**Java Arrays – Topic 3: Thinking & Manipulating Arrays**

**1. Introduction**

* Meaning of array manipulation
* Why manipulation is needed in real programs

**2. Categories of Array Operations**

* **Analysis Operations** – calculations on data
* **Search Operations** – finding elements
* **Modification Operations** – changing values or order

**3. Analysis Operations 🧠**

**3.1 Sum of All Elements**

* Definition
* Logic explanation
* Code example
* Dry run table

**3.2 Average of Elements**

* Definition & formula
* Type casting to avoid integer division
* Code example

**3.3 Finding Maximum Value**

* Definition
* Step-by-step logic
* Code example

**3.4 Finding Minimum Value**

* Definition
* Step-by-step logic
* Code example

**4. Search Operations 🔍**

**4.1 Searching for a Value (Linear Search)**

* Definition
* Code example
* Using boolean found

**4.2 Finding Index of a Value**

* Definition
* Code example with for loop

**5. Modification Operations 🛠️**

**5.1 Increasing All Elements**

* Adding a fixed value to all elements
* Code example

**5.2 Multiplying All Elements**

* Multiplying all elements by a number
* Code example

**5.3 Reversing an Array**

* Swapping logic explanation
* Code example

**5.4 Left Shift**

* Definition
* Logic with first element stored temporarily
* Code example

**5.5 Right Shift**

* Definition
* Logic with last element stored temporarily
* Code example

**6. Special Problems ⭐**

**6.1 Finding Second Largest Element**

* Two-pass approach
* Explanation

**6.2 Removing Duplicates (Basic Method)**

* Using nested loops
* Limitations

**6.3 Copying an Array**

* Manual copy with loop
* Using Arrays.copyOf()

**Java Arrays – Topic 3: Thinking & Manipulating Arrays**

*(Full Detailed Student Edition)*

**1. Introduction**

📖 **Definition:**  
Array manipulation means **reading, analyzing, and modifying the values stored in an array** to achieve a specific result.

💡 **Why it matters:**  
In real-world programs, we don’t just store data — we calculate, compare, search, or modify it. For example:

* In a marks list → finding topper, calculating average.
* In a stock prices list → finding highest price.

**2. Categories of Array Operations**

We can split all array work into **three major types**:

1. **Analysis Operations** – Calculations like sum, average, min, max.
2. **Search Operations** – Finding values or positions.
3. **Modification Operations** – Changing or rearranging values.

**3. Analysis Operations 🧠**

**3.1 Sum of All Elements**

📖 **Definition:** The process of adding all values in the array to get a total.

💡 **Why important:** Used in statistics, finance totals, marks totals, etc.

**Example:**

int[] arr = {10, 20, 30, 40, 50};

int sum = 0;

for (int num : arr) {

sum += num;

}

System.out.println("Sum = " + sum);

**Step-by-step Dry Run:**

| **Step** | **num** | **sum (before)** | **sum (after)** |
| --- | --- | --- | --- |
| 1 | 10 | 0 | 10 |
| 2 | 20 | 10 | 30 |
| 3 | 30 | 30 | 60 |
| 4 | 40 | 60 | 100 |
| 5 | 50 | 100 | 150 |

**3.2 Average of Elements**

📖 **Definition:** Total sum divided by the number of elements.

**Formula:**

average = sum / arr.length;

*(Use (double) for accurate decimal results)*

**Example:**

double avg = (double) sum / arr.length;

System.out.println("Average = " + avg);

**3.3 Finding Maximum Value**

📖 **Definition:** The largest element in the array.

💡 **Why important:** To find the highest marks, highest sales, etc.

**Example:**

int max = arr[0];

for (int num : arr) {

if (num > max) {

max = num;

}

}

System.out.println("Max = " + max);

**Dry Run for {15, 50, 22, 90, 10}:**

| **Step** | **num** | **max (before)** | **Condition (num > max)** | **max (after)** |
| --- | --- | --- | --- | --- |
| 1 | 15 | 15 | false | 15 |
| 2 | 50 | 15 | true | 50 |
| 3 | 22 | 50 | false | 50 |
| 4 | 90 | 50 | true | 90 |
| 5 | 10 | 90 | false | 90 |

**3.4 Finding Minimum Value**

📖 **Definition:** The smallest element in the array.

**Example:**

int min = arr[0];

for (int num : arr) {

if (num < min) {

min = num;

}

}

System.out.println("Min = " + min);

**4. Search Operations 🔍**

**4.1 Searching for a Value (Linear Search)**

📖 **Definition:** Check each element one by one to see if it matches the target.

💡 **When used:** When data is unsorted.

**Example:**

int search = 30;

boolean found = false;

for (int num : arr) {

if (num == search) {

found = true;

break;

}

}

if (found)

System.out.println("Found");

else

System.out.println("Not Found");

**4.2 Finding Index of a Value**

📖 **Definition:** Return the position where the element is stored.

**Example:**

int search = 30;

for (int i = 0; i < arr.length; i++) {

if (arr[i] == search) {

System.out.println("Found at index " + i);

break;

}

}

**5. Modification Operations 🛠️**

**5.1 Increasing All Elements**

📖 **Definition:** Adding a fixed number to each element.

**Example:**

for (int i = 0; i < arr.length; i++) {

arr[i] += 5;

}

**5.2 Multiplying All Elements**

📖 **Definition:** Scaling each element by multiplying with a constant.

**Example:**

for (int i = 0; i < arr.length; i++) {

arr[i] \*= 2;

}

**5.3 Reversing an Array**

📖 **Definition:** Change the order of elements so the last becomes first.

**Logic:** Swap first with last, second with second-last, etc.

**Example:**

**Int[] arr={10,20,30};**

int start = 0, end = arr.length - 1;

while (start < end) {

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

**5.4 Left Shift**

📖 **Definition:** Move all elements one position left, first element goes to last position.

**Example:**

int first = arr[0];

for (int i = 0; i < arr.length - 1; i++) {

arr[i] = arr[i + 1];

}

arr[arr.length - 1] = first;

**5.5 Right Shift**

📖 **Definition:** Move all elements one position right, last element becomes first.

**Example:**

int last = arr[arr.length - 1];

for (int i = arr.length - 1; i > 0; i--) {

arr[i] = arr[i - 1];

}

arr[0] = last;

**6. Special Problems ⭐**

**6.1 Second Largest Element**

📖 **Logic:** Find max, then find next max ignoring the first.

**6.2 Removing Duplicates (Basic Method)**

📖 **Logic:** Compare each element with all others, skip repeats.

**6.3 Copying an Array**

📖 **Logic:** Create a new array of same size and copy elements.