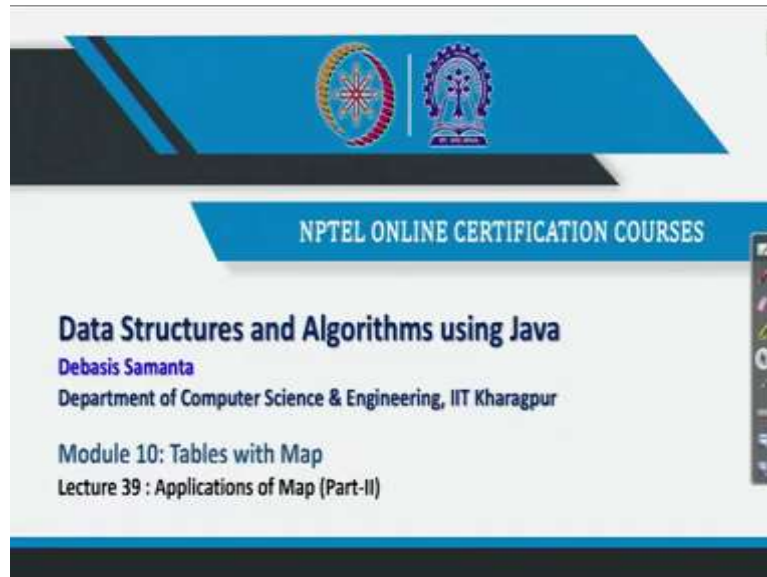


Data Structure and Algorithms using Java
Professor Debasis Samanta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture 39
Applications of Map (Part-2)

(Refer Slide Time: 00:32)



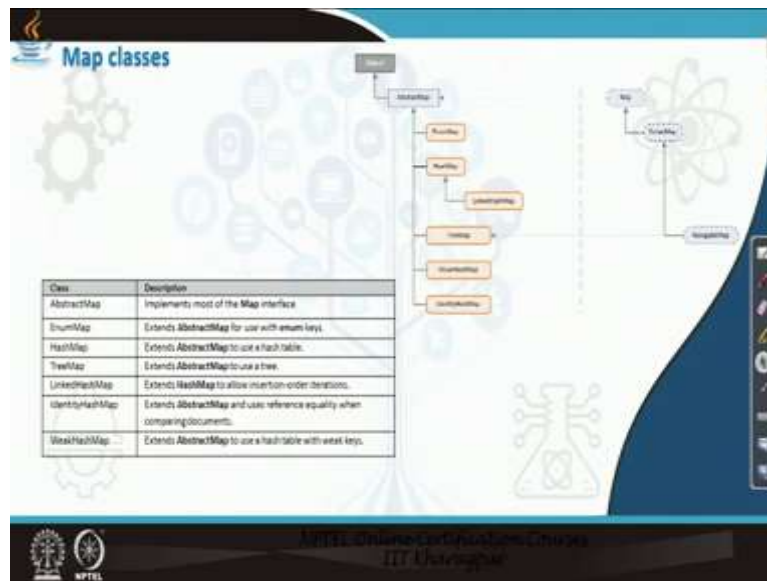
We are discussing about tables and so far we have discussed the map framework and linked tables and one important class we have discussed about hash map.

(Refer Slide Time: 00:38)



There are few more what is called the utilities are there in the Java dot util basically to dealing with tables they are basically tree map, linked hash map, and Enum map, so we will discuss all those things.

(Refer Slide Time: 00:50)



The slide titled "Map classes" shows a class hierarchy diagram. At the top is the `Map` interface. Below it are `AbstractMap`, `EnumMap`, `HashMap`, `TreeMap`, `LinkedHashMap`, `IdentityHashMap`, and `WeakHashMap`. `AbstractMap` is the superclass for `EnumMap`, `HashMap`, `TreeMap`, `LinkedHashMap`, `IdentityHashMap`, and `WeakHashMap`. `HashMap` is the superclass for `IdentityHashMap` and `WeakHashMap`. `TreeMap` is the superclass for `LinkedHashMap`. A table below the diagram provides descriptions for each class.

Class	Description
<code>AbstractMap</code>	Implements most of the <code>Map</code> interface.
<code>EnumMap</code>	Extends <code>AbstractMap</code> for use with enum keys.
<code>HashMap</code>	Extends <code>AbstractMap</code> to use a hash table.
<code>TreeMap</code>	Extends <code>AbstractMap</code> to use a tree.
<code>LinkedHashMap</code>	Extends <code>HashMap</code> to allow insertion-order iterations.
<code>IdentityHashMap</code>	Extends <code>AbstractMap</code> and uses reference equality when comparing documents.
<code>WeakHashMap</code>	Extends <code>AbstractMap</code> to use a hash table with weak keys.

They are basically the similar to hash map actually but better if we can discuss those things and illustrate the different utilities, the method those are there in these methods.

(Refer Slide Time: 01:01)



The slide titled "TreeMap Class" features a central image of a coffee cup with steam rising from it. The text "TreeMap Class" is prominently displayed in the center. The background includes decorative icons of gears, a tree, and a molecular structure. A small video inset in the bottom right corner shows a man speaking. The NPTEL logo and course information are visible at the bottom.

Creating a Map container as a TreeMap class object

TreeMap class defines the following constructors:

Constructor	Description
<code>TreeMap()</code>	The first form constructs an empty tree map that will be sorted by using the natural order of its keys.
<code>TreeMap(Comparator c)</code>	The second form constructs an empty tree-based map that will be sorted by using the Comparator comp. (Comparators are discussed later in this chapter.)
<code>TreeMap(Map m)</code>	The third form initializes a tree map with the entries from m, which will be sorted by using the natural order of the keys.
<code>TreeMap(SortedMap sm)</code>	The fourth form initializes a tree map with the entries from sm, which will be sorted in the same order as sm.

Note:
 TreeMap has no map method beyond those specified by the NavigableMap interface and the AbstractMap class.

Now, let us first discuss about the tree map class, now tree map class first how we can create a tree map and what is basically the tree map and how it is different from the hash map? So, from a programmer point of view absolutely you do not find any difference between the two; hash map and tree map they are basically same all methods, all programs that we have exercised for the hash map can also re-do for the tree map also.

But internally how the objects are stored in the form of a table or what internal data structure is used that is the totally different and basically it remain hidden and programmers should not bother about it. So, what we can say that hash map used may be array list type of data structure as a collection and here tree map used t-type structure as a collection. So, it basically storing mechanism which basically Java Virtual Machine follows in order to facilitate the table to the programmer anyway so tree map objects can be created by many constructors I have listed few important constructors see here.

So, the first constructor that tree map it is a default constructor, so it will create a default value initially it is empty we can say and then tree map can be created with giving an another comparator class because tree is record to create a comparison, so comparator method for string and upper method comparator is define but some other user type the comparator method to be defined we have already learn about how a comparator method can be define for this one.

So, it is basically that constructor can be called to create a tree map where the comparator can be define like and here is basically you can create a tree map giving already existing map to it and that map can be any type whether it is a linked map or it is a hash map no issue it can be

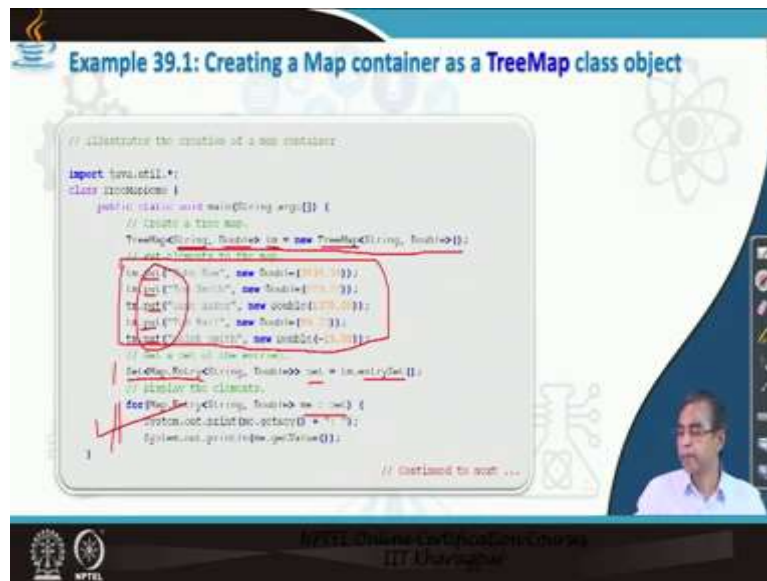
not necessary to be pass any type, so that is why the generic it is defined there and here these are another example if we pass the sorted map then also a tree map can be created.

So, any map hash map or linked hash map here the tree map can be created with using sorted map already available then all this things so, having one map can be created another map this map is content in other map this actually concept is there and so, this tree map method does not have its own method define it or all the methods which are defined in the interface map and then sorted map are implemented in this class also.

And all the methods those are there in tree map is also (extend) abstract map so, this way all methods those are declared in map interface are implemented in this class.

(Refer Slide Time: 03:59)





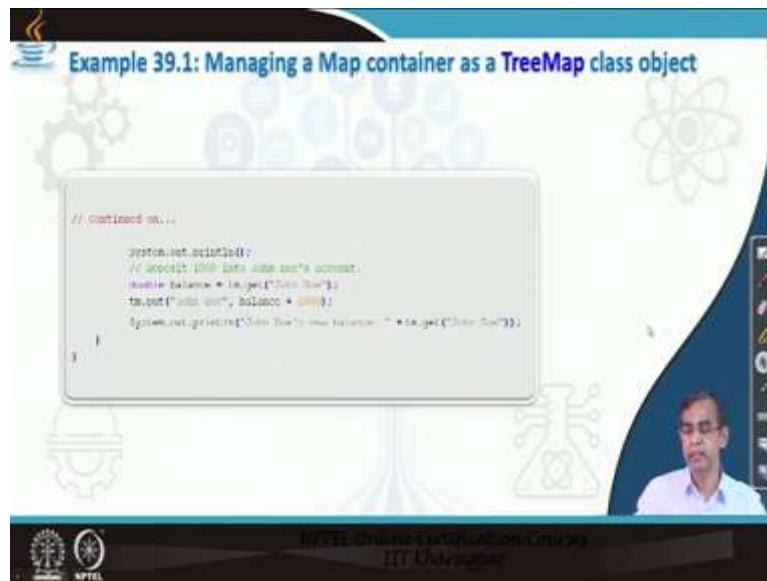
Now, let us see what are the different way this tree map objects can be created and how they can be utilized in the program so, we will just simply give a simple illustrative program. Here let us look at the first program in this study this program basically gives an idea about how a tree map as a container that is a table can be created. So here we create a tree map the name of the container is tm and here is the key is of type string and object is of type double.

So, basically this is the key object here and we just you see it is the same fashion as the earlier one same only the put method those are defined we have used for the hash map, we can use the same put method is for the tree map but their implementation is different whereas the hash map the implementation is different those are internal you do not have to bother.

So, here just we use in live or hash map we using tree map that is all, other things are very similar, common. So no issue there, now here actually set we have already used that means it basically create a collection of set using empty set that means all these elements will be stored as a set collection, set will be discuss in details in the next lectures actually.

Now here is basically traversing so, it basically using for each loop we create this one, one important thing is that for is loop cannot directly apply to this table because they are basically for mapping only, mapping cannot be implicated to this one. So, that is why we have to store this as a set and on the set we have to traverse all elements in it so, this is an example so that how a tree map container can be created and it can be viewed.

(Refer Slide Time: 6:08)

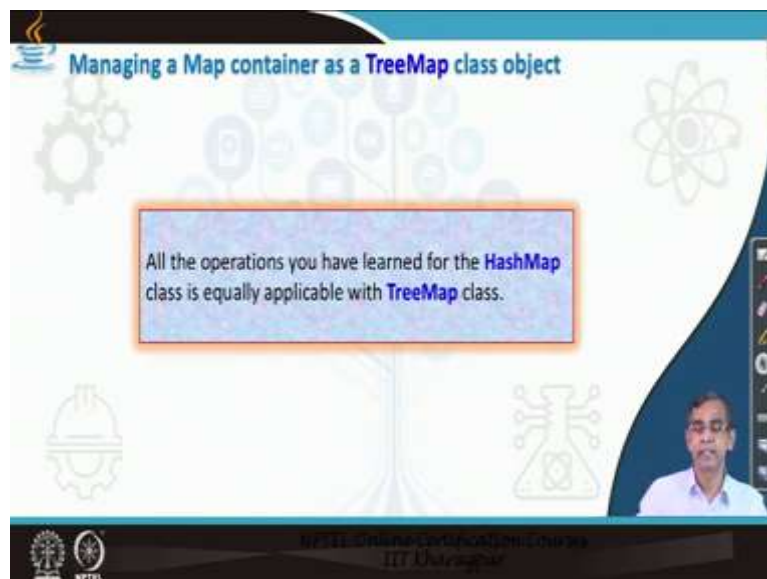


The slide displays the following Java code snippet:

```
// continued ex. 1.1  
  
System.out.println();  
// deposit 1000 into John Doe's account.  
treemap.put("John Doe", balance + 1000);  
System.out.println("John Doe's new balance: " + treemap.get("John Doe"));
```

Now, let us have an another example so this is basically same thing how the get method call in this one so same methods are there.

(Refer Slide Time: 06:14)

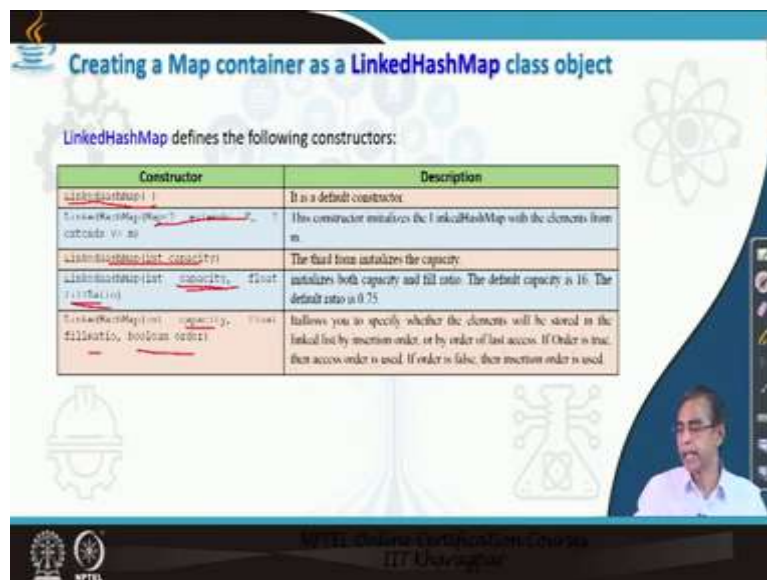


The slide contains the following text in a blue-bordered box:

All the operations you have learned for the **HashMap** class is equally applicable with **TreeMap** class.

Now, what I want to repeat it again all the operations which we have already studied for the hash map is equally applicable to tree map, all the programs which we have demonstrated in last lectures you can again try with just simply replacing hash map as tree map this means that all the methods are applicable to the tree map only. But, internally it different that is all internal means which is faster all those things other than these things whatever it is there.

(Refer Slide Time: 06:49)



So, now let us consider a example of linked hash map again it is same as hash map but storage internal storage mechanism is different. Now, again let us see what are the different constructors are there for the linked hash map class by which this kind of container can be created.

So, there is a default one constructor which basically create a default container which initial size is empty. Now it can be created having another map that is passed an argument to this constructor so that means it is basically created a new container using already existing container in it a new container can be created giving as a size capacity initial capacity this one.

So, this is another capacity can be any integer value now it is also capacity as well as fill ratio because this linked map like hash map also dynamically grow automatically as it exhausted its capacity then automatically it will allocate the memory but what is the size of incrementation of the next what is called the enhancement that can be decided by fill ratio so 0.75 you mention this means that the 75 percent that you can mention, 0.5 means 50 percent, 0.3 means 30 percent, like whatever it is there.

So the default capacity is 16 and otherwise it will be increase each time 50 percent of the existing size of the it is there. Now, linked hash map it is also can be created by mentioning capacity, the fill ratio and order so if you want to store the data in an order either order is possible in case of key values whether ascending or descending order of the keys you can mention this one.

So order is possible for only for some data types not for all if it is so for numeric type actually it is applicable. So if it is true then it basically that container will be maintained in an order if it is false it is not in order it is basically first come first service in the order of insertion adding it is there. Now, let us consider example illustrating the concept of this method it is there.

(Refer Slide Time: 09:19)

```
// This program illustrates the creation of a Map object using LinkedHashMap class.
import java.util.*;

class MyHashMap {
    private static final int MAX = 6; // Set as the max size of the map.

    public static void main(String arg[]) {
        // Creating the linked hash map and implementing insertMethod() to add size
        LinkedHashMap<Integer, String> m =
        new LinkedHashMap<Integer, String>();
        // Inserting values until it starts throwing exception, string<Object> {
        return m.size() > MAX;
    }
}

// Continued to next ...
```

I have given a simple example to understand it. And this is the example just you check it, this example basically tell how using this class linked hash map a table can be maintained so I define one variable it is called the max is a size it is basically size of maximum size of the table in that case 6 you can mention 60 or 16 whatever is there.

So, initially I define it this is the maximum size of the table will be 6. Now we create here you can see this the one way we can creating the linked hash map you are created it is I mean pair if the keys type integer object is type of string name of the container is lhm means called the type of linked hash map and here we just call this constructor here `removeEldestEntry` I will discuss about this method later on.

Actually `EldestEntry` means it will basically decided whether the eldest entry that is there will be removed to accommodate the next entry or not depending on this method if we override it is there. So, here basically if size exceeds this matrix then this method will be called I mean this `removeEldestEntry` method that means the first entry that is the oldest entry will be removed so, that it can accommodate the new entries.

So, it is sometimes useful because all old objects are not, some required so we can remove automatically so, using this method we can automatically remove the oldest entries by the new entries like. So it is basically just like linked means it is basically whenever we add new elements it add at the end and at the same time it basically remove the front element like so, this concept it is basically `removeEldestEntry` method that can be.

So, here basically we are creating this linked list with this criteria decided if we do not use it then it automatically default that it will not there if size is limited then it will stop it there, so this concept it is there I have given an idea about if we can have this kind of features into it then we can redefine or override this method if it is there where we have overridden that it is here.

(Refer Slide Time: 11:53)

Example 39.2: Managing a Map container as a LinkedHashMap class object

```
// CONTINUED IN ...
// Adding more elements
LinkedHashMap<String, Integer> map = new LinkedHashMap<>(4);
map.put("The", 1);
map.put("Map", 2);
map.put("is", 3);
map.put("a", 4);

System.out.println("Map:");

// Adding more elements
map.put("is", 5);

// In printing the map after adding one more element
System.out.println("Map:");

// Adding more elements
map.put("is", 6);

// Displaying the map after adding one more element
System.out.println("Map:");
```

Now, let us have some continuing this program again so that you can understand it better here so, this is in continuation of the previous part of the program we just add few elements we have mention the size so we already reach the maximum size. Now, here if we put the next element then it will basically will see the first element will be removed this element will be removed because this is the eldest element and it will add it on.

And then if you print you can see that it will print up to this elements it is there. Then again 7 it will remove this one then next element it will added and so on. So this basically tell that how automatically popping will takes place when you push some elements into it and this is the new characteristic in the linked hash map it is not available in other class like hash map or tree map like.

(Refer Slide Time: 12:53)

The top slide, titled "User Defined Data Type in Maps", features a central illustration of a coffee cup on a saucer. The background is light blue with various icons like gears, a tree, and a molecule. A small video inset of a speaker is in the bottom right corner.

The bottom slide, titled "Example 39.3: Map container with user defined data type", displays a code editor window with the following Java code:

```
// This example illustrates how a map container will be with objects of book class.
import java.util.*;

class book {
    int id;
    String name, author, publisher;
    int quantity;

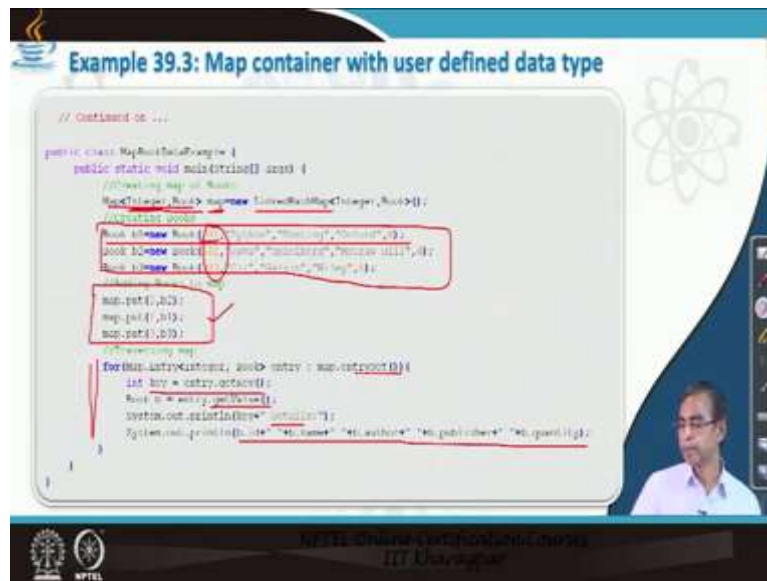
    public book(int id, String n, String a, String p, int q) {
        this.id = id;
        name = n;
        author = a;
        publisher = p;
        quantity = q;
    }
}
```

The code editor window has a red border and a scrollbar on the right. A small video inset of the speaker is also present in the bottom right corner of the slide.

So this is about the eldest method that we have defined. Now, let us consider we have discussed about all the simple objects that is the string and then maybe integer, double this one now, we want to see whether user defined objects can be stored in tables which is more realistic there. So, next example will give an illustration where you can define your own objects and those objects can be stored in a table.

Let us first define the objects that we want to store it. So here we define one object called the class book so, it has few fields like these are the fields and this is the constructor to initialize the objects. So, this is the declaration of an objects now we want to store a few objects of class book into a table and then we can use table either hash map, or tree map, or any type.

(Refer Slide Time: 13:55)



I will give you an example using a particular type it is called the linked hash map only in this case but, you can repeat the same example using hash map or tree map or sorted map also. Now, let us see how we can create few objects and add it into here it is very simple first we create here you note we create integer this is the key and book this is basically the object which we want to store.

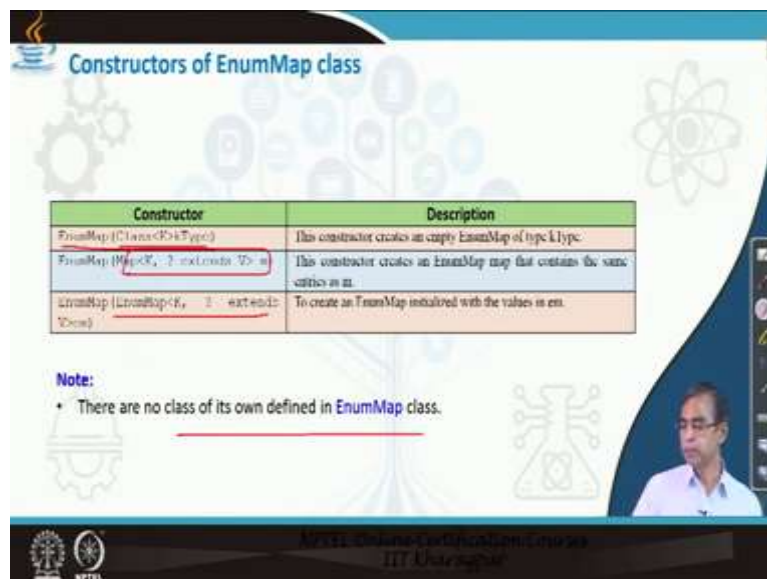
So, this is the key object pair and this is the type of container is the linked hash map pair and this is the name of the container it is map in this case. Now, in this case we create 3 objects book; b1, b2, b3 and all those 3 objects are added into the table in the next few statement so, the objects are added to this map now, here traversing is the map using for loop here you see we just using the map empty set.

So that the set is created here and for each entry into this one we just get key and get value and then we just print the results here using this one. So, for each objects that will be there we can be created there so this basically gives an idea about how user defined objects also can be stored and this is a one simple example that we have given how the different objects can accessed or it can be traversed.

There are many ways it can be retrieved also using get value get key is the method that you can consider or get method also you can create you can pass the get for a particular integer here integer is basically this one that means every objects are mapped with this are the key value it is there. So, if you want to particular get so get for this object map dot get 101 it basically get access this book object to you. And these are the idea about if you want to

access a particular object like this one. So this is example is basically tell you how your table can store your defined object in it.

(Refer Slide Time: 16:05)

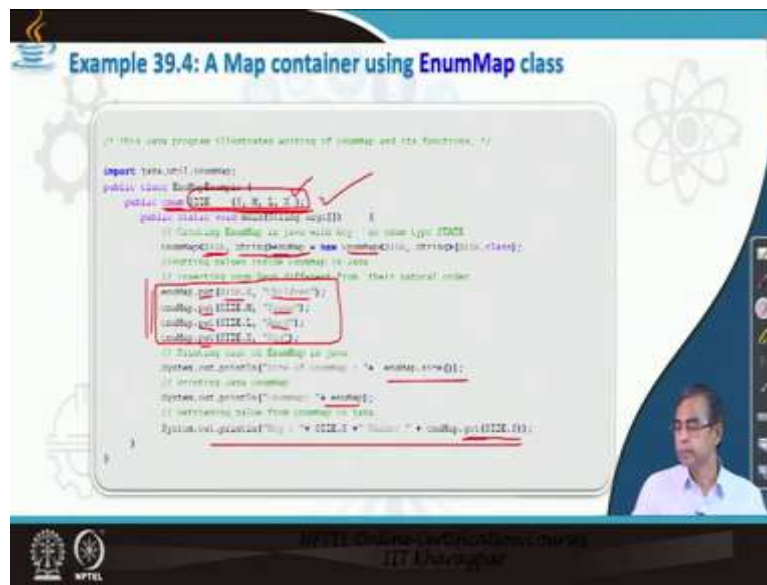


Now, this is last one topic that we want to discuss about it is called the EnumMap class, EnumMap class it is of type to store the enumerated type you know the enumerated type is basically some elements that is of enumerated. Now, there are 3 constructors you can create in order to store the enumerated type of objects into table and this is our first constructor that you can consider where this class basically represent the enumerated type.

And here is the another map existing map can be added into the newly created map if you want to that you can do it and this, okay so this is the existing map. And another also existing map but it basically with upper bound that can be limited to this one, so these are the 3

different constructors can be used to create table of type enumerated and it does not have any its own method, all methods which are there in abstract map it is also available there in return basically all methods which are define in the map interface is basically are implemented in this enumerated class. Now, let us have some examples so that we can understand this concept better.

(Refer Slide Time: 17:32)



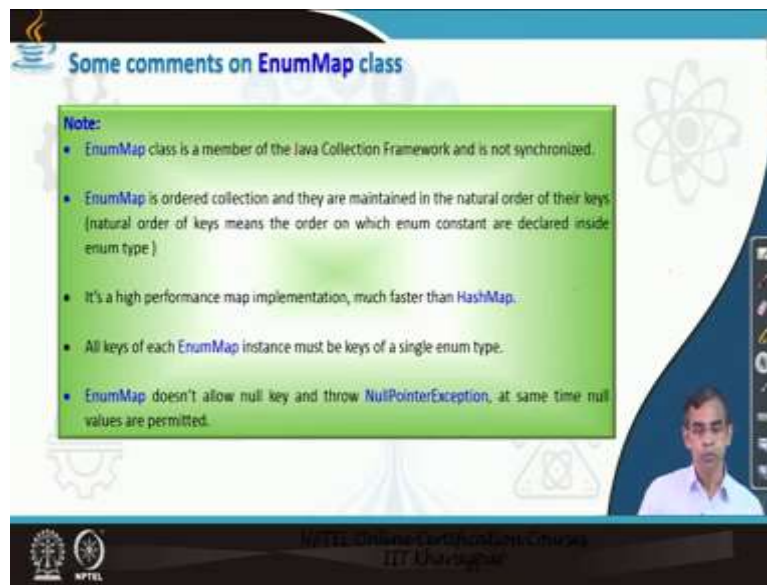
So, here is an example it basically gives you how a enumerated class type of container can be created. So now here we define Enum class this basically here you see public Enum this is the keyword that is there to define any enumerated type and then size is basically the enumerated set actually s, m, l, x this is the enumerated elements which belongs to this enumerated class.

Now, here we see we create a container of type enumerated Enum map so name of this container is enum map of type this one and here we see the first the set of key values will be limited to this enumerated sets, so this is the size, this is the set of keys that means set of keys will varies from s, m, l and x within this sets. If it integer it can varies from any values but here we limit to this enumerated this one that is the different. And string is basically is the objects type of objects.

Now, here we add few entries into this container our newly defined enumerated container and you see the put method which is already used for other classes in this category we have used it then we just created. Now you see we pass the value size dot s that means this class size dot s this is there and then this is basically the string objects that is there. So, this basically created a table with a enumerated value as the key.

And then we can just simply size and then we can have the print of all the tables if you want to have you can print and then particular key value if you want to print using get method it basically get size s that means this value will be printed, children will be printed. So, these are the basically example by which enumerated method can be used to store a enumerated type into the table.

(Refer Slide Time: 19:53)



Some comments on EnumMap class

Note:

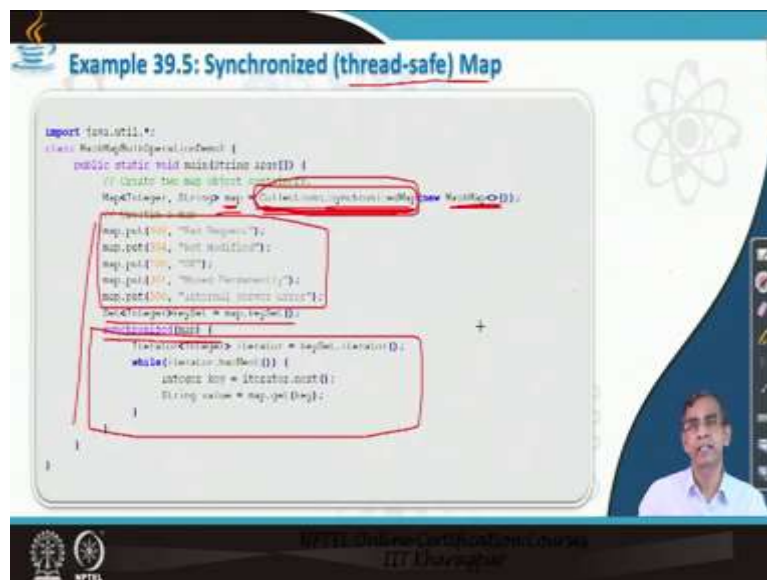
- EnumMap class is a member of the Java Collection Framework and is not synchronized.
- EnumMap is ordered collection and they are maintained in the natural order of their keys. (natural order of keys means the order on which enum constant are declared inside enum type.)
- It's a high performance map implementation, much faster than HashMap.
- All keys of each EnumMap instance must be keys of a single enum type.
- EnumMap doesn't allow null key and throw NullPointerException, at same time null values are permitted.

So, here few points that you should note EnumMap class is a member of the Java collection framework and it is not synchronized, it is not synchronized in the sense that if you write the multi-threading program using enum map then your program will give the compile time error because it is not advisable to have the synchronized method for this type of objects.

Enum map, is ordered collection and they are maintained in the natural order, natural order of their keys for example Enum map different s, l, m, x in that order it will basically store there and this is basically very highly performed one data structures for the table maintenance compare to hash map or other structure actually. All keys of Enum map instance must be keys of single enum type that is one important concept that is there all key values should be of Enum type only.

A Enum map does not allow null key to be added into the sets if you add null key then it throw an exception called the NullPointerException. So, you cannot create an object or entry using NullPointer it is there. Even null key values cannot allowed in other method also in that case they will also pass a NullPointerException error there that means key values should be non-null value actually, it is true for all classes it is there also including hash map it is there.

(Refer Slide Time: 21:29)



Now, synchronization is an issue and whenever we used table, particularly database or whatever is there we cannot live without synchronization because Multithreading is anyway it has to be there. Now, in case of Java map framework the recently they have introduced a new synchronized rather it is called the Thread Safe map framework it is there.

Now, I will just give an example about how the synchronized map set can be created here with few examples here and to do this things you see what exactly the new things that you have to do it for that purpose there is a collection class you have to consider here in this collection class there is a method it is called the synchronized map method.

So, if you want to create a map, a container of any type here for example you are creating a hash map but it is to be used for the synchronized now none of the hash map, tree map all

they are basically synchronized or they are not thread safe so, here is attempt that how we can make it thread safe implementation using that map class for the table.

So what exactly you have to do is that you have to call this method while we are creating a new objects so this is very important. So, if we call this then they that means the map which will create of any type become a thread safe map, rest of the things is very same.

And then you can use as a synchronized keyword you can already know about in a multi-threading programming synchronized means if we can create more than one thread and then you can create it will allow to enter into this object here object is map only one what is called the thread.

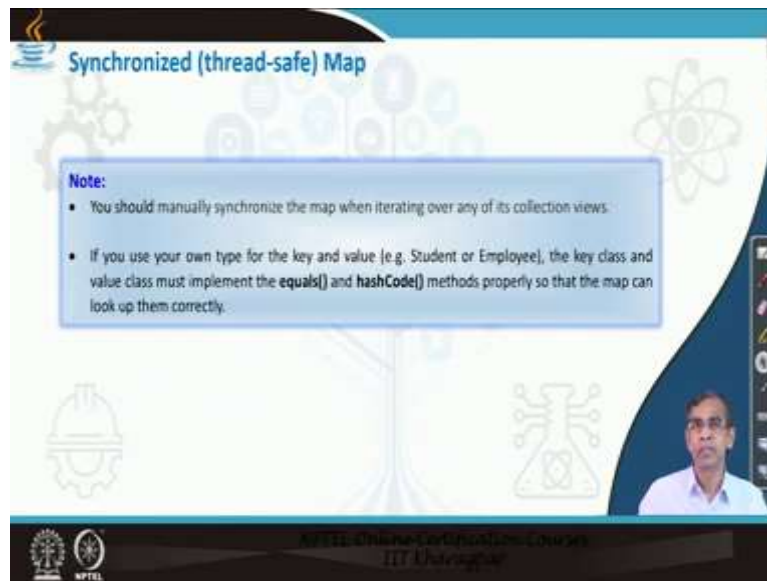
So, this way it basically ensure the mutual exclusion of critical issue of implementation now here in this example we create a container first this container is now according to the definition is a thread safe container then we create a key sets and then we just simply perform some operations which we want to make either remove or replace or whatever the operation you can move it under this synchronized keyword you can do it.

And then better it can be used it if you call 2 or 3 threads and calling the different methods it is there and you can call this then you can see it is there but, if you use a threads without this synchronized then it will give compile error actually. So, that basically I advise you to have the multithreading program again.

Here we have given this one so you can just write a main method there and few thread you can define and for each thread you can define run method there and in each run method all this code can be embedded and you can call, you can see the multithreading program. So, you have to just exercised repeat or recall the multithreading program how you can do it and then in multithreading program you just simply use collection as a object.

Now, likewise you know all Java collection framework, array list there they are also not thread safe, so that is why you can use this way to make them thread safe. On the other hand vector the legacy class is basically synchronized they have thread so you do not have to do it for them actually which we have discussed details while we are discussing about this things in there. Now so, this is the example that is an idea about how the concurrent programming can be done and you can check it.

(Refer Slide Time: 25:07)



Synchronized (thread-safe) Map

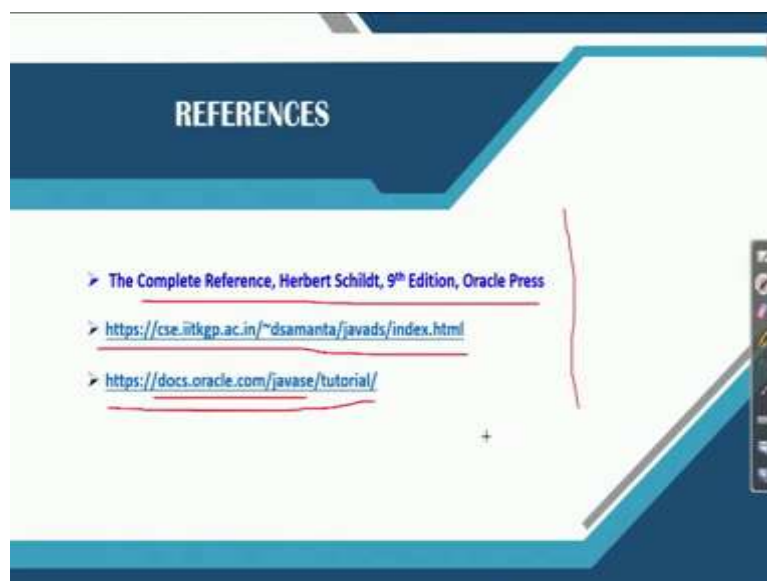
Note:

- You should manually synchronize the map when iterating over any of its collection views.
- If you use your own type for the key and value (e.g. Student or Employee), the key class and value class must implement the `equals()` and `hashCode()` methods properly so that the map can look up them correctly.

NPTEL Online Course on Java
IIT Madras

And there are few more methods equals and hashCode usual things if the two objects are same or not that you can check by calling the equal method for the same container it is there.

(Refer Slide Time: 25:19)



REFERENCES

- The Complete Reference, Herbert Schildt, 9th Edition, Oracle Press
- <https://cse.iitkgp.ac.in/~dsamanta/javads/index.html>
- <https://docs.oracle.com/javase/tutorial/>

+

HashCode we have discussed about you can override the hashCode by for given user defined object that you can think about it and you can practice it. So, there are many more programs that you can have it, few more programs you can find in this link I have given few extra programs also so that you can check those programs and the good material if you want to learn many more other things the exhaustive discussion that you can obtained from the Java Oracle I mean official websites this link it is there.

And this book also I should suggest you to I mean go through so that you can study much more, all the programs that we have discussed about intermittently you can use for any type of class and you can create it and I have given as an illustration only for few 3-4 objects instead you can add few more objects into them.

Vary last set of objects and even you can create a set of objects as array list or some vector also to store the table and into this one and how it can done it represent the exercise for you also that mean a set of objects are available to you as a collection array list for example. And we want to make them as table how you can do this.

So that idea you can find an idea you can definitely you can get an idea and then you can write the program what I want to say that array list to an hash map container how you can do or array list to a tree map or any other linked list structure to the linked list hash map also all those things is possible only the mapping is there.

What I want to say is that again all array list and everything searching and everything (is a trivial) it is a non-trivial issue because, they are not efficient but using this hash map it will basically store the collection but it is the first test, first test in the sense that order of $O(1)$ constant time it can be retrieve the elements and that is the big thing. And it is independent of size of object table that is the most utility which it is basically useful.

Okay with these things I would like to stop this discussion for today and if you have any doubts or any other things you can just put your question in the collection framework. So, this module is concluded here and will discuss about the next module including the another data structure called the set, set is a very common data structures we from class 10 onwards we are studying set but, how the set can be stored as data structure in this example also we have used set also but, will see about the set structure in details in the next module, the module 11 where we will cover set and programming with sets. Thank you very much.