Introduction to Java for C++ Programmers

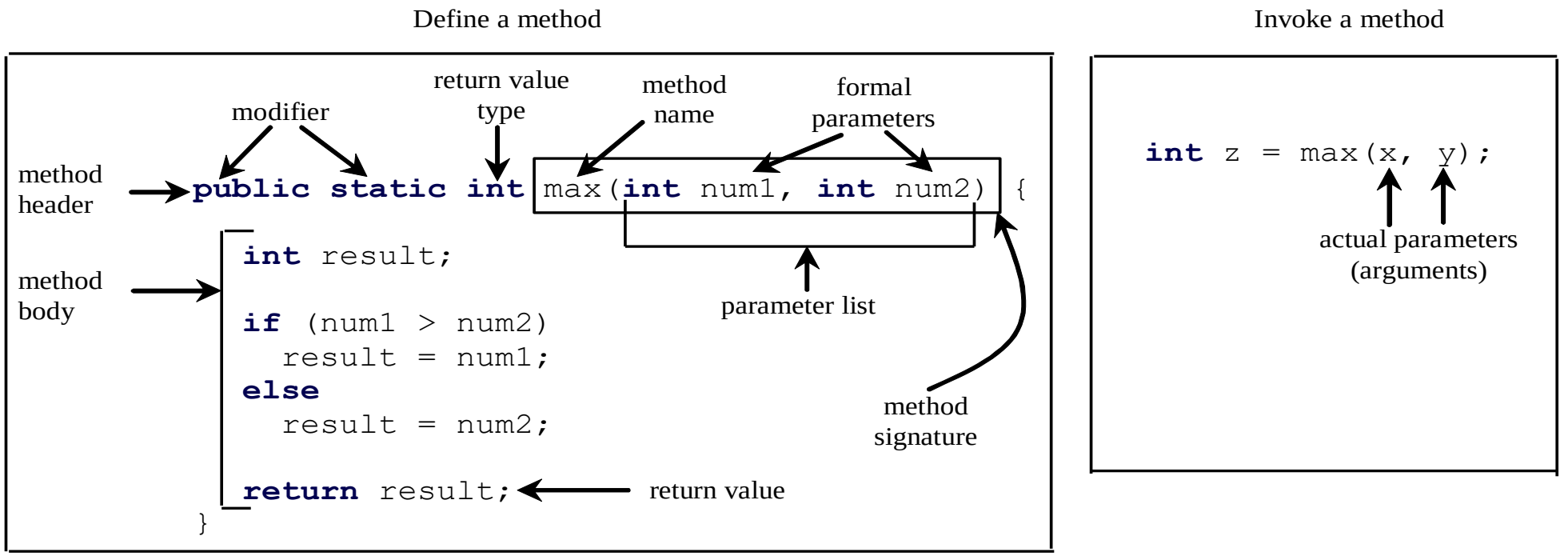
Segment - 1

JAC 444

Professor: Mahboob Ali

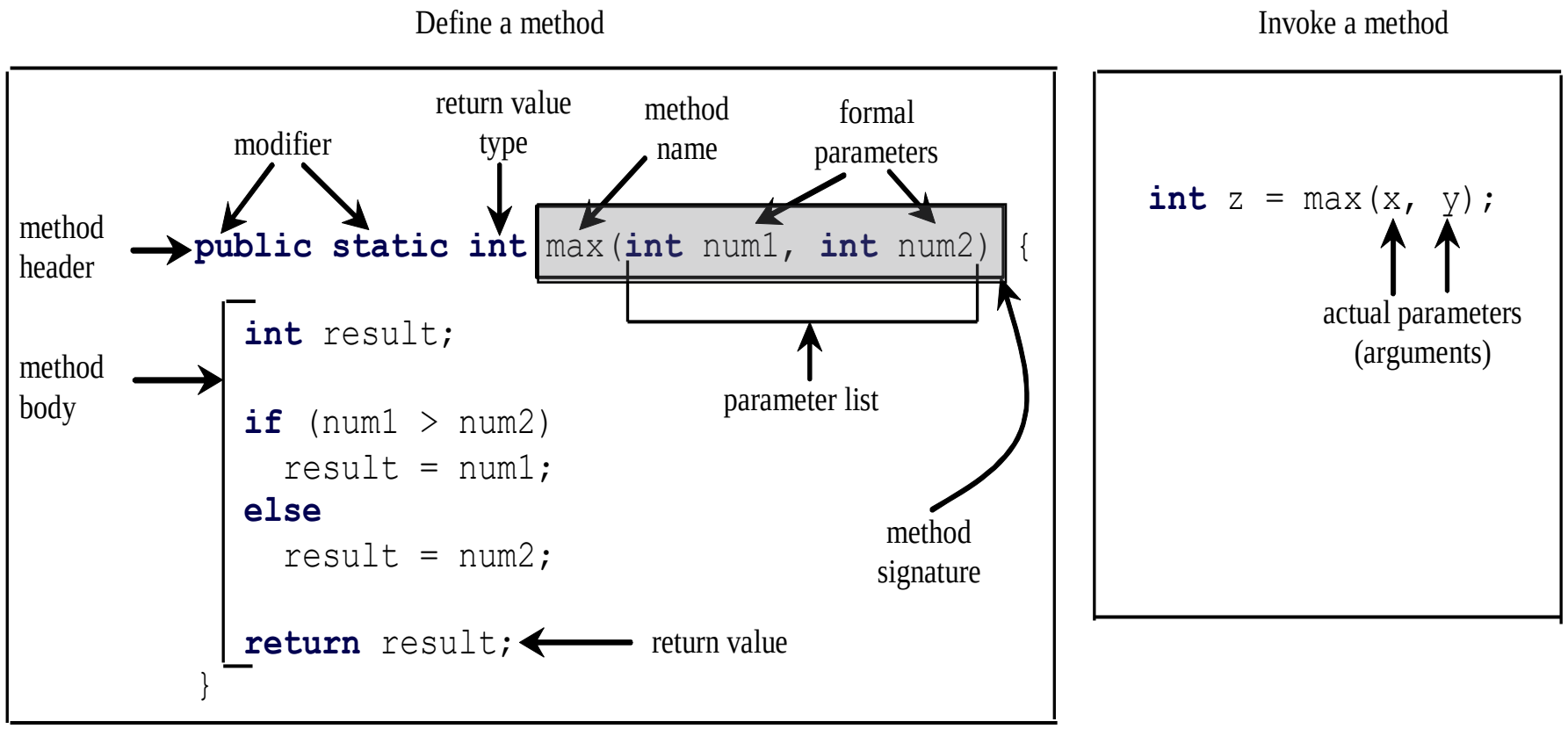
Methods

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



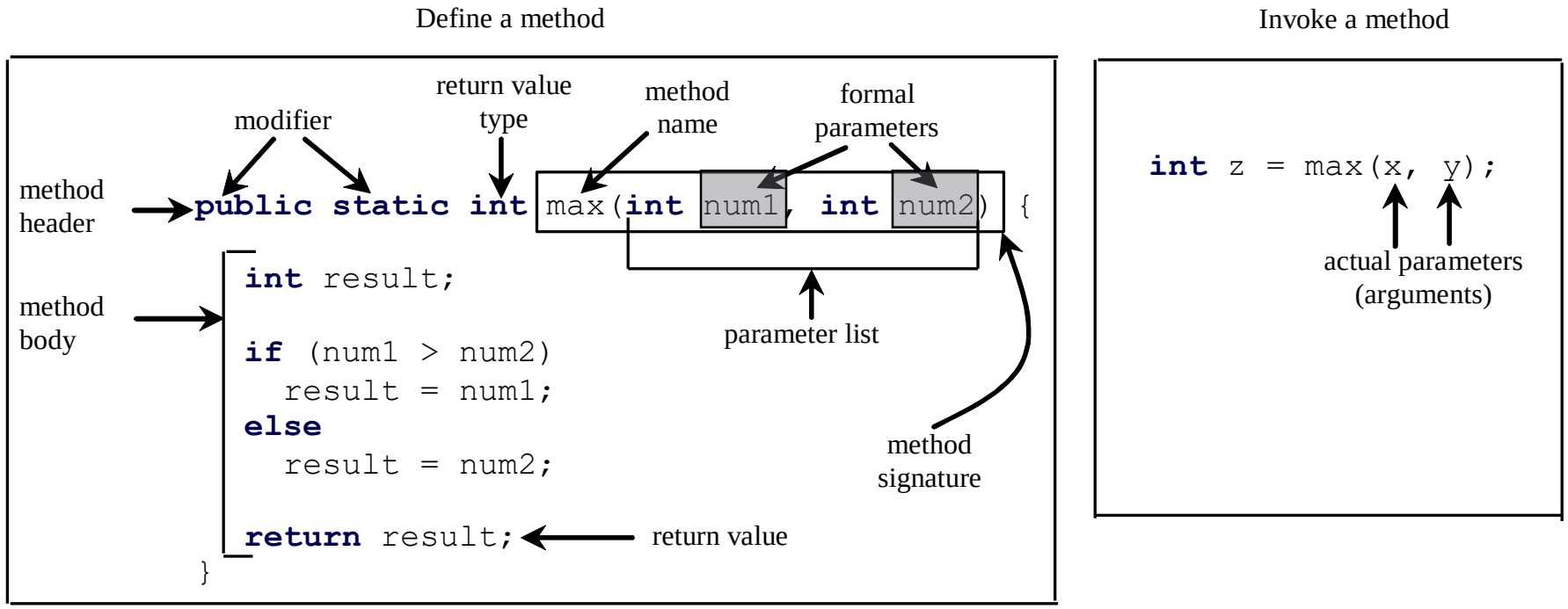
Method Signature

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



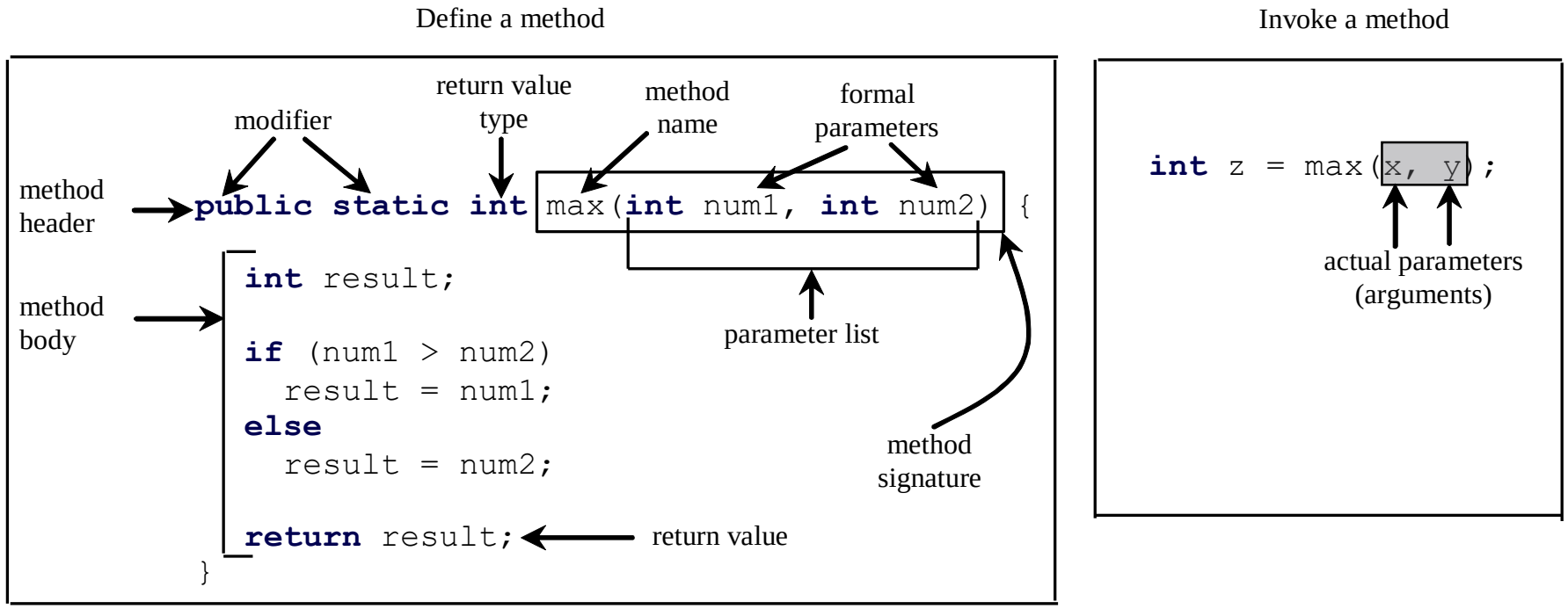
Formal Parameters

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



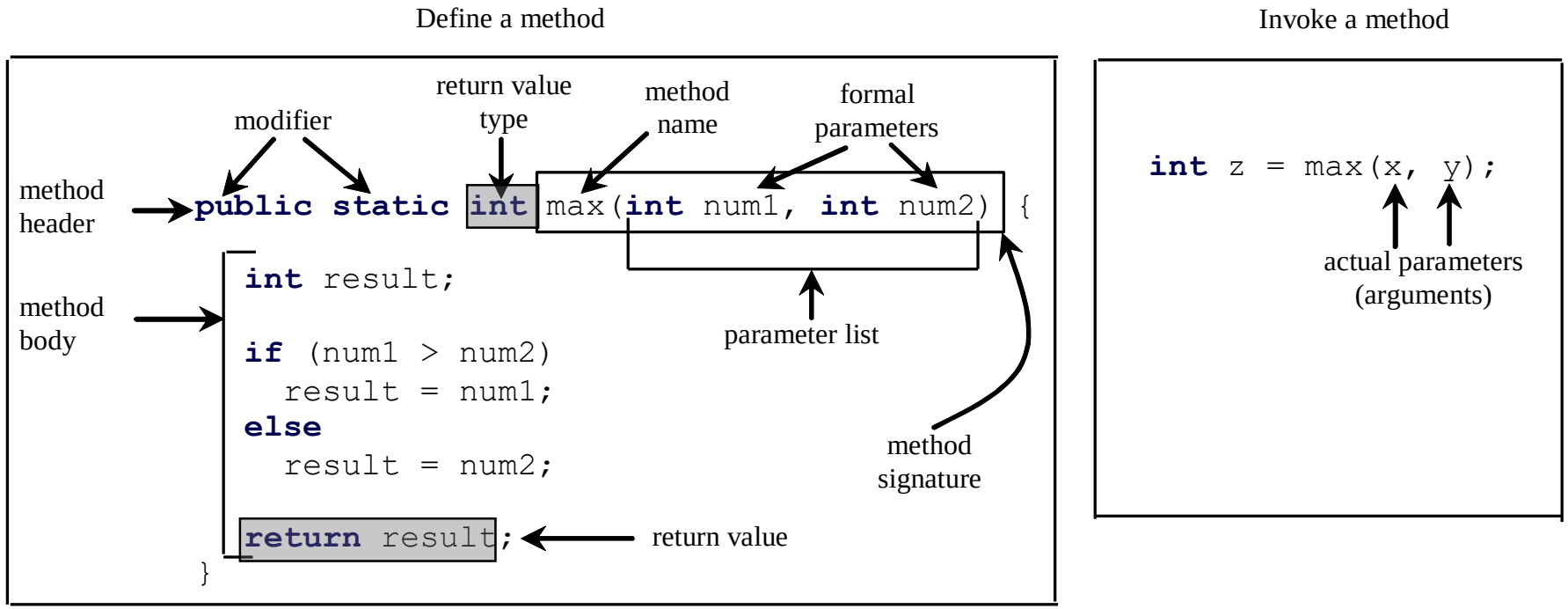
Actual Parameters

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

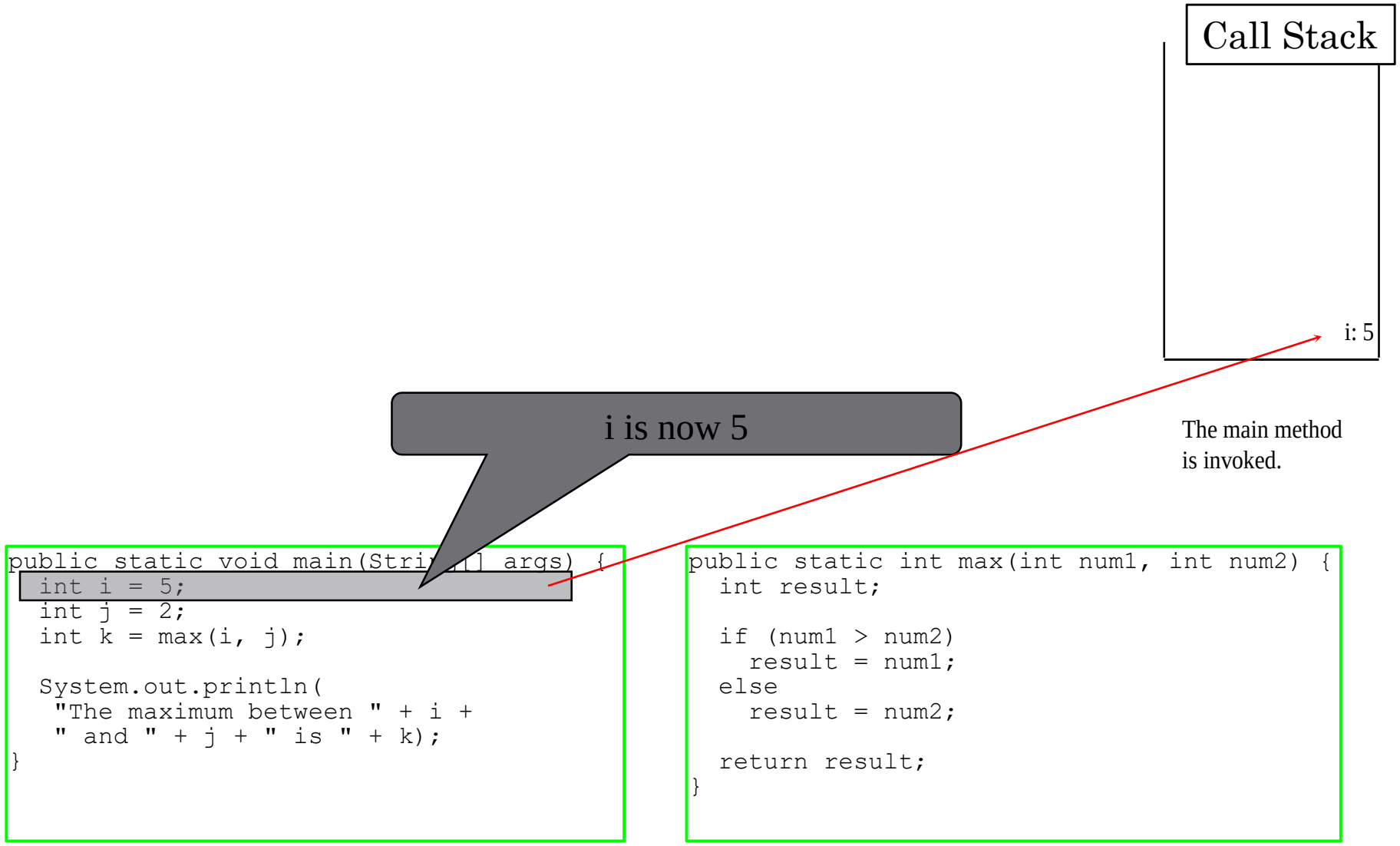


Return Value Type

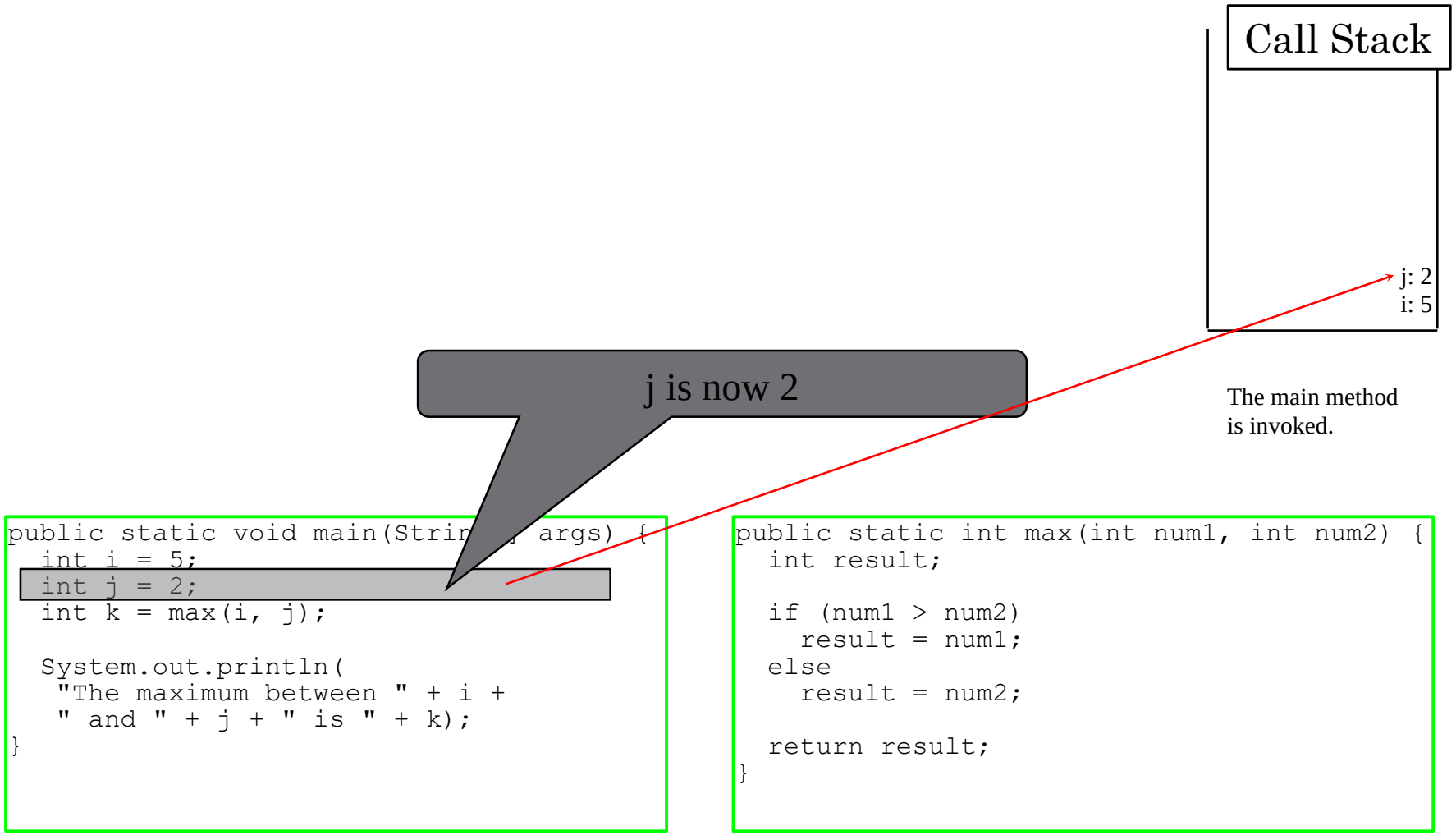
A method may return a value. The returnValueType is the data type of the value the method returns. If the method does not return a value, the returnValueType is the keyword void. For example, the returnValueType in the main method is void.



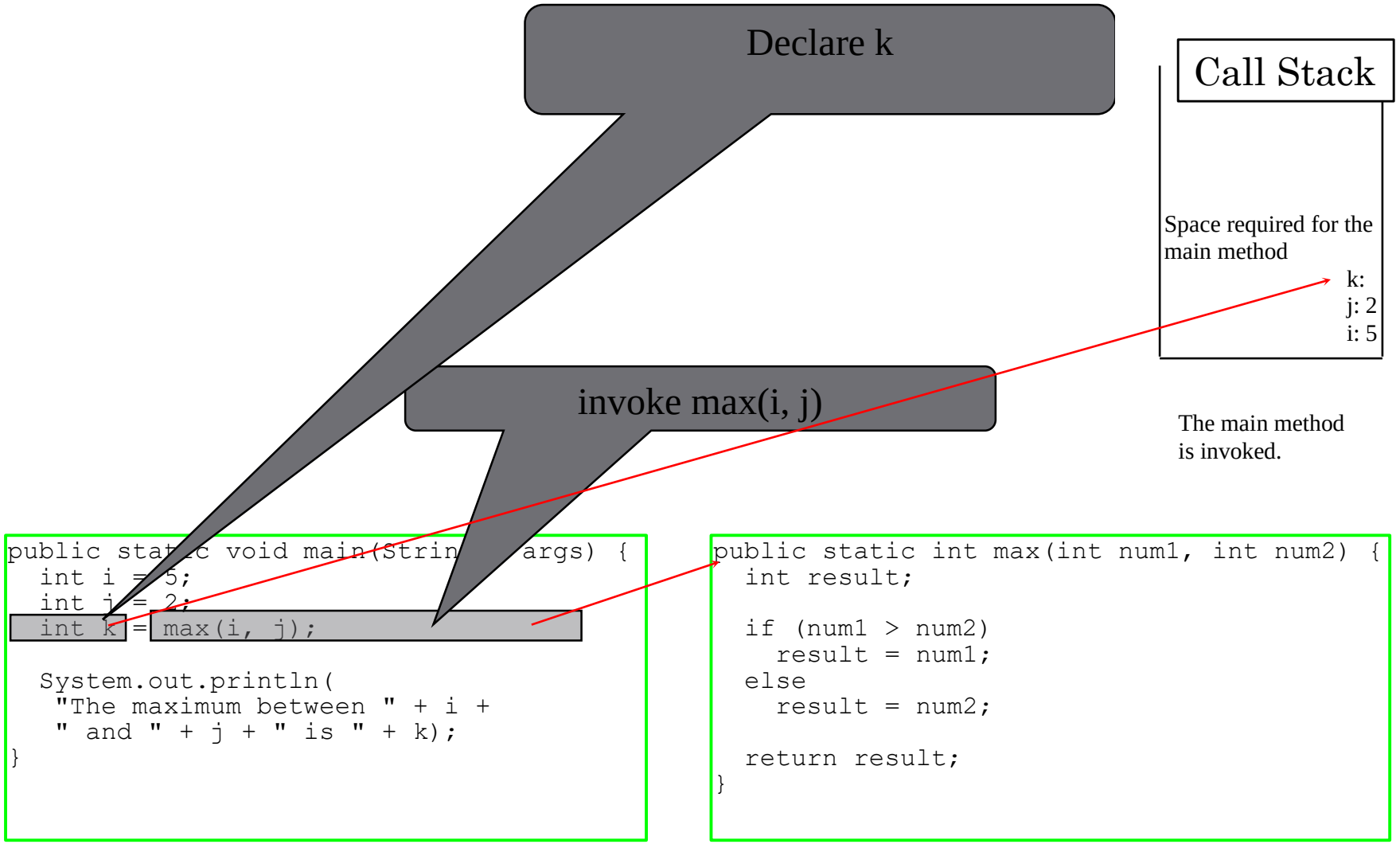
Trace Method Invocation



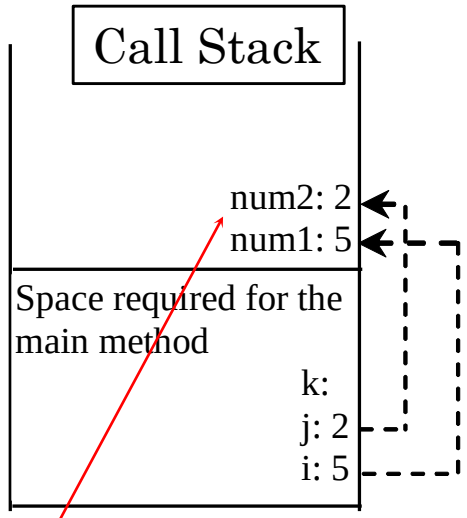
Trace Method Invocation



Trace Method Invocation



Trace Method Invocation



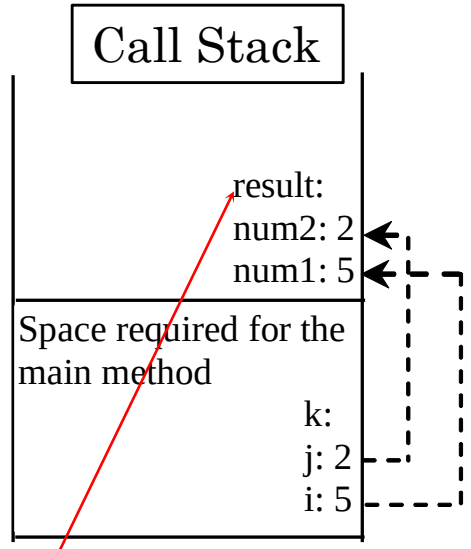
invoke max(i, j)
Pass the value of i to num1
Pass the value of j to num2

The max method is invoked.

```
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;
}
```

Trace Method Invocation



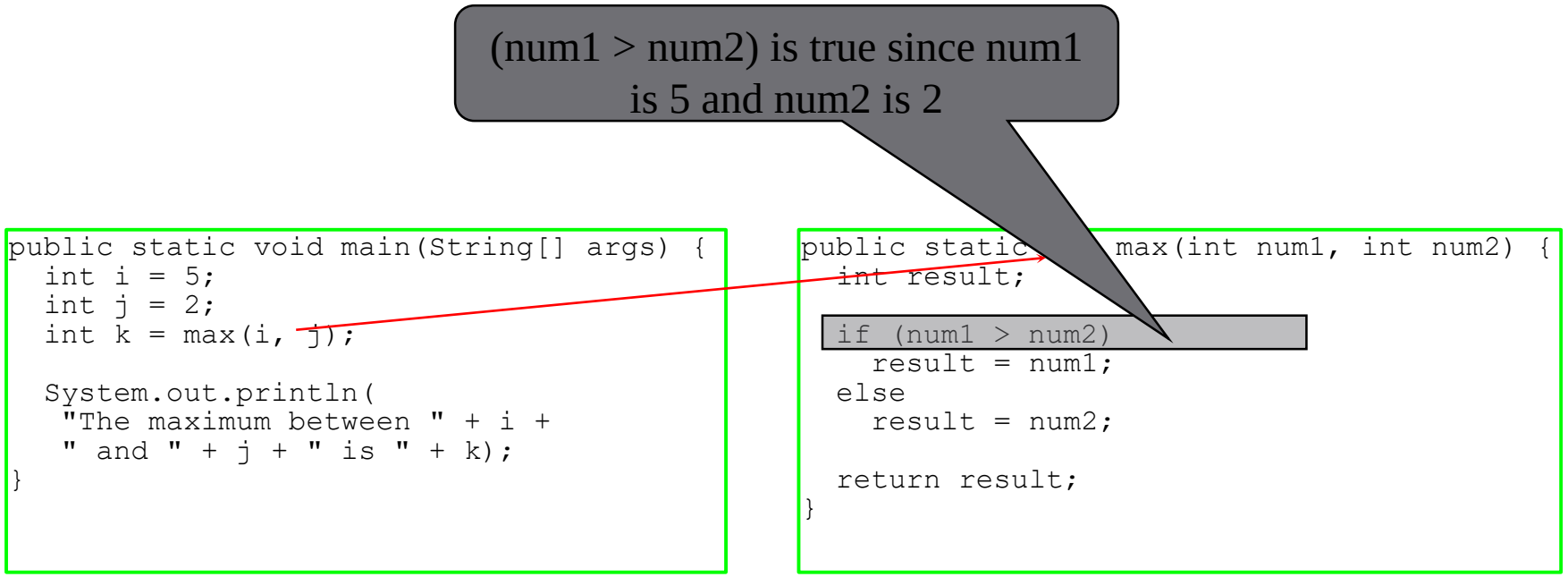
declare variable result

The max method is invoked.

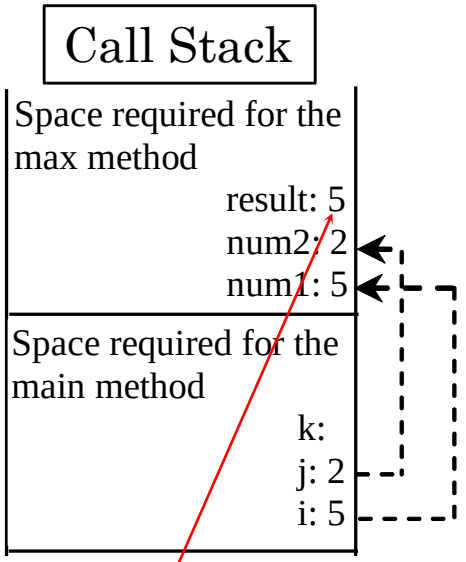
```
public static void main(String[] args) {
  int i = 5;
  int j = 2;
  int k = max(i, j);

  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;
}
```

Trace Method Invocation



Trace Method Invocation



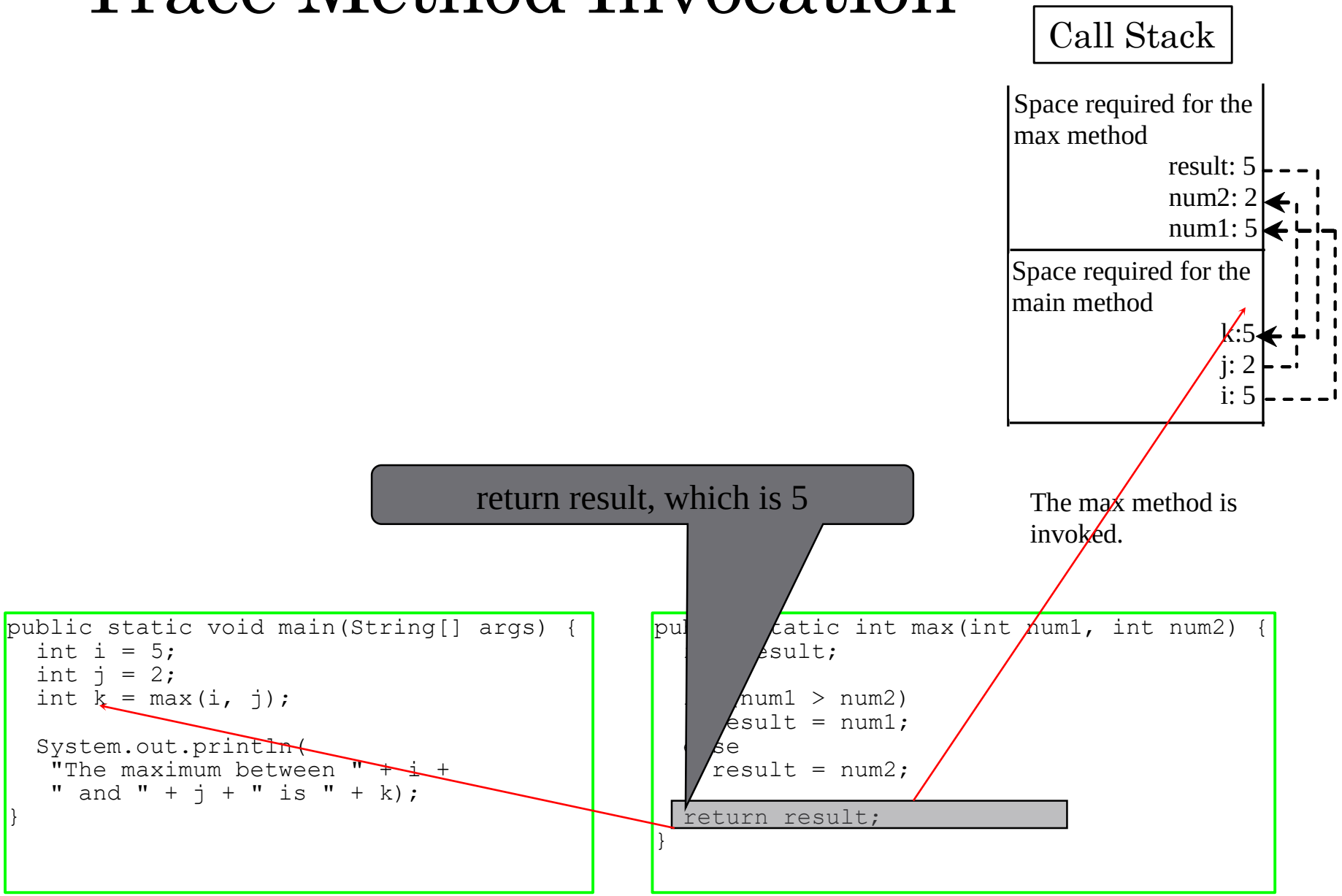
result is now 5

The max method is invoked.

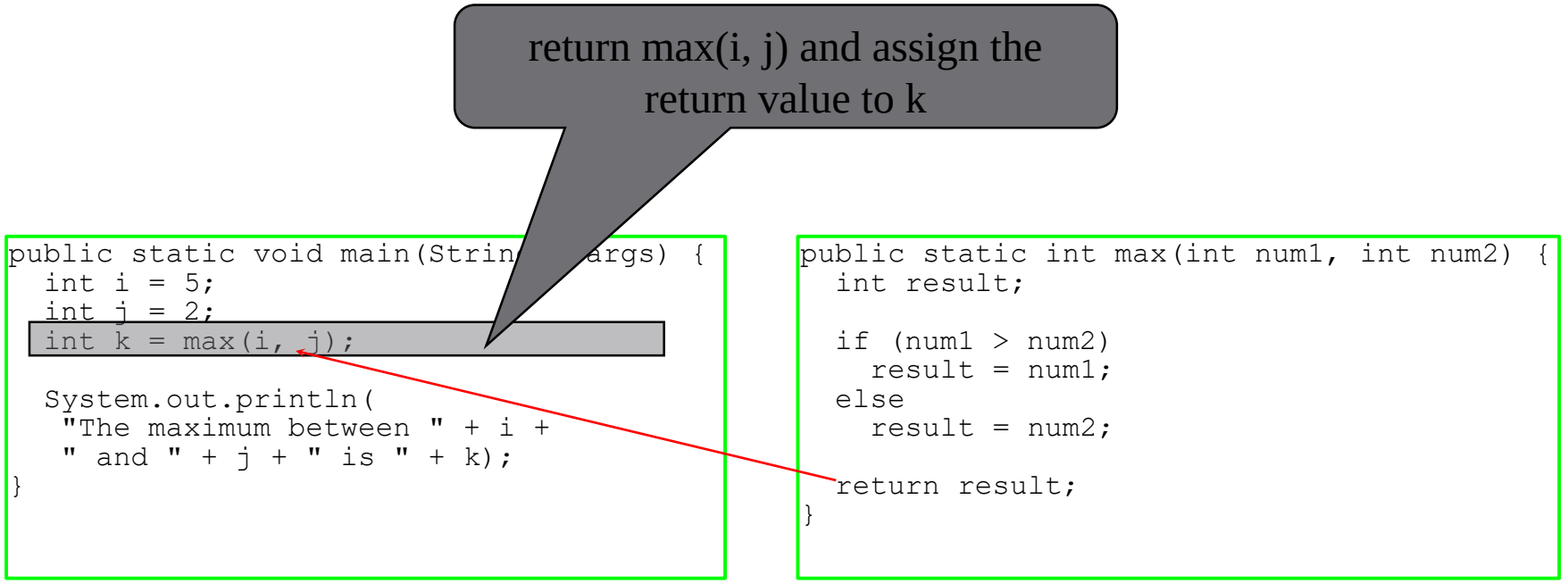
```
public static void main(String[] args) {
  int i = 5;
  int j = 2;
  int k = max(i, j);

  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;
}
```

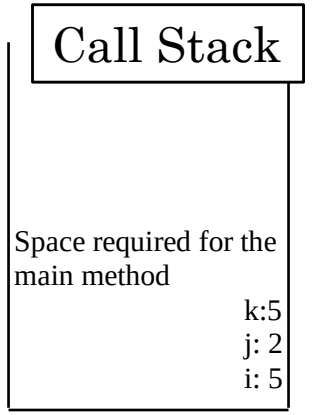
Trace Method Invocation



Trace Method Invocation



Trace Method Invocation



The main method is invoked.

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;
}
```

Types of Methods

- Instance Methods
  - Object level methods
  - Invocation: ObjRef.methodName()
  - Affect object state
- Static Methods
  - static keyword
  - Class level methods, i.e., No access to state, can't access instance variables/ methods.
  - Can access static variables.
  - Invocation: className.methodName()

What is wrong with this program?

```
public static int sign(int n) {
    if (n > 0)
        return 1;
    else if (n == 0)
        return 0;
    else if (n < 0)
        return -1;
}
```

```
public static int sign(int n) {
  if (n > 0)
    return 1;
  else if (n == 0)
    return 0;
  else if (n < 0)
    return -1;
}
```

(a)

Should be

```
public static int sign(int n) {
  if (n > 0)
    return 1;
  else if (n == 0)
    return 0;
  else
    return -1;
}
```

(b)

Method Overloading

- If a class has multiple methods having same name but different in parameters, it is known as **Method Overloading**.
- You can overload a method in Java by two ways:
  - By changing the number of arguments.
  - By changing the data type.

Can you overload a method by changing the return type?

- In java, method overloading is not possible by changing the return type of the method only because of **ambiguity**.

```
class Adder{
  static int add(int a,int b){return a+b;}
  static double add(int a,int b){return a+b;}
}

class TestOverloading3{
  public static void main(String[] args){
  System.out.println(Adder.add(11,11));//ambiguity
}}
```

Can we overload java main() method?

- Yes, by method overloading. You can have any number of main methods in a class by method overloading.
- But JVM calls main() method which receives string array as arguments only.

```
class TestOverloading4{
public static void main(String[] args){
    System.out.println("main with String[]");}
public static void main(String args){
    System.out.println("main with String");}
public static void main(){
    System.out.println("main without args");}
}
```

String[] args

- String[] arguments is the array for runtime argument to our java program, if required we can pass arguments to our java program through the command line.

```
public class ArgumentTest{
        public static void main(String[] args){
          for(String str : args){
              System.out.println(str);
          }
      }
}
```

```
public class Addition{
	public static void main(String[] args){
		if(args.length == 2){
			int a = Integer.parseInt(args[0]);
			int b = Integer.parseInt(args[1]);

			System.out.println("The addition of two numbers: "
+ (a+b));
		}
		else
			System.out.println("Please enter two integer
values!!!");
	}
}
```