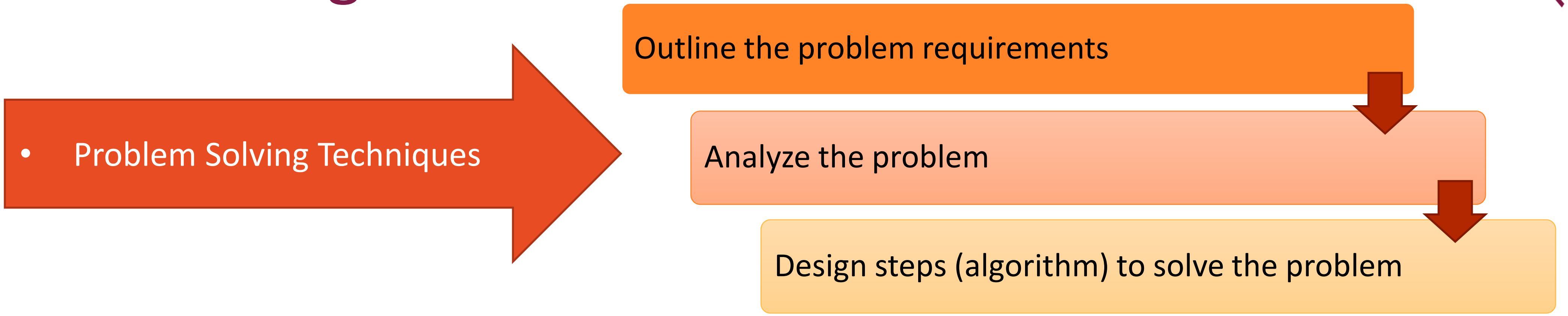
Variables and Data Types
User Inputs

Computer Engineering

Yusramohammed@tiu.edu.iq

2022 - 2023

Problem Solving Process



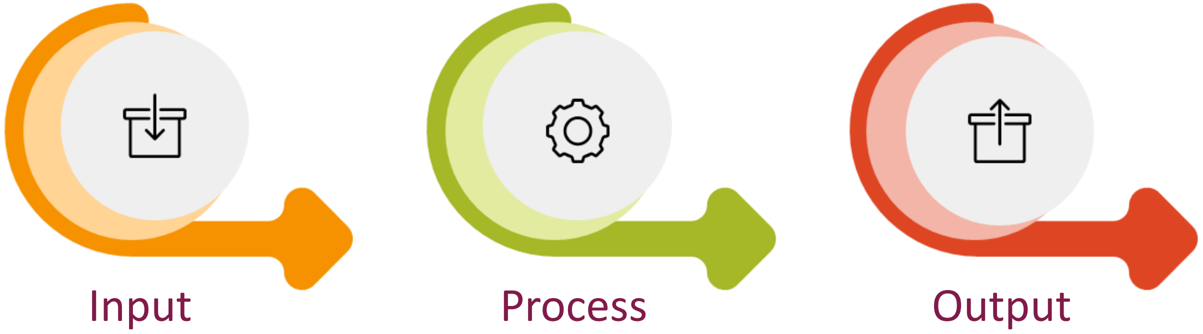
- Problem Solving Techniques

Outline the problem requirements

Analyze the problem

Design steps (algorithm) to solve the problem

- Three steps that a program typically performs:



Input

Process

Output

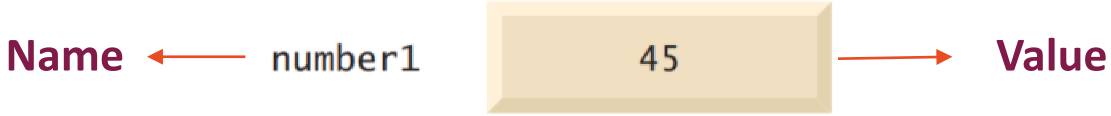
- Gather input
  - from keyboard
  - from files on disk drives

Process the input

- Display the result as output
  - send it to the screen
  - write to a file

Memory Concept & Variables

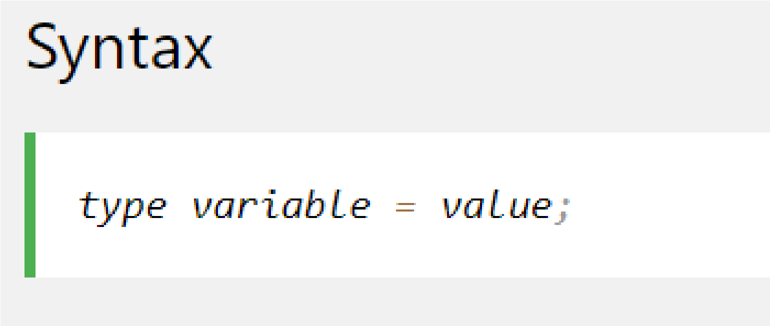
- Memory is a location in a computer to store data (Example: Numbers, Text, Characters, etc.).
- Programs remember numbers and other data in the computer’s memory and access that data through program elements called ***variables.***
- A ***variables*** are actually corresponding to a location in the computer’s memory where a value can be stored for use later in a program.
- Every variable has a **name**, a **type**, a **size** (in bytes) and a **value**.



Name ← number1 45 → Value

- A variable’s name enables the program to access the value of the variable in memory.

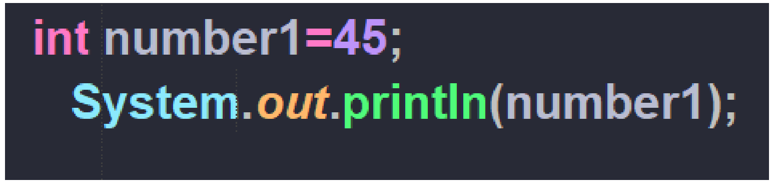
Memory Concept & Variables



Syntax

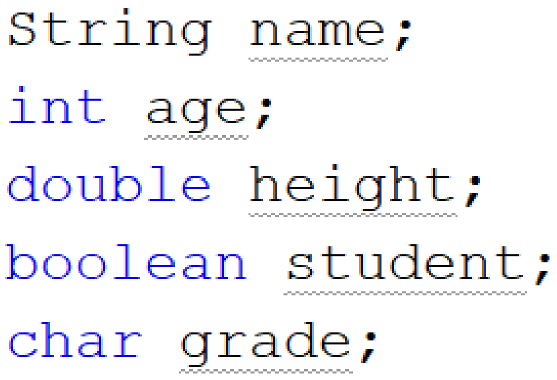
```
type variable = value;
```

Example



```
int number1=45;
  System.out.println(number1);
```

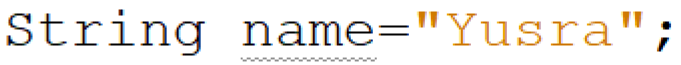
- A variable’s name can be any valid identifier.
- A variable’s type specifies what kind of information is stored at that location in memory.
- Like other statements, declaration statements end with a semicolon (;)
- Declaring variables to store values:



```
String name;
int age;
double height;
boolean student;
char grade;
```

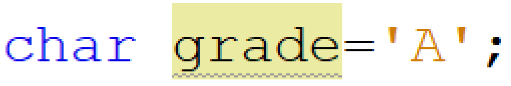
Variables in Java

- In Java, there are different types of variables, for example:
- **String** → stores text, such as "Hello". String values are surrounded by double quotes “ “.



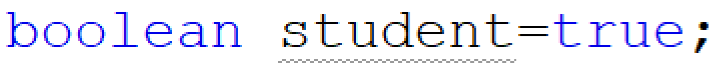
```
String name="Yusra";
```

- **char** → stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes ‘ ‘



```
char grade='A';
```

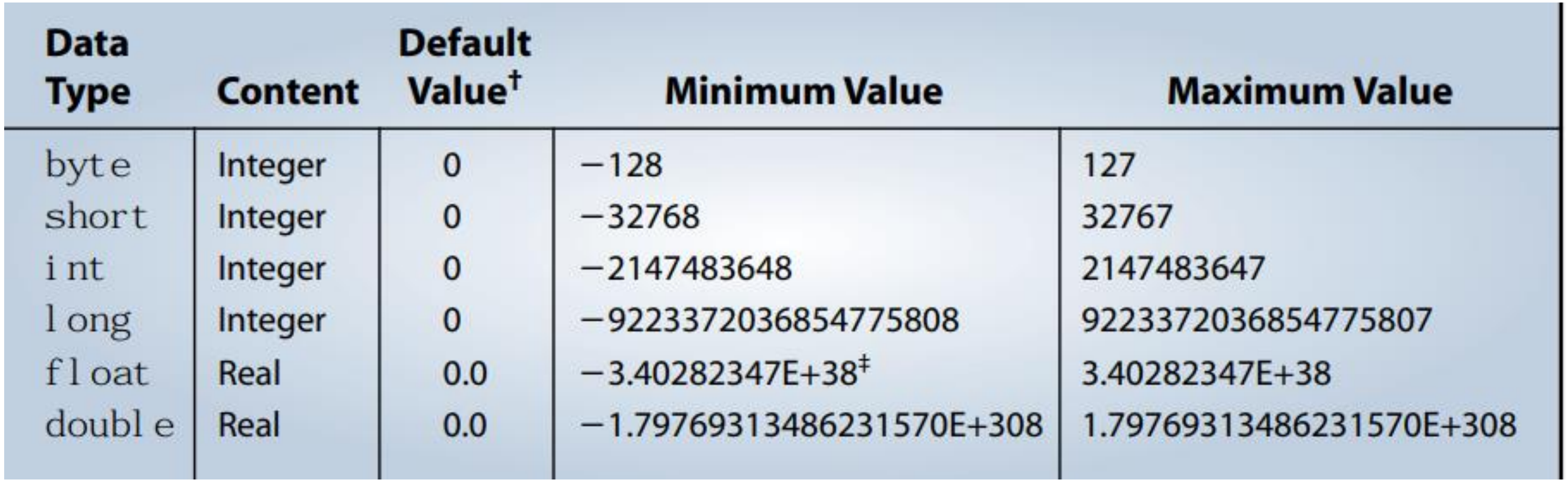
- **boolean** → stores values with two st



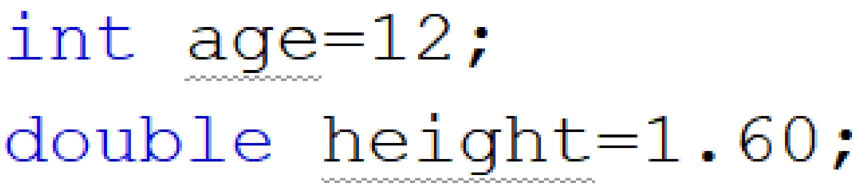
```
boolean student=true;
```

Variables in Java

Numerical variables → There are six numerical data types in Java: byte, short, int, long, float, and double

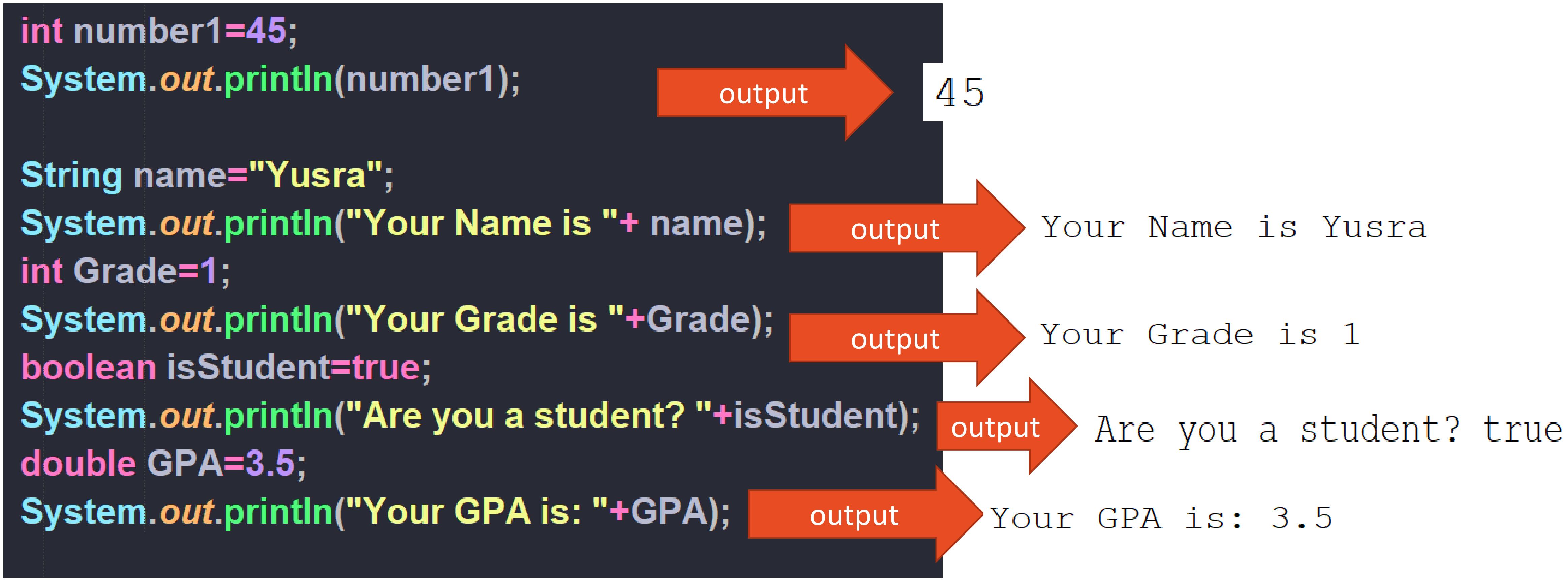


Data Type	Content	Default Value†	Minimum Value	Maximum Value
byte	Integer	0	−128	127
short	Integer	0	−32768	32767
int	Integer	0	−2147483648	2147483647
long	Integer	0	−9223372036854775808	9223372036854775807
float	Real	0.0	−3.40282347E+38‡	3.40282347E+38
double	Real	0.0	−1.79769313486231570E+308	1.79769313486231570E+308



```
int age=12;
double height=1.60;
```

Printing Values of Variables



```java
int number1=45;
System.out.println(number1);

String name="Yusra";
System.out.println("Your Name is "+ name);
int Grade=1;
System.out.println("Your Grade is "+Grade);
boolean isStudent=true;
System.out.println("Are you a student? "+isStudent);
double GPA=3.5;
System.out.println("Your GPA is: "+GPA);
```

```
45
```

```
Your Name is Yusra
```

```
Your Grade is 1
```

```
Are you a student? true
```

```
Your GPA is: 3.5
```

Working with Variables (Adding to Numbers)

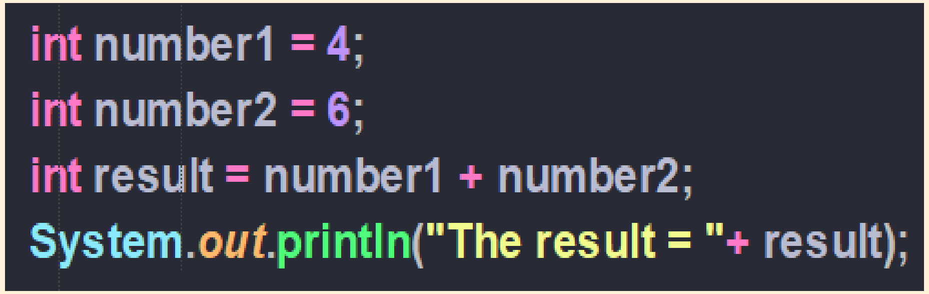
- Example: Write a java program to add two integer numbers and display the result.

Pseudo-code

```
1. Start
2. Variable for number 1
3. Variable for number 2
4. Variable for result
     Result=number1+number2;
5. Display the result
6. End
```

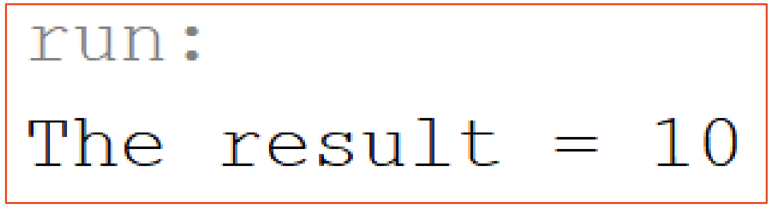
Convert to Java Code

JAVA Code



```java
int number1 = 4;
int number2 = 6;
int result = number1 + number2;
System.out.println("The result = "+ result);
```

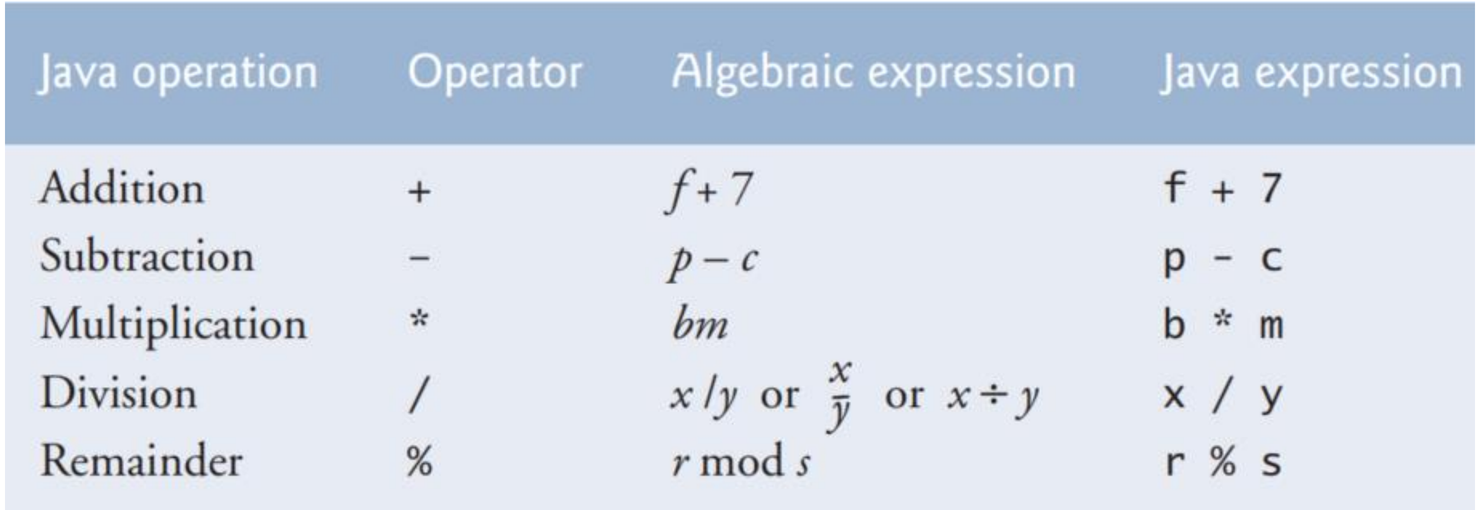
output



```
run:
The result = 10
```

Arithmetic Operators

- Most programs perform arithmetic calculations.
- The **asterisk** (*****) indicates multiplication, and the percent sign (**%**) is the **remainder operator**.
- The arithmetic operators are *binary* operators, because each operates on *two* operands. For example, the expression f + 7 contains the binary operator + and the two operands f and 7.



Java operation	Operator	Algebraic expression	Java expression
Addition	+	$f + 7$	`f + 7`
Subtraction	–	$p - c$	`p - c`
Multiplication	*	bm	`b * m`
Division	/	x / y or $\frac{x}{y}$ or $x \div y$	`x / y`
Remainder	%	$r \bmod s$	`r % s`

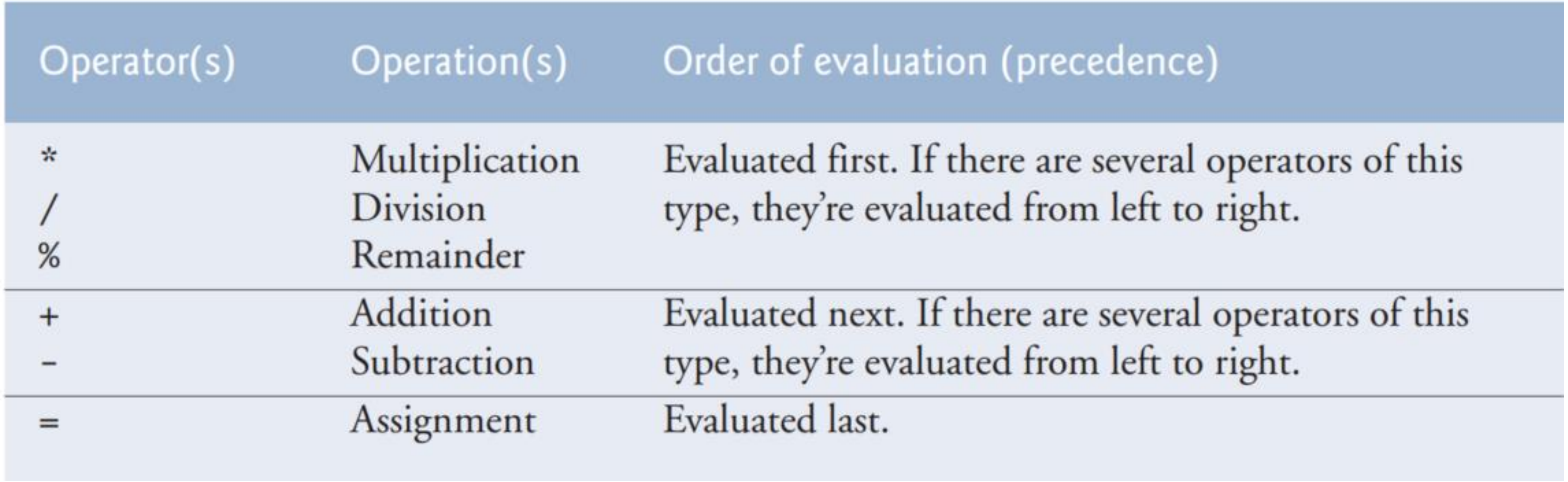
- Java provides the remainder operator, %, which yields the remainder after division. The expression x % y yields the remainder after x is divided by y. Thus, 7 % 4 yields 3, and 17 % 5 yields 2.
- The % remainder operator is most commonly used with integer operands but can also be used with other arithmetic types

Rules of Operator Precedence

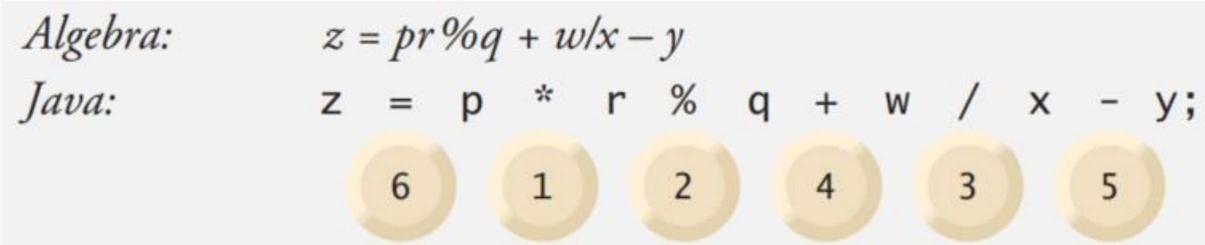
First

Second

Last



Operator(s)	Operation(s)	Order of evaluation (precedence)
* / %	Multiplication Division Remainder	Evaluated first. If there are several operators of this type, they're evaluated from left to right.
+ -	Addition Subtraction	Evaluated next. If there are several operators of this type, they're evaluated from left to right.
=	Assignment	Evaluated last.



Algebra: $z = pr\%q + w/x - y$

Java: `z = p * r % q + w / x - y;`

Example

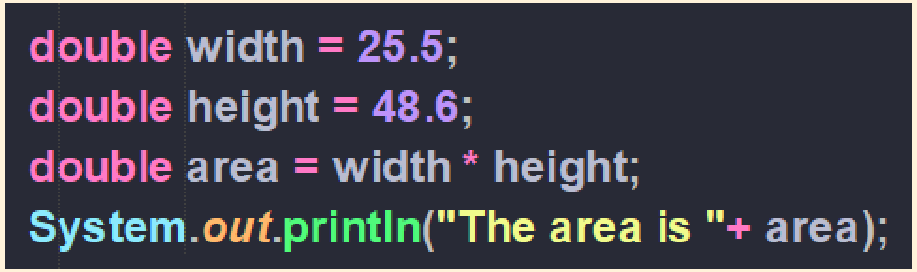
- Find the area for the rectangle of width 25.5 and length 48.6 and display the area.

Pseudo-code

```
1.  Start
2.  Variable for width (double)
3.  Variable for height (double)
4.  Variable for area (double)
      area=width*height;
5.  Display the area
6.  End
```

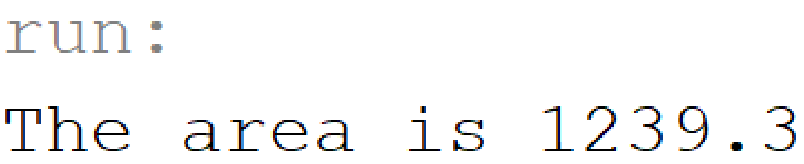
Convert to Java Code

JAVA Code



```java
double width = 25.5;
double height = 48.6;
double area = width * height;
System.out.println("The area is "+ area);
```

output

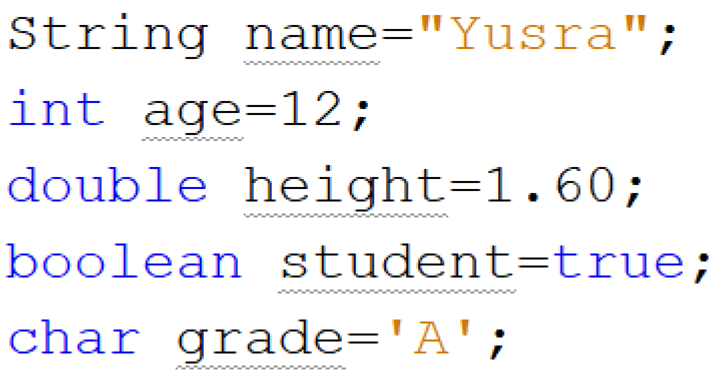


```
run:
The area is 1239.3
```

Assign values to variables (Initializing Variables)

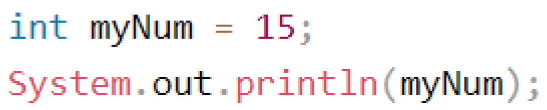
- We can give value to a variable in two ways:

1. **Directly while creating:** We can give values to a variable while creating.



```
String name="Yusra";
int age=12;
double height=1.60;
boolean student=true;
char grade='A';
```

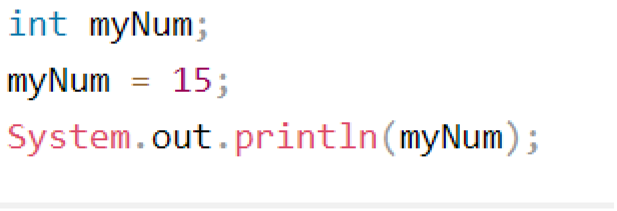
- **Example**



```
int myNum = 15;
System.out.println(myNum);
```

- We can declare a variable without assigning the value, and assign the value later:

- **Example**



```
int myNum;
myNum = 15;
System.out.println(myNum);
```

Assign values to variables (Initializing Variables)

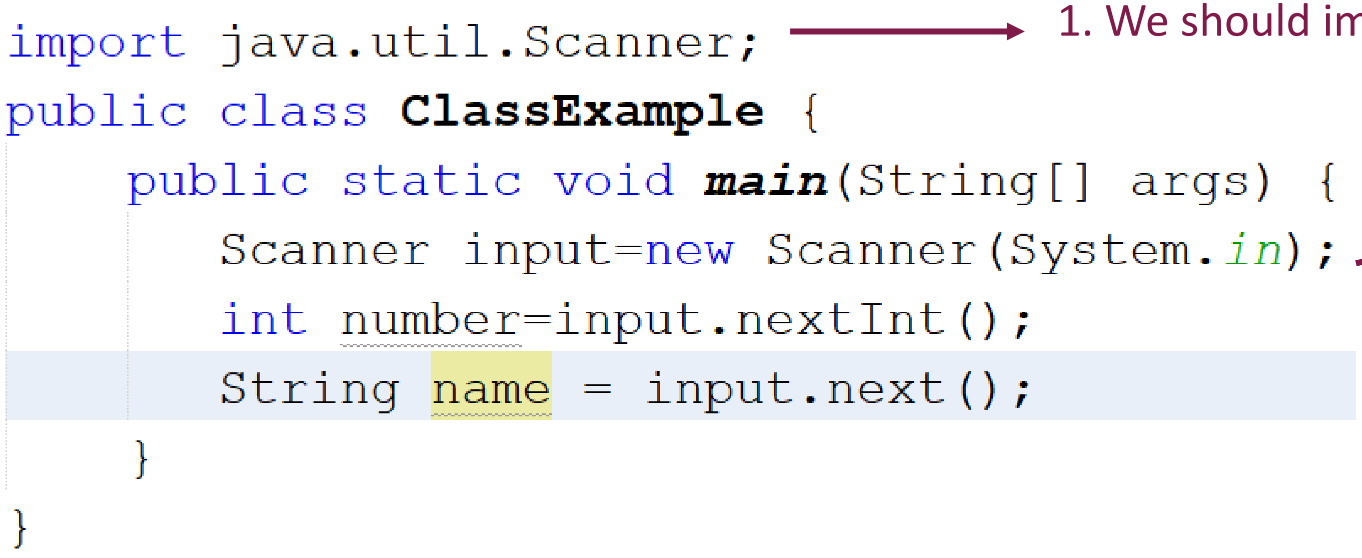
2. Get Values from the users using *Scanner* class:

- The ***Scanner*** class is used to get user input, and it is found in the ***java.util*** package.

Java.util.Scanner;

- To use the Scanner class, create an object of the class and use any of the available methods found in the Scanner class documentation

Scanner input=new Scanner(System.in);



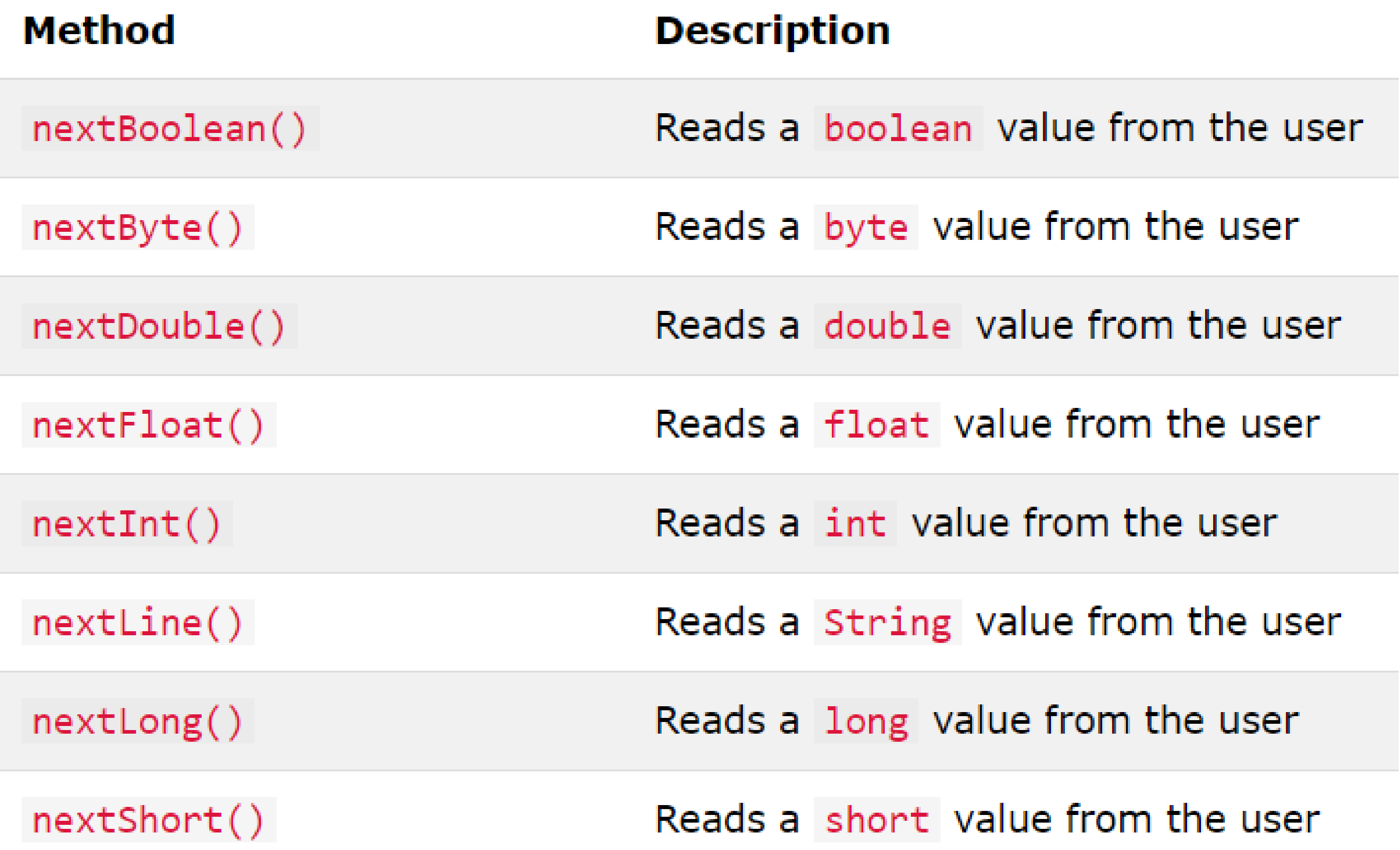
```java
import java.util.Scanner;
public class ClassExample {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        int number=input.nextInt();
        String name = input.next();
    }
}
```

1. We should import the Scanner Class

2. Create Scanner Object

Assign values to variables (Initializing Variables)

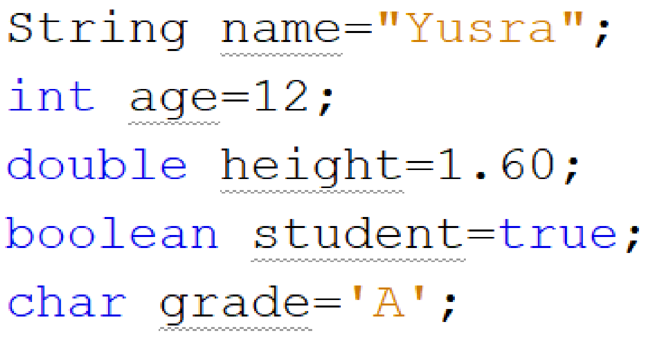
- Methods used for getting values from the users.



Method	Description
`nextBoolean()`	Reads a `boolean` value from the user
`nextByte()`	Reads a `byte` value from the user
`nextDouble()`	Reads a `double` value from the user
`nextFloat()`	Reads a `float` value from the user
`nextInt()`	Reads a `int` value from the user
`nextLine()`	Reads a `String` value from the user
`nextLong()`	Reads a `long` value from the user
`nextShort()`	Reads a `short` value from the user

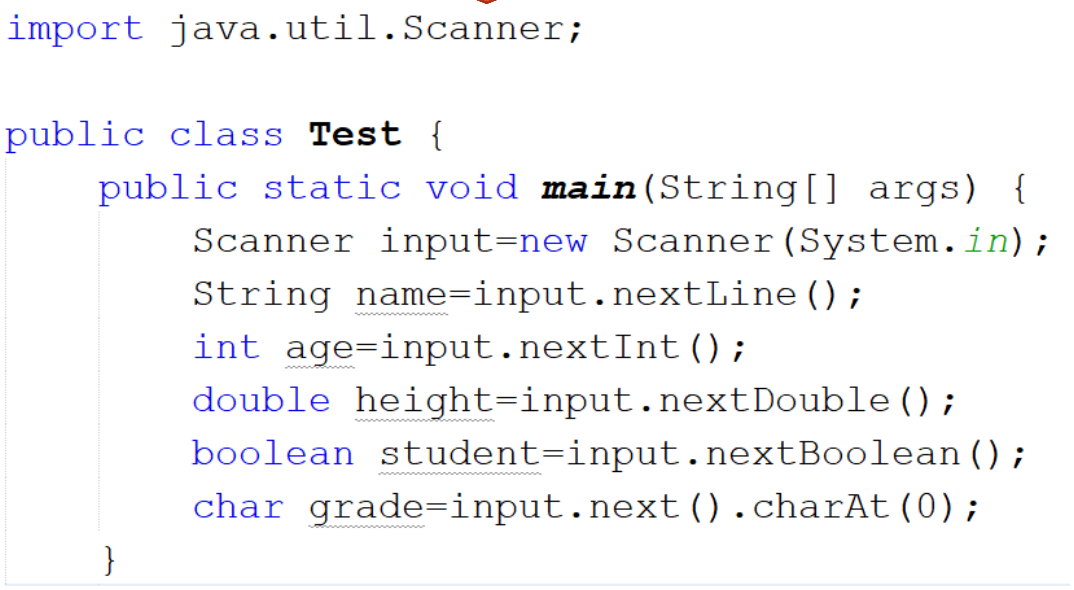
Assign values to variables (Initializing Variables)

Directly while creating



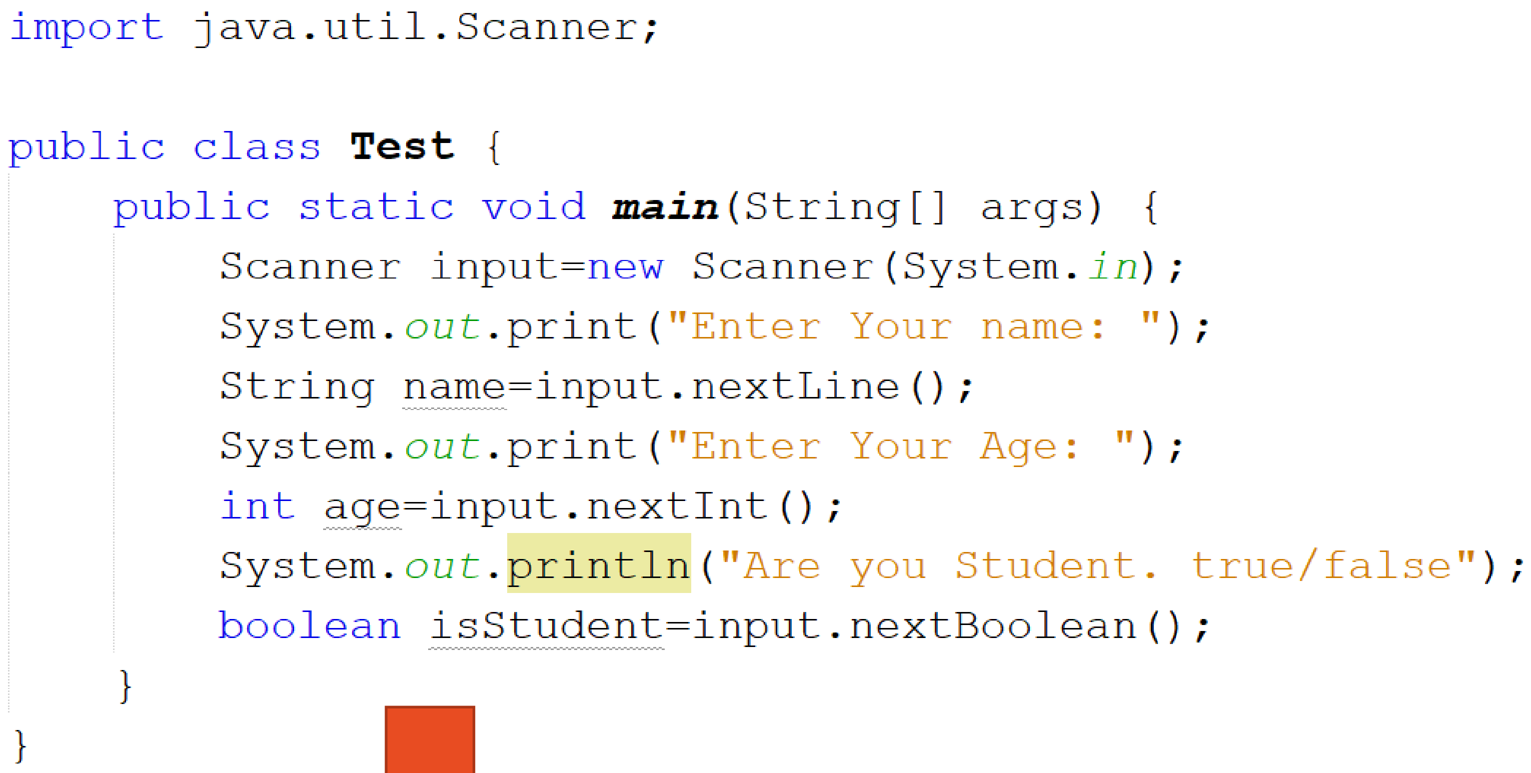
```java
String name="Yusra";
int age=12;
double height=1.60;
boolean student=true;
char grade='A';
```

From User using Scanner class

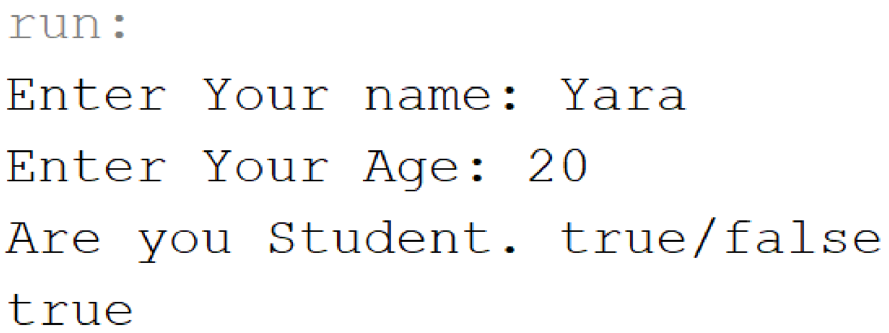


```java
import java.util.Scanner;

public class Test {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        String name=input.nextLine();
        int age=input.nextInt();
        double height=input.nextDouble();
        boolean student=input.nextBoolean();
        char grade=input.next().charAt(0);
    }
```

Assign values to variables (Initializing Variables)



```
import java.util.Scanner;

public class Test {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        System.out.print("Enter Your name: ");
        String name=input.nextLine();
        System.out.print("Enter Your Age: ");
        int age=input.nextInt();
        System.out.println("Are you Student. true/false");
        boolean isStudent=input.nextBoolean();
    }
}
```



```
run:
Enter Your name: Yara
Enter Your Age: 20
Are you Student. true/false
true
```

Example

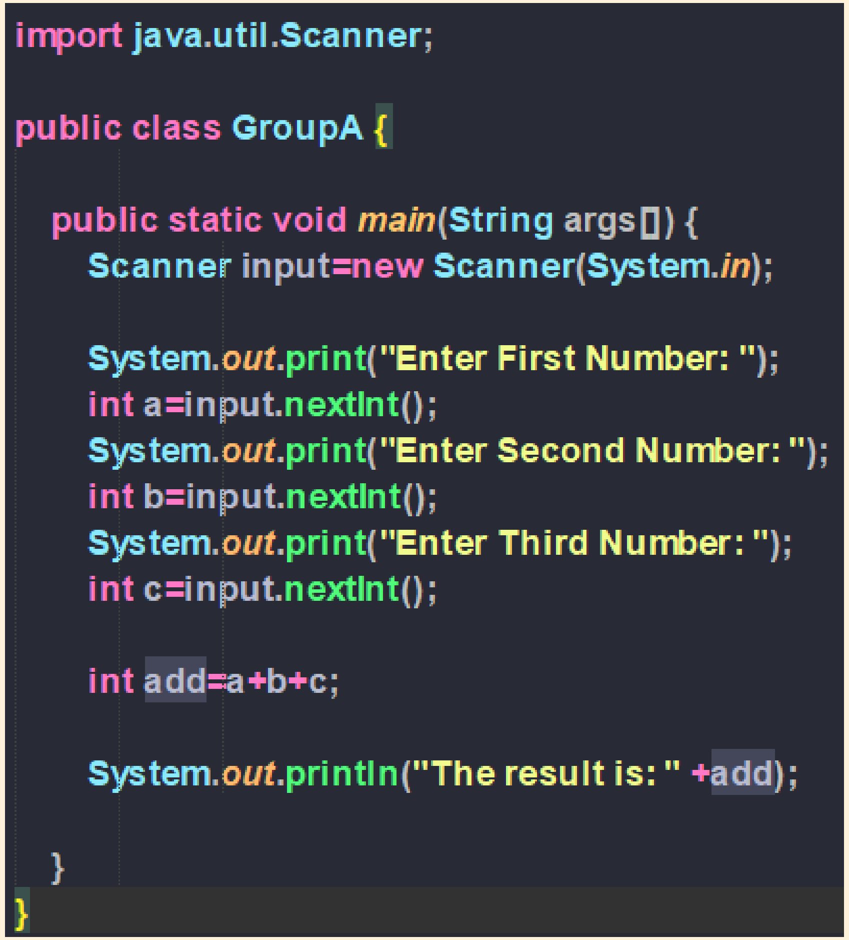
- Ask a user to enter three numbers and print the addition of them.

Pseudo-code

```
1.  Start
2.    Create Scanner Object
3.    Ask the user to enter first number
4.    Create variable to store first number
5.    Ask user to enter second number
6.    Create variable to store second number
7.    Ask user to enter third number
8.    Create variable to store third number
9.    Create variable to store the addition of three
      numbers
10.   Print the addition result to the user
11. End
```

JAVA CODE

JAVA Code



```java
import java.util.Scanner;

public class GroupA {

  public static void main(String args[]) {
    Scanner input=new Scanner(System.in);

    System.out.print("Enter First Number: ");
    int a=input.nextInt();
    System.out.print("Enter Second Number: ");
    int b=input.nextInt();
    System.out.print("Enter Third Number: ");
    int c=input.nextInt();

    int add=a+b+c;

    System.out.println("The result is: " +add);

  }
}
```

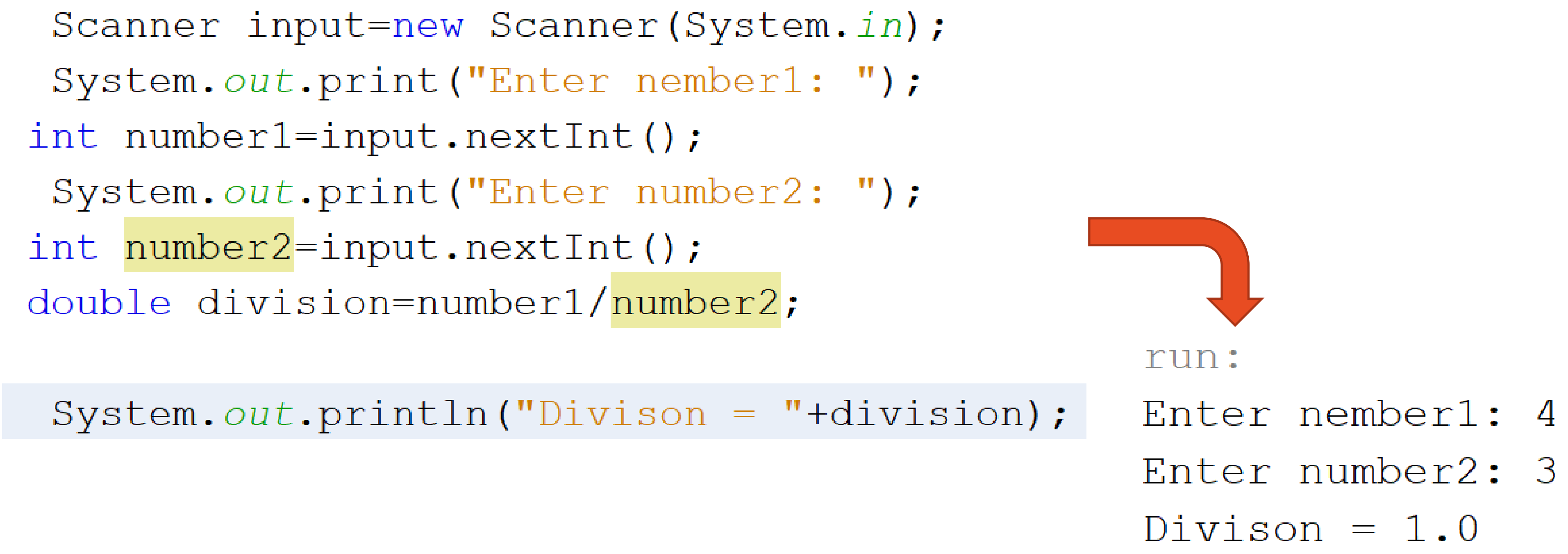
Try and See The OUTPUT

Casting Variables

- Type casting is when you assign a value of one primitive data type to another type.
- In Java, there are two types of casting:

1. ***Widening Casting (automatically)*** - converting a smaller type to a larger type size

byte -> short -> char -> int -> long -> float -> double



```
 Scanner input=new Scanner(System.in);
 System.out.print("Enter nember1: ");
int number1=input.nextInt();
 System.out.print("Enter number2: ");
int number2=input.nextInt();
double division=number1/number2;

 System.out.println("Divison = "+division);
```

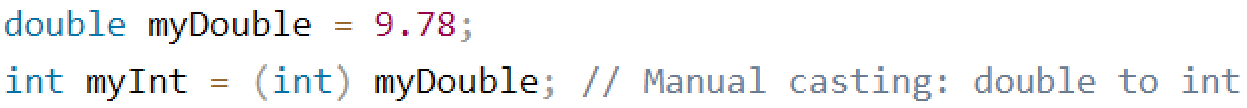
```
run:
Enter nember1: 4
Enter number2: 3
Divison = 1.0
```

Casting Variables

2. Narrowing Casting (manually) - converting a larger type to a smaller size type

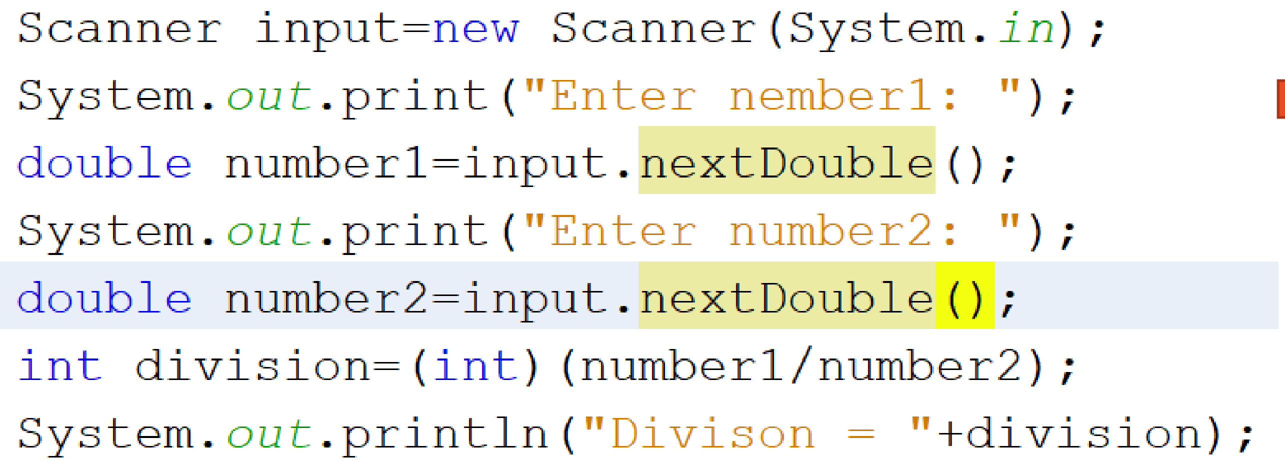
double -> float -> long -> int -> char -> short -> byte

- Narrowing casting must be done manually by placing the type in parentheses in front of the value:

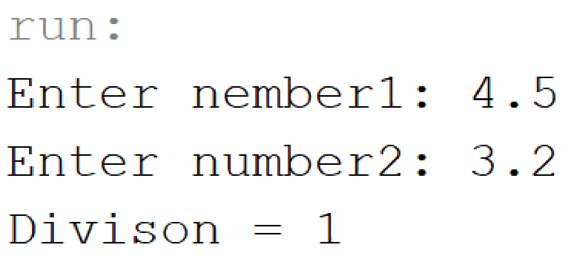


```
double myDouble = 9.78;
int myInt = (int) myDouble; // Manual casting: double to int
```

- **Example**



```
Scanner input=new Scanner(System.in);
System.out.print("Enter nember1: ");
double number1=input.nextDouble();
System.out.print("Enter number2: ");
double number2=input.nextDouble();
int division=(int)(number1/number2);
System.out.println("Divison = "+division);
```



```
run:
Enter nember1: 4.5
Enter number2: 3.2
Divison = 1
```

Example

- Develop a JAVA for a program to calculate employee income tax based on the following formula:

Tax = 0.25 * (monthly Salary * 11 – number of kids * 450)

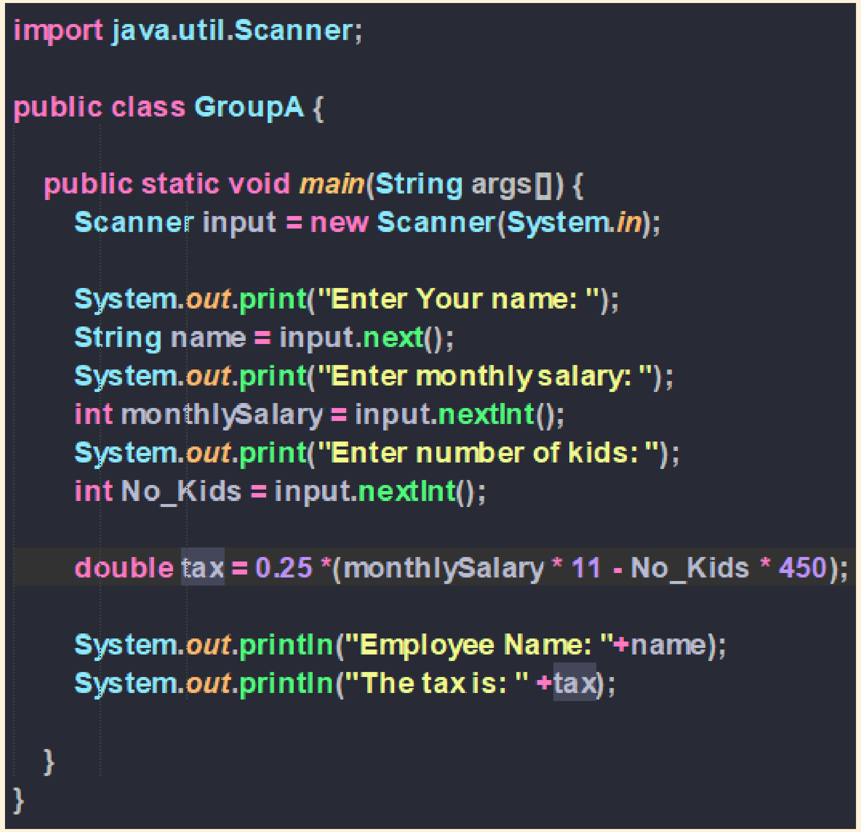
- Your program will display the name of the employee and amount of tax on the screen.

Pseudo-code

```
Start
    Create Scanner Object
    Ask the user to enter the employee name
    Create variable to store the name
    Ask user to enter monthly salary
    Create variable to store monthly salary
    Ask user to enter number of kids
    Create variable to store number of kids
    Create variable to calculate and store the tax
    Print the employee name and the tax value to the user
End
```

JAVA CODE

JAVA Code



```java
import java.util.Scanner;

public class GroupA {

  public static void main(String args[]) {
    Scanner input = new Scanner(System.in);

    System.out.print("Enter Your name: ");
    String name = input.next();
    System.out.print("Enter monthly salary: ");
    int monthlySalary = input.nextInt();
    System.out.print("Enter number of kids: ");
    int No_Kids = input.nextInt();

    double tax = 0.25 *(monthlySalary * 11 - No_Kids * 450);

    System.out.println("Employee Name: "+name);
    System.out.println("The tax is: " +tax);

  }
}
```

Try and See The OUTPUT

Sample Development

Write an application that inputs three integers from the user and displays the sum, average, product, of the numbers. [*Note:* The calculation of the average in this exercise should result in an integer representation of the average. So, if the sum of the values is 7, the average should be 2, not 2.3333….]