JAVA: Lesson 3

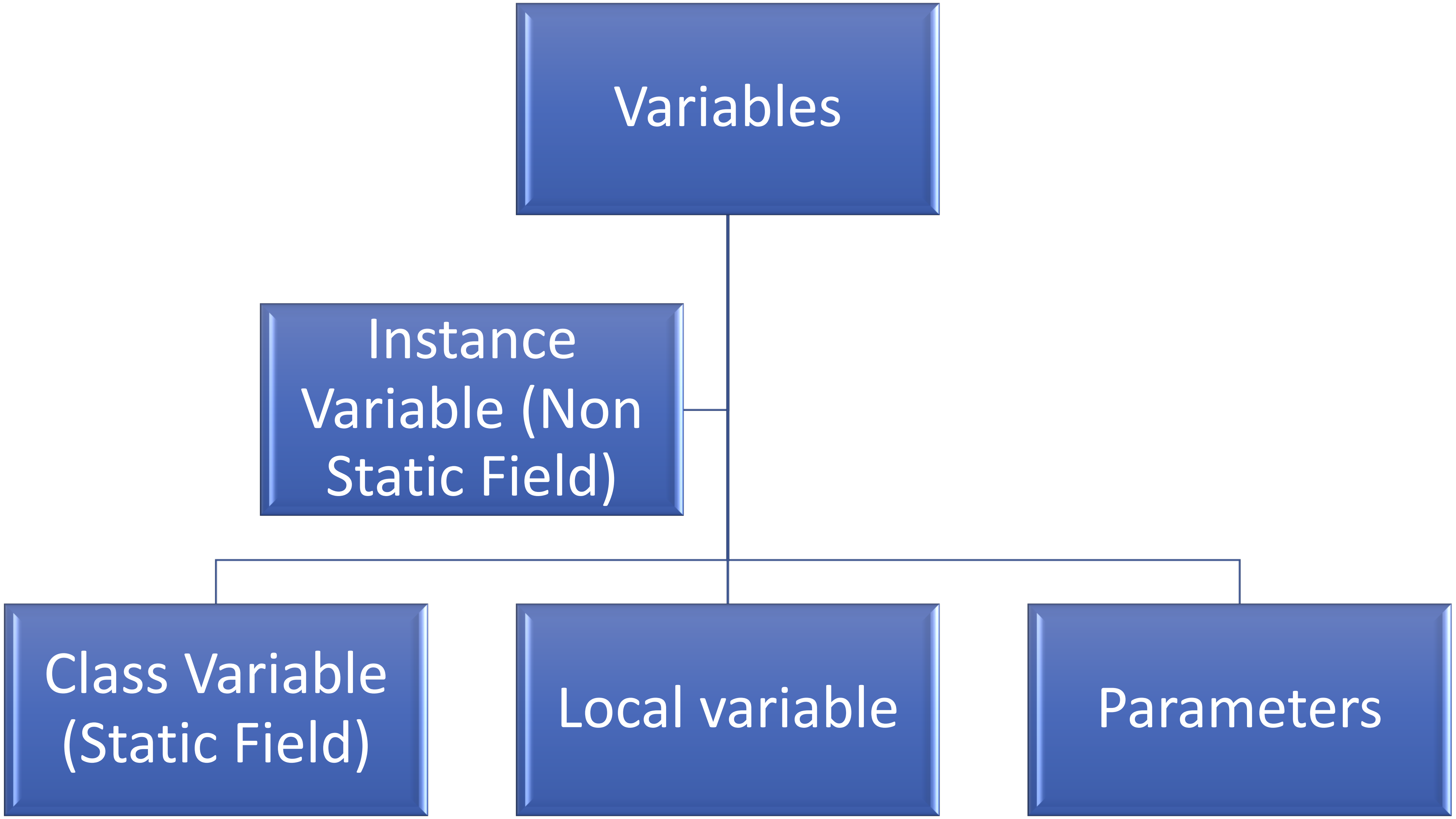
Assignment

- ☐ ident-fier
- ☐ identifier++
- ☐ $Identifier
- ☐ super
- ☐ class
- ☐ Identifier
- ☐ public
- ☐ reserved

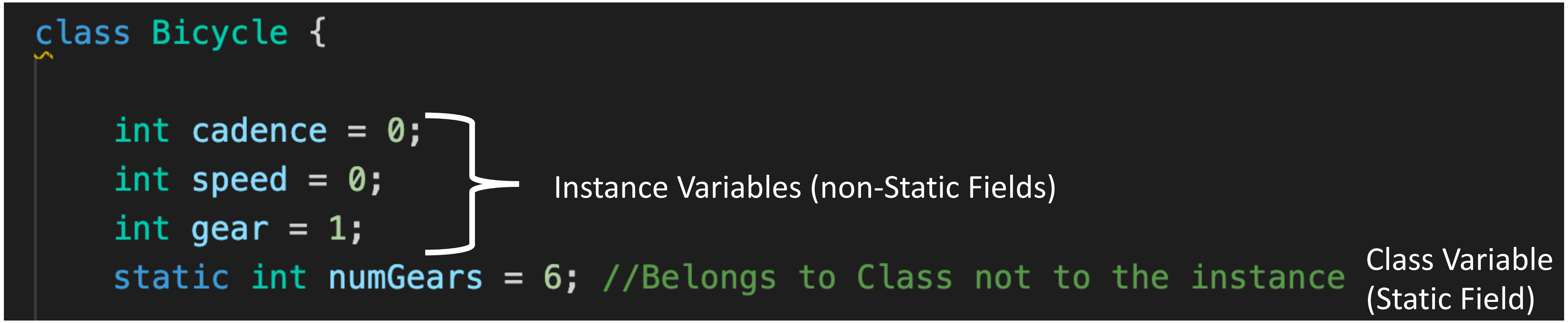
```
class super {
 public static void main(string[] arguments) {
 }
}
```

- ☐ super should be public.
- ☐ super is a keyword and may not be used as a class name.
- ☐ The argument to the main() method should be of type String[].
- ☐ The argument to the main() method should be named args.

Variables


Variables
Instance Variable (Non Static Field)
Class Variable (Static Field)
Local variable
Parameters

Instance Variable vs Class Variable

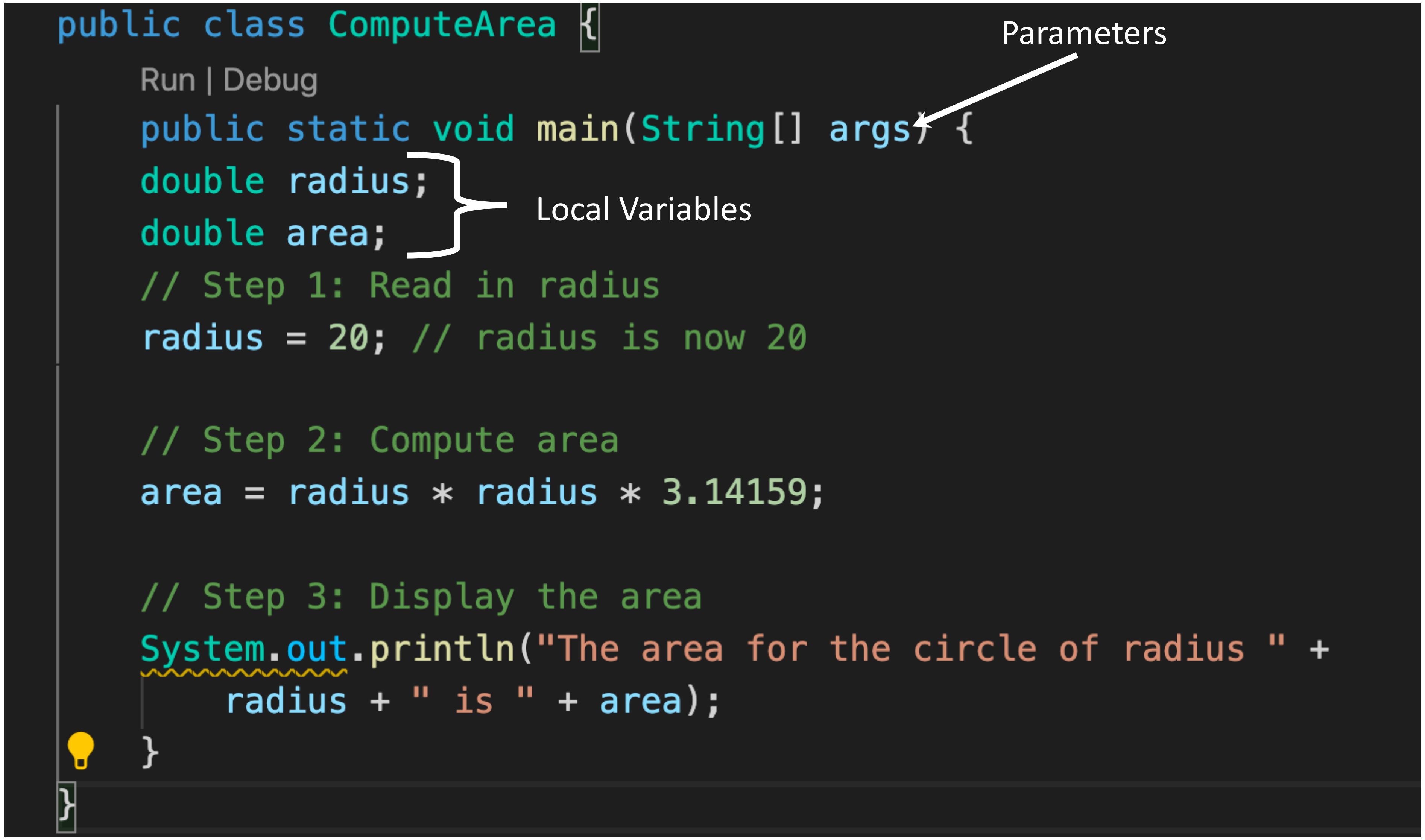


```java
class Bicycle {

    int cadence = 0;
    int speed = 0;
    int gear = 1;
    static int numGears = 6; //Belongs to Class not to the instance
```

Instance Variables (non-Static Fields)
Class Variable (Static Field)

Parameters vs Arguments



```
public class ComputeArea {
    public static void main(String[] args) {
    double radius;
    double area;
    // Step 1: Read in radius
    radius = 20; // radius is now 20

    // Step 2: Compute area
    area = radius * radius * 3.14159;

    // Step 3: Display the area
    System.out.println("The area for the circle of radius " +
        radius + " is " + area);
    }
}
```

Variable name *args* is not compulsory

Mathematical Operators

Name	*Meaning*	*Example*	*Result*
+	Addition	34 + 1	35
-	Subtraction	34.0 – 0.1	33.9
*	Multiplication	300*30	9000
/	Division	1.0 / 2.0	0.5
%	Remainder	20 % 3	2

Practice Questions

- Is the given number odd or even?
- What will be the day in 100 days?

More Practice

Show the result of the following code:

```
System.out.println(2 * (5 / 2 + 5 / 2));
System.out.println(2 * 5 / 2 + 2 * 5 / 2);
System.out.println(2 * (5 / 2));
System.out.println(2 * 5 / 2);
```

Are the following statements correct? If so, show the output.

```
System.out.println("25 / 4 is " + 25 / 4);
System.out.println("25 / 4.0 is " + 25 / 4.0);
System.out.println("3 * 2 / 4 is " + 3 * 2 / 4);
System.out.println("3.0 * 2 / 4 is " + 3.0 * 2 / 4);
```

Practice Session

- Write a program to accept following inputs:
    - **Double** principal,
    - **Integer** annual rate of interest,
    - **Byte** number of years
- You should calculate the interest and amount according to following formula

$$I = \frac{P.r.t}{100}$$

$$A = P + I$$

- Print the Interest and Amount to the console.

Operator Precedence

Operators	Precedence
postfix	*expr++, expr--*
unary	*++expr, --expr, +expr, -expr, ~, !*
multiplicative	**, /, %*
additive	*+, -*
shift	*<<, >>, >>>*
relational	*< , >, <=, >=, instanceof*
equality	*==, !=*
bitwise AND	*&*
bitwise exclusive OR	*^*
bitwise inclusive OR	*|*
logical AND	*&&*
logical OR	*||*
ternary	*? :*
assignment	*=, +=, -=, *=, /=, %=, &=, ^=, |=, <<=, >>=, >>>=*

Demonstration 1

- PostfixAndUnaryOperator
- ComparisonDemo

Meaning of Operators

Operator	*Name*	*Example*	*Equivalent*
`+=`	Addition assignment	`i += 8`	`i = i + 8`
`-=`	Subtraction assignment	`i -= 8`	`i = i - 8`
`*=`	Multiplication assignment	`i *= 8`	`i = i * 8`
`/=`	Division assignment	`i /= 8`	`i = i / 8`
`%=`	Remainder assignment	`i %= 8`	`i = i % 8`

Operator	*Name*	*Description*	*Example* (assume i = 1)
`++var`	preincrement	Increment `var` by `1`, and use the new `var` value in the statement	`int j = ++i;` // j is 2, i is 2
`var++`	postincrement	Increment `var` by `1`, but use the original `var` value in the statement	`int j = i++;` // j is 1, i is 2
`--var`	predecrement	Decrement `var` by `1`, and use the new `var` value in the statement	`int j = --i;` // j is 0, i is 0
`var--`	postdecrement	Decrement `var` by `1`, and use the original `var` value in the statement	`int j = i--;` // j is 1, i is 0

Mathematical Operator Precedence

$$\frac{3 + 4x}{5} - \frac{10(y - 5)(a + b + c)}{x} + 9\left(\frac{4}{x} + \frac{9 + x}{y}\right)$$

can be translated into a Java expression as follows:

```
(3 + 4 * x) / 5 – 10 * (y - 5) * (a + b + c) / x +
 9 * (4 / x + (9 + x) / y)
```

Common Errors: Demonstrations 2

- Common Error 1: Undeclared/Uninitialized Variables and Unused Variables
- Common Error 2: Value Overflow
- Common Error 3: Round-off Errors
- Common Error 4: Unintended Integer Division
- Common Pitfall 1: Redundant Input Objects
- Warning: In real Life App never use double/float to store currency! Use BigInt.

```
Scanner input = new Scanner(System.in);
System.out.print("Enter an integer: ");
int v1 = input.nextInt();

Scanner input1 = new Scanner(System.in);
System.out.print("Enter a double value: ");
double v2 = input1.nextDouble();
```

BAD CODE

```
Scanner input = new Scanner(System.in);
System.out.print("Enter an integer: ");
int v1 = input.nextInt();
System.out.print("Enter a double value: ");
double v2 = input.nextDouble();
```

GOOD CODE

Software Development Process

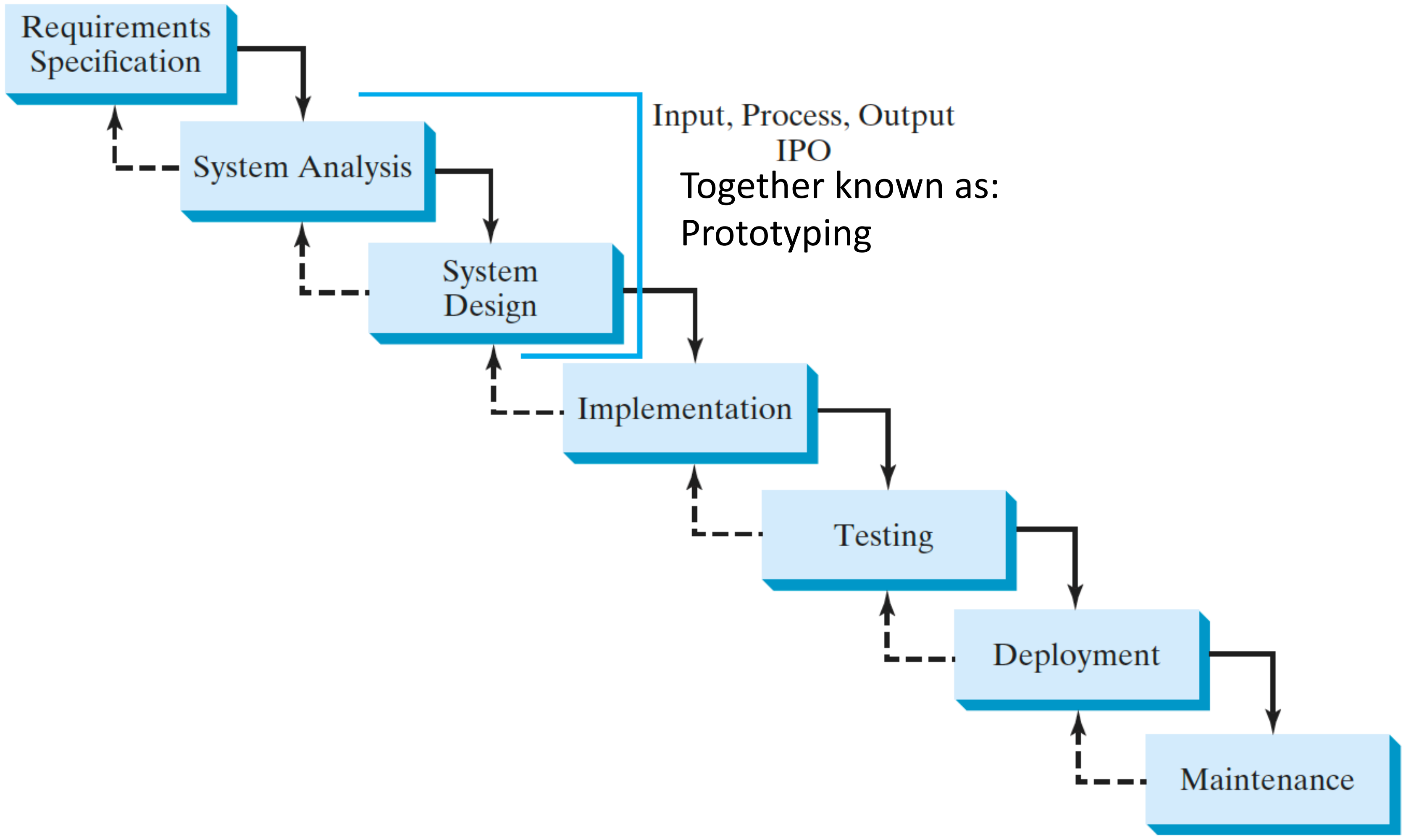


FIGURE 2.3 At any stage of the software development life cycle, it may be necessary to go back to a previous stage to correct errors or deal with other issues that might prevent the software from functioning as expected.

Demonstration: Displaying Current Time

- ShowCurrentTime

Time Format

Converter	Flag	Explanation
d		A decimal integer.
f		A float.
n		A new line character appropriate to the platform running the application. You should always use %n, rather than \n.
tB		A date & time conversion—locale-specific full name of month.
td, te		A date & time conversion—2-digit day of month. td has leading zeroes as needed, te does not.
ty, tY		A date & time conversion—ty = 2-digit year, tY = 4-digit year.
tl		A date & time conversion—hour in 12-hour clock.
tM		A date & time conversion—minutes in 2 digits, with leading zeroes as necessary.
tp		A date & time conversion—locale-specific am/pm (lower case).
tm		A date & time conversion—months in 2 digits, with leading zeroes as necessary.
tD		A date & time conversion—date as %tm%td%ty
	08	Eight characters in width, with leading zeroes as necessary.
	+	Includes sign, whether positive or negative.
	,	Includes locale-specific grouping characters.
	-	Left-justified..
	.3	Three places after decimal point.
	10.3	Ten characters in width, right justified, with three places after decimal point