ARRAYS

Passing An Array To A Function

- An Array can be passed to a function by
  - **Passing individual element**
    - Same way as normal variables are passed

```
for(i=0; i<5; i++)
 add(arr[i]);      //Function Call
```

    - Using a Pointer
  - **Passing an Entire array**
    - By passing only the name of the array which represents the base address
    - ➢ Using Subscript (**[]**) Operator
    - ➢ Using a Pointer

Pointer Arithmetic

- Basic arithmetic operations can be performed on pointer variables .
- But to perform these operations , the pointers must point to the elements of same array.

```
int arr[5] = {10, 20, 30, 40, 50}
int *p1, *p2;
p1 = &arr[0];
p2 = &arr[2];
p2 = p1;                    //Both pointers should hold same data type
p1 = p1+2;                  //moves pointer two locations forward
p2 = p2-2;                  //moves pointer two locations backward
p1++;                       //point to one location ahead of current location
p1--;                       //point to one location behind of current location
```

```
int howFar = p2 – p1;  //gives integer value depending on how many locations
                       //they are apart
if(p1 == p2)
printf(“Pointers point to same locations”);
else
printf(“Pointers don’t point to same locations”);
```

- The following operations are not performed on pointers :

1. Addition of two pointers
2. Multiplication of a pointer with number
3. Dividing a pointer with a number

Accessing Array elements in different ways

- When we say, **num[i]**, the C compiler internally converts it to ***(num+i)**.

- This means that following expressions are same:

- **num[i]**
- **i[num]**
- ***(num+i)**
- ***(i+num)**

Returning an Array