SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

COURSE NAME : 23ITT101- PROBLEM SOLVING & C PROGRAMMING

I YEAR /I SEMESTER

Unit IV – FUNCTIONS AND POINTERS

Topic : POINTERS

Topics Covered

- **Pointers:**
  - » **Introduction**
  - » **Understanding pointers**
  - » **Accessing the address of a variable**
  - » **Declaring pointer variables**

Pointers - Introduction

- Pointer is a **derived data type** in C. It is built from one of the fundamental data types available in C.
- Pointers **contains memory address as their values.**
- Since these **memory addresses** are the **locations in the computer memory** where program instructions and data are stored, pointers can be **used to access and manipulate data stored in the memory**.

Pointers - Introduction

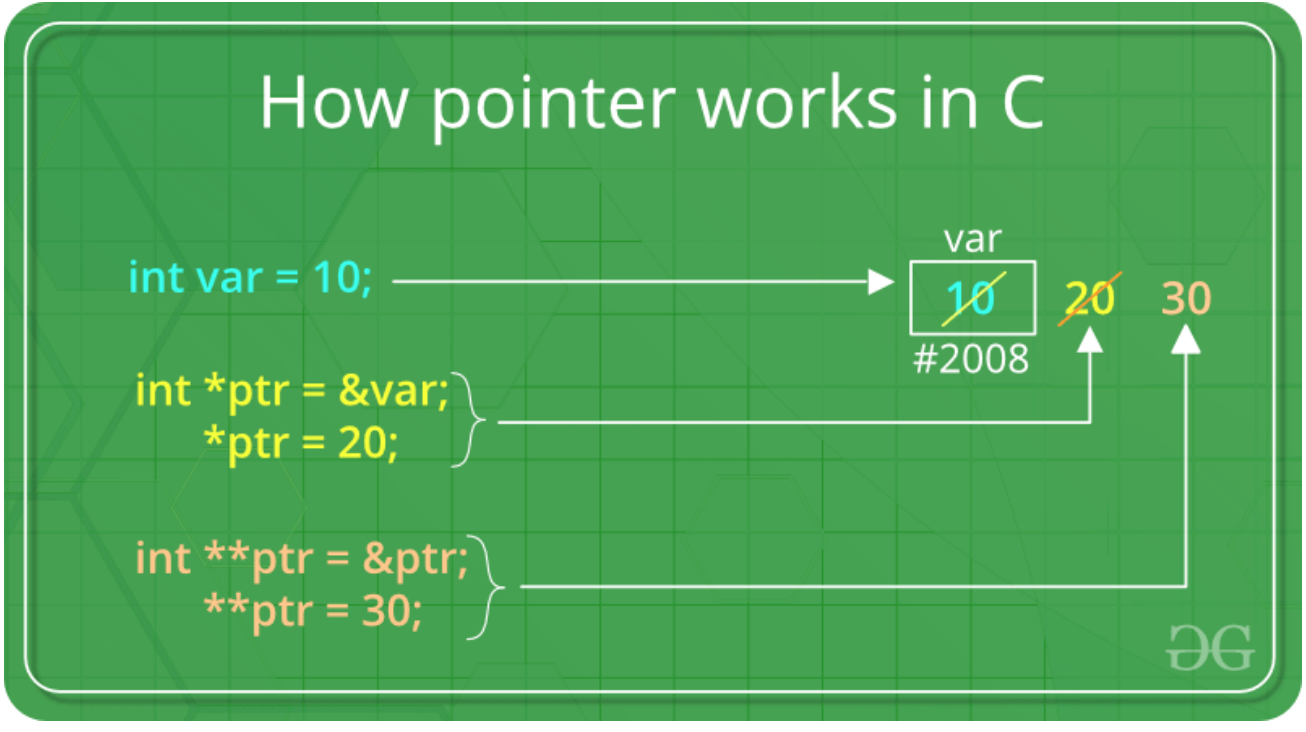
- **Variables that hold memory addresses** are called **pointers**.
- Pointer variables are **declared using asterisk (*).**

Benefits of pointers include the following,

1. Pointers are more **efficient in handling arrays and data tables.**
2. Pointers can be **used to return multiple values** from a function via function arguments.
3. Pointers **permit references to function**.
4. The use of **pointer arrays to character strings** results in **saving data storage space in memory**.

Pointers - Introduction

5. Pointer allow C to **support dynamic memory management.**
6. Provide an **efficient tool for manipulating dynamic data structures**(Structures, Linked list, queues, linked list, trees).
7. Pointers **reduce length and complexity of programs.**
8. They **increase execution speed** and **reduces program execution time.**


How pointer works in C
int var = 10;
var
10
20
30
#2008
int *ptr = &var;
*ptr = 20;
int **ptr = &ptr;
**ptr = 30;

Understanding Pointers

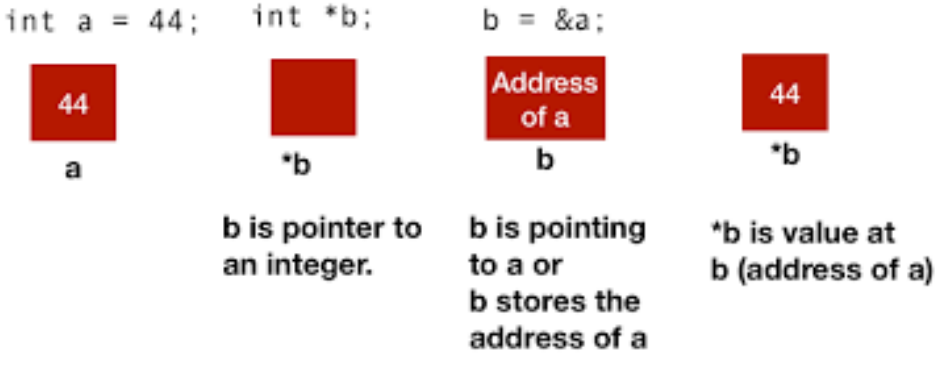
- Memory organization
- Representation of a pointer variable
- Concepts of pointers

Memory Organization:

- Computer's memory is a **sequential collection of storage cells**, each cell commonly known as a **byte**, **has a number called address** associated

Understanding Pointers

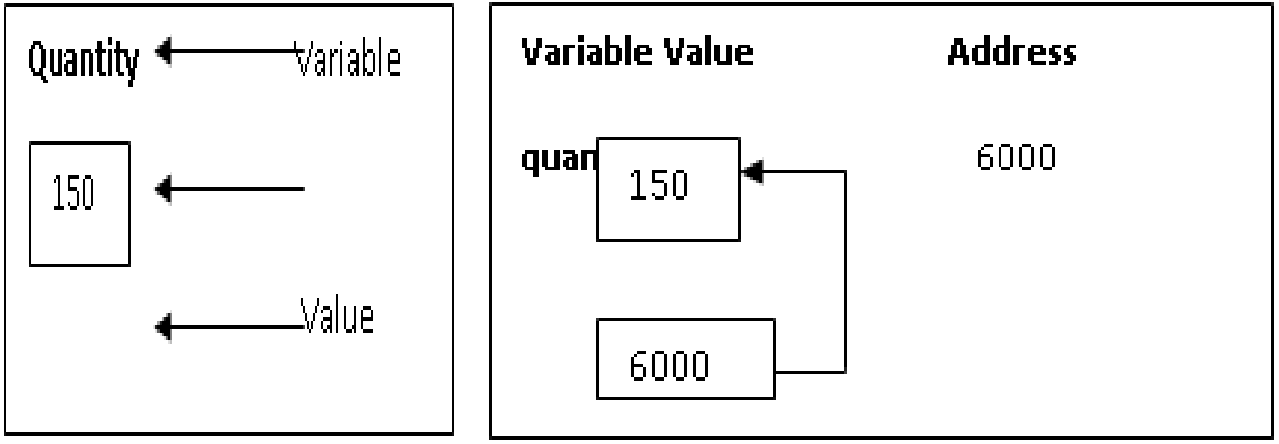
- Whenever we declare a variable, the **system allocates, appropriate location to hold the value of the variable**.
- Consider the following statement.
- int a = 44;



Understanding Pointers

- **During execution** of the program, the **system always associates the name quantity with the address 5000(Example).**
- To **access** the value 44 we **use either the name quantity or the address 5000**.
- Since **memory addresses** are **simply numbers**.
- Such **variables** that **hold memory addresses are called pointers.**

Understanding Pointers



Understanding Pointers

- p holds the address of quantity.
- We can also access the value of quantity by using p.
- We can say that p points to the variable quantity.
- Thus p gets the name **pointer.**

Understanding Pointers

- **Pointer Constant : Memory address within a computer**.
- **Pointer Value :** we can obtain the memory address by **using & address operator**. The value obtained is called pointer value.
- **Pointer Variable :** The **variable that contains a pointer value** is called pointer variable.

Understanding Pointers

```
#include<stdio.h>
void main( )
{
    char m;  int n;        float a, b;
    m = ‘A’;
    n = 150;
    a = 15.25, b = 10.75;
    printf(“%c is stored at addr %u.\n”, m, &m);
    printf(“%d is stored at addr %u.\n”, n, &n);
    printf(“%f is stored at addr %u.\n”, a, &a);
          printf(“%f is stored at addr %u.\n”, b, &b);
}
```

Accessing the address of a variable

- The **actual location of a variable** in the memory is **system dependent.**
- The address of a variable is not known to us immediately.
- However we **determine the address of a variable** by **using the operand &** available in C.

Accessing the address of a variable

Example:

```
p = &quantity;
```

- would assign the address 5000 (the location of quantity) to the variable p.
- The & operator can be remembered as ‘address of’.

Accessing the address of a variable

- The & operator can be only used with a simple variable or an array element.

The following are illegal use of address operator:

1. &125 (pointing at constants).
2. int x[10];

   &x (pointing at array names).
3. &(x+y) (pointing at expressions).

Accessing the address of a variable

- If x is an array ,then expressions such as,

  &x[0] and &x[i + 3]

  are valid and represent the addresses of 0th and (i+3)th elements of x.

- The program shown below declares and initializes four variables and then prints out these values with their respective storage locations.

Accessing the address of a variable

```
main( )
{
char a; int x; float p, q;
a  = 'A'; x = 125;
printf("A is stored at addr %u . \n", &a);
printf("125  is stored at addr %u . \n", &x);
}
```

Output:

A is stored at addr 44336.

125 is stored at addr 4434.

DECLARING POINTER VARIABLES

- A **pointer** is a variable whose **value is the address of another variable.**
- Pointer can be **used to access both address and values of a variable**.
- It is used to provide direct address of the memory location.

DECLARING POINTER VARIABLES

Data_type *pt_namae;

- This tells the compiler three things about the variable pt _name:

1. The asterisk(*) tells that the variable pt _name is a pointer variable.
2. pt _name needs a memory location.
3. data type to be identified.

DECLARING POINTER VARIABLES

Example:

```
int *p;
float *x;
int*  p;
int  *p;
int *  p;
int* x;
float *p;
char * x;
```

Try it Yourself

What is the output of this C code?

```
int main()
{
int i = 10;
void *p = &i;
printf("%d\n", (int)*p);
return 0;
}
```

```
int main()
{
int i = 10;
void *p = &i;
printf("%f\n", *(float*)p);
return 0;
}
```

```
int main()
{
char *p = NULL;
char *q = 0;
if (p)
printf(" p ");
else
printf("nullp");
if (q)
printf("q\n");
else
printf(" nullq\n");
}
```

Summary

- Pointer is a derived data type and important feature in C.
- Computer's memory is a sequential collection of storage cells, each cell commonly known as a byte, has a number called address associated with it.
- Pointers support dynamic memory management
    - Pointer constants - only we can use them to store data values
    - Pointer values may change from one run of the program to another.
- The variable that contains a pointer value called pointer variable.


Thank
you