Chapter 7:- Arrays

An Array is Collection of Similar Elements. Array allows a single variable to take multiple value.

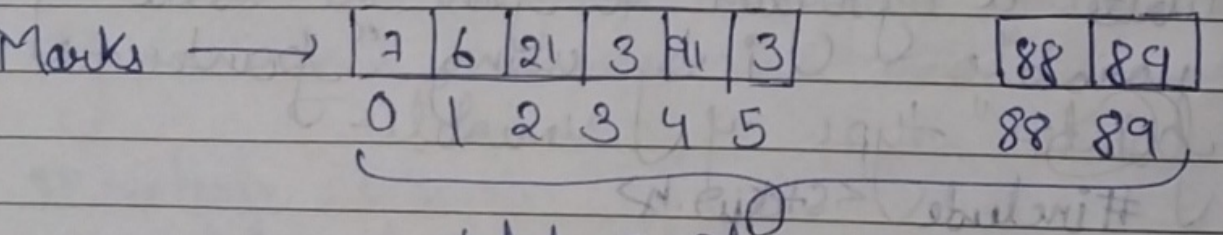
Syntax

```
int marks [90];
char name [20];
float percentile [90];
```

the values can now be assigned to make array like this

```
marks [0] = 33;
marks [1] = 12;
```

Note:- It is very important to note that the array index starts with 0



Accessing Elements

Elements of an array can be accessed using

```
scanf ("%d", &marks [0]);
printf ("%d", marks [0]);
```

Quick Quiz:

Write a program to accept marks of five Students in an array and print them on the Screen.

```
#include <stdio.h>
int main (){
    int marks [5];
for(int i=0; i<5; i++) {
    printf(" Enter marks of Students %d: ", i+1);
    Scanf("%d", &marks[i]);
    }
    printf("Marks Entered are: \n");
    for(int i=0; i<5; i++){
        printf(" Student %d: %d\n", i+1, marks[i]);
    }
    return 0;
}
```

Initialization of an Array.

There are many other ways in which an array can be initialized.

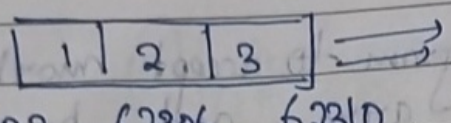
```
int cgpa[3] = {9, 8, 8};
float marks[] = {33.40};
```

Arrays in Memory

Consider this array

```
int arr[3] = {1, 2, 3}  // 1 integer = 4 Bytes.
```

This will reserve 4x3 = 12 bytes in memory (4 Bytes for Each integer).



Pointer Arithmetic

A pointer can be incremented to point to the next memory location of that type.

Consider this Example

```
{ int i = 32;
  int *a = &i; // a = 87994
  a++;         // address of i or value of a = 87998
```

```
{ char a = 'A';
  char *b = &a; // a = 87994
  b++;          // now a = 87995
```

```
{ float i = 1.7;
  float *a = &i; // a now a = 87994
  a++;           // now a = 87998
```

Following Operations can be performed on a pointer:

(1) Addition of a number to a pointer.
(2) Subtraction of a number from a pointer.
(3) Subtraction of one pointer from another.
(4) Comparison of two pointer variables

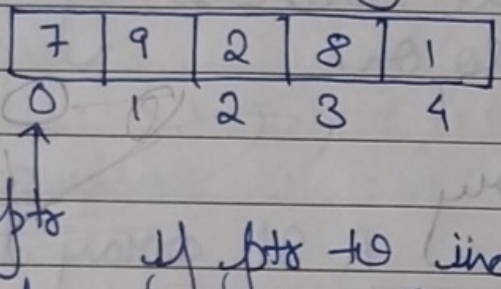
Quick Quiz

Try those Operations on another variable by Creating pointers in a Seperate program. Demostrate all the four Operations.

=>

Accessing Array Using pointers

Consider this array



If ptr to index 0, ptr++ will point to index 1 & So on.

This way we can have an integer pointer pointing to first Element of the array like this

```
int *ptr = &arr[0]; // Or Simple arr.
ptr++;
*ptr // will have 9 as its value
```

Passing Array to Functions

Array can be passed to the functions like this:

```
PrintArray(arr, n); // Function call
void PrintArray(int *i, int n); // Function Prototype
```

Or

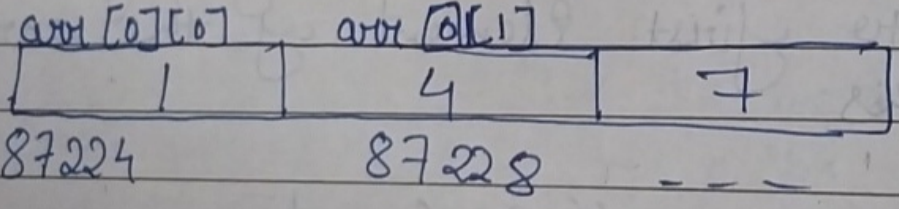
```
void PrintArray(int i[], int n);
```

Multidimensional Array An Array can be of 2 Dimension / 3 dimensions / n dimensions.
A 2 dimension array can be defined like this

```
int arr[3][2] = { {1, 4}
                  {7, 9}
                  {11, 22} };
```

we can access the Elements of this array as arr[0][1] & so on.

2D Array in Memory A 2d array like a 1d array stored in contiguous memory block like this:



Chapter 7 Practice Set

3 Create an array of 10 numbers Verify using pointer arithmetic that (ptr+2) points to the third Element where ptr is a pointer pointing to the first Element of the array.

```
#include <stdio.h>
int main()
{
```

```
int a[] = {1,2,3,4,5,6,7,8,9,10};
int *ptr = a;

printf("The value at address %u is %d", ptr+2, *(ptr+2));
return 0;
}
```

Q2) if S[3] is a 1-D array of integers then *(S+3) refers to the third element:-

(1) True
(2) false (✓)
(3) Depends

Q3) write a program to create an array of 10 integers and store multiplication table of 5 in it.

```
#include <stdio.h>
int main(){
    int arr[10];
    for (int i = 0; i<10; i++)
    {
        arr[i] = 5*(i+1);
    }
    for (int i = 0; i<10; i++)
    {
        printf("The value of 5 x %d = %d\n", i+1, arr[i]);
    }
    return 0;
}
```

Q4) Repeat problem 3 for a general output
→ Done Earlier.

Q5) Write a program containing a function which reverses the array passed to it

```
#include <stdio.h>
void reverse (int arr[], int n){
    for (int i=0; i<n/2; i++){
        int temp = arr[i];
        arr[i] = arr[n-i-1];
        arr[n-i-1] = temp;
    }
}
int main(){
    int arr[100];
    int n;
    printf("Enter the number of Elements
                in the array: ");
    Scanf("%d", &n);
    printf("Enter %d Elements:\n", n);
    for (int i=0; i<n; i++){
        Scanf("%d", &arr[i]);
    }
    reverse (arr, n);
    printf("Reversed array:\n");
    for (int i=0; i<n; i++){
        printf("%d", arr[i]);
    }
    return 0;
}
```

86) Create a three-dimensional array and print the address of its element in increasing order.

```
#include <stdio.h>
int main(){
    int arr[2][3][4];
    for(int i=0; i<2; i++){
        for(int j=0; j<3; j++){
            for(int k=0; k<4; k++){
                printf("%u", &arr[i][j][k]);
            }
        }
    }
    return 0;
}
```