# Array Techniques, Sorting & Searching - Cheat Sheet (Theory Only)

---

## Array Techniques

### 1. Array Order Reversal

- **Reverses the elements of an array.**
- **Algorithm (Using Two-Pointer Method):**

```
Start
Initialize two pointers: Left at 0, Right at N-1
While Left < Right
   Swap Arr[Left] and Arr[Right]
   Increment Left, Decrement Right
End While
End
```

---

### 2. Array Counting (Histogramming)

- **Counts occurrences of each element in an array.**
- **Example: Counting frequency of digits (0-9) in an array.**

```
Start
Initialize Count[10] to 0
For each element in Array
   Increment Count[element]
End For
Print Count
End
```

---

### 3. Maximum and Minimum of a Set

- **Finds the largest and smallest elements in an array.**
- **Algorithm:**

```
Start
Initialize Max = Arr[0], Min = Arr[0]
For i = 1 to N-1
   If Arr[i] > Max, Update Max
   If Arr[i] < Min, Update Min
End For
Print Max, Min
End
```

---

### 4. Removal of Duplicates from an Array

- **Removes duplicate values, keeping only unique elements.**
- **Algorithm (Using Sorting & One-Pass Check):**

```
Start
Sort the Array
Create New_Array
For i = 0 to N-1
```

```
   If Arr[i] ≠ Arr[i+1], Add Arr[i] to New_Array
End For
Print New_Array
End
```

---

## 5. Partitioning an Array

- **Divides an array into two parts based on a pivot value.**
- **Used in QuickSort algorithm.**
- **Algorithm:**

```
Start
Choose Pivot (e.g., Last Element)
Initialize Left = 0, Right = N-1
While Left < Right
   If Arr[Left] < Pivot, Move Left
   If Arr[Right] > Pivot, Move Right
   Else Swap Arr[Left] and Arr[Right]
End While
End
```

---

## 6. Longest Monotone Subsequence

- **Finds the longest increasing or decreasing subsequence in an array.**
- **Algorithm (Increasing Subsequence):**

```
Start
Initialize Length = 1, Max_Length = 1
For i = 1 to N-1
   If Arr[i] > Arr[i-1], Increment Length
   Else Reset Length to 1
   If Length > Max_Length, Update Max_Length
End For
Print Max_Length
End
```

---

# Sorting Algorithms

## 1. Bubble Sort

- **Repeatedly swaps adjacent elements if they are in the wrong order.**
- **Time Complexity: $O(N^2)$**
- **Algorithm:**

```
Start
For i = 0 to N-1
   For j = 0 to N-i-1
      If Arr[j] > Arr[j+1], Swap them
End For
Print Sorted Array
End
```

---

## 2. Selection Sort

- **Finds the smallest element and swaps it with the first element.**
- **Time Complexity: O($N^2$)**
- **Algorithm:**

```
Start
For i = 0 to N-1
   Find the minimum element from Arr[i] to Arr[N-1]
   Swap it with Arr[i]
End For
Print Sorted Array
End
```

---

## 3. Insertion Sort

- **Inserts each element in its correct position in a sorted part of the array.**
- **Time Complexity: O($N^2$)**
- **Algorithm:**

```
Start
For i = 1 to N-1
   Store Arr[i] in Temp
   j = i - 1
   While j >= 0 and Arr[j] > Temp
      Shift Arr[j] to Arr[j+1]
      Decrement j
   Insert Temp at Arr[j+1]
End For
Print Sorted Array
End
```

---

# Searching Algorithms

## 1. Linear Search

- **Checks each element one by one.**
- **Time Complexity: O(N)**
- **Algorithm:**

```
Start
Input Array and Target
For i = 0 to N-1
   If Arr[i] == Target, Print "Found" and Stop
End For
Print "Not Found"
End
```

---

## 2. Binary Search

- **Works on a sorted array by dividing the search space in half.**
- **Time Complexity: O(log N)**
- **Algorithm:**

```
Start
```

```
• Input Sorted Array and Target
• Initialize Low = 0, High = N-1
• While Low ≤ High
•    Mid = (Low + High) / 2
•    If Arr[Mid] == Target, Print "Found" and Stop
•    Else If Arr[Mid] > Target, High = Mid - 1
•    Else Low = Mid + 1
• End While
• Print "Not Found"
• End
•
```

---

# Pseudocode & Flow Chart Example

## Example: Bubble Sort Pseudocode

```
Start
For i = 0 to N-1
   For j = 0 to N-i-1
      If Arr[j] > Arr[j+1]
         Swap Arr[j] and Arr[j+1]
   End For
End For
Print Sorted Array
End
```

## Flowchart Symbols

| Symbol | Meaning |
|---|---|
| **Oval** | Start/End |
| **Rectangle** | Process (Calculation) |
| **Diamond** | Decision (If/Else) |
| **Parallelogram** | Input/Output |

---

This **Sorting & Searching Cheat Sheet** covers **array techniques, sorting (Bubble, Selection, Insertion), searching (Linear, Binary), pseudocode, and flowcharts**. Let me know if you need further explanations!