop body**
  - Block of statements to repeat
- **Loop condition**
  - Boolean condition to indicate when to stop the loop
  - Usually contains a variable

# while Statement

```
while ( <loop condition> )  
    <loop body>
```

- Loop condition is evaluated **before** executing the loop body
- Loop body is executed **as long as** the condition evaluates to true
- Loop body is executed only if condition is true
  - Loop body may not be executed at all
    - if condition is false first time it is evaluated

# while Statement

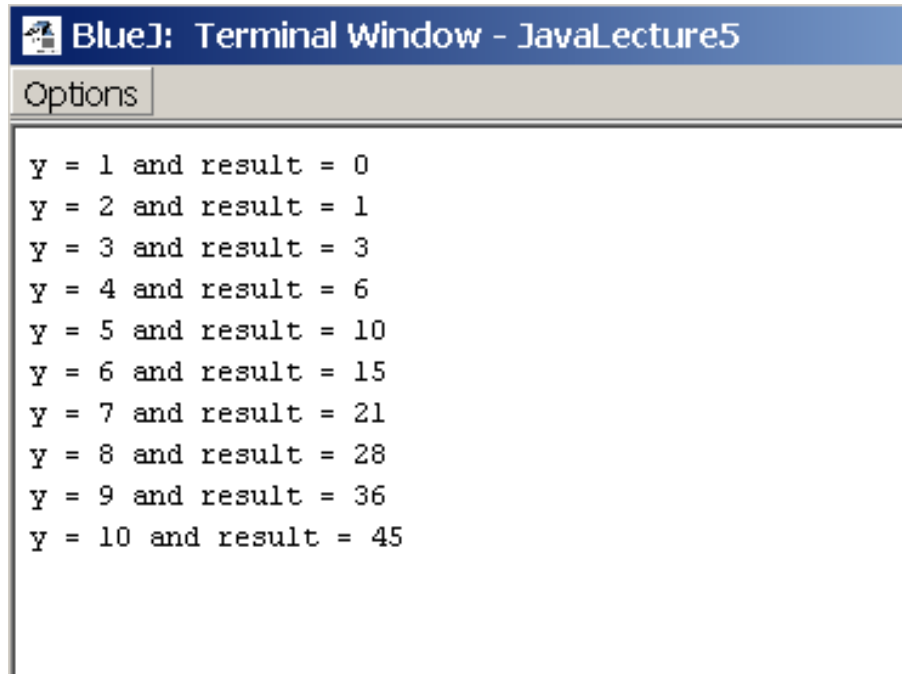
```
while ( <loop condition> )  
    <loop body>
```

```
public static void sumOfInt(){  
    int y = 0;  
    int result = 0;  
    while (y < 10) {  
        result = result + y;  
        y++; // similar to y = y + 1;  
        System.out.println("y = " + y + " and result = " + result );  
    }  
}
```

Loop condition: Must evaluate to true

Loop Body:  
Executed only if condition is true  
(may never be executed)

# while Statement



The screenshot shows a terminal window titled "BlueJ: Terminal Window - JavaLecture5". The window contains the following output:

```
y = 1 and result = 0  
y = 2 and result = 1  
y = 3 and result = 3  
y = 4 and result = 6  
y = 5 and result = 10  
y = 6 and result = 15  
y = 7 and result = 21  
y = 8 and result = 28  
y = 9 and result = 36  
y = 10 and result = 45
```

# Loop Variable

- Involved in the loop condition
- Modified in the loop body (otherwise infinite loops)

```
public static void sumOfInt(){
    int y = 0;      ← Loop variable
    int result = 0;

    while (y < 10) { ← Loop condition
        result = result + y;
        y++; // similar to y = y + 1; ← Modification of loop variable
        System.out.println("y = " + y + " and result = " + result );
    }
}
```

# while Statement

```
while ( <loop condition> )  
    <loop body>
```

```
public static void sumOfInt2(int x){
```

```
    int y = 0;
```

```
    int result = 0;
```

Must evaluate to true

```
    while (y < x) {
```

```
        result = result + y;
```

```
        y++; // similar to y = y + 1;
```

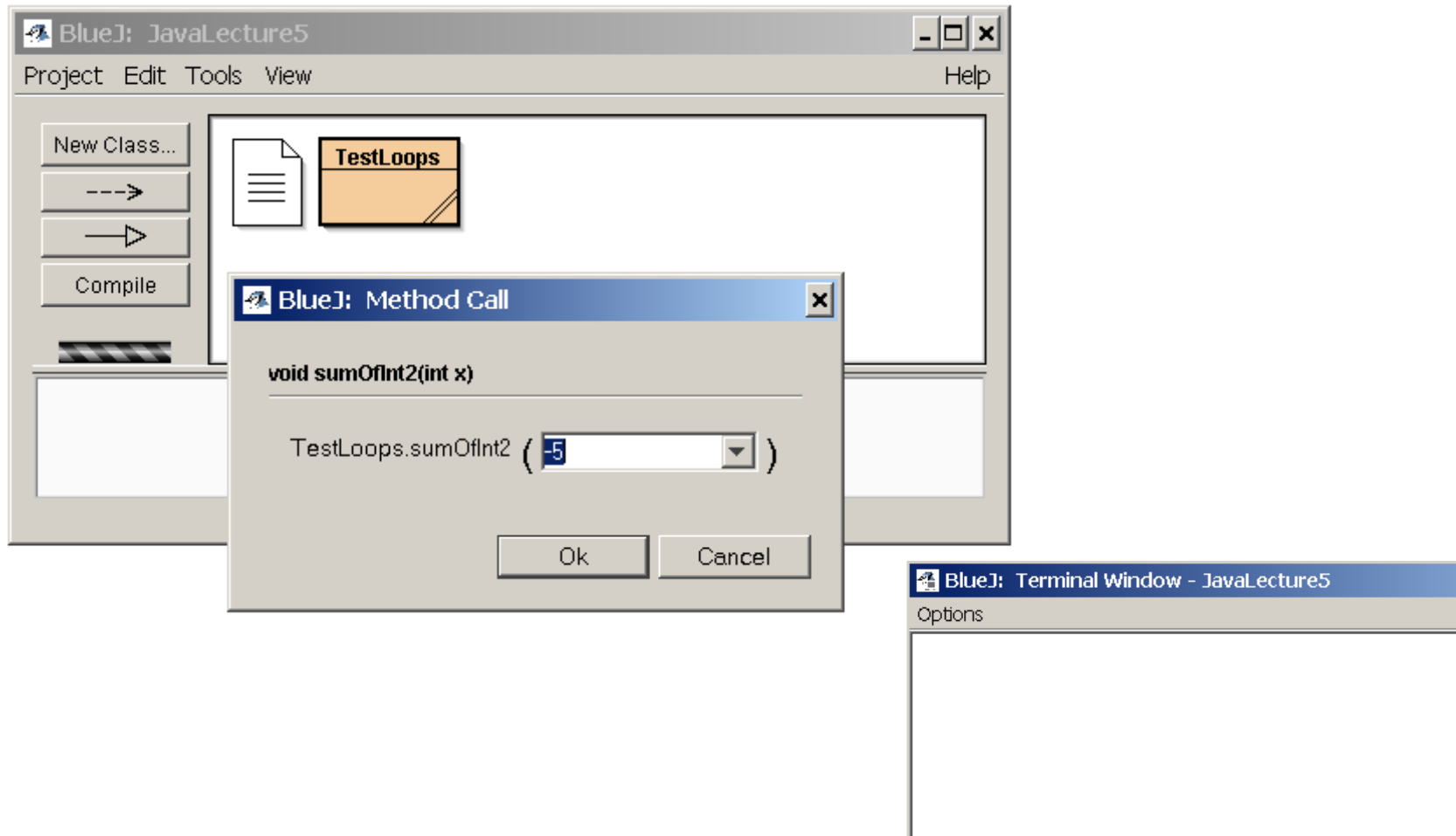
```
        System.out.println("y = " + y + " and result = " + result );
```

```
    }
```

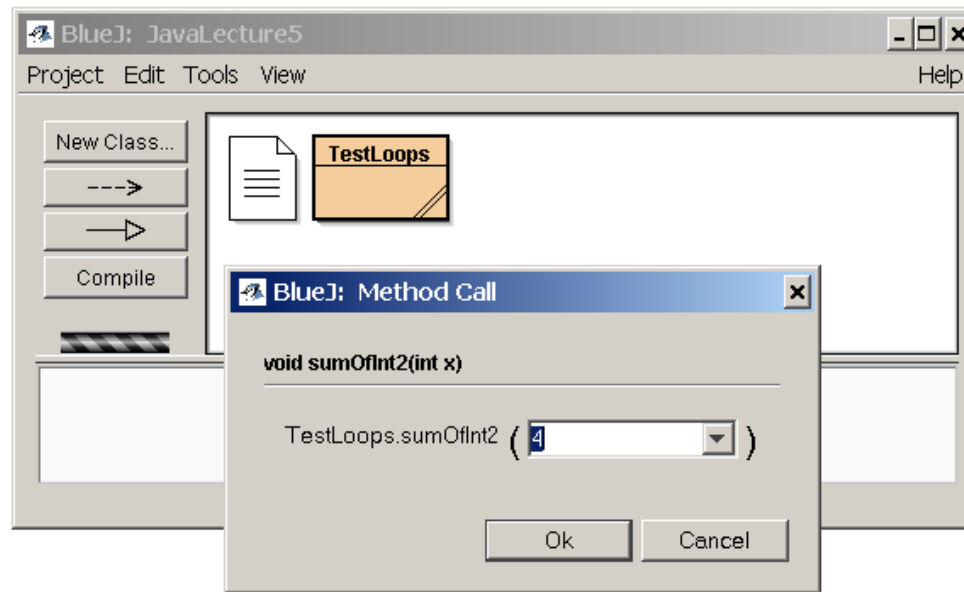
```
}
```

Executed only if condition is true  
(may never be executed)

# while Statement



# while Statement



```
BlueJ: Terminal Window - JavaLecture5
Options
y = 1 and result = 0
y = 2 and result = 1
y = 3 and result = 3
y = 4 and result = 6
```

# while Statement

- About the use of { and of ;

```
public static void sumOfInt3(int x){
    int y = 0;
    int result = 0;

    while (y < x)
        result = result + y;
        y++; // similar to y = y + 1;
}
```

```
public static void sumOfInt4(int x){
    int y = 0;
    int result = 0;

    while (y < x);
        result = result + y;
        y++; // similar to y = y + 1;
}
```

No { } End of while

End of while

- Result is: “Infinite Loop”

# do-while Statement

```
do
    <loop body>
while ( <loop condition> )
```

- Loop condition is evaluated **after** executing the loop body
- Loop body is executed **as long as** the condition evaluates to true
- Loop body is executed **at least once**
  - Even if condition is false first time

# do-while Statement

do

<loop body>

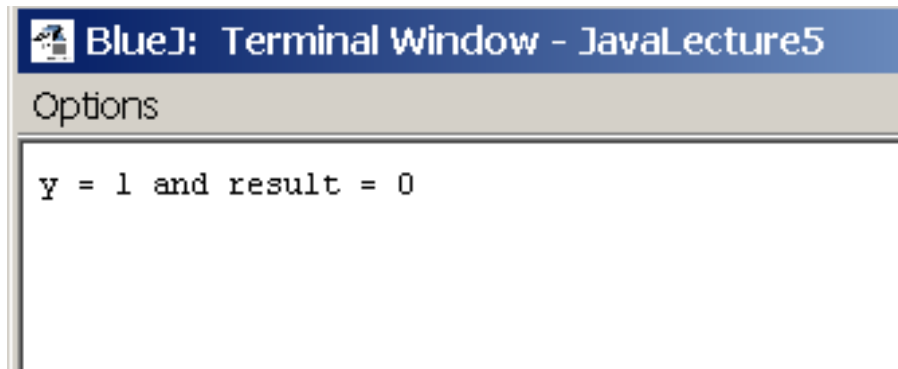
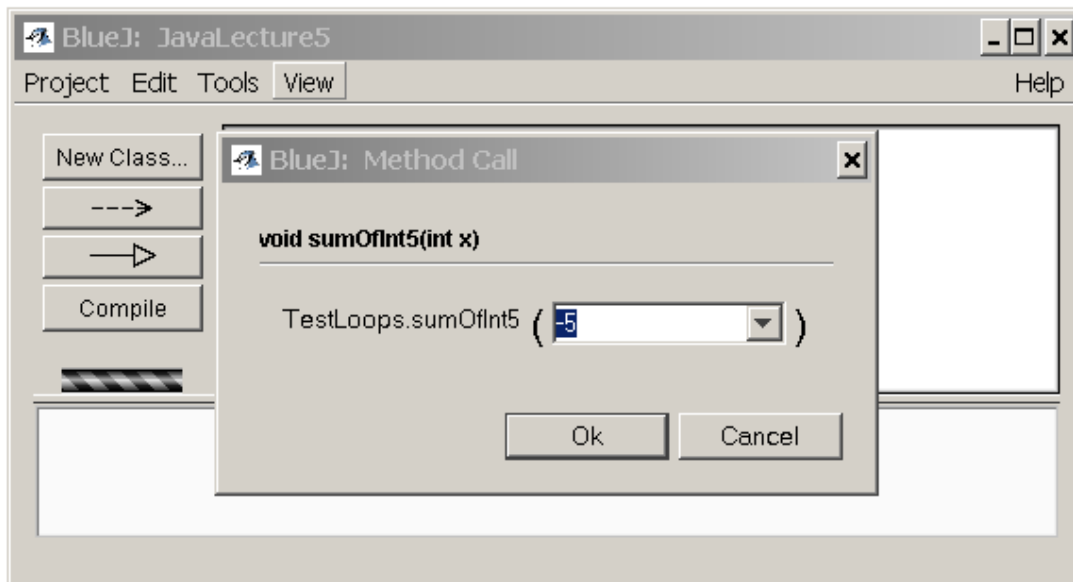
while ( <loop condition> )

```
public static void sumOfInt5(int x){  
    int y = 0;  
    int result = 0;  
  
    do {  
        result = result + y;  
        y++;  
        System.out.println("y = " + y + " and result = " + result );  
    } while (y < x);  
}
```

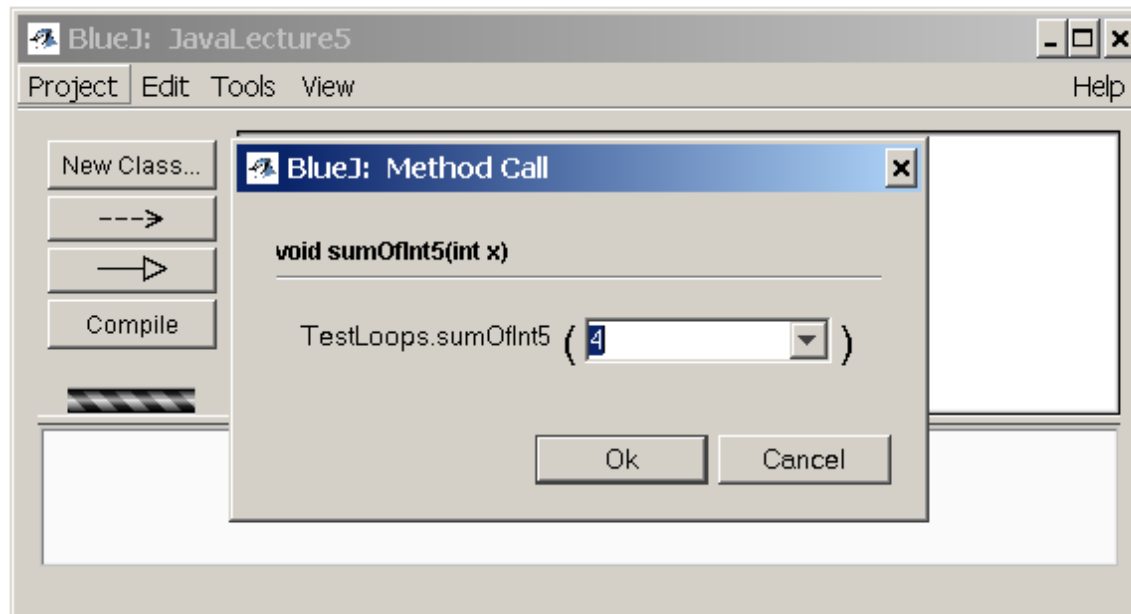
Loop Body: Executed at least once  
(even if condition is false)

Loop condition: Must evaluate to true

# do-while Statement



# do-while Statement



The screenshot shows a terminal window titled "BlueJ: Terminal Window - JavaLecture5". The output of the `sumOfInt5` method is displayed as follows:

```
Options
y = 1 and result = 0
y = 2 and result = 1
y = 3 and result = 3
y = 4 and result = 6
```

# Loop Variable

- Involved in the loop condition
- Modified in the loop body (otherwise infinite loops)

```
public static void sumOfInt5(int x){
    int y = 0;      ← Loop variable
    int result = 0;

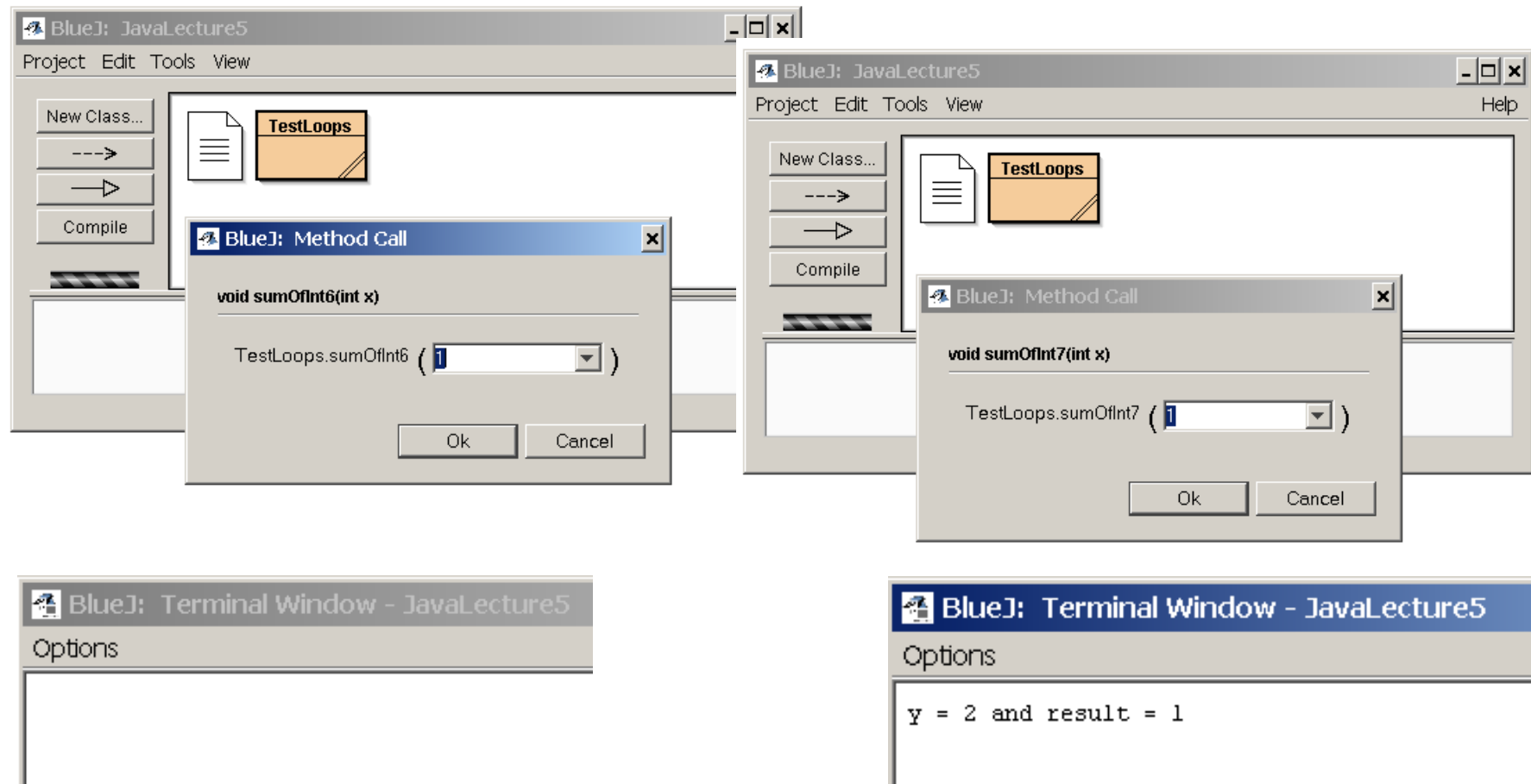
    do {
        result = result + y;
        y++;       ← Modification of loop variable
        System.out.println("y = " + y + " and result = " + result );
    } while (y < x); ← Loop condition
}
```

# while and do-while Statements

```
public static void sumOfInt6(int x){  
    int y = 1;  
    int result = 0;  
  
    while (y < x) {  
        result = result + y;  
        y++;  
        System.out.println("y = " + y + "  
            and result = " + result );  
    }  
}
```

```
public static void sumOfInt7(int x){  
    int y = 1;  
    int result = 0;  
  
    do {  
        result = result + y;  
        y++; // similar to y = y + 1;  
        System.out.println("y = " + y + "  
            and result = " + result );  
    } while (y < x);  
}
```

# while and do-while Statements



# for Statement

```
for ( <initialisation> ; <loop condition> ;  
      <increment expression> )  
    <loop body>
```

- **<initialisation>**
  - Declares and initialises loop variable
  - Only executed once
- **<loop condition>**
  - Boolean expression (must be true)
- **<increment expression>**
  - Modifies the value of the loop variable

# for Statement

```
for ( <initialisation> ; <loop condition> ;  
      <increment expression> )  
  <loop body>
```

```
public static void sumOfInt8(int x){  
    int result = 0;  
    for (int y = 1; y < x; y++){  
        result = result + y;  
        System.out.println("y = " + y + " and result = " + result );  
    }  
}
```

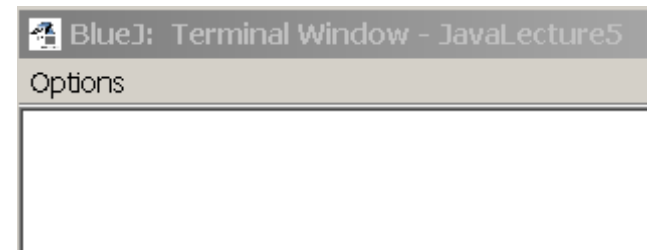
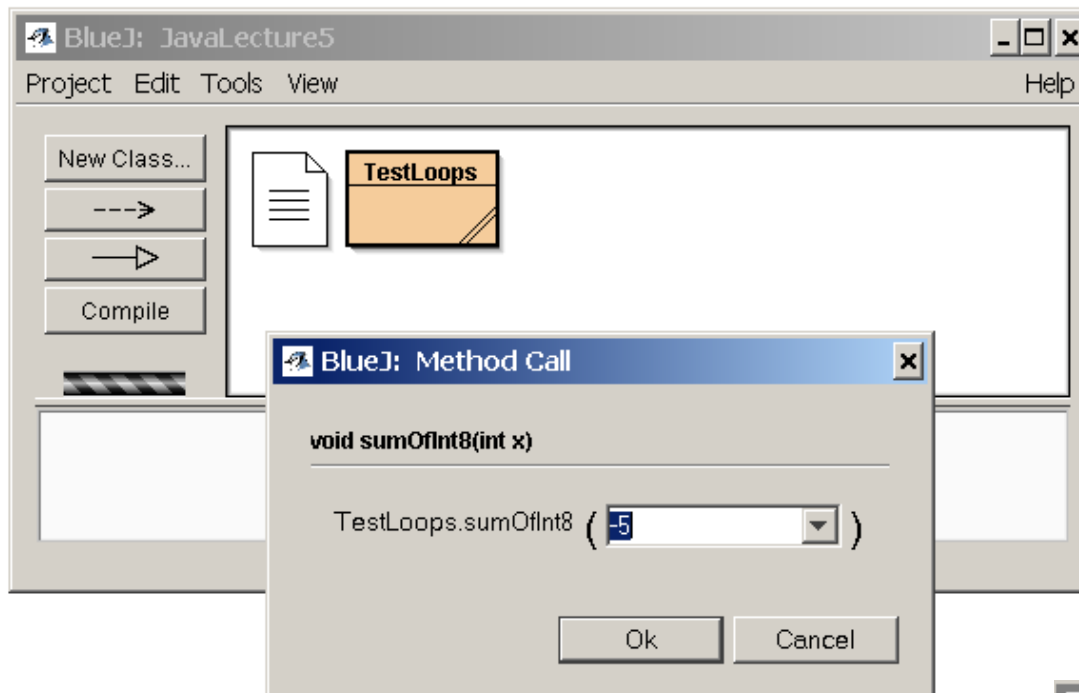
Loop variable

Loop condition

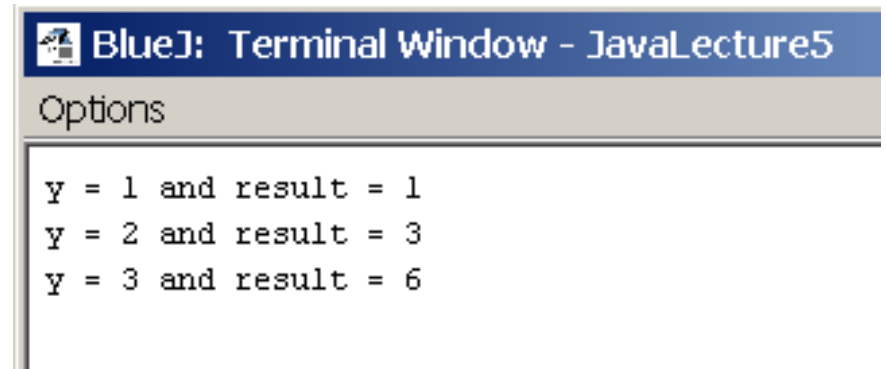
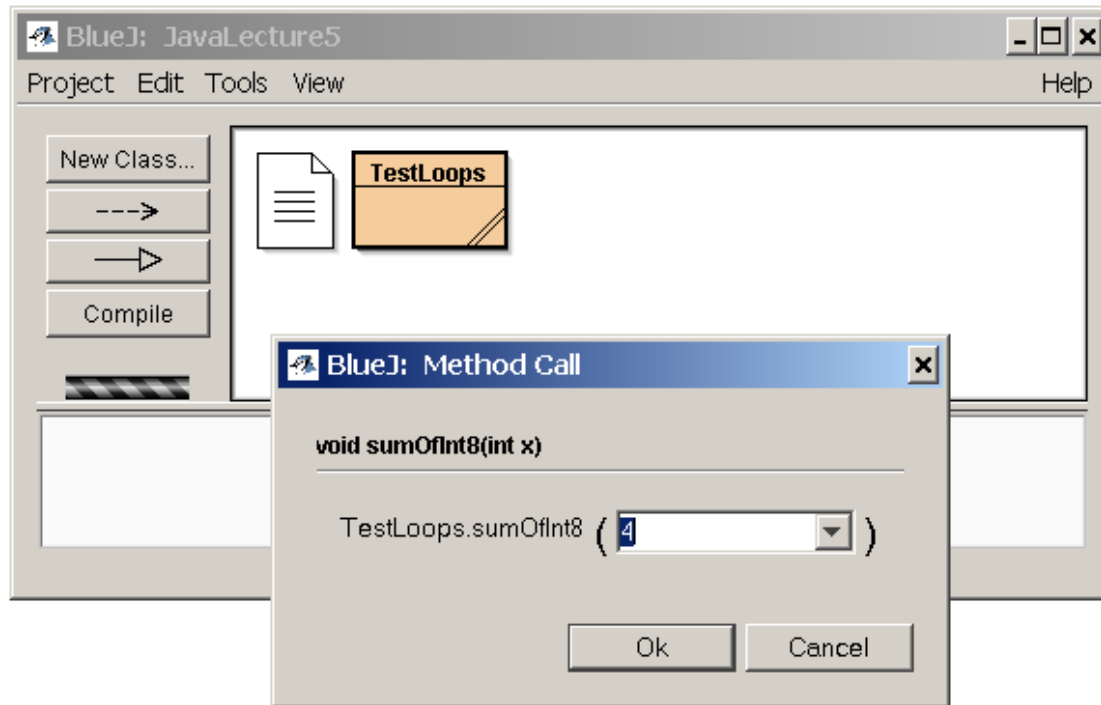
Modification of loop variable

Loop Body:  
Executed only if condition is true

# for Statement

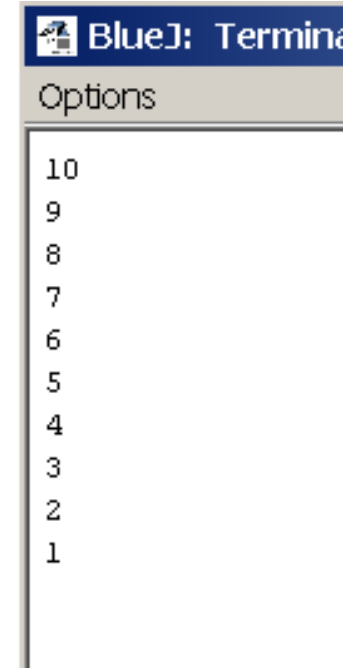
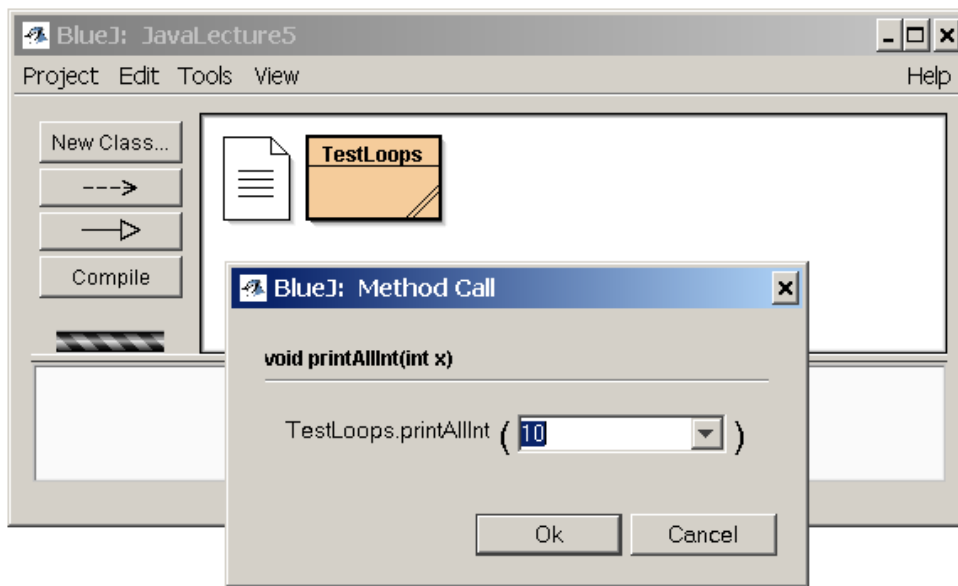


# for Statement



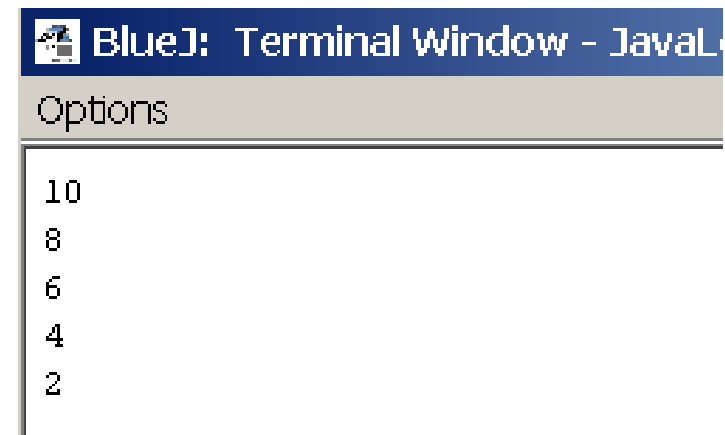
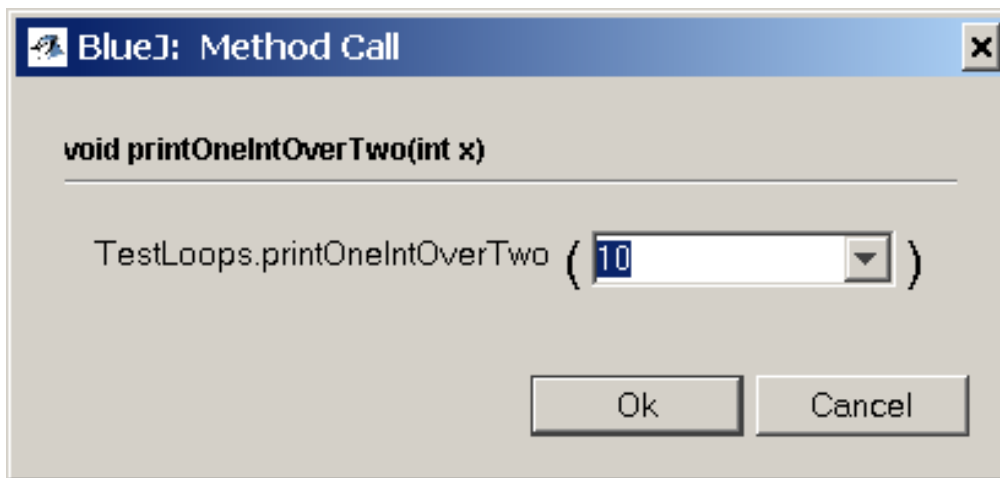
# for Statement

```
public static void printAllInt(int x){  
    for (int y = x; y > 0; y--){  
        System.out.println(y);  
    }  
}
```



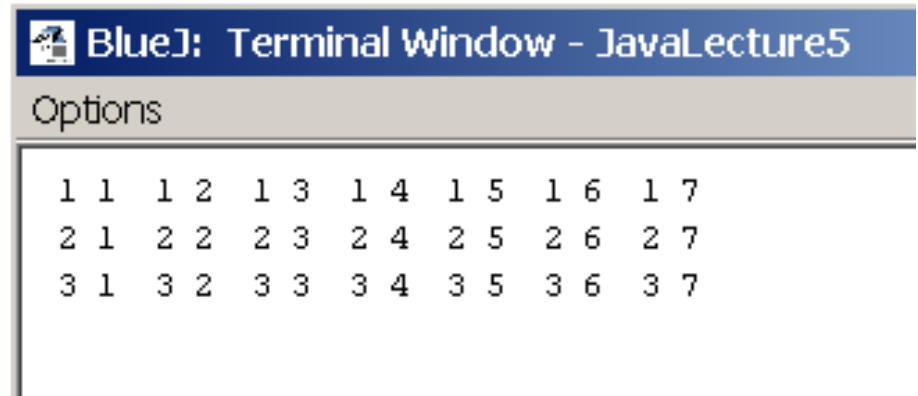
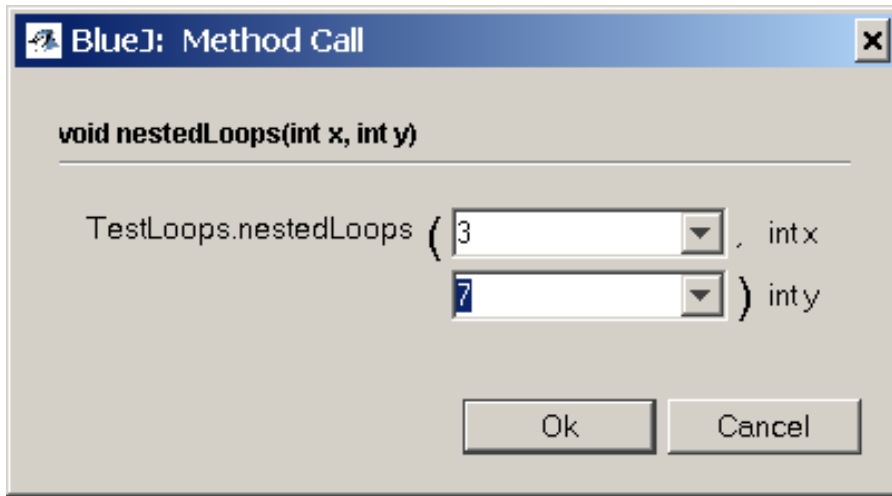
# for Statement

```
public static void printOneIntOverTwo(int x){  
    for (int y = x; y > 0; y = y-2){  
        System.out.println(y);  
    }  
}
```



# Nested Loops

```
public static void nestedLoops(int x, int y){  
    for (int i = 1; i <= x; i++){  
        for (int j = 1; j <= y; j++) {  
            System.out.print(" " + i + " " + j + " ");  
        }  
        System.out.print("\n");  
    }  
}
```



# Iteration Statements

- while
  - Used when number of iterations **is not known**
- do-while
  - Used when number of iterations **is not known**
  - Block of statements need to be done **at least once**
- for
  - Used when number of iterations **is known**

# Infinite Loops

- Infinite loops
  - When loop condition is never false
    - loop variable is not modified
    - loop variable is modified but not towards the end of the condition
  - No loop condition

```
for ( ; ; ) { ... }  
  
while (true) { }  
  
do { .. } while (true);
```

# Infinite Loops

- How to go out of infinite loops
  - No loop condition

```
for ( ; ; ) { ... }
```

```
while (true) { ... }
```

```
do { ... } while (true);
```

```
for ( ; ; ) { ... ; if (...) break; ... }
```

```
while (true) { ... ; if (...) break; ... }
```

```
do { ... ; if (...) break; ... } while (true);
```

# About `y++` and `++y`

```
public static void testPlusPlus(){  
    int x = 3; int y = 5; // x == 3, y == 5
```

```
    y++; // y = y + 1, y == 6  
    System.out.println("y = " + y);  
    ++y; // y = y + 1, y == 7  
    System.out.println("y = " + y);
```

```
    x = y++; // x = y; y = y + 1; (x == 7, y == 8)  
    System.out.println("x = " + x + " and y = " + y);  
    x = ++y; // y = y + 1; x = y; (x == 9, y == 9)  
    System.out.println("x = " + x + " and y = " + y);  
}
```

 BlueJ: Terminal Window - JavaLecture5

Options

```
y = 6  
y = 7  
x = 7 and y = 8  
x = 9 and y = 9
```

# How to learn more on Java

- Java Books

- <http://java.sun.com/docs/books/>

- Java Sun Tutorial

- <http://java.sun.com/docs/books/tutorial/>

- Java API

- <http://java.sun.com/j2se/1.5.0/docs/api/>