Linear Search in Java

Linear search is used to search a key element from multiple elements. Linear search is less used today because it is slower than binary search and hashing.

Algorithm:

- Step 1: Traverse the array
- Step 2: Match the key element with array element
- Step 3: If key element is found, return the index position of the array element
- Step 4: If key element is not found, return -1.

Code for Linear Search

```java
package LinearSearch;

public class LinearSearch {
    public static void main(String[] args) {

        int[] arr = {1,24,4,6,6,4,6,854,55,45};
        System.out.println(linearSearch(arr, 4));
    }
    public static int linearSearch(int[] arr, int target) {
        if (arr.length==0){
            return -1;
        }
        for (int i=0; i<arr.length; i++) {
            int element = arr[i];
            if (element == target){
                return i;
            }
        }
        return -1;
    }
}
```

Arrays to String

```java
package LinearSearch;

import java.util.Arrays;

public class ArraystoString {
    public static void main(String[] args) {
        char[] ch = {'m','e','h','u','l','m','e','s','h','r','a','m'};
        System.out.println(toStr(ch));
    }

    public static String toStr(char[] ch) {
        String str = String.valueOf(ch);
        return str;
    }
}
```

Even No Digit

```java
package LinearSearch;

public class EvenNoDigits {
    public static void main(String[] args) {
        int[] arr = {23,2,43,545,34,5455,342};
        System.out.println(isEven(arr));
    }

    static int isEven(int[] arr) {
        int even = 0;
        int count = 0;
        for (int ar: arr) {
            count = (int) (Math.log10(ar))+1;

            if (count%2==0) {
                even++;
            }
        }
        return even;
    }
}
```

find Max_Min

```java
package LinearSearch;

public class Max_Min {
    public static void main(String[] args) {
        int[] arr = {343,4,4,33,7,55,34,76};
        search(arr);
    }
    public static void search(int[] arr) {
        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;


        for (int i = 0; i < arr.length; i++) {
            if (max<arr[i]) {
                max = arr[i];
            }
            if (min>arr[i]) {
                min = arr[i];
            }
        }
        System.out.println("max no is: "+max);
        System.out.print("min no is: "+min);

    }
}
```

Max in 2D Arrays

```
package LinearSearch;

import java.util.Arrays;

public class MaxIn2DArray {

        public static void main(String[] args) {
            int[][] arr = {
                    {1,3,5},
                    {54,653,0,53},
                    {8,6}
            };
            int ans = max(arr);
            System.out.println(ans);
        }

        public static int max(int[][] arr) {
            int max = Integer.MIN_VALUE;
            for (int i = 0; i < arr.length; i++) {
                for (int j = 0; j < arr[i].length; j++) {
                    if (max < arr[i][j]) {
                        max = arr[i][j];
                    }
                }
            }
            return max;

        }
}
```

Find Min Number

```
package LinearSearch;

public class MinNumber {
    public static void main(String[] args) {
        int[] arr = {4,6,4,6,78,86,4,43,5,54};
        System.out.println(min(arr));
    }
    static int min(int[] arr) {

        int min = arr[0];
        for (int i = 0 ; i < arr.length; i++) {
            if (min>arr[i]) {
                min = arr[i];
            }
        }
        return min;
    }
}
```

Search in 2D Array

```
package LinearSearch;

import java.util.Arrays;
public class SearchIn2DArray {
    public static void main(String[] args) {
        int[][] arr = {
                {1,3,5},
                {54,653,0,53},
                {8,6}
        };
        int[] ans = search(arr, 0);
        System.out.println(Arrays.toString(ans));
    }

    public static int[] search(int[][] arr, int target) {
        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[i].length; j++) {
                if (target == arr[i][j]) {
                    return new int[]{i, j};
                }
            }
        }
        return new int[]{-1,-1};
    }
}
```

Searching Char in String

```
package LinearSearch;

import javax.naming.PartialResultException;
import java.util.Arrays;
public class SearchingInString {
    public static void main(String[] args) {
        String name = "aniket";
        char c = 'a';

        System.out.println(Arrays.toString(name.toCharArray())); // change
string into String Arrays
        System.out.println(search(name, c));
    }
```

```java
    static boolean search(String name, char ch) {
        if (name.length()==0) {
            return false;
        }

        for (int i = 0 ; i < name.length(); i++){
            if (ch==name.charAt(i)) {
                return true;
            }
        }


        //another way
        for (char str: name.toCharArray()){
            if (ch==str) {
                return true;
            }
        }
        return false;
    }
}
```

Search In Range

```java
package LinearSearch;

public class SearchInRange {
    public static void main(String[] args) {
        int[] arr = {1,2,3,-4,5,7,8};
        System.out.println( search(arr, 1, 2, 5));
    }

    public static int search(int[] arr, int target, int start, int end){

        if (arr.length == 0) {
            return -1;
        }
        for (int i = start; i<=end; i++) {
            if (target==arr[i]){
                return i;
            }
        }
        return -1;
    }
}
```

String to Arrays

```
package LinearSearch;


import java.util.Arrays;
public class StringToArray {
    public static void main(String[] args) {
        String name = "mehul";
        System.out.println(Name(name));
    }

    public static String Name(String name) {
        String str = "";
        for (int i = 0; i < name.length(); i++) {
             str = Arrays.toString(name.toCharArray());
        }
        return str;
    }
}
```