1. **Arrays Overview / نظرة عامة على المصفوفات**
    - **Array / المصفوفة**: A collection of variables of the same type, stored in contiguous memory locations. Each element in the array is accessed using an index.
      مجموعة من المتغيرات من نفس النوع، مخزنة في مواقع متجاورة في الذاكرة. يتم الوصول إلى كل عنصر في المصفوفة باستخدام فهرس.
        - **Example / مثال**: int[] numbers = new int[5]; creates an array of 5 integers.
          int[] numbers = new int[5]; تنشئ مصفوفة من 5 أعداد صحيحة.
    - **Purpose / الغرض**: Arrays are used to store multiple values in a single variable, making it easier to manage and manipulate large sets of data.
      تُستخدم المصفوفات لتخزين عدة قيم في متغير واحد، مما يسهل إدارة ومعالجة مجموعات كبيرة من البيانات.
2. **One-Dimensional Arrays / المصفوفات أحادية البعد**
    - **Declaration / التصريح عن المصفوفة**:
        - **Syntax / الصيغة**: BaseType[] arrayName = new BaseType[length];
        - **Example / مثال**: int[] counts = new int[4]; creates an array of 4 integers.
          int[] counts = new int[4]; تنشئ مصفوفة من 4 أعداد صحيحة.
    - **Initialization / التهيئة**:
        - At declaration / عند التصريح: int[] a = {1, 2, 3, 4};
        - Using loops / باستخدام الحلقات:

```
java

Copy

for (int i = 0; i < a.length; i++) {
    a[i] = i;
}
```

    - **Accessing Elements / الوصول إلى العناصر**:
        - Use the index / استخدم الفهرس: arrayName[index]
        - **Example / مثال**: counts[0] = 7; assigns the value 7 to the first element.
          counts[0] = 7; تعيين القيمة 7 للعنصر الأول.
    - **Length / الطول**:
        - The number of elements in an array is accessed using arrayName.length.
          يتم الوصول إلى عدد العناصر في المصفوفة باستخدام arrayName.length.
        - **Example / مثال**: counts.length returns 4 for the array counts.
          counts.length تُرجع القيمة 4 للمصفوفة counts.
3. **Partially Filled Arrays / المصفوفات المملوءة جزئيًا**
    - **Partially Filled Array / المصفوفة المملوءة جزئيًا**: An array where not all elements are used.
      مصفوفة لا يتم استخدام جميع عناصرها.
        - **Example / مثال**: An array of size 10 may only have 5 elements filled.
          مصفوفة بحجم 10 قد تحتوي فقط على 5 عناصر مملوءة.
        - Use a separate variable to track the number of used elements.
          استخدم متغيرًا منفصلًا لتتبع عدد العناصر المستخدمة.
4. **Multidimensional Arrays / المصفوفات متعددة الأبعاد**

- **Two-Dimensional Arrays / المصفوفات ثنائية الأبعاد**:
  - **Syntax / الصيغة**: BaseType[][] arrayName = new BaseType[rows][columns];
  - **Example / مثال**: int[][] table = new int[10][6]; creates a 10x6 table.
    int[][] table = new int[10][6]; تنشئ جدولًا بحجم 10x6.
  - Accessing elements / الوصول إلى العناصر: table[3][2] accesses the element at row 3, column 2.
    table[3][2] يصل إلى العنصر في الصف 3 والعمود 2.
- **Ragged Arrays / المصفوفات غير المنتظمة**: Arrays where rows can have different lengths.
  مصفوفات يمكن أن تحتوي صفوفًا بأطوال مختلفة.
  - **Example / مثال**: int[][] b = new int[3][]; where each row can have a different number of columns.
    int[][] b = new int[3][]; حيث يمكن أن يكون لكل صف عدد مختلف من الأعمدة.

5. **Array Traversal / اجتياز المصفوفات**
   - **Traversal / الاجتياز**: Looping through the elements of an array.
     التكرار خلال عناصر المصفوفة.
     - **Example / مثال**:

```
java

Copy

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

   - **Searching / البحث**: Traversing an array to find a specific element.
     اجتياز المصفوفة للعثور على عنصر معين.
     - **Example / مثال**:

```
java

Copy

public static int search(double[] array, double target) {
    for (int i = 0; i < array.length; i++) {
        if (array[i] == target) {
            return i;
        }
    }
    return -1; // not found / غير موجود
}
```

6. **Copying Arrays / نسخ المصفوفات**
   - **Shallow Copy / النسخ السطحي**: Copying the reference of an array.
     نسخ عنوان المصفوفة.
     - **Example / مثال**: int[] b = a; makes b refer to the same array as a.
       int[] b = a; يجعل b يشير إلى نفس المصفوفة التي يشير إليها a.

- **Deep Copy / النسخ العميق**: Copying the elements of an array to a new array. نسخ عناصر المصفوفة إلى مصفوفة جديدة.
  - **Example / مثال**:

```java
int[] b = new int[a.length];
for (int i = 0; i < a.length; i++) {
    b[i] = a[i];
}
```

7. **Arrays in Methods / المصفوفات في الدوال**
   - **Passing Arrays to Methods / تمرير المصفوفات إلى الدوال**:
     - **Syntax / الصيغة**: public static void methodName(BaseType[] arrayName)
     - **Example / مثال**:

```java
public static void incrementArrayBy2(double[] anArray) {
    for (int i = 0; i < anArray.length; i++) {
        anArray[i] += 2;
    }
}
```

   - **Returning Arrays from Methods / إرجاع المصفوفات من الدوال**:
     - **Syntax / الصيغة**: public static BaseType[] methodName(Parameter_List)
     - **Example / مثال**:

```java
public static int[] getArray() {
    int[] temp = {1, 2, 3};
    return temp;
}
```

8. **Common Array Operations / العمليات الشائعة على المصفوفات**
   - **Summing Elements / جمع العناصر**:

```java
public static double sum(double[] array) {
    double total = 0.0;
    for (int i = 0; i < array.length; i++) {
        total += array[i];
    }
```

```
    return total;
}
```

- **Finding Minimum/Maximum / إيجاد القيمة الصغرى/العظمى**:

```
java

Copy

public static double findMin(double[] array) {
    double min = array[0];
    for (int i = 1; i < array.length; i++) {
        if (array[i] < min) {
            min = array[i];
        }
    }
    return min;
}
```

9. **Array Equality / مساواة المصفوفات**
   - **Equality Check / فحص المساواة**:
     - == checks if two arrays refer to the same memory location.
       == يتحقق مما إذا كانت المصفوفتان تشيران إلى نفس الموقع في الذاكرة.
     - To check if two arrays have the same elements, use a custom method:
       للتحقق مما إذا كانت المصفوفتان تحتويان على نفس العناصر، استخدم دالة مخصصة:

```
java

Copy

public static boolean equals(int[] a, int[] b) {
    if (a.length != b.length) return false;
    for (int i = 0; i < a.length; i++) {
        if (a[i] != b[i]) return false;
    }
    return true;
}
```

10. **Summary of Key Concepts / ملخص المفاهيم الرئيسية**
    - Arrays are used to store multiple values of the same type.
      تُستخدم المصفوفات لتخزين عدة قيم من نفس النوع.
    - One-dimensional arrays are accessed using a single index.
      يتم الوصول إلى المصفوفات أحادية البعد باستخدام فهرس واحد.
    - Multidimensional arrays are arrays of arrays, accessed using multiple indices.
      المصفوفات متعددة الأبعاد هي مصفوفات من مصفوفات، ويتم الوصول إليها باستخدام عدة فهارس.
    - Array traversal is used to search, sum, or manipulate elements.
      يُستخدم اجتياز المصفوفات للبحث أو الجمع أو معالجة العناصر.
    - Arrays can be passed to methods and returned from methods.
      يمكن تمرير المصفوفات إلى الدوال وإرجاعها منها.
    - Array equality requires checking each element, not just the reference.
      تتطلب مساواة المصفوفات التحقق من كل عنصر، وليس فقط العنوان.