

COP-2210

Computer Programming I

Instructor: Dr. Antonio Hernandez

Text: Big Java: Early Objects, Interactive Edition, 6th Edition

Defining Your Own Classes

26. Methods: getting started

Code in Class Constructor: *Try it yourself*

```
public class Prog26_01
{
    public static void main(String[] args)
    {
        Prog26_01 myAppl = new Prog26_01();
    }

    public Prog26_01()
    {
        System.out.println("Body of \"main\" was moved here!");
    }
}
```

Loop: for-each

FOR-EACH loop (enhanced *for* loop):

```
for (<type> <var>: <array or collection>
{
}
```

- **Variable in the header of the loop is the same type as array type**
- **In the body of the loop we use the variable in the header; not an array indexed item**
- **Direct access to the location of an item is not possible**

For-each: *Try it yourself*


```
import java.util.ArrayList;

public class Prog26_02
{
    public static void main(String[] args)
    {
        Prog26_02 myAppl = new Prog26_02();

        public Prog26_02()
        {
            String[] fruits = {"apple", "banana", "mango",
                               "orange", "peach"};

            for (String s : fruits)
            {
                System.out.print(s + " ");
            }

            System.out.println();
        }
    }
}
```



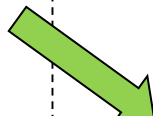
```
System.out.println();

ArrayList<Integer> numbers = new ArrayList<>();

numbers.add(10);
numbers.add(21);
numbers.add(32);

for (int n : numbers)
{
    System.out.print(n + " ");
}

System.out.println();
}
```



PRACTICE

Program 26_03:

Write a Java program that initializes an array with 10 numbers (initialization must occur in the array declaration) and determines the maximum.

Note: Use the structure just given and a for-each loop.



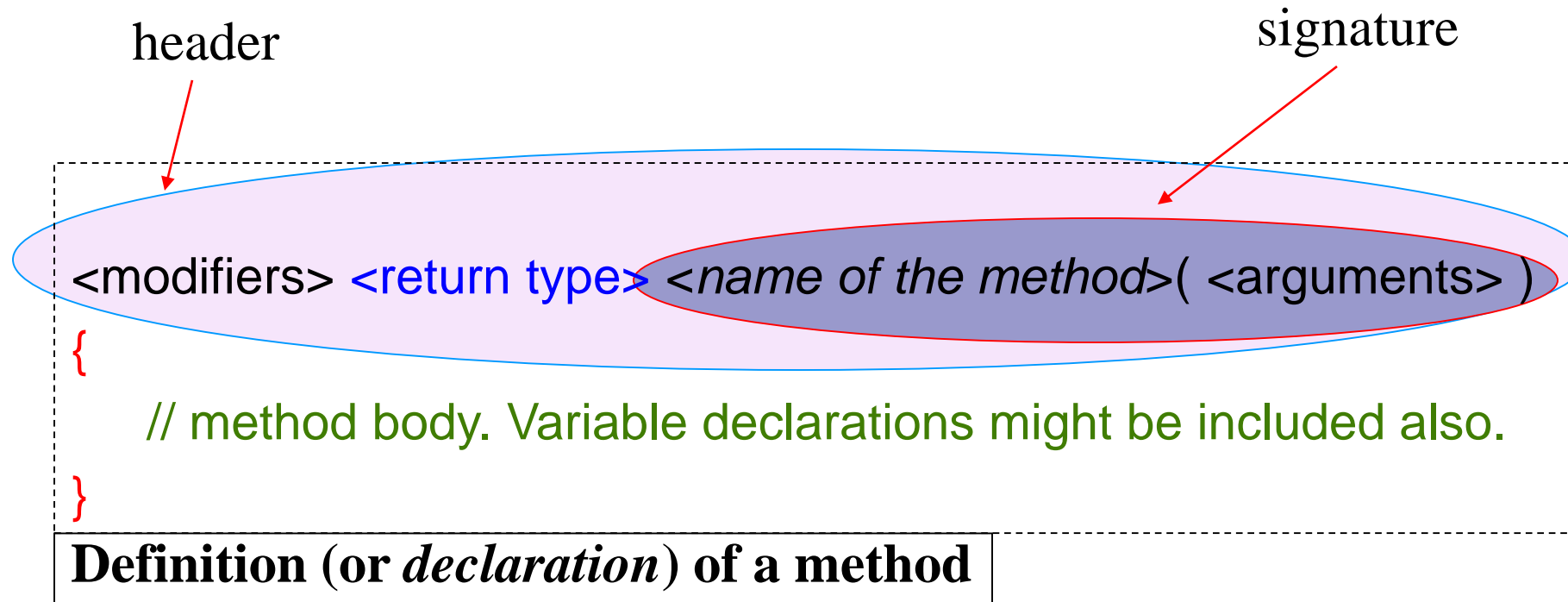
Methods

Method: A basic programming unit in Java. Its basic elements are:

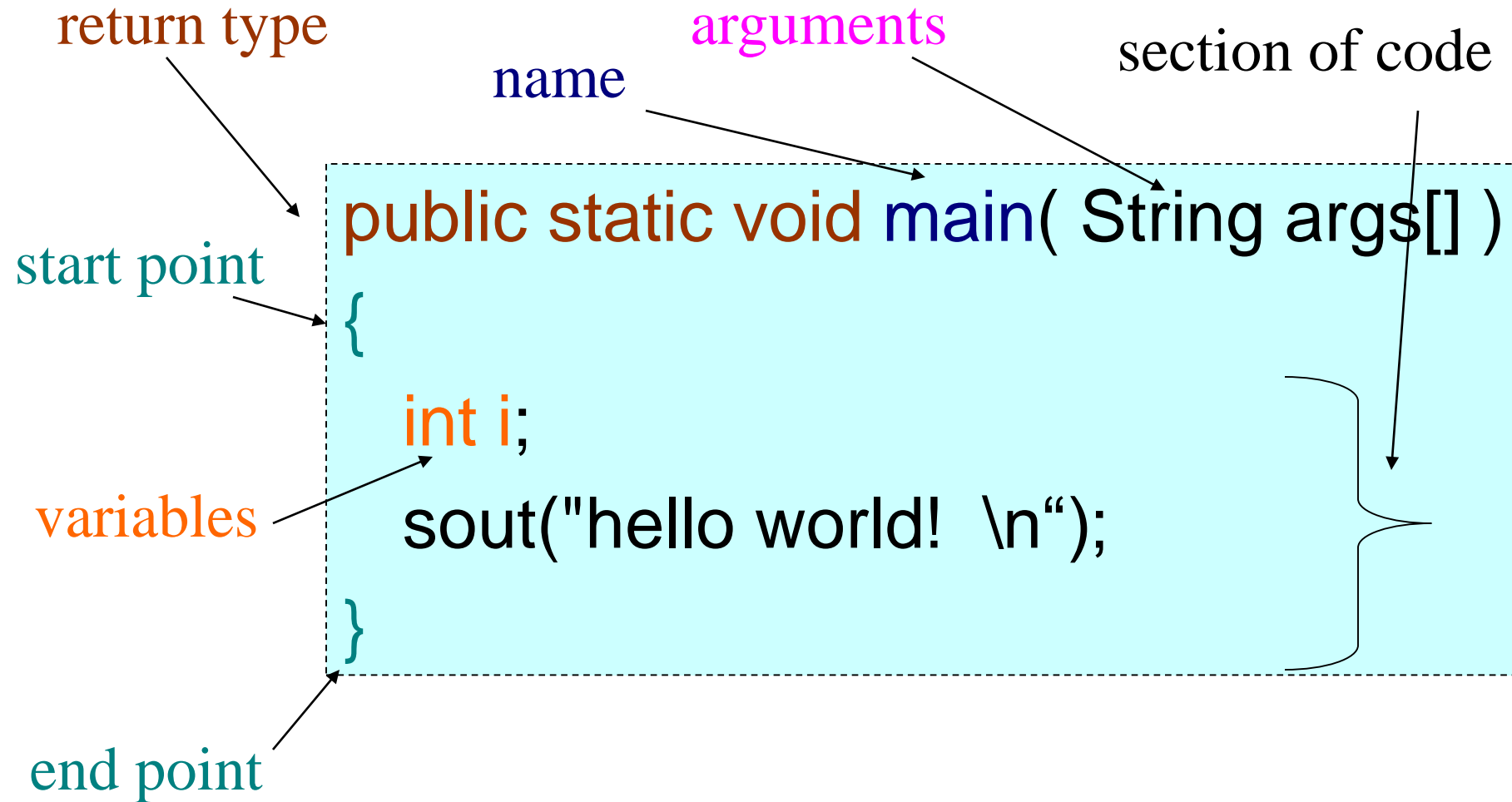
- a complete section of code
- a start point and an end point
- own set of variables
- modifier(s)
- a *return* statement
- list of arguments
- a return type
- a name

Method: parts

Method: a sequence of instructions that performs a task.



Elements of a method



Methods: *Try it yourself*

```
public class Prog26_04
{
    public static void main(String[] args)
    {
        new Prog26_04();
    }

    public Prog26_04()
    {
        writeHello();
    }

    public void writeHello()
    {
        System.out.println("Hello!");
    }
}
```

calling method



called method



Methods: *return* statement

RETURN:

***return* type:** the data type of the value returned from the method

(***void:*** no value will be returned)

returning values:

The value is returned to the calling method in the *return* statement.

Methods: returned values

Return type:

Primitive type,
class name,
void

Example

Returns an *int*

```
if ( rnd.nextInt(11) > 5)
```

```
{
```

```
    JOptionPane.showMessageDialog ( null, ... );
```

```
}
```

Returns nothing
(*void*)

Returning Values from Methods: *Try it yourself*

```
import java.util.Scanner;

public class Prog26_05
{
    public static void main(String[] args)
    {
        new Prog26_05();
    }

    public Prog26_05()
    {
        writeHello();
        String name = getName();
    }

    public void writeHello()
    {
        System.out.println("Hello!");
    }
}
```

```
public String getName()
{
    Scanner in = new Scanner(System.in);

    System.out.print("What is your name? ");
    String s = in.nextLine();

    return s;
}
```

Passing Values to the Called Function

To pass values to the called function:

```
<type> <f. name>( <type1> <arg1>, <type2><arg2> ...)  
{  
  ...  
}
```

In the called method:

```
void myfunction( int a, double b )  
{  
  int c;  
  c = a + b;  
}
```

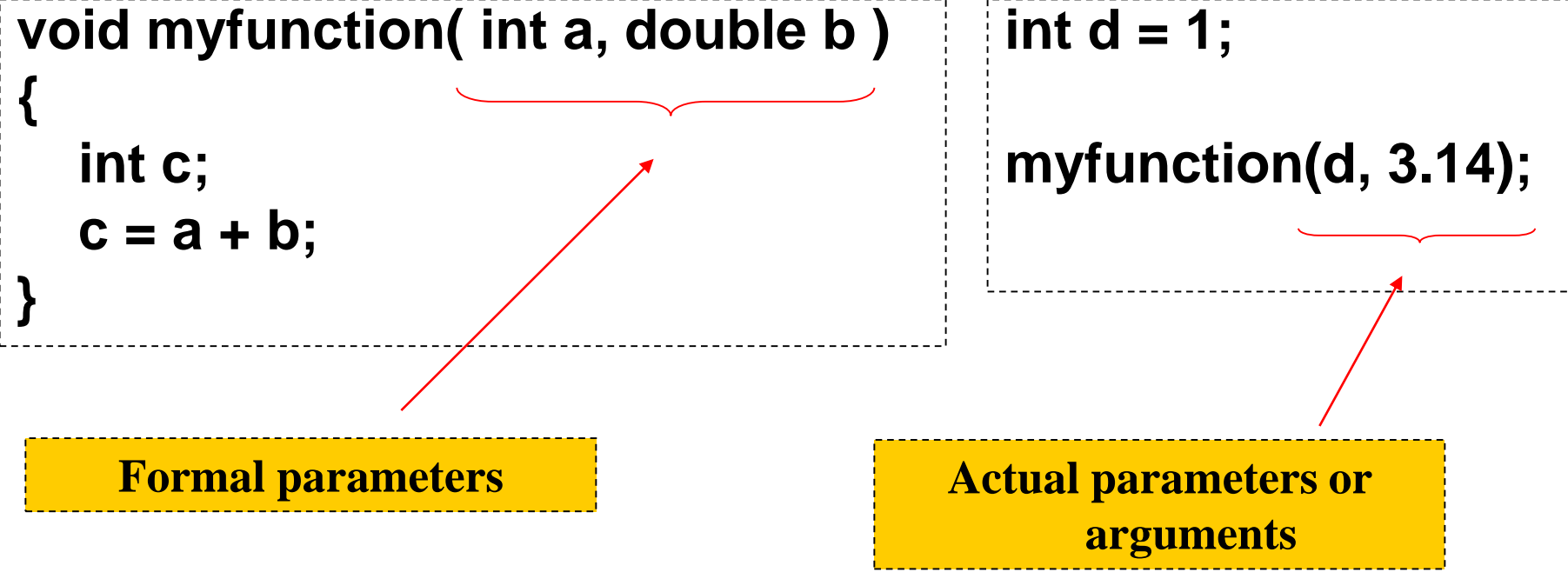
In the calling method:

```
int d = 1;  
  
myfunction(d, 3.14);
```

Passing Values: Terminology

In the called method:

```
void myfunction( int a, double b )  
{  
    int c;  
    c = a + b;  
}
```



Formal parameters

In the calling method:

```
int d = 1;  
myfunction(d, 3.14);
```

**Actual parameters or
arguments**

Passing Values to a Method: *Try it yourself*

```
import java.util.Scanner;

public class Prog26_06
{
    public static void main(String[] args)
    {
        new Prog26_06();
    }

    public Prog26_06()
    {
        writeHello();
        String name = getName();
        welcomeMessage(name);
    }

    public void writeHello()
    {
        System.out.println("Hello!");
    }
}
```

```
public String getName()
{
    Scanner in = new Scanner(System.in);

    System.out.print("What is your name? ");
    String s = in.nextLine();

    return s;
}

public void welcomeMessage(String s)
{
    System.out.println("Welcome aboard, " + s + "!");
}
}
```


Passing and Returning: *Try it yourself*

```
import java.util.Random;
import java.util.Scanner;

public class Prog26_07
{
    public static void main(String[] args)
    {
        new Prog26_07();
    }

    public Prog26_07()
    {
        Scanner in = new Scanner(System.in);
        System.out.print("Range for random number: ");
        int lowerBound = in.nextInt();
        int upperBound = in.nextInt();

        System.out.println("Random number = " +
            randomNumber(lowerBound, upperBound));
    }
}
```

```
public int randomNumber(int a, int b)
{
    Random rnd = new Random();

    return a + rnd.nextInt(b-a+1);
}
```