ARRAY

Cos'è un Array?

Un array è una sequenza di elementi dello stesso tipo. Funziona come una fila di cassetti numerati:

- Ogni cassetto ha un numero (indice) che parte da 0
- Puoi mettere o prendere valori conoscendo il numero del cassetto
- La dimensione viene decisa quando lo crei e non può cambiare
- Puoi sapere quanti cassetti ci sono con la proprietà length

Come si Usa?

```
// Creazione
int[] numeri = new int[5];         // array vuoto di 5 elementi
int[] voti = {8, 7, 6, 9, 7};      // array già con valori

// Accesso agli elementi
numeri[0] = 10;                    // mette 10 nella prima posizione
int primoVoto = voti[0];           // legge il primo voto

// Scorrere un array
for(int i = 0; i < voti.length; i++) {
    System.out.println(voti[i]);
}
```

Operazioni Comuni

1. **Ricerca di un Elemento**

```
public static int trova(int[] array, int numero) {
    for(int i = 0; i < array.length; i++) {
        if(array[i] == numero) {
            return i;  // posizione trovata
        }
    }
    return -1;  // elemento non trovato
}
```

2. **Trovare Massimo e Minimo**

```java
public static int trovaMax(int[] array) {
    if(array.length == 0) {
        System.out.println("Array vuoto!");
        return -1;
    }

    int max = array[0];
    for(int i = 1; i < array.length; i++) {
        if(array[i] > max) {
            max = array[i];
        }
    }
    return max;
}
```

3. **Copia di un Array**

```java
public static int[] copia(int[] array) {
    int[] nuovo = new int[array.length];
    for(int i = 0; i < array.length; i++) {
        nuovo[i] = array[i];
    }
    return nuovo;
}
```

Errori Comuni con gli Array

1. **Indice Fuori dai Limiti**
   - Ricorda: gli indici vanno da 0 a length-1
   - Esempio sbagliato: `array[array.length]`
2. **Array Vuoto**
   - Controlla sempre se l'array è vuoto prima di operare
   - Usa `if(array.length == 0)`
3. **Modifiche Inaspettate**
   - Gli array sono passati per riferimento
   - Se modifichi l'array in un metodo, cambia anche fuori

CLASSI

Cos'è una Classe?

Una classe è come un progetto per creare oggetti. Esempio: la classe Studente è il progetto, mentre Mario Rossi è un oggetto Studente specifico.

Struttura di una Classe

```java
public class Studente {
    // 1. ATTRIBUTI (sempre private!)
    private String nome;
    private String cognome;
    private int età;

    // 2. COSTRUTTORE
    public Studente(String nome, String cognome, int età) {
        this.nome = nome;      // this.nome è l'attributo
        this.cognome = cognome;// nome è il parametro
        this.età = età;
    }

    // 3. METODI
    // Get - per leggere gli attributi
    public String getNome() {
        return nome;
    }

    // Set - per modificare gli attributi
    public void setNome(String nome) {
        this.nome = nome;
    }

    // toString - per stampare l'oggetto
    public String toString() {
        return nome + " " + cognome + ", età: " + età;
    }
}
```

Come si Usa una Classe?

```java
// Creazione di un oggetto (istanza)
Studente s1 = new Studente("Mario", "Rossi", 16);

// Uso dei metodi
System.out.println(s1.getNome());     // legge il nome
s1.setNome("Luigi");                  // cambia il nome
System.out.println(s1.toString());    // stampa tutto
```

Regole Importanti

1. **Attributi**
    - Sempre private
    - Rappresentano le caratteristiche dell'oggetto
    - Si accede solo tramite metodi get/set

2. **Costruttore**
    - Ha lo stesso nome della classe
    - Serve per creare nuovi oggetti
    - Inizializza tutti gli attributi
3. **This**
    - Si usa quando parametro e attributo hanno lo stesso nome
    - `this.nome` si riferisce all'attributo della classe
    - `nome` si riferisce al parametro del metodo

Come Testare una Classe

```
public class StudenteTester {
    public static void main(String[] args) {
        // 1. Crea oggetto
        Studente s1 = new Studente("Mario", "Rossi", 16);

        // 2. Verifica costruttore
        System.out.println("Test costruttore:");
        System.out.println(s1.toString());

        // 3. Verifica get/set
        System.out.println("\nTest getNome:");
        System.out.println("Nome: " + s1.getNome());

        s1.setNome("Luigi");
        System.out.println("\nDopo setNome:");
        System.out.println("Nuovo nome: " + s1.getNome());
    }
}
```

Errori Comuni con le Classi

1. **Attributi Public**
    - Mai fare gli attributi public
    - Usa sempre private e metodi get/set
2. **This Dimenticato**
    - Nel costruttore, usa sempre this quando necessario
    - Esempio: `this.nome = nome;`
3. **Costruttore Incompleto**
    - Inizializza sempre tutti gli attributi
    - Non lasciare attributi non inizializzati

Suggerimenti per il Testing

1. Testa ogni metodo separatamente
2. Verifica i casi normali e i casi limite
3. Stampa i risultati per controllare
4. Commenta il codice per capire cosa stai testando