Pointers – These are special type of variables which hold the address of other variables from memory.

Consider the following declaration

int i = 5;

i name of variable in memory

5

Value stored in var i

600 Address of var i in memory

Features of memory address

1. It is unique number (never duplicated)
2. It is assigned by the compiler auto
3. It is +ve integer (unsigned int) with no decimals

To make use of these memory addresses, they need to be stored in another variable, which is called as pointer / pointer variable.

int i = 5;

~~unsigned int j;~~ // error, though pointers contain unsigned int value, they r not declared

// like this. They are normally declared depending upon the value

// type to which they are pointing/referring

int \*j; // \* is “value at address” operator. Meaning is “value to which j is pointing is an

// integer

i j

5

j = &i;

600

600/1/2/3 764

Now you may feel that as pointers are unsigned integers, what is the harm in

declaring unsigned int j? Right? Now the reason is that using pointers, if you

decide to access the value, then how is the compiler to know, how many

memory blocks it has to read? Hence if we say

int \*j, the compiler will read 4 memory blocks bcoz size of int is 4 bytes

char \*j, the compiler will read 1 memory block bcoz size of char is 1 byte

double \*j, the compiler will read 8 blocks bcoz size of double is 8 bytes

float pi = 3.14;

~~unsigned int ptr\_pi;~~ // error

float \*ptr\_pi; // We are allocating memory to a pointer var. These vars are

// declared depending upon the value to which they

// are pointing

ptr\_pi = &pi;

pi ptr\_pi

3.14

1024

1024 9537

Similarly if the pointer points to a char var, it’s declaration will be

char \*ptr\_ch;

This doesn’t mean ptr\_ch is containing a char value but “value to which

ptr\_ch is pointing is a char value.

Note that pointers are used only between function calls wherein the variable of calling function needs to be referred in called function.

Call By Value Vs Call By Reference/Address

main()

a b a b

20

10

~~20~~ 25

~~10~~ 13

600 700 600 700

incr() a b ptr\_a ptr\_b

700

600

~~20~~ 25

~~10~~ 13

800 900

\*ptr\_a += 3;

\*ptr\_b += 5;

Addresses are unique &

are not controlled by function

scopes.