[How to pass the number of elements of an array to a function](https://stackoverflow.com/questions/55551722/how-to-pass-the-number-of-elements-of-an-array-to-a-function)

I would like to pass a n number of elements part of an array to a function in order to calculate the average. Essentialy i would like to keep the number of elements dynamic in the code so that a custom number can be entered.

float average(float num[]);

int main()

{

int n,i,k;

float num[n];

printf("Enter the numbers of elements: ");

scanf("%d",&k);

for(i = 0; i < k; ++i)

{

printf("%d. value: ", i+1);

scanf("%f", &num[i]);

}

printf("Average = %.2lf",average(num));

return 0;

}

float average(float num[])

{

int i,n;

float sum = 0.0, avg;

n = sizeof(num)/sizeof(int);

for(i = 0; i < n; ++i)

{

sum += num[i];

}

avg = sum / n;

return avg;

}

the n = sizeof(num)/sizeof(int); is somehow not passing the correct number of elements. I tried to dig around on the web and tried different options but nothing seem work correctly. I guess this is because I am not passing the array to a function correctly.. but do not know how, please advise, many thanks

**Ans:**

In *main*

float num[n];

must be moved under because *n* is not initialized, so :

int n,i,k;

float num[n];

printf("Enter the numbers of elements: ");

scanf("%d",&k);

must be like

int i,k;

printf("Enter the numbers of elements: ");

if (scanf("%d",&k) != 1) {

puts("invalid size");

return -1;

}

float num[k];

In *average* :

n = sizeof(num)/sizeof(int);

is wrong because the number of elements in *num* is unknown, so sizeof(num) values the size of a pointer, you need to give the number of element in parameter

float average(float num[], int n);

and in *main*

printf("Average = %.2lf",average(num, k));

and

float average(float num[], int n)

{

int i;

float sum = 0.0;

for(i = 0; i < n; ++i)

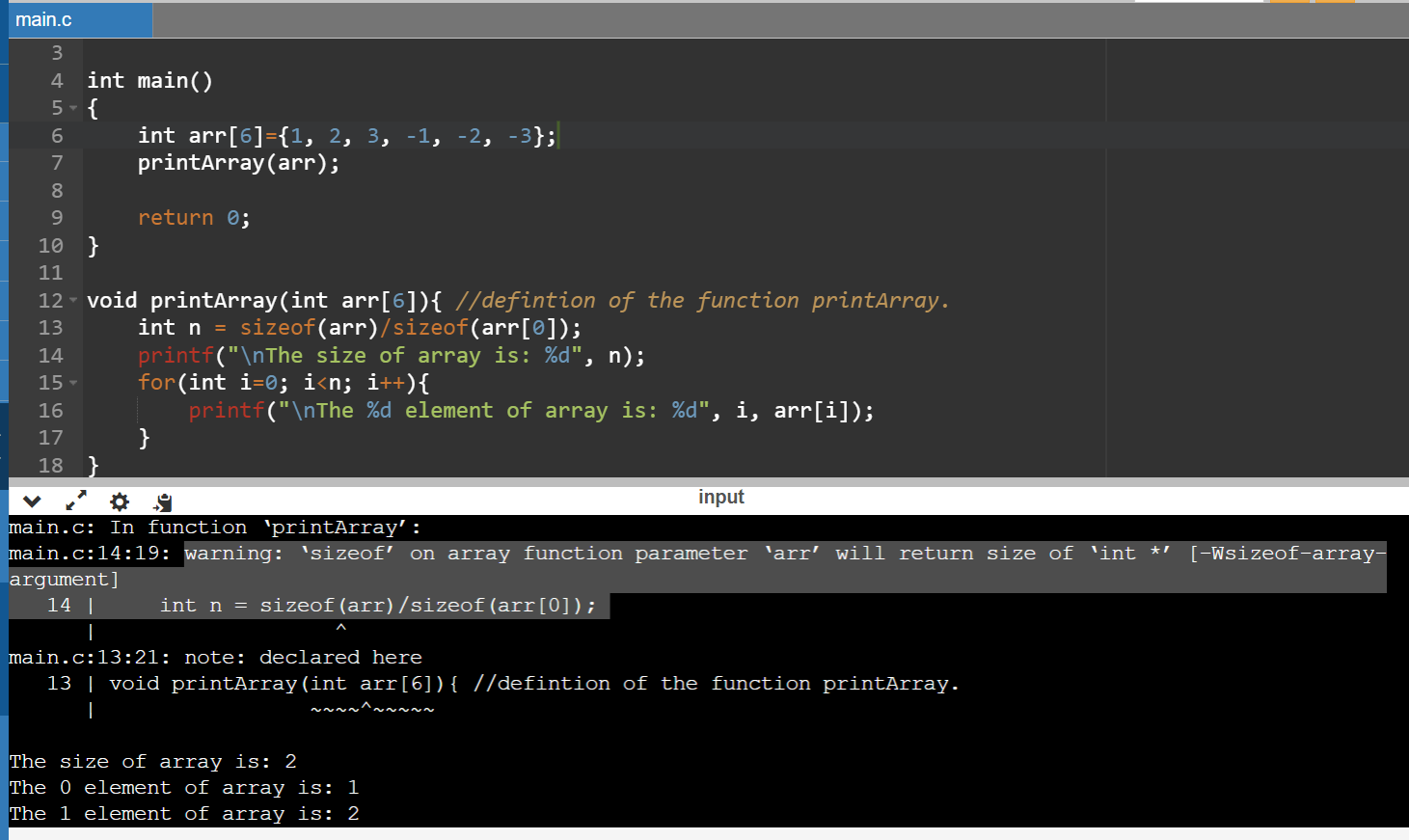
{

sum += num[i];

}

return sum / n;

}



##### What does the following declaration mean? int (\*ptr)[10];

In this example, ptr is defined as a pointer to an array of 10 elements, where each element of the array is of type int. No array is created and no int is created, just a variable named ptr whose data type is pointer to a 10-element array of ints. ptr is not initialized to actually point to anything yet.   
  
If this pointer variable definition appears outside of any function, it is guaranteed to be initialized to NULL. If it appears inside of any function/block, it will contain whatever happens to be in memory (random garbage). Again, there is no array to point to yet.  
  
This is definitely not the same as int \*ptr[10] (which is an array of 10 integers) and is definitely not a "pointer to a pointer."