

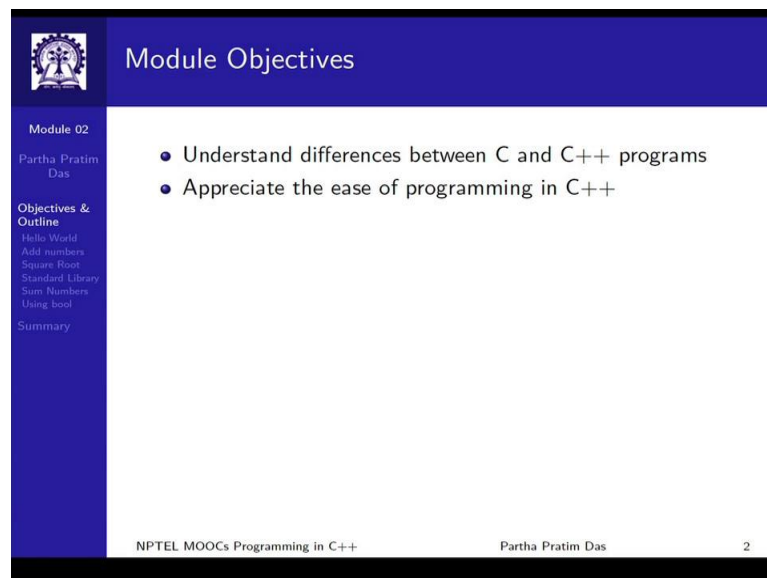
Programming in C++
Prof. Partha Pratim Das
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 04
Programs with IO and Loop

We will now discuss the module 2, the second module in the programming in C++ course.

In the first module, we have made a revisit of C programming language. We have recapitulated the concepts, the common concepts in C and made sure that we are prepared to slowly get familiar with C++. In this module and in the next 3 modules, we will talk about various programming examples and show how in C++, this program can be written more efficiently more elegantly and often with better ease than what is needed in C. So, in module 2, we will get started with understanding the basic differences between C and C++ programs.

(Refer Slide Time: 01:16)

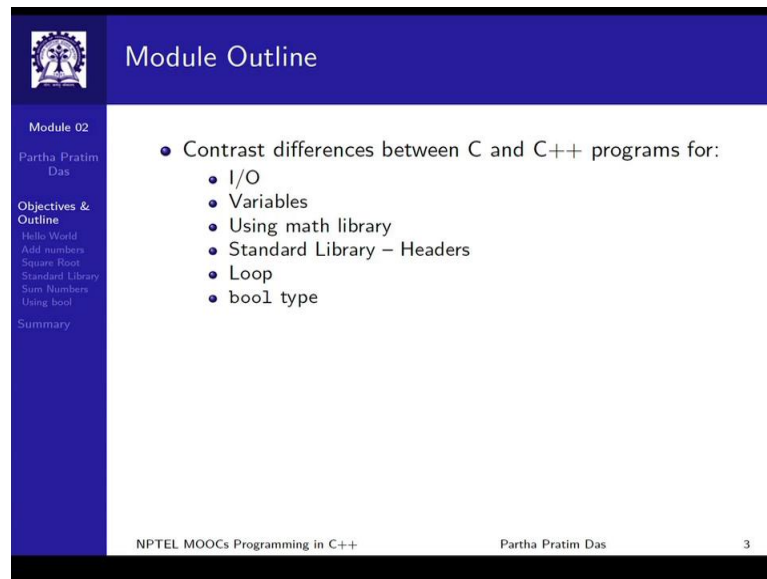


The slide is titled "Module Objectives" and features a blue header with the IIT Kharagpur logo. A sidebar on the left lists the module content. The main area contains two bullet points. At the bottom, there is a footer with "NPTEL MOOCs Programming in C++", "Partha Pratim Das", and the page number "2".

Module 02	Module Objectives
Partha Pratim Das	<ul style="list-style-type: none">• Understand differences between C and C++ programs• Appreciate the ease of programming in C++
Objectives & Outline	
• Hello World	
• Add numbers	
• Square Root	
• Standard Library	
• Sum Numbers	
• Using bool	
• Summary	

And we will try to appreciate the ease of programming in C++ as we go over this module as well as the following 3 modules.

(Refer Slide Time: 01:32)



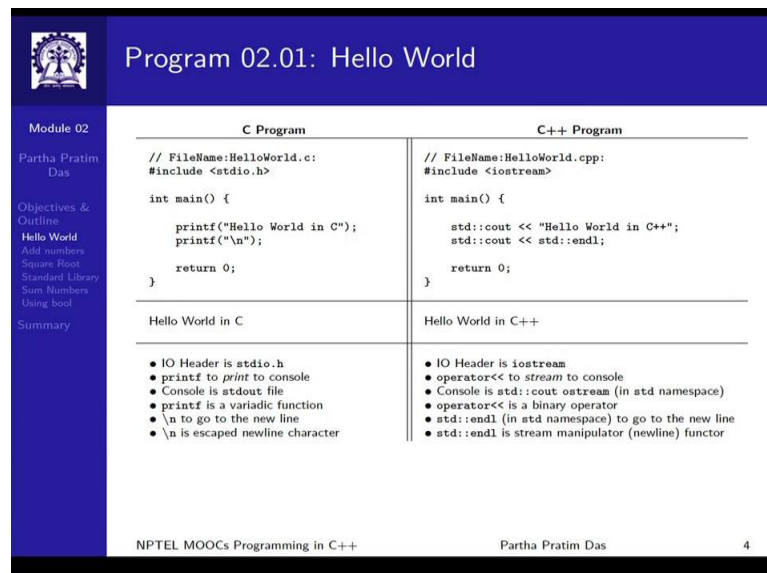
The slide is titled "Module Outline" and features a blue header with a logo on the left. A vertical sidebar on the left lists the module content: "Module 02", "Partha Pratim Das", "Objectives & Outline", "Hello World", "Add numbers", "Square Root", "Standard Library", "Sum Numbers", "Using bool", and "Summary". The main content area lists the topics to be contrasted between C and C++:

- Contrast differences between C and C++ programs for:
 - I/O
 - Variables
 - Using math library
 - Standard Library – Headers
 - Loop
 - bool type

At the bottom, it reads "NPTEL MOOCs Programming in C++", "Partha Pratim Das", and the page number "3".

We will primarily talk about the contrast in the areas of IO variables, math library standard library headers, loop and the bool type.

(Refer Slide Time: 01:43)



The slide is titled "Program 02.01: Hello World" and features a blue header with a logo on the left. A vertical sidebar on the left lists the module content: "Module 02", "Partha Pratim Das", "Objectives & Outline", "Hello World", "Add numbers", "Square Root", "Standard Library", "Sum Numbers", "Using bool", and "Summary". The main content area is divided into two columns: "C Program" and "C++ Program".

C Program	C++ Program
<pre>// FileName:HelloWorld.c: #include <stdio.h> int main() { printf("Hello World in C"); printf("\n"); return 0; }</pre>	<pre>// FileName:HelloWorld.cpp: #include <iostream> int main() { std::cout << "Hello World in C++"; std::cout << std::endl; return 0; }</pre>
Hello World in C	Hello World in C++
<ul style="list-style-type: none">• IO Header is <code>stdio.h</code>• <code>printf</code> to <code>print</code> to console• Console is <code>stdout</code> file• <code>printf</code> is a variadic function• <code>\n</code> to go to the new line• <code>\n</code> is escaped newline character	<ul style="list-style-type: none">• IO Header is <code>iostream</code>• <code>operator<<</code> to <code>stream</code> to console• Console is <code>std::cout ostream</code> (in <code>std</code> namespace)• <code>operator<<</code> is a binary operator• <code>std::endl</code> (in <code>std</code> namespace) to go to the new line• <code>std::endl</code> is stream manipulator (newline) functor

At the bottom, it reads "NPTEL MOOCs Programming in C++", "Partha Pratim Das", and the page number "4".

So, we start with the same initial program in C that is to print “Hello World”. So, here in two columns we show the program to print “Hello World” in C as well as in C++. You

can note some of the basic differences, first is the IO header has changed. In C, it was `stdio dot h`; in C++ it becomes IO stream. In C, when we do `printf`, we write to console; we print to console.

Here, we use an operator a pair of left arrow operators called the output streaming operator to stream to control and the console was `stdout` file in C, now it is a `cout` stream in C++. Also note that we are using a prefix before `cout` that is called `std` and it is written with `std colon colon`. This notations we will get used to quickly, this `std` is called a name space; the standard name space. So, any standard library symbol in C++ will be prefixed with this a particular prefix stream `std`.

Now, another major point to note in this sample 'hello world' program is when we do `printf`, we can have arbitrary number of parameters, we call this variadic function. So, here in the 'hello world' example we are seeing two use of `printf`, both of which use one parameter, the format stream. Of course, we do not have a format here; it is a constant string being printed. In contrast, in C++ the output streaming operator is a binary operator which as the stream on the left hand side and the content to print on the right hand side and it prints in this form.

So, if we look at the first output line, `std colon colon cout` output operator and within double quotes, we have a constant string it means that the hello world in C++ string will be streamed to the console `cout`. Also note that, the new lined character which was escape character back slash `n` in C. In C++ the same can be used, but there is another way to go to new line that is called `endl`, which is a short form of end line and we learn later on that. It, `endl` is basically a stream manipulated. So, this step we are trying to observe that the basic output system in C++ program can be done using `cout` and the output streaming operator.

(Refer Slide Time: 05:04)

Program 02.02: Add two numbers	
C Program	C++ Program
<pre>// FileName: Add_Num.c: #include <stdio.h> int main() { int a, b; int sum; printf("Input two numbers:\n"); scanf("%d%d", &a, &b); sum = a + b; printf("Sum of %d and %d", a, b); printf(" is: %d\n", sum); return 0; }</pre>	<pre>// FileName: Add_Num_c++.cpp: #include <iostream> int main() { int a, b; std::cout << "Input two numbers:\n"; std::cin >> a >> b; int sum = a + b; // Declaration of sum std::cout << "Sum of " << a << " and " << b << " is: " << sum << std::endl; return 0; }</pre>
<pre>Input two numbers: 3 4 Sum of 3 and 4 is: 7</pre>	<pre>Input two numbers: 3 4 Sum of 3 and 4 is: 7</pre>
<ul style="list-style-type: none">• <code>scanf</code> to <code>scan (read)</code> from console• Console is <code>stdin</code> file• <code>scanf</code> is a variadic function• Addresses of <code>a</code> and <code>b</code> needed in <code>scanf</code>• All variables <code>a</code>, <code>b</code> & <code>sum</code> declared first (C89)• Formatting (<code>%d</code>) needed for variables	<ul style="list-style-type: none">• <code>operator>></code> to stream from console• Console is <code>std::cin istream</code> (in <code>std</code> namespace)• <code>operator>></code> is a binary operator• <code>a</code> and <code>b</code> can be directly used in <code>operator>></code> operator• <code>sum</code> may be declared when needed• Formatting is derived from type (<code>int</code>) of variables
NPTEL MOOCs Programming in C++	Partha Pratim Das 5

We move to the next program, where we are illustrating a very simple arithmetic program, which has two variables, `a` and `b` and adds them to generate their sum. The values of these variables are read from the console again, which in C is `std in` and we use the `scanf` function, which all of you are familiar with the format string `scanf`, like `printf` is a variadic function that is, it takes variable number of parameters. Here, we see a form of `scanf` which is taking 3 parameters; the format string and the addresses of `a` and `b`, respectively.

In contrast, in C++ program we introduce another operator which is used for streaming from an input stream. So, this is called an input streaming operator, It is again a pair of arrows, but the arrows now direct in the left from left to right. So, if you look into `std colon C` in, the input operator then `a` it means that is being read by C in. What is interesting is, in this format in C++ you can actually put multiple operators multiple operands multiple variables one after the other as we are showing here.

So, after streaming `a`, from the standard input we again, stream `b` from the standard input. So, this notation means that first `a` and then variable `b` will be read from the standard input of the console. We also show how these variables are output to the standard streaming output which is `cout` or `std out` as in C program.

The two major differences is, that you must note here is, one we do not need to use the format string in C++. In C we know, if I want to print an integer variable I need to specify in the format string that the variable is to be printed in percentage d format, which denotes that is an integer type of data to be printed, In C++, in contrast the variable does not need to be specified with a format, the compiler knowing that it is an integer variable will automatically decide the format that is required for printing it and print it in the right way.

A second major difference that you should note in between these two programs is when we do scanf, we are reading values from the std in and since in reading that value the value original value of variable a has to be changed to the value that is input by the user. We need to pass the address of a as it would be familiar in C this is kind of a call by address mechanism that is being used, where we pass the address of the variable as a call by value parameter to the scanf function.

In contrast, in C++ when we are reading from the input stream, we do not need the address to be passed; we can simply specify the variable and the compiler will take care of the rest. When we learn more of the C++ operators and call mechanisms, we will understand how this really works, but it certainly becomes more convenient to get read of the format string as well as the need to specify either the variable for printf or the address of the variable for scanf in C++ everything can be done uniformly.

There is a other small difference that you may note in terms of the declaration of variable sum, in the C program the variable sum is declared at the top after the variables a and b, because the original C or the old version of C, which is C89 specified that all declaration of variables must happen before the first executable statement in the program.

In the C program that we see here in the first executable is the printf function call. So, all variables must be declared before that. In C++ this restriction does not exist. So, we can declare the variable as and when we need then when we are starting the name we just need the variables a and b because there need to be read, but when we need to do their sum at that point we can declare sum as a variable, and then use a plus b and put that result into sum to initialize the value of sum. Of course, this should be noted that later

version of C that is C99 does allow you to defer the declaration of variables to like in C++ to the point when you actually need the variable. So, please carefully read and run this program at your computer to understand the similarity and the differences better.

(Refer Slide Time: 11:01)

Program 02.03: Square Root of a number

	C Program	C++ Program
<p>Module 02</p> <p>Partha Pratim Das</p> <p>Objectives & Outline</p> <p>Hello World</p> <p>Add numbers</p> <p>Square Root</p> <p>Standard Library</p> <p>Sum Numbers</p> <p>Using bool</p> <p>Summary</p>	<pre>// FileName:Sqrt.c: #include <stdio.h> #include <math.h> int main() { double x; double sqrt_x; printf("Input number:\n"); scanf("%lf", &x); sqrt_x = sqrt(x); printf("Sq. Root of %lf is:", x); printf(" %lf\n", sqrt_x); return 0; } Input number: 2 Square Root of 2.000000 is: 1.414214</pre> <ul style="list-style-type: none"> • Math Header is math.h (C Standard Library) • Formatting (%lf) needed for variables • sqrt function from C Standard Library • Default precision in print is 6 	<pre>// FileName:Sqrt_c++.cpp: #include <iostream> #include <cmath> using namespace std; int main() { double x; cout << "Input number:" << endl; cin >> x; double sqrt_x = // Declaration of sqrt_x sqrt(x); cout << "Sq. Root of " << x; cout << " is: " << sqrt_x << endl; return 0; } Input number: 2 Square Root of 2 is: 1.41421</pre> <ul style="list-style-type: none"> • Math Header is cmath (C Standard Library in C++) • Formatting is derived from type (double) of variables • sqrt function from C Standard Library • Default precision in print is 5 (different)
	NPTEL MOOCs Programming in C++	Partha Pratim Das 6

We will next move on to another program, which is again a simple program using mathematical computation, which I am sure you have done at some points of time in the C program. So, we know that in C there is a math dot h header as a part of the C standard library, which has a number of useful functions. So, here we are just illustrating the use of one such function called Sqrt, for finding the square root of a double variable. It takes a double variable and returns a double result which is a square root of the parameter that is passed to Sqrt.

The similar function can be invoked in C++ as well. So, we are showing how to do that, you please note that the header that we use in C++ now as changed in its name. In C we are calling it math dot h the same header, in C++ is called cmath and we will see this is a common convention that any C standard library header can be used in C++ as well, but when you use that, you add a C at the beginning of the name. The C means that the standard library header is coming from the C standard library and the other difference is you drop the dot h extension to the file name that existed in C, you just call it cmath.

Of course, when we do this, as I had mentioned earlier in terms of the names of cout and endl, these also are in the same name space of std, which means that the function Sqrt in C is just called Sqrt, whereas in C++ this function name will become Sqrt prefixed with std, that is full name is std::cout, std::endl. Now, here we also show another short cut or convenient way to express the standard library symbols, note after the hash includes in the C++ program, we have written a line saying using namespace std. This means that if we include this line then any standard library symbol that occurs in the C++ program will be assumed to have std:: as a prefix, we will not have to every time write std::cout, std::endl or std::Sqrt.

So, this is a convenient way of doing that you could either make use of using namespace feature in C++ or if you are not using it then all standard library symbols will need to be prefixed with std::. Rest of the program is very easily understandable and it is pretty much like the C program that you see on the left or the changes are according to the input and the output streaming as we have already seen. So, with this we will move on and take a look into the C++ standard library.

(Refer Slide Time: 14:31)

namespace std for C++ Standard Library

C Standard Library	C++ Standard Library
<ul style="list-style-type: none"> All names are global stdout, stdin, printf, scanf 	<ul style="list-style-type: none"> All names are within std namespace std::cout, std::cin Use <code>using namespace std;</code> to get rid of writing <code>std::</code> for every standard library name
<p>W/o using</p> <pre>#include <iostream> int main() { std::cout << "Hello World in C++" << std::endl; return 0; }</pre>	<p>W/ using</p> <pre>#include <iostream> using namespace std; int main() { cout << "Hello World in C++" << endl; return 0; }</pre>

NPTEL MOOCs Programming in C++ Partha Pratim Das 7

Just summarizing, what I have already specified that in a C standard library all names are global, that is all standard library functions, macros; they are available to any function by

the name that they have. So, therefore, all C standard library names are actually for all practical purposes get reserved in a way that is you cannot write a printf function of your own and also continue to use the printf function given in the C standard library because the moment you write a printf function of your own, that also as a name in the global space.

You will understand this more when we study about name spaces formally in C++, but please note that all names are just available in the global space. In contrast, in C++ standard library all names are in the std name space. This is specifically reserved for the standard library this name spaces and all names are prefixed with std colon colon meaning that the name occurs within this std name spaces. So, name space is like when we use in our own names, it is like a family name or the last name that we use. So, I am Partha Pratim Das. So, Das is my last name and my name is Partha. So, there could be another Partho in some other family names, say Partha Pratim Chakrabarti. So, these are distinguished by the different family names that exist.

So, names space is something similar to that. We will talk more about it subsequently. So, we just show also illustrate here that if you do the short cut of putting using names spaces std, then you do not need to prefix all standard library names with that std colon colon name space.

(Refer Slide Time: 16:38)

Standard Library Header Conventions

	C Header	C++ Header
C Program	Use .h. Example: <code>#include <stdio.h></code> <i>Names in global namespace</i>	Not applicable
C++ Program	Prefix c, no .h. Example: <code>#include <cstdio></code> <i>Names in std namespace</i>	No .h. Example: <code>#include <iostream></code>

- Any C standard library header is to be used in C++ with a prefix 'c' and without the .h. These symbols will be in std namespace. Like:

```
#include <cmath> // In C it is <math.h>
...
std::sqrt(5.0); // Use with std::
```


It is possible that a C++ program include a C header as in C. Like:

```
#include <math.h> // Not in std namespace
...
sqrt(5.0); // Use without std::
```


This, however, is not preferred.
- Using .h with C++ header files, like `iostream.h`, is disastrous. These are deprecated. It is dangerous, yet true, that some compilers do not error out on such use. Exercise caution.**

NPTEL MOOCs Programming in C++ Partha Pratim Das 8

Now, I would like to highlight something, which is very specific about headers in of standard library. So, we have noted that C++ is backward compatible to C what does that means; it means that any C program should be executable as a C++ program also. This brings in another question that what do we do with the standard library of C, as I have already illustrated that standard library of C can also be used in a C++ program, but there is a small point to be noted by you mix the program from C with the program from C++ in terms of how you specify the standard library headers.

So, this table show you how you can do this on the left, on the rows we show language in which you are writing the program and on the column we are showing the header being used from which standard library, whether it is from C or it is from C++. So, if you are writing a C program and you are using C standard library headers, we all know you will include like `stdio dot h`. If you are writing a C++ program and including a C standard library header then as I have mentioned, you will need to prefix the C standard library name with a C. So, `stdio dot h` now becomes `C stdio` and you will have to drop the dot h from the name and all of these symbols from the C standard library now gets into the std name space and we will have to be prefixed with `std colon colon`.

If you write a C++ program and include the C++ standard library, you will simply include it as hash include IO stream. All standard library headers in C++ do not have any dot h in their file extension, this is for a historical reason which I would try to explain at a later point of time, but please note that IO stream dot h should not be included and the last box in this matrix that if you are writing a C program and you want to use a C++ header certainly is not applicable because you cannot do that C++ has a lot of features which C does not support and therefore, such a use cannot be done.

Specifically note and I have highlighted with red that by mistake or by the practice of using dot h as a file name extension for standard library headers in C. If you include IO stream dot h in a C++ program, your compiler may not actually give you an error which means that your compiler is actually dated and you should move to a more recent compiler and therefore, this is a very dangerous preposition because you are making a mistake IO stream dot h or for that matter any C++ standard library header with dot h extension file has been; all of them have been depicted. They are no more in use, but some compiler still continue to allow them because they were written before all these changes were done in the C++ standard. So, please keep in mind these conventions of standard library headers.

(Refer Slide Time: 20:26)

Program 02.04: Sum n natural numbers	
C Program	C++ Program
<pre>// FileName:Sum_n.c: #include <stdio.h> int main() { int n; int i; int sum = 0; printf("Input limit:\n"); scanf("%d", &n); for (i = 0; i <= n; ++i) sum = sum + i; printf("Sum of %d", n); printf(" numbers is: %d\n", sum); return 0; }</pre>	<pre>// FileName:Sum_n_c++.cpp: #include <iostream> using namespace std; int main() { int n; int sum = 0; cout << "Input limit:" << endl; cin >> n; for (int i = 0; i <= n; ++i) // Local Decl. sum = sum + i; cout << "Sum of " << n ; cout << " numbers is: " << sum << endl; return 0; }</pre>
<pre>Input limit: 10 Sum of 10 numbers is: 55</pre>	<pre>Input limit: 10 Sum of 10 numbers is: 55</pre>
<ul style="list-style-type: none"> • i must be declared at the beginning (C89) 	<ul style="list-style-type: none"> • i declared locally in for loop
NPTEL MOOCs Programming in C++	Partha Pratim Das 9

Next, we will look into the use of loops which is very similar to what you we have in C. So, here we are just adding a set of a sequence of numbers starting from 0 up to n and summing them using a for loop. The same program almost identically will work for C++ except for the differences in the IO headers and the cout streaming convention. You may also note that for loop the loop index, i can be declared within the parenthesis for construct.

If you do that then this declaration of i is local to this for loop that is once you come out of the for loop you are in the subsequent cout statement or later on i will not be considered to have been declared. So, this was introduced, so that you can just whenever you need local index variables you could quickly locally use them and do not have to really think about whether you have declare that variable earlier whether it is being used in some other context and so on. You can just locally declare them and use them in C++. This was not possible in C89, but this is now possible in C99 also.

(Refer Slide Time: 22:02)

Program 02.05: Using bool

C Program	C++ Program	
<pre>// FileName:bool.c: #include <stdio.h> #define TRUE 1 #define FALSE 0 int main() { int x = TRUE; printf ("bool is %d\n", x); return 0; }</pre>	<pre>// FileName:bool.c: #include <stdio.h> #include <stdbool.h> int main() { bool x = true; printf ("bool is %d\n", x); return 0; }</pre>	<pre>// FileName:bool_c++.cpp: #include <iostream> using namespace std; int main() { bool x = true; cout << "bool is " << x; return 0; }</pre>
<pre>bool is 1</pre>	<pre>bool is 1</pre>	<pre>bool is 1</pre>
<ul style="list-style-type: none"> • Using int and #define for bool • May use .Bool (C99) 	<ul style="list-style-type: none"> • stdbool.h included for bool • _Bool type & macros (C99): bool which expands to _Bool true which expands to 1 false which expands to 0 	<ul style="list-style-type: none"> • No additional headers required <pre>bool is a built-in type true is a literal false is a literal</pre>

Module 02

Partha Pratim Das

Objectives & Outline

Hello World

Add numbers

Square Root

Standard Library

Sum Numbers

Using bool

Summary

NPTEL MOOCs Programming in C++

Partha Pratim Das

10

Finally, in the last section of this module, we illustrate the use of the Boolean type. We all know that C has a possible use of a Boolean type, which is a type where we say that it can take a value, either true or false. Now, C which is C89, the original old C that we have did not have a separate type for bool. So, what it did is it was using the int to

interpret bool that is anywhere you want to put a Boolean condition or a Boolean value you will declare a int variable and set that to 0, if you want to mean false and set that to something non-zero, if we want to mean true.

So, out of these 3 columns if you look at the left most columns is the most common way that C programs had been dealing with the Boolean, you could for convenience define two constant; true and false to be 1 and 0 and use them in your program, but as I show the int variable to be the variable x to be used for the Boolean condition is declared as of int type and is initialized with true. So, if you print it will show that it has a value 1; this is what existed in C89. Subsequently in C99, a change has been made to introduce a bool type.

Now, before looking into that let us first look into the right most columns, which is the C++ program. In C++ you have bool as a built-in type as you have int, char, float, double. Similarly, you have a bool type, this bool type as only two literals; true and false both in lower case. So, they are keywords now reserved as well. So, you want to similarly define x for use as a Boolean variable, you can directly use bool which mix it very easy to understand that you are actually dealing with a Boolean value and you could initialize it with true or false and, but if you try to print the value of this variable, it will not print true or false, it will actually print 1 or 0. If it 1, if it true and 0, otherwise.

Naturally, being able to use and explicit built-in type bool has a lot of advantages. The most significant of that is from the C++ program, if you have used bool to specify your Boolean value, anyone else who reads the program will be able to understand that this variable cannot take any other value, other than true or false. In contrast, if we use the C style of using int to mean Boolean value, then it can actually take multiple different values which are interpreted as true or false, as a case may be.

Now, in the middle column, we see an interesting extension of C programming language in C99 standard what C99 came up with. C99 introduced explicit Boolean type and that is given by the name underscore bool, where b is written in capital, but that since that is not a very normal natural way to write bool, it has also provided a new standard library header called std bool dot h, where three macros are provided.

The first macro defines `bool` in lower case as same as underscore capital `bool`. So, if we use the `bool` in a lower case in a C99 program, then we will you are actually using that new predefined type underscore capital `bool` and it also defines `true` and `false` as 1 and 0 in the header, so that you could use them as constant here. So, if we are; whenever you are using C, now you should always the `bool` type and not use `int` and interpret it as a `bool` type. In C++ certainly it is come out to be a built-in type. So, we show that it has several other advantages as well as we go along with the different types.

(Refer Slide Time: 26:54)

Module Summary

- Understanding differences between C and C++ for:
 - IO
 - Variable declaration
 - Standard Library
- C++ gives us more flexibility in terms of basic declaration and input / output
- Many C constructs and functions are simplified in C++ which helps to increase the ease of programming

NPTEL MOOCs Programming in C++ Partha Pratim Das 11

In this module, we have tried to understand the basic differences between C and C++ with specific focus to; how you perform input output? How you declare variables? And how the standard library of C and C++ are used in C++? We have started to see that C++ gives us more flexibility in terms of how we can declare and how we can do input output.

Now, we do not need to have those complicated `printf` statements, stream `printf` function calls where the formats are into a separate strings, variables are listed separately. We do not need to remember that `scanf` needs the address of variables and so on and in this way many construct and functions have been simplified in C++, which will help in increases ease of programming.