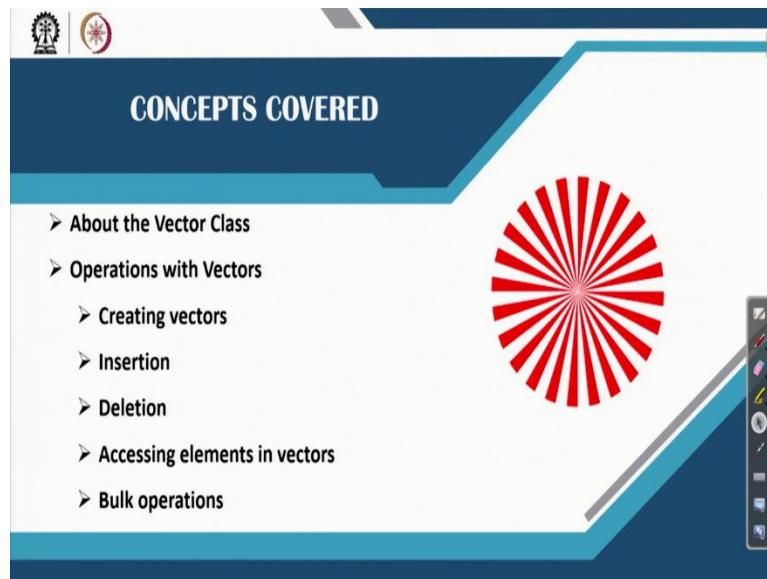


Data Structures and Algorithms Using Java
Professor. Debasis Samanta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture No. 15
Vector Class for Arrays

In this lecture we shall try to learn one very old concept in Java to store a collection more comfortable way and this is called Class Vector. So, our today's topic is how an array or a collection can be maintained using a class vector. The vector class like other collection as you have already learned in the last few lectures, it is defined in the Java dot util package.

(Refer Slide Time: 1:06)



Now, our today's topic includes a basic introduction to vector class. Then there are several operations those are possible on this class namely how to create a vectors, how insertion, deletion and other necessary operation like searching and then copying and everything can be done with this class utility.

(Refer Slide Time: 1:32)

About class Vector

- There are some popular Java utility classes, which are either obsolete or deprecated. There are many such classes, which JDK 5 has retrofitted to build very popular collections. Such classes are called legacy classes.
- Dictionary
- Hashtable
- Propertioe
- Stack
- **Vector**
- The legacy classes are defined in `java.util` package and the class `Vector` is one important of them.

NPTEL Online Certification Courses
IIT Kharagpur

Now, let us see exactly what is this? What is the this vector class is basically? Now for vector is, snow you know Java programming is almost 25 years old, but from the very beginning the programmer was comfortable with one class. This class is called vector and in addition to vector, there are few more classes are there like dictionary, hashtable, propertioe, stack and everything.

So, all these classes were to manipulate data structures actually. But later on the Java developer thought that there should be some more what is called more efficient way all those data structure can be managed and they introduced the collection framework.

So, this is around JDK 5 and onwards. But prior to JDK 5, all those classes, they are called legacy classes are known, very popularly used. This like like Java collection framework the these legacy classes also defined in Java dot util package. Now they are called legacy class because they are still in use. In fact, if you see Java collection framework, they are built on, they are built on based on this legacy class only.

So, legacy class people say that they have been deprecated. But if you want, you can use but Java developer do not recommend to use them in our new code that you are going to develop because more smarter utilities are known. So why you should use the old things are there, but still, if you can write program and use those class your program will run successfully, it will compile also no issue.

Now what I want to say is that all those legacy classes were used or have been used to build today's Java collection framework. Java developer used those legacy classes to build the new Java collection framework, in fact. So, that is a class has its own importance. So today we will discuss about this class vector. Let us start about the vector class actually. Now, if you see, we are discussing about array, array is linear collection.

So, if you want to store set of numbers or strings or some set of objects of type say, student or whatever it is. So, they are basically arrays of collection. Now, vector is nothing but an array of collection actually, but it names vector and you know, if you know mathematics, then the concept of vector is there. So, vector is basically is an element or you can say quantity which has number of components or you can say number of dimensions are there.

So, a vector for example, in a 2 dimensional vector consisting of 2 components like x and y. So, usually we represent a vector say A is equals to ix plus gy , where x and y are the 2 components along 2 axis, i axis, g axis like this one. Now, here the vector comes that way because it also has a number of components. Actually, our objective in Java programming is basically to manipulate objects and if you consider an object, object, in fact, a high dimensional things actually all the fields that is there in an objects or we can class definition, they are basically is a components.

So, if a student is has fields say name of type string, integer type sat age, roll number is another string, marks is float, then we can say there is a 4 component, so 4 directions. So, w, x, y, z like. So, this is a vector. So, that concept come here also. So, vector is basically mean for storing objects very nice nice way actually.

(Refer Slide Time: 6:11)

About the Vector as collection

1. All vectors are started with an initial capacity.
2. After this initial capacity is reached, the next time that you attempt to store an object in the vector, the vector automatically allocates space for that object in addition to few extra additional space for other objects.
3. By allocating more than just the required memory, the vector reduces the number of allocations that must take places as the vector grows. This reduction is important, because allocations are costly in terms of time.
4. The amount of extra space allocated during each reallocation is determined by the increment that you specify when you create the vector.
5. If you don't specify an increment, the vector's size is doubled by each allocation cycle.

NPTEL Online Certification Courses
IIT Kharagpur

Now, there are a few more things about the vector which is very interesting like other collection framework that you have studied. So, we can create a vector; vector is a collection of objects we can see right now. Now, if we create a vector, so, whenever we create a vector, this vector can be created with an initial size of it. So, that is called the capacity of the vector. So, initial capacity.

Now, so actually all the collection that is required in programming it should be dynamic in nature. That means whenever you decide that this is a vector of size say, 10 but later on, you thought that , no, the size should be 100 or more than that.

So, automatically there is a need to grow the vector or size of the vector actually. So, a vectors you can create initially with some initial size of it and later on if you want to increase it automatically it will increase where the programmer does not have to do anything that is a great advantage of this one. So, automatically it basically allocate the memory space to fit with the more objects that needs to be stored in a vector.

So, there is a memory requirement. So, from the memory requirement, the efficient memory management is not at all the headache of the programmer, these are great advantage. And there are many other issues which basically very elegantly have been implemented in this vector class, the class realization that is basically called the automatically incrementation and how it will increment depending on your memory, depending on your speed that is required to run an application, you can increment either slowly or in a very fast way.

So, that increment increment of the size also can control. For example, initially you can decide a vector of size say 10 and later on you want to increase it by say 10 each time or say 20 each time, so, 5 each time that you can control it is there. So, these are the things that is very important and another most significant things which basically differs these vector compared to the other classes they are in the Java collection framework is called the synchronization.

So, actually the vector is basically so suitable for multi threading or multi programming, programmer is writing and where the thread is required hand thread can run parallelly. So, if you want to maintain many collections and all these collections you want to share among the different threads. So, the threads equation is possible with vector class which is not possible with other collection framework classes arrays or arrays like this one.

So, these are the very nice things there, that is why vector is still popular among the programmers, because of those features actually. Now, let us come to the discussion about operations with vector.

(Refer Slide Time: 9:13)

Operations with vector

1. Creating a vector
2. Insertion of an item into a vector
3. Removing an item from a vector
4. Accessing a vector and its content.
5. Some frequently used bulk operations

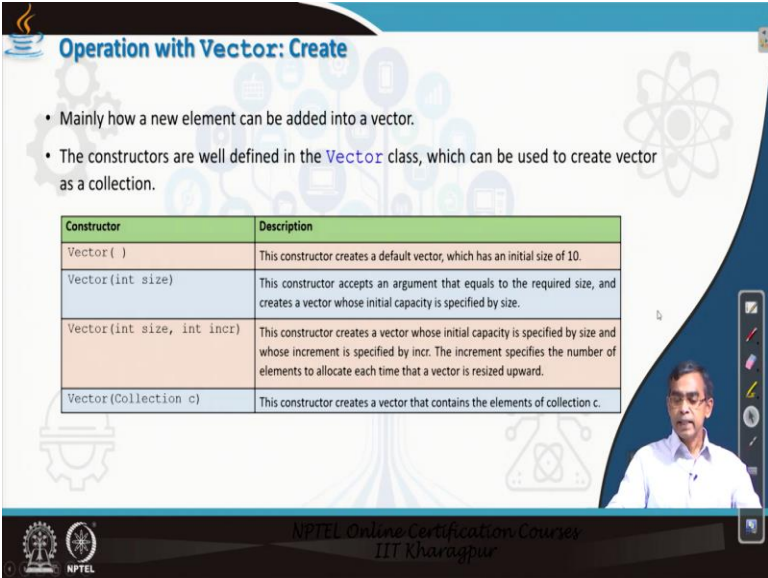
NPTEL Online Certification Course
IIT Kharagpur

Now there are any operations are possible. So, here I list different operation that you can apply on the vector object. So creating a vector, this is our basic operation that you have to how to create a vector of your own to store your own objects. Then inserting an element into a vector, removing an item from a vector is basically division operation, accessing vector and its content so, you can access the vector in a different way.

So, you can access the vector at a particular location or at the end or at the beginning. So, there are things here. So, all those things are built in there. So, you do not have to write the code for that only you have to call that what do you want to do? And in addition to this thing, there are some bulk operations. So, it is basically operation on a particular element and operation as a whole. So, bulk operations.

So, all the bulk operation also can be possible on vector classes. So, we will discuss all the operations in this lecture. Now, let us first discuss about how you can create a vector.

(Refer Slide Time: 10:24)



The slide is titled "Operation with Vector: Create". It contains two bullet points and a table. The first bullet point states: "Mainly how a new element can be added into a vector." The second bullet point states: "The constructors are well defined in the `Vector` class, which can be used to create vector as a collection." The table below lists four constructors and their descriptions.

Constructor	Description
<code>Vector()</code>	This constructor creates a default vector, which has an initial size of 10.
<code>Vector(int size)</code>	This constructor accepts an argument that equals to the required size, and creates a vector whose initial capacity is specified by size.
<code>Vector(int size, int incr)</code>	This constructor creates a vector whose initial capacity is specified by size and whose increment is specified by <code>incr</code> . The increment specifies the number of elements to allocate each time that a vector is resized upward.
<code>Vector(Collection c)</code>	This constructor creates a vector that contains the elements of collection <code>c</code> .

At the bottom of the slide, there is a logo for NPTEL (National Programme on Technology Enhanced Learning) and the text "NPTEL Online Certification Courses IIT Kharagpur".

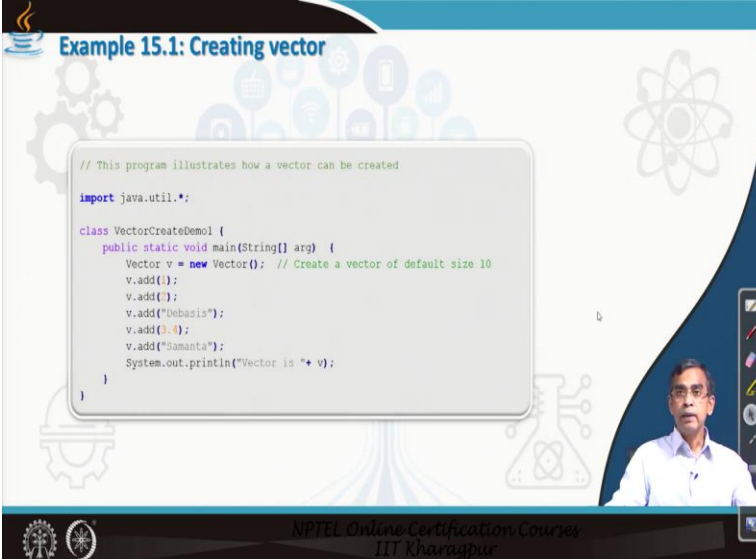
So, creating a vector means an instance of the vector object can be, and there are 3 4 constructors I have listed 4 constructor here. As you can see, the first constructor is called a default constructor which does not require any arguments. So, this basically creates a vector of an default initial size. So, default initial size is 10. The another constructor, which basically allow you to define a vector of a initial size. So, initial size is an argument `int`, so that you can decide. The second vector, another important argument is there in addition to the size in the second constructor, it basically add another argument called increment.

That means each time if the vector has to grow automatically, then by which size that it will increment automatically, so `incr` is there, so initial size may be 100. And if you specify `incr` value as 25, so each time whenever a vector is filled and you want to add few more elements, then it

automatically grows by 25, then 100 to 125, and so on. And then the last constructor is very useful.

If you have already collection at the moment you want to create a vector, then you can pass this collection as an argument and a vector will be created initialize with all elements in the collection. This collection can be anything this collection may be dictionary, this collection, maybe arrays or any other things are there. So, these are the different constructors that is there in the class vector definition and you can use these constructor to create your vector. Now, so, this is basically let us see some example.

(Refer Slide Time: 12:09)



The slide displays a code editor window with the following Java code:

```
// This program illustrates how a vector can be created
import java.util.*;

class VectorCreateDemol {
    public static void main(String[] arg) {
        Vector v = new Vector(); // Create a vector of default size 10
        v.add(1);
        v.add(2);
        v.add("Debasis");
        v.add(3.4);
        v.add("Samanta");
        System.out.println("Vector is "+ v);
    }
}
```

The slide also features a video feed of a presenter in the bottom right corner and the NPTEL logo in the bottom left corner. The footer text reads "NPTEL Online Certification Course IIT Kharagpur".

So, you can understand about how these can be used and this is a very simple example, I made the example very simple so, they can understand at the very first end. So, this basically, as you know this is defined in Java dot util you have to import in every program if you want to use a vector class and here you see I used the default constructor.

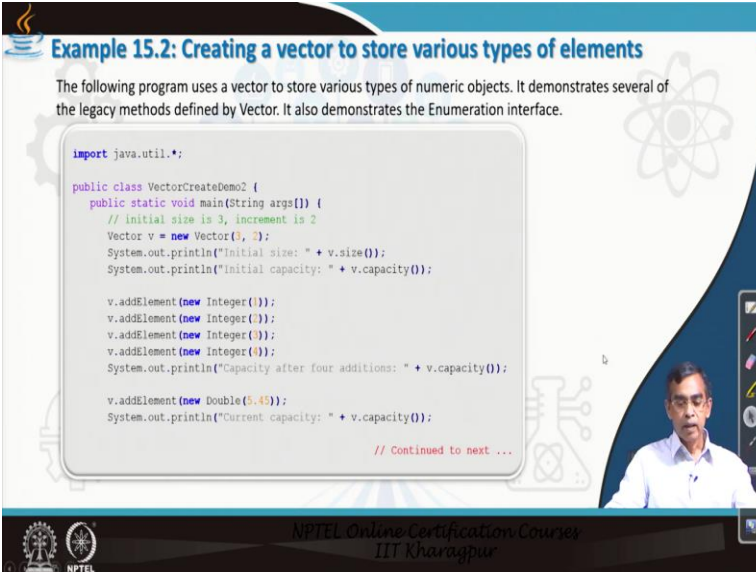
So, v is a vector is created, whose initial size is the default size that means, size is 10 and then few next statement as you see, v that mean this is the vector v and then dot add is the add method. Add method is defined there in the collection and you know the standard method that how you can add an element into a vector. So, with this statement actually starting with vector v is initially empty and then add elements 1 by 1.

So, initially empty and then next element 1 is added 2 and here you can note, this is another object is added. So, the name of the object is a string here, this is a double value, another string it is there. So, we have perform in this example, 5 add operations is basically to initialize the vector v that we have created here. And you can note this statement is very simple statement, the println statement actually println statement print this.

Now, here this v actually here 1 thing you should note that all these 1, 2 now it is string, the double. So, these are the type of different I mean, values of different types actually; elements of different types. No issue because the vector will store them as a string format. And here whenever you print v as an argument of the println statement system dot out dot println, it basically convert all the collections of all elements into a string and it will print.

So, it will print basically 1, 2, and (())(14:15) 3.4 something like this one. So, this is a very simple example that you can understand that how a vector can be created and the vector can be initialized with few elements like 5 here in this case. So, so, if this is a very simple example.

(Refer Slide Time: 14:33)



Example 15.2: Creating a vector to store various types of elements

The following program uses a vector to store various types of numeric objects. It demonstrates several of the legacy methods defined by Vector. It also demonstrates the Enumeration interface.

```
import java.util.*;

public class VectorCreateDemo2 {
    public static void main(String args[]) {
        // Initial size is 3, increment is 2
        Vector v = new Vector(3, 2);
        System.out.println("Initial size: " + v.size());
        System.out.println("Initial capacity: " + v.capacity());

        v.addElement(new Integer(1));
        v.addElement(new Integer(2));
        v.addElement(new Integer(3));
        v.addElement(new Integer(4));
        System.out.println("Capacity after four additions: " + v.capacity());

        v.addElement(new Double(3.45));
        System.out.println("Current capacity: " + v.capacity());

        // Continued to next ...
    }
}
```

NPTEL Online Certification Course
IIT Kharagpur

Let us consider one more example little bit, moderate level of codes are there. So, in this examples, we want to demonstrate that how a vector if you can create store the different type of objects now we are storing objects actually again we will consider very simple object like integer objects. Now here if you see an object 1.

1 is basically the value and if we pass this value to a class integer, so basically they construct an integer, integer class is defined in Java dot Lang Package actually there are many, so object, a number can be created into an object from either integer or string or double or float or whatever it is. So, it is basically 1 is an integer value, and we create an object of this value by using the integer class actually.

So, so this new integer 1 is basically is an object version of the integer 1, fine. Now, here let us see how we create in this example using the third constructor that we have discussed. There are as you see, the 2 arguments we have used. The first argument is basically the size and the second argument is basically the increment that means, initially as a vector of size 3 and then increment is decided by 2.

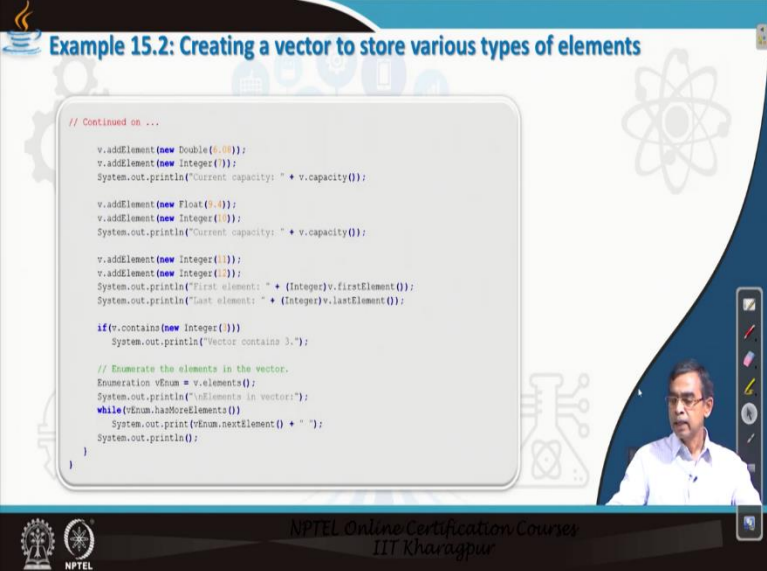
So, each time either vector is already full, then it will increment automatically by size 2. Now, let us come to here. So, these are the two basic statement that you can call for this object v size and capacity. So, what exactly the size it says that what is the total number of elements are there and capacity is basically says that what is the total number that you can load into the vector. So, in this case both the value or the size is 0, initially and then capacity will be 3 line.

Now, if you go on adding, but here we will add elements or objects using some other method not the add method. So, here add element is basically adding an object into the vector v. So, we use the add element rather than inserting some element into the vector actually, these are new methods that we are using here and so, here we see add elements have been called for 4 times into the vector objects v and all the element that is added there 1, 2, 3, 4 like this one, but here if you see, when you come at the point, this point, so, vector is already full, because initial size is 3, and then we are going to add another 1 element.

So, automatically the vector will be increased by size. So, at this point after loading whenever we want to, when you attempt to add 1 more element namely 4 there, so, vector size will be 5 like. So, if you print the v dot capacity, you will see that it will print 5. Now, after adding a limit option, no problem vector need not to be because it will can accommodate the new element because it is 5 so, it will be capacity 5 also it will show this one.

So, so we have understood about that how automatically the vector can grow as the number of elements or the elements, go on inserting into the vector and it is absolutely not an issue for the programmer that you have to is the memory and then accordingly arrange it.

(Refer Slide Time: 18:14)



The slide displays a Java code snippet for creating a vector and adding various elements. The code is as follows:

```
// Continued on ...  
v.addElement(new Double(6.66));  
v.addElement(new Integer(7));  
System.out.println("Current capacity: " + v.capacity());  
  
v.addElement(new Float(9.9));  
v.addElement(new Integer(3));  
System.out.println("Current capacity: " + v.capacity());  
  
v.addElement(new Integer(1));  
v.addElement(new Integer(1));  
System.out.println("First element: " + (Integer)v.firstElement());  
System.out.println("Last element: " + (Integer)v.lastElement());  
  
if(v.contains(new Integer(1))){  
    System.out.println("Vector contains 1.");  
}  
  
// Enumerate the elements in the vector.  
Enumeration vEnum = v.elements();  
System.out.println("All elements in vector:");  
while(vEnum.hasMoreElements()){  
    System.out.print(vEnum.nextElement() + " ");  
}  
System.out.println();  
}
```

The slide also features a small video inset of a man in a white shirt and glasses, and the NPTEL logo at the bottom left.

Let us continue this program again few more insertion carrying on and then automatically as the vector can grow you can see it I am adding more elements here this is another object double, integer, 7 float.

So, any type of objects here you if you can create your user defined object like student I also you can do absolutely no not an issue there. Now, so, up to this part, as we see, we have added a large number of objects into that vector v and vector automatically grow its size as by the printing of this, we will be able to see it. Now let us come to that few more interesting methods, namely the first element and that last element.

As the name implies, you can understand that so if you call the method first element for the vector v, so it will basically access what is the element at the very beginning in the vector and in contrast to the first element, last element, basically, which is the last element in the row or vector, so it is a first element, last element that can be accessed from the vector and can be printed and as this is a first element should be integer form, because it is already stored there in a string.

So, that is how we have to cast casting these type by integer because they are integral. So, casting is important because in which form you want to store because if you create a double value, so you should cast as a double that this element if you want to print for example, so you can similarly if you want to print a string, then you have into cast a string like this one. So, casting is important.

And here is another method, the contents method. We will discuss this content method again later on. So, this method is basically check whether a particular element present in the vector for example, the contents is a method arguments is 3 that means, this statement actually, what it does is basically check whether integer 3 is present in the existing vector v or not. So, if it present it returns these contents return true and if he does not present it will return false and if it returns it is present true, so, it will mean that the vector contents the number or objects it is like this.

Now, if you want to print the entire vector, we have already learned about how it can be done using simple println statement passing the vector as an argument, but there are some other it is called the iterators so enumerator is there So, using the call of enumerator you can call it here actually, you have to declare enumerator object.

This is basically that elements of the so v dot elements indicates that it is enumerator that means I want to count or enumerate each element in vector and then it basically go on printing the elements. So, that for this basically this is basically scanning one after another, so has more elements that basically it is starting from the beginning and go to the next and so on, so on and it will basically print the elements actually.

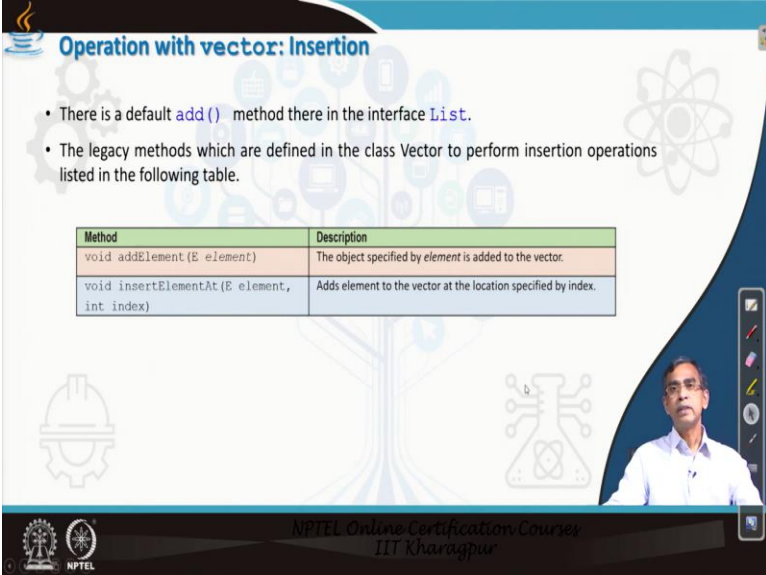
So, next element. So, here, so, while these basically continue till the vector is not come to an end and it will basically print each element whenever it is scanning and it will basically print. So, this way it will print. So, this this is a very 1 structure code you can follow you can practice of your own later on, and you can see exactly how it works. So, it basically scan the entire vector and print each elements in it.

This is 1 another way many programmers prefers it compared to the simple println statement because there no customization no control over how to print but this element you can use you can print in 1 row, 1 column or 3 elements in 1 rows, then so many columns So, many ways that you can show that is why this has its own advantage. So, this is 1 example that you can study that

how vector can grow automatically and then vector can contains many elements in it of different types and all these things can be stored as an object.

So, this is an idea that I want to convey it. Now, let us see, there are some other operations we have already learned about, few operations we have learned like add and then add elements then contents, then first element, last element. Now regarding in insertion, specifically, there are few more operations are known very popular, very popular they are very popular for this class.

(Refer Slide Time: 22:41)



The slide is titled "Operation with vector: Insertion". It contains two bullet points and a table. The first bullet point states that there is a default `add()` method in the `List` interface. The second bullet point states that legacy methods defined in the `Vector` class for insertion operations are listed in the following table.

Method	Description
<code>void addElement(E element)</code>	The object specified by <i>element</i> is added to the vector.
<code>void insertElementAt(E element, int index)</code>	Adds element to the vector at the location specified by <i>index</i> .

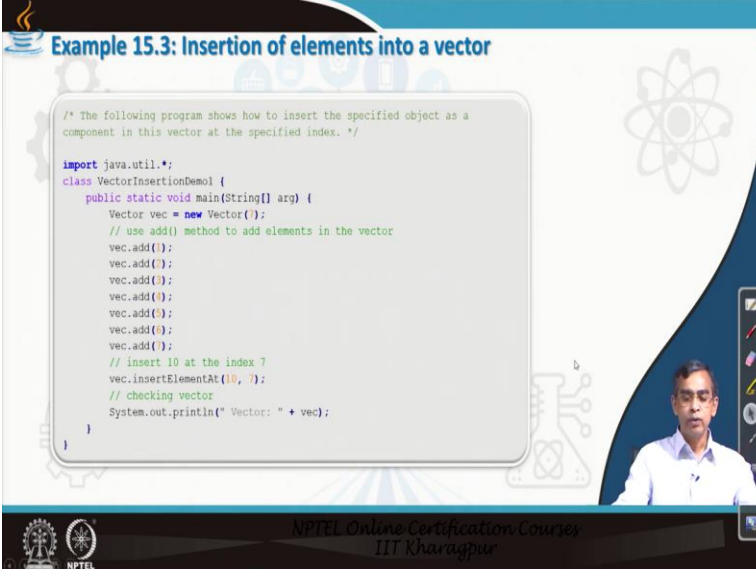
The slide also features a small video inset of a presenter in the bottom right corner and logos for NPTEL and IIT Kharagpur at the bottom.

They are basically add element; we are already have an experience about the add element method in the last example. And in addition to this add element method, there is one method is called, `insertElementAt`, as the name of the method implies that you can insert a particular elements at a particular location of a vector.

Now here for example, in an array if you want to insert at a location say 10 were the size of the array is 5, then definitely it will throw array index out of bounds exception, an error, but here in case a vector, you have created one vector of size say, 10 and you want to insert at elements are 20 absolutely it will not an error because it will automatically grow to accommodate element in a particular location although the different locations remain vacant before this because you are jumping into some location without filling the elements.

Anyway. So, this is the one these are 2 methods that you can think for inserting elements into the vector in addition to the very primitive methods add, add is always there to add any objects into a particular collection. Now, let us see how these method again can be exercised with an example.

(Refer Slide Time: 24:16)



The slide displays a Java code snippet for inserting an element into a vector. The code includes a comment explaining the purpose, an import statement for java.util.*, and a class VectorInsertionDemo1 with a main method. The main method creates a Vector, adds five elements using the add() method, inserts the value 10 at index 7 using insertElementAt(10, 7), and prints the resulting vector.

```
/* The following program shows how to insert the specified object as a
component in this vector at the specified index. */

import java.util.*;
class VectorInsertionDemo1 {
    public static void main(String[] arg) {
        Vector vec = new Vector();
        // use add() method to add elements in the vector
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        // insert 10 at the index 7
        vec.insertElementAt(10, 7);
        // checking vector
        System.out.println(" Vector: " + vec);
    }
}
```

So, this is our third program you can consider how insertion of elements into a vector can be done and as you have see. As you see here, the the add method we have called for this. So, add 1, 2, 3 now, in in addition to the add, we can use the method add element also. Add element will automatically add at the end of the existing objects actually. So, if we call add 5 without mentioning where to add automatically it will add at the end of 4 that means till time this 1 so, it is basically go sequential. Now here in this here we call, insertElementAt'.

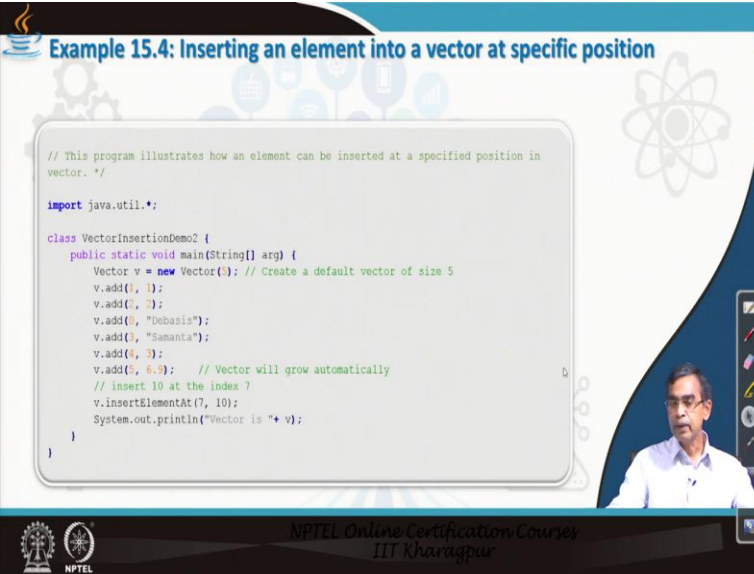
So, as you see, we want to insert the elements insert the element say 10 we want to insert 10 but at` the location 7. Now, here you see. So, these basically element is stored at 1 location, but 0 is there. So, 7 location if you say, so, index 7 is the index 7 is basically this is basically sixth, seventh index. So, so, whenever you call thus index, so, this one in the insert element at 10 7 it will basically come to that.

So, this automatically make an adjustment so that this element will be more one place towards down and it will make a space for the element to be loaded here. So whenever it is there, it is not that it will overwrite, it will basically whatever the movement in the existing element is required,

it will do it and then make an account, make a place for that and automatically adjust that place to fit into there and this is a simple vector, printing.

So, we we after doing this one if you want to print it, then you will see it will print the entire vectors. So, this basically shows that if you can insert any element at any you can insert any element at any place in the vector no issue that vector will be overwritten or it will loss some data it automatically make an adjustment to accommodate the newly inserted element into that vector. So, this is the insertion beauty that you can do it.

(Refer Slide Time: 26:46)



The slide displays a Java code snippet for inserting an element into a vector at a specific position. The code is as follows:

```
// This program illustrates how an element can be inserted at a specified position in
vector. */
import java.util.*;

class VectorInsertionDemo2 {
    public static void main(String[] arg) {
        Vector v = new Vector(5); // Create a default vector of size 5
        v.add(1, 1);
        v.add(1, 2);
        v.add(0, "Debasis");
        v.add(3, "Samanta");
        v.add(4, 3);
        v.add(5, 6.9); // Vector will grow automatically
        // insert 10 at the index 7
        v.insertElementAt(7, 10);
        System.out.println("Vector is "+ v);
    }
}
```


The slide also features the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur" at the bottom. A small inset image of a man is visible in the bottom right corner of the slide.

Without any headache of your own and it will do automatically everything. Now, here is another example. So, again inserting element here you can see this example has little bit different compared to the previous just now, we have learned the example here. Now add element has the 2 poly method. Initially, if you say add 1, it will add the element at the current location.

Now here, if you specify at which location you want to add. So the second arguments, the first arguments says that the in this method add the first arguments say that index, and then the next argument says that what value that needs to be inserted there. So, here 1 needs to be inserted at the location 1, 2 is the location 2. This is another string at the location 0 that means 0 and so on so on. So, these are first arguments show that at what position you can do it and at any order you can do it we will do it absolutely that way.

So, index wise insertion that can be there and this is the example that we have already insert at insertElementAt this way. So, you have learned about how the elements can be inserted into a vector in a different way, we have learned many ways of inserting elements into the vector.

(Refer Slide Time: 28:14)



The slide displays a code editor window with the following Java code:

```
/* To append all of the elements in the specified collection to the end of
this Vector. */

import java.util.*;

class VectorInsertionDemo3 {
    public static void main(String[] arg) {
        ArrayList arr = new ArrayList();
        arr.add();
        arr.add("Oracle");
        arr.add("Java");
        arr.add();

        Vector v = new Vector(); // Creating a default vector
        v.addAll(arr); // copying all element of array list into vector
        System.out.println("vector v: "+ v); // checking vector v
    }
}
```

The slide also features a small video inset of a man in a white shirt and a logo for NPTEL Online Certification Courses at IIT Kharagpur.

Now, another the insertion operation were, were you want to insert a set of elements but in a appending mode. Appending when vector is already content, some elements and you want to add after that, and then this is called the appending. Now let us see one example how the appending that means adding some set of elements into an existing vector can be carried out. And here is an example you can see first we create a collection called arr is a collection of type ArrayList, ArrayList we have already learnt in the previous our lessons.

And here so, these are the basically simple add methods that we can apply into the collection array, ArrayList collection. So, we add 3 Oracle Java 4 that means, the array collection ArrayList collection AR is loaded here. Now, the vector we want to create a vector v, so initialize the vector create this is a default vector constructor that we can call initially the size is 10. We do not have any elements right now, you can add v dot at 3, 4, 5 or whatever you can do you can do fine, no issue.

Now after that, if you call this method, the addAll method, addAll method is defined in the collection framework collection class actually is a method but it is implemented in vector class for that. So, here addAll and passing this arr, then what will happen it basically if the vector v

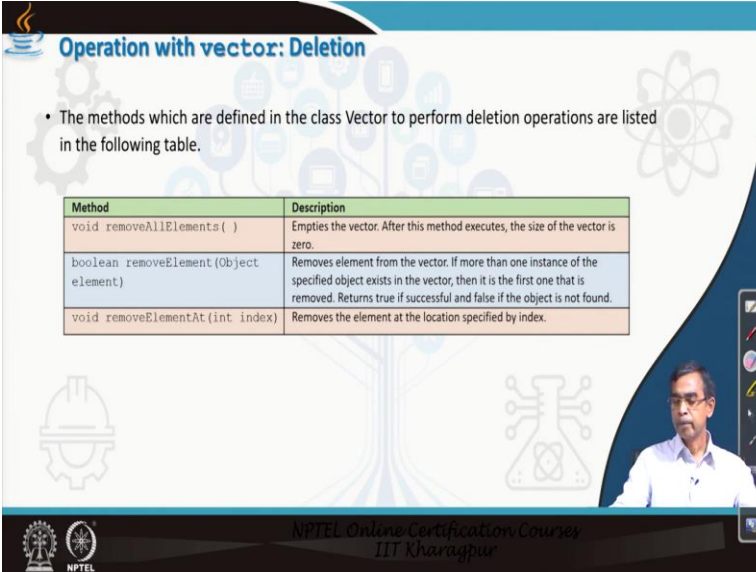
contains initially some elements, then all the elements which are here in this collection will be added after that.

So, these basically, so how appending can be done. Other way also again you can do you can recall there is 1 constructor where an argument can be a collection. So, if you do that, then initially a vector can be created with a collection of elements in it. So, this arr can be passed as an argument here, and you can write this, you can call this vector constructor so that it will basically initialize with this array elements into it.

So, this is a different way that you can do and you can also study practice this program you can modify and you can run it to understand how it works for you. So, these are the different way that the insertion operation or the elements can be added into a vector. Now, let us come to the discussion of deletion operation. So, it is just opposite to the insertion is basically the removing there are many methods for example, you already know clear method.

So, if you can call clear method for a collection, it will basically delete all the existing element into that collection, it is an ArrayList or some arrays or whatever it is there. So, that method is defined in class collection and every collection class has implemented them. So, vector also has implementation some of this one. Now, so, clean is 1 method I just remember.

(Refer Slide Time: 31:43)



The slide is titled "Operation with vector: Deletion". It contains a bullet point stating: "The methods which are defined in the class Vector to perform deletion operations are listed in the following table." Below this is a table with two columns: "Method" and "Description".

Method	Description
<code>void removeAllElements()</code>	Empties the vector. After this method executes, the size of the vector is zero.
<code>boolean removeElement(Object element)</code>	Removes element from the vector. If more than one instance of the specified object exists in the vector, then it is the first one that is removed. Returns true if successful and false if the object is not found.
<code>void removeElementAt(int index)</code>	Removes the element at the location specified by index.

The slide also features a small video inset of a man in the bottom right corner and a footer with the NPTEL logo and text: "NPTEL Online Certification Courses IIT Kharagpur".

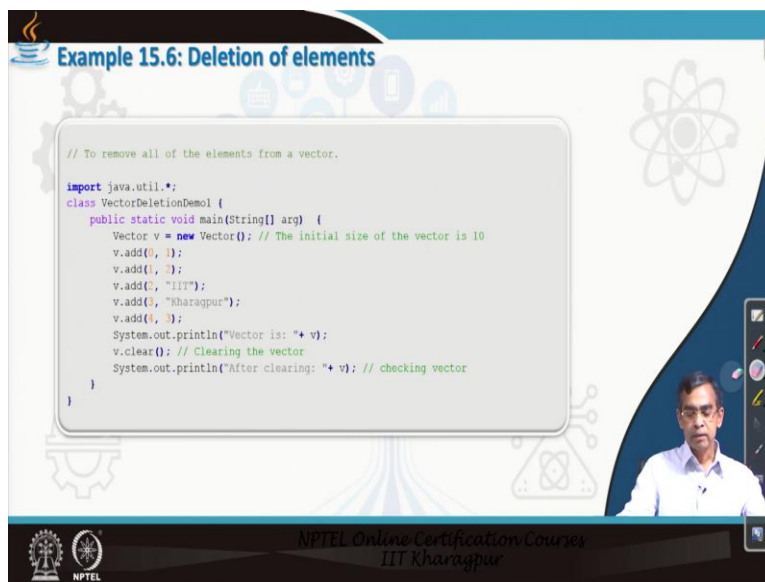
In addition to the clear there are many other methods also. This table lists the different methods for deletion operation. And the name of the method actually, there is a one (com) what is called

the (critic) criticism about these vector class is there. All the name of the method is very big, insertElementAt, removeAllElements like like so, some programmer may not like that the name of the method in any defined classes will be so large like, but it has certain pedagogical issue if you give the name of the method a little bit with more number of characters, it basically gives certain advantage to you that understand that what method does for you.

So, removeAllElements by the name itself, you can understand what it does for you, it basically just like a clean operation. So, this is equivalent to the clean. Now, another operation, remove element passing an argument object, so, if you can pass the object then it will basically search whether the object present there or not, if it present then particular element with its first occurrence will be removed.

Then, removeElementAt, now, in this case you can pass an index. So, at a particular location if you want to remove a particular element, then you can call these. So, these are the few standard methods those are defined in vector class to perform the deletion operation on the vector.

(Refer Slide Time: 33:08)



The slide displays a code editor window with the following Java code:

```
// To remove all of the elements from a vector.
import java.util.*;
class VectorDeletionDemo1 {
    public static void main(String[] arg) {
        Vector v = new Vector(); // The initial size of the vector is 10
        v.add(0, 1);
        v.add(1, 2);
        v.add(2, "IIT");
        v.add(3, "Kharagpur");
        v.add(4, 3);
        System.out.println("Vector is: "+ v);
        v.clear(); // Clearing the vector
        System.out.println("After clearing: "+ v); // checking vector
    }
}
```

The slide also features a small video inset of a presenter in the bottom right corner and logos for NPTEL and IIT Kharagpur at the bottom.

Now, this is an very simple example again, so that how we can perform the deletion. So, now, this part of the code is as usual. You have to create a vector first then add some elements into it whatever the elements you want I have created here around 5 elements into that and then we call the v dot clear is a removeAll actually removeAll equivalent you can again instead of this

method you can v dot removeAll elements also you can and you can see and after this if you can print it will not print it basically it will clean everything.

So, this does not mean any element so, null will be printed here. So, this is very simple but it is very a little bit risky method because it will clean and once you delete it you will not be able to recover it anyway any remove or deletion operation you like this only removing after this thing, there is no way to get the back the same element again.

(Refer Slide Time: 34:07)

Example 15.7: Deletion the first and last element from a vector

```
/* To remove an element at a specified location. It also illustrates the
removal of a specific element. */

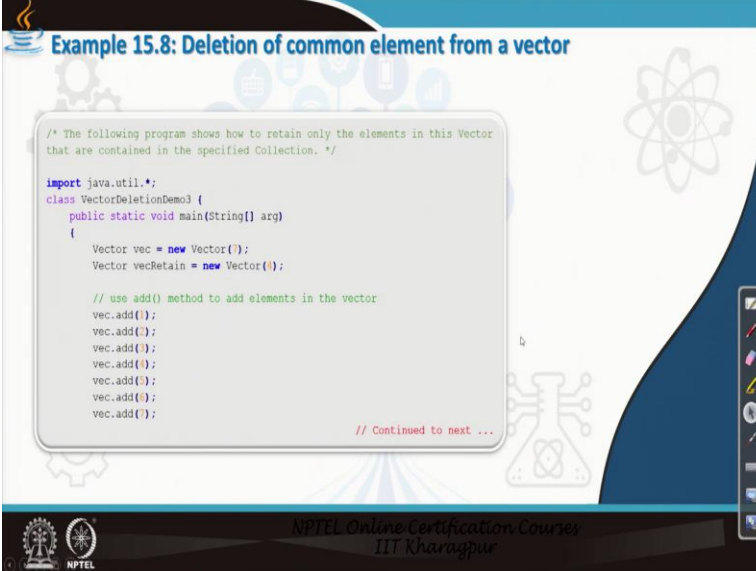
import java.util.*;
class VectorDeletionDemo2 {
    public static void main(String[] arg) {
        Vector v = new Vector(); // Create a vector of (default) capacity 10
        v.add();
        v.add();
        v.add("India");
        v.add("Japan");
        v.add();
        v.removeElementAt(); // Removing the element at 0, if it occurs
        System.out.println("After removal: "+ v); // Checking vector
        v.removeElement("Japan");
        System.out.println("After removal: "+ v); // Checking vector
    }
}
```

NPTEL Online Certification Courses
IIT Kharagpur

And this is one another example. So, these are deletion and deletion from a particular location. So, you can mention the index. So, mentioning the index that is 1 convenient way to remove a particular remove an element from a particular location. So, this example demonstrate how you can do that. Now, this is as usual the code to create a vector first and then you see removeElementsAt. So, this index is 0. So, in this case, so, this is basically the index at 0 so, this will be removed and it will clean after removing this 1 it will clean rest of the elements.

Now, this is the another remove elements by passing the arguments as the object is Japan. So, in this vector that this one is present here, so, it will remove the element and then after removing if you want to print it will print all the elements except Japan. So, this is the way that the different, different ways that the deletion operation can be done on vector.

(Refer Slide Time: 35:19)



The slide displays a code editor window with the following Java code:

```
/* The following program shows how to retain only the elements in this Vector
that are contained in the specified Collection. */

import java.util.*;
class VectorDeletionDemo3 {
    public static void main(String[] arg)
    {
        Vector vec = new Vector();
        Vector vecRetain = new Vector();

        // use add() method to add elements in the vector
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();

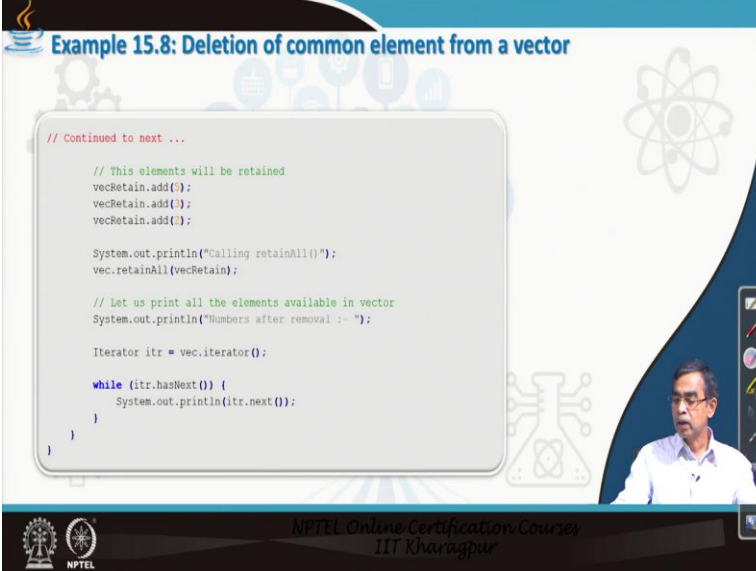
        // Continued to next ...
    }
}
```

The slide also features the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur" at the bottom.

And this is another example, where you can see you can delete elements from a vector, but not all elements is a basically filter element. So, how the filter element can be done, so, it is called you you can say that, some set of elements should be retained there, except all the elements other elements will be removed. So, this is the one example that you can see deletion of common elements from a vector it is just like intersection sort of things It is like that.

Now, let us first see the example then I can explain it. Just let us create 2 vectors, one is vec and another is that vecRetain that means, you want to remove some elements which are there in this, but retaining all the elements which are here in this one. Now, we create initially the vec of size 7 and this is a vecRetain of size 4. Now, we add 7 elements into these vec, so, this is basically adding so 7 elements are stored here. And so, we have created a first vector vic the next set of vectors are vecRetain.

(Refer Slide Time: 36:39)



The slide displays a code editor window with the following Java code:

```
// Continued to next ...

// This elements will be retained
vecRetain.add(5);
vecRetain.add(3);
vecRetain.add(2);

System.out.println("Calling retainAll()");
vec.retainAll(vecRetain);

// Let us print all the elements available in vector
System.out.println("Numbers after removal :- ");

Iterator itr = vec.iterator();

while (itr.hasNext()) {
    System.out.println(itr.next());
}
}
```

The slide also features a small video inset of a presenter in the bottom right corner and the NPTEL logo in the bottom left corner.

So, we can create these `vecRetain` is adding some elements here. Let us add 4 elements or 3 elements we have added here. So, these are basically loading the vector retain so 5, 3, 2. Now, if we can, if we call this method, you can understand what it retains it basically it will remove all the elements from the `vec`, `vec`, but not the common elements that is there, the list 5, 3, 2 will be returned there and except this 5, 3, 2 all the vectors all the elements in this vector removed. So, this is a simple way that the common element can be removed.

Common element in a vector can be removed from other vector like this one. And this is another method by which you can scan this for the sake of difference I have used a different way the vector can be printed this is another way using it is called the iterator in one example previous one example, we have learned about using enumerator the iterator is also one way that it can iterate that means scan or traverse the entire vector.

So it hasNext and this next is is basically hasNext, check that whether we reach the end and if we do not reach, it we will go and and then the current element will be printed at each time it traverse from starting element to the last element like this. So, this basically shows about the removal operations from the vector and there are a few more operation is called accessing elements from a vector from different there. There are a couple of methods are there.

(Refer Slide Time: 38:22)

The slide is titled "Operation with vector" and features a table of methods for the Vector class. A presenter is visible in the bottom right corner of the slide.

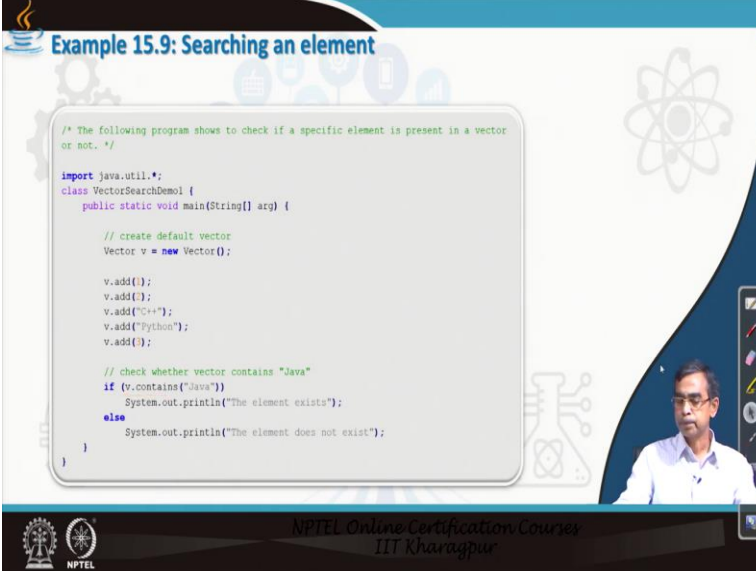
Method	Description
int capacity()	Returns the capacity of the vector.
boolean contains(Object element)	Returns true if element is contained by the vector, and returns false if it is not.
E elementAt(int index)	Returns the element at the location specified by index.
Enumeration<E> elements()	Returns an enumeration of the elements in the vector.
void ensureCapacity(int size)	Sets the minimum capacity of the vector to size.
E firstElement()	Returns the first element in the vector.
int indexOf(Object element)	Returns the index of the first occurrence of element. If the object is not in the vector, -1 is returned.
int indexOf(Object element, int start)	Returns the index of the first occurrence of element at or after start. If the object is not in that portion of the vector, -1 is returned.
boolean isEmpty()	Returns true if the vector is empty, and returns false if it contains one or more elements.
E lastElement()	Returns the last element in the vector.
int lastIndexOf(Object element)	Returns the index of the last occurrence of element. If the object is not in the vector, -1 is returned.
int lastIndexOf(Object element, int start)	Returns the index of the last occurrence of element before start. If the object is not in that portion of the vector, -1 is returned.
void setElementAt(E element, int index)	The location specified by index is assigned element.
int size()	Returns the number of elements currently in the vector.

I have listed very commonly used methods those are defined for class vector. Here for example, capacity we are already familiar to, then contains also we have already discussed, elementAt also sometimes we have discussed about. First element, indexOf the, by name actually you know exactly if you can call a method for this particular vector objects and what it does and automatically argument you can cast.

So, my advice is that you can run the programs of your own to exercise how long does methods works for you it is easy, so you have to create vector first add the vector with some initial element and then apply all those method and you can see how it results. So, you have to print you can use any iterator or enumerator or in simple system dot out dot println dot print v after each operation.

For example, if you call call isEmpty, so, if you can check isEmpty returns true Boolean value, so, is that true means it basically empty false means it is not that it contains some element like this one. So, all those methods are very simple, easy to learn, easy to use, easy to apply in your program. And you can see it I just left as an exercise, because time will not permit to give illustration for each methods and discuss it, but it is easy. So you can try you can run your hand and then you can see it.

(Refer Slide Time: 39:43)



Example 15.9: Searching an element

```
/* The following program shows to check if a specific element is present in a vector or not. */
import java.util.*;
class VectorSearchDemo1 {
    public static void main(String[] arg) {

        // create default vector
        Vector v = new Vector();

        v.add();
        v.add();
        v.add("C++");
        v.add("Python");
        v.add();

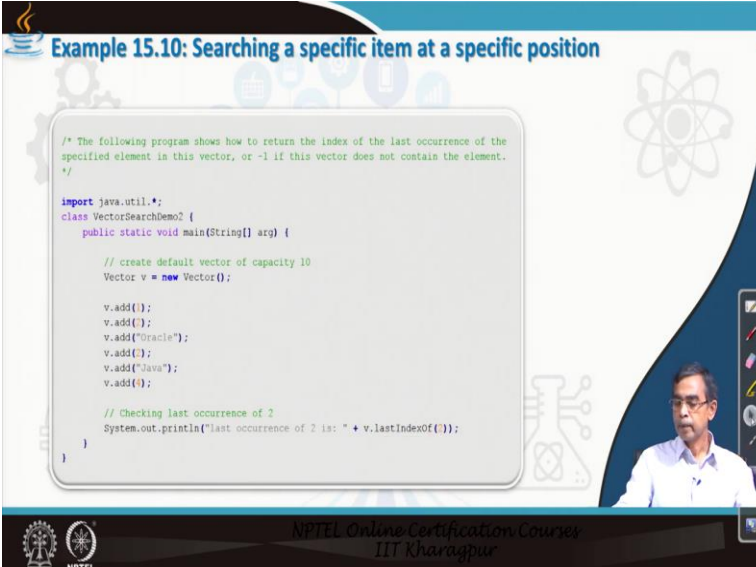
        // check whether vector contains "Java"
        if (v.contains("Java"))
            System.out.println("The element exists");
        else
            System.out.println("The element does not exist");
    }
}
```

NPTEL Online Certification Courses
IIT Kharagpur

Now, I just want to give one more specific example about searching an element. So, contains method we have already discussed about so this example shows how the contains method you can apply in this here. So this is the usual code. So, here this is, these are the usual code for creating a vector and loading it and use v.contains, so this is one, it checks that whether this c is there.

So, it return to it for depending on if the element is present there or not then it will write. So, these are very pretty simple examples. Like this you can exercise all these methods in this class and check that how they work for you.

(Refer Slide Time: 40:26)



Example 15.10: Searching a specific item at a specific position

```
/* The following program shows how to return the index of the last occurrence of the specified element in this vector, or -1 if this vector does not contain the element. */
import java.util.*;
class VectorSearchDemo2 {
    public static void main(String[] arg) {

        // create default vector of capacity 10
        Vector v = new Vector();

        v.add();
        v.add();
        v.add("Oracle");
        v.add();
        v.add("Java");
        v.add();

        // Checking last occurrence of 2
        System.out.println("last occurrence of 2 is: " + v.lastIndexOf(2));
    }
}
```

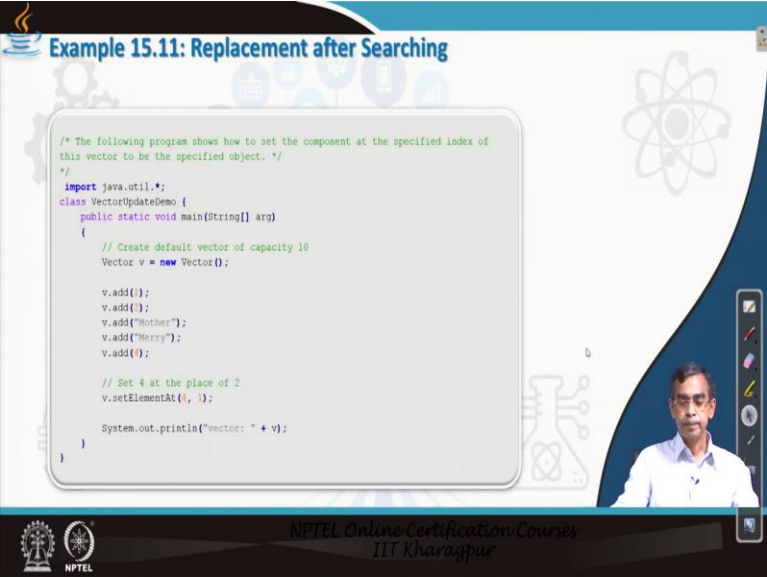
NPTEL Online Certification Courses
IIT Kharagpur

So, here again searching over the entire list is a but searching can be done with a specific elements here. Now, here searching at a last element or index element and like this one, for example here. This methods are very interesting to see; the last index of 2. Now, here in this particular vector that you have created here 2 occurs two times here; for example, this is a first location, second and then again fourth location it is there.

Now, you can call lastIndexOf method; that means it will if you just contains it will basically check the first occurrence of the element. So, if you say v dot contains 2, it will basically find this one. But, if you say last index of 2. So, that means from the bottom which is the last occurrence of the element having a particular value it can basically search and return these things.

So, the this is one use that a specific element location searching at a particular location also can be carried out.

(Refer Slide Time: 41:38)



The slide displays a Java code snippet for a Vector. The code starts with a comment explaining the purpose: to set a component at a specific index. It imports java.util.* and defines a class VectorUpdateDemo with a main method. In the main method, a Vector v is created with a capacity of 10. Elements are added to the vector: 1, 2, "mother", "merry", and 1. Then, the element at index 2 is replaced with 4 using the setElementAt method. Finally, the vector is printed to the console.

```
/* The following program shows how to set the component at the specified index of
this vector to be the specified object. */
import java.util.*;
class VectorUpdateDemo {
    public static void main(String[] arg)
    {
        // Create default vector of capacity 10
        Vector v = new Vector();

        v.add(1);
        v.add(2);
        v.add("mother");
        v.add("merry");
        v.add(1);

        // Set 4 at the place of 2
        v.setElementAt(4, 2);

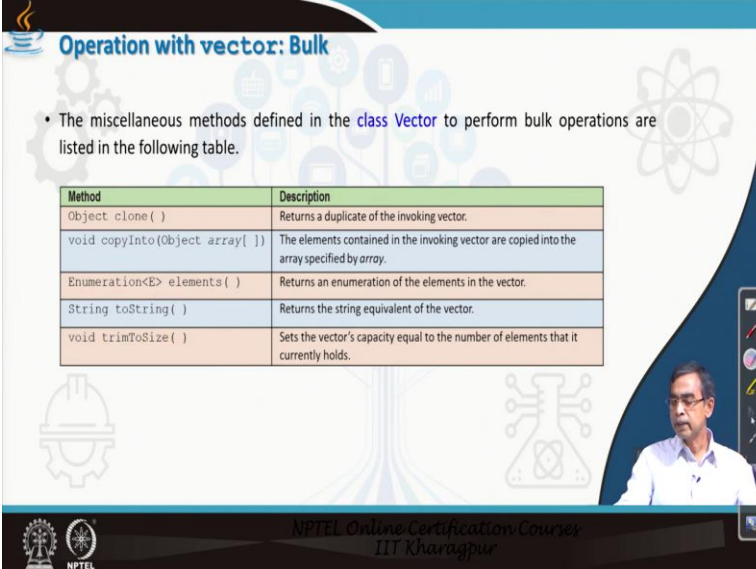
        System.out.println("vector: " + v);
    }
}
```

So, these are searching up and then like searching you can do some setting elements also, it is called the replacement and this example shows the how the replacement of a particular element can be done here. For example, the method, setElementAt can be used. So, it basically says that, index and this is basically 1.

So, it is set 4 at the place for; it is basically index start from 0 to, that is why it is here set at the location 2 actually so it basically, this method replace the element which is presently at the

location or index 1, by 4. And after executing this method if you can print this you can see this element will be replaced by 4. And there are few more bulk operations that you can carry, you can apply on the vector class.

(Refer Slide Time: 42:44)



The slide is titled "Operation with vector: Bulk". It contains a bullet point stating: "The miscellaneous methods defined in the class Vector to perform bulk operations are listed in the following table." Below this is a table with two columns: "Method" and "Description".

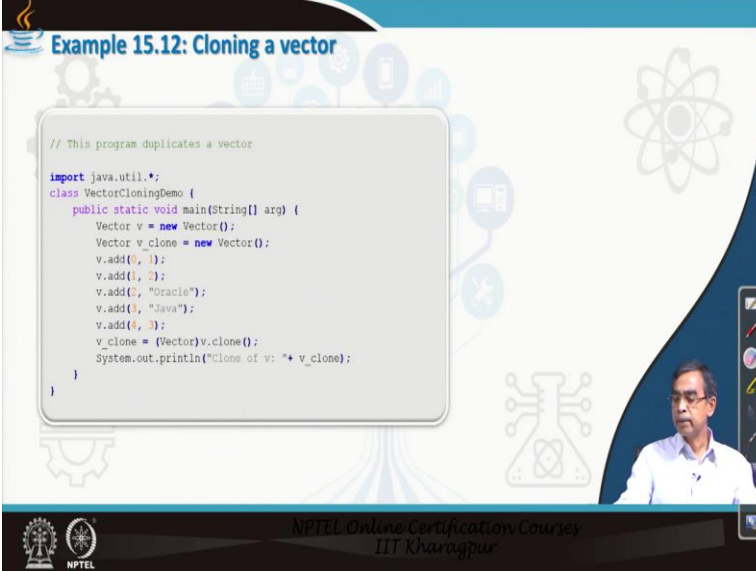
Method	Description
Object clone()	Returns a duplicate of the invoking vector.
void copyInto(Object array[])	The elements contained in the invoking vector are copied into the array specified by array.
Enumeration<E> elements()	Returns an enumeration of the elements in the vector.
String toString()	Returns the string equivalent of the vector.
void trimToSize()	Sets the vector's capacity equal to the number of elements that it currently holds.

The slide also features a small video inset of a man in the bottom right corner and the NPTEL logo at the bottom left.

I have listed the methods here. This is the clone method basically it will make a duplicate of an existing vector, copyInto a collection object array, object of any type of array you can create and then you can copy. This is same as clone. Enumeration element, so, all the elements can be enumerated if you have the familiarity of the enumeration you can understand.

So, if the 7 elements say Monday, Tuesday, Sunday like this one and they can be enumerated type and then all elements can be converted toString. So, it is another method, trimToSize, if you can reduce the vector to a particular size deleting some element also the defined methods is called as bulk operations. Now, let us quickly visit some examples in this category so that you can understand.

(Refer Slide Time: 43:27)



Example 15.12: Cloning a vector

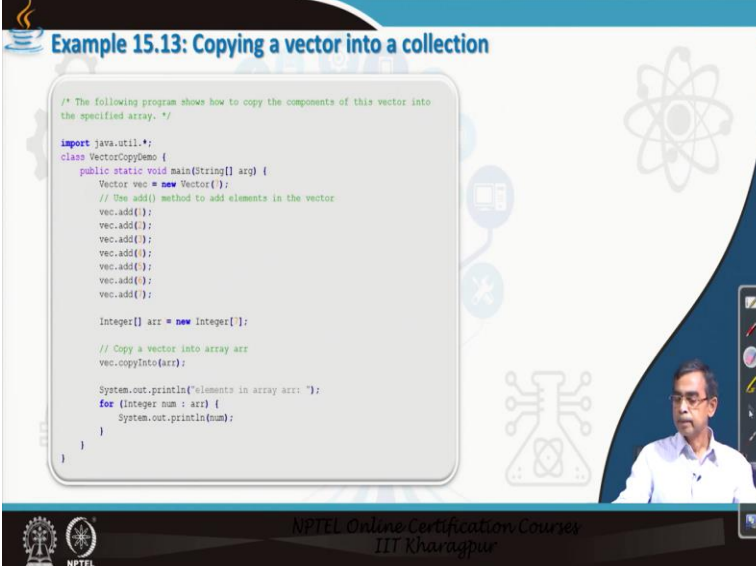
```
// This program duplicates a vector

import java.util.*;
class VectorCloningDemo {
    public static void main(String[] arg) {
        Vector v = new Vector();
        Vector v_clone = new Vector();
        v.add(0, 1);
        v.add(1, 2);
        v.add(2, "Oracle");
        v.add(3, "Java");
        v.add(4, 3);
        v_clone = (Vector)v.clone();
        System.out.println("Clone of v: "+ v_clone);
    }
}
```

NPTEL Online Certification Course
IIT Kharagpur

So, cloning a vector, this operation a vector *v* and another you define another *v clone* and initialize this vector and then call this *v clone*. Then it will basically duplicate of all the elements and copy into this vector. So, this is the clone operation that can be performed to a vector class.

(Refer Slide Time: 43:53)



Example 15.13: Copying a vector into a collection

```
/* The following program shows how to copy the components of this vector into
the specified array. */

import java.util.*;
class VectorCopyDemo {
    public static void main(String[] arg) {
        Vector vec = new Vector();
        // Use add() method to add elements in the vector
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();

        Integer[] arr = new Integer[];

        // Copy a vector into array arr
        vec.copyInto(arr);

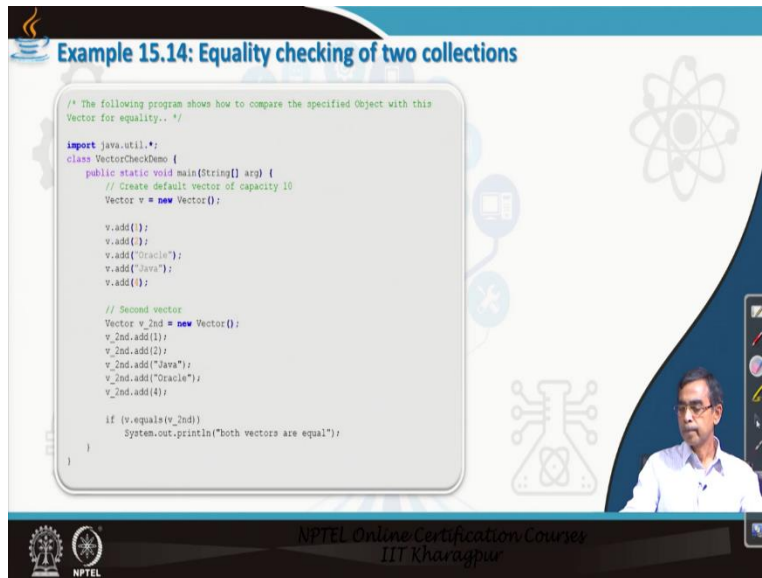
        System.out.println("Elements in array arr: ");
        for (Integer num : arr) {
            System.out.println(num);
        }
    }
}
```

NPTEL Online Certification Course
IIT Kharagpur

And copying a vector into an array collection, collection actually. So, here is an example, you create a vector and then you create an array of elements so it is an array of integers and then copy into array that means these vector will be copied into these array. So, this is another copy, this is

just like clone method and this is a printing of the array or printing of the vector also you can do in the usual way.

(Refer Slide Time: 44:24)



Example 15.14: Equality checking of two collections

```
/* The following program shows how to compare the specified Object with this
Vector for equality.. */
import java.util.*;
class VectorCheckDemo {
    public static void main(String[] arg) {
        // Create default vector of capacity 10
        Vector v = new Vector();

        v.add(1);
        v.add(1);
        v.add("Oracle");
        v.add("Java");
        v.add(1);

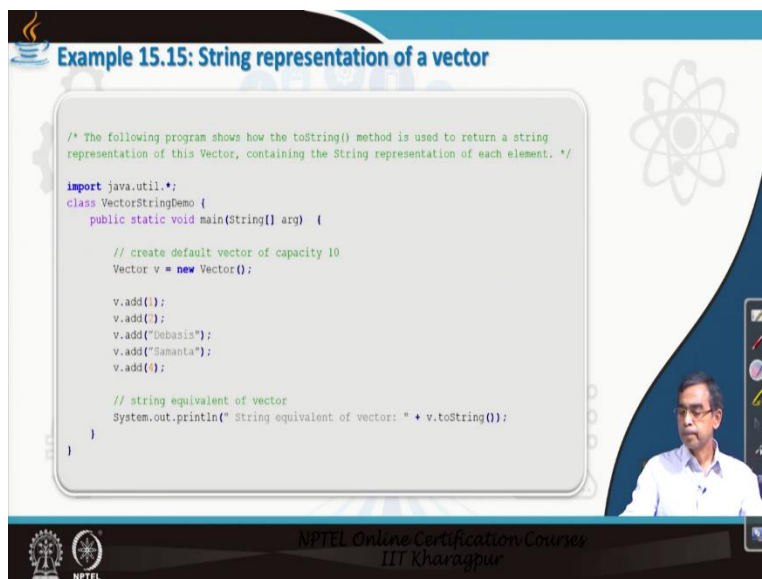
        // Second vector
        Vector v_2nd = new Vector();
        v_2nd.add(1);
        v_2nd.add(2);
        v_2nd.add("Java");
        v_2nd.add("Oracle");
        v_2nd.add(4);

        if (v.equals(v_2nd))
            System.out.println("both vectors are equal");
    }
}
```

NPTEL Online Certification Courses
IIT Kharagpur

And then you can check whether the 2 vectors are equal or not, so there is a method; the method is called Equals method. So, here you create two vectors and in this case as is a two vectors are almost equal, same either they are in order or they contain the same elements so it basically check it and then print. Accordingly if they contain the same set of elements or not.

(Refer Slide Time: 44:55)



Example 15.15: String representation of a vector

```
/* The following program shows how the toString() method is used to return a string
representation of this Vector, containing the String representation of each element. */
import java.util.*;
class VectorStringDemo {
    public static void main(String[] arg) {

        // create default vector of capacity 10
        Vector v = new Vector();

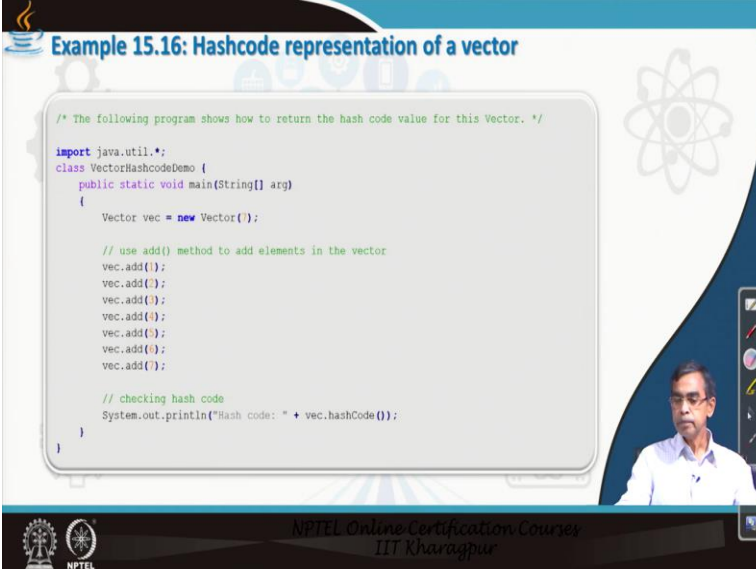
        v.add(1);
        v.add(7);
        v.add("Debasis");
        v.add("Samanta");
        v.add(4);

        // string equivalent of vector
        System.out.println(" String equivalent of vector: " + v.toString());
    }
}
```

NPTEL Online Certification Courses
IIT Kharagpur

Then string represent these toString methods you can use so this is basically creating a vector and then 2 string method that mean all the elements those are there will be converted in the string and then print.

(Refer Slide Time: 45:14)



The slide displays a Java code snippet for calculating the hashcode of a Vector. The code is as follows:

```
/* The following program shows how to return the hash code value for this Vector. */
import java.util.*;
class VectorHashcodeDemo {
    public static void main(String[] arg)
    {
        Vector vec = new Vector();

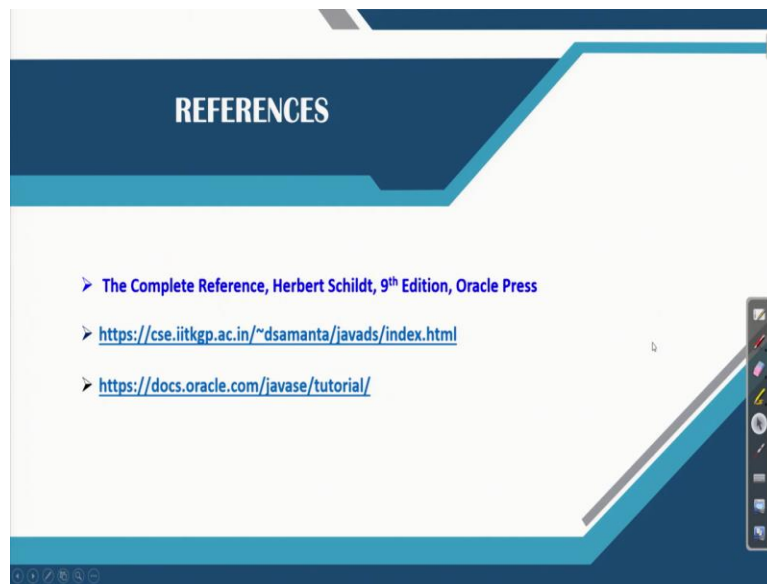
        // use add() method to add elements in the vector
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();
        vec.add();

        // checking hash code
        System.out.println("Hash code: " + vec.hashCode());
    }
}
```

The slide also features a small video inset of a presenter in the bottom right corner and the NPTEL logo in the bottom left corner. The footer text reads "NPTEL Online Certification Courses IIT Kharagpur".

And there is last method Hashcode. This basically you create a vector; now, regarding hashing we will discuss later on in details. So, hashcode is basically is a encryption form, so, all the elements will be encrypted and then encrypted version which basically is very difficult to, difficult to understand for which this is the number like this one. So, the hashcode we will discuss in details later on but here for a vector class you can call the hashcode method to encrypt all the elements. So, this is a method that called as a vec, bulk operation.

(Refer Slide Time: 45:55)



Now we have discuss about many methods those are very common to use in vector. I hope that you have little bit studied about the vector class, if you want to study more about it, there are link I have given. This is a 1 link all the program that we have used in this, in addition few more programs also you can find it there.

And for details about the vector class you can go to the link; this is the official websites of the oracle set so tutorial and go to the vector class you can find details about each methods that is there in the vector class with some illustrations. Very good materials and moreover this book is good so you can follow this book for studying. Thank you!