PROGRAMMING FUNDAMENTALS

WEEK 9: ARRAYS IN C

TABLE OF CONTENTS

1D ARRAY

SECTION 1

1D ARRAY

- Fixed-size collection of same-type elements

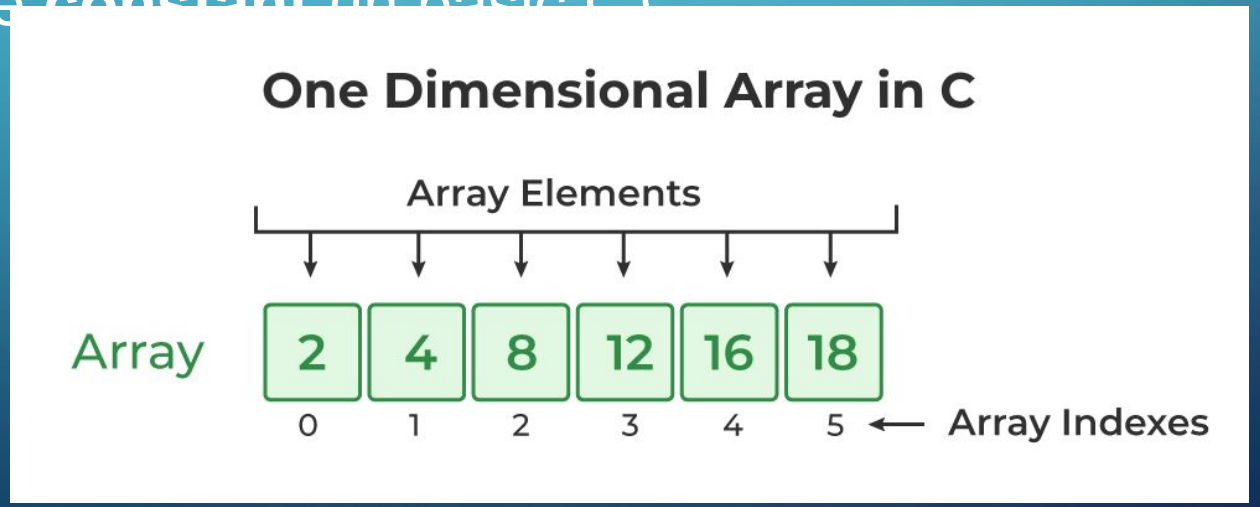
Syntax:

- `int arr[5]; // Declaration`
- `int arr[5] = {1,2,3,4,5}; // Initialization`

1D ARRAY

Key Points:

- Index starts at **0**
- Access: arr[i]
- Size must be **constant** (in basic C)



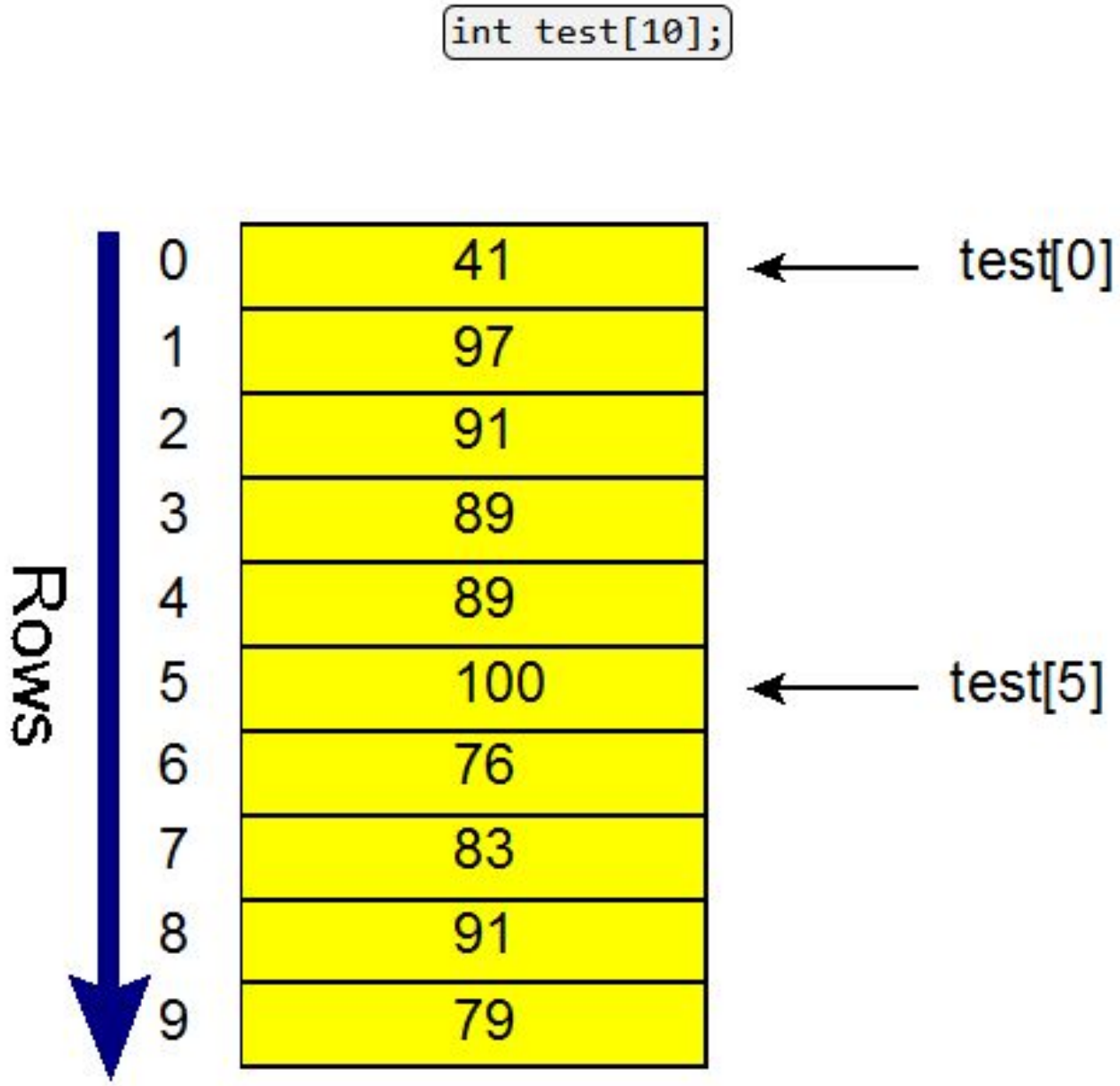


Figure 2. A one-dimensional array. The array definition at the top creates a one-dimensional array of 10 integer elements or variables arranged in rows numbered from 0 to 9.

1D ARRAY – PRACTICE QUESTION 1

- **Problem**: *Find the sum of all elements in an array.*
- **Why this?** Simple, reinforces loops + array access.
- **Show**:

Input: int arr[] = {10, 20, 30, 40};
Expected output: 100

1D ARRAY – PRACTICE QUESTION 1

Code

```
#include <stdio.h>
int main() {
    int arr[] = {10, 20, 30, 40, 50};
    int n = 5;
    int sum = 0;
    for (int i = 0; i < n; i++) {
        sum = sum + arr[i];
    }
    printf("Sum of array elements = %d\n", sum);
    return 0;
}
```

1D ARRAY – PRACTICE QUESTION 2

Problem: Find the maximum element in an array.

Input: {5, 12, 3, 18, 7} → Output: 18

Concepts: Comparison, tracking max

1D ARRAY – PRACTICE QUESTION 2

Tip: Initialize max = arr[0]

mistake: initializing max = 0 (fails for negative numbers)

1D ARRAY – PRACTICE QUESTION 2

Code

```
#include <stdio.h>
int main() {
    int arr[] = {5, 12, 3, 18, 7};
    int n = sizeof(arr) / sizeof(arr[0]);
    int max = arr[0];
    for (int i = 1; i < n; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
    }
    printf("Maximum element = %d\n", max);
    return 0;
}
```

ASSIGNMENT

1. C Program to reverse an array in place in a one-dimensional array.
   Input: {1,2,3,4} → Output: {4,3,2,1}
2. C Program to count even numbers in a one-dimensional array.
   Input: {1, 2, 3, 4} Output: 2
3. C Program to two largest numbers in a one-dimensional array.
   Input: {1, 2, 3, 4} Output: {3, 4}

Instructions:
Submit as .c files.
Use only concepts taught (no functions, pointers, etc.)