Dr.SNS RAJALAKSHMI COLLEGE OF ARTS AND SCIENCE

(AUTONOMOUS)
COIMBATORE-641049
Accredited by NAAC(Cycle III) with "A+" Grade
Recognised by UGC, Approved by AICTE, New Delhi and
Affiliated to Bharathiar University, Coimbatore.

DEPARTMENT OF COMPUTER SCIENCE

Programming in C

UNIT 3 – INTRODUCTION TO POINTERS

TOPIC: Pointers

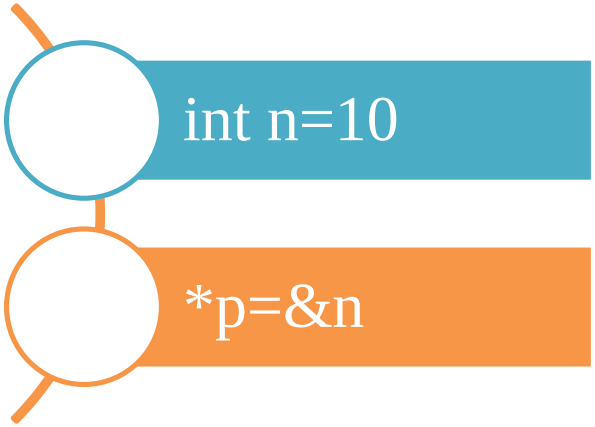
Outline

- ✓Pointers
- ✓How to Declare Pointers
- ✓Back to function call
- ✓Recursion and stack
- ✓Recursion Example

Pointers

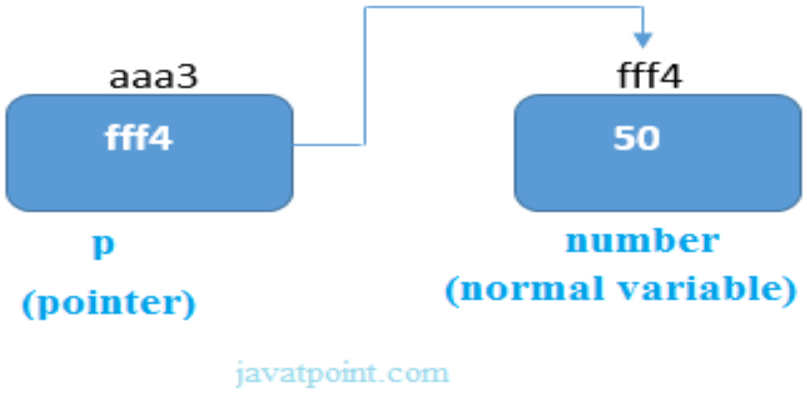
- The pointer in C language is a variable which stores the address of another variable.
- This variable can be of type int, char, array, function, or any other pointer.
- The size of the pointer depends on the architecture.

Define a pointer



Declare a pointer

int *a;//pointer to int
char *c;//pointer to char



Example

```
1.#include<stdio.h>
2.void main(){
3.int n=50;
4.int *p;
5.p=&n;
6.printf("Address of p variable is %x \n",p);

7.printf("Value of p variable is %d \n",*p);
8.getch();
9.}
```

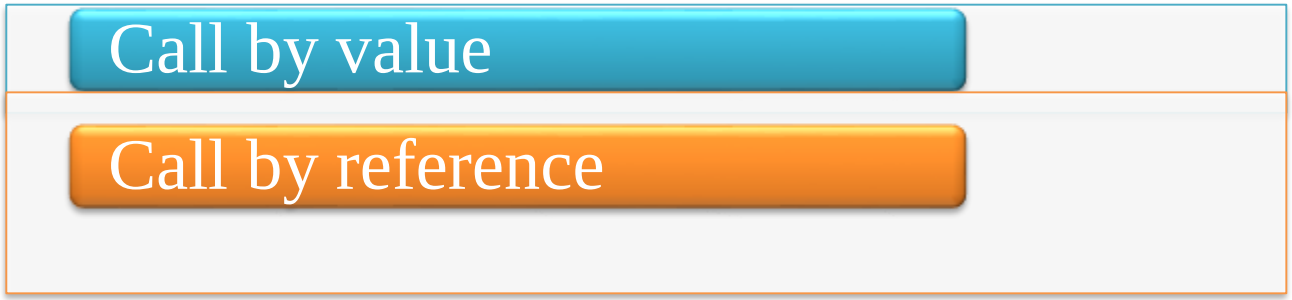
OUTPUT

Address of p variable is fff4

Value of p variable is 50

Back to function call

- Function can be call in two ways:



Call by value

```
#include <stdio.h>
void swap(int x, int y);
void main () {
   /* local variable definition */
   int a = 100;
   int b = 200;
printf("Before swapping, a : %d\n b:%d\n,a ,b);
  swap(a, b);
  printf("After swapping, value of a : %d\nb : %d\n", a,b);
  }
void swap(int x, int y) {
 int temp;
temp = x;
   x = y;
   y = temp;
}
```

Call by Reference

```
#include <stdio.h>
void main () {
  int a = 100;
  int b = 200;
 printf("Before swap, value of a : %d\nb : %d\n", a,b );
 swap(&a, &b);
 printf("After swap, value of a : %d\nb : %d\n", a,b);
   }
void swap(int *x, int *y) {
  int temp;
temp = *x;
 *x = *y;
 *y = temp;
}
```

Recursion

- ✓ Recursion is the process of repeating items in a self-similar way.
- ✓ In programming languages, if a program allows you to call a function inside the same function, then it is called a recursive call of the function.

```
void recursion()
{
recursion(); /* function calls itself */
}
void main()
 {
recursion();
 }
```

Recursion

EXAMPLE:

```
#include <stdio.h>
int factorial(int i)
{
 if(i == 0)
{ return 1; }
else
{
return i * factorial(i - 1);
}
}
void main()
{
int i = 6;
printf("Factorial of %d is %d\n", i, factorial(i));
getch();
}
```

Recursion

Recursion and Stack :

- ❑ Recursion means function call by itself
- ❑ It uses the internal stack and stores the activation record
- ❑ Activation record stores the details like local data , parameters, return value etc…

Structure:

- ♣ It has Base case & Progressive case
- ♣ It avoids infinite running of function

Basecase:

- ☺ There are must be atleast 1 basecase or condition
- ☺ When the base condition is met the recursion stops

Progressive Case:

- ✓ Function call should be progressively move to base case

Structure of Recursive Funtion

```
Function_name(parameter)
{
   if(condition)
      return 1;
   else
      recursive call (parameters tovwards base cond);
}

void main()
{
   int …;
   Function_name(parameter);
}
```

Base Case

Progressive Case

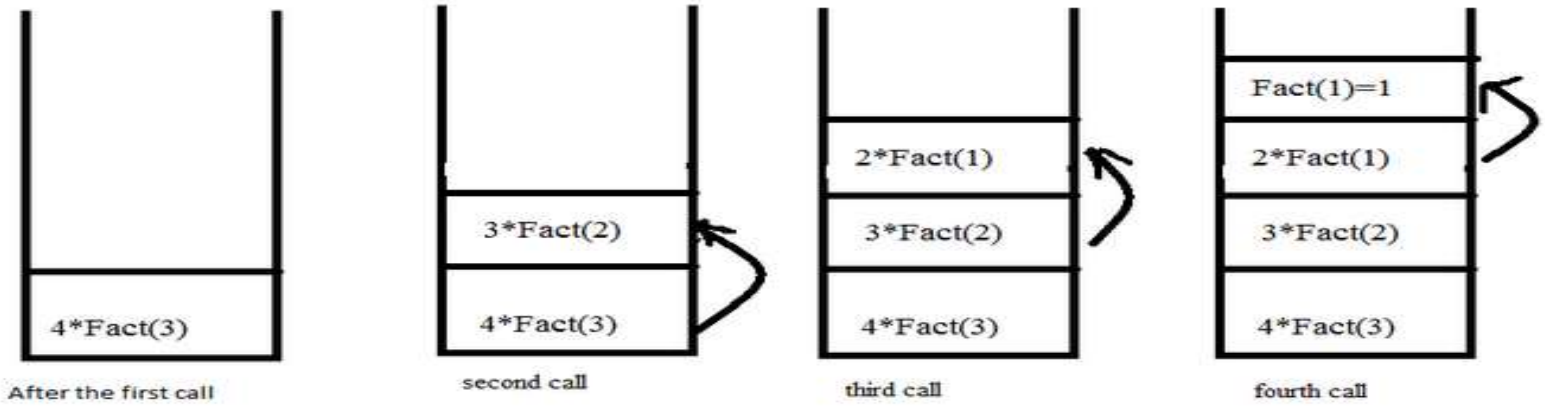
Initial Recursive Call

- A stack is **a linear data structure that follows the**

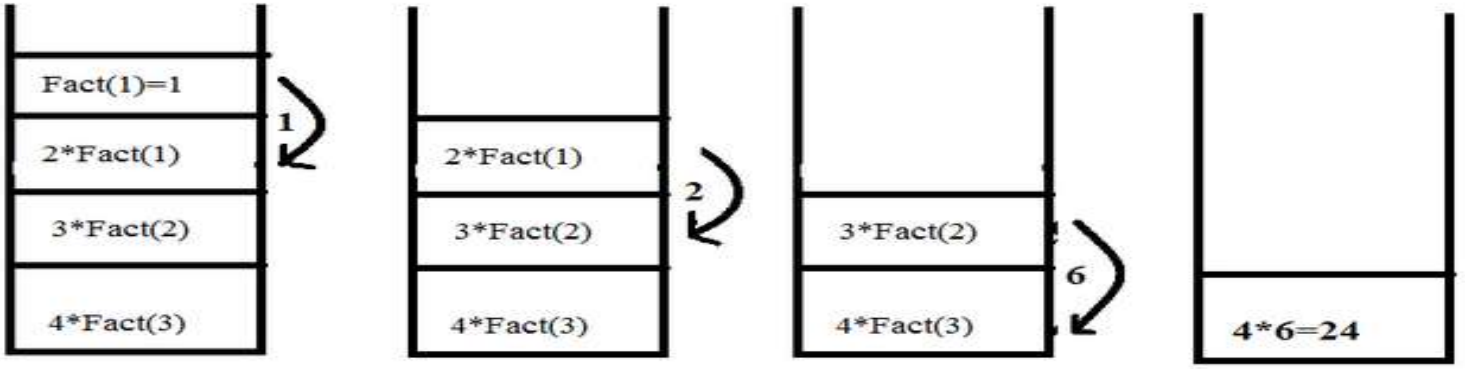
Last in, First out principle

- By using Stack :structure of Recursion progress

When function call happens previous variables gets stored in stack



Returning values from base case to caller function



QUERIES ???

(if any..)

17

Colourful Thank You Slide Design

THANK YOU