



IIT Madras

ONLINE DEGREE

Computational Thinking
Professor Madhavan Mukund
Department of Computer Science
Chennai Mathematical Institute
Professor G. Venkatesh
Indian Institute of Technology, Madras
Lecture 4.8
Summary of Contents Introduced in Week 1 to 4

So, we have now finished four weeks of classes in this course, so it might be a good idea to take stock of where we are, what we have studied in the four weeks, especially since what you studied in these four weeks will also make up for the content that you will have for the qualifier exam which is coming shortly. So, take a quick review of all the things that we have studied so far.

(Refer Slide Time: 00:42)

Iterators and Variables

- The **iterator** is the most commonly used pattern in computational thinking
- Represents the procedure of doing some task repeatedly
 - requires an initialisation step,
 - the steps for the task that needs to be repeated,
 - and a way to determine when to stop the iteration
- **Variables** keep track of intermediate values during the iteration
 - Variables are given starting values at the initialisation step
 - At each repeated step, the variable values are updated
- Initialisation and updates of variables are done through **assignment statements**



Summary of concepts introduced in weeks 1-4

2 / 12



We started with the idea of an iterator, the iterator, which is basically about going through a number of steps in a repeated way is in some sense the most commonly used pattern that we will use in computational thinking. We will see later as we go through the course. We will see some other patterns which are also sometimes occasionally used but among all the things that are that you might find in computational thinking iterator or iteration or doing something repeatedly is the most common pattern.

So, that is the first thing that is why we started with iterator in this particular course the first lecture itself and what it does basically iterator is it represents a procedure of doing some task

repeatedly, which means you have to start by initializing some variables, you have to start with an initialization step and then you have to number of tasks which are basically that have to be repeated there it continuously done one after the other.

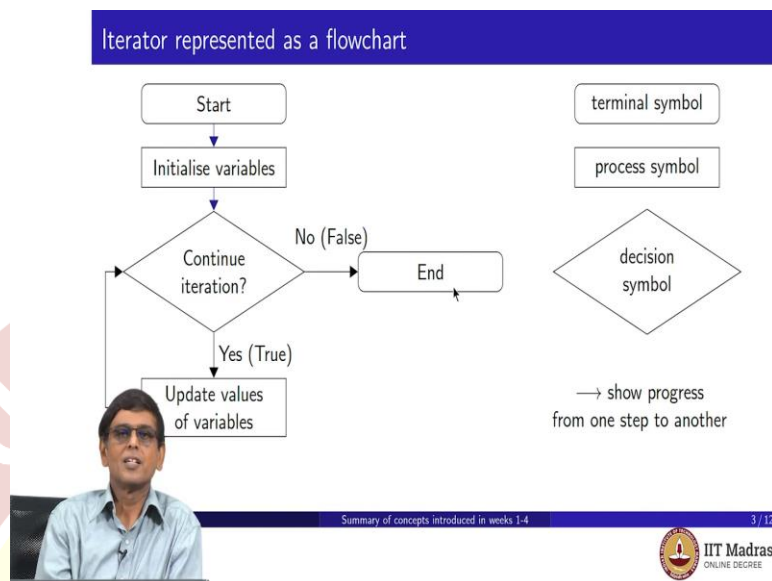
And there has to be a way of determining when to stop. So, you have to start by initializing you have to do something repeatedly and then you have to know when to stop. So, this process of starting and then repeating something and then stopping is what you might call as iteration. So, one of the things that we need to do while iterating on a on something is to keep track of values intermediate values.

So, we did that by using this concept called variables. So, variables basically are things that whose values change during the steps. Whereas constants if you look at the fields of the data elements that we had those fields values do not change but the variable values they keep changing as we do the step.

So, at each step at each stage in each iteration step, we basically look at the elements that are there examine the elements are there and then use those values that are of the elements to update the variable values. So, the variable values in some sense keep changing and when we come to the end of the iteration the variable values will hold some total final value, which is meaningful to us, For example, the sum of all the marks or something like that.

So, initialization of the variables. So, when you say initialization it basically means initialization of the values of the variables. So initialization of the variables and the update that we do to the variable values at each iteration step, we saw a basically is can be done through what one might call as an assignment statement. So, an assignment statement basically is used basically to assign a value to a variable.

(Refer Slide Time: 03:28)



So, these are also basically that this process of iteration can be expressed very nicely through the concept of a flow chart. So, flow chart basically if we recall quickly as number of symbols, but the symbols that we used for our flow charts there are only three of them, one is a terminal symbol. So, specifically the start and the end of the flow chart basically I represented so the start and the end are represented by terminals symbols. In terminals symbol is a box with a rounded edge rounded corner. So, it is got box with the rounded corner is a terminal symbol and the idea was a doing something with the variables.

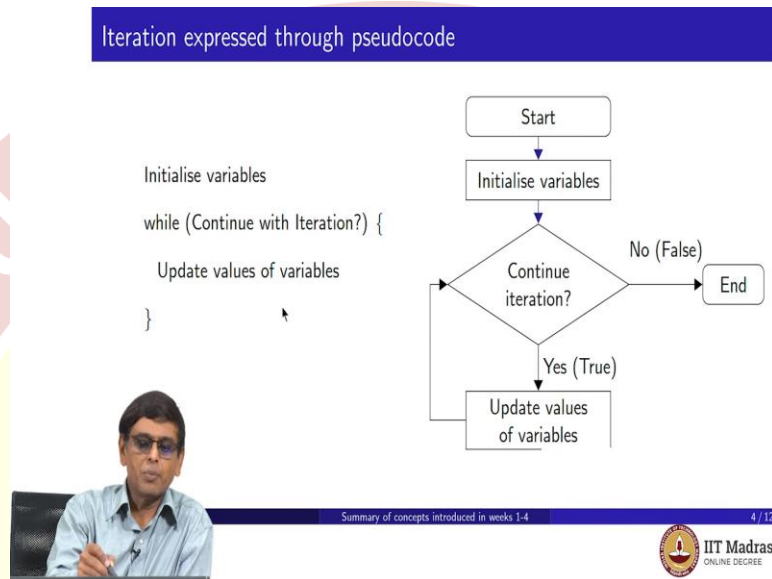
So, when you are doing an assignment statement to initialize the variable or when we are doing assignment statements to update the values of variables, these are written in square boxes and these are called process steps. So, the symbol for process is a square box and some in the middle you have these diamond shaped boxes which we use for basically to make decisions.

So, specifically the one decision we have to make for iteration is to decide whether to continue the iteration or to stop the iteration. If you are going to stop the iteration then you go to the end symbol you are going to continue with the iteration then basically update the variables and then you go back.

So, with some very simple symbols and arrows you can actually depict what is going on in the computation and the iteration is a very simple pattern. So, whenever you see something that looks like this a box to initialize a diamond a box to update and then an exit then you know

basically that this shape this the this shape diamond shape represent an iterator represents an iterator. And any time you see a shape like this you can identify that as an iteration pattern and use it however you want.

(Refer Slide Time: 05:22)



Now, we also saw basically that this flow chart the iteration flow chart can also be written textually because the flow charts are not very convenient to communicate, to a computer or to somebody else you would like to write it in textual form. So, the textual way of depicting the flow chart is what we call pseudocode and the pseudocode has again some assignment statements to initialize the variables and then it has this construct which is called while.

So, while a condition there is a condition in this case basically is whether you should continue with iteration or not and then this curly brace and inside the curly brace you have the update steps, whatever update you are doing to the variables. So, the same flow chart this diagram that we draw here can be written succinctly very nicely in terms of pseudocode like this assignment statement for initializing variables, while statement and then assignment statements to update the variables.

So, whenever you see, pseudocode pattern which looks like this there is some assignment followed by a while and some assignment inside the while that represents a iterator pattern written in pseudocode.

(Refer Slide Time: 06:32)

Iteration to systematically go through a set of items

```
Initialise variables
while (Pile 1 has more cards) {
    Pick a card X from Pile 1
    Move X to Pile 2
    Update values of variables
}
```



Summary of concepts introduced in weeks 1-4

5 / 12



Now, we specifically use iteration to systematically go through a set of items. It could have been a bunch of cards representing the marks of students. It could have been a bunch of cards that represented the words in a paragraph or it could be cards that where the shopping bills represent the shopping bills. So, those are the kinds of cards that we had. So, whenever we went through this item systematically the, we could do various things with those cards and specifically for example, you could count the number of cards, you could find the sum of all the marks or something like that.

So, there would be variables that you would initialize or you will set to 0 in the beginning and then there is a while statement which basically is the iterator pattern and the condition that we used in this case was to check whether the pile from where we are drawing cards is empty or not. So, while that pile has still has cards you will continue the iteration when the pile becomes empty, you will stop iteration.

So, the condition basically was to check whether pile has some cards are not and in terms of the steps update steps. So, besides updating value the variables you have to do a couple of other things in order to make sure that you are not seeing the same card more than once. So, we picked a card X from the pile and moved it to another pile.

So, we had these update variable update steps and we had some you can say update steps which are which were making to the pile. So, there is a pile of cards which also represents something

and they are making some updates to that pile. So, we are updating the pile and you are updating some variables. So, that is how we basically did used iteration to go through systematically go through all the cards in a deck and then using going by going through we were able to do things various things with it.

(Refer Slide Time: 08:16)

The set of items need to have well defined values

- Sanity of different data fields of the item
... leads us to the concept of **datatypes**, which clearly identifies the values and allowed operations
- Basic data types - **boolean, integer, character**
- Add to this **string** data type
- **Subtypes** put more constraints on the values and operations allowed
- Lists and Records are two ways of creating bigger bundles of data
- In a **list** all data items typically have the same datatype
- Whereas, a **record** has multiple named fields, each can be of a different datatype

Summary of concepts introduced in weeks 1-4 6 / 12

IIT Madras
ONLINE DEGREE

Now, one of the things that we wanted to make sure while we are going through the set of cards was to ensure basically that the cards are in good shape, which means that they are not invalid item sitting in there because otherwise what will happen is each time we are trying to do some iteration step. You have to first check whether the card is okay, whether the fields are okay, whether the values are okay and then whether the operations that you can perform can indeed be performed and then you have to do it. So, that is too many checks to do.

So, it would be nice if somebody guaranteed to us that the cards were in good shape the data was seen and so on and that led us to the concept of a data type and we saw that the basic data types were Boolean which can have two values true and false. The integer data type which can take negative 0 or positive values and the character data type which can take alpha numeric character values, which means A to Z in capital or small, 0 to 9 digits or it could have special characters like full stops, semicolon and so on.

And we added to this the string data type, which is a sequence of characters and we defined the definite notion of a subtype which basically puts additional constraints on the values that can be

taken and the operations that can be allowed. For example, we said gender which is M or F is a character data type but only two values are allowed M or F and the operations also are limited to only equal to for example.

So, with subtypes, you can basically put more constraints and ensure more sanity. Then we saw that there are two ways to aggregate data types together. One of them is called list where we typically have a single type of data that we are making a collection of. So, for example you can make a collection of marks cards. You can make a collection of marks just the mark physics marks or you can make a collection of customer names or shop names something like.

And that is called a list and record basically is like what you see on a card typically so that is called record. So, it has name fields, multiple name fields each field has a different possibly a different data type. So, for example the marks card had several marks, it had a name of a student date of birth of the student at things like that.

(Refer Slide Time: 10:30)

Iteration with Filtering

- **Filtering** makes a decision at each repeated step whether to process an item or not
- This introduces a decision step within the iteration loop
- Expressed in pseudocode, it would look something like this:

```
Initialise variables
while (Continue with Iteration?) {
    ...
    if (condition is satisfied?) {
        Update some variables
    }
    ...
}
```

Prepare final results from variable values

Summary of concepts introduced in weeks 1-4

7 / 12

IIT Madras
ONLINE DEGREE

Then we went from there to the idea of filtering. Filtering basically means that while we have the iteration pattern which means that we have doing something repeatedly inside the iteration you can actually check for some condition and decide whether to do the iteration step for this particular data element or not to do that step for the data limit. So, you can there is a decision item sitting inside the iteration loop.

So, expressed in pseudocode, it basically looks like this you start with some value initialization for the variables then you do this while which is basically the iteration loop and inside the iteration loop you have an if statement, you have an if some condition you are checking and then using that condition to decide whether you want to update variables are not. So, for example, you may only add the boy's marks for example, so you are checking whether the card is a boy card in that case you will add the marks to sum something like that.

So, this is called filter iteration which means that from the cards you are selecting some subset of cards and doing the operation only for that subset and whatever final results you get whatever variable values you have you can use that to determine the final result. For example, I can compute the average of the boy's mark I can find the average of the girls mark and then I can find out whether the boy's average is better than the girl's average or the girls average is better than the boy's average.

(Refer Slide Time: 11:47)

Iteration with Filtering

- **Filtering** makes a decision at each repeated step whether to process an item or not
- The filtering condition can compare the item values with a constant
 - ⇒ The filtering condition does not change after each iteration step (is constant)
 - Example: Count, Sum
- Or, it could compare item values with a variable

Summary of concepts introduced in weeks 1-4 8 / 12

IIT Madras
ONLINE DEGREE

So, we essentially filtering is making a decision about the element at each step and then based on that condition it is determining whether or not to do something. So, that condition basically will compare the data element and with something so you could compare it with a constant or you could compare it to the variable. So, when you compare it with a constant, it means basically that the filtering condition itself that is there inside the iterator is not changing with every iteration step.

The same condition is used. So, the same filter method is used at each step of the iteration. So, you are comparing with the constant. For example, you are checking whether something is greater than 25. So, extract all the mark which are higher than 25 let us say for example. So, you are comparing with 25 and that comparing the 25 is not changing from one step to another.

So, examples of using this basically is like counting the boys marks adding up the boys marks and so on. These kind of things basically are examples of where the comparison with a constant, constant means you comparing whether the gender is male for example this case. Sometimes we compared with a variable.

So, when we compare with the variable, it means that the filtering condition is changing with every iteration step because when we compare with the variable the variables value is changing with each iteration step and therefore the filtering condition is also changing with every iteration step.

So, if you are doing different filtering at different steps and that way in which you doing different filtering is by comparing with a variable then you get something slightly more interesting like for example, you can find the max of all the marks by doing this because you will be comparing each time with the max variable which is a variable. So, you are comparing the marks on the card with max if the value of the card is higher than max then you will replace max with the value of the card something like that.

(Refer Slide Time: 13:40)

Procedures and parameters

- Sometimes we have to write the same piece of code again and again with small differences
- A piece of pseudocode can be converted into a **procedure** by separating it out from the rest of the code
- Some variables (or constants) used in this piece of code can be replaced by a **parameter** variable
- Instead of writing the code again with a small difference, we now just have to make a **call** to the procedure with a different parameter value



Summary of concepts introduced in weeks 1-4

9/12



So, we saw basically that we can compare with constants. We can compare variables and some of these comparisons we may have to do again and again, this is another thing we noticed. Sometimes you have to write the same piece of code again and again only thing is that when we are doing the code again we have some small differences.

Like for example when you finding the maximum of physics marks and then we want to find the maximum of chemistry marks there is only one place in the code where something is different everywhere else it is the same. So, when there is only one or small amount of change between one way one when you use the code again than it makes sense to write that piece of code as a procedure.

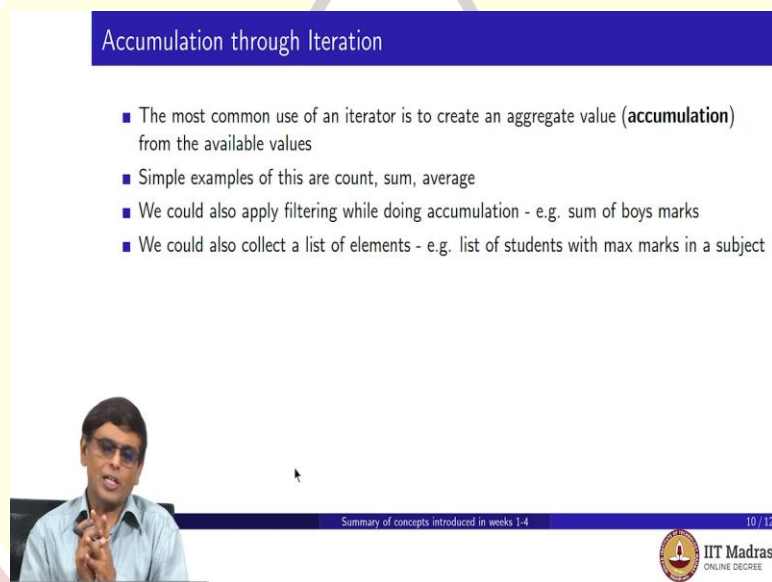
So, you write it as a procedure and separate it out from the rest of the code and the part which is changing you call it a parameter and make it a parameter for the procedure and this parameters value can then be changed for each what we call, call to the procedure. So, instead of writing the code again and again with a small difference you now make a call to the procedure with a different parameter value this is another thing we saw.

So, effectively basically what we have done we have gone from iteration and variables and comparing the variable comparing the using filtering which means comparing with constant comparing variable and using that for a number of things.

Now, we have figured out basically that the code is getting too complex. So, we want to take some parts of the code and separate it out into some procedure and put parameters to this procedure and call this procedure so the code becomes a lot easier to read become smaller to read.

So, this is the next thing that we did, for example, if you want to find the max for each subject, we describe max as a procedure and then we let the subject we have parameters so, you can find the max for chemistry, max for physics, max for biology and then you can add all these things and then you can find the max total and then you can see whether the addition of the max's is less than the total and how much is it and so on, you can do all that.

(Refer Slide Time: 15:34)



Accumulation through Iteration

- The most common use of an iterator is to create an aggregate value (**accumulation**) from the available values
- Simple examples of this are count, sum, average
- We could also apply filtering while doing accumulation - e.g. sum of boys marks
- We could also collect a list of elements - e.g. list of students with max marks in a subject

Summary of concepts introduced in weeks 1-4 10 / 12

IIT Madras
ONLINE DEGREE

Now, the most common use of an iterator is to create some kind of an aggregate value. So, you make a pass through the cards and you do some aggregate, like for example, you want to find the average total or the maximum total or something like that. So, you want to do something like that. So, this business of taking all the cards and summarizing the cards into a single set of values is what we might have studied in statistics course, we may have studied this it is called descriptive statistics.

So, you can find the average of bunch of cards you can find the range which means the minimum and the maximum of the set of cards. So, these parameters like the average or mean value or the

minimum maximum value and the range value or whatever it is, these kind of descriptors are a measure of what is happening for the entire card deck.

So, we could also do other kinds accumulation like for example, we could accumulate the list of students who satisfies the property. Like for example, they you know you want to find out who all have scored the maximum mark in physics. So, that might not be one student many students may have got 92 for example in physics. So, you collect all the students, list of all the students who got maximum marks. So, that is another way of accumulating accumulate all the students who satisfy some condition.

(Refer Slide Time: 16:50)

Doing two iterations - one after another

- Use the first iteration to do some accumulation
- The variables in which these accumulations are done can be called accumulators
- Second iteration can do filtering using the accumulator variables
- e.g. find above average students - average is an accumulator from the first iteration
- This establishes a relationship between any element and the aggregate of all elements
- e.g. find out the more frequently occurring word, higher spending customers, etc

Summary of concepts introduced in weeks 1-4 11 / 12

IIT Madras
ONLINE DEGREE

So, the variables that typically accumulate things they are called accumulators, like some is an accumulator because it captures the some value of all the marks and so on. So, you can capture these aggregate values this is descriptive statistics values in this accumulator variables and then you can compare in the bunch you can compare students against this descriptive statistics. For example, so the first pass you go through all the cards and then you compute these accumulator values and in the second pass you go through these cards and you compare the students against these accumulated values.

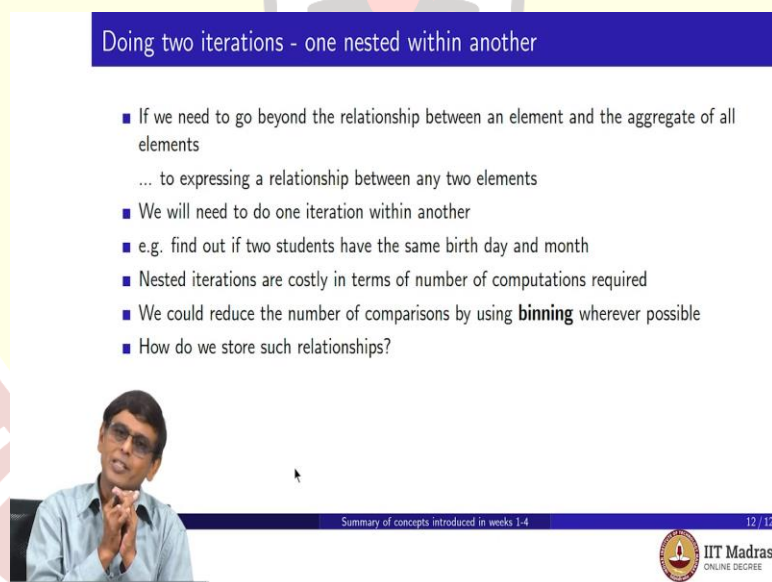
So, for example I want to find out whether the students are above average who which students are doing better than average and so on. So, the first pass I compute the average the second pass I go through all the cards and pick out all the students whose marks are above average. So, this

gives you a pattern in which there are two iterations steps. First iteration step is used to collect some accumulator values and the second iteration step is to compare the elements with the accumulated values.

And then decide basically to bucket them into one group another above average below average or for example in the first pass you may find the frequency of words and then you able to second pass you may find high frequent words words occurring with higher frequency and words which are occurring with lower frequency like that.

So, you can decide based on the first iteration the accumulations that you have done the first iteration, how to bin or classify the elements into different categories. So, in some sense the one iteration followed by another iteration is allowing us to write a pattern where we are comparing each element with the aggregate. So, we are seeing for example, whether a student is above average which means you are comparing a student with the aggregate value.

(Refer Slide Time: 18:31)



Doing two iterations - one nested within another

- If we need to go beyond the relationship between an element and the aggregate of all elements
... to expressing a relationship between any two elements
- We will need to do one iteration within another
- e.g. find out if two students have the same birth day and month
- Nested iterations are costly in terms of number of computations required
- We could reduce the number of comparisons by using **binning** wherever possible
- How do we store such relationships?

Summary of concepts introduced in weeks 1-4 12 / 12

IIT Madras
ONLINE DEGREE

So, sometimes we saw it is not enough to compare a student with an aggregate value. We may need to compare one student with another student. So, the most interesting example that we saw over here was to find out whether two students had the same date of birth date of birth meaning the same date and month of birth, same day and same month they were born. So, if you want to do something like that, then we have to compare every card with every other card it is not enough to compare the card with average date whatever that means.

You can try to find the average date or something and then you can see whether the card is higher than or lower than the average date that does not help. So to find out whether two cards have the same date of birth we have to compare every card with every other card. So, this kind of thing where we have to make establish relationship between every card and every other card requires us to do nested iteration means you have to put one iteration loop inside another iteration loop.

So if you look at it in the flow chart, it basically means that you will have this outer loop and inside that outer loop you will have an inner loop or if you see in a pseudocode, you will have a while condition open curly brace close curly brace and inside the body of that while you will have another while with open and closed curly brace.

So, you have one iteration inside another iteration. So this is as we have seen very costly because in order to do this, you have to compare every element with every other element and so a large number of computations are required and could try and reduce the number of comparisons by binning wherever possible, we saw that you could reduce the number of comparisons in this birth date problem by looking only inside one month so you can bin by month and then do the comparison so that reduces of number of comparisons.

Now, one question this all leads us to basically is if I have to compare every element with every other element, I might as well do this systematically this comparison and store away this relationship between the variables in some data structure someplace somewhere. So, that I can use this relationship later on for other computation just like we did this aggregate value.

So, we had one pass after another pass, so first pass we computed average or something like that and in the second pass we use that average to find out who is above who is below average. The same way I can do one nested iteration and in that nester iteration I can find out this relationship between the variables and once I found this relationship between variables I want to store this relationship somewhere and then use this relationship in other computations.

How do I do that? Now for that I need some method of storing relationships. Which we have not discussed in these four weeks lecture. So, that gives you a prelude to what we will be discussing in the next four weeks. So, the coming four weeks is all about how to store these relationships properly and so that we can use these relationships to do further processing. So, with that we

come the end of the first module which is the first four weeks and hopefully you prepare well with all the material that you have with you and write the qualifier exam very well. All the best.

