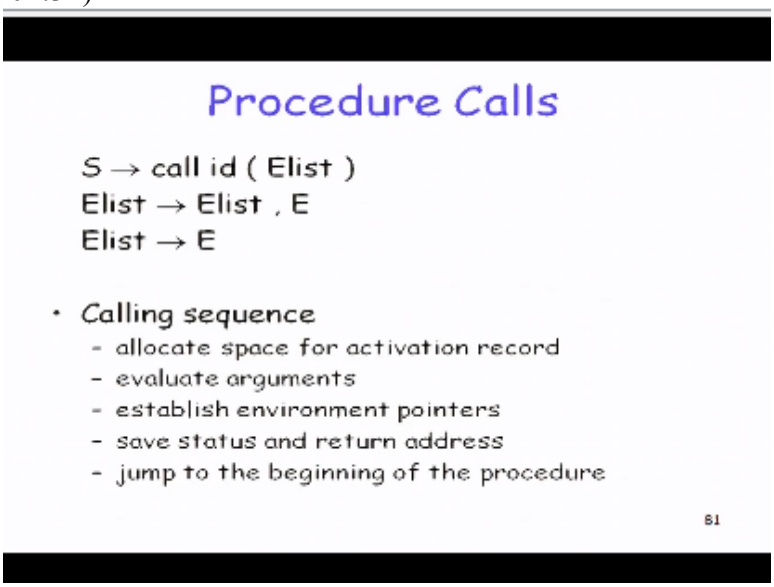
Indian Institute of Technology
Kanpur
NP-TEL
Nation Programme
On
Enhanced Learning
Course Title
Compiler Design
Lecture – 25

by…
Prof. S.K. Aggarwal.
Dept. of Computer and Engineering.

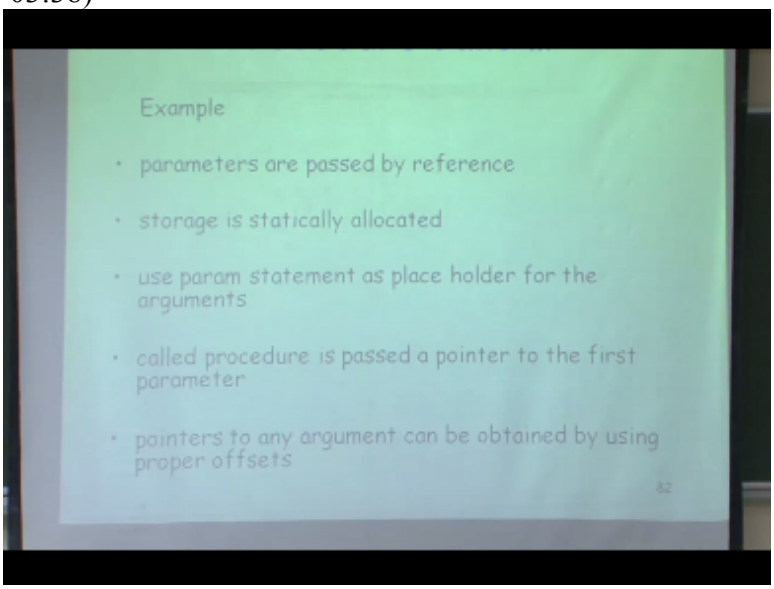
Good morning. So let is start about the discussion on distract to in the next half you finished all the straight line to traditional and terms the equations and all the conclusion but we still do not flow had to take this is the code and then pushing confutes the position. Now pussy cats have conflates of facial functionality and therefore the equity is slightly different in the two parts is. One is the paddle part which already has picked and part and all the arguments are be pass. In the second part is important were for each procedure need to generate the traditional code which will executed a front time a procedure code okay. Now one buying a programme I do not know whether procedure to the promoter. So it let us we go to the things. So let us look at both the parts were, so the first part is that when you talking about so see here,

(Refer Slide Time: 01:32)

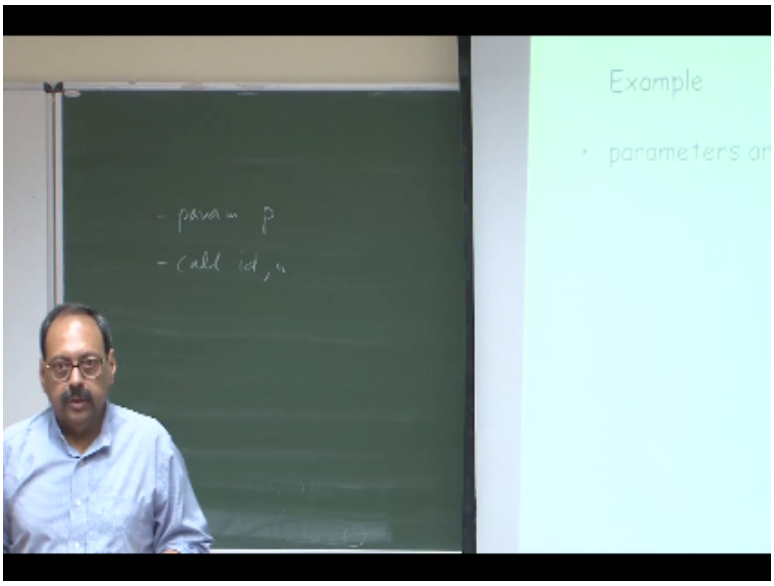


What we have is that if I look at the syntax of the procedure calls what we saying in that these are identification associate in the name. so is not time to I am saying calls and procedure when argument so what is my argument list, argument list can be request is to be find us list of arguments were arguments is nothing but E list forwarded by E and dare this is the base value.

So this way I can so I am not worried about whether my arguments are expected or not because it could be a single variable or an expression depending upon how this works assume that I am talking about general expression.
But most of the time but basically what activation record is say that one spawning which causes certain data has to be created contract and what was the data I need to have the local variables I need to have space for the return value and sometimes they may be resources which are associated with the video and therefore I need to have some space where all this information will be stored and then we also have two email accounts because of something like saying all function.
(Refer Slide Time: 03:38)



With X + 1 so X to be many so I need to evaluate all the arguments then I must establish a point this basically environment pointers of the ones which are going to capture the spoke information. So if I have for example Nested procedures and functions if I have a function f1 at this level and within this I have a net and then after this then we might use what we have two instructions.
(Refer Slide Time: 05:30)

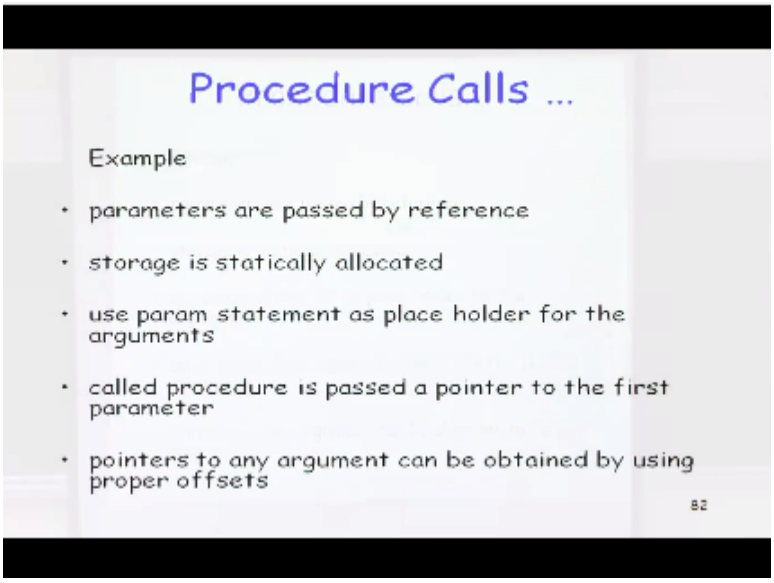


One was a parameter that I said that T is a parameter advance another was said al righty with some ID so these are the only two things ahead and rest of the things are going to be part of so this part is the part of all the evaluation and this is really the part when I say that I need to create all these parameters and then I need to jump so this is so these are the things we do that we look at parameters.

We need to know where the parameters are being passed by reference or any other mechanism. We look at what are the parameter passing mechanism so when I move the procedure I need to low water the parameter passing the head inside and passed by reference is going to be little more competent or by value and therefore I need to go into details of more details of storage then I need to worry about that I am going to the storage is required.

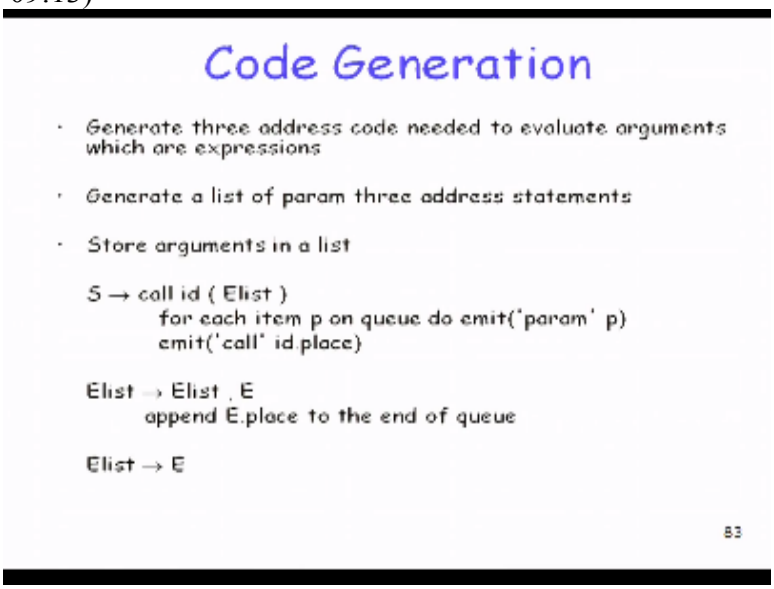
So for example in C is procedure allocations or Adam okay so if it is static what is going to happen is that if I have a procedure P suppose I have a function at one okay. So this is the elevation I allocate where one on the stack now suppose that one also itself so these are the kinds of questions we have to ask when we are dealing with procedure boss that weather parameters are passed by reference or not but the storage is statically allocated dynamically allocated then I use all these arguments like,

(Refer Slide Time: 08:37)



And we say that what are the placeholders for the arguments I have so these are really the placeholders for actual parameters then I all you see here and we say that here. We pass pointer to the first argument and say that I had an argument and pointer to any argument adopted by this using the proper office because if I just pass the people to the first pointer to the first argument.

(Refer Slide Time: 09:13)



So when I start doing code generation and this is for generation only for this part and not for rest for the things. Which is initialization understand it so this part is actually fairly straightforward that I want to get rid Tiraspol and militarists for means that first for this if I have an invocation like this first time we say that has or for evaluation of X +Y and if once this is negative this will be some location which is associated with this expression.

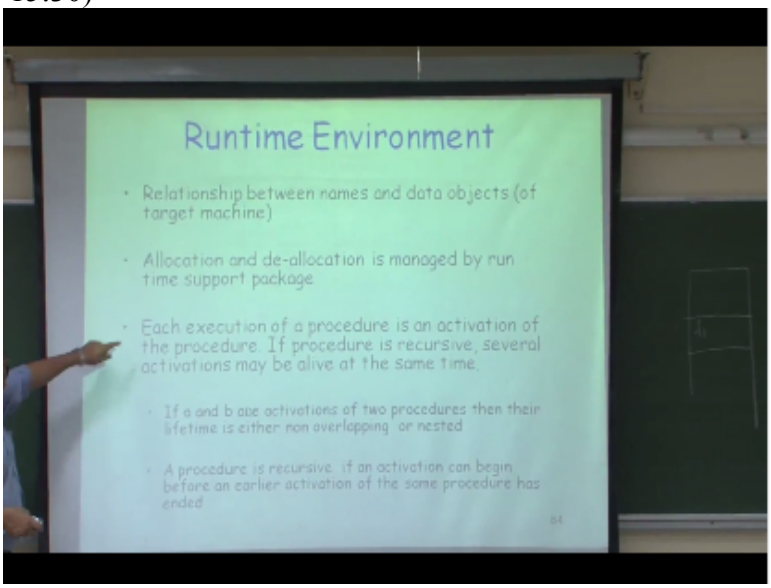
And then I will say that I have parameter P we are what his oops these thoughts of X + 1 so this way I have all the spaces for all the arguments and then I'm going to generate a list of which is the statement and then store arguments in the list so basically all these parameters are the arguments and then this each item which is so this is actually the much simpler part so there is nothing less than this I am saying that every time you have expression here then just to eat less and this one when I see the first are too much I am just saying that I initialize.
So this part as I said is pretty straightforward all I am going to do at the end of this activities that I have sold for expression you magician okay and this will then after this I create a list and then I generate from the list these instructions sequence of these instructions and finally this so that really is not a complex part so for example now so let us take an example suppose I say that I want to invoke F one way to say X + D see if this is what I want to invoke what is the kind of what I generate for this I say that T 1 is X +Y, E 2 is a + B or C I do not require a temporary variable.
That is going to be right there and then I say instructions like parameter t 1c X so these are the three sections I will generate from this particular Q which has been created of arguments and then I will say after this all f1 and water like paths here either I can say that pointer to this and three arguments or if this list is there then I just pass this pointer to the list and so this is really the formulation for as far as this part is concerned but here we all I said there were five six steps and this actually corresponds to the after the first few steps are not even.
So what are those first three steps in first few steps are actually part of what is known as so what is going to happen is let us understand what is happening in procedure invocation invite you to see this out slightly differently and our special cases one that I do not know whether a procedure or function is going to get involved second thing I do not know whether it will require space so normally what is done is that no space is allocated third thing that happens is that for all the variables if you look at lower variance a new space allocation.
Once but for all the local variables of procedure and they have to do multiple locations depending upon how many times there we go n in which environment every so suppose a procedure is invoked recursively and n times recursion goes deep end times then I have to enter the patients of all those variables therefore I do all that but I generate sufficient puts so that is that poor gets executed wherever the program is getting is so that is what my turn that environment is so runtime environment basically code generation strategy for position function.
And the second issue which will come as part of runtime environment is that at some point of time I will also do dynamic allocation of memory how many languages do not support either

means but when in language is new. So when you have dynamic allocation of memory how do i allocate space at compile time I cannot, so when you do a memo for example in seeing as we say that allocate 512 bytes somebody has to then allocate that specially it must be in your office space if it is not in your closet space in front people accessing.

So two things have to be managed by runtime environment one so see your function calls yourself a runtime. In second dynamic allocation of the media so this is the kind of coordination strategy we have so this is mirror time environment is basically giving me a relationship between name and it outfit of the code averages what does that mean this is now saying that suppose I say this variable X then oh well this variable X this variable X will be somewhere in memory and different executions of this program will put this variable at different cases.

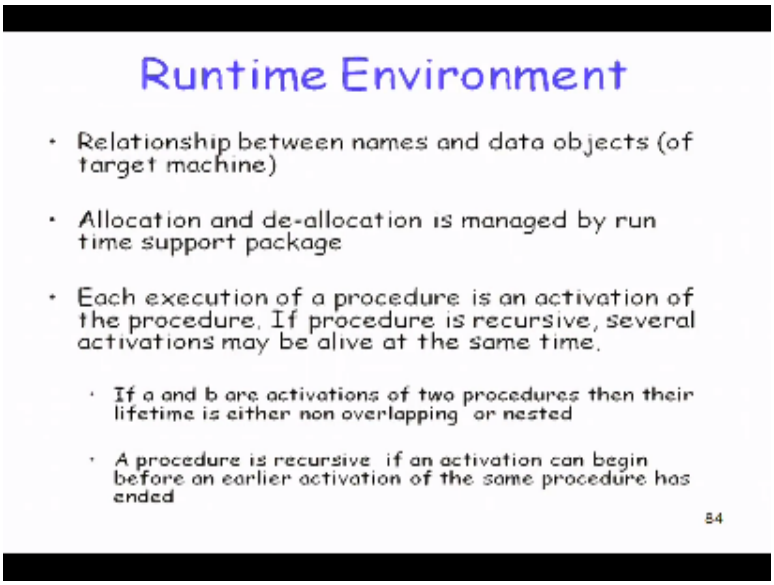
So runtime environment we give me this managing at every time I want to access X I should go through simple table and simple table will only give me an offset but not the base address and then I actually go through my page tables and access this every time I go to this data into memory this mapping we teach in this runtime environment is the one that is going to give me this relationship between the name and the date object of the target machine and then a location in below when you see the space that time is all this and then what you say now is that whenever I execute a program.

(Refer Slide Time: 15:30)



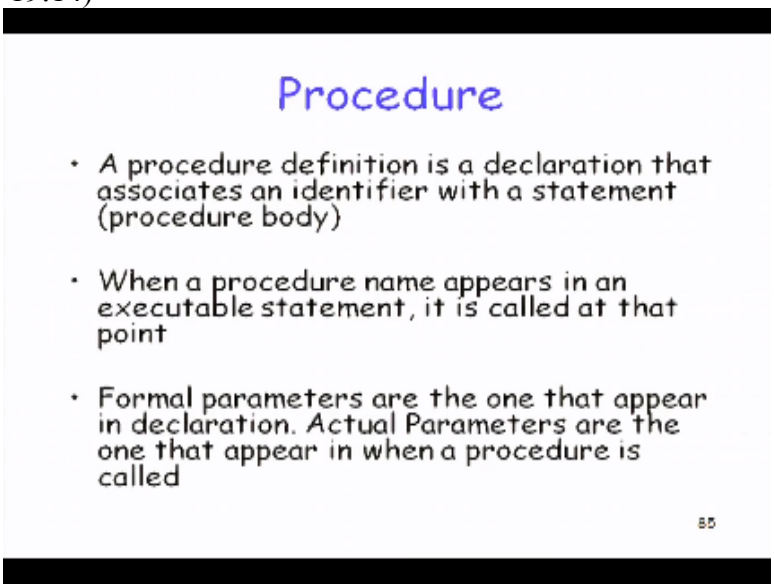
We say that or whenever I lose a function of Lucia we say that I am activate I am actually putting in activations on the stack or I am saying that a new activation corresponding to that particular is created what that means is that if is recursive procedure can have several activation switch our life so for example I just gave you this example that suppose I this confusion inside if I have call

it 1. I depend on the conditions and the argument is that the first time some activation of f1 is going to be on the stack and then next time it is called activation.
And I keep on also the space for local variables every time we will have to be replicated and then depending upon again which activation is currently being executed values of the local variables every time will rusticated and then depending upon again which activation is currently be educated values of the local will keep changing in different stacks so if I am suppose this is the current execution and there is some local variable X whose value is tending or that is not confuse with argument suppose there is some local variable said here whose value is changing in this particular invocation it will not change him but it will only change in this particular activity.
So this needs to be managed by time so what we say is that if peruses because if there are several activations at the same time each having its own sets of vehicles and set of arguments and set of machines dates and now we say that if they are true activations in V who activations are different of the same procedure so any put the activations of the same procedure or they could be activations of two different procedures does not matter so beyond this point then.
I start looking at set I do not really distinguish between set who see here me what I say is that each time a procedure is in book an activation is speeded we will be looking at activations so now we say that is NVR two activations then either their lifetime is going to be non-overlapping or it is going to be next what does that mean what that means is that either A will start here some light times I say that lifetime of A starts here and ends here and then B starts here and ends there or it could be next and that means lifetime of a starts.
And then life and of B starts here and ends and a continues I will not have a situation where I say that A starts here and then we start somewhere and then a finishes and these countries what that means is that he was invoked by a but these continuing an definitions that is not possible.
(Refer Slide Time: 18:47)



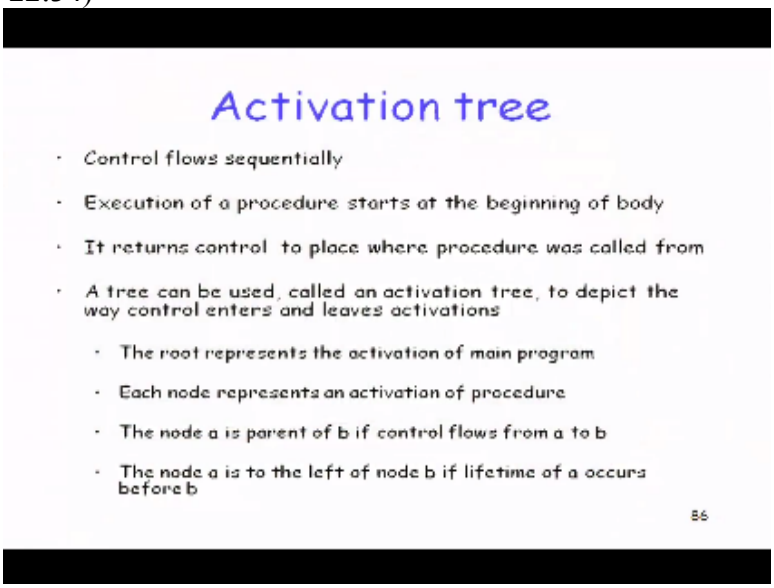
So only two situations can occur either you have non-overlapping light that is good like these kind of so activations of truth position they like image either non-overlapping or business head to see any leadership if you can have activation.

(Refer Slide Time: 19:14)



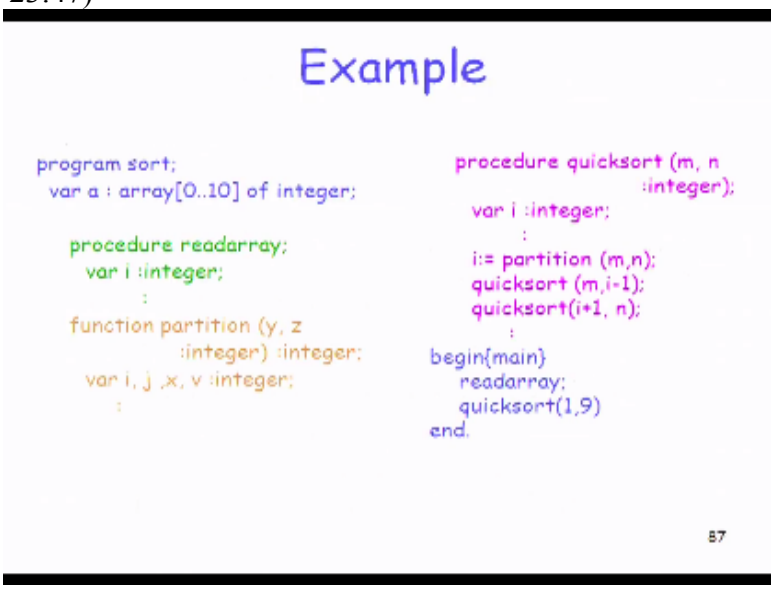
So again now coming to the definition of the procedure once again our position is going to be not slightly different to what we are saying this procedure is nothing but A declaration so procedure is declaration of the name which returns the value of certain type and which takes a list of arguments what about the body of the procedure that is the statement which is associated to the procedure that needs to be executed so header has to be treated differently than the body now we have already seen how to do for definition for the body part okay.

So that is really of no issue anymore the only issue is of the headed making that needs to be treated carefully every so procedure definition is a declaration that is to associate an identifier with the statement it is the procedure bottle so when I look at this F X Y I am only nearly interested in this that what is the return type of F 1 what is the procedure name on is the argument list and what is the type of each what about the body part body part you know is just going to be report and you know how to handle that okay we already could put upon okay.
And whenever a procedure will appear in exsiccate statement and it will call it. So when I say that call F1 this is I am calling it and this is variable statement okay. Then I also have this example here so this is so the roof corresponds to the invocation of the main program and then tilt internal node corresponds to invocation of a function of C here and what is the leaf nodes each node is the one which does not evoke any other function just internal is the one that is corresponding to invocation of another procedure or function.
Now what Pope edges are telling you that which function or procedure is invoking the extended part of this but in activation when we talk of activation we will also talk about definite right series so we will say that in this order if I have activation so institution of procedure starts at the beginning of the body.
(Refer Slide Time: 22:34)

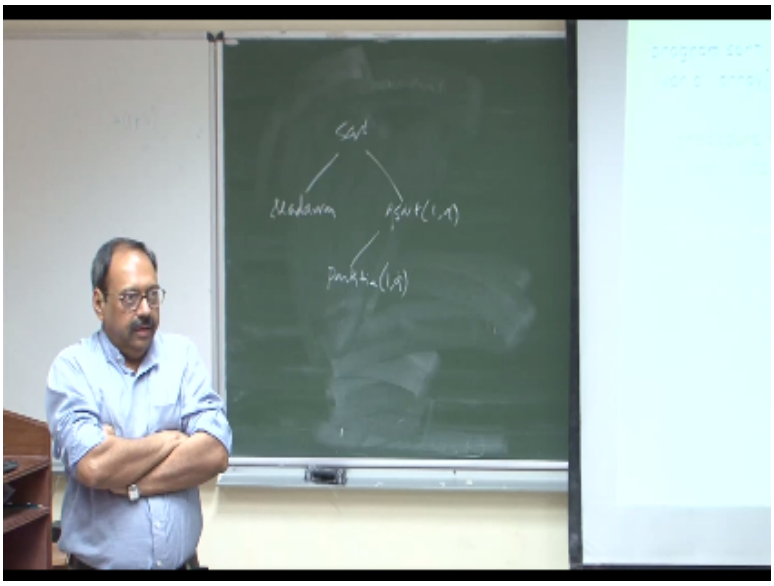


And dossier was or and T is the one which is used in this situation salute represent activation of the game tree each node represents activation of opposition and age but now I put another condition which says that if node is to the left of B then the lifetime of a so if I have a situation like this what this means is that corresponding to this suppose this is where my invocation of Main and then this is like so this way I can capture the full information about the control flow in

my program in which order my Posse over the body move it. So I am not going into the body of the procedure but I mean means the still together call graphs in the headers the activities it so let us take an example.
(Refer Slide Time: 23:47)

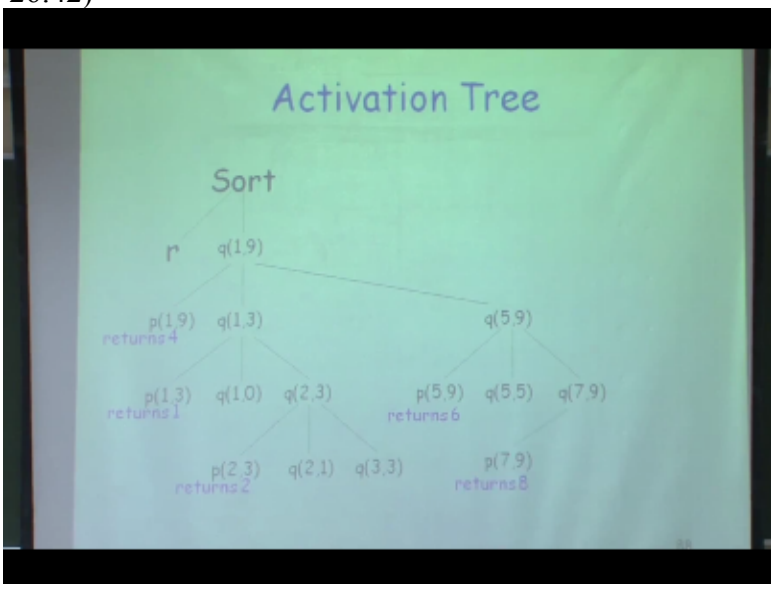


And try to create an activation key for this so here is some port then we say that I have this thought program and then this program we have seen earlier in pick sort we also looked at table for this so we have this senior beat away then I have a function which is partitioning which is taking two arguments Y and Z of type India it has its own local variable and then again I have a function quick sort and did not fix sort I thought partition which turns the values and I recursively call to it sorted in the main program I first initialize my array and then on how can we create an activation key for this program join definition.
(Refer Slide Time: 24:55)



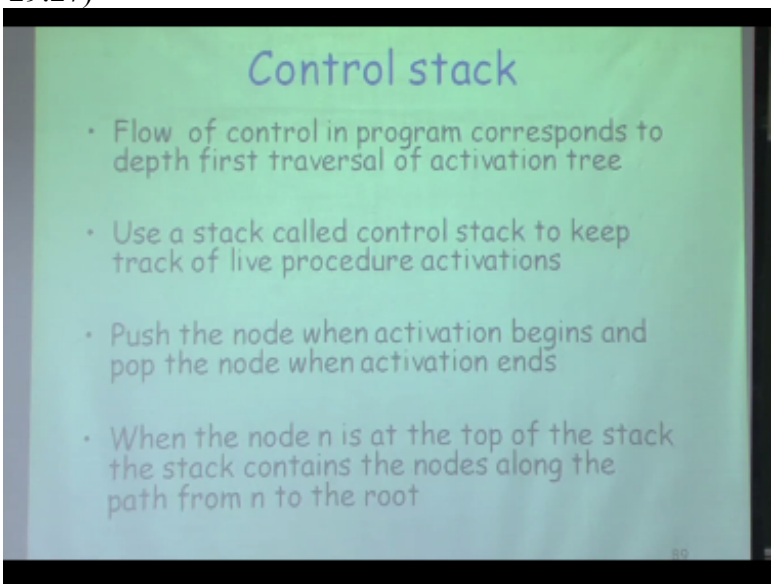
So it is clear that the first thing will be to fill with the sort and sort is calling read array and after read array finishes then I am calling with spot one line and when I go to fix our and what do I do I come on partition so this false partition with one of the next note I go pick solve it so assuming few things that some data is being returned this is how the activation key may look.

(Refer Slide Time: 26:42)



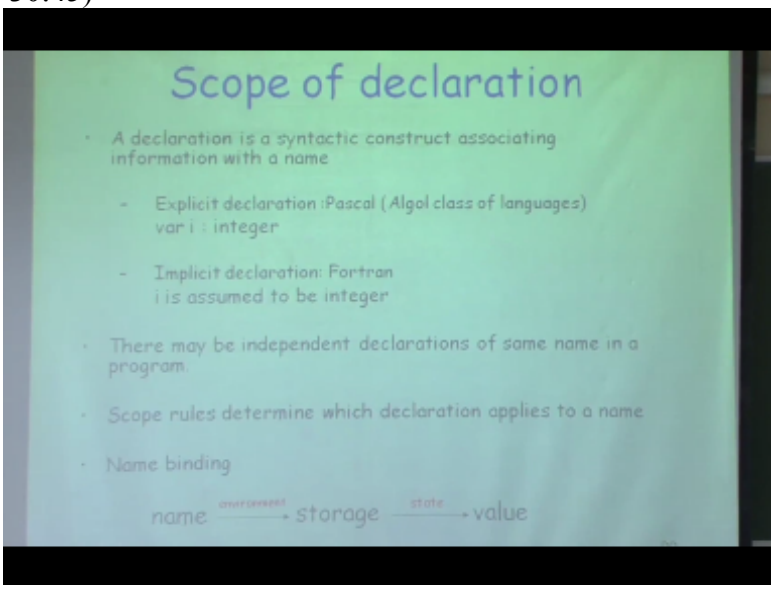
But this is only one of the instances of activation three or different data set excavation would be different that is first of all redirect and then I quick sort is all partition and now assuming that partition returns value but that is the property of the data. So assuming that partition returns that you for then I am going to convict sort one and then quick sort one is with the fall partition 1 3 which again let us do that it returns value 1 and you want in return I am going to fixed sort 1/9 and then I am going to fall to X or partition.

And so looking for party canceled so this is how the goal made in this particular theme. So first thing you have to remember here is that this activation tree is corresponding to that programme foe the dataset. Second thing is that at some point of time you will realize that I don't have to remember the woods now let us look at some random node in this suppose I say that this is the current active procedure that an X equal to each partition five nine if I say that there is a current activation procedure okay then from this definition and I created.
I set that let us begin handwriting what are the procedures which have already been executed and what are the procedures which are which I'm not meaning and what are the procedures currently we get stupid by looking at this tree in this particular data set if I say this is the currently age will be procedure what are the procedures have been which are finished execution so actually all this if you see this all this has finished execution this part is liquid glue but these are the current technique procedures.
So if I just take the path from root procedure already in active execution of that is T so I will only be interest rate therefore in going my current path and knowing their finished sequence and therefore I do not need to remember the whole tree I only need to remember that is what is active quick sort one line is active so finding that we had partitions and for this obviously the structure to do this.
(Refer Slide Time: 29:27)



So what we can do is I can be inter step and every time an activation is scored an activation is created I push it on the stack every time it finishes. I pop it from the stack so if I step therefore look at this activation three and say that this is on top of stack then I do not media mean that below this is of the stack you know that. So what we are using is control stack which says that

flow of control in the program corresponds to really this depth-first traversal of the activation tree and we use the stack for the whole stack of only they like to see is not all the to see.
So we are only interested in the procedures which are currently light and I keep them on the stack and then we say that wherever a load is an activation begins we push the nodes from the stack and after it finishes notice at the top after the finishes then I pop it from the stack and node n is a powerful stack the stack contains the nodes which are and all the paths from so these are the only activation sighs beautiful and not everything else.
(Refer Slide Time: 30:45)



So continue on this information now let us look at that halfway capture spoke information scope is some tasty to eat changing because of runtime bindings so if I say that this is there is some current position which is acting when I was looking at variables within that then binding of those measurements will be determined by the scope Pascal okay. Now if I look at scope rules there is something called lexical scoping of syntactic spoken language say that is going to be done so we look at both static scoping for lexical scoping as well as that is looking.
So let us start this discussion with assuming that I have lexical of static scoping so declaration is syntactic construct which associates information to the name so first thing that happens is to take a broad class of languages all these are going to have an explicit information opening and therefore a time associated with this so you will have something like a variable which would be a keyword in this language and on a pile of name of an identifier and the type name all you have language is like for that okay where you can have explicit declarations.
But now I am not dealing with languages which are interpreted so their languages which are interpreted energy again have implicit techniques okay but since you are not dealing with in

languages we are only my language at this point of time therefore I have picked up this example if you look at languages like they say and - and so on there are also these relations are explicit but then there are interpreted so we leave that for the time because then that includes something else.

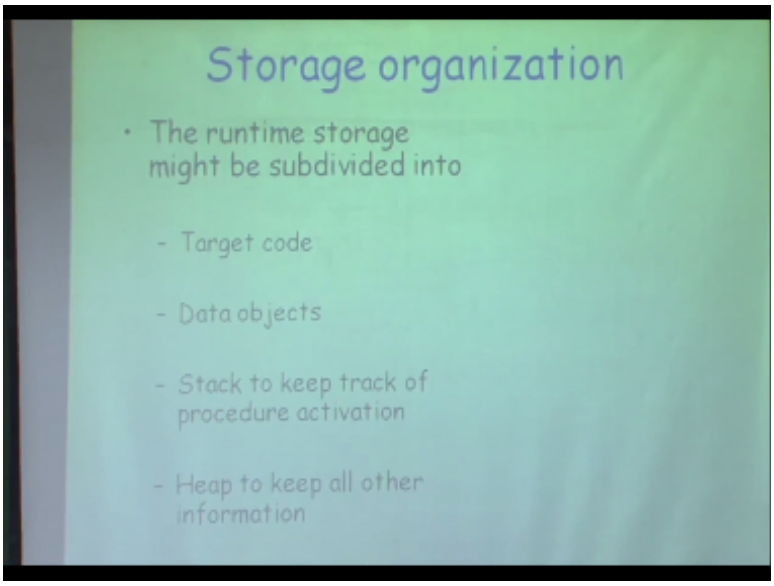
So there what happens is that although declaration is implicit but there is some assumption which is made if so assumption for example portion matrix I J P and I know or any variable starting with these letters they are in deist and anyway they will starting with any other letter than these six is bigger unless you explicitly detail and in fact most of the time you will notice that when people write programs this tradition for some reason continues because when the initial Fortran programs are written I pass the first available everywhere I want to end with this provision.

So even now when you write something in C or P will still say for I 1 to n we have I give me a fluting this variable most of the time people will those who are in software engineering and I worry about funny because no program will say that your counter name should be meaningful name but people continue to use I this we write will be totally node script name okay so I is a typical nothing like this like quarters and then we need independent declarations of the same naming program so what can happen is that same variables X can occur at multiple places in the same program for two different spoke is different types.

So I need to worry about has compiler writer how do I associated this variable with the I scope so if I just say X. X is meaning unless I say so there will be independent declarations of the same name a program that is because of the spoke truth and spoke rules are the ones which determine which declaration is so we need to know this and how does link binding happen some fight time in run time that you have something like a name and a storage Association now how do I find out which name associates with exfoliation.

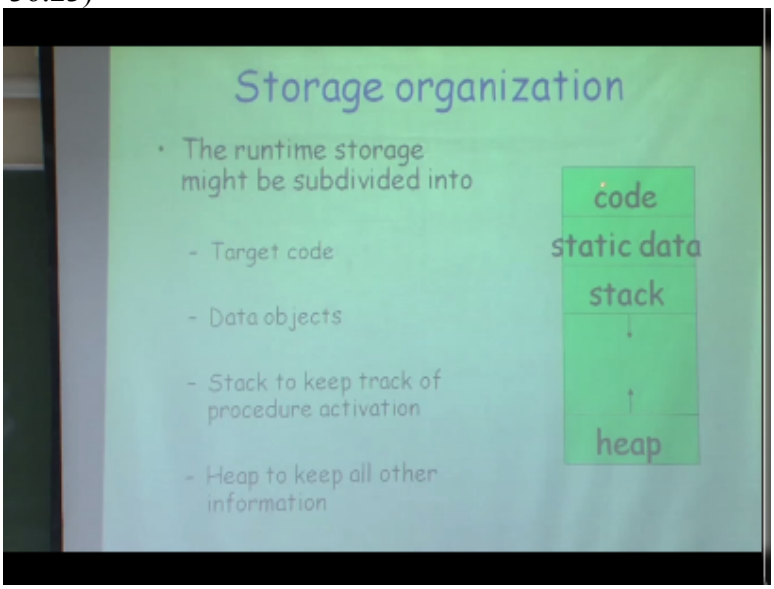
That is scope but what is the value associated with that particular storage that is the state of the program so this finally comes from the environment and we are only interested in this point it so we need to see how basically spooking rules to make sure that I am always able to find the right binding once I have located a space to this then I can always actually go to this particular symbol table find out at this okay so this is very important that we are able to for each name we are able to find environments. With it so that I can write this you want this figure it is time to defeat yes okay so let us look at now when I say that I am now filled in space for a program.

(Refer Slide Time: 35:52)



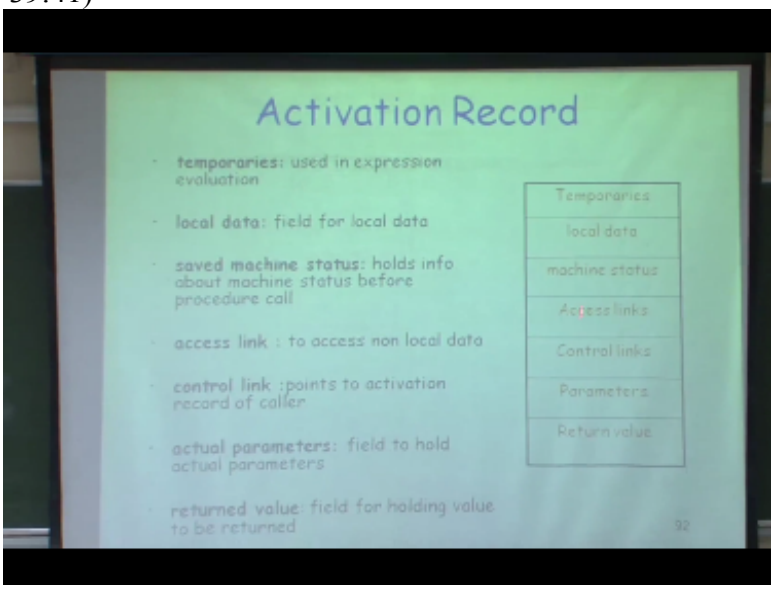
]So when I say that I compile the program some space gets allocated to my process now what are the things what are the kind of parts I have in my program so I obviously have a target good then I have to get out it so this is the machine code and the data object and then I need two parts yeah but where I can revise that I can have information about the procedures and then I need one part.

(Refer Slide Time: 36:23)



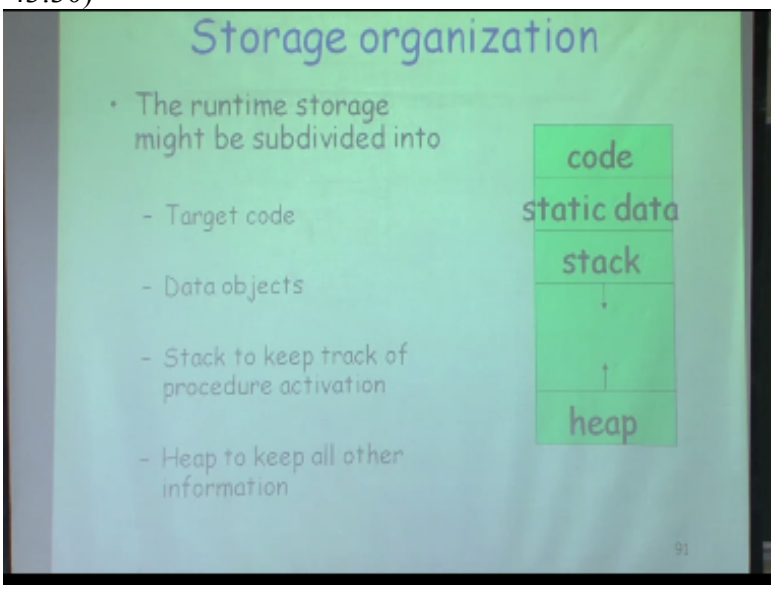
And if this is a total space which is available to eat normally you say that my code and staff is beaming off this allocation and then this back and heat will start going towards each other form two different times and this obviously obvious advantage that if I if I change this data structure for example and now that then fits space for stacking what may happen is that I may say that I devised this as for my data back and eat it and I say that stack and heap go in this direction I can end up wasting space.

So this is possible that my staff will not be full but he becomes full of people not beautiful like that but this data structures gives me an advantage because as long as there is some space available either stack and now this should also give you a hint now that if you write a program now many times you must have see these messages that the program is getting executed and that sudden you say take over or that means is potentially you have gone into an infinite recursion.
And beyond this packets are going to go and say hello so sexless all it says people will flow that means you are allocating too many little structures and you are not free good and at some point of time you point with this boundary and say that beyond this act how to grow it is and this is always a finite state it depends upon them or does it depend upon each of this so this is the so first let me give you. So what is my activation record look?
(Refer Slide Time: 39:41)



And then we will discuss each of these so this is our typical activation record one of the spaces you put in this activation This is not in any particular order we will discuss author slightly later but this is the information you are going to put in each activation so I have space for 10 minutes so basically what will happen is that you find you look at some port in the procedure I need certain factory locations to get all those expressions I then each local data also ye to return certain values while location .
Where I can put the return values and I need location now what about this I have this machine status access. So if you look at this so this spot and this part actually relates to the body of the function and procedure but if you look at this part machine status what it means is that machines is going to have each machine is going to have certain resources so for example suppose we say

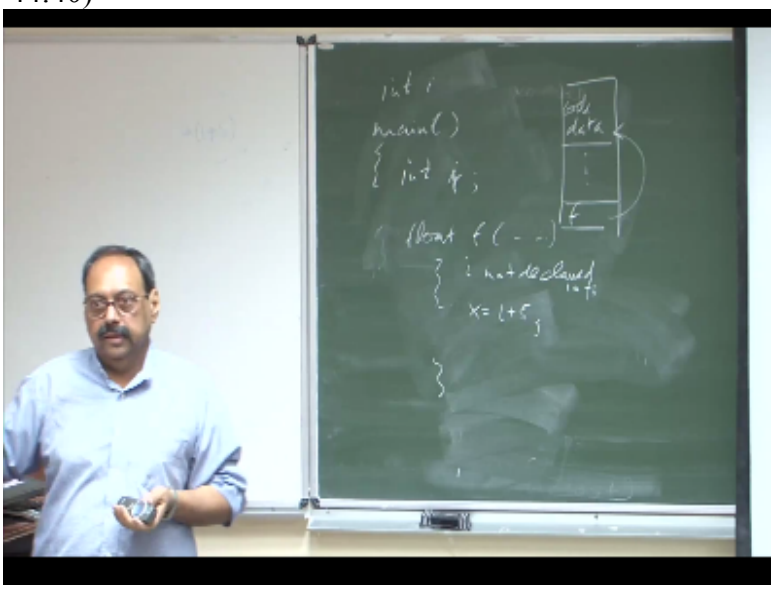
that I have 16 villages now the distance are also going to be used for ten great locations so somebody steps on like EC then you take the statistics.
And so they goes on to the program but when you now say that I have registered spot at three locations and you see every most by and is keeping certain variables in that business now as soon as control comes to be you know that hope for a rocket execution and therefore if you keep a register still bound to variables okay you're actually wasting a resource so what I can do is before I start executing me I can say cell values of all the and after I finished execution of beef I can go and copy those hands back into the distance.
So I need to save information about the Machine status and I need to save all those resources and you to have some space where I can save all those missiles so this really is the space for saving of the Machine status and one of these sources in come space is the one which tells me that from where do I get spoke of a particular variable so when I say that user variable X. X is which will tell me that in which activation this particular value of X can be found. So it could either be in the static data or it could be activation of both and the control genes tell me that which provoked.
So typically this is the information you are going to have on activation report and every time I create a procedure I know procedure I need to create this data structure initialized it and push it understand and every time to see your returns control I need to copy these values back into the activation and then deal. So now I mean if you recall we started with by saying that to see a course on soon activation so you can see that what this activation is also it does not have code is only dealing with the header entity okay. Procedure code is going to be where is the port of procedures for so if I go back to in this,
(Refer Slide Time: 43:30)



Where was four code copies into the sack or in the 4 8 see procedure to the body corresponding to that that it has a machine code where do I store that in this particular code area of the stacking that what is the code in code in slightly really contains the data so stacking you are basically corresponds to the data which will be active and much cozier is X so all the variables corresponding to the procedures are going to be here but all the poor corresponding to the cord is will quote corresponding to main program as well as all that procedures that is going to be their support will procedure code back okay.

So this is what my activation record. And what the issue. I need to worry about when I am designing my activations access this point of finds only thing I can explain at this point of vanished without going into details that so look at situation like this.

(Refer Slide Time: 44:40)



So suppose I say I have a program and I have this so let me deal with first the language is not likely but which are deep nesting of the recent conference so we are in this suppose I have now a declaration of a function let it go else so this whole argument list than the body of this code and here I say X is the sine I+5 and I mock clarinet. So now when I have time to excess I from where do I get this value five I need to know the location where is obviously not in my local scope it is not available so it will not be in my division but it will be in some other activation.

It will either be in this activation or it will be the static activation how do I access this now I do not know how many times will go okay so my spec will look something like this I have some so let us say for instance a that is here okay and after that somewhere I have activation of it that time I say I want to modify this I want to assign to X whatever is that you 5+5 how do I exercise.

So activation record is basically saying that I now have enough code who say that I must have when I created this activation of that I must have to be able to link saying.
That because of my scope rules this I coming either from this or from this if so suppose this was not I this was did then I am saying that this is actually a global variable. And therefore I must have a pointer to that this points basically nothing for us. So whenever I am pushing my activations in I need to have these existence that is only okay so we go into details of saying that how do we create existence but this is the basic definition of fact that using the spoke information which variable and this access.
Any language uses muster procedure hello Austin in fact even see NCC does not have nested procedures but we can write actually less the procedures in see and use PCC to compile them but I mean you should not be writing those kind of programs because those are not standard but PCC box are all that world class of vandalism happens including model of a hostile - object or the object see opportunities they will offer deep level of nesting is basically see that whether you can less the procedure inside the procedure in see you cannot right so I think so before we take up the next pocket I think we will have to wait for the next class so let us take care of today and a next class.

Acknowledgment
Ministry of Human Resources & Development
Prof. Phalguni Gupta
Co-ordinator, NPTEL IIT Kanpur
Satyaki Roy
Co Co-ordinator, NPTEL IIT Kanpur
Camera
Ram Chandra
Dilip Tripathi
Padam Shukla
Manoj Shrivastava
Sanjay Mishtra
Editing
Ashish Singh
Badal Pradhan
Tapobrata Das
Shuubham Rawat
Shikha Gupta
Pradeep Kumar
K.K Mishra
Jai Singh
Sweety Kanaujia
Aradhana Singh
Sweta

Preeti Sachan
Ashutosh Gairola
Dilip Katiyar
Ashutosh Kumar
Light& Sound
Sharwan
Hari Ram

Production Crew

Bhadra Rao
Puneet Kumar Bajpai
Priyanka Singh

Office

Lalty Dutta
Ajay Kanaujia
Shivendra Kumar Tiwari
Saurabh Shukla

Direction

Sanjay Pal

Production Manager

Bharat Lals

an IIT Kanpur Production