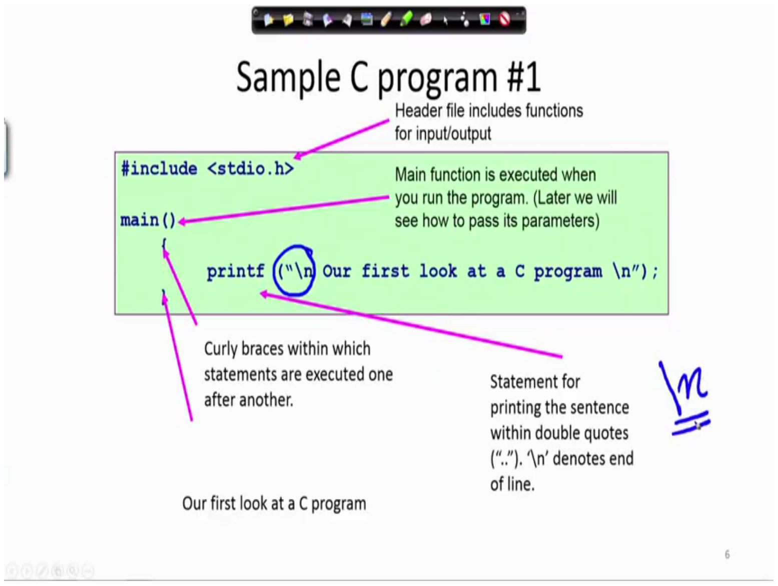
Problem Solving through Programming In C
Prof. Anupam Basu
Department of Computer Science & Engineering
Indian Institute of Technology, Kharagpur

Lecture - 08
Variables and Variable Types in C

So, we are looking at a sample C program.

(Refer Slide Time: 00:18)



Sample C program #1

```
#include <stdio.h>

main()
    {
        printf ("\n Our first look at a C program \n");
    }
```

Header file includes functions for input/output
Main function is executed when you run the program. (Later we will see how to pass its parameters)
Curly braces within which statements are executed one after another.
Statement for printing the sentence within double quotes ("..") . '\n' denotes end of line.
\n
Our first look at a C program
6

And as we had discussed in the last lecture, we have a structure as is shown here. You can see that there is a main function here a main function is there, and that main function is covered by two parentheses. Now, we can see that we start with the header file that includes stdio dot h. An stdio dot h stands for standard IO.

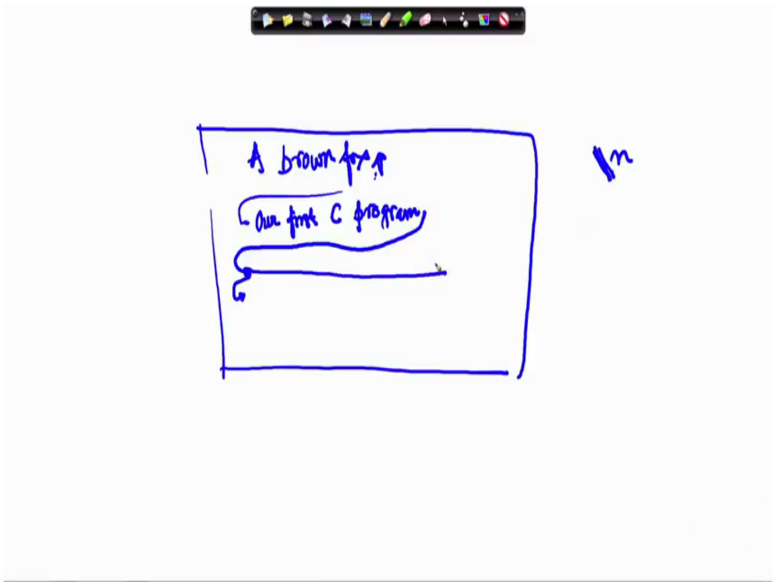
That means, whenever we will get some instructions to carry out input or output that will be in reference to or with respect to the standard input that is a keyboard; that means, the data will be taken from the keyboard. And if anything is to be printed it will be printed on the screen. So, that is a default thing for any C program we have to put in stdio dot h include hash include, this is known as this is read as hash include stdio dot h. Unless you want to take the file do not want to take it from the keyboard or do not want to print on the screen.

Next thing is that the main function which as I said in the last class that there must be a ma main function. And the main function will have a place for the parameters, you can see here for the parameters here which may or may not be empty. The third point is the structure the overall boundary of the main function of for that matter for any function there should be a boundary specified, and that boundary specified with by this to curly brackets.

Next we come to the statement. As I had said that just like any other language c also has got some words which are understood by any by the compiler; print if is one such word which is the statement for printing the sentence that is given within these double quotes. You can see that double quotes here and so, what will it print? It will print our first look at the C program here as his shown here that will be printed.

Print f within quote here there is a quote n quote and here is a start quote our first look at the C program. In addition there is I am sorry let me go up, we can see to special symbols which are these backslash ns right let me just clear it out, here you can see this backslash n something of the sort. This means go to a new line or end of line now suppose.

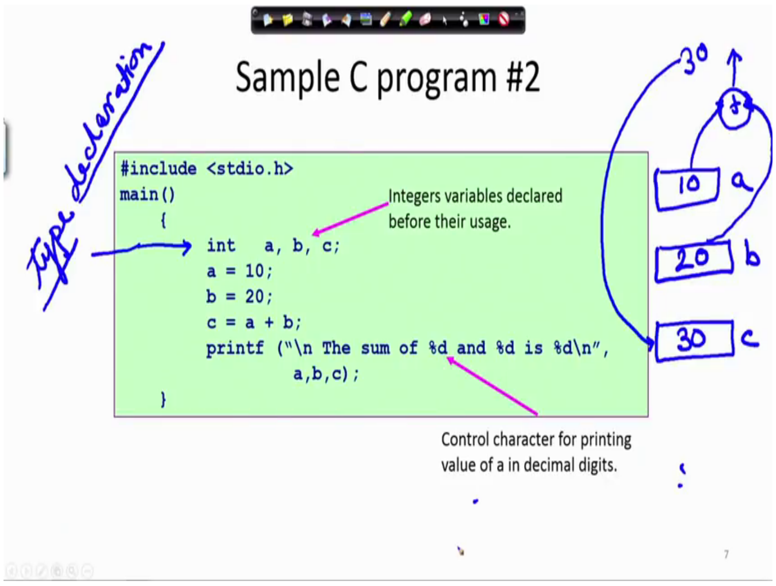
(Refer Slide Time: 04:09)



Let us see suppose I am I had something like this I had printed something all right I had printed something this is a screen, I had printed a brown fox. Now after I printed; that means, a computer printer you can see where my pen is my pen is lying here right my pen is here. Now if I say backslash n this is no radar backslash, backslash n; that means, my pen will come to the beginning of the new line and here I will write our first C program, and by default my pen is here, but since I have given another backslash n the pen will come here.

So, next time if I again start with the backslash n it will come to the new line, and if I do not give a backslash n it will continue from here. You will understand this more when we look at more number of programs, it will be much more clear to you. So, this the structure of a very simple C program we will see more of this.

(Refer Slide Time: 05:39)



Sample C program #2

```
#include <stdio.h>
main()
    {
            int   a, b, c;
            a = 10;
            b = 20;
            c = a + b;
            printf ("\n The sum of %d and %d is %d\n",
                        a,b,c);
    }
```

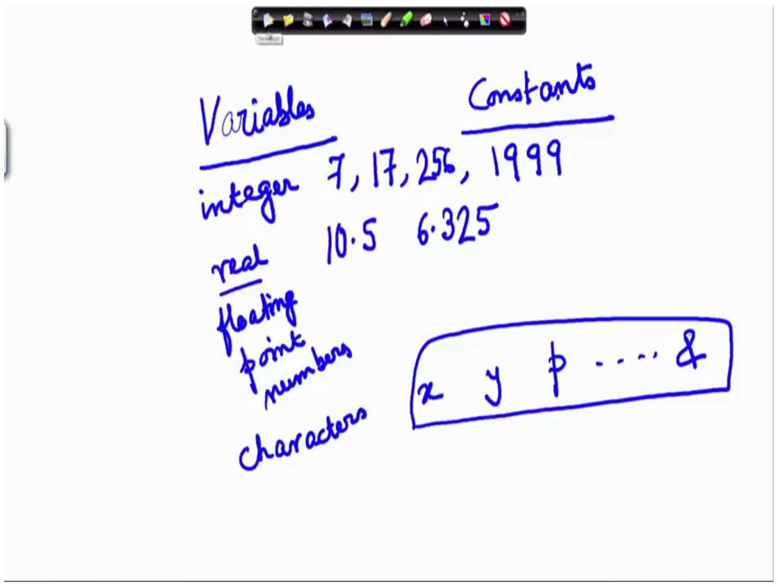
Integers variables declared before their usage.

Control character for printing value of a in decimal digits.

Now, here is a another program. Again now it is easier to understand you have got an includes t d i o dot h which has to be there for any C program, there will be a main function as is being shown here, and there are parenthesis between these two the program should be written, there is a boundary of the main function. Now here there are some more new things which are being shown to you in the form of example.

So, you see the first line is int a, b, c what is meant by that is that I am I will be using in this program 3 variables a b and c and each of them is of type integer. Now you are already acquainted with the term variables, but you are probably not acquainted with the term type of a variable.

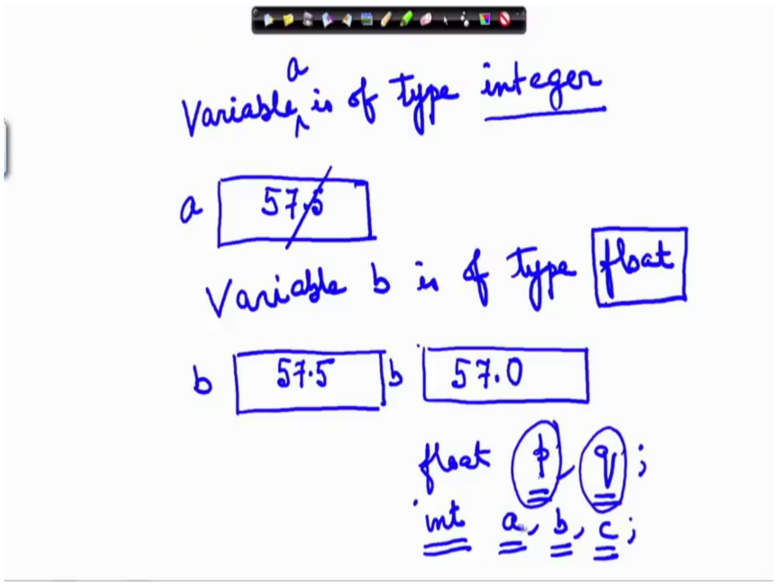
(Refer Slide Time: 06:50)



So, quickly let us go to that, we have got variables and the variables can be of different type, also constants can be of different type for example, the variables can be of type integer for example, 7,17 all these things 256, 1999 all these are integers. Another type of variable can be real numbers which are say 10.5, 6.325 fractional numbers right these are real.

This is also known as in c as floating point numbers. Now integers are these reals of flow floating point numbers of these. Similarly I can have characters like say x y p whatever these are different characters or maybe and is a character. So, each of any of these alphanumeric and all those can be characters now. So, we know what is an integer, what is a real what is a floating point, what is a floating point what is a character.

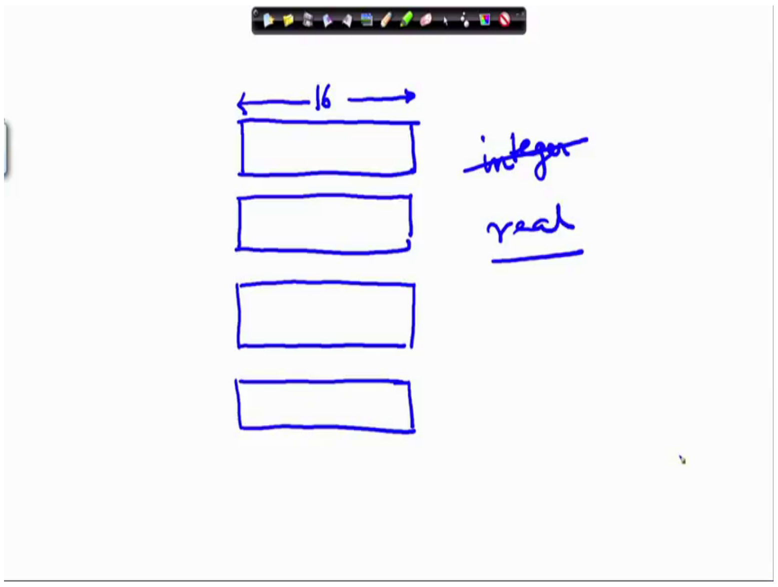
(Refer Slide Time: 08:51)



Now, a variable is of type integer, what does it mean? As you know a variable is nothing, but a memory location right it means and suppose this variable I say variable a is of type integer. So, there is a memory location corresponding to a and this memory location can only hold values which are integers. So, 57 I can store here, but if I try to store 57.5 here, it will not store 57.5, because it has been told to be an integer.

Similarly if I say variable b is of type floating point, I am in short I am writing float; that means, this variable b can store a floating point number. So, 57.5 can be stored here. Even if I try to store in b 57 in b if I store 57 just 57 then that will be stored as 57.0 all. Similarly there can be types of characters. Now depending on the type of the variable the compiler assigns different sizes of memory for the different variables.

(Refer Slide Time: 10:50)



For example in standard c compilers and it is a convention that for an integer two memory locations are allocated.

Now, how big this will be say sometimes it is my each of these memory locations can be 16 bits, in that case I am using 2 16 bits; that means, 32 bits to store an integer whereas, for a real number. Now this is a convention, 4 locations are used for storing a real number. So, 64 bits 16, 16, 16, 16. So, that will be for real numbers now. So, depending on what I write, what how I define the type of the variable when I say a b c I also say what type of variability; is it an integer, is it a real or what it is.

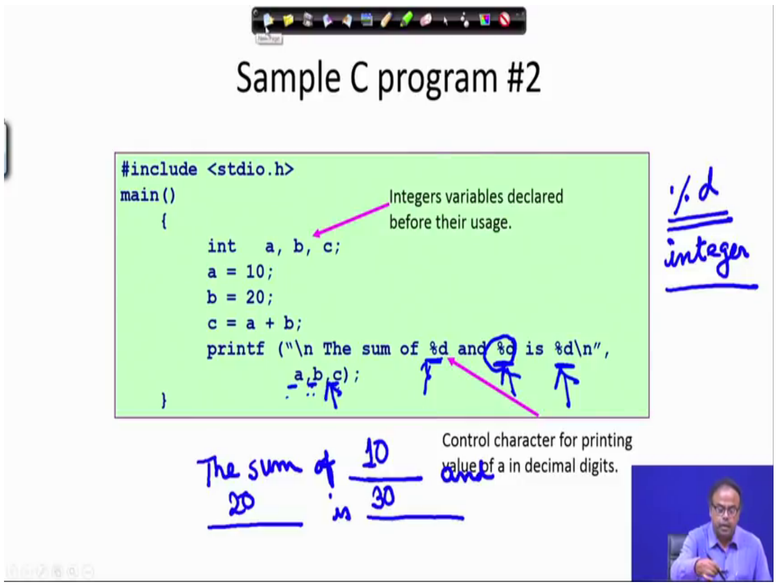
Now, if you come here we will see that here I have written float and I did not write floating point number. Now in c in order to specify variable to be real we declare that to be float say p q; that means, semi colon; that means, p and q are two variables which are of type floating point number right and if I write int a b c; that means, a b and c are 3 variables, which all of type floating integer type int. So, we do not write integer we just write in c int. Now obviously, then for p how many bytes how many depending on how many bits will be given how many memory locations will be given for p may be 4, q it will be 4, but for a b c it will be only 2 all right.

Now, let us therefore, go back to the program here, here we find int a b c now this statement this is called a type declaration all right this is known as a type declaration this is the first statement. So, I have declared the variables now in C program before the variable is used it should be declared about its type. The other thing is you can see this. So, here you can see a has been assigned 10, b has been assigned 20; that means, what.

That means in the memory location a corresponding to a memory location corresponding to a 10 has been written, and to the memory location corresponding to b 20 has been written all right and this statement c assigned a plus b, I had explained in another lecture that; that means, this data this value 10, and these value 20 will be taken out on the from the left hand side from the corresponding memory locations they will be added and the result will be 30 all right; the result will be 30 and the 30 will be written into the location c. So, this will be 30 this much is clear.

Now what about this line? I know that a print if statement just tells me that I have to print whatever is there in the quote. So, how will the print look like? Please note the sum of percentage d and percentage d is percentage d this is equivalent to writing this.

(Refer Slide Time: 15:38)



Sample C program #2

```
#include <stdio.h>
main()
    {
        int   a, b, c;
        a = 10;
        b = 20;
        c = a + b;
        printf ("\n The sum of %d and %d is %d\n",
                a,b,c);
    }
```

Integers variables declared before their usage.

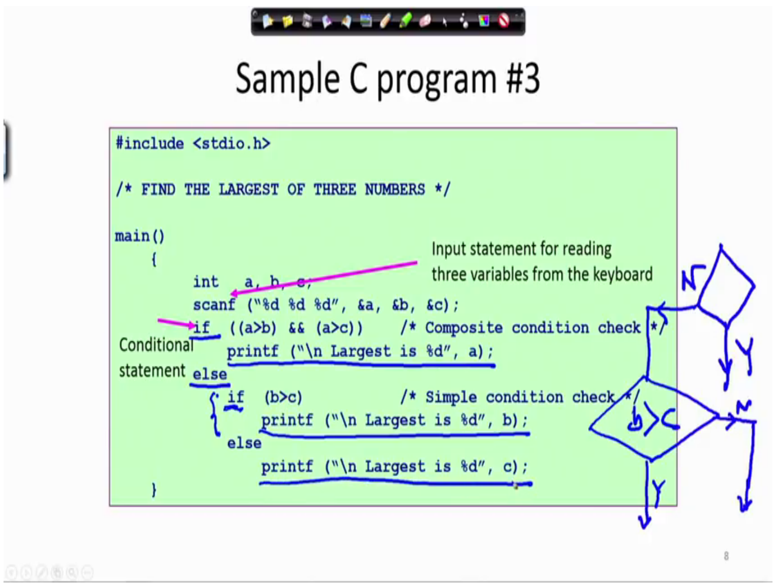
Control character for printing value of a in decimal digits.

The sum of dash and dash is dash there are 3 gaps, one here, one here and one here now how will these gaps we filled? They will be filled by the values of a, b and c respectively each of these dashes will be filled up by the respective values of a b c. Now this percentage d is a format statement is saying that this gap can be filled by a digit or by an integer.

This gap can be filled only with an integer. Now here I a is an integer therefore, this gap will be filled with the integer value 10 and percentage d, and it is being printed as it is because it is within this double quote and dash. this dash will be filled with another digit, and what is the digit? The second space b and b is 20. So, it will be 20 is what is a plus b is c and what is the value of c c the value of c is 30.

So, this is what will be printed all right. So, this is how we print a sentence, where I want I have got some places for different variables and these places will be filled up by the values of the variables, whose names are being specified here. I repeat this gaps will be filled by the values of the variables whose names or whose identifier are been specified here and the type of this variable, and the specification of this dash should match. Now, it will be printed sum of 10 and 20 is 30; now coming to the third sample program.

(Refer Slide Time: 18:13)



Sample C program #3

```
#include <stdio.h>

/* FIND THE LARGEST OF THREE NUMBERS */

main()
   {
        int   a, b, c;
        scanf ("%d %d %d", &a, &b, &c);
        if  ((a>b) && (a>c))    /* Composite condition check */
           printf ("\n Largest is %d", a);
        else
           if  (b>c)             /* Simple condition check */
              printf ("\n Largest is %d", b);
           else
              printf ("\n Largest is %d", c);
   }
```

Input statement for reading three variables from the keyboard
Conditional statement
N
Y
b>c
N
Y
8

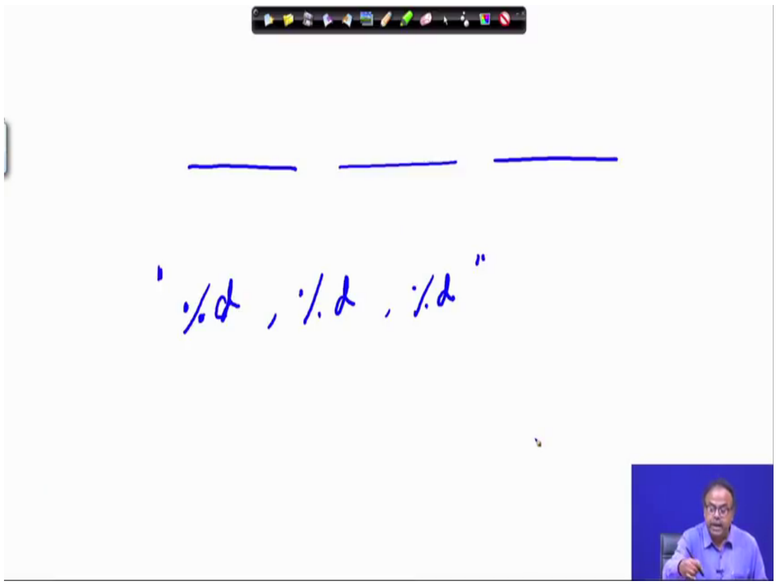
Now, this is a little more complicated, and this is trying to find out the largest of 3 numbers; that means, what we had done in a flowchart exercise finding the max of 3 numbers. Now that was discussed using flowcharts and pseudo code and here we are discussing that using a program a C program how is that idea translated into C program. Again let us start with even before that I would like to point out something, again let us revise the structure of the program, we start with header stdio dot h, now we start we put in a comment what is this? We saw in the last class it is a comment this comment is just telling us the compiler has got nothing to do with, it is just telling us that this program finds the largest of 3 numbers.

Then as usual we have got a main function which must be there and there should be a parenthesis which is delineating the boundary of this program. Now for and also we have seen this, I am using 3 variables a b and c and I have put in the type of that. The next new thing that is coming up is here scan f, this is an input statement for reading 3 variables from the keyboard. Now recall that I said that for every language there are there is a vocabulary, there is the set of words that the language understands.

So in c, in an earlier one we have seen in an earlier program we have seen the statement print f right we had seen a program print f. So, print f is a particular word, now we now encounter another word int is also another word; int is another word which specific to c and the meaning of this every word will have a meaning. So, this means that whatever follows are variable names of type integer. Whatever follows this word int are variable names of type integer.

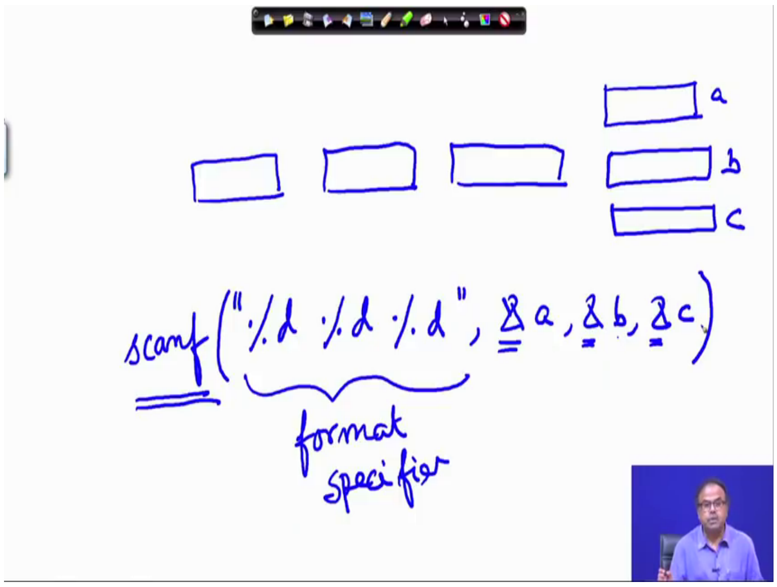
Next, new word that we are getting is scan f. Just like we had seen print f earlier all right print f is a word, that tells us that something is being printed here scan f is an int is a word that tells that whatever is inside this parenthesis is a input statement. Now let us study this a little bit here. Here again you will see that there is a percentage d percentage, d percentage 3 percentage ds within quote and that percentage d means it is a specified for an integer, some integer is being will be that scan f means say assume it is really.

(Refer Slide Time: 21:47)



So, therefore, it something like this, I am going I am creating as if 3 places, because corresponding to each percentage d, percentage d, percentage d, percentage d within the quote, I am creating 3 spaces and followed by that let us go back to this, I am sorry here there should be no comma all right; I am just giving 3 percentage ds.

(Refer Slide Time: 22:21)



So, let me go back here. So, it should be within the court here is scan f, within quote percentage d blank percentage d blank, percentage d and the quote is closed comma; that means, as if just to understand I am creating 3 spaces each of which I have ready to accept an integer and where will those be stored? Here I am writing and a, and b, and c. Now here I will request you to just forget about this ampersand sign forget about this for the time being this will explain a little later.

Just assume that a, b and c, but before that for any read statement will have to put an amperesand before the variable names there is a reason for that. What is the meaning of this sentence? The meaning of this sentence is that there are 3 places which are ready to hold 3 variables, which will come to the locations a b c; when I am reading as if I am reading from the user 3 variables a b c ok.

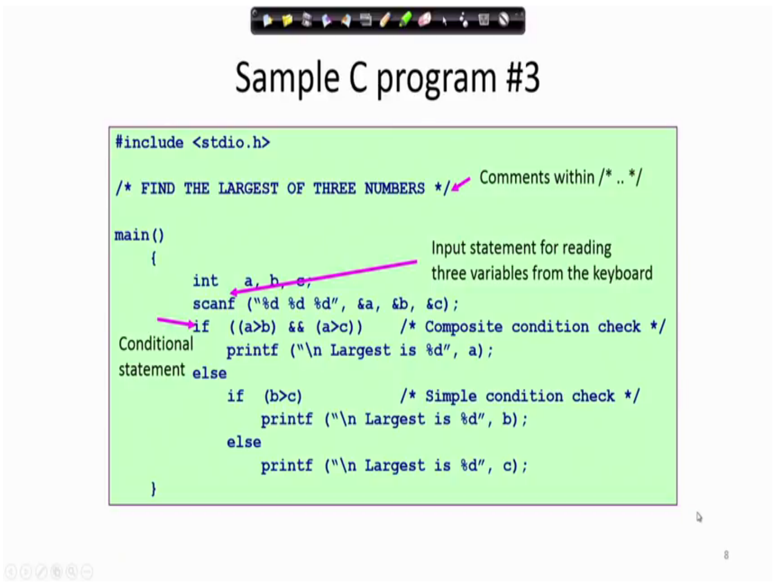
Now, why I put that amperesand will explain a little later. So, 3 space have been created and what are the spaces? These spaces are nothing, but these 3 memory locations which 3 memory locations whose names are being specified here all right. So, this is something that you have to be a little careful and practice a little bit it will be very easy later on. So, within the quote I specify the format specify that is integer in this case percentage d and this other variable names.

Now, let us go back to that anything that is new here. Now a new word we are encountering here, if is a conditional statement you did not bother about it, if you recall we had in the flowchart we had a thing called diamond right where we are taking some decisions based on some conditions yes and no right. We are doing that here we are looking at some condition here if a is greater than b and a is also greater then c, if that is yes then I am printing that a is the largest number. Try to apply simple logic here, if is a conditional word condition is a conditional word and this entire statement starting from if to this semi colon is one statement, and where I check the condition whether a is greater than b and a is greater than c.

If that is so, that is if the diamond comes out with that yes, then I am printing that the largest number is a. Otherwise, that means if the answer to this is no otherwise I am again checking again if again I am checking another condition, I am checking another condition here what to do I what my checking? Is b garter than c yes if yes then I am printing the largest number is b otherwise is yes, otherwise if it is no then I am painting the largest number is c.

So, we are encountered in some new words if and else. If means if in the diameter result of this diamond box is yes then this statement will be executed otherwise; that means, if the answer to this diamond box is no, then this part will be executed. Now here again the second block I am checking if b is greater than c, again if this is true for this condition this will be executed when this condition is true or yes if it is false then this statement will be executed. So, it is a little more complicated program, but a very useful example.

(Refer Slide Time: 27:30)



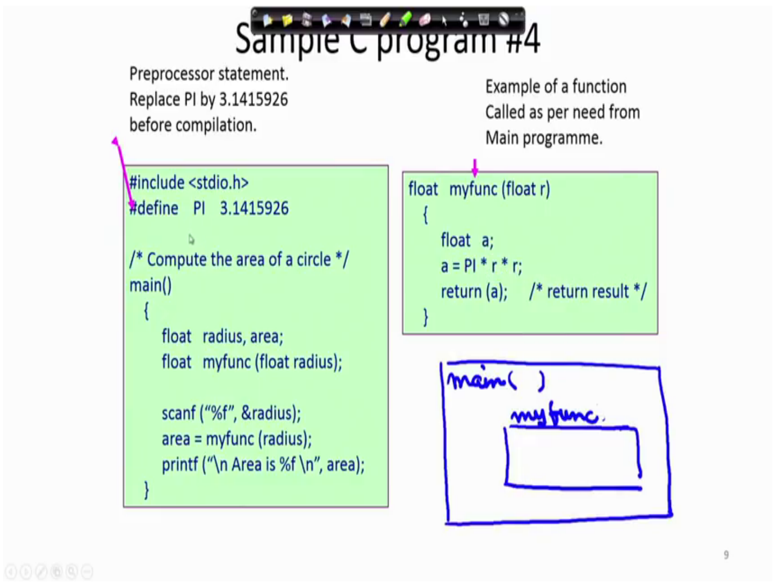
Sample C program #3

```
#include <stdio.h>

/* FIND THE LARGEST OF THREE NUMBERS */

main()
    {
        int   a, b, c;
        scanf ("%d %d %d", &a, &b, &c);
        if   ((a>b) && (a>c))     /* Composite condition check */
             printf ("\n Largest is %d", a);
        else
             if   (b>c)            /* Simple condition check */
                  printf ("\n Largest is %d", b);
             else
                  printf ("\n Largest is %d", c);
    }
```

Comments within /* .. */
Input statement for reading three variables from the keyboard
Conditional statement
8

So, we see the comments we have already mentioned, now you are coming to another program.

(Refer Slide Time: 27:32)



Sample C program #4

```
#include <stdio.h>
#define   PI   3.1415926

/* Compute the area of a circle */
main()
  {
     float  radius, area;
     float  myfunc (float radius);

     scanf ("%f", &radius);
     area = myfunc (radius);
     printf ("\n Area is %f \n", area);
  }
```

```
float  myfunc (float r)
  {
     float  a;
     a = PI * r * r;
     return (a);    /* return result */
  }
```

This is using what we earlier had said that a big machine can we divide into sub machines. So, big program this one is not a big program, but any programs can you broken down into different functions. Here is the main function this one is main let me draw it a little nicely. Here we have got a main function and inside there is another function whose name is my function I am sorry.

Now, let us look at this, now first here there is another new thing we are introducing and that you should understand that is define PI to be 3.1415926; that means, this verified in this program this PI will appear, there will replace it with this value 3.1415926, but inside the program I will not right PI, I am just defining it once for all and this means I am replacing pi by this value before the compilation is done. So, this is again a p processor statement.

We will continue with this example in the next lecture.