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 : Call by Value and Call by Reference

Topics Covered

- Function call
  - Call by value
  - Call by reference

Calling Functions: Call by Value and Call by Reference

- Used when invoking functions
- Call by value
  - Copy of argument passed to function
  - Changes in function do not effect original
  - Use when function does not need to modify argument
    - Avoids accidental changes
- Call by reference
  - Passes original argument
  - Changes in function effect original
  - Only used with trusted functions
- By default calling functions use **call by value**

Calling Functions: Call by Value

```
#include <stdio.h>
int main()
{
 int x=100;
printf("Before function call x=%d \n",
x);
 change(x);//passing value in function
 printf("After function call x=%d \n",
x);
return 0;
}
```

```
void change(int num)
{
printf("Before adding value inside
function num=%d \n",num);
    num=num+100;
    printf("After adding value inside
function num=%d \n", num);
}
```

```
Before function call x=100
Before adding value inside function num=100
After adding value inside function num=200
After function call x=100
```

Calling Functions: Call by Reference

```
#include <stdio.h>
int main()
{
 int x=100;
printf("Before function call x=%d \n",
x);
 change(&x);//passing value in
function
 printf("After function call x=%d \n",
x);
return 0;
}
```

```
void change(int *num)
{
printf("Before adding value inside
function num=%d \n",*num);
    (*num)+ =100;
    printf("After adding value inside
function num=%d \n", *num);
}
```

```
Before function call x=100
Before adding value inside function num=100
After adding value inside function num=200
After function call x=200
```

Call by Value – Number swap

```
#include <stdio.h>
// Function prototype
void swapByValue(int x, int y);       // Call by value

int main() {
int a = 10, b = 20;

printf("Before swapping:\n");
printf("a = %d, b = %d\n", a, b);

// Swap using Call by Value
swapByValue(a, b);
printf("\nAfter swapByValue (no change in actual values):\n");
printf("a = %d, b = %d\n", a, b);
return 0;
}
```

```
// Function to swap using Call by Value
void swapByValue(int x, int y) {
int temp = x;
x = y;
y = temp;

printf("\nInside swapByValue:\n");
printf("x = %d, y = %d\n", x, y);
//This change won't affect `a` and `b` in main.
}
```

Call by Value – Number swap

```
#include <stdio.h>
// Function prototypes
void swapByReference(int *x, int *y);
// Call by reference

int main() {
int a = 10, b = 20;

printf("Before swapping:\n");
printf("a = %d, b = %d\n", a, b);

// Swap using Call by Reference
swapByReference(&a, &b);
printf("\nAfter swapByReference (actual values swapped):\n");
printf("a = %d, b = %d\n", a, b);
return 0;
}
```

```
// Function to swap using Call by Reference
void swapByReference(int *x, int *y) {
    int temp = *x;
    *x = *y;
    *y = temp;

    printf("\nInside swapByReference:\n");
   printf("*x = %d, *y = %d\n", *x, *y);
// This change affects `a` and `b` in main.
}
```

Output – Number Swap

Before swapping:
a = 10, b = 20

Inside swapByValue:
x = 20, y = 10

After swapByValue (no change in actual values):
a = 10, b = 20

Inside swapByReference:
*x = 20, *y = 10

After swapByReference (actual values swapped):
a = 20, b = 10

Summary

- A function call is provided with function name and arguments (data) in the calling part of the program. Two types of calling functions include call-by-value and call-by-reference.


Thank
you