CS F111: Computer Programming

(Second Semester 2020-21)

Lect 15: Functions-intro

Dr. Nikumani Choudhury
Asst. Prof., Dept. of Computer Sc. & Information Systems
nikumani@hyderabad.bits-pilani.ac.in

BITS Pilani

Hyderabad Campus

Modular Programing: User Defined Functions

```c
#include<stdio.h>
void greater(void);

int main(){
   greater();
return 0;
}
void greater(void) {
 int i, j;

 printf("Enter 2 numbers”);
 scanf("%d%d", &i, &j);

 if(i > j)
   printf("The greater number is: %d", i);
 else
   printf("The greater number is: %d", j);
}
```

(No arguments and No return value)

```c
#include<stdio.h>
int greater();

int main(){
 int result;
 result = greater();
 printf ("The greater no: %d",
result);
return 0;
}
int greater() {
 int i, j, k;
 printf("Enter 2 numbers");
 scanf("%d%d", &i, &j);
 if(i > j)
  k = i;
 else
  k = j;
return k;
}
```

(No arguments and a return value)

Modular Programing: User Defined Functions

```c
#include<stdio.h>
int greater();
int main(){
 int i,j;
 scanf("%d %d", &i, &j);
 greater(i,j);
 return 0;
}

int greater(int x, int y){
 if (x > y)
   printf("Greater no: %d", x);
 else
   printf("Greater no:%d", y);
}
```

(With arguments and no return value)

```c
#include <stdio.h>
int add_numbers(int a, int b);
int main(){
  int n1,n2,sum;
  printf("Enter two numbers: ");
  scanf("%d %d",&n1,&n2);
  sum = add_numbers(n1, n2);
  printf("sum = %d",sum);
  return 0;
}
int add_numbers(int a, int b){
  int result;
  result = a+b;
  return result;
}
```

(With arguments and a return value)

Parameter Passing in C

- A Parameter is the symbolic name for "data" that goes into a function.
- Call-by-Value, means that a copy of the data is made and stored by way of the name of the parameter. Any changes to the parameter have NO affect on data in the calling function.
- There are two ways for Call-by-Reference parameter:
- Arrays
  - Arrays are always passed by reference in C. Any change made to the parameter containing the array will change the value of the original array.
- The ampersand used in the function prototype.
  - function (&parameter_name)

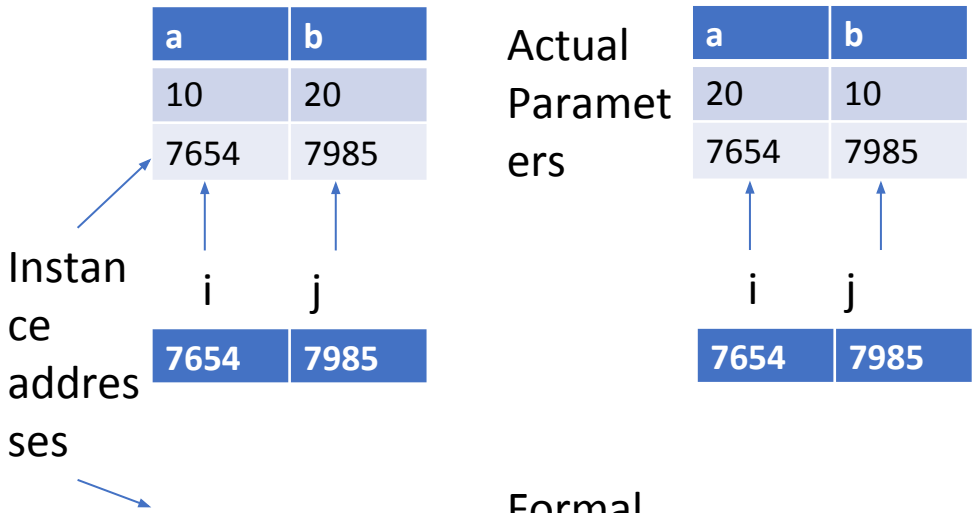
Call-by-Value: Another Example

```
#include <stdio.h>
void swap (int x, int y);
int main (void) {
int x =10, y=50, temp=0;
swap(x,y);
printf ("In main: %d %d %d\n", x, y, temp);
return 0;
}
void swap (int x, int y) {
int temp;
temp = x;
x = y;
y = temp;
printf ("In swap: %d %d %d\n", x, y, temp);
return; }
```

```
In swap: 50 10 10
In main: 10 50 0
```

Is it a successful swap?

Call-by-Reference




```c
#include <stdio.h>
void swapnum(int *i, int *j)
{
  int temp;
  temp = *i;
  *i = *j;
  *j = temp;
}
int main(void) {
  int a = 10, b = 20;
  swapnum(&a, &b);
  printf("%d %d\n", a, b);
return 0;
}
```