

**Intro to Computer Science**  
CSCI-UA-0101

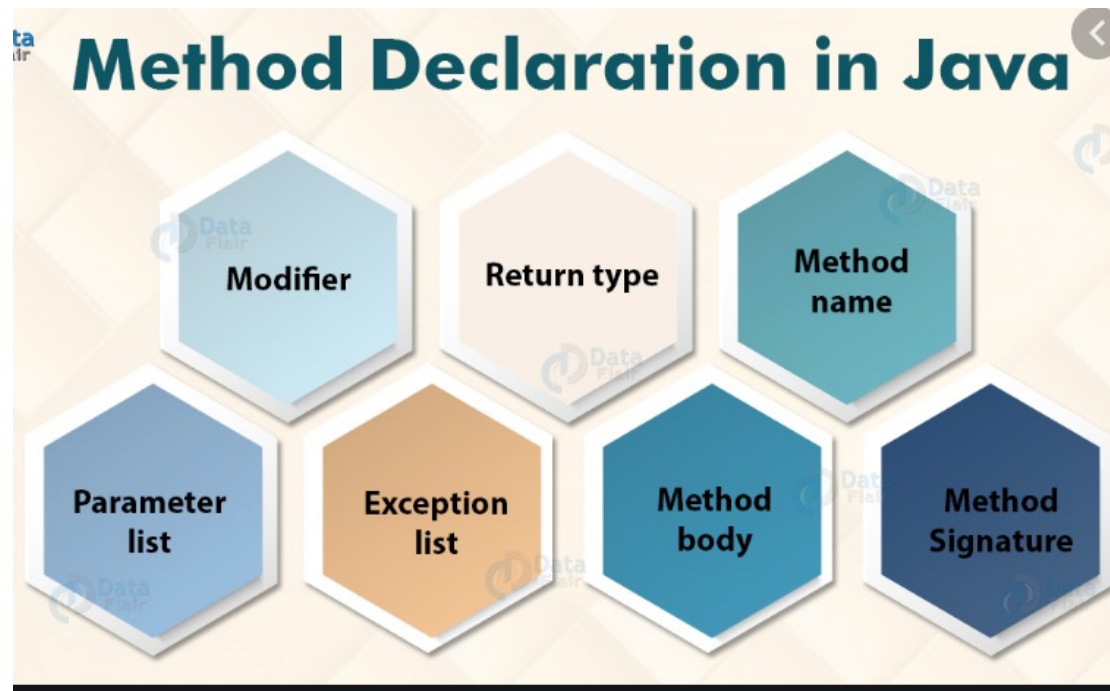
## **User defined Methods**

Readings:

Chapter 6 from the liang book and chapter 4 [in "Think Java, Second Edition"](#)

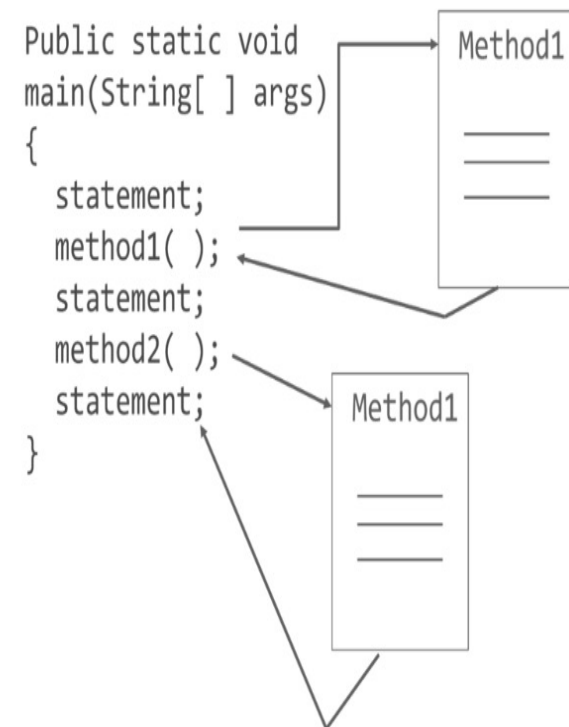
Prof. Sana Odeh  
sana@nyu.edu

# User-defined Methods in Java



# Why Methods

- The concept of method abstraction can be applied to the process of developing programs. When writing a large program, you can use the “**divide and conquer**” strategy, also known as **stepwise refinement**, to decompose it into subproblems.
- The subproblems can be further decomposed into smaller, more manageable problems.
- Methods allows you to manage programs better and **modelize your program (divide into small pieces so you can reuse and also manage your program better**
- A method is a collection of statements that are grouped together to perform an operation/one action/one task similar to function.



# Defining Methods

A method is a collection of statements that are grouped together to perform an one action. Helps modalize the program and allows for a re-use of code creating efficient programs.

Methods definition  
and body

Method Call

Define a method

formal Parameters

Returns int

If this is  
"void"  
then it  
returns  
nothing

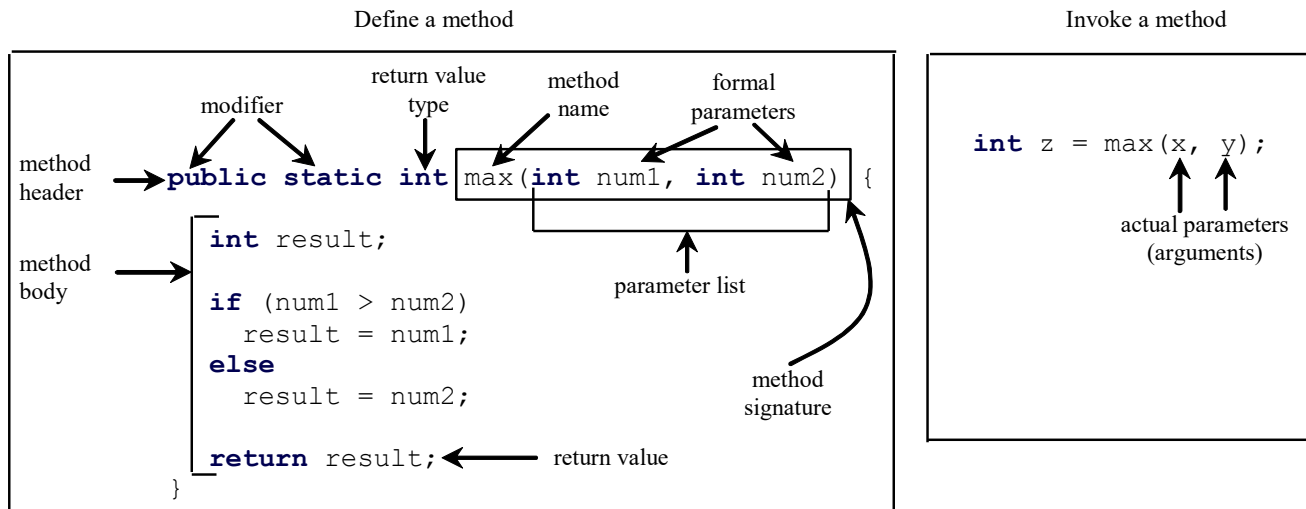
```
public static int max(int num1, int num2) {  
    int result;  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
    return result;  
}
```

Invoke a method

```
int z = max(x, y);  
        ↑ ↑  
    actual parameters  
    (arguments)
```

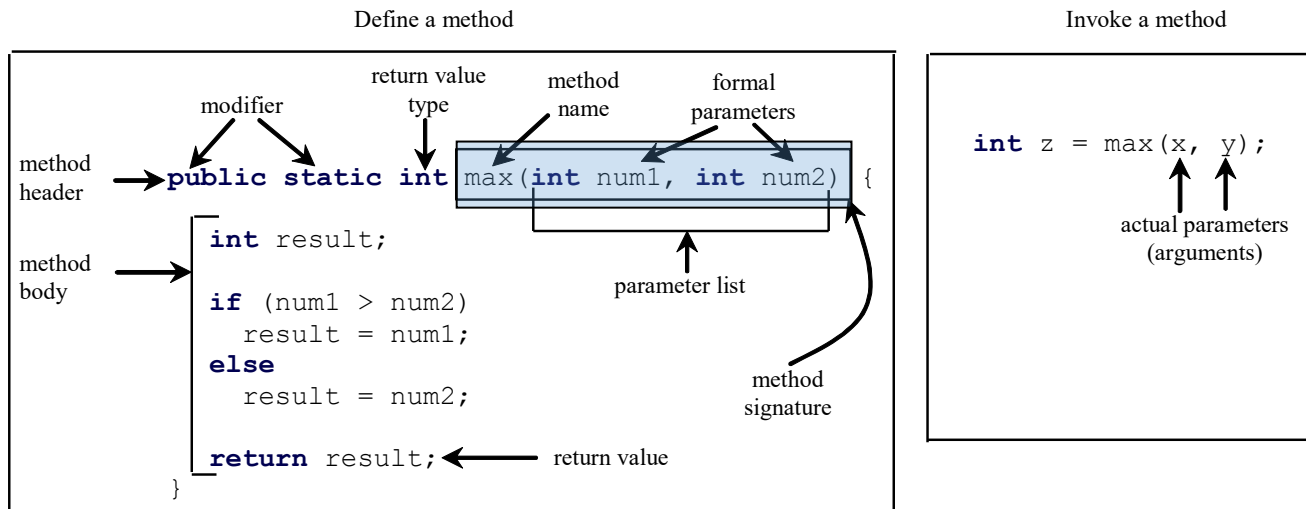
# Defining Methods

A method is a collection of statements that are grouped together to perform an operation.



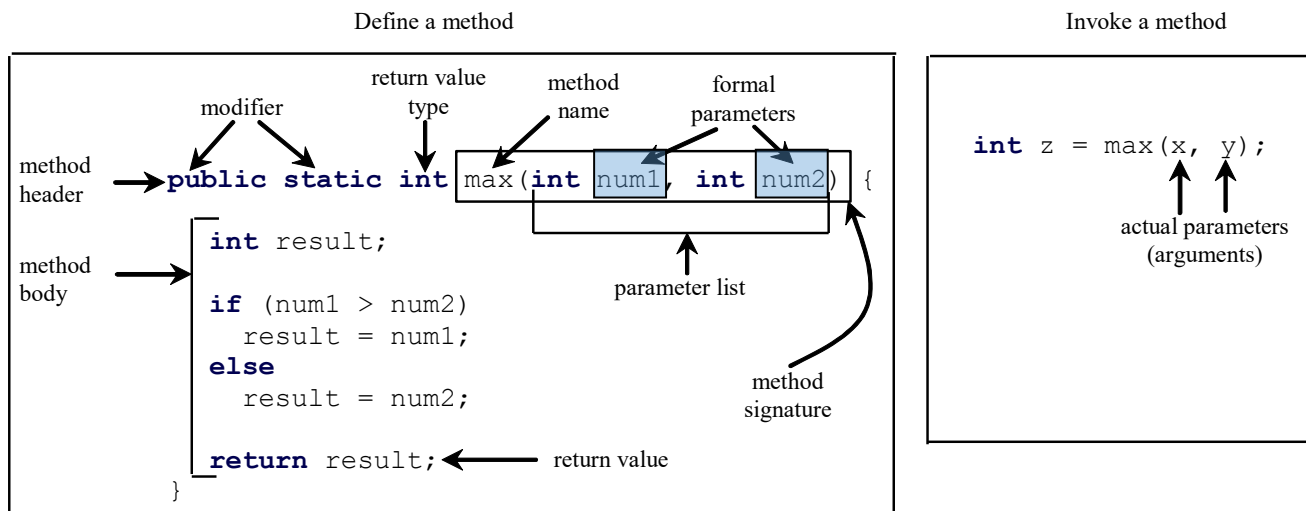
# Method Signature

*Method signature* is the combination of the method name and the parameter list.



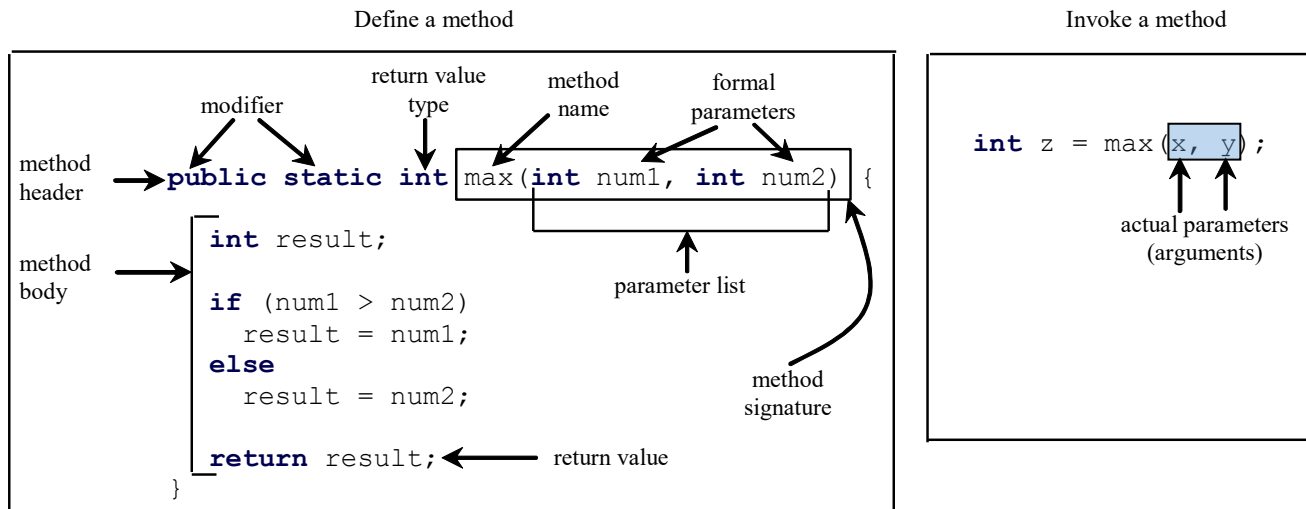
# Formal Parameters

The variables defined in the method header are known as *formal parameters*.



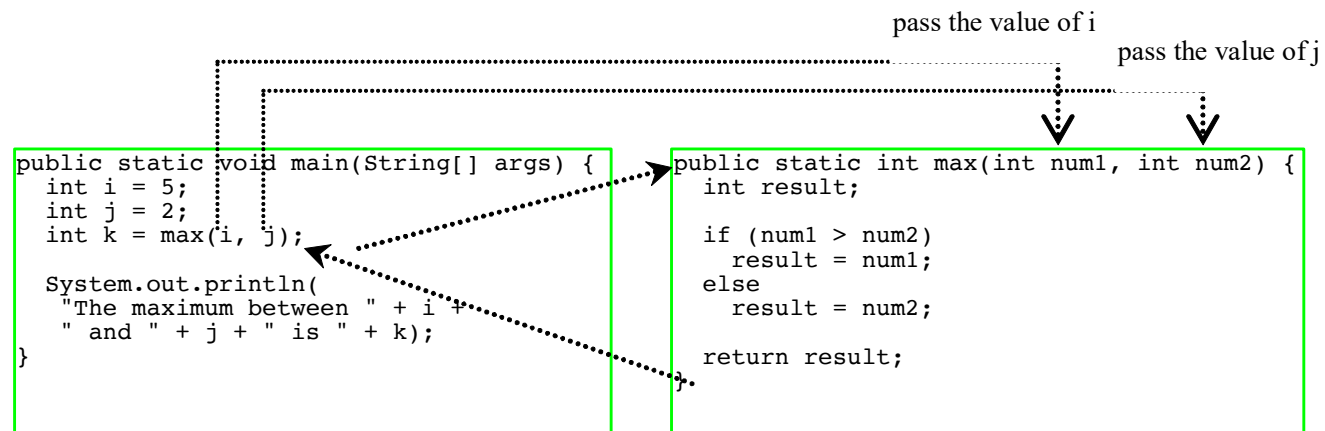
# Actual Parameters

When a method is invoked, you pass a value to the parameter. This value is referred to as *actual parameter or argument*.





## Calling Methods, cont.



animation

# Trace Method Invocation

i is now 5

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

j is now 2

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

invoke max(i, j)

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

invoke max(i, j)  
Pass the value of i to num1  
Pass the value of j to num2

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

declare variable result

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

(num1 > num2) is true since num1 is 5 and num2 is 2

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

result is now 5

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```



animation

# Trace Method Invocation

return result, which is 5

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```

animation

# Trace Method Invocation

return max(i, j) and assign the  
return value to k

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
    return result;  
}
```

animation

# Trace Method Invocation

Execute the print statement

```
public static void main(String[] args) {  
    int i = 5;  
    int j = 2;  
    int k = max(i, j);  
  
    System.out.println(  
        "The maximum between " + i +  
        " and " + j + " is " + k);  
}
```

```
public static int max(int num1, int num2) {  
    int result;  
  
    if (num1 > num2)  
        result = num1;  
    else  
        result = num2;  
  
    return result;  
}
```