Pointer as a function parameter is used to hold addresses of arguments passed during function call. This is also known as **call by reference**.

When a function is called by reference any change made to the reference variable will effect the original variable.

**Example: Swapping two numbers using Pointer**

#include <stdio.h>

void swap(int \*p1, int \*p2){

//value exchanging using dereferencing pointer

//int temp = \*p1;

//\*p1 = \*p2;

//\*p2 = temp;

//using third pointer

int \*\*temp = p1;

p1 = p2;

p2 = temp;

printf("after swapping a is %d and b is %d\n",\*p1,\*p2);

}

int main ()

{

int a = 5, b = 10;

int \*p1 = &a, \*p2 = &b;

swap(p1,p2);

printf("after swapping a is %d and b is %d (main)\n",a,b);

return 0;

}

### **Functions returning Pointer variables**

A function can also return a pointer to the calling function. In this case you must be careful, because local variables of function doesn't live outside the function. They have scope only inside the function. Hence if you return a pointer connected to a local variable, that pointer will be pointing to nothing when the function ends.

**Ex: find the largest number**

#include <stdio.h>

int\* larger(int\*, int\*);

void main()

{

int a = 15;

int b = 92;

int \*p;

p = larger(&a, &b);

printf("%d is larger",\*p);

}

int\* larger(int \*x, int \*y)

{

if(\*x > \*y)

return x;

else

return y;

}

**Output** : 92 is larger