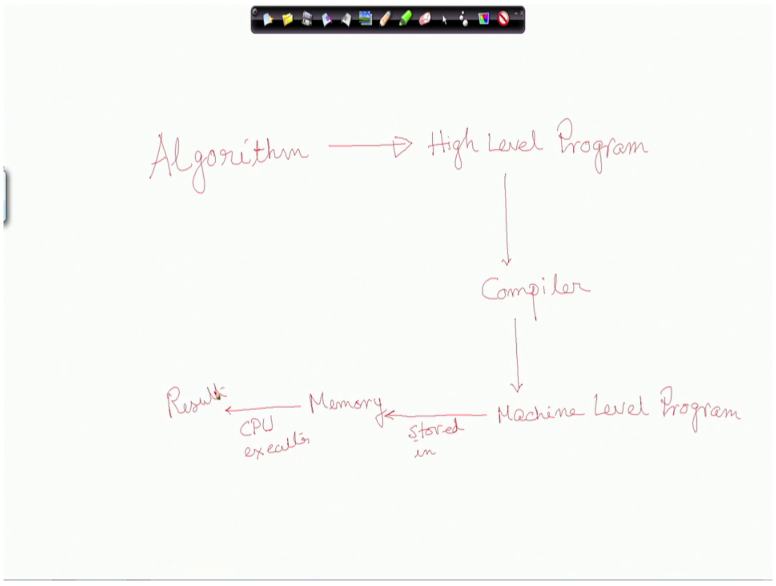
Problem Solving through Programming In C

Prof. Anupam Basu
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 05
Variables and Memory

In the earlier lecture we had seen that we start with an algorithm and convert that algorithm into some sort of high level program.

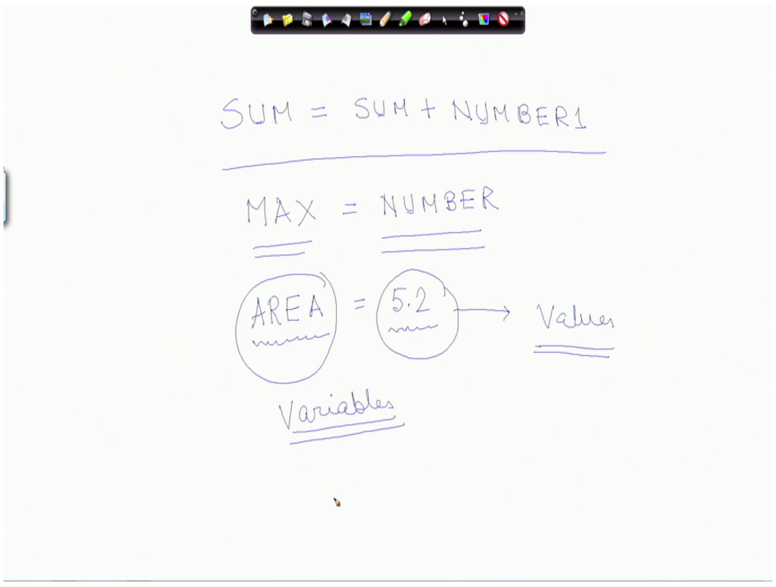
(Refer Slide Time: 00:22)



We convert it to some high level program and that high level program is fed to a compiler and the compiler prepares the machine level program. This machine level program is fed is stored in the memory of the computer and from this memory stored in the memory of the computer.

Now, from the memory the CPU execute set and we get the desired result that is the overall flow of the whole thing. Now we will now come back to some of the statements that we used in the earlier example programs will come back to that.

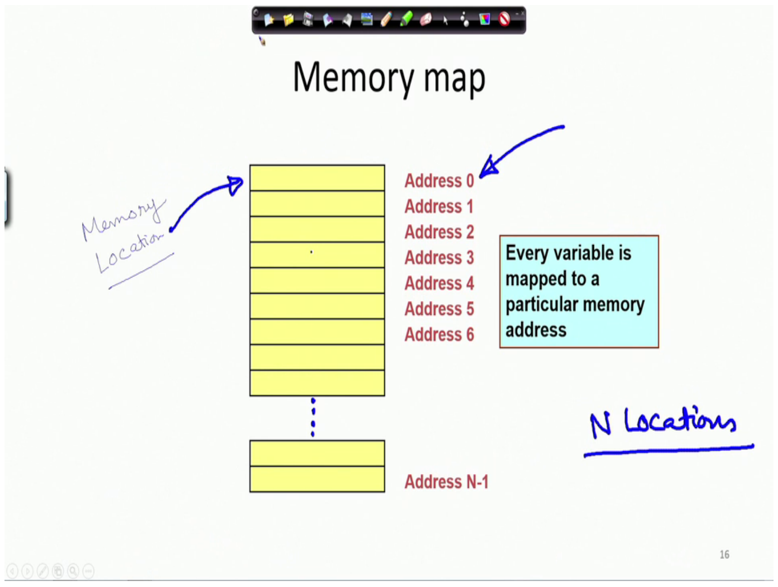
(Refer Slide Time: 02:38)



Say we had statements like SUM is equal to SUM plus number 1 right yesterday we accounted we encountered such statements SUM equal to SUM plus number 1 now also say for example, MAX assigned number 2 then the question is what are these things max number all right or for example, if I write AREA is assigned 5.2, what is this area and what is this 5.2 now this is some very fundamental concept that will be discussing today these are known as the variables and these are known as the values.

Now, we must be very clear before we start translating or writing a program in any high level language it is imperative to know very clearly what is a variable and what is a value. So, today's lecture will be devoted to explaining the difference between variables and values and what is the significance of these variables.

(Refer Slide Time: 04:43)



Memory map

Every variable is mapped to a particular memory address

Now, let us look at this diagram the memory can be considered to be a cvr number of racks in a book rack sort of thing or you can think of a number of a drawers in the chest of a drawers essentially is these are specific places as is being shown here, now each of this place is known as a memory location all right.

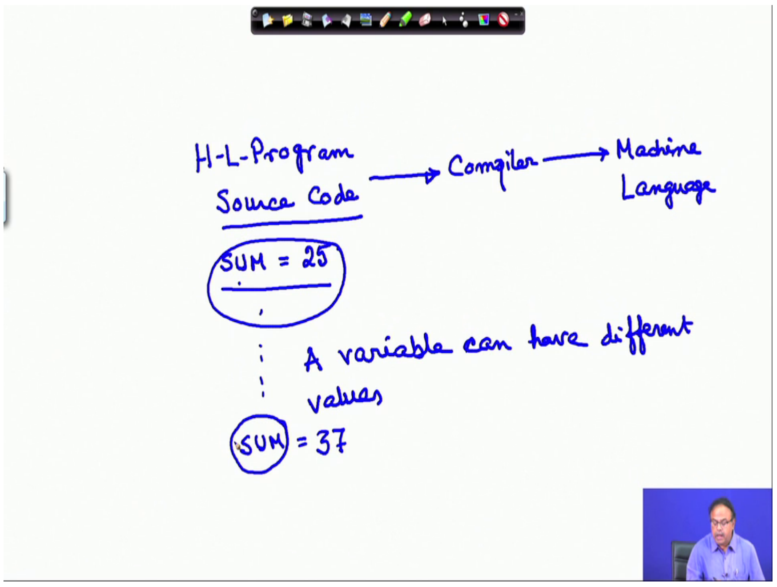
Student: (Refer Time: 05:40).

Now, here you can see that each of these locations have been marked with a particular address, these are the addresses and each address of every memory location. So, think of a scenario that in a locality there are a number of houses each of this house is a location, this is a location right, now I want to reach a particular house; that means, a particular location, how can I identify where I should go for that we need some address. So, similarly in the case of memory for every location there is an address, here we can we are showing 1, 2, 3, 4, 5, 6, 7, 8, 9, actually there are more locations which have not been shown.

We are showing N locations, N number of locations are being shown and each of these have got some address there is a peculiarity here the reason of that will be clear later let us for the time being accept that we are starting our journey with the number 0. The first address is address 0 and, the Nth address will be address N minus 1 because the second address will be address 1, the Nth address will be N minus 1.

Now recall that we had the scenario that we had the high level program.

(Refer Slide Time: 07:52)

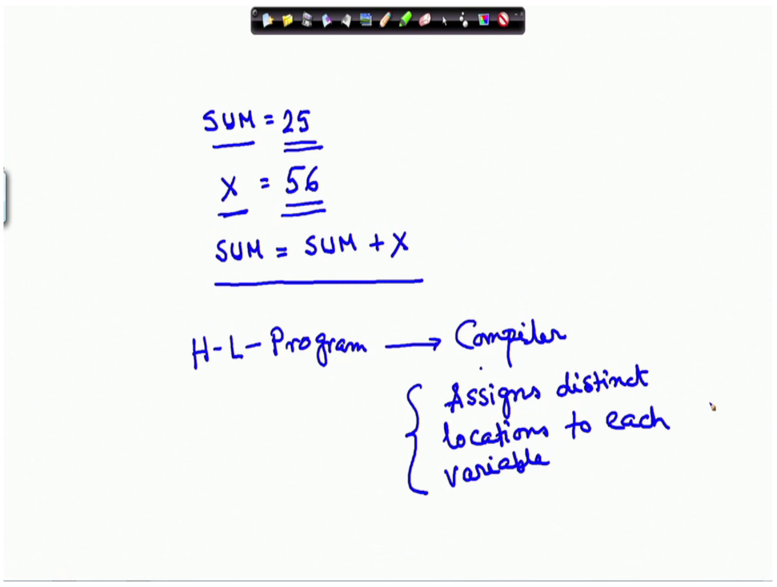


High level program or if you remember we call it also the source code and from the source code, the compiler converts them into a machine code machine language. Now in my high level source code I have a statement like SUM equals 25 say now when I say that SUM is equal to 25 the compiler will look at this and will assign these variable now this SUM is a variable means what it is a variable because it can be loaded with different values.

A variable can have different values for example, here I am making SUM equal to 25 after a while I can make SUM equal to 37 therefore, SUM is a variable and it can have different values right.

So, say in a program in a program I have got here SUM equals to 25.

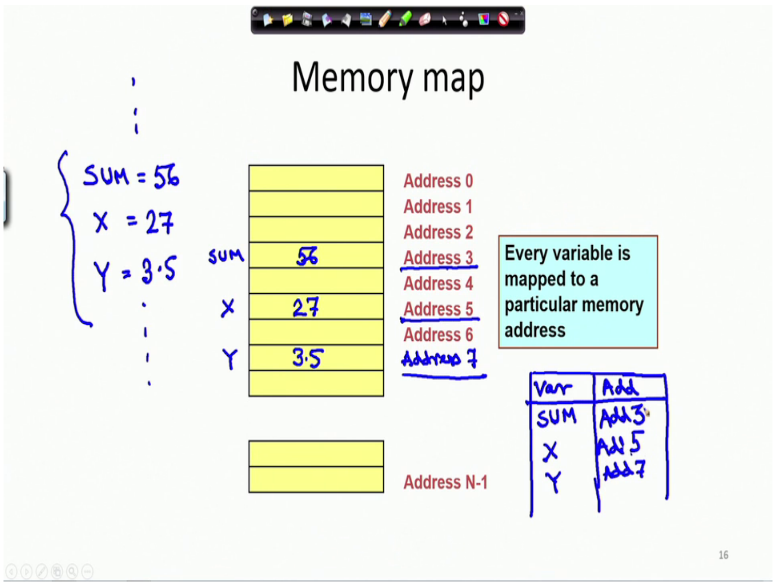
(Refer Slide Time: 09:33)


SUM = 25
X = 56
SUM = SUM + X
H-L-Program → Compiler
Assigns distinct locations to each variable

X equal to 56 and SUM equals to SUM plus X something of that sort, here how many variables do you see and how many values do you see we find that sum is a variable and X is a variable and these variables have been repeated here and what are the how many values do you see 25 is a value, 56 is another value. Now when the compiler takes the high level program and it looks at the compiler looks at all the variables and assigns or allocates distinct locations to each variable for the sake of simplicity let us say that it assigns distinct locations to each variables all right.

Sometimes advanced compilers also share variable share location, but that is not our concern right, therefore, let us come back to the slide once again you can see.

(Refer Slide Time: 11:20)



Memory map

Every variable is mapped to a particular memory address

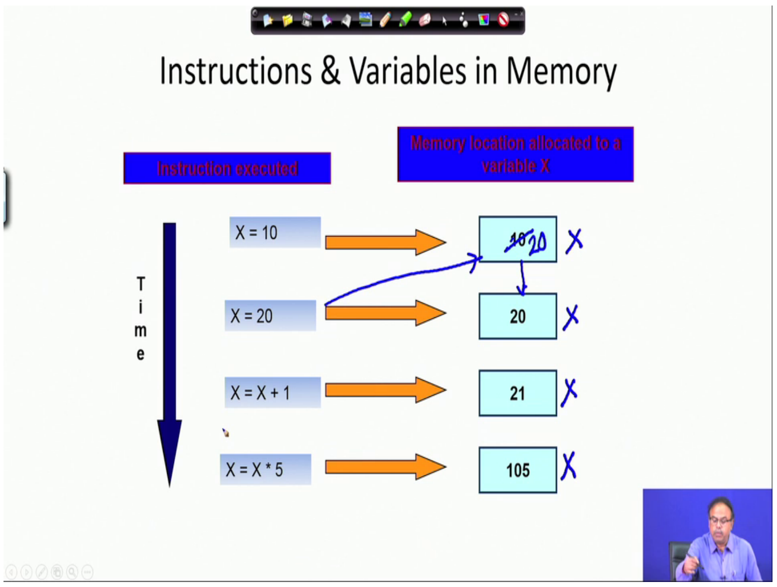
If I have a number of variables when every variable is mapped to a particular memory address for example, here if I write SUM equals 56, X equal to 27 and Y is equal to 3.5 the compiler will look at this piece of program and you will allocate and we will allocate some location to each of these variables maybe SUM can be located allocated here, X is allocated here, Y is allocated here say therefore, SUM can be identified now by the computer or the CPU when it will run the program it can look at whenever a sum appears in the program the compiler will convert it to the corresponding address of SUM. SUM has got the address 3, X has got the address 6 and Y has got the address 7, therefore, this thing is converted to a scenario where I am actually writing 56 into address 3.

So, as if 56 is being written here I will explain it later, X is being written here, 27 is being written here and 3.5 is being written here all right because while the computer executes the program it will take the data from the memory location only and the memory locations it no longer understands whatever name you gave to this particular variable, whether it is sum or any other name, but it is identified uniquely by the address that the compiler has given it.

The compiler can depending on the availability of the memory locations allocated different addresses to different variables once that is done there is a table for example, there is a table like this a record that is kept that you can think of that we have got the variables here the variable and the address. So, some table of that sort is prepared SUM is address 3, X is address 5 and Y is address 7.

Now, the CPU whenever it, the program in the machine language is converted in terms of these addresses this variables will no longer appear in the machine language, let us look at this in a little more detail.

(Refer Slide Time: 15:17)

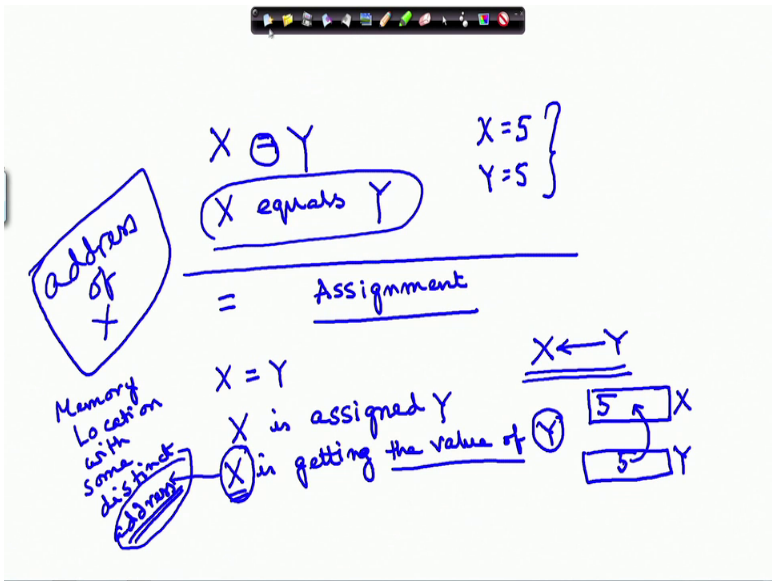


Instructions & Variables in Memory

Now, suppose look at this instruction this is a program now one thing to mention here is typically a program consists of a sequence of instructions that we know that one instruction after another will be executed. So, this is usually done in a sequence where is an exception from that sequence? We had seen in the flowcharts that whenever there is a loop I am repeating an instruction time and again, we have seen when we are trying to find the maximum of n numbers or average of n numbers we are repeating some instructions time and again, in that case we will go back to the earlier instruction otherwise one instruction is executed after another right.

As time passes we have got some instruction here not doing any meaningful computation say just to illustrate X equals 10, now here is another point what does this equal mean all right.

(Refer Slide Time: 16:48)



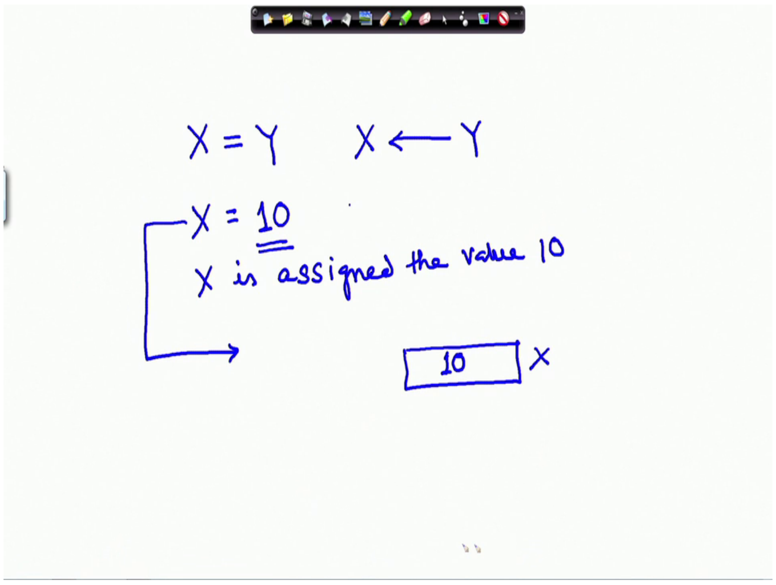
This is something that we should understand in our school we often do say X equal to Y that means X, the value of X is equal to the value of Y. If X is 5, then Y is 5, but here this symbol whenever we are using now is this symbol actually means assignment now this is something which you should understand and this is fundamental to any programming language that we will use. When I write X here I am writing X this symbol Y I will read it as X is assigned Y what does it mean X is getting the value of Y.

Now often is therefore, in order to avoid this confusion we write it in this way that the value of Y is assigned to X now here is something to really understand what is X, X is it a value no, X is a variable therefore, X is a memory location the compiler has allocated some memory location with a specific address, this X is nothing, but a memory location with some distinct address is it readable, memory location with some distinct address that is X and Y is also a memory location with distinct address, but this assignment means it is getting the value that is in Y.

Let us think of like this that here is a memory location that has that is X. So, X when I am writing X that is means that it is actually the address of X, that has been address of this memory location and Y is also memory location and when I am writing Y this variable name Y just for ease of our understanding otherwise I would have to write the address here and the address in a computer system in machine language will be a string of binary bits I mean a bits bit string, that will be really complicated. So, for the sake of understanding we are just writing X and Y, but you must understand that this X or this Y means the address of X and who has allocated the address of X or the address of Y the compiler. Now when I am doing assignment this X is assigned the value of Y suppose Y had 5 all right the value was 5 now this 5 is then written over here and this also becomes 5 this is the meaning of assignment it is not exactly meaning equal to Y essentially after this what happened is the value of X is actual equal to the value of Y, but this sign typically means assignment in order to show this equality we have got other symbols will come across that later.

Let us move to this, here we are executing an instruction one instruction after another. So, here in the first instruction we find that X has been assigned the value 10 is not say once again let me come here.

(Refer Slide Time: 22:05)

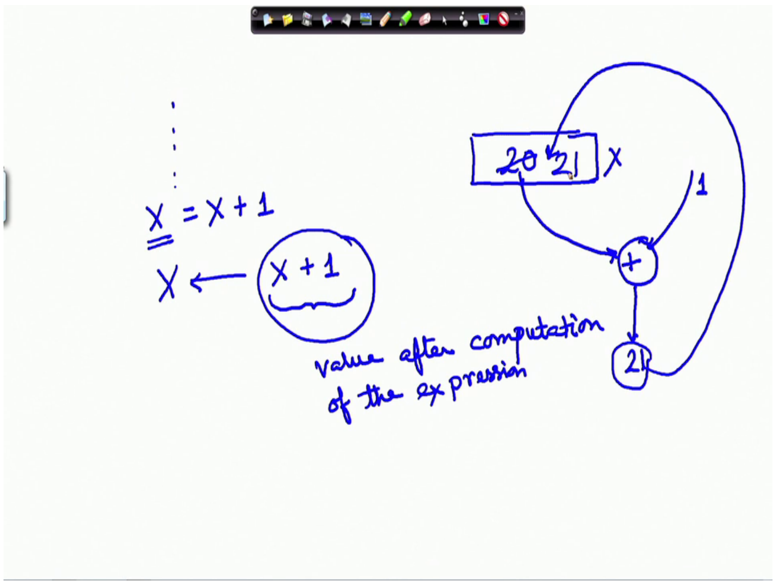


This is this means X is assigned the value of Y if I write this X assigned 10 that means, X is assigned the value 10 because here I had some value in Y whatever it is and that was being transferred to X, but here it is not this side the right hand side is no longer a variable, but a value right the right hand side is a value therefore, X is a memory location.

For this example X is a memory location which is having the value 10 after execution of this instruction this will be the situation in the memory after the execution of this instruction right. Here let us now you will understand as I assign 10 to X and this is a same memory location this is a memory location for X the same memory location see how the picture is changing, here that memory location X is getting the value 10 on this side I am showing the memory location allocated to the variable X. So, 10 next step I assign 20, this location gets 20.

This 10 that was there same location 10 was there 10 is being over written by 20 here. So, when this is being executed then this 10 is being over written and 20 is being written, I am getting here 20. Now X has got 20 what is X now X is 20 now I do another arithmetic operation here what how do you explain this X assigned X plus 1.

(Refer Slide Time: 24:44)

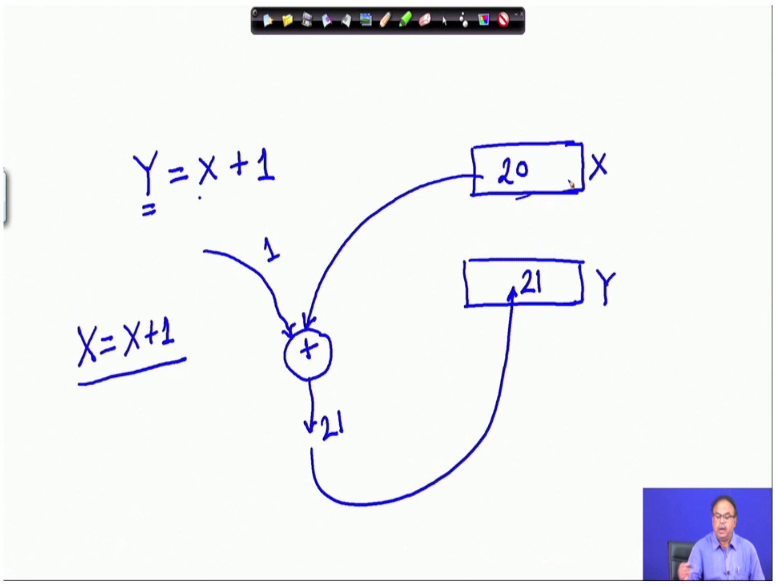


Let us go back one moment to this X was this is X it was 20 and I have come across an instruction, which says X equal to X plus 1 what does it mean? It means assign to X what? X plus 1; that means, whatever value you get after computation of the expression will be assigned to X. So, how will it be done? Right now what is the value of X 20, the CPU will have doing some addition operation, will it read this value 20 here and some incrementation by 1.

So, that these 2 will be added the value from the location X will be taken it will be added with 1 or incremented with 1. So, here I will be getting after addition I will be getting 21, now this assignment means whatever value you get after computation will be assigned to this. Why this because on the left hand side I have got the same location X, this will be changed to 21.

Now suppose if instead I had the situation like this.

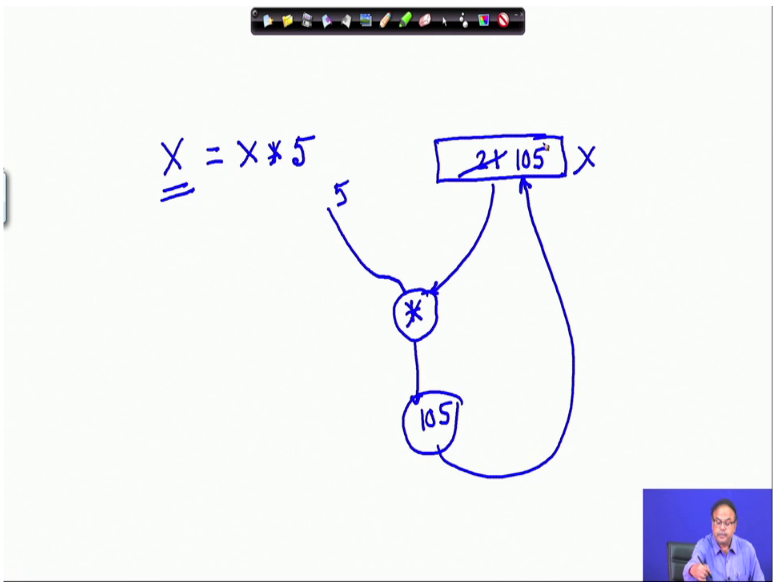
(Refer Slide Time: 26:32)



X at 20 and I write Y assigned X plus 1 what does it mean? It means that this Y is another memory location and what is been told over here, you take X take the content of X and, you will take 20, and have 1 added add them you will get 21 and assign it to Y in that case you see the picture, the main content of the memory location the value of the memory location X remains unchanged and another location gets the value 21 right unlike the earlier case. When we had written X assigned X plus 1 when I do that then in this case this value will be overwritten this 20 will be over written, but in this case it is not being over written it is remaining all right.

Now, let us go back to the slide here, when I therefore, you understand X was 20 assign 20, then next step X is assigned X plus 1, it becomes 21 the same location X is becoming 21, now I am doing X assigned X multiplied by 5.

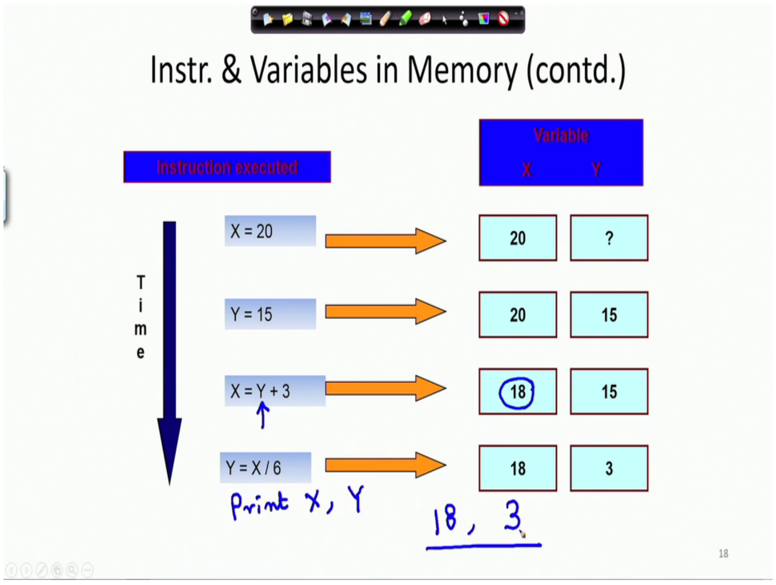
(Refer Slide Time: 28:22)



So, again what I am doing? I am the same thing X assigned X times 5, I had X and X was 21 here I took that value, and this time instead of adding the CPU is doing some multiplication and with this constant value 5 and is getting the value 105. Now this 105 is again written back to X, it is going over there and it will over write this 21 and we will write 105.

Therefore here we will have the result as 105.

(Refer Slide Time: 29:05)



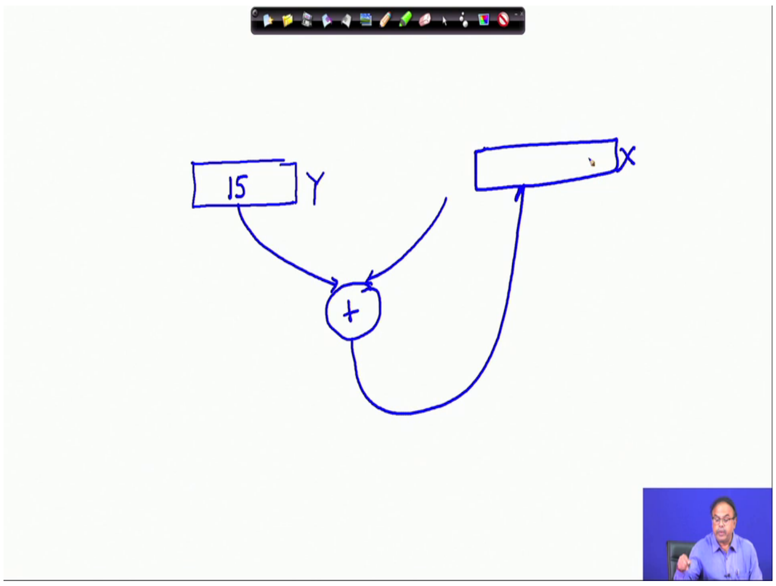
Instr. & Variables in Memory (contd.)

Instruction executed	Variable X	Variable Y
X = 20	20	?
Y = 15	20	15
X = Y + 3	18	15
Y = X / 6	18	3

I hope you have been able to understand the difference between the variables and values here, and the variables the fact that the variables are nothing, but memory locations. I will continue with this in the next class, quickly let us have a look at this. Let us revise here another situation another program segment the part of the program. X is assigned 20, Y is assigned 15, next step Y is assigned 15, X is assigned Y plus 3, Y is assigned X divided by 6 now; obviously, since there are 2 variables in this program, there will be 2 memory locations assigned.

Now, here in the first instruction X is being assigned 20 therefore, this location has got 20 written in it what is there in Y I do not know; anything that was earlier from the beginning was remaining that is there I do not care. Next step I am assigning 15 to Y therefore, now X remains as it is 20 and Y becomes 15. Now what am I doing in the third step? I am doing X assigned Y plus 3 just as I had explained in the last slide just now that I will be actually taking the content of Y.

(Refer Slide Time: 31:13)



Whatever Y was it was 15 or something I take it and add something to this and store it back to X all right that is what I am doing right.

X assigned Y plus 3, Y was 15, that is taken 3 is added to that the result is 18 and where is that loaded, that it loaded into X and, it becomes 18 and what happens to Y? Y is not disturbed because here at this point Y has just been read it is not been written into Y has been just read, I read from this location I did not disturb it I disturbed x. So, I added 3 and made it 18 here what I am doing? I am now disturbing Y how? I am reading the value of X dividing it by 6, 18 divided by 6 will be 3 I am writing that to Y therefore, now I am disturbing Y and Y has got a new value, now, my result if I had written here print X and Y then my result would be 18 and 3 right.

So, that is how the variables are handled by a programme and I tried to explain to you the difference between variables and values, you please look at the pseudo code of the examples that we did in the last lecture and see what are the variables and what are the values and try to draw a diagram as I have done today just to have clearer conception for that.