**Linear Search:**

Time Complexity🡪Best: O(1) [If the ele is at 0th index]

[only one comparison irrespective of

Size of the array].

Worst:O(n) where n is size of array

[Does not find the item, Iterate

through every item, number of

comparisons is equal to size of array].

Time Complexity tells about how the time increases with increase in input.

**Linearsearch:**

**public** **class** main {

**public** **static** **void** main(String[] args) {

**int** [] nums= {23,45,1,2,8,19,-3,16,-11,28};

**int** target=55;

**int** ans=*linearSearch2*(nums,target);

System.***out***.println(ans);

}

//search in the array: return the index if item found otherwise return -1;

**static** **int** linearSearch(**int**[] arr, **int** target) {

**if**(arr.length==0) {

**return** -1;

}

//run a for loop

**for**(**int** index=0;index<arr.length;index++) {

//check for element at every index if its is=target]

// int element=arr[index];

**if**(arr[index]==target) {

**return** index;

}

}

// this line gets executed if any of the above return statements have not executed hence the target not found

**return** -1;

}

//search for the element and return the element

**static** **int** linearSearch2(**int**[] arr, **int** target) {

**if**(arr.length==0) {

**return** Integer.***MAX\_VALUE***;

}

//run a for loop

**for**(**int** element:arr) {

//check for element at every index if its is=target]

**if**(element==target) {

**return** element;

}

}

// this line gets executed if any of the above return statements have not executed hence the target not found

**return** Integer.***MAX\_VALUE***;

}

//search for the element and return true or false

**static** **boolean** linearSearch3(**int**[] arr, **int** target) {

**if**(arr.length==0) {

**return** **false**;

}

//run a for loop

**for**(**int** index=0;index<arr.length;index++) {

//check for element at every index if its is=target]

**int** element=arr[index];

**if**(element==target) {

**return** **true**;

}

}

// this line gets executed if any of the above return statements have not executed hence the target not found

**return** **false**;

}

}

**Search In Range:**

**public** **class** SearchInRange {

**public** **static** **void** main(String[] args) {

**int**[] arr= {18,12,-7,3,14,28};

**int** target=18;

System.***out***.println(*linearSearch*(arr,target,1,4));

}

**static** **int** linearSearch(**int**[] arr, **int** target, **int** start, **int** end) {

**if**(arr.length==0) {

**return** -1;

}

//run a for loop

**for**(**int** index=start;index<=end;index++) {

//check for element at every index if its is=target]

**int** element=arr[index];

**if**(element==target) {

**return** index;

}

}

// this line gets executed if any of the above return statements have not executed hence the target not found

**return** -1;

}

}

**SearchInStrings**

**import** java.util.Arrays;

**public** **class** SearchInStrings {

**public** **static** **void** main(String[] args) {

String str="Hemasri Seetha";

**char** target='S';

**boolean** ans=*search*(str,target);

// System.out.println(ans);

System.***out***.println(Arrays.*toString*(str.toCharArray()));

}

//using for each

**static** **boolean** search2(String str, **char** target) {

**if**(str.length()==0) {

**return** **false**;

}

**for**(**char** ch:str.toCharArray()) {//{

**if**(ch==target) {

**return** **true**;

}

}

**return** **false**;

}

**static** **boolean** search(String str, **char** target) {

**if**(str.length()==0) {

**return** **false**;

}

**for**(**int** i=0;i<str.length();i++) {

**if**(target==str.charAt(i)) {

**return** **true**;

}

}

**return** **false**;

}

}

**Search In 2D Arr:**

**import** java.util.Arrays;

**public** **class** SearchIn2DArr {

**public** **static** **void** main(String[] args) {

**int**[][] arr= {

{23,4,1},

{18,12,3,9},

{78,99,34,56},

{18,12}

};

**int** target=56;

**int**[] ans=*search*(arr,target);//format of return value {row,col}

System.***out***.println(Arrays.*toString*(ans));

//System.out.println(max2(arr));

}

**static** **int** max(**int**[][]arr) {

**int** ans=Integer.***MIN\_VALUE***;

**for**(**int** row=0;row<arr.length;row++) {

**for**(**int** col=0;col<arr[row].length;col++) {

**if**(arr[row][col]>ans) {

ans=arr[row][col];

}

}

}

**return** ans;

}

//find maximum using enhanced for loop

**static** **int** max2(**int**[][]arr) {

**int** ans=Integer.***MIN\_VALUE***;

**for**(**int**[] ints:arr) {

**for**(**int** element:ints) {

**if**(element>ans) {

ans=element;

}

}

}

**return** ans;

}

**static** **int**[] search(**int**[][]arr,**int** target) {

**for**(**int** row=0;row<arr.length;row++) {

**for**(**int** col=0;col<arr[row].length;col++) {

**if**(arr[row][col]==target) {

**return** **new** **int**[] {row,col};

}

}

}

**return** **new** **int**[] {-1,-1};

}

}

**Find Min:**

**public** **class** FindMin {

**public** **static** **void** main(String[] args) {

**int**[] arr= {18,12,-7,3,14,28};

System.***out***.println(*min*(arr));

}

//assume arr.length!=0

**static** **int** min(**int**[] arr) {

**int** ans=arr[0];

**for**(**int** i=1;i<arr.length;i++) {

**if**(arr[i]<ans) {

ans=arr[i];

}

}

**return** ans;

}

}

**Given an array nums of integers, return how many of them contain an even number of digits.**

**public** **class** evennumber {

**public** **static** **void** main(String[] args) {

**int** []nums= {12,345,2,6,7896};

//System.out.println(findNumbers(nums));

System.***out***.println(*digits2*(-234));

}

**static** **int** findNumbers(**int**[] nums) {

**int** count=0;

**for**(**int** num : nums) {

**if**(*even*(num)) {

count++;

}

}

**return** count;

}

//function to check whether number of digits is even or not

**static** **boolean** even(**int** num) {

**int** numberofDigits=*digits*(num);

**return** numberofDigits%2==0;

}

//function to count number of digits

**static** **int** digits(**int** num) {

**if**(num<0) {

num=num\*-1;

}

**if**(num==0) {

**return** 1;

}

**int** count=0;

**while**(num>0) {

count++;

num=num/10;

}

**return** count;

}

**static** **int** digits2(**int** num) {

**if**(num<0) {

num=num\*-1;

}

**return** (**int**)(Math.*log10*(num)+1);

}

}