Java Cheatsheet

Basics

Boilerplate

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

- `public` → Access modifier (needed for JVM to call `main`).
- `static` → Allows JVM to call without creating an object.
- `void` → Method returns nothing.
- `String[] args` → Command-line arguments.

Showing Output

```java
System.out.print("Hello");     // prints without newline
System.out.println(" World");  // prints with newline
System.out.printf("Age: %d", 20); // formatted output
```

Taking Input (Scanner)

```java
import java.util.Scanner;

class InputExample {
```

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

        String name = sc.nextLine();   // String
        int age = sc.nextInt();        // int
        float f = sc.nextFloat();      // float
        double d = sc.nextDouble();    // double
        boolean b = sc.nextBoolean();  // boolean
        char c = sc.next().charAt(0);  // char (first char of input)

        System.out.println("Name: " + name);
    }
}
```

Primitive Types

Eight primitives: `byte`, `short`, `int`, `long`, `float`, `double`, `boolean`, `char`.

- `byte` (8-bit, -128 to 127)
- `short` (16-bit, -32,768 to 32,767)
- `int` (32-bit, default for integers)
- `long` (64-bit, append `L` for literals)
- `float` (32-bit, append `f`)
- `double` (64-bit, default for decimals)
- `boolean` (`true` or `false`)
- `char` (16-bit Unicode)

Comments

```
// Single line
/* Multi-line */
/** Javadoc comment */
```

Constants

```
final double PI = 3.14159;
static final int MAX = 100;
```

Arithmetic Operators

`+ - * / % ++ --`

- Division between integers truncates result.
- `%` gives remainder.

Assignment Operators

`=, +=, -=, *=, /=, %=`

Comparison Operators

`==, !=, >, <, >=, <=`

Logical Operators

`&& (AND), || (OR), ! (NOT)`

Escape Sequences

- `\n` → newline
- `\t` → tab
- `\\` → backslash
- `\'` → single quote
- `\"` → double quote
- `\r` → carriage return
- `\b` → backspace

(`\?` is not valid in Java; use `"?"` literally.)

Type Casting

Widening (automatic)

```
int x = 45;
double y = x; // OK
```

Narrowing (manual)

```
double d = 45.9;
int n = (int) d; // truncates decimal
```

Control Flow

if / else if / else

```
if (x > 0) { ... }
else if (x == 0) { ... }
else { ... }
```

Ternary

```
String result = (x > 0) ? "Positive" : "Negative";
```

switch (Java 7+ supports Strings)

```
switch(day) {
    case 1: System.out.println("Mon"); break;
    ...
    default: System.out.println("Invalid");
}
```

Loops

while

```
while (i < 5) {
    i++;
}
```

do-while

```
do {
    i++;
} while (i < 5);
```

for

```
for (int i = 0; i < 5; i++) { ... }
```

for-each

```
for (int n : arr) { ... }
```

break / continue

- `break` → exits loop
- `continue` → skips iteration

Arrays

```
int[] nums = new int[5];
String[] names = {"Harry", "Rohan"};
System.out.println(names.length);
```

Multi-dimensional

```
int[][] matrix = {
  {1, 2, 3},
  {4, 5, 6}
};
System.out.println(matrix[1][2]); // 6
```

Methods

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

public static void main(String[] args) {
    System.out.println(add(5, 10));
}
```

Method Overloading

```
void sum(int a, int b) { ... }
void sum(double a, double b) { ... }
```

Recursion

```
int fact(int n) {
    if (n == 0) return 1;
    return n * fact(n - 1);
}
```

Strings

Strings are objects of `java.lang.String`.

```
String s = "Hello";
int len = s.length();
s.toUpperCase();
s.toLowerCase();
s.indexOf("l");
s.contains("He");
```

```
s.equals("Hello");
s.equalsIgnoreCase("hello");
s.substring(0, 3); // "Hel"
s.replace("H", "J");
```

Math Class

```
Math.max(5, 10);
Math.min(5, 10);
Math.sqrt(16);
Math.pow(2, 3);
Math.abs(-10);
Math.random(); // [0.0, 1.0)
```