

# PASSING ARRAYS AS FUNCTION ARGUMENTS IN C

[http://www.tutorialspoint.com/cprogramming/c\\_passing\\_arrays\\_to\\_functions.htm](http://www.tutorialspoint.com/cprogramming/c_passing_arrays_to_functions.htm)

Copyright © tutorialspoint.com

If you want to pass a single-dimension array as an argument in a function, you would have to declare function formal parameter in one of following three ways and all three declaration methods produce similar results because each tells the compiler that an integer pointer is going to be received. Similar way you can pass multi-dimensional array as formal parameters.

## Way-1

Formal parameters as a pointer as follows. You will study what is pointer in next chapter.

```
void myFunction(int * param)
{
.
.
.
}
```

## Way-2

Formal parameters as a sized array as follows:

```
void myFunction(int param[10])
{
.
.
.
}
```

## Way-3

Formal parameters as an unsized array as follows:

```
void myFunction(int param[])
{
.
.
.
}
```

## Example

Now, consider the following function, which will take an array as an argument along with another argument and based on the passed arguments, it will return average of the numbers passed through the array as follows:

```
double getAverage(int arr[], int size)
{
    int i;
    double avg;
    double sum;

    for (i = 0; i < size; ++i)
    {
        sum += arr[i];
    }

    avg = sum / size;

    return avg;
}
```

Now, let us call the above function as follows:

```
#include <stdio.h>

/* function declaration */
double getAverage(int arr[], int size);

int main ()
{
    /* an int array with 5 elements */
    int balance[5] = {1000, 2, 3, 17, 50};
    double avg;

    /* pass pointer to the array as an argument */
    avg = getAverage( balance, 5 ) ;

    /* output the returned value */
    printf( "Average value is: %f ", avg );

    return 0;
}
```

When the above code is compiled together and executed, it produces the following result:

```
Average value is: 214.400000
```

As you can see, the length of the array doesn't matter as far as the function is concerned because C performs no bounds checking for the formal parameters.