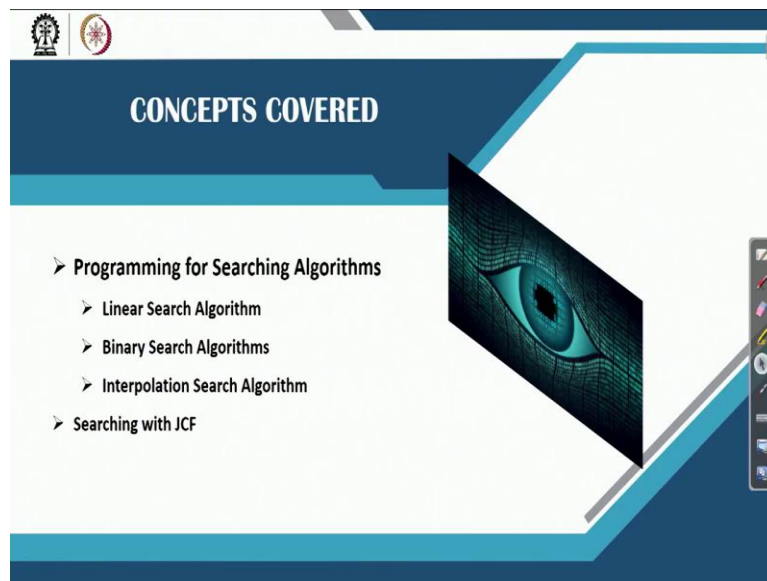


Data Structures and Algorithms using JAVA
Professor Debasis Samanta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture 49
Programs for Searching

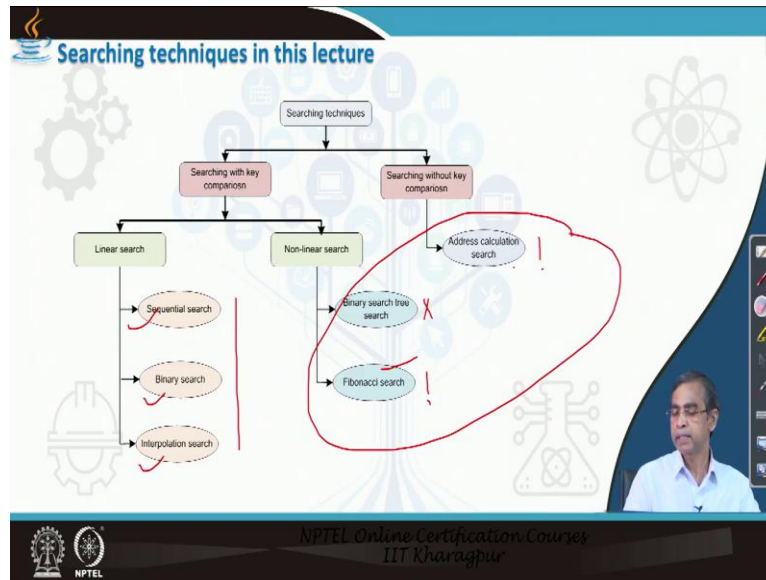
Here we have discussed about many searching techniques; linear searching, non-linear searching and address calculation search. Now it is time to write programs. I will give you how to do the programming based on some searching algorithm. While I was discussing some different searching techniques, there also I have discussed the algorithms also.

(Refer Slide Time: 0:44)



But in this lecture, we will be able to learn how to write the program for linear search algorithm, binary search algorithm, interpolation search algorithm. And then finally I will conclude this lecture with giving the searching with the Java collection framework facilities. Because this is the readymade solution is available to you, you can do it. But if you want to have your own customized solution, then you can write the program. Now how you have to write the program that idea I will give you in this discussion.

(Refer Slide Time: 1:16)



So let us first the different searching technique that we have already learned about, the linear searching where we have discussed sequential search, then we have discussed binary search and interpolation search, then non-linear searching category where we have discussed binary search tree search and then the Fibonacci search. And then address calculation search also we have discussed.

But writing program for all, it is not possible, I will try to give the idea about all this technique I will give on that how to write the code for this and binary search and interpolation search. Binary search tree search, already we have discussed while I was discussing the tree, so I can skip it right now. Fibonacci search I have, okay it is not discussed, but it is left as an exercise, you can try of your own. Address calculation search also you can try, because it is the same concept we have discussed while we were discussing mapping. So it is not discussed there.

So these are the things that I will try to send, give you the code, so that it can give enough idea about how the searching algorithm can be given and for these two, about for this you can write the code of your own for your practice. So let us have the discussion first how we can write the program for searching, first start with linear search or you can say sequential search, the simple search. And again writing the program, we will consider the generic programming that means we will write the algorithm or programs, so that it can take care any type of data.

(Refer Slide Time: 2:44)

Linear Search Algorithm

NPTEL Online Certification Courses
IIT Kharagpur

Linear search with array: Data are stored in random order

Start
 $i = 1$
Decision: $K = A[i]$
Yes: Print "Successful"
No: $i = i + 1$
Decision: $i > n$
Yes: Print "Unsuccessful"
No: Loop back to $K = A[i]$
Stop

5	8	3	2	7	6	0	4	1
---	---	---	---	---	---	---	---	---

NPTEL Online Certification Courses
IIT Kharagpur

Example 49.1: Programming for linear search algorithm

```
/* This program search an array of elements. The program works well whether
the array elements are in sorted order or not. */

class LinearSearch<T extends Comparable<T>> {
    public int search(T[] arr, T x, int len) {
        for(int i = 0; i < len; i++) {
            if(arr[i] == x)
                return i;
        }
        return -1;
    }
}

// Continued to next...
```

NPTEL Online Certification Courses
IIT Kharagpur

So linear search technique, this is an algorithm that we have given it and here is the code that you can follow, very simple code it is there. First follow this one and here actually you see this is our class that we are going to define and this program is for generic that means this method that we are going to discuss, let the search method actually, we can call this method for any type of data.

So you can store an integer array or a reporting point array or character array or any type you can do it. But here T should be type comparable that means those are the types basically eligible for comparing, they are applicable to this one. Now comparable means all integers or maths numbers rather, all numbers like say int, long, floats, double, short they are basically under this category, they are comparable category.

And string is also a comparable category but for other data type, user defined data type so that comparable data is they are not comparable, so we have to write, we have to implement this comparable to for them also. We have used this concept in earlier discussion also, same concept. So now we are considering for simplicity, let us assume that all, this method can be invocable to the number strings like this one, so you can array or strings also you can call, numbers also, any numbers you can use it.

Now here is a technique, very simple for, so you can start from the first location till the last location, is basically length and you have to go it and then check that whether element matches or not. If it matches then return i, if it does not match then it basically this one. So it is here but

it is basically simple search and it is, it does not consider that whether the elements, that we have stored whether in order or not.

So this will work in any cases whether order or not but unnecessary it will loop the role till time, if it matches it will come out. But for the sorted order, some modification needs to be done which is not here. Now you can understand that what modification that you can follow, so that you can do it. So this left as an exercise, if the element is sorted order, what modification that you can do, only this portion you can do.

If this equals to x and x is greater than ai then we can stop it there, so whatever it is there. So that check you can follow it here and little bit rewriting the code you can follow, so that code can work for sorted elements also. So here also assume that element is stored in an array, so this is the array, generic array of type T. So this program you can run.

(Refer Slide Time: 5:51)

Example 49.1: Programming for linear search algorithm

```
class LinearSearchDemo {
    public static void main(String args[]) {
        LinearSearch<Integer> l = new LinearSearch<Integer>();
        Integer arr[] = { 2, 3, 7, 10, 40 };
        int n = arr.length;
        Integer x = 10;

        // Integer result = search(arr, x, n);
        if (l.search(arr, n) == -1)
            System.out.print("Element is not present in array");
        else
            System.out.print("Element is present at index " + l.search(arr, x, n));
    }
}
```

NPTEL Online Certification Courses
IIT Kharagpur

Here I can give some idea about the driver class or the master program, this basically the master program, giving the idea about this one. So it is a linear search demo and this basically the list that we create, it is of type integer. The element stored here in this order and this is basically total number of elements stored and the target element say 10, so we want to search 10. And if you just, call this method, this l is your data structure that you have create a linear search l dot search.

For this class we are calling this method, passing this is the data where it is stored x is the target element and this n is the total number of elements that is stored in the list. Now if return minus 1

means search is unsuccessful and then otherwise it basically search is successful, giving the, it return the location, because this method we have return that way, so it will return it. So this is a simple program that you can write in Java programming and then you can check it, you can verify and then you can modify if you find some other logic to implementable then you can do that.

(Refer Slide Time: 6:59)

The image displays two slides from an NPTEL presentation. The top slide, titled "Binary Search Algorithm", features a coffee cup icon and a background of a tree with various icons. The bottom slide, titled "The technique", shows a diagram of an array with indices l , $mid = (l+u)/2$, and u . Below the diagram, the text reads: "(a) An ordered array of elements with index values l , u and mid ". Both slides include the NPTEL logo and the text "NPTEL Online Certification Courses IIT Kharagpur".

The technique

Search this half the same way
if $K < A[mid]$

(b) Search the entire list turns into the searching of left-half only

NPTEL Online Certification Courses
IIT Kharagpur

The technique

Search this half the same way
if $K > A[mid]$

(c) Search the entire list turns into the searching of right-half only

NPTEL Online Certification Courses
IIT Kharagpur


Now let us see the binary search algorithm, it is better that we can divide the algorithm, you can write the algorithm rather using some recursive, writing recursive method. Here the idea that you can recall, you just have to find the mid location, then speed and then you have to go to either left or right path depending on the value of the key.

(Refer Slide Time: 7:23)

Binary search algorithm

Input: An array $A[1..n]$ of n elements and K is the item of search.
Output: If K is present in the array A then print a successful message and return the index where it is found else print an unsuccessful message and return -1.

```
1. l = 1, u = n // Initialization of lower and upper indexes
2. flag = FALSE // Status of the search, initially false
3. While flag / TRUE and (l <= u) do
4.   mid = (l + u) / 2 // Calculate the index at middle
5.   If (K = A[mid]) then // K matches and search is successful
6.     Print "Successful"
7.     flag = TRUE // Status of the search is now true
8.     Return(mid)
9.   EndIf
10.  If (K < A[mid]) then // Let us check for possibility in left part
11.    u = mid - 1 // This is the rightmost index of the left-half
12.  Else // K > A[mid] and check the possibility at right part
13.    l = mid + 1 // This is the leftmost index of the right-half
14.  EndIf
15. EndWhile
16. If (flag = FALSE) then // Search is failed
17.   Print "unsuccessful"
18.   Return(-1)
19. EndIf
20. Stop
```




Example 49.2: Programming for Binary Search algorithm

```
/* this program implements the Binary Search Algorithm over an array of sorted numbers. */
class BinarySearchCr extends Comparable<T> {
    int binarySearch(T[] arr, int l, int r, T x) {
        if (l >= r) {
            int mid = l + (r - l) / 2;

            // if the element is present at the middle itself
            if (arr[mid] == x)
                return mid;

            // If element is smaller than mid, then it can only be present in left subarray
            if (arr[mid].compareTo(x) < 0)
                return binarySearch(arr, l, mid - 1, x);

            // else the element can only be present in right subarray
            return binarySearch(arr, mid + 1, r, x);
        }
        // We reach here when element is not present in array
        return -1;
    }
} // Continued to next...
```



Example 49.2: Programming for Binary Search algorithm

```

/* This program implements the Binary Search Algorithm over an array of sorted numbers. */
class BinarySearch<T> extends Comparable<T> {
    int binarySearch(T[] arr, int l, int r, T x) {
        if (r >= 0) {
            int mid = l + (r - l) / 2;

            // If the element is present at the middle itself
            if (arr[mid] == x)
                return mid;

            // If element is smaller than mid, then it can only be present in left subarray
            if (arr[mid].compareTo(x) < 1)
                return binarySearch(arr, l, mid - 1, x);

            // Also the element can only be present in right subarray
            return binarySearch(arr, mid + 1, r, x);
        }
        // We reach here when element is not present in array
        return -1;
    }
}

```

// Continued to next...

This is the algorithm that you can follow and following this algorithm the code that I have written for you, this is the code. Again like the previous algorithm we will consider about generic programming, so this is the class that we are defining that means this class contains the method, it is the binary search method, array this one.

Again we are assuming that the elements are stored in an order, it is basically ascending order we are assuming here. This basically tell the list, what is the left most element that means each time you are partitioning the list, so this is basically left most location of the list and then right most location of the list. And x is basically the element to be searched. That means this is the target element.

Now it is the logic that we have already given the algorithm, so we have to just calculate the middle location this formula you can calculate. And then here is the search that you stop, this is the termination condition. Otherwise we can just call the method for other method, so here we are calling compare to x because we have to compare any user type also you can use it here.

Now x, so here basically we call if it is equal then we have to just take the what is the called the, this is basically the left most part. If it is not equals to 1 that means minus 1, then you have to take the right most part. So this basically compare to for x you check that whether we have to go this direction. So this is the code that is return in Java program that you can use to find the location using binary search algorithm and this is return recursively. Because it is the recursive call is there and this is a termination condition that you can think about.

So this program is basically works for binary search algorithm, you can test it, however you can, while you are testing you have to consider several test cases, you can try again with an array of random number also and sorted number and oppositely descending order whether it works or not. Now, so if it is not in order, either an descending order, ascending order, it will not work. And this algorithm plan for ascending order only, if you want to I mean device into descending order, you can just change the code.

By that idea you can think about because this algorithm, this class can be little bit modified by introducing one another argument that it means order, order is basically you can declare it is Boolean, true means ascending order and false means descending order. Then little bit change in the code you can do it and accordingly you can do it.

So, just that left as an exercise for you so that how you can modify, so that this definition, this implementation can works for any type of elements. And it requires that element should be in sorted order, so that is the prerequisite for this technique that you have to think about it.

(Refer Slide Time: 10:36)

Example 49.2: Programming for Binary search algorithm

```
// Continued on...  
  
class BinarySearchDemo{  
    // Driver method to test above  
    public static void main(String args[])  
    {  
        BinarySearch<Integer> ob = new BinarySearch<Integer>();  
        Integer arr[] = { 2, 3, 4, 10, 40 };  
        int n = arr.length;  
        Integer x = 10;  
        int result = ob.binarySearch(arr, 0, n - 1, x);  
        if (result == -1)  
            System.out.println("Element not present");  
        else  
            System.out.println("Element found at index " + result);  
    }  
}
```

NPTEL Online Certification Courses
IIT Kharagpur

Now let us consider the master program, which basically run this, so we create an integer array and then is our target element as it is earlier also. And then we call the method, so this is the starting location 0 and then last location n minus 1, so whole list is a list for that, x is the target and then it basically call this method and the result will be available.

Now if result is minus 1 search is unsuccessful otherwise it will print the result because it return the location of the element. And this is the idea about linear search because if the element stored in an array and the different techniques that you have. So you learn about how to code, write the algorithm for linear search as well as binary search.

(Refer Slide Time: 11:19)

Interpolation Search Algorithm

NPTEL Online Certification Courses
IIT Kharagpur

Interpolation search algorithm: Concept

- The binary search method probes at the center of the region of search.
- It would be better if the locations probed are not at the center rather decided by the key element itself.

0	1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---	---

NPTEL Online Certification Courses
IIT Kharagpur

Interpolation search algorithm: Concept

- The interpolation search calculates the next probe location using the following interpolation formula

$$loc = \left[\frac{K - K_l}{K_u - K_l} \right] \times (u - l) + l$$

0	1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---	---

NPTEL Online Certification Courses
IIT Kharagpur

Now let us come to the discussion of interpolation search. Interpolation search is bit tedious because we have to consider calculation there. So if it is the element then interpolation search basically needs to calculate the address where to heat. It is not exact to the middle location, so this formula basically you should consider to calculate, in which location you have to have the proofing.

So this calculation, this formula needs to be return in Java code, I will give you some idea about how you can calculate this code because it is a key value, key values can be string, key values can be float, key values can be double, so depending on this key values this calculation also varies.

(Refer Slide Time: 12:08)

Interpolation search algorithm

```
1  l = 1, u = n // Initialization: Range of searching
2  flag = FALSE // Hold the status of searching
3  While (flag = FALSE) do
4      loc = floor( (k - a[l]) * (u - l) / (a[u] - a[l]) )
5      If ( l < loc <= u ) then // If loc is within the range of the list
6          Case: K <= A[loc]
7              u = loc - 1
8          Case: K > A[loc]
9              flag = TRUE
10         Case: K >= A[loc]
11             l = loc + 1
12     Else
13         Exit()
14     EndIf
15 EndWhile
16 If (flag) then
17     Print "Successful at" loc
18 Else
19     Print "Unsuccessful"
20 EndIf
21 Stop
```

Example 49.3: Programming for Interpolation search algorithm

```
/* Interpolation Search with Generic type */
class InterpolationSearch<T> extends Number & Comparable<T> {
    // Declaring an array, which should store any type T of data
    T a[]; // Define that the array a[] can store any type of data

    InterpolationSearch(T a[]) { // Define a constructor
        this.a = a;
    }

    T getData(int i) { // To return the element stored in the i th place in the array
        return a[i];
    }

    void printData() { // A generic method to print the elements in array a
        for(int i = 0; i < a.length; i++)
            System.out.println(a[i] + " "); // Print the i th element in a
        System.out.println(); // print a new line
    }
}
```

So that is why the code, this is the algorithm that you can follow. And following this algorithm I have given an code for you, so you can check the code. I am discussing the interpolation search method here; this is the class that we are discussing. We should apply to the number and it should be linked it to comparable that is the generic things it is there, this is the array and this is the constructor that we define, so that the data can be initialized and this is the print data. This basically in order to load the data, in this program I say that, okay the data can be passed as an argument like and then we can just initialize the data for the searching. So this basically the list is ready.

(Refer Slide Time: 12:51)

Example 49.3: Programming for Interpolation search algorithm

```
// Continued on
T sub(T x, T y) {
    if (x == null || y == null)
        return null;
    if (x instanceof Double)
        return (T) new Double(x.doubleValue() - y.doubleValue());
    if (x instanceof Integer)
        return (T) new Integer(x.intValue() - y.intValue());
    if (x instanceof Short)
        return (T) new Short(x.shortValue() - y.shortValue());
    if (x instanceof Byte)
        return (T) new Byte(x.byteValue() - y.byteValue());
    if (x instanceof Float)
        return (T) new Float(x.floatValue() - y.floatValue());
    if (x instanceof Long)
        return (T) new Long(x.longValue() - y.longValue());
    else
        throw new IllegalArgumentException("type " + x.getClass() + " is not supported");
}
// Continued to next...
```

NPTEL Online Certification Course
IIT Kharagpur

Now next part of the program is basically to calculate that address location. So this formula calculate the address location but depending on the different type of value that we have given. So if it is a double, if it is a integer, if it is byte, if it is a float, long, so different calculation is required, so that is why we have given the calculation. So this basically calculate the ak minus kl divided by ku minus kl formula actually there.

(Refer Slide Time: 13:23)

Example 49.3: Programming for Interpolation search algorithm

```
// Continued to next...
int interpolationSearch(T a) {
    int l = 0, u = a.length - 1;
    while (l <= u && k.compareTo(a[l]) >= 0 && k.compareTo(a[u]) <= 0) {
        if (l == u) {
            if (a[l].compareTo(k) == 0) return l;
            return -1;
        }
        int loc = (int) ((float) (a[u]-a[l])) * (u-l) + l;
        int result = a[loc].compareTo(k);
        if (result == 0) return loc;
        else if (result < 0) l = loc;
        else u = loc;
    }
    return -1;
}
// End of the method interpolationSearch
// End of the class InterpolationSearch
// Continued to next...
```

NPTEL Online Certification Course
IIT Kharagpur

And then, so this calculation is required once this is calculated we can use this calculation to calculate the address which basically follows here. Now here we calculate the address that means

this is a sub part that is the formula and then this basically denominator and the numerator calculation, it basically return the integer value so that address because address is basically integer. And we are using this formula because your input data can be any type. That is why we are converting this one.

And then the result is basically right, it basically the value that we have to calculate, you have to comparable that it matches in here, then if it is not matches then we have to just check it and then accordingly we have proof into the same direction. So it is basically the calculation that is there, so this is basically interpolation search method that we are discussing. And so it is basically end of the method interpolation and end of the class interpolation search it is there.

(Refer Slide Time: 14:22)

```
// Continued on...  
  
public class InterpolationSearchDemo {  
    public static void main(String[] args){  
        // Searching with integer data  
        Integer i[] = {10, 20, 30, 40, 50};  
        // Store the data into generic array  
        InterpolationSearch<Integer> arrayInt = new InterpolationSearch<Integer>(i);  
        // printing the data...  
        arrayInt.printData();  
        int searchInt=20;  
        int pos=arrayInt.interpolationSearch(searchInt);  
        if(pos== -1)  
            System.out.println(searchInt+" not found in the array");  
        else  
            System.out.println(searchInt+" found at position "+pos);  
        System.out.println();  
        // Continued to next...
```

Now let us see the master program, the master program is defined here, so this is the master program. Master program here, this is the demo program we can say and this is the list we consider is a i is an array of integers and then we call this right method that we have defined earlier and then that print data, we call the data actually it is there.

Then we call the interpolation search technique here and then result it will return, if it is not successful it return minus 1, otherwise it return the position. Now this example show that for an integer we can again repeat the same (proc), I mean test different testing can be done for the other type of the data which is followed here. So this is for integer array.

(Refer Slide Time: 15:10)

The slide displays the following Java code for searching in a double array:

```
// Continued on ...  
// Searching with float values  
  
double d[] = {10.5, 20.5, 30.5, 40.5, 50.5};  
// Store the data into generic array  
InterpolationSearch<Double> arrayDouble = new InterpolationSearch<Double>(d);  
// Printing the data...  
arrayDouble.printData();  
Double searchDouble=40.5;  
pos=arrayDouble.interpolationSearch(searchDouble);  
if(pos!=-1)  
    System.out.println(searchDouble+" not found in the array");  
else  
    System.out.println(searchDouble+" found at position "+pos);  
System.out.println();  
  
// Continued to next...
```

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

This is for double data it is working.

(Refer Slide Time: 15:14)

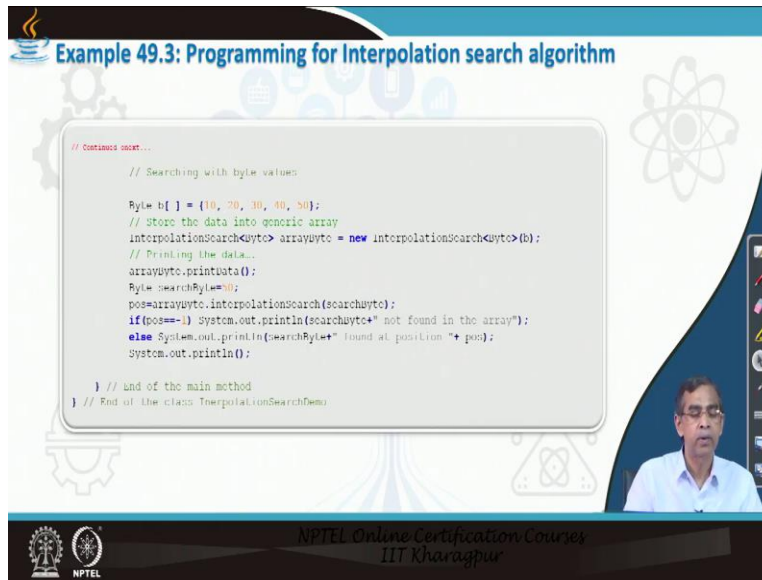
The slide displays the following Java code for searching in a short integer array:

```
// Continued on ...  
// Searching with short values  
  
Short s[] = {10, 20, 30, 40, 50};  
// Store the data into generic array  
InterpolationSearch<Short> arrayShort = new InterpolationSearch<Short>(s);  
// printing the data...  
arrayShort.printData();  
Short searchShort=40;  
pos=arrayShort.interpolationSearch(searchShort);  
if(pos!=-1)  
    System.out.println(searchShort+" not found in the array");  
else  
    System.out.println(searchShort+" found at position "+pos);  
System.out.println();  
  
// 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 for short integer also you can test how it works.

(Refer Slide Time: 15:19)



Example 49.3: Programming for Interpolation search algorithm

```
// Continue next...  
  
// Searching with byte values  
  
Byte b[] = {10, 20, 30, 40, 50};  
// Store the data into generic array  
interpolationSearch<Byte> arrayByte = new interpolationSearch<Byte>(b);  
// Printing the data...  
arrayByte.printData();  
Byte searchByte=30;  
pos=arrayByte.interpolationSearch(searchByte);  
if(pos!=-1) System.out.println(searchByte+" not found in the array");  
else System.out.println(searchByte+" found at position "+ pos);  
System.out.println();  
  
} // End of the main method  
} // End of the class InterpolationSearchDemo
```

NPTEL Online Certification Courses
IIT Kharagpur

This is for byte array also. So for any data type it will work.

(Refer Slide Time: 15:23)



Searching Using JCF

NPTEL Online Certification Courses
IIT Kharagpur

Now we have discussed about three different searching algorithms and you got some idea about that how to write the program and particularly to implement the searching algorithm, so for your hint only I have given the code, that code not necessary to be followed, if you follow the technique and the algorithm that is there better you can write the program of your own, then it is better because this is the code that is written in a professional way.

So learning the code sometimes it maybe tedious but writing the codes of your own, it is better habit. Now let us come to the next part of this discussion where we want to give an idea about that how the searching can be done using build in procedure, that is the procedure defined in Java dot util package and which is the Java collection framework.

(Refer Slide Time: 16:13)

Methods of class Arrays for searching

Method	Description
<code>int binarySearch(<T> array[], <T> value)</code>	It searches for the specified element in the array with the help of Binary Search algorithm. It returns the index of an array location, if found else NULL.

The `binarySearch()` method uses a binary search to find a specified value. This method must be applied to sorted arrays. Here are some of its forms.

```
static int binarySearch(byte array[], byte value)
static int binarySearch(char array[], char value)
static int binarySearch(double array[], double value)
static int binarySearch(float array[], float value)
static int binarySearch(int array[], int value)
static int binarySearch(long array[], long value)
static int binarySearch(short array[], short value)
static int binarySearch(Object array[], Object value)
static <T> int binarySearch(T[] array, T value, Comparator? super T> c)
```

Note: Array of type boolean is not applicable to such a method.

NPTEL Online Certification Courses
IIT Kharagpur

Now Java collection framework provides one class call arrays. So in this class arrays there are many methods are defined to perform the search and this arrays actually class provides one method call searching element and this basically follow only one search technique which is the binary search technique. So there is a method call binary search that be called for the list which is stored in the form of an arrays.

So we have to first store the elements as a collection, collection is an arrays and then you can call the method for there. I am giving an idea about how this method is defined there. Now there is one method called binary search which takes an array of elements and one target value that needs to be searched. And it return integer, so this integer basically return the location if the element is found in the array, otherwise it returns null, null indicates that the element is not present.

Now this method can be called for different type, for byte array like the interpolation search we have discussed in the same way, it can be called for character array, it can be called double, float, int, long, short. So all user data type it can be used except the Boolean, so this basically does not

support the Boolean type to be performed. Actually Boolean, searching over the Boolean does not matters actually, that is why it is not in to date.

Also you can use this call for any type of object arrays, that means it can be any user defined type also you can use it but in that case, the user defined type should be compatible with the compare to method. If compare to method not defined, then you have to overwrite the compare to method according to which it basically compare the element. So that is very important factor that you should consider.

Ad so that is why here if you have the compare to method to be defined then you can define then it can be applied to any type of, type, any type of array actually, any type of array you can apply. So this basically tell that how the searching can be done using binary search method which is already defined in arrays and this is the only class where it is defined, for any other collection the search technique is not there actually.

(Refer Slide Time: 18:55)

```
// This program illustrates the use of Binary Search method.
import java.util.Arrays;

public class BinarySerachArraysDemo {
    public static void main(String[] args){
        // Get the Array
        int intArr[] = { 10, 20, 15, 22, 35};
        Arrays.sort(intArr);
        int intKey = 22;
        System.out.println(intKey
            + " found at index - "
            + Arrays.binarySearch(intArr, intKey));
    }
}
```

Now we can just create the list for an illustration and then you can check how it is work, how it works for, there is a code that you can follow. This code says using this, these arrays, array is basically collection that I told you that the method binary search is defined in the collection arrays, so you have import this. And you have to create an integer array, defined int integer array according to your own, I am applying for integer array, you can do for long, short, whatever it is there.

Now here is a arrays dot short you see I am calling this short method because binary search needs to be applied to a sorted array and this may not be sorted, so we have apply the short technique is there they passing this integer array, it basically called the short method, and this arrays you know it is 30 plus, you do not have to create a new object, so that is what we are doing here.

Now so this integer array is now sorted, if you can print then you can print the array, then you can see it will basically elements are stored in a sorted order. Now once it is sorted then you can just decide one target key, that is mean by the key that you want to search, let it be 22, then you can call this binary search method which is defined in this class. So binary search array, you pass the array that you want to search, here the integer array and the target that is the int key that is defined there.

So this you call for the arrays class, so this basically return either the location where it is found or it basically return null, so then you can print the result. So this is the idea, now I get the idea about a particular array, now it is up to you to test whether the same technique applied for any type of data long, float, double or not. And more interestingly it gives an it a, okay I left it as an exercise to study, if you want to apply the same techniques for a screen time.

That means some names are given and you want to search a given name, that you can take it and I can tell that okay it is possible and you have to because string is also, it is a compatible to compare to method, so it will work for you also and short also method it is there comparable. So you can apply it.

Now so that is fine for a any numbers as well as string, now again I gave many hints while I was discussing the program previously, I do not what to repeat it again here. The idea is that you can define some user define type say book or student or person, then you can perform searching. And how you can do that? So that you can think about it, only the thing is that you have to overwrite the compare to method because compare to method is not defined.

I gave some idea about how you can do that in some discussion earlier also, you can follow, if you do not forget it, then you can just think, I think I have discussed it while I was discussing about mapping then that portion it is there. So anyway that is possible also, it is left as an exercise, little bit work hard so that you can think something of your own and finally you can

solve some problem, then it basically increase your confidence. So this is basically the idea about how the JCF can be used and then it can be solved there.

(Refer Slide Time: 22:24)

Method of class Arrays for searching (sub array)

Method	Description
<code>int binarySearch(<u><T> array[]</u>, <u>int start</u>, <u>int end</u>, <u><T> value</u>)</code>	It searches for the specified element in the sub array from the location start to end, both inclusive with the help of Binary Search algorithm. It returns the index of an array location, if found else NULL.

Note:

- This method is **not** applicable to an array of type **boolean**.

NPTEL Online Certification Courses
IIT Kharagpur

So this is basically another technique that you can apply, this binary search also can be applied to the part of the arrays, not the entire arrays, sub list. And here is the code that you can think about, so that how it is working.

(Refer Slide Time: 22:45)

Example 49.5: Binary search on a sub-list of an array

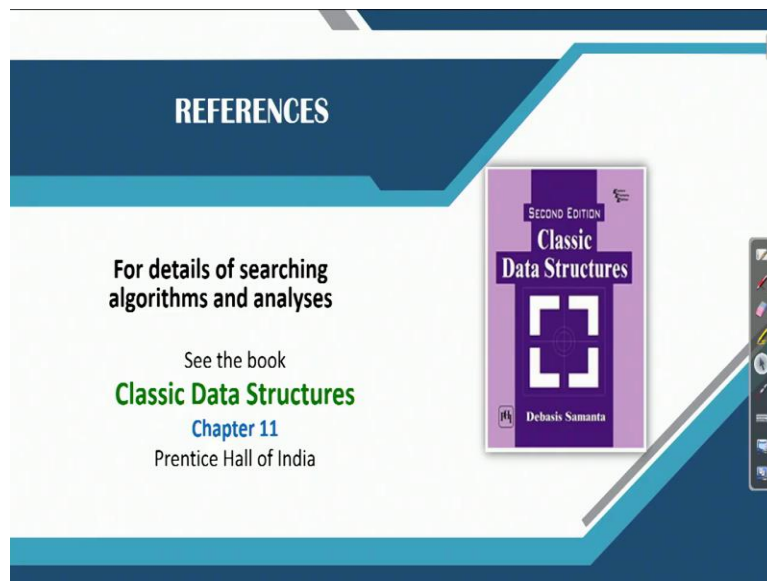
```
/* This program illustrates the use of Binary Search method within a sub list. */  
  
import java.util.Arrays;  
  
public class ArraysBinarySearchSublistDemo {  
    public static void main(String[] args) {  
        int intArr[] = { 10, 20, 15, 22, 35}; // An int array as input  
        Arrays.sort(intArr); // Sort the array  
        int intKey = 22;  
        System.out.println ( intKey + " found at index - "  
            + Arrays.binarySearch(intArr, 1, 3, intKey));  
    }  
}
```

NPTEL Online Certification Courses
IIT Kharagpur

There is a code that we have given, it is the same thing, this is the integer array, it is a input list, these are target element, we sort it and then call the same method but within this 1 and 3, that

basically it is a sub part, sub list starting from the location 1 and to 3, inclusive 1 and 3 both then key. So you can test it that it is also working for you. Anyway so these are the facilities that is provided in the Java collection framework and then you can utilize it.

(Refer Slide Time: 23:17)



So this is fine for different searching techniques and then I have given the programming also, there are many other searching techniques also included in this book that you can consider as a case study and you can write the program for each. It will be really gives enough what is called the enough rather scope to improve your skill. So there are several, actually these algorithms are only mean for this purpose.

Because if you practice the algorithm your programming skill will improve, that is why more programming, I mean more algorithm implementation, more programming and more competency that you can learn. So if I give you the readymade solution as a programming that is very bad, because your spin putting is not good always because you cannot improve your programming skill. I gave some idea about that okay you can follow this idea and write the code of your own that is a best way to learn any programming things actually. Thank you very much.