Java Program

1.Bubble sort

**package** interviewQ;

**public** **class** BubbleSort {

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

// **TODO** Auto-generated method stub

**int** arr[] = **new** **int**[]{10,23,1,90,34};

**int** len = arr.length;

System.***out***.println("Before sorting an array is");

**for**(**int** i=0;i<len;i++){

System.***out***.print(arr[i]+"\t");

}

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

**for**(**int** i=0;i<len-1;i++){

**for**(**int** k=0;k<len-i-1;k++){

**if**(arr[k]>arr[k+1]){

**int** temp = arr[k];

arr[k] = arr[k+1];

arr[k+1] = temp;

}

}

}

System.***out***.println("After sorting an array is ");

**for**(**int** i=0;i<len;i++){

System.***out***.print(arr[i]+"\t");

}

}

}

2.Merge sort

3.Decimal to binary

**package** interviewQ;

**public** **class** BinaryNumber {

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

// **TODO** Auto-generated method stub

**int** val =10;

**int** temp=0;

String sum=" ";

**while**(val>0){

temp = val%2;

sum = sum+temp;

val =val/2;

}

//System.out.println(sum);

**for**(**int** i=sum.length()-1;i>0;i--){

System.***out***.print(sum.charAt(i)+"\t");

}

}

}

4.binary to decimal

**package** interviewQ;

**public** **class** DecimalNumber {

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

// **TODO** Auto-generated method stub

String binValue = "1011011";

**int** mulByTwo = 1;

**int** decVal=0;

System.***out***.println(Integer.*parseInt*(binValue, 2));//built-in method to test our result with this statement

//System.out.println(Integer.toBinaryString(Integer.parseInt(binValue, 2)));

**int** len = binValue.length();

**for**(**int** i=len-1;i>=0;i--){

String charVal = " "+binValue.charAt(i);

**int** temp = Integer.*parseInt*(charVal.trim());

**if**(temp==1 && (len-1)==i){

mulByTwo = mulByTwo\*1;

decVal = decVal+mulByTwo;

}

**else** **if**(temp==1){

mulByTwo = mulByTwo\*2;

decVal = decVal+mulByTwo;

}

**else** **if**(temp!=1 && (len-1)==i){

mulByTwo = mulByTwo\*1;

}

**else**{

mulByTwo = mulByTwo\*2;

}

}

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

}

}

5.Linear search

**package** interviewQ;

**public** **class** DeleteElementsFromArray {

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

// **TODO** Auto-generated method stub

**int** a[] = **new** **int**[]{12,2,9,23,2};

**int** del =9;

**int** count=0;

**int** pos=0;

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

**if**(a[i]==del){

pos = i;

**for**(**int** j=i;j<a.length-1;j++){

a[j]=a[j+1];

}

count++;

**break**;

}

}

**if**(count!=0){

System.***out***.println("Element is found at position "+pos+"and deleted from an array");

}

**else**{

System.***out***.println("Element is not found in array");

}

System.***out***.println("After deletion of an array from positioned");

**for**(**int** i=0;i<a.length-1;i++){

System.***out***.print(a[i]+"\t");

}

}

}

Time complexity is o(n)

6.Binary search

Binary search can be possible on Sorted order only.

Algorithm

We basically ignore half of the elements just after one comparison.

1. Compare x with the middle element.
2. If x matches with middle element, we return the mid index.
3. Else If x is greater than the mid element, then x can only lie in right half subarray after the mid element. So we recur for right half.
4. Else (x is smaller) recur for the left half.

**package** interviewQ;

**class** BinarySearch

{

**int** binarySearch(**int** arr[], **int** l, **int** r, **int** x)

{

**if** (r>=l)

{

**int** mid = l + (r - l)/2;

**if** (arr[mid] == x)

**return** mid;

**if** (arr[mid] > x)

**return** binarySearch(arr, l, mid-1, x);

**return** binarySearch(arr, mid+1, r, x);

}

**return** -1;

}

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

{

BinarySearch ob = **new** BinarySearch();

**int** arr[] = {2,3,4,10,40};

**int** n = arr.length;

**int** x = 40;

**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);

}

}

7.Find missing number from array

**package** interviewQ;

**public** **class** FindMissingWord {

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

// **TODO** Auto-generated method stub

**int** a[] = **new** **int**[]{1,4,5,6,2};

**int** missedWord = *missedWord*(a,a.length);

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

}

**private** **static** **int** missedWord(**int**[] a, **int** length) {

// **TODO** Auto-generated method stub

**int** total = (length+1)\*(length+2)/2;

**for**(**int** j=0;j<length;j++){

total-=a[j];

}

**return** total;

}

}

8.Find out the finocci series with recursion

Without recursion

**public** **class** FobonacciSeries {

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

// **TODO** Auto-generated method stub

**int** first=0,second=1;

System.***out***.print(first+"\t"+second);

//System.out.println("\t");

**for**(**int** i=2;i<=10;i++){

**int** third =first+second;

System.***out***.print(third+"\t");

first = second;

second = third;

}

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

}

}

With recursion

**package** interviewQ;

**class** FibonacciSeries

{

**static** **int** *n1*=0,*n2*=1,*n3*=0;

**static** **void** printFibonacci(**int** count){

**if**(count>0){

*n3* = *n1* + *n2*;

*n1* = *n2*;

*n2* = *n3*;

System.***out***.print(" "+*n3*);

*printFibonacci*(count-1);

}

}

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

**int** count=10;

System.***out***.print(*n1*+" "+*n2*);//printing 0 and 1

*printFibonacci*(count-2);//n-2 because 2 numbers are already printed

}

}

9.Find add all the number present in string

**package** interviewQ;

**public** **class** NumberInString {

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

// **TODO** Auto-generated method stub

String str = "43String231";

**int** sum=0;

**int** len = str.length();

**for**(**int** i=0;i<len;i++){

**if**(Character.*isDigit*(str.charAt(i))){

sum = sum+Integer.*parseInt*((" "+str.charAt(i)).trim().toString());

}

}

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

}

}

10.count number of repeated character present in string using own logic and using collection.

Using collection

**package** interviewQ;

**import** java.util.HashMap;

**import** java.util.Map;

**public** **class** RepeatedCharacter {

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

// **TODO** Auto-generated method stub

String str = "Malyalamboyinnewyorkstreat";

Map<Character,Integer> repeatedChar = **new** HashMap<Character,Integer>();

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

**if**(repeatedChar.containsKey(str.charAt(i))){

repeatedChar.put(str.charAt(i), repeatedChar.get(str.charAt(i))+1);

}

**else**{

repeatedChar.put(str.charAt(i), 1);

}

}

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

}

}

Using logic and without any collection or utility method.

package interviewQ;

import java.util.HashMap;

import java.util.Map;

public class RepeatedCharacter1 {

public static void main(String[] args) {

String s = "vickyjaiswalatnewyorkstreat";

int distinct=0;

System.out.println(s.length());

for(int i=0;i<s.length();){

for(int j=0;j<s.length();j++){

if(s.charAt(i)==s.charAt(j)){

distinct++;

}

}

System.out.print(s.charAt(i)+"--"+distinct+"\t");

String sRepVal = String.valueOf(s.charAt(i)).toString();

s = s.replaceAll(sRepVal, "");

distinct=0;

}

}

}

11.Swap two number without third variable.

a.by divide

a =10 , b=5

a = a\*b;//50

b = a/b;//5

a = a/b//10

b.by addition

a =10 , b=5

a=a+b;//15

b=a-b;//10

a=a-b;//5

c.by one’s completement

x=10 in binary=1010

y = 5 in binary 0101

x=x^y;

x = x xor y

1010

Xor 0101

\_\_\_\_\_\_\_\_\_

1111

y = x^y

y = 1111 Xor 0101=1010

x = x^y

x = 1111 xor 1010 = 0101

0 for same different for 1

12.Find the HCF or GCD and LCF

13.reverse the string

**package** interviewQ;

**public** **class** ReverseString {

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

// **TODO** Auto-generated method stub

String name= "Vicky";

**for**(**int** i =name.length()-1;i>=0;i--){

System.***out***.println(name.charAt(i));

}

}

}

14.reverse the number

**package** interviewQ;

**public** **class** ReverseTheNumber {

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

// **TODO** Auto-generated method stub

**int** number = 12697;

**int** temp=0 ;

**int** rev=0;

**while**(number>0){

temp = number%10;

rev = rev\*10+temp;

number = number/10;

}

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

}

}

15.count number of digit in given number

**package** interviewQ;

**public** **class** CountNumber {

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

// **TODO** Auto-generated method stub

**int** num = 123;

**int** sum = 0;

**int** count = 0;

**int** temp = 0;

**while** (num > 0) {

temp = num % 10;

sum = sum + temp;

num = num / 10;

count++;

}

System.***out***.println("sum of the number=" + sum + " "

+ "total number present in number" + " " + count);

}

}

16. How do you find the duplicate number on a given integer array?

**public** **class** DuplicateNumber {

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

// **TODO** Auto-generated method stub

**int** [] arry = {10,2,3,1,2,3,17,17,6,7};

**for**(**int** i=0;i<arry.length-1;i++){

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

**if**(arry[i]==arry[j] && i!=j){

System.***out***.println("Duplicate Element:="+arry[i]);

}

}

}

}

}

17. How do you find the largest and smallest number in an unsorted integer array?

Ans:-

**package** interviewQ;

**class** Test

{

**static** **int** *arr*[] = {500,10, 3240, 45, 90, 98};

// Method to find maximum in arr[]

**static** **int** largest()

{

**int** i;

// Initialize maximum element

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

// Traverse array elements from second and

// compare every element with current max

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

**if** (*arr*[i] > max)

max = *arr*[i];

**return** max;

}

**static** **int** smallest()

{

**int** i;

// Initialize maximum element

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

// Traverse array elements from second and

// compare every element with current max

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

**if** (*arr*[i] < min)

min = *arr*[i];

**return** min;

}

// Driver method

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

{

System.***out***.println("Largest in given array is " + *largest*());

}

}

18. How do you find all pairs of an integer array whose sum is equal to a given number?

**package** interviewQ;

**public** **class** PairOfElementsInArray {

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

// **TODO** Auto-generated method stub

FindPairs p = **new** FindPairs(**new** **int**[]{21,32,12,12,-24,36},24);

FindPairs p1 = **new** FindPairs(**new** **int**[] {4, 6, 5, -10, 8, 5, 20}, 10);

FindPairs p2 = **new** FindPairs(**new** **int**[] {4, -5, 9, 11, 25, 13, 12, 8}, 20);

FindPairs p3 = **new** FindPairs(**new** **int**[] {12, 13, 40, 15, 8, 10, -15}, 25);

FindPairs p4 = **new** FindPairs(**new** **int**[] {12, 23, 125, 41, -75, 38, 27, 11}, 50);

}

}

19. How do you find duplicate numbers in an array if it contains multiple duplicates?

**public** **class** DuplicateNumber {

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

// **TODO** Auto-generated method stub

**int** [] arry = {10,2,3,1,2,3,17,17,6,7};

**for**(**int** i=0;i<arry.length-1;i++){

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

**if**(arry[i]==arry[j] && i!=j){

System.***out***.println("Duplicate Element:="+arry[i]);

}

}

}

}

}

**package** interviewQ;

**public** **class** FindPairs {

**public** FindPairs(**int**[] is, **int** i) {

// **TODO** Auto-generated constructor stub

**int** len=is.length;

**for**(**int** l=0;l<len;l++){

**for**(**int** k=l+1;k<len;k++){

**if**(is[l]+is[k]==i){

System.***out***.println(is[l]+"+"+is[k]+"= "+i);

}

}

}

}

}

20. How are duplicates removed from a given array in Java

**public** **class** RemoveDuplicateElementFromArray {

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

// **TODO** Auto-generated method stub

**int** arr[] = {10,10,20,20,30,30,50};

**int** len = arr.length;

**int** j=0;

**for**(**int** i=0;i<len-1;i++){

**if**(arr[i]!=arr[i+1]){

arr[j++] = arr[i];

}

}

arr[j++]=arr[len-1];

**for**(**int** k=0;k<j;k++){

System.***out***.println(arr[k]);

}

}

}

21

|  |  |
| --- | --- |
| Q: | Why can't you use switch with strings? |
| A: | Strings are objects, and switch in Java works only for the primitive types byte, char, short, and int. To compare strings, you have to use nested ifs, which enable more general expression tests, including string comparison. |

22. How is an integer array sorted in place using the quicksort algorithm?

23. How do you reverse an array in place in Java?

**package** interviewQ;

**public** **class** ReversenArray {

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

// **TODO** Auto-generated method stub

**int** a[] = {1,3,4,3,6,7,8,10};

**for**(**int** i =a.length-1;i>=0;i--){

System.***out***.print("\t"+a[i]);

}

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

}

}

24. How are duplicates removed from an array with library library?

**package** interviewQ;

**import** java.util.Arrays;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

**public** **class** RemoveDuplicateElementFromArrayUsingCollection {

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

// **TODO** Auto-generated method stub

Integer array[] = {10,20,10,30,50,60,30};

Set setArray = **new** HashSet();

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

setArray.add(array[i]);

}

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

}

}

25. How do you find the second largest and smallest number in an unsorted integer array?

**package** interviewQ;

**class** GFG {

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

**int** arr[] = { -1,14,18,-1,7,20,22 };

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

**int** secondLargest=0;

System.***out***.println("The given array is:" );

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

System.***out***.print(arr[i]+"\t");

}

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

**if** (arr[i] > largest) {

secondLargest = largest;

largest = arr[i];

} /\*else if (arr[i] > secondLargest) {

secondLargest = arr[i];

}\*/

}

System.***out***.println("\nSecond largest number is:" + secondLargest);

}

}

27.)what is alternative of InstanceOf?

Ans:-Polymorphism

e.g with InstanceOf

**public** **final** **class** BadInstanceOf {

**public** **static** **void** doSomething(Animal animal){

**if** (animal **instanceof** Fish){

Fish fish = (Fish)animal;

fish.swim();

}

**else** **if** (animal **instanceof** Spider){

Spider spider = (Spider)animal;

spider.crawl();

}

}

// PRIVATE

**private** **static** **class** Animal {}

**private** **static** **final** **class** Fish **extends** Animal {

**void** swim(){}

}

**private** **static** **final** **class** Spider **extends** Animal {

**void** crawl(){}

}

}

e.g. with Polymorphism

The mistake is corrected by using an overridable method: 

/\*\*

\* Using polymorphism to call different methods.

\* Does not use instanceof.

\*/

**public** **final** **class** BadInstanceOfFixed {

**public** **static** **void** main(String... aArgs){

log("Starting...");

Animal animal = **new** Animal();

doSomething(animal);

//repoint the same 'animal' reference to other objects:

animal = **new** Fish();

doSomething(animal);

animal = **new** Spider();

doSomething(animal);

log("Done.");

}

/\*\*

\* This method implementation doesn't care at all

\* about Fish/Spider. It just knows that it has

\* been passed an Animal. Different versions of

\* 'move' are called, specific to each Animal.

\*/

**public** **static** **void** doSomething(Animal aAnimal){

aAnimal.move();

}

// PRIVATE

**private** **static** **class** Animal {

**void** move(){

log("Move like an animal...");

}

}

**private** **static** **final** **class** Fish **extends** Animal {

**@Override** **void** move(){

log("Move like a fish...");

}

}

**private** **static** **final** **class** Spider **extends** Animal {

**@Override** **void** move(){

log("Move like a spider...");

}

}

**private** **static** **void** log(String aMessage){

System.out.println(aMessage);

}

}

3rd approach is getClass in reflection

4th approach is visitor pattern in java 1.8

Object msg = //...

whenTypeOf(msg).

is(Date.class). then(date -> println(date.getTime())).

is(String.class). then(str -> println(str.length())).

is(Number.class). then(num -> println(num.intValue())).

orElse(obj -> println("Unknown " + obj));

28.)why we use static in java?

Suppose we have a class employee

First thing is the name of the employee

So the value of the string will vary for each employee but when it comes to company name it will be same for all those employees.

1.When we have any value common to all objects in that case we make it static.

Whenever static variable is declared in java it belongs to class not the object.

In runtime only one string variable of company will be created which will also help in saving the memory.

You will also notice that the main method is also static …..so in order to execute any java program you need to have atleast one static main method or else you will not be able to execute your program.

2. If a method doesn't modify state of object, or not using any instance variables.

3.

Real example of static given below.

Factory design pattern

Ans:-

29.)which one is god to use String or StringBuilder?

Since String is immutable in java, whenever we do String manipulation like concat, substring etc, it generates a new String and discard the older String for garbage collection.

These are heavy operations and generate a lot of garbage in heap. So Java has provided StringBuffer and StringBuilder class that should be used for String manipulation.

StringBuffer and StringBuilder are mutable objects in java and provide append(), insert(), delete() and substring() methods for String manipulation.

30.)write a program to create a utilities method for the union and intersection

**package** interviewQ;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

**public** **class** IntersectionAndUnionOfList {

**public** **static** **void** main(String... args) **throws** Exception {

List<String> list1 = **new** ArrayList<String>(Arrays.*asList*("A", "B", "C"));

List<String> list2 = **new** ArrayList<String>(Arrays.*asList*("B", "C", "D", "E", "F"));

System.***out***.println(**new** IntersectionAndUnionOfList().intersection(list1, list2));

System.***out***.println(**new** IntersectionAndUnionOfList().union(list1, list2));

}

**public** <T> List<T> union(List<T> list1, List<T> list2) {

Set<T> set = **new** HashSet<T>();

set.addAll(list1);

set.addAll(list2);

**return** **new** ArrayList<T>(set);

}

**public** <T> List<T> intersection(List<T> list1, List<T> list2) {

List<T> list = **new** ArrayList<T>();

**for** (T t : list1) {

**if**(list2.contains(t)) {

list.add(t);

}

}

**return** list;

}

}

//written by me

**package** interviewQ;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

**public** **class** IntersectionAndUnion {

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

ArrayList<Integer> listA = **new** ArrayList<Integer>(Arrays.*asList*(2,3,4,5,6));

ArrayList<Integer> listB = **new** ArrayList<Integer>(Arrays.*asList*(2,3,4,5,6,7,1));

System.***out***.println(*union*(listA,listB));

System.***out***.println(*intersection*(listA,listB));

}

**private** **static** List<Integer> intersection(ArrayList<Integer> listA,

ArrayList<Integer> listB) {

// **TODO** Auto-generated method stub

//Set<Integer>

List<Integer> list = **new** ArrayList<Integer>();

**for**(Integer val :listA){

**if**(listB.contains(val)){

list.add(val);

}

}

**return** list;

}

**private** **static** Set<Integer> union(ArrayList<Integer> listA,

ArrayList<Integer> listB) {

// **TODO** Auto-generated method stub

Set<Integer> union = **new** HashSet<Integer>();

union.addAll(listA);

union.addAll(listB);

**return** union;

}

}

31.)how to combined two List object in java?

Using addAll()

32.)How to delete the particular element from array?use linear search to delete an element and display all elements after deletion.

**package** interviewQ;

**public** **class** DeleteElementsFromArray {

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

// **TODO** Auto-generated method stub

**int** a[] = **new** **int**[]{12,2,9,23,2};

**int** del =9;

**int** count=0;

**int** pos=0;

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

**if**(a[i]==del){

pos = i;

**for**(**int** j=i;j<a.length-1;j++){

a[j]=a[j+1];

}

count++;

**break**;

}

}

**if**(count!=0){

System.***out***.println("Element is found at position "+pos+"and deleted from an array");

}

**else**{

System.***out***.println("Element is not found in array");

}

System.***out***.println("After deletion of an array from positioned");

**for**(**int** i=0;i<a.length-1;i++){

System.***out***.print(a[i]+"\t");

}

}

}

**package** interviewQ;

**import** java.util.Arrays;

**import** java.util.HashSet;

**import** java.util.List;

**import** java.util.Set;

**public** **class** RemoveDuplicateElementFromArrayUsingCollection {

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

// **TODO** Auto-generated method stub

Integer array[] = {10,20,10,30,50,60,30};

Set setArray = **new** HashSet();

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

setArray.add(array[i]);

}

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

}

}

33.)merge two array in java?

**package** interviewQ;

**public** **class** MergeArray {

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

// **TODO** Auto-generated method stub

**int** a1[] = **new** **int**[]{1,2,1,4};

**int** a2[] = **new** **int**[]{2,3,4};

**int** a3[] = **new** **int**[a1.length+a2.length];

**for**(**int** i=0;i<a1.length;i++){

a3[i] = a1[i];

}

**int** index=a1.length;

**for**(**int** i=0;i<a2.length;i++){

a3[index++] = a2[i];

}

**for**(**int** i=0;i<a3.length;i++){

System.***out***.print(a3[i]+"\t");

}

}

}

34.)how to print an array of element without loop in one statement?

Ans:-Array.toString(arrayObject);

35.)Remove duplicate number from array in java

Simple program

**package** interviewQ;

**public** **class** RemoveDuplicateElementFromArray {

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

// **TODO** Auto-generated method stub

**int** arr[] = {10,10,20,20,30,30,50};

**int** len = arr.length;

**int** j=0;

**for**(**int** i=0;i<len-1;i++){

**if**(arr[i]!=arr[i+1]){

arr[j++] = arr[i];

}

}

arr[j++]=arr[len-1];

**for**(**int** k=0;k<j;k++){

System.***out***.println(arr[k]);

}

}

}

Program with return

**package** interviewQ;

**public** **class** RemoveDuplicateInArray{

**public** **static** **int** removeDuplicateElements(**int** arr[], **int** n){

**if** (n==0 || n==1){

**return** n;

}

**int**[] temp = **new** **int**[n];

**int** j = 0;

**for** (**int** i=0; i<n-1; i++){

**if** (arr[i] != arr[i+1]){

temp[j++] = arr[i];

}

}

temp[j++] = arr[n-1];

// Changing original array

**for** (**int** i=0; i<j; i++){

arr[i] = temp[i];

}

**return** j;

}

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

**int** arr[] = {10,20,20,30,30,40,50,50};

**int** length = arr.length;

length = *removeDuplicateElements*(arr, length);

//printing array elements

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

System.***out***.print(arr[i]+" ");

}

}

Using collection

**package interviewQ;**

**import java.util.Arrays;**

**import java.util.Collections;**

**import java.util.HashSet;**

**import java.util.List;**

**import java.util.Set;**

**public class RemoveDuplicateElementFromArrayUsingCollection {**

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

**// TODO Auto-generated method stub**

**Integer array[] = {10,20,10,30,50,60,30};**

**Set setArray = Collections.EMPTY\_SET;**

**setArray = new HashSet();**

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

**setArray.add(array[i]);**

**}**

**//setArray = Collections.EMPTY\_SET;**

**assert(setArray!=null):"set should not be null";**

**String s =null;**

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

**}**

**}**36.)How to avoid null pointer exception?

Ans:-first use literal object then other like below

**if**("vicky".equals(s)){

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

}

Or

String Vicky=”Vicky”;

**if**(vicky.equals(s)){

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

}

37.)Sort using one for loop and what is complexity of it.

**package** interviewQ;

**public** **class** SingleLoopArraySort {

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

**int**[] arr = { 5,6,1,3 };

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

System.***out***.println("arr[i] < arr[i - 1]"+" "+arr[i]+" "+ arr[i - 1]);

**if** (arr[i] < arr[i - 1])

{

arr[i] = arr[i] + arr[i - 1];

arr[i - 1] = arr[i] - arr[i - 1];

arr[i] = arr[i] - arr[i - 1];

i = 0;

}

}

System.***out***.println("sorted Array :");

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

System.***out***.print(arr[i] + " ");

}

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

}

}

Inside the for loop it is working on sorting mechanism like given below

e.g.

a=6;

b=2;

a =a+b

b=a-b

a=a-b

Output

arr[i] < arr[i - 1] 6 5

arr[i] < arr[i - 1] 1 6

arr[i] < arr[i - 1] 1 5

arr[i] < arr[i - 1] 5 1

arr[i] < arr[i - 1] 6 5

arr[i] < arr[i - 1] 3 6

arr[i] < arr[i - 1] 5 1

arr[i] < arr[i - 1] 3 5

arr[i] < arr[i - 1] 3 1

arr[i] < arr[i - 1] 5 3

arr[i] < arr[i - 1] 6 5

sorted Array :

1 3 5 6

After understanding

**package** interviewQ;

**public** **class** SingleLoopToSortArray {

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

// **TODO** Auto-generated method stub

**int** a[]= **new** **int**[]{2,3,45,23,12,8,1};

**int** len = a.length;

**for**(**int** i=1;i<len;i++){

**if**(a[i]<a[i-1]){

a[i] = a[i]+a[i-1];

a[i-1] = a[i]-a[i-1];

a[i] = a[i]-a[i-1];

i=0;

}

}

**for**(**int** i=0;i<len;i++){

System.***out***.print(a[i]+"\t");

}

}

}

38.)Method name to display the common elements

retainAll()

**package** interviewQ;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** RetainAll {

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

// **TODO** Auto-generated method stub

List<Integer> l1 = **new** ArrayList<Integer>();

l1.add(1);

l1.add(2);

l1.add(3);

List<Integer> l2= **new** ArrayList<Integer>();

l2.add(4);

l2.add(2);

l2.add(3);

System.***out***.println("l1 == "+l1);

System.***out***.println("l2 == "+l2);

List<Integer> l3 = **new** ArrayList<Integer>(l2);

l3.retainAll(l1);

System.***out***.println("l3 == "+l3);

}

}

ContainsAll() will check the all elements present in left side collection elements .if its found will display true otherwise false.

**package** interviewQ;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** RetainAll {

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

// **TODO** Auto-generated method stub

List<Integer> l1 = **new** ArrayList<Integer>();

l1.add(1);

l1.add(2);

l1.add(3);

List<Integer> l2= **new** ArrayList<Integer>();

l2.add(4);

l2.add(2);

l2.add(3);

System.***out***.println("l1 == "+l1);

System.***out***.println("l2 == "+l2);

List<Integer> l3 = **new** ArrayList<Integer>(l2);

l3.retainAll(l1);

l3.add(5);

System.***out***.println("l3 == "+l3);

System.***out***.println(l1.containsAll(l3));

}

}

39.)Write an example of Generic.

40.)what is variable array argument or Vararg in main method?

class Test1

{

    // A method that takes variable number of intger

    // arguments.

    static void fun(int ...a)

    {

        System.out.println("Number of arguments: " + a.length);

        // using for each loop to display contents of a

        for (int i: a)

            System.out.print(i + " ");

        System.out.println();

    }

    // Driver code

    public static void main(String args[])

    {

        // Calling the varargs method with different number

        // of parameters

        fun(100);         // one parameter

        fun(1, 2, 3, 4);  // four parameters

        fun();            // no parameter

    }

}

class Test2

{

    // Takes string as a argument followed by varargs

    static void fun2(String str, int ...a)

    {

        System.out.println("String: " + str);

        System.out.println("Number of arguments is: "+ a.length);

        // using for each loop to display contents of a

        for (int i: a)

            System.out.print(i + " ");

        System.out.println();

    }

    public static void main(String args[])

    {

        // Calling fun2() with different parameter

        fun2("GeeksforGeeks", 100, 200);

        fun2("CSPortal", 1, 2, 3, 4, 5);

        fun2("forGeeks");

    }

}

**Important points:**

* + Vararg Methods can also be overloaded but overloading may lead to ambiguity.
  + Prior to JDK 5, variable length arguments could be handled into two ways : One was using overloading, other was using array argument.
  + There can be only one variable argument in a method.
  + Variable argument (varargs) must be the last argument.

**Erroneous varargs Examples**

* + Specifying two varargs in a single method:
  + void method(String... gfg, int... q)
  + {
  + // Compile time error as there are two
  + // varargs
  + }
  + Specifying varargs as the first parameter of method instead of last one:
  + void method(int... gfg, String q)
  + {
  + // Compile time error as vararg appear
  + // before normal argument

}

41.)Add all number given in number.

**package** interviewQ;

**public** **class** CountNumber {

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

// **TODO** Auto-generated method stub

**int** num = 123;

**int** sum = 0;

**int** count = 0;

**int** temp = 0;

**while** (num > 0) {

temp = num % 10;

sum = sum + temp;

num = num / 10;

count++;

}

System.***out***.println("sum of the number=" + sum + " "

+ "total number present in number" + " " + count);

}

}

42.)Prime number between range

**package** interviewQ;

**import** java.util.Scanner;

**public** **class** PrimeFromRange {

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

// initialize and declare here.

**int** s1, s2, count = 0, i, j;

Scanner s = **new** Scanner(System.***in***);

System.***out***.print("Enter the lower limit : ");

s1 = s.nextInt();

System.***out***.print("Enter the upper limit :");

s2 = s.nextInt();

System.***out***.println("Prime numbers between given range are :");

**for** (i = s1; i <= s2; i++) {

**if**(*prime*(i)){

System.***out***.print(i+"\t");

}

}

}

**private** **static** **boolean** prime(**int** i) {

// **TODO** Auto-generated method stub

**int** count=0;

**for**( **int** j = 2; j < i; j++)

{

**if**(i % j == 0)

{

count = 0;

**break**;

}

**else**

{

count = 1;

}

}

**if**(count == 1)

{

**return** **true**;

}

**else**

**return** **false**;

}

}

43.)Write a qsl query to display 2nd highest paid salary

SELECT MAX(sal) FROM `employee`

WHERE sal NOT IN (SELECT MAX(sal) FROM employee);

For nth paid salary

SELECT \* /\*This is the outer query part \*/

FROM Employee Emp1

WHERE (n) = ( /\* Subquery starts here \*/

SELECT COUNT(DISTINCT(Emp2.Sal))

FROM Employee Emp2

WHERE Emp2.Sal > Emp1.Sal)

N can be replaced by number.

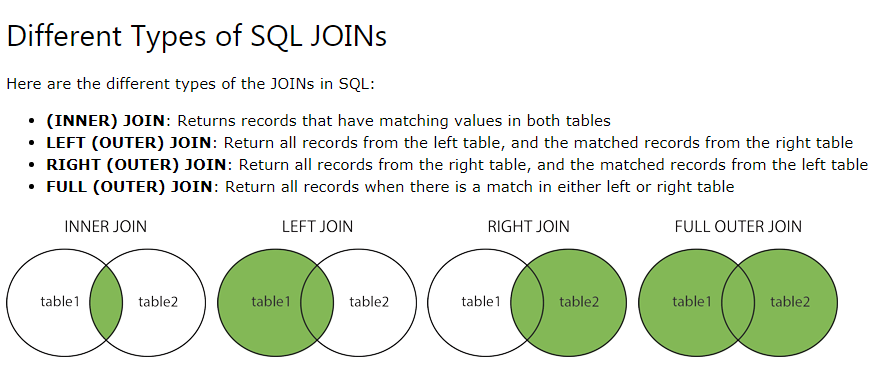
44.)write a quey to find out the odd position records.

45.)write a query to find out the even position records.

46.)write a query for group by clause

47.)write a query to find out the senior employee in company with age.

48.)write a query for different join



Inner Join

49.)write a query for self and

References:

<https://simpleprogrammer.com/programming-interview-questions/>

<https://www.javacodegeeks.com/2012/06/avoid-null-pointer-exception-in-java.html>