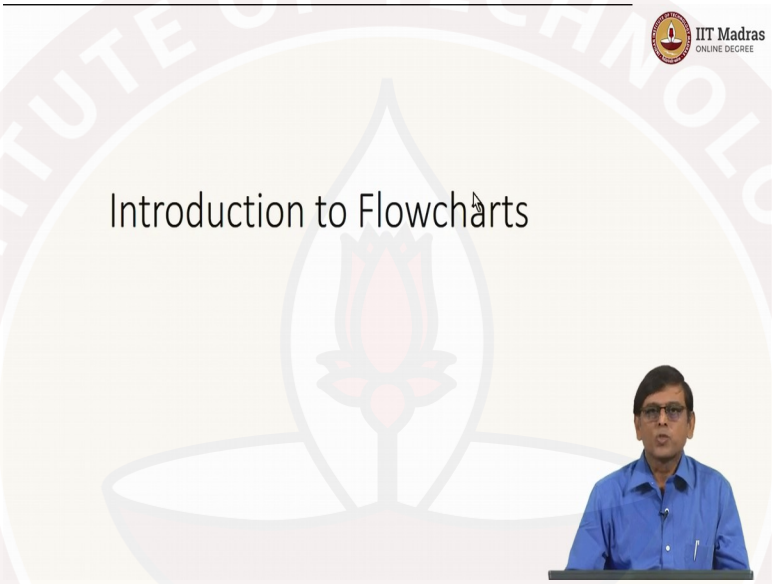
IIT Madras

ONLINE DEGREE

Computational Thinking
Professor Madhavan Mukund
Department of Computer Science
Chennai Mathematical Institute
Professor G. Venkatesh
Department of Electrical Engineering
Indian Institute of Technology, Madras
Introduction to flowcharts

(Refer Slide Time: 00:15)



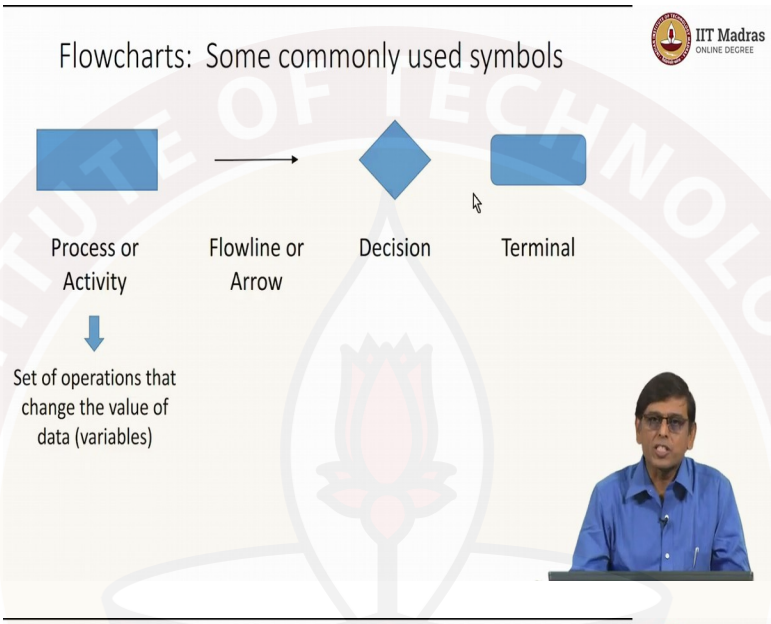
So in the previous few lectures, we have seen how to do an iteration in which there is an initialization step and then there is a set of things that we keep repeating. And we use iteration basically to do a few things. One is to count how many cards are there in a pile. And also to actually sum up, add up all the mathematics marks, or the total marks, so these kinds of activities.

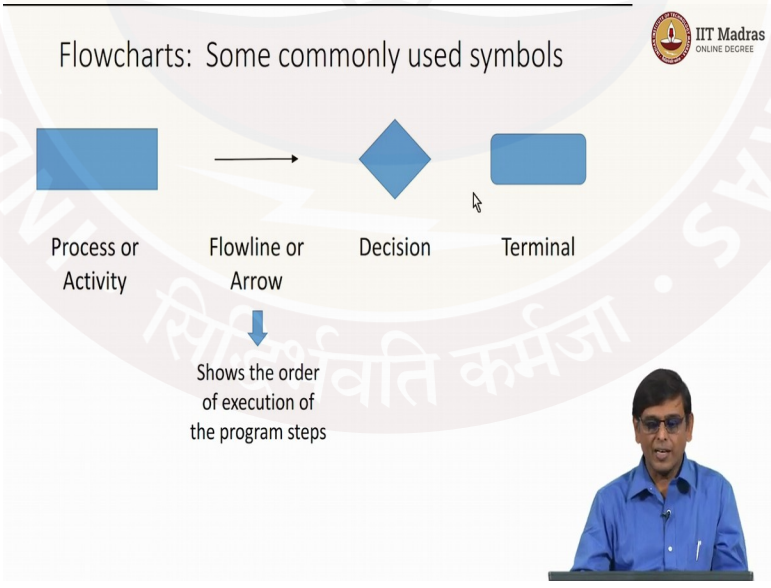
Now one item we did not discuss and that is very important basically is that we need some mechanism by which these step wise procedures that we have described to you can be written down and communicated to another person.

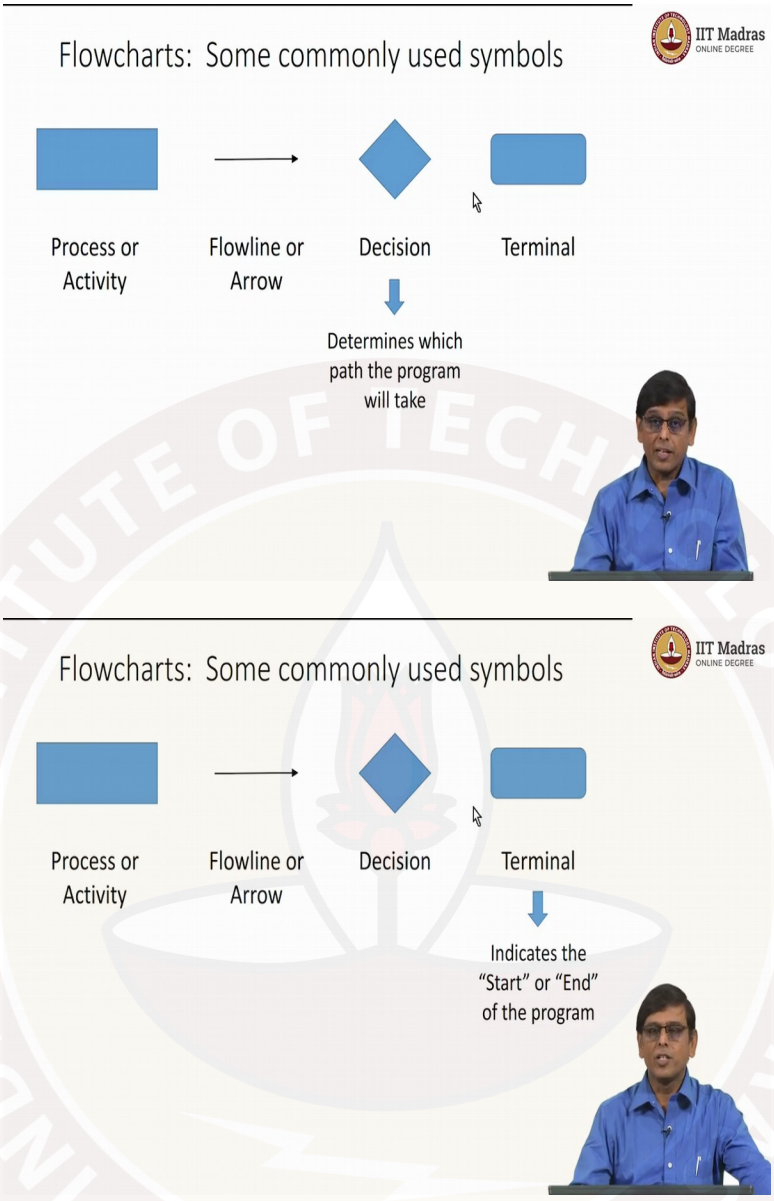
So before we start programming, it is important basically that we write down these step wise procedures in some formal form, so that we know what exactly we are programming the computer about. So there has to be some language or mechanism by which something like this is

done. So typically there are two ways to do it. In this lecture, I am going to introduce to you one method which is called Flowcharts and subsequently we will talk about another method which is called pseudocode, which is like English way of describing a sequence of steps, okay. So this set of lectures is about an introduction to flowcharts.

(Refer Slide Time: 01:42)







And so when you look at flowcharts, these are basically diagrammatic representations of the sequence of steps that we are going to use, or algorithms that we are going to use. And for this diagram, there will be a few basic constructs or symbols that we will be using. There are many more symbols and this they are typically used in flowcharts, but for the purpose of these lectures, these 4 symbols will suffice.

So the, so you see basically this rectangular box in which we can write things, and then there is an arrow, there is a diamond and there is a oval shaped box. So the rectangular box is called a process or activity. And what that box is used for basically is to write down a set of operations

that can change the value of a data, change the value of some data that we have. In our case basically we have discussed variables like count, and sum and so on.

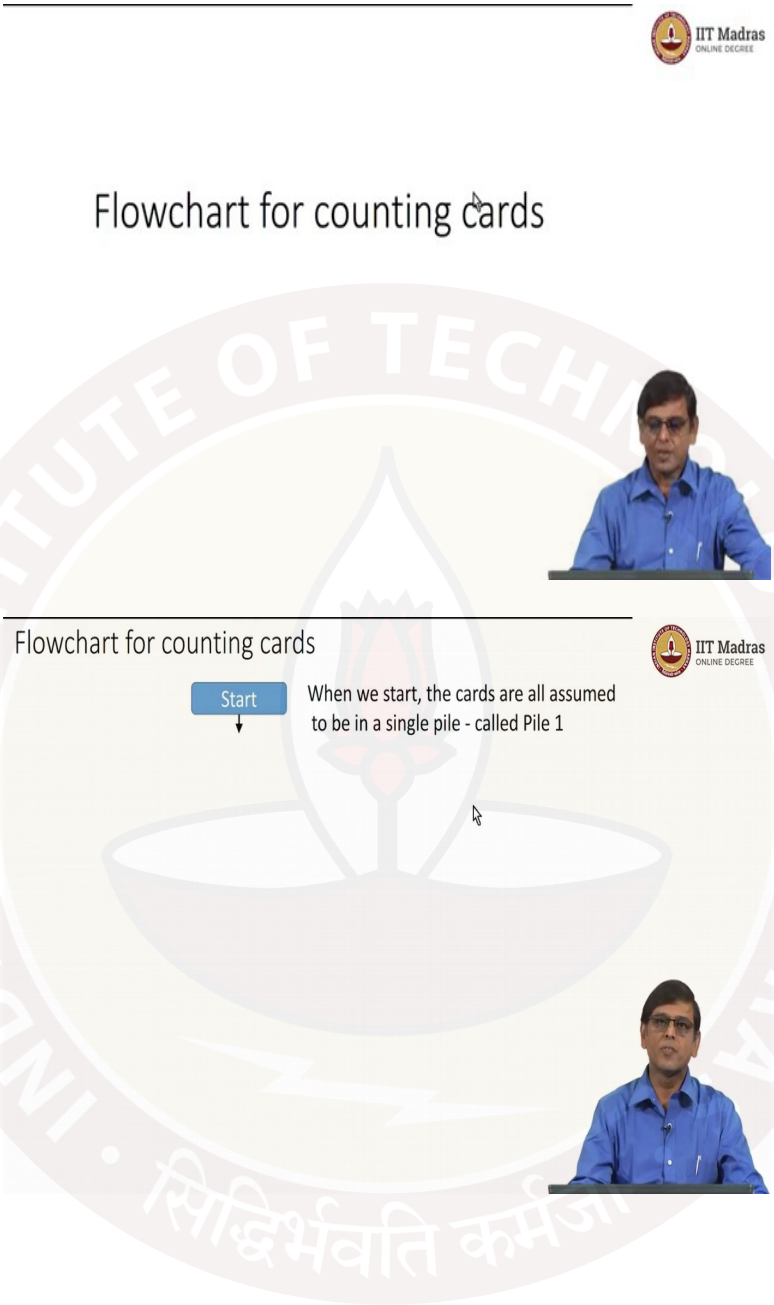
So activities that change, operations that change the value of variable like count or sum, is written inside this rectangular box which is called a process. The second symbol that we have is an arrow symbol. And the arrow symbol is basically used to connect these other boxes that is diamond or a rectangle or something like that. So the arrow shows the order of execution of the program steps.

So if you have two rectangular boxes connected by an arrow, then basically the first, it shows you the direction in which the program will execute. It will show from the first then you follow the arrow and then you go to the next okay that is how it goes. The third box, third symbol which is the diamond symbol is called a decision symbol.

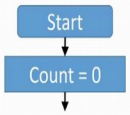
So here we can make a decision about which direction the program should take. So there could be a condition that you are checking for. And when that condition is true for example, you will take one path. And when it is false, you will take another path. So the decision diamond basically is used to make a choice between two different alternative paths that the program can take.

And finally this oval shaped symbol terminal is basically used to indicate the beginning or end of the program. So we have 4 symbols, and we can use these 4 symbols basically to describe the flowchart for counting.

(Refer Slide Time: 04:16)

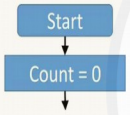


Flowchart for counting cards



We initialise the count variable to 0

Flowchart for counting cards

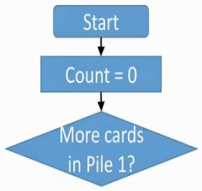


This is where the iterator starts.
We need to repeat the following steps:

- Pick a card from Pile 1
- Move it into a different pile (say Pile 2)
- Increment the value of count

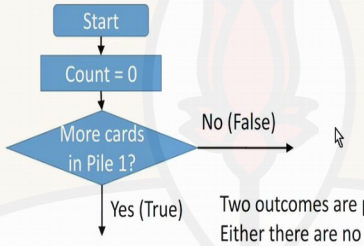
The iteration stops when Pile 1 is empty

Flowchart for counting cards

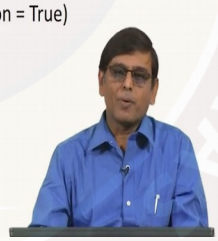


We need to pick a card from Pile 1
But we can do that only if there
are more cards in Pile 1 to pick !

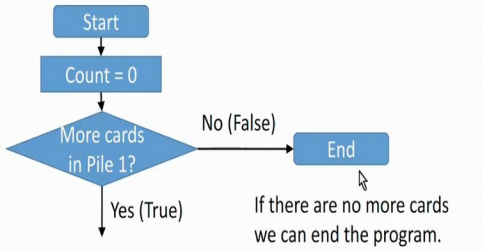
Flowchart for counting cards



Two outcomes are possible here:
Either there are no more cards
(condition = False) or there
are more cards (condition = True)

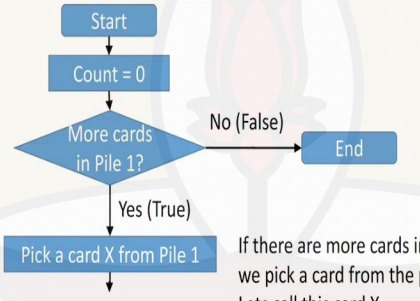


Flowchart for counting cards



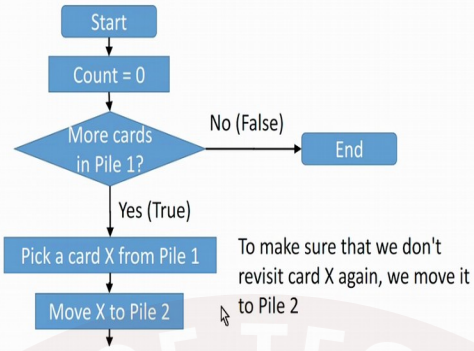
If there are no more cards we can end the program. The variable count carries the required value.

Flowchart for counting cards



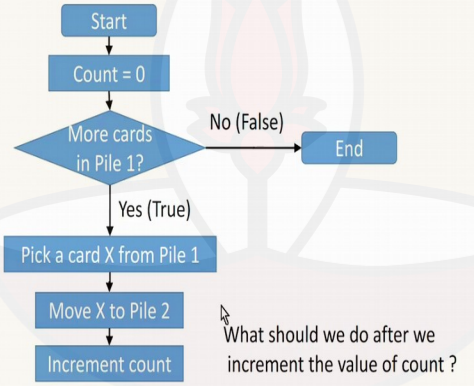
If there are more cards in Pile 1 we pick a card from the pile. Lets call this card X

Flowchart for counting cards

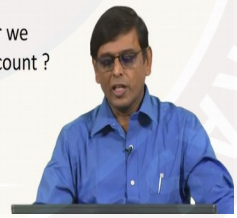


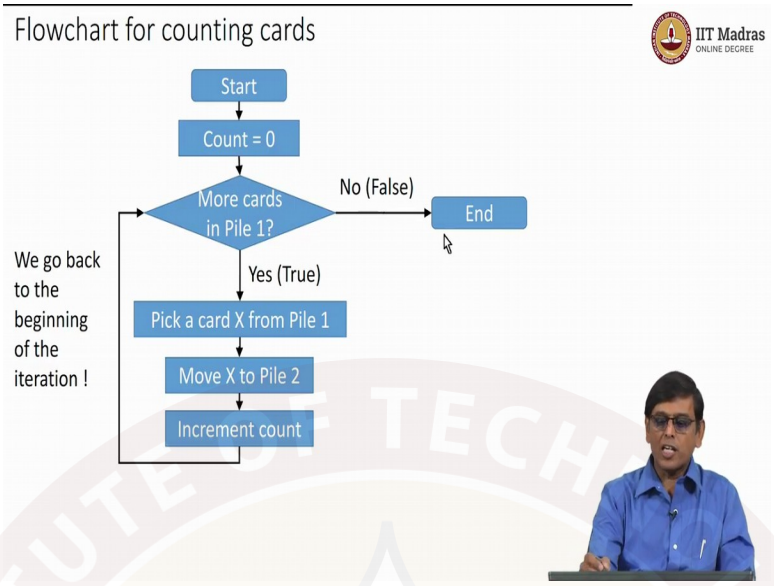
To make sure that we don't revisit card X again, we move it to Pile 2

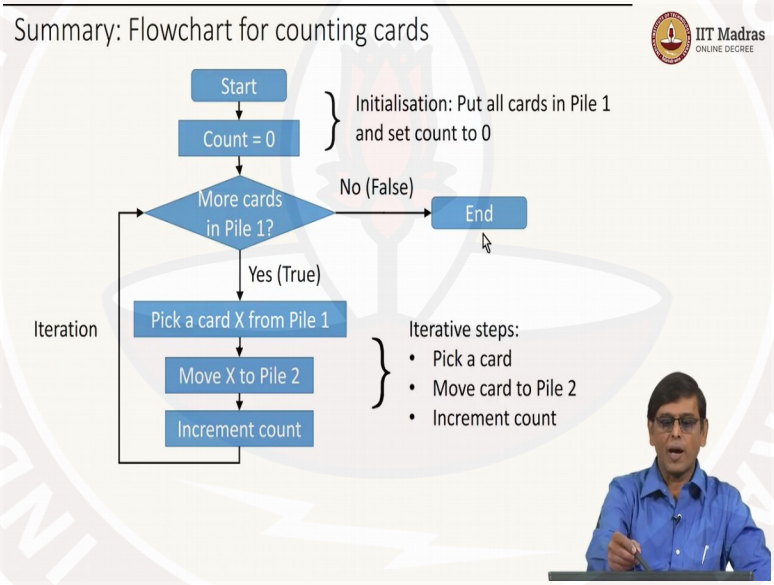
Flowchart for counting cards



What should we do after we increment the value of count ?







So let us look at how do we count cards, how to we write how to count cards using a flowchart, so that somebody else who is looking at the flowchart can also understand what is it that we are discussing in terms of a sequence of steps.

So the first thing that we need to do when we count basically is to start. So the first symbol will always be start symbol, followed by an arrow because we are saying that from start we are going move in to the next step.

And we are assuming basically that when we start all the cards that we want to count are in a single pile, we call that pile 1. And the first thing that we did when we described the procedure to you basically is that we started with a variable called count and this variable count was initialized to 0.

So the way to describe that in the flowchart is to put count equal to zero inside this rectangular box. So the first process or the first activity is to initialize count to 0. And then again there is an arrow which basically indicates that now the control or the program is now moving to the next step. And the next step basically is where the iterator is supposed to start.

So if you remember, we said that the iterator will have to repeat the following steps after initialization. We have to pick a card from pile 1. Then we have to move this card into a different pile, pile 2, so that the same card is not visited again. And then we have to increment the value of count.

And we keep repeating this step till we find that pile 1 is empty. At this point of time, there is no further card to pick, so we stop. So we need somehow to describe this iterator. So we have to start this iterator somewhere. Now usually the iterator is started usually with the condition. So the condition is when do we stop? And the stopping condition basically is when the pile 1 is empty.

So we are going to start with this condition. So the check, the condition that we are checking is, is are there any more cards to look at in pile 1. So are there more cards in pile 1? So that is the condition we are checking. And clearly if there are no more cards in pile 1 then we stop. And if there are more cards in pile 1 then we proceed to the iteration steps, which is basically to pick a card from pile 1. So there are two possible outcomes for this condition check.

One is that there could be more cards in pile 1, which means the condition comes out true or there could be no more cards in pile 1, which means the condition comes out false. So these are the two outcomes that are possible at this stage. So let us explore the no condition, which means that there are no more cards, it means the condition turns out false, in which case, we have nothing further to do.

So we can stop. So the way to denote stop basically is to put this terminal symbol end over there. So this basically says that we have ended the program. So there are no more cards and we can now end the program. And at this point, one could output the (valuable) variable counts value, but we are not discussing output right now.

But when we reach end if you look at the value of this variable count, then that variable count should basically be carrying the value of the number of cards. So the other path that we can take is when there are more cards in pile 1 which means that the condition evaluates to true, so we are down this path now.

And when we are, when you are down that path, then we have to do the first step which is picking a card from pile 1. So let us call this card that we are going to pick card X, because we will be referring to this card later on. So let us pick a card X from pile 1. And after we have picked card X from pile 1, we have to mark it as having been visited or having been seen, so that we do not visit it again. And the way we are doing that is by moving this card X to pile 2.

So we moved it from pile 1 to pile 2. And the next step basically is to increment the value of count, because we have now seen one more card, so we increment the value of count. And after we have incremented count, now the next question that we ask is, what should we do now after we increment the value of count?

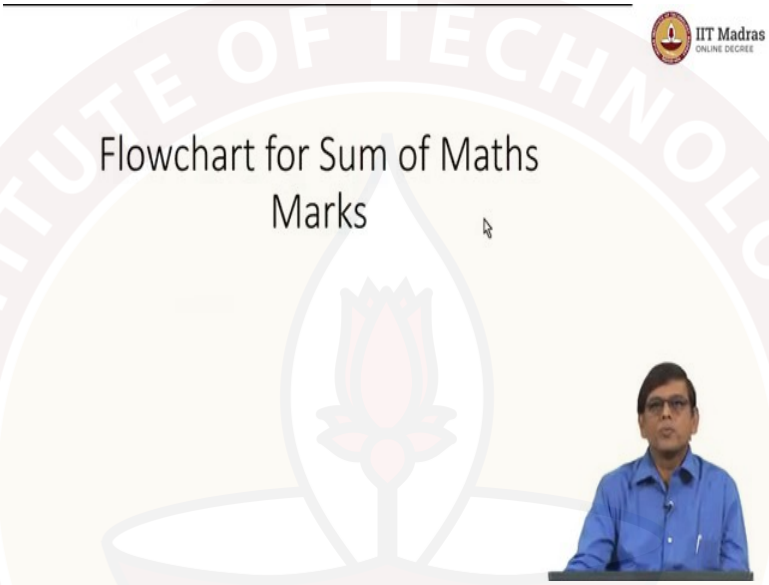
So remember that we are inside an iteration process, so we have to repeat some set of steps again and again. The steps that we have to repeat are picking a card, moving the card and incrementing. So these three steps have to be repeated.

So we have to go back to a place where all this three steps can be repeated. But remember that also we have to be able to stop this iteration, and the stopping of the iteration happens when there are no more cards. So the appropriate point to go back to really is to this condition again. So after you have incremented count we go back to the beginning of the iteration where again we check whether there are any more cards in pile 1.

And if there are more cards in pile 1, then we repeat the iteration steps again, that is these 3 steps again. On the other hand if there are no more cards, then we can stop. So here now we have all

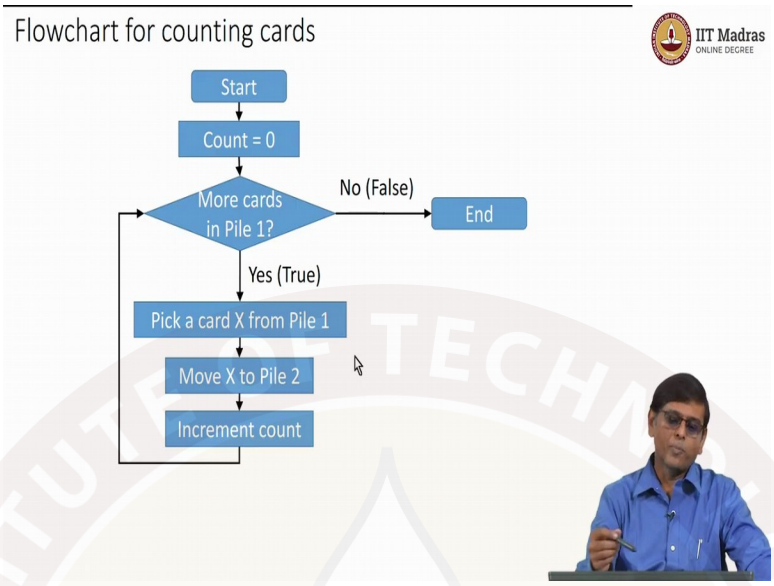
the steps that we need to do counting. We have this initialization part, which is basically setting count equal to 0 and putting all the cards in pile 1 which we have not described in the box, but we assume that at the beginning that is already done. And then we have this condition check followed by these 3 steps which we keep repeating, till there are no more cards at which point we can end. So we have described in some sense now, the flowchart for counting cards.

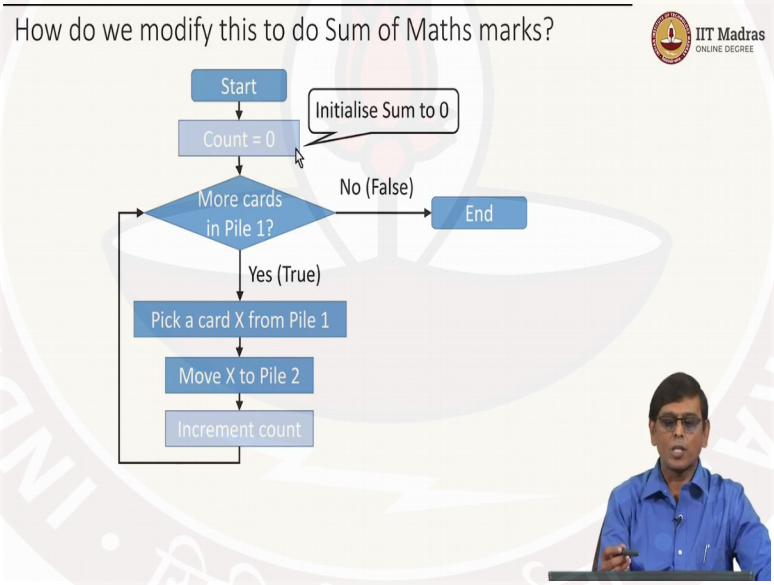
(Refer Slide Time: 09:45)

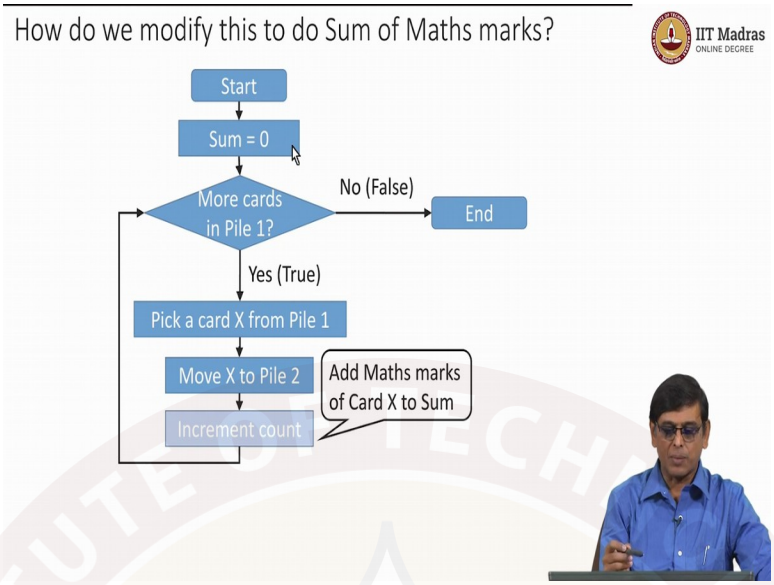


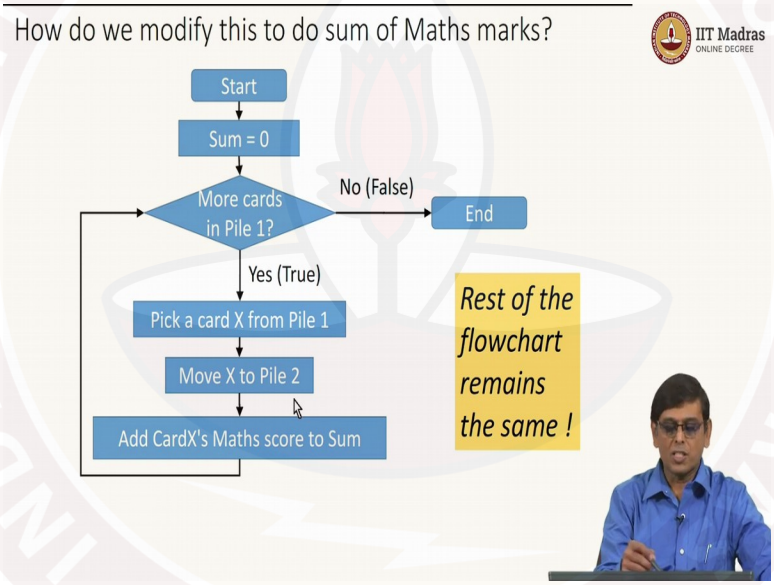
We saw how to build a flowchart for counting the number of cards, and after that we saw that we could modify that procedure for counting cards to do the sum of the maths marks. So now the question we are asking is, can we convert this procedure which is a sequence of steps for summing or adding up all the maths marks, can we turn that into a flowchart? This is the question we are asking.

(Refer Slide Time: 10:14)









So, so just to revisit the flowchart for counting cards, we said that the flowchart for counting cards starts with the terminal symbol start. Then we initialize the value of count to 0. Then we check whether there are any more cards in pile 1 or not. If there are no more cards, we can stop. And at this point the value of count reflects the number of cards.

On other hand if there are more cards in pile 1 in which case the condition turns out to be true, then we repeat these 3 steps which is we pick a card X from pile 1, we move it to pile 2 and we increment count.

Now the question we are asking is, how do we modify this, this flowchart to not just count the cards but to add up all the maths marks from the cards? How do we add up all the maths marks from the cards? So the summing up the maths marks also needs start, so I guess the start is the same but there should be some change over here.

Because we do not have count as the variable, but we have a new variable which is called sum, so we have to chase this. And similarly instead of increment count, we have to do something else over here. So there has to be some change over here. But it looks like all the other steps are exactly the same, so let us leave them in for, for now, and see whether we can change these two steps and see where that leads us, okay.

So basically we need to make some change to this flowchart and it looks like the changes we to make are to these two boxes. The other boxes seem to be roughly the same. So there is no count variable, we have a sum variable. So we have to make some change to this box. And instead of incrementing count, we need to make some change to this box as well.

So let us see what changes we got to make to these two boxes. So after we start, we are instead of initializing count to 0, we want to initialize sum to 0. So the way to write that in the flowchart is by replacing, to put inside this process box, think sum equal to 0. So that basically initializes sum equal to 0.

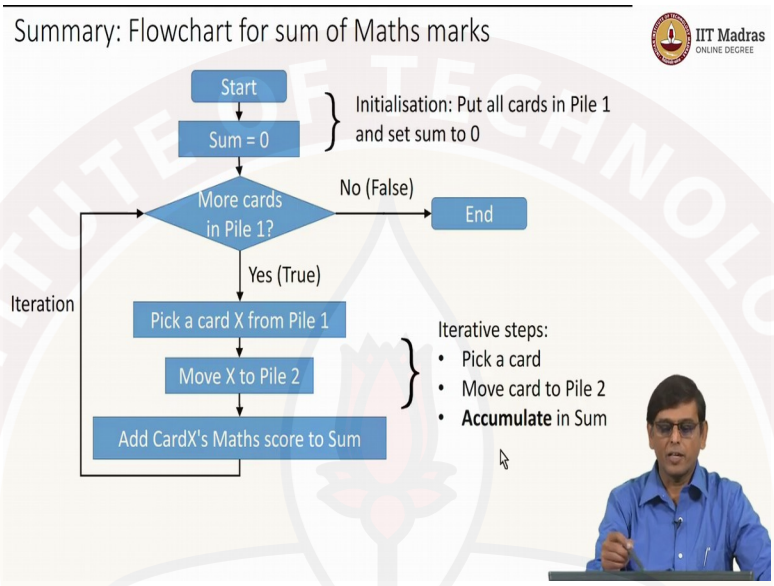
And in this box, instead of incrementing count, we are adding the maths marks of the card that we have seen to sum. So the way to do that is basically is to write something inside this box which says, add card excess maths score to sum. This is some way of doing that, some activity which will basically add to sum, the value of card excess maths, card excess maths score.

So the rest of the flowchart if you can look is the same. So let us just revisit the flowchart and you can see basically the rest is the same. We start, we set sum equal to 0. We see whether there are any more cards in pile 1. If there are not, then we stop. And at this point in time the variable sum will carry the total of all the maths marks in all the cards we have seen.

On the other hand if there are more cards to be seen then we pick a card X from the pile 1, move it to pile 2, so that we do not see this card again. And then we add, from that card X we look at

the maths score and add that maths score to sum. And after we have done that, we go back to this stage, this, this stage because this is the point at which we are again going to check whether there are more cards in pile 1 or not. If there are no then we stop. Otherwise we will repeat these 3 steps again. So this is the iteration process.

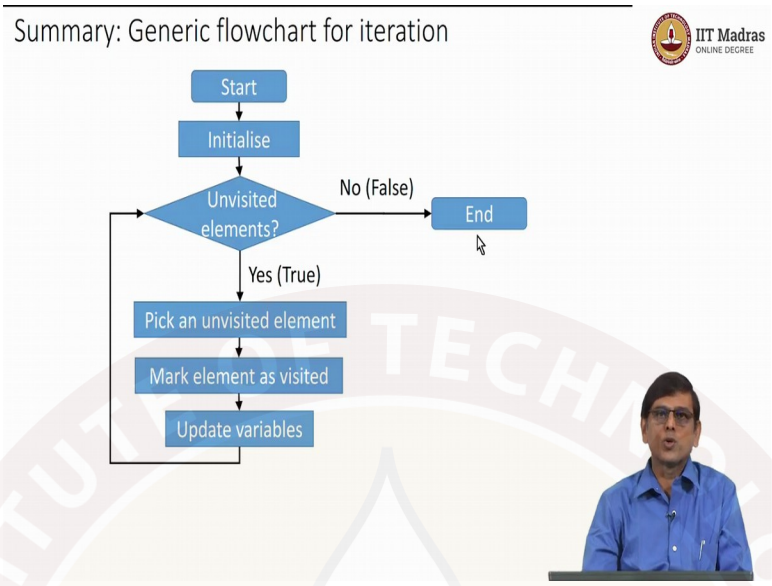
(Refer Slide Time: 13:42)



So you can see again here that the sum of maths marks also has an initialize step in which we are initializing the variable sum to 0, and it has an iterative process. The iterative process has 3 steps inside it, and this is basically to pick a card to move it into another pile and to add the maths score. And the third one, add to maths score is what we called an accumulate right, an accumulate because the sum is accumulating the value of all the maths scores.

So we are seeing picking a card, moving the card and accumulating, these are the 3 steps that we are seeing and that is being iterated many times. So from the two flowcharts that we have seen so far, one for count and other for sum, it looks like both these flowcharts are very similar in nature.

(Refer Slide time: 14:33)



So one could ask the question whether there is a general flowchart for iteration? So iteration itself is there a general flowchart. It looks like there is. So you start the, start the process of iteration, you initialize the iteration and inside the things over which the structure over which you want to iterate, in this case it is the set of cards, you see whether there are any unvisited elements or not.

Any cards, in our case basically the way to look at whether there are unvisited elements is to look at whether there are any cards in pile 1. If there are unvisited elements, then you basically pick an unvisited element, mark it as visited and our way of marking it as visited was to move it into another pile, which is pile 2.

And then you do update some variables which are representing some status of what is going on. So in the case of count, this was basically incrementing count. In the case of sum, this was accumulating the value in sum.

And then you go back to this decision check of unvisited elements, and you keep doing this iteration till such a point in time where there are no further unvisited elements. So at this point you can stop. So this is a generic flowchart for any iterator and hopefully we should be able to use this flowchart repeatedly when we see more requirements for iteration.