Java Classes and Main Function — Complete Cheatsheet

1■■ BASIC STRUCTURE OF A JAVA PROGRAM

Every Java program must have a class and a main method. Execution always starts from the main() method.

```
class Main { public static void main(String[] args) { System.out.println("Hello,
World!"); } }
```

Explanation:

• **class Main** → Defines a class (container for code)
• **public static void main(String[] args)** → Entry point of the program
• **System.out.println()** → Prints text to console
• **;** → Ends a statement

Keywords meaning:

• **public** → Accessible from anywhere
• **static** → Belongs to class, not object
• **void** → No return value
• **main** → Special entry point
• **String[] args** → Command line arguments

2■■ MULTIPLE CLASSES IN JAVA

You can define multiple classes in one Java file, but only one class should contain the main() method.

```
class Greeter { void sayHello(String name) { System.out.println("Hello, " + name +
"!"); } } class Main { public static void main(String[] args) { Greeter g = new
Greeter(); g.sayHello("Asmi"); } }
```

■ Output: Hello, Asmi!

Rules & Fun Facts:

- If only one public class → file name must match it (e.g., Main.java).
- Other classes can be non-public and inside same file.
- You can have multiple main() methods in different classes, but only one runs at a time.
- Classes can use each other — very common in OOP.

3■■ MULTIPLE MAINS EXAMPLE

```
class Hello { public static void main(String[] args) { System.out.println("Hello
class main"); } } class World { public static void main(String[] args) {
System.out.println("World class main"); } }
```

■ Run Hello → prints 'Hello class main' ■ Run World → prints 'World class main'

4■■ QUICK RECAP

• main() → entry point for JVM
• Other classes → helper/logic classes
• Every statement ends with ;
• Execution always starts from class containing main()

- Classes help organize and reuse code