Functions with Arrays

* Like variables, it is also possible to pa the values of an array to a function
* To pass an array to a func[illegible], it is suffici to list the name of the array without any subscripts and the size of array as arguments.

Eg: largest (a, n);
will pass all the elements contained in array a of size n.

```
/* finding the largest value in an array of
                                   elements */
main( )
{
    float largest( );        /* Prototype declaration */
    float value [4]   =  { 2.5, -4.75, 1.2, 3.67 };
    float largest;    largest = largest (value, 4);
    printf ("%f \n", largest ~~(value, 4)~~);
}
```

When the function largest (value, 4) is called, the value of all the elements of the array "value" are passed to the elements of array 'a' of called function

```
float largest (a, n)
float a[ ];
int n;
{   int i;
    float max;
    max = a[0];
    for ( i=1; i<n; i++)
        if (max < a[i])
            max = a[i];
    return (max);
}
```

WAP to
an array
from
mean

Sol^n:-

WAP to calculate the standard deviation of an array of values. The array elements are read from the terminal. Use functions to calculate mean and standard deviation.

Solⁿ:- Standard deviation of a set of n values is given by:

$$\boxed{SD = \frac{1}{n}\sum_{i=1}^{n}(\bar{x} - x_i)^2}$$

where $\bar{x}$ is the mean of the values

- A multifunction program consisting of main(), std-dev() and ~~mai~~ mean() functions are defined
- main() reads the elements of the array "value" from the terminal and calls the function std-dev() to print the SD of the array elements.
- std-dev() calls another function mean() to get the average of ~~of~~ the array elements.
- The return value of both std-dev() and mean() are float and therefore they are declared in their calling functions

```
# include <stdio.h>
# include <math.h >
# define SIZE 5

main( )
{
    Float value[SIZE], std-dev( ), sd;
    int i;

    Printf (" Enter the array elements \n");
    for(i= 0 ; i< SIZE ; i++)
        scanf ("%f", &value[i] );
        sd = std-dev (value, SIZE);      // array passed as argument
        printf (" std. deviation is %f \n", sd);
}


Float std-dev (a, n)
Float a[ ];
int n;
{   int i;
    Float mean( ), x, sum = 0.0, sd;
    x = mean (a, n);               // array passing
    For (i=0; i< n; i++)
    sum + = (x - a[i]) * (x - a[i] )
    sd = sqrt (sum/(Float)n );
    return (sd);
}
```

```
float mean(a, n)
float a[];
int n;
{
    int i;
    Float sum = 0.0, avg;
    For(i=0; i<n; i++)
        sum = sum + a[i];
        avg = sum / (Float) n;
        return (avg);
}
```

$$SD = \frac{\sum_{i=1}^{n}(\bar{x} - x_i)^2}{n}$$

$\bar{x}$ = mean

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Difference between ordinary arguments (variables) and entire array passed as an argument

* When an entire array is passed an an argument, the contents of the actual parameter array are not copied into the formal parameter array. Instead, information about the address of array elements are passed on to the called function. Therefore, any changes introduced to the array elements are reflected in the original array in the calling function.