



RAMA UNIVERSITY

www.ramauniversity.ac.in

FACULTY OF ENGINEERING & TECHNOLOGY

CSPS103: Object Oriented Programming

Lecture- 10

Preeti Singh

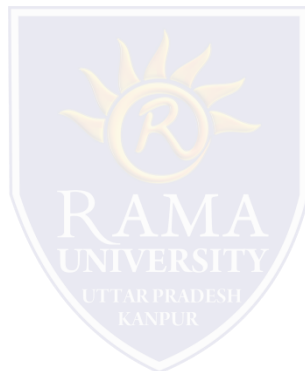
Department of Computer Science & Engineering
Rama University, Kanpur

preeti.ru@ramauniversity.ac.in

OBJECTIVES

In this lecture, you will learn to:

- ❖ **Inline Functions**
- ❖ **Call by value in C++**
- ❖ **Call by reference in C++**
- ❖ **Default arguments**



INLINE FUNCTIONS

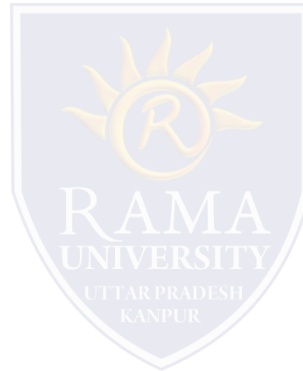
- ❑ An inline function is a function that is expanded inline at the point at which it is invoked, instead of actually being called.
- ❑ The reason that inline functions are an important addition to C++ is that they allow you to create very efficient code.
- ❑ Each time a normal function is called, a significant amount of overhead is generated by the calling and return mechanism.
- ❑ Although expanding function calls in line can produce faster run times, it can also result in larger code size because of duplicated code.
- ❑ A function can be defined as an inline function by prefixing the keyword `inline` to the function header as given below:

```
inline function header {  
function body  
}
```

PROGRAM : INLINE FUNCTION

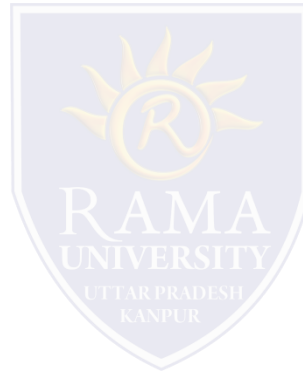
// A program illustrating inline function

```
#include<iostream.h>
#include<conio.h>
inline int max(int x, int y){
if(x>y)
return x;
else
return y;
}
int main( ) {
int a,b;
cout<<"enter two numbers";
cin>>a>>b;
cout << "The max is: " <<max(a,,b) << endl;
return 0;
}
```



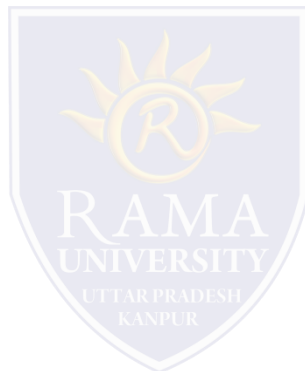
CALL BY VALUE IN C++

- In call by value, original value is not modified.
- In call by value, value being passed to the function is locally stored by the function parameter in stack memory location.
- If you change the value of function parameter, it is changed for the current function only.
- It will not change the value of variable inside the caller method such as main().



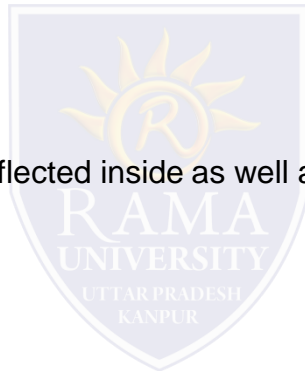
EXAMPLE : CALL BY VALUE IN C++

```
#include <iostream.h>
void change(int data);
int main()
{
int data = 3;
change(data);
cout << "Value of the data is: " << data<< endl;
return 0;
}
void change(int data)
{
data = 5;
}
```



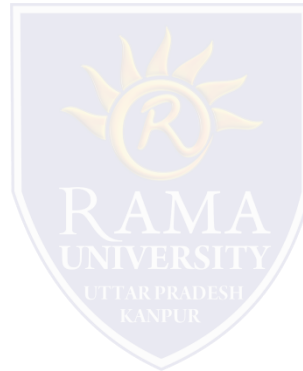
CALL BY REFERENCE IN C++

- ❑ In call by reference, original value is modified because we pass reference (address).
- ❑ Here, address of the value is passed in the function, so actual and formal arguments share the same address space.
- ❑ Hence, value changed inside the function, is reflected inside as well as outside the function.



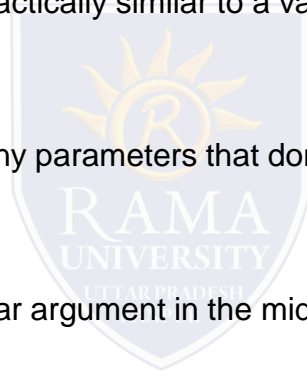
EXAMPLE : CALL BY REFERENCE IN C++

```
#include<iostream.h>
void swap(int *x, int *y)
{
    int swap;
    swap=*x;
    *x=*y;
    *y=swap;
}
int main()
{
    int x=500, y=100;
    swap(&x, &y); // passing value to function
    cout<<"Value of x is: "<<x<<endl;
    cout<<"Value of y is: "<<y<<endl;
    return 0;
}
```



DEFAULT ARGUMENTS

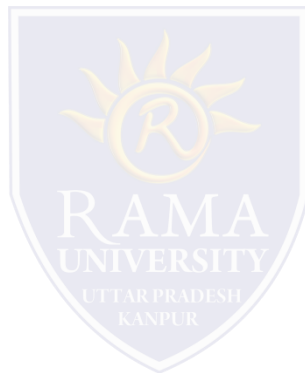
- ❑ C++ allows a function to assign a parameter a default value when no argument corresponding to that parameter is specified in a call to that function.
- ❑ The default value is specified in a manner syntactically similar to a variable initialization.
- ❑ All default parameters must be to the right of any parameters that don't have defaults.
- ❑ We cannot provide a default value to a particular argument in the middle of an argument list.
- ❑ When you create a function that has one or more default arguments, those arguments must be specified only once: either in the function's prototype or in the function's definition if the definition precedes the function's first use.



PROGRAM : DEFAULT ARGUMENTS

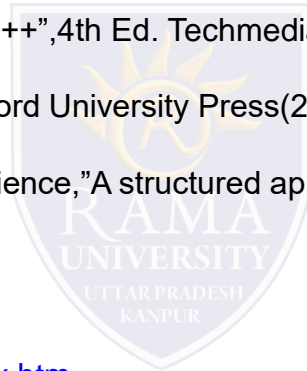
Example

```
#include <iostream.h>
#include<conio.h>
int sum(int a, int b=20){
return( a + b);
}
int main (){
int a = 100, b=200, result;
result = sum(a, b); //here a=100 , b=200
cout << "Total value is :" << result << endl;
result = sum(a); //here a=100 , b=20(using default value)
cout << "Total value is :" << result << endl;
return 0;
}
```



REFERENCES

- Kernighan, Brian W., and Dennis M. Richie. The C Programming Language. Vol. 2. Englewood Cliffs: Prentice-Hall, 1988.
- King, Kim N., and Kim King. C programming: A Modern Approach. Norton, 1996.
- Bjarne Stroustrup, "C++ Programming language", 3rd edition, Pearson education Asia (1997)
- Lafore R. "Object oriented Programming in C++", 4th Ed. Techmedia, New Delhi (2002).
- Yashwant Kenetkar, "Let us C++", 1st Ed., Oxford University Press (2006)
- B.A. Forouzan and R.F. Gilberg, Compiler Science, "A structured approach using C++" Cengage Learning, New Delhi.
- <https://www.javatpoint.com/cpp-tutorial>
- <https://www.tutorialspoint.com/cplusplus/index.htm>
- [https://ambedkarcollegevasai.com/wp-content/uploads/2019/03/ CPP.pdf](https://ambedkarcollegevasai.com/wp-content/uploads/2019/03/_CPP.pdf)
- https://onlinecourses.nptel.ac.in/noc20_cs07/unit?unit=3&lesson=19



MULTIPLE CHOICE QUESTION

Multiple Choice Question:

Q1. How many minimum number of functions should be present in a C++ program for its execution?

- a) 0
- b) 1
- c) 2
- d) 3



MULTIPLE CHOICE QUESTION

Multiple Choice Question:

Q2. Which is more effective while calling the functions?

- a) call by value
- b) call by reference
- c) call by pointer
- d) call by object



MULTIPLE CHOICE QUESTION

Multiple Choice Question:

Q3. How many ways of passing a parameter are there in c++?

- a) 1
- b) 2
- c) 3
- d) 4

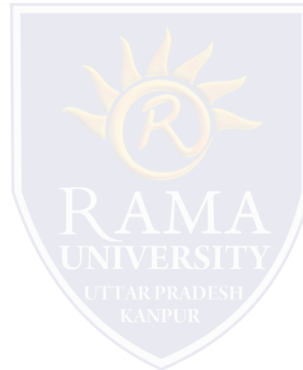


MULTIPLE CHOICE QUESTION

Multiple Choice Question:

Q4. Which of the following is the default return value of functions in C++?

- a) int
- b) char
- c) float
- d) void



MULTIPLE CHOICE QUESTION

Multiple Choice Question:

Q5. By default how the value are passed in c++?

- a) call by value
- b) call by reference
- c) call by pointer
- d) call by object



Summary

In this lecture, you learned that:

➤ Concepts of :

- 1) Inline Functions
- 2) Call by value in C++
- 3) Call by reference in C++
- 4) Default arguments

