UNIT-2 (POINTERS) (RAM GOPAL GUPTA- http://ramgopalgupta.com/)

DEFINITION, DECLARATION AND INITIALIZATION

Basic Concept

Before starting the topic pointer, we should understand the concept of address in memory. If we have a variable name "var" in our program &var will give us the memory address of that variable.

Pointers are special variable that are used to store address of another variable rather than values.

Declaration of Pointer syntax:

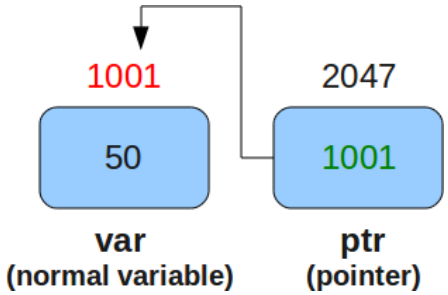
int *ptr;

Here we declare a pointer 'ptr' of type 'int'.

Initialization of Pointer syntax:

1. Declare a Pointer Variable and Note down the Data Type.
2. Declare another Variable with Same Data Type as that of Pointer Variable.
3. Now initialize pointer by assigning the address of ordinary variable to pointer variable.

```
main()
{
int *ptr;              // Step 1
int var;               // Step 2
ptr = &var;            // Step 3
var = 50;              // Step 4
printf("%d",*ptr);     // Step 5 (output is 50)
}
```



Address-of Operator (&) this operator returns the address of a variable as shown in ***step 3 above***.

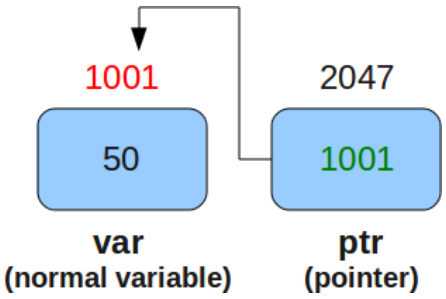
Indirection operator (*) is a unary operator represented by the symbol (*). It is an operator used to obtain the value of a variable to which a pointer points. While a pointer pointing to a variable provides an indirect access to the value of the variable stored in its memory address, the indirection operator dereferences the pointer and returns the value of the variable at that memory location as shown in ***step 5 above.***

Program code example:

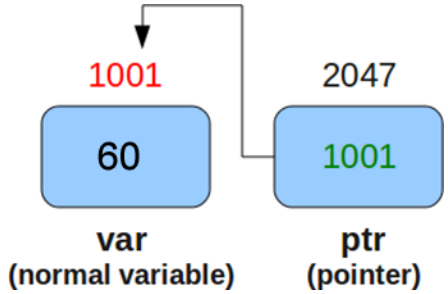
```
main()
{
int *ptr;                  // Step 1
int var;                   // Step 2
ptr = &var;                // Step 3
var = 50;                  // Step 4
printf ("%d \n",*ptr);  // Step 5 (output is 50)
var = var + 10;            // Step 6 (var = 60)
printf ("%d \n", *ptr); // Step 7 (output is 60)
*ptr = *ptr +10;           // Step 8 (var = 70)
printf ("%d \n", *ptr); // Step 9 (output is 70)
printf ("%d \n", var);  // Step 10 (output is 70)
}
```

Explanation of the above program:

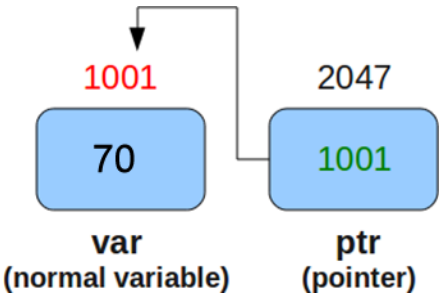
- When step 1, 2, 3 and 4 executed the situation was represented like below:



- When step 5 executed *ptr will display "50" because pointer 'ptr' holds the address of variable 'var' therefore *ptr will display the value from address it holds that is '1001'.
- When step 6 executed the situation was like below:



- When step 7 executed *ptr now will display "60" because pointer 'ptr' holds the address of variable 'var' therefore *ptr will display the value from address it holds that is '1001'.
- When step 8 executed then it calculated like this
  *ptr = *ptr + 10;
  *ptr = 60 + 10;
  *ptr = 70;
  As we know *ptr is accessing the memory location '1001 therefore:
  *ptr =70 will store value 70 into memory location 1001. It means var = 70; pictorially represented as below:



- When step 9 and 10 executed they will print ‘70’ separately.