📋 Cheat Sheet - Introducción a Java

🏗️ Estructura Básica de un Programa

```java
public class Main {
    public static void main(String[] args) {
        // Tu código aquí
        System.out.println("Hola, mundo!");
    }
}
```

Importante: El archivo debe llamarse `Main.java` (igual que la clase).

📊 Tipos de Datos Primitivos

Tipo	Tamaño	Rango	Ejemplo
`byte`	8 bits	-128 a 127	`byte edad = 18;`
`short`	16 bits	-32,768 a 32,767	`short año = 2024;`
`int`	32 bits	-2 mil millones a 2 mil millones	`int poblacion = 1000000;`
`long`	64 bits	Números muy grandes	`long distancia = 999999L;`
`float`	32 bits	Decimales (precisión simple)	`float peso = 72.5F;`
`double`	64 bits	Decimales (precisión doble)	`double altura = 1.75;`
`char`	16 bits	Un solo carácter	`char letra = 'A';`
`boolean`	1 bit	true o false	`boolean activo = true;`

💡 Tips:

- Usa `int` por defecto para enteros
- Usa `double` por defecto para decimales
- Añade `L` al final de `long` y `F` al final de `float`

String - Cadenas de Texto

Crear un String

```java
String nombre = "Juan";
```

Operaciones Principales

Método	Descripción	Ejemplo
+ o .concat()	Unir cadenas	"Hola " + "Mundo" → "Hola Mundo"
.length()	Longitud	"Juan".length() → 4
.charAt(i)	Carácter en posición i	"Juan".charAt(0) → 'J'
.substring(i, j)	Subcadena	"Juan".substring(1, 3) → "ua"
.replace(old, new)	Reemplazar	"Juan".replace('J', 'L') → "Luan"
.toUpperCase()	A mayúsculas	"Juan".toUpperCase() → "JUAN"
.toLowerCase()	A minúsculas	"Juan".toLowerCase() → "juan"
.trim()	Eliminar espacios	" Hola ".trim() → "Hola"
.equals(otro)	Comparar (case-sensitive)	"Hola".equals("hola") → false
.equalsIgnoreCase()	Comparar (sin case)	"Hola".equalsIgnoreCase("hola") → true

Entrada y Salida

Salida - Imprimir en Consola

```java
System.out.println("Con salto de línea");
System.out.print("Sin salto de línea");

int edad = 25;
System.out.println("Edad: " + edad);  // Concatenar
```

Entrada - Leer del Teclado con Scanner

```java
import java.util.Scanner;  // ¡No olvides importar!

Scanner scanner = new Scanner(System.in);

// Leer String (línea completa)
System.out.print("Nombre: ");
String nombre = scanner.nextLine();

// Leer int
System.out.print("Edad: ");
int edad = scanner.nextInt();
scanner.nextLine();  // ⚠️ Limpiar buffer

// Leer double
System.out.print("Altura: ");
double altura = scanner.nextDouble();

scanner.close();  // Cerrar cuando termines
```

⚠️ **Truco importante:** Después de `nextInt()` o `nextDouble()`, usa `scanner.nextLine()` para limpiar el buffer antes de leer otro String.

🔧 Funciones (Métodos)

Estructura General

```java
public static tipoRetorno nombreFuncion(parametros) {
    // código
    return valor;  // Si no es void
}
```

Componentes

- `public`: Accesible desde cualquier lugar
- `static`: Pertenece a la clase (no necesita objeto)
- `tipoRetorno`: `int`, `double`, `String`, `void` (sin retorno)
- `parametros`: `(int a, String b)` o vacío `()`

Ejemplos

java

```java
// Función que suma y devuelve resultado
public static int sumar(int a, int b) {
    return a + b;
}

// Función que NO devuelve nada
public static void saludar(String nombre) {
    System.out.println("Hola, " + nombre);
}

// Función sin parámetros
public static void imprimirMenu() {
    System.out.println("1. Opción 1");
    System.out.println("2. Opción 2");
}

// Llamar funciones en main
public static void main(String[] args) {
    int resultado = sumar(5, 3);      // resultado = 8
    saludar("Ana");                   // Imprime: Hola, Ana
    imprimirMenu();                   // Imprime el menú
}
```

📚 ArrayList - Listas Dinámicas

Importar y Crear

java

```java
import java.util.ArrayList;

ArrayList<String> nombres = new ArrayList<>();
ArrayList<Integer> numeros = new ArrayList<>();
```

Nota: Para tipos primitivos usa: `Integer`, `Double`, `Boolean`, etc.

Operaciones Principales

Operación	Método	Ejemplo
Agregar al final	`.add(elemento)`	`nombres.add("Juan");`
Insertar en posición	`.add(index, elem)`	`nombres.add(1, "Ana");`
Obtener elemento	`.get(index)`	`String n = nombres.get(0);`
Modificar elemento	`.set(index, elem)`	`nombres.set(1, "Lucía");`
Eliminar por índice	`.remove(index)`	`nombres.remove(0);`
Eliminar por valor	`.remove(objeto)`	`nombres.remove("Juan");`
Tamaño	`.size()`	`int tam = nombres.size();`
Está vacío	`.isEmpty()`	`if (nombres.isEmpty())`
Contiene elemento	`.contains(elem)`	`if (nombres.contains("Ana"))`
Limpiar todo	`.clear()`	`nombres.clear();`

Recorrer un ArrayList

```java
// Forma 1: for-each (recomendado)
for (String nombre : nombres) {
    System.out.println(nombre);
}

// Forma 2: for tradicional
for (int i = 0; i < nombres.size(); i++) {
    System.out.println(nombres.get(i));
}
```

Ejemplo Completo

java

```java
import java.util.ArrayList;

public class EjemploLista {
    public static void main(String[] args) {
        ArrayList<String> frutas = new ArrayList<>();

        // Agregar elementos
        frutas.add("Manzana");
        frutas.add("Banana");
        frutas.add("Naranja");

        // Insertar en posición 1
        frutas.add(1, "Pera");

        // Modificar
        frutas.set(0, "Mango");

        // Eliminar
        frutas.remove("Banana");

        // Recorrer e imprimir
        System.out.println("Frutas:");
        for (String fruta : frutas) {
            System.out.println("- " + fruta);
        }

        // Tamaño
        System.out.println("Total: " + frutas.size());
    }
}
```

Salida:

```
Frutas:
- Mango
- Pera
- Naranja
Total: 3
```

🎯 Consejos Rápidos

✅ Buenas Prácticas

- Nombres de variables: `camelCase` → `nombreCompleto`, `edadUsuario`
- Nombres de clases: `PascalCase` → `Main`, `Calculadora`
- Indenta correctamente tu código
- Comenta las partes complicadas
- Cierra el Scanner cuando termines

⚠️ Errores Comunes

- Olvidar `import` para Scanner o ArrayList
- No cerrar las comillas: `"texto`
- Olvidar punto y coma `;` al final
- Confundir `=` (asignación) con `==` (comparación)
- No limpiar el buffer después de `nextInt()` o `nextDouble()`
- Intentar usar `==` para comparar Strings (usa `.equals()`)

💡 Para Recordar

- Los **índices** empiezan en **0**
- `String` se compara con `.equals()`, no con `==`
- Después de `nextInt()` usa `scanner.nextLine()` para limpiar
- `ArrayList` crece automáticamente, los arrays no
- `void` significa que la función no devuelve nada

📖 Plantilla de Programa Completo

java

```java
import java.util.Scanner;
import java.util.ArrayList;

public class MiPrograma {

    // Función auxiliar
    public static void mostrarMenu() {
        System.out.println("=== MENÚ ===");
        System.out.println("1. Opción 1");
        System.out.println("2. Salir");
    }

    // Función principal
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ArrayList<String> datos = new ArrayList<>();

        // Tu código aquí
        System.out.print("Ingrese su nombre: ");
        String nombre = scanner.nextLine();

        System.out.println("Hola, " + nombre);

        scanner.close();
    }
}
```

🚀 **¡A practicar! La programación se aprende programando.**