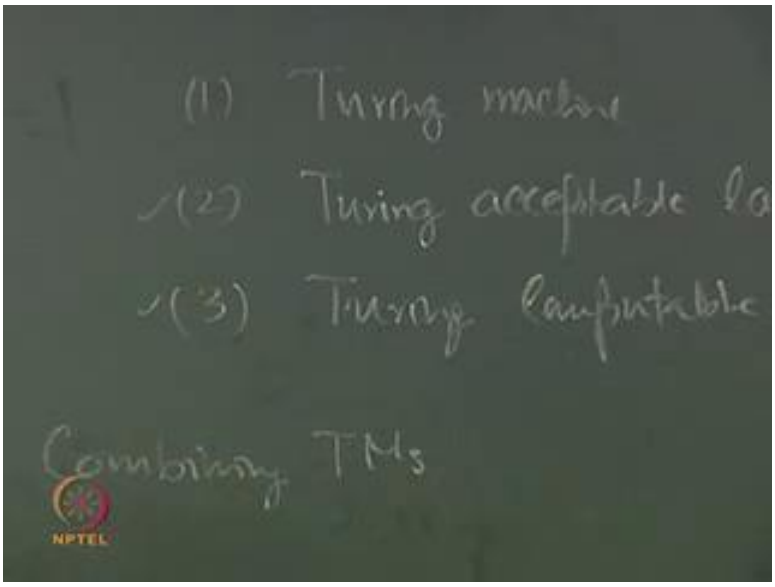
Formal Languages and Automata Theory
Prof. Dr. K. V. Krishna
Department of Mathematics
Indian Institute of Technology, Guwahati

Module - 11
Turing Machines
Lecture - 05
Turing Decidable Languages

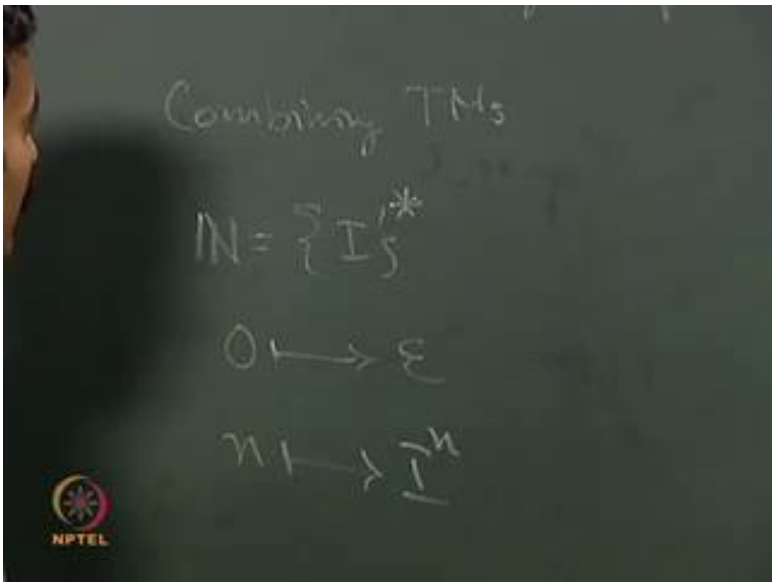
Yes, so far I have discussed in Turing machines, Turing acceptable languages, Turing computable functions. And to construct Turing machines as for using Turing machines that you have already constructed some composite Turing machines, so far we have discussed.

(Refer Slide Time: 00:43)



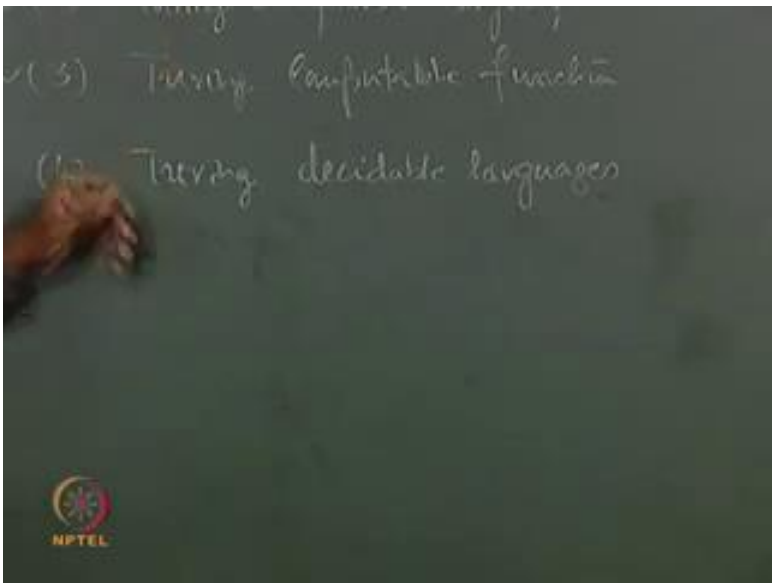
So, you know the standard Turing machine the definition of Turing machine and what is Turing acceptable language, Turing computable functions in Turing machines. We have discussed these two concepts and we have constructed Turing machines some little more complicated using composition of Turing machines. There we have introduced the concept composition of Turing machines combining Turing machines we have discussed.

(Refer Slide Time: 01:39)



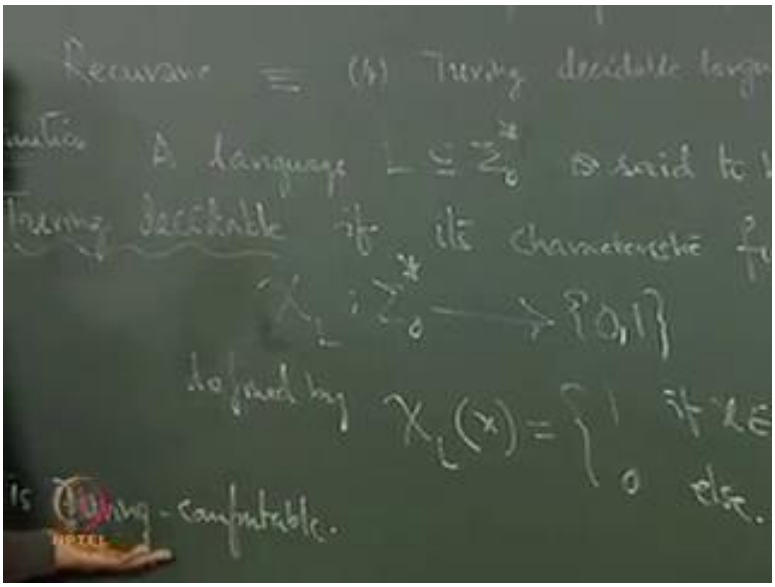
Now, in case of Turing computable functions for multi input Turing computable functions also I have discussed in particular computing with natural numbers. Also I have introduced where natural numbers will be considered in the unary representation. That means natural number over one element alphabet that single ton I star here the correspondence is 0 is empty string and each number n is considered to be I power n. This is the string we consider and we give the input to the Turing machine, so these are the things that I have already introduced.

(Refer Slide Time: 02:38)



Now, let me talk about another concept that is Turing decidable languages. Turing decidable languages here Turing acceptable languages, we have already named them as recursively enumerable languages is a recursively enumerable.

(Refer Slide Time: 03:03)

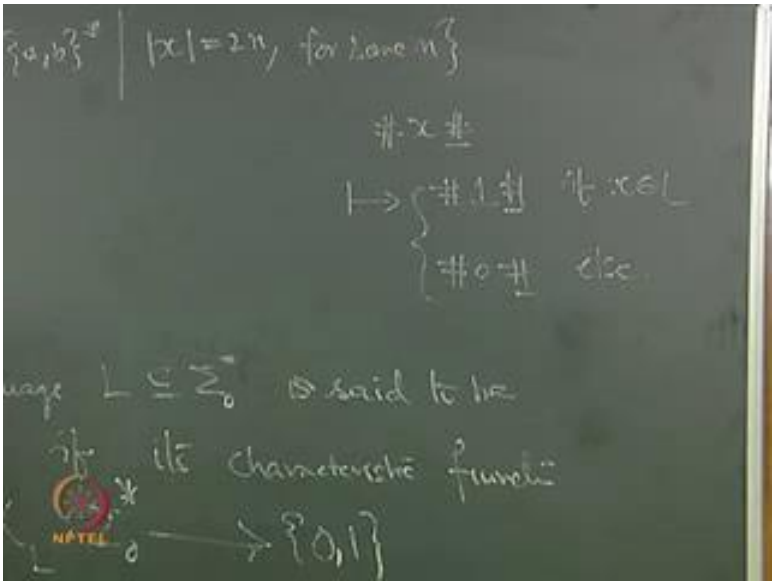


And in this case we called this is recursive languages, the recursive languages Turing decidable languages, so first let me define what is that definition. A language L over some alphabet, let me write or an alphabet sigma naught is said to be Turing decidable is said to be Turing decidable. If its characteristic function its characteristic function, let me write that as chi L this is from sigma naught star to 0,1 or I can essentially take two symbols. Here talking about yes or no is chi L that is defined by defined by this characteristic function you know the definition any string you take you will give L if that string is in L 0 else. So, the string if it in L we are giving L I may you may put yes or no sort of for 0. You can take the symbol no n and for one you can take y and you can assign accordingly.

So, this characteristic function is Turing computable, so a language is Turing decidable if its characteristic function is Turing computable. That means you give a string from sigma naught star as an input to a Turing machine, which computes this chi L. If x is in L that returns L if x is not in L then it returns 0, that means essentially it decides it says whether a particular string from the alphabet whether over an alphabet whether it is in the language or not in the language, this is a decision making Turing machine. Let me

give you an example, you know the Turing computable we should have a Turing machine computing this function a very simple example that you know.

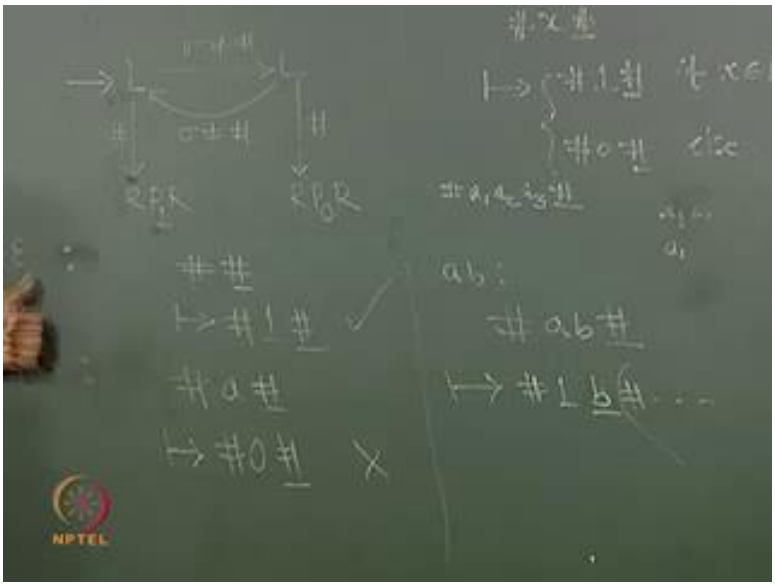
(Refer Slide Time: 06:28)



If you consider the language set of all strings say over a b such that the length of x is even you know this is a regular language for some natural number n. So, like this even this is a Turing decidable language, how do we say this we have to construct a Turing machine. If you give x as an input any x from over a b it has to say L or 0 it has to say yes or no, let me construct Turing machine for that purpose.

So, to start with because as earlier we give the input in this form any string you give if x is in l this returns L if x is in l this returns 0 if else if x is not in L, this is what is expectation. So, we will start with reading and writing at here and this terminates with this, so if this is the input given you will take a left move and you have to understand whether it is of even length that is the cross check. So, you will be stretching between because you know the finite automaton accepting a even length string you will be stretching between two states you will have in two states.

(Refer Slide Time: 08:16)



If it is in the initial state when is in the first state that you are accepting if it is in the second state you are rejecting. So, that is how you are you are constructing finite automaton, so let us construct the Turing machine following that logic. So, first you take a left move and cross check whether it is blank or not blank. So, there are two situations if this is blank I am coming to this place if it is not blank let me now see what is the next symbol that means I will go left, now if this is blank that means I have odd length.

So, this is one situation if it is not blank further I will go to left and cross check what is happening with that, now in the at the first place when you get the blank you know clearly that is empty string and therefore you can accept it. So, you take a right move and print L and take a right move and halt. If this is the place when you are getting blank that means after one symbol you are reaching to this and you are going to left move that means if one length string is there then after that we are reaching this blank. So, at this place you will print 0 in the next and take a right move and halt. Now, for even length we will always be getting empty symbol here for odd length string we are always be getting empty.

Symbol at this location after taking this left move and ultimately you are getting this printing 0, if it is odd length printing L, if it is even length. Let me just demonstrate this through an example for example you have empty string as input if this is the input then that means you are giving this is the input format for empty string. And you take a left

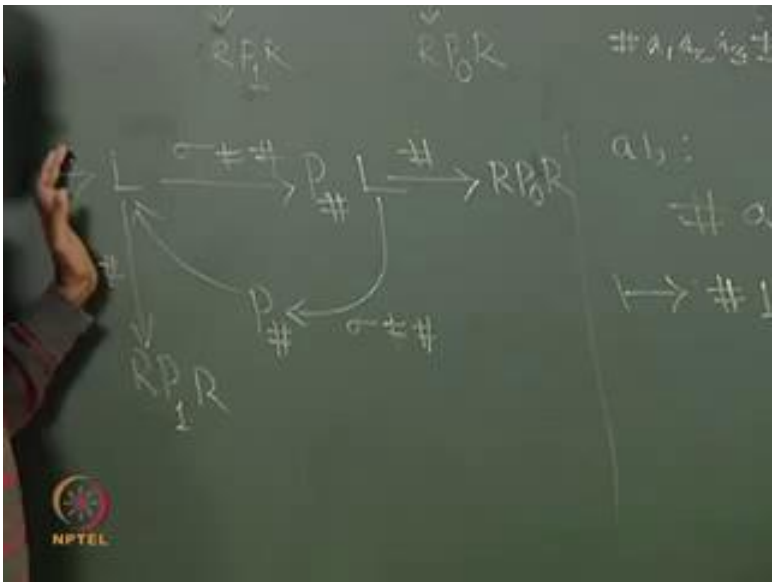
move here and since it is blank here it takes one right move. So, that means you are cross checking and this will be transformed. So, you are taking one left move and then take a right move at this place you print L and take a right move and halt here.

So, this is accepting if you take for example the string a this is L length string for this how the computation. So, the input is given in this format input is given in this format, now take a left move cross check this is not blank. So, I will take one more left move at this place I have got blank.

So, then take a right move here print 0 and take a right move and halt, so this will be transformed to 0 this, so that means the string is rejected. Now, let me consider the input a b let me consider the input a b in this machine what is happening a b is considered with the input format like this how this will be transformed that is the question now. So, from here first left move will it will take left move and it will come here it is not blank it will take another left move this is not blank and now take another left move you are getting blank here. So, this is since it is even length string when you are getting blank you are at this position at this position you are coming to this branch. So, it takes now a right move print L and take a right move what is happening, now you have to cross check this situation. So, thus thing is this from here you are taking a right move take one take right move.

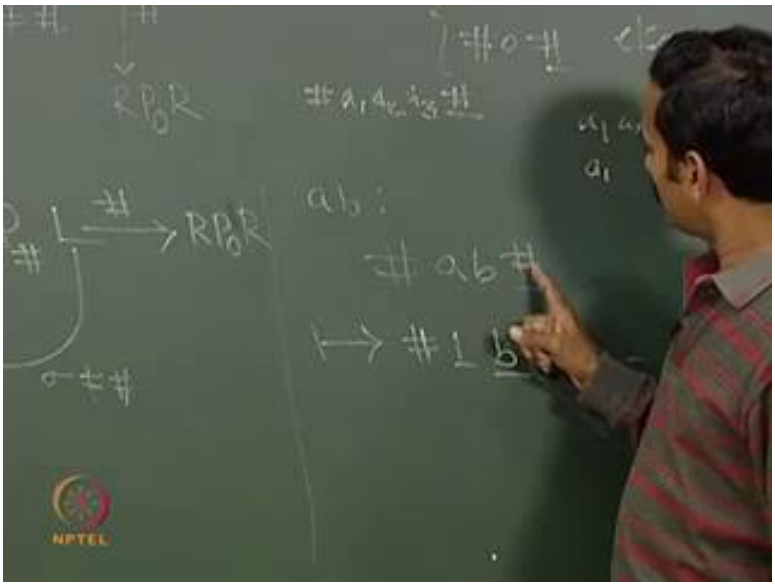
So, at this place you have b and it halts here of course you have several blanks here. So, we are not going to represent this, so we are the our target is to prepare you know the output of in this format. So, when I am giving empty string when I am giving L length string I am getting appropriately L at 0. But, if I have more than L length string if I have more than L length string, then what is happening it is printing L and it is halting in the next symbol where this is not blank. So, simultaneously when we are reading you erase this you erase the symbols, because that is not happening here you erase this and ultimately that machine will give you the appropriate output. So, modifications in this machine, now if you have more than L length string the symbols are lying over there.

(Refer Slide Time: 13:14)



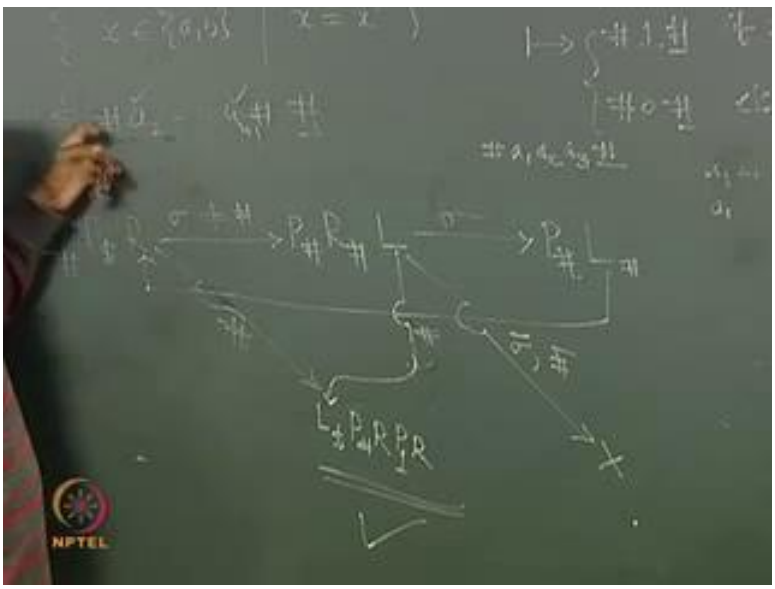
So, let us modify this way take a left move take a left move and check whether this is non empty if it is not empty then print blank then take a left move. Now, if it is blank then as earlier if it is not blank sigma not equal to blank then print blank there take a left move, now this continues as earlier. So, if it is blank take a right move print L take right move here you print 0 take a right move. Now, what is happening with this what are the input that you are giving where as long as you are getting non empty symbols here non empty symbols here. Then it is that particular cell we are making it blank, then taking left move again if you are having non empty here non blank then you are making the particular cell blank then you are taking a left move, so keep doing this.

(Refer Slide Time: 14:26)



Now, while coming from this place to this place we were visiting each and every symbol and you are cross checking whether it is empty or not if it is blank, if it is blank that means you are coming to this place. So, what you have to do here at this place we are making them blank. So, while coming here making this blank you are coming to till this end take a right move you print appropriately one and take a right move and halt here. So, this is how we can construct a decider this is a Turing machine that halts that halts on any input by printing yes or no that. Now, let me take one more example if I consider the language such that palindromes.

(Refer Slide Time: 15:23)



Now, this is a regular language for which you have finite automaton this is a contextual language for which you have push down automaton we have constructed those things. And for this I hope you are constructed Turing machine to show to show this is a Turing acceptable language. Now, a decider let us look at a decider for this purpose, so though mechanism you know as earlier.

This is say a 1 a 2 a n what did we do we are reading this symbol and cross checking with this symbol and going to this point reading this symbol cross checking with this symbol and so on. At that point of time what did we do we have erased this and we have erased this after cross checking, when the match is there. We are continuing further n minus L a n minus L rather it is equal to a 2 we are crossing checking we are erasing and continuing like this in the middle if till this middle if you continue. And the matching is happening we are declaring that it is a palindrome, otherwise we are going to you know left keep going to left move.

And hanging the machine or we are going to the infinite loop that is, how we have constructed to show this language is Turing acceptable language the palindromes. But, here when I am finding that proper match at every stage that means if it is a palindrome I am somewhere on the tape in between. Now, how do we recognize that you know to give the output of this format whether it is L or 0 because when the input is given this way we have to leave either 0 or L yes or no. So, when I am finding this situation when I am finding the mismatch by that time if I had made certain blank symbols.

Here we have made certain blank symbols here somewhere mismatch if it is happening I am not clear that how many cells I have to go left. And where exactly you know I have to print in the second cell 0 or 1. Because once you made certain cells blank we will not be clear where exactly you are because there is no counting business here to understand that yes this many blanks are there I will go back and do that we are not cross checking that. So, what we have to do in the beginning before erasing this state whatever in the beginning what you do you first print some special symbol and then once you do this cross checking wherever the mismatch happens of course you erase everything.

And then you have to and then you come to this place and print 0, if there is match till the end of story, then how do you come to this and take a right move and print 1. So, this is how the decider can be given here this is one important point and has to understand if

you erase certain symbols here. If certain number of blanks are created and after cross checking the input if there is a match or mismatch, we will not be clear that to what extent you have to go to left and print L or 0. So, for that purpose first we have to print at this place some special symbol, so that what are the rest of the business as earlier we can continue.

So, what we will do from here I will come to this blank that means L hash. So, you come to this place you print a special symbol there say for example, dollar I am printing here. Now, the situation is at this place I have dollar and dollar is not part of this input and thus you can distinguish the symbols between this two. Now, may be you can start from here this symbol you can cross check with this and keep cross checking, so take a right move. Now, sigma that is not blank if that is not blank then you can print blank here print blank.

Now, I have erased this place I will go this end and cross check whether with this symbol where they are equal or not. So, what I have to do after print the blank here I go to this blank that is R hash then take a left move I am coming to this position and now here you cross check what is that symbol whether it is sigma. So, if the same sigma is occurring here then I am happy you print blank there you print blank there, now come to this blank and then take a right move. So, here you print blank once you print blank then you can take L h ash come to this place and then take a right move. So, at this position we have to connect look here this is a composite machine.

This is not point into this entire machine if it is point into the entire machine that means it is actually starting from here. But, the situation here is it is not pointing to the entire you I am just pointing to this component that means after this I have to just pursue right move. So, take a right move and see if it is not equal to blank, then you print blank here again, then go till the end because we have already met blank here. Now, I have say a n minus 1 is available then take a left move here if it is matching with a n minus 1. Then print blank then come back to this and keep doing, this is the logic, we have followed to cross check this language is Turing acceptable, so rest of the things are same.

Now, here in the beginning suppose you have empty that means that means this is a symbol that you have here this is the string you know even length string that is a

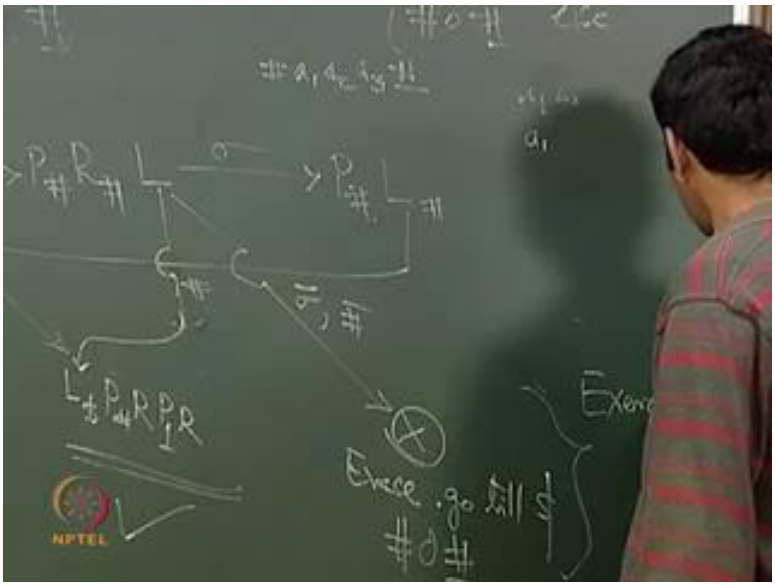
palindrome. So, in which situation you get blank here when you take a right move you get blank here.

We just do distinguish may be after right move you get blank after right move if you get blank then of course this is an accepting situation. Now, you have consume L symbol here and then when you take a left move when you are getting the same symbol that is a non empty symbol, then you are erasing it and continuing. Now, here there are two situations, one you get a blank symbol or some non empty non empty symbol some non blank symbol that is different from sigma. So, there are two situations in one situation you have to accept it in another situation you have to reject it. So, what do I do when you have this is here blank when you have here blank then you will accept that it is a odd length palindrome and in the situation that is different from sigma. And it is not blank also if this is the situation, now here we have to reject, so these are the situations that we arise.

Now, when I am getting blank in the beginning we know left of this if you have erased certain symbols there all erased. So, you simply go to this dollar this dollar you convert it to blank and take a right move print L and accept this is the situation even in this case also the similar machine can be connected. But, in this case we have to erase whatever is the left over input when the mismatch is happening if it is not sigma if it is not blank. That means some other non blank symbol is available in which situation whatever left to this we have to erase it and then you have to you have to print you have to print 0.

So, here this is easy take L dollar that means you go till end of the tape left end of the tape there you print blank there you print blank take a right move here print L here print 1. And then take a right move and halt we are happy here, similarly if you receive this I can connect to the same machine go till L dollar print blank here then take a right move print blank here then take a right move print 1 and take a right move and halt. So, this is the halting situation with acceptance. Now, when you want to reject there may be more symbols on the tape first you have to erase them then go till dollar there take a right, it is at dollar you will make it blank take a right move there you print 0 which is not accepting.

(Refer Slide Time: 24:59)

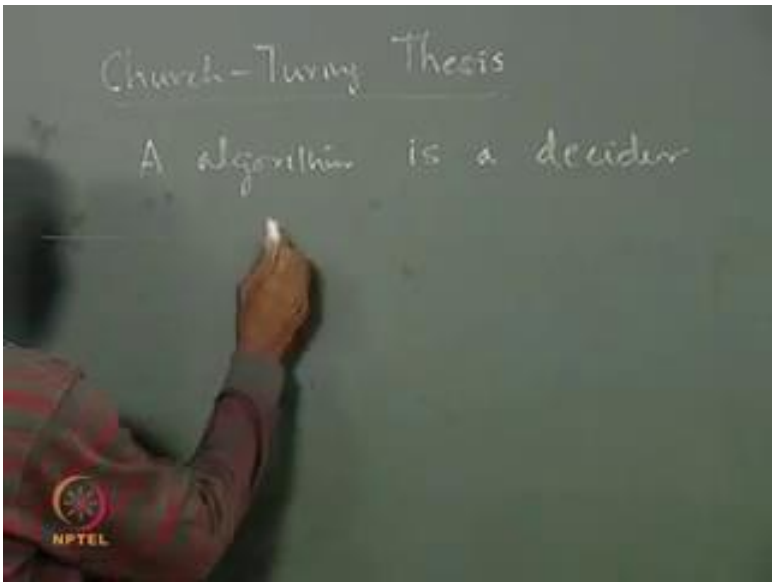


So, that means at this situation to reject the string you have to erase and go till dollar. Then you have to give the output in this format, so this portion. Now, I hope you understand what one has to do here at this portion. You have to do this business the rest of the symbols we have to erase then go till dollar then print 0 finally as output. So, this particular machine like here whatever we have mentioned take an exercise and construct that particular component to give a decider for this language palindromes.

So, a decider is one once again I emphasis you give any input from the decided alphabet you give any input the situation is earlier in case of Turing acceptable languages or Recursive enumerable languages, what we are doing. You give an input if it is in the language that simply halts otherwise it may you are creating whether it is hanging or we are putting it in the infinite loop. So, that the string is not accepted that is how we have managed.

But, in case of this Turing decidability that means to say a language Turing decidable you give any input from the alphabet over the alphabet; there are two situations you will have precisely either one you will print finally or 0 you will print. Finally, that means you may print y for example you print yes or you print no to indicate. So, that means on any input this Turing machine whatever that we are constructing that has to halt. So, you one has to observe this remark that this type of Turing machine the Turing machine that is existing for a Turing decidable language that halts on any input a decider.

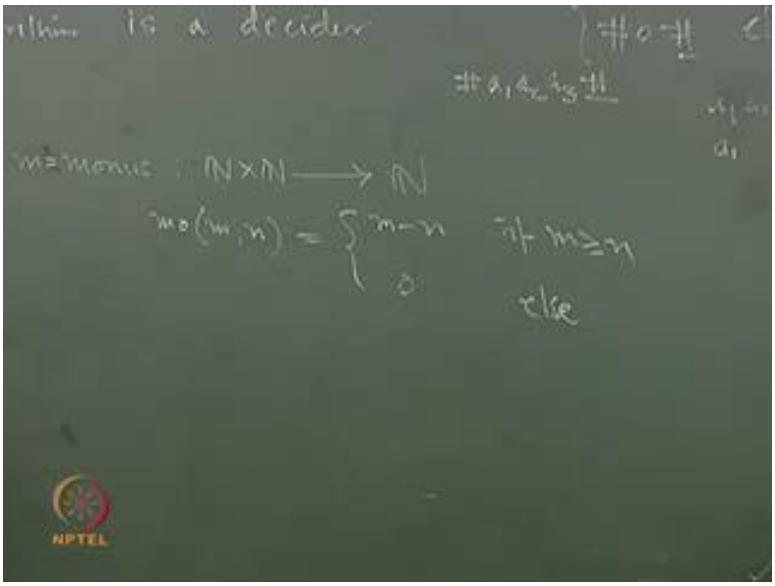
(Refer Slide Time: 27:08)



Let me call a decider is a Turing machine that halts on any input by printing by accepting or rejecting the input by accepting or rejecting the input. So, let me present this way here printing 1, we are calling it is accepting by printing 0, we are calling it is rejecting. So, this kind of Turing machine which is halting on any input this is the notion of algorithm church turing thesis. What church turing thesis claims is an algorithm is a decider the decider that means an algorithm is a Turing machine that halts on any input that is how it is considered. And this is the formal notion of giving algorithm, now either I have promised earlier.

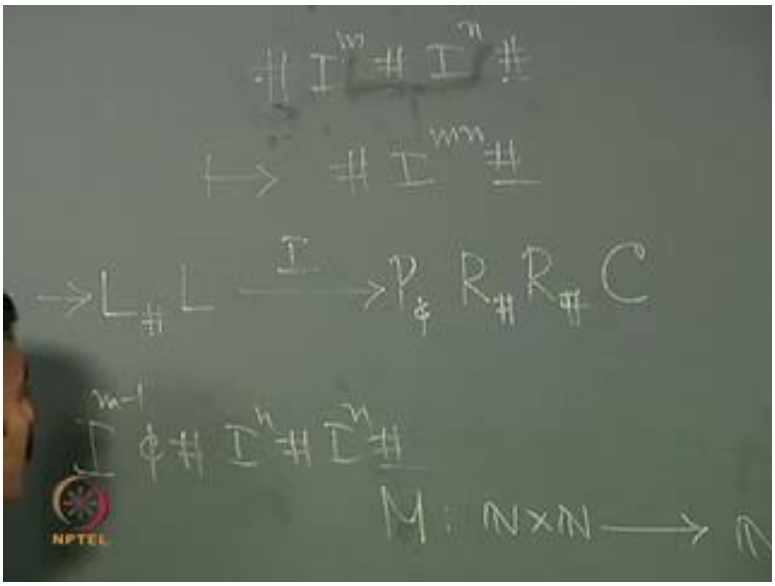
Let me now consider some examples of Turing computable languages over natural numbers. We have already discussed say successor function edition their Turing computable functions and this using Turing computable to we have to introduce the notion of decider, now let me cross check this property for example.

(Refer Slide Time: 29:30)



Example you consider a function this is whether I have mentioned monus this from n cross n to n in short. Let me use may be n monus m n is m minus n, if m greater than or equal to n 0 else thus this is a non negative number we have to give the natural number. So, instead of considering minus, we consider monus in this computation, so the Turing machine which can perform this job. Let me give you certain hints to construct a Turing machine to show this is a Turing computable function monus defined like this is a Turing computable function. The Turing computable function the Turing machine for this, let me gives you some idea in this direction you may be giving input I m I n like this. This is the input format the input format after finitely many steps the Turing machine has to leave this, we use the short hand for this m minus with a dot here called monus.

(Refer Slide Time: 31:33)



Let me use m monus n with minus over which I am putting a dot I power m monus n this should to be the left over thing this may be 0. If this is smaller if this is bigger or equal then m minus n that what we performing in this machine what you have to do you erase one. Here for example you one here and you go and erase one here erase one here you go and erase one more I here and keep doing this if you receive blank after erasing these symbols. Then you cross check what is the situation here whatever is left over that is if this bigger then something will be left over those may I s will be there.

If this is smaller than when you are erasing and you are cross checking with this you will encounter this blank quickly. So, you just understand that when I am doing this business erasing one I here erasing one corresponding to that one I.

Here if I am reaching the to this blank first then m is smaller than n if I am reaching to this blank first then n smaller than m. So, appropriately you have to leave the output, so can we start with this logic. So, take a left move here you have I or blank that is how we will be considering. So, take a left move and see whether you have I or blank that is the situation, if you are already receiving blank, if I have by that time we have erased certain things whatever is left over till that point I have to go and see. But, now the bottle neck is like this, if they are equal, if they are equal I have erased till this point and then I have erased these things then how do I recognize that to this point I will come and whatever is left over I have to restrain.

So, for which you can bring some notion of special symbol put it here put it here and for which you will be cross checking. So, you can now consider some special symbol printing here, then you do this business you go this another special symbol here cross check whether that other particular special symbol is the arrived or not.

So, that is how you can manage, because when I am making them blanks I have to be clear like to what extent I am going to what extent that you are going. So, to understand these things may be at this place you can have a special symbol and at this place also you can have special symbol and continue this business by cross checking to construct the Turing machine which compute this function. For one more example these are little easy let me put say capital M to take about multiplication m n. This is also Turing computable function multiplication of two numbers is Turing computable is a Turing computable what you have to do in this case again.

The input will be in this form the input will be in this form, now what I have to do here I have to multiply these two and I have to leave that as output m. So, what is the procedure that we can follow here this I power n for example you copy this m number of times. So, that you will be creating I power m n. And then of course you can erase say for example, in this case I power n is here you erase one I you erase one I and this is I power n using copy it here and once again erase one more I copy I power n here and so on. So, this is the logic if you follow then we can consider this business let me.

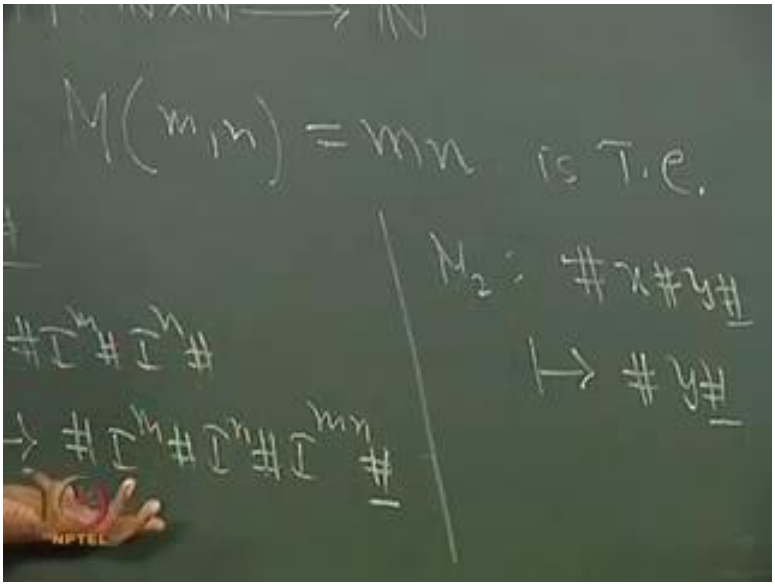
So, what we do you just again certain hints I am giving from here I come to this I come to this place say L hash take a left move if I have I there if I have I there then I may just convert into a special symbol. For example say print say some sent just to indicate this then I will use R hash I am here I will come to this first blank. Then once again coming here R hash that means R hash square I can use. Now, you apply the copying machine, which can copy this I power n and print it here, but that is what happens here.

Now, I will apply say copying machine, so I am at this place I am applying copying machine. Now, what do you do after copying this what will be left over this I power m minus L one sent is here blank I power n, this is the situation we see. Now, in the next place what I will do I have to come to this point and its skip sends if you are getting here you have to skip sends if you are getting blank. That means this block is over if you are getting I then again you print send there then take right move take right move. Now, in

this situation I have to go to this point and then I have to copy. So, here I have to go till this end because here I have to copy this.

So, what you do because this copying now we have to give this copying not this string because I will be generating afterwards another I power n and so on, so copying one string that is that is leaving. So, that means you have to know have a Turing machine which may for example when I have this kind of situation a Turing machine which prints this because this second block of x need to be printed here.

(Refer Slide Time: 38:52)



Say x let it be there and y x if you create a Turing machine which can perform this business, then you can appropriately modify this. And construct multiplication Turing machine I hope e this hint is sufficient for you to you know construct this Turing machine. Once again to show multiplication is Turing machine Turing computable function you take I power n I power m as inputs and I have to leave this as output.

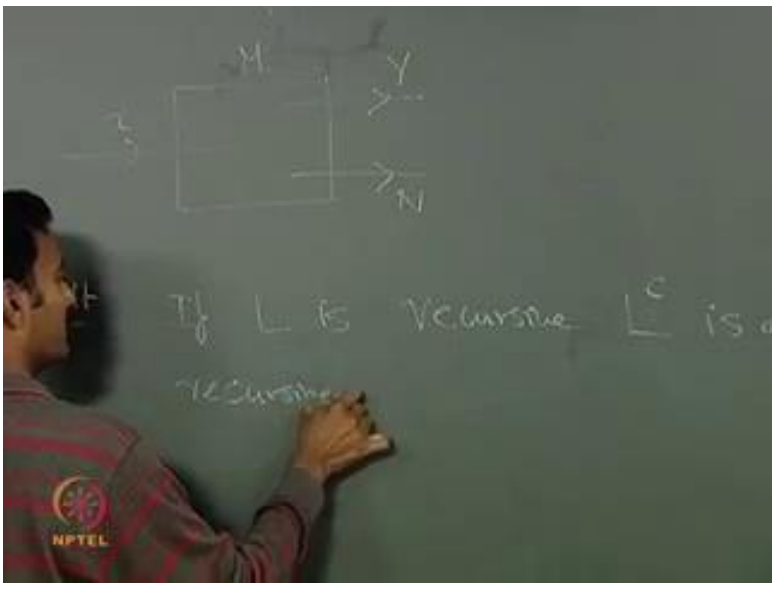
I am just starting the business, now you see the logic is from here I will count the number of I s here; that means essentially cross checking each I am just cutting here may be by printing. So, sent or whatever I will go to the end and every time I mark one I copy this block to the end and keep doing this at the end. Of course, we have to erase these two and move. So, that means essentially what I am doing, if I have some situation like this when I am here x y is input, if I can create something like this that means I power n will always be copied that many number of times and create I power m n here.

So, end of the story once you create that machine and connect it appropriately end of the story what will be the situation when I am giving this as the input I power m I power n. This is the input with this kind of process you will have I power m I power n I power m n I will get I power m n here.

Now, you have you create one machine that can erase two blocks of strings before one particular string that means given. So, another type of machine that you can look at this I am calling m L, for example m 2 is a machine which you can try when I am giving x y as input 2 strings. This machine leaves by erasing x this leaves y as output for example if I create such a machine then I have to do at this point I will apply this machine once. Then I am left with I power m I power m n on the tape once again I will apply this machine and then I will be left with I power m n.

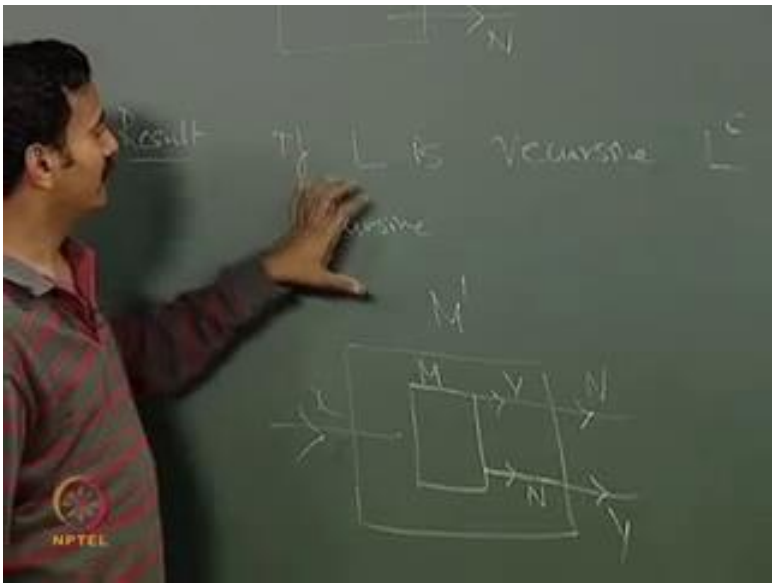
So, using this you can using this phenomenon using this machine m L m 2 connecting appropriately you can construct a multiplication machine not only multiplication, you can consider you know little more complicated arithmetic. And construct appropriately these machines Turing machine which computes this arithmetic operations and certain manipulations over natural numbers that are computable. What is the time [FL] 50 [FL] [FL] 4 minutes [FL] ok thanks, now let me look at some important properties related to Turing decidable languages.

(Refer Slide Time: 42:50)



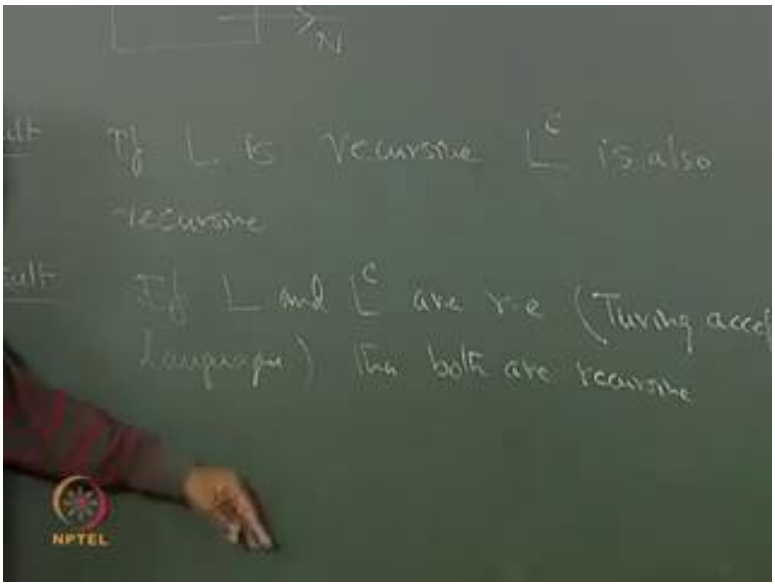
That is as we have understood there is a Turing machine that is a decider. Suppose this is a Turing machine you give any input from sigma star it is resulting yes or no it is printing yes or no, so this how it is working. Now, using this phenomena, we can quickly understand this result if L is Turing decidable that means recursive L compliment is also recursive the Turing decidable language. How it is what are the Turing machine that you are constructing whenever it is printing L, because you have the Turing machine M. Now, you construct the Turing machine M dash the transition map is exactly same except that whenever it is printing L finally you take the input whenever it is printing 1.

(Refer Slide Time: 43:57)



So, you give it to N, so whenever it is saying yes then you ask it to say to no whenever it is saying no you ask it to say yes. So, this is how you can manipulate that means the Turing machine the Turing machine M is used only thing is before. Finally, it is halting you just take a left move and see whether it has printed 1 if it has printed 1 you print 0 there and take a right move and halt and vice versa. Thus, you can understand quickly that if L is recursive it is compliment is also a recursive language, so this is the construction that one can make.

(Refer Slide Time: 44:45)



And now another result if L and L compliment are recursively enumerable that is Turing acceptable languages if they are Turing acceptable languages, then both of them both are recursive what we have to do. If L is recursively enumerable you have a Turing machine for those strings, which are in L it says yes and since L compliment is also Recursively enumerable you have a Turing machine for those strings which are in l compliment that says yes. So, given input what do you do you manipulate it like this. So, let me say M 1 is a machine Turing machine.

(Refer Slide Time: 45:44)

That takes input and says yes for L, so this for the machine land. Now, what you do simultaneously you give the input to M as well as M 1 as well as M 2 that if for L compliment that is where L compliment take this. And for the input x either if this will say yes or this will say yes both cannot say yes, so what you do whenever it says yes. So, you simply report that if this is saying yes you say yes if it saying yes you report no. Now, given an input one of them only will precisely say yes both cannot respond for example in this case it may hang or it may go to infinite look like this.

So, what we have to do whenever it is saying yes that means after finitely many steps you are getting this information if it is not in L that means if it is in L compliment this machine will say yes this will halt. So, in which case you asked to say no thus you can understand that if L and it is compliment if L and it is compliment both are Recursively enumerable languages that means Turing acceptable languages both of them are Recursive, because L is recursive that I have shown.

Here again using this result if L is Recursive L compliment is also Recursive you can understand that both of them are Recursive languages Turing decidable languages and regarding a pair of L and L compliment. Now, we can say this point that both are Recursive if and if 1 is recursive other is becoming Recursive. Now, the situation is a language, which is Recursively enumerable. But, not it is compliment that kind of situation is another situation there are three situations. Now, you see clearly that this is happening this way we can discuss more about this properties in future lectures.