Lets take example first.

#include<stdio.h>

int main(){

int x[5];

int y;

y = &x[0];

y = x;

printf("%lu\n", sizeof(x));

printf("%lu\n", sizeof(&x[0]));

return 0;

}

/\*

eg85.c:5:7: error: incompatible pointer to integer conversion assigning to 'int' from 'int \*'; remove & [-Wint-conversion]

y = &x[0];

^ ~~~~~

eg85.c:6:7: error: incompatible pointer to integer conversion assigning to 'int' from 'int[5]' [-Wint-conversion]

y = x;

^ ~

2 errors generated.

\*/

We can see

y = &x[0];

y = x;

In these two lines we are getting the same error. While

printf("%lu\n", sizeof(x));

printf("%lu\n", sizeof(&x[0]));

These two line compiled successfully.

Int x[5]; x is an identifier,x represents a collection of 5 inputs

← – –– –– –– –– –– –– –– –– –– –– –– X –– –– –– –– –– –– –– –– –– –– –– –– –– –– – →

700 (20 Bytes)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |

X[0] x[1] x[2] x[3] x[4]

Type of x[0] is int

Type of &x[0] is int \*

When we rite identifier x other than sizeof function then it will be considered as

&x[0];

Means

Y = x;

Will be converted into

Y = &x[0];

But when we will write sizeof(x) then it will not change it will represent the int \*[5].

#include<stdio.h>

int main(){

int x[5], y;

int \*p;

p=x; // shoule be treated as p=&x[0];

\*p=100;

p++;

\*p=200;

p++;

\*p=300;

p++;

\*p=400;

p++;

\*p=500;

for(y=0;y<=4;y++){

printf("%d\n", x[y]);

}

return 0;

}

/\*

100

200

300

400

500\*/

How to pass one d array to a function.

#include<stdio.h>

int sum(int \*p, int sz){

int y,t;

for(y=0,t=0;y<sz;y++,p++){

t=t+\*p;

}

return t;

}

int main(){

int x[5];

int y, total;

for(y=0;y<5;y++){

printf("Enter a number: ");

scanf("%d", &x[y]);

}

// total = sum(x, 5);

total = sum(x, sizeof(x) / sizeof(x[0]));

printf("Total is %d\n", total);

return 0;

}

Technically we are not passing any array

Technically address of 1st element of the array will be passed.