JAVA Programs :

1. Write a java program to find common elements between two arrays?
2. How to swap two numbers without using temporary variable in Java?
3. Write a java program to find out duplicate characters in a string?
4. Write a java program to find palindrome number?
5. Write a java program to print Fibonacci series?
6. Write a java program to find the given number is Armstrong number or not?
7. Write a java program to reverse a string?
8. Write a java program to count the number of word in a given string?
9. Write a java program to find a missing number in an array?
10. Write a java program to remove repetitive character?
11. Write a java program to implement ArrayList?
12. Write a java program to Search word in a sentence?
13. Write a java program to compare two strings using compareTo() method?
14. Write a java program for Bubble Sort?
15. Write a java program to create deadlock between two threads?
16. Write a java program to find top two maximum numbers in a array?
17. Write a java program to Find longest substring without repeating characters?
18. Write a java program to find first occurrence of integer with adjacent difference 1 in an integer array?
19. Write a java program to check the given number is a prime number or not?
20. Write a java program for singleton class?

Data Structure programs in java.

================================

1. Write a java program to Implement Stack?
2. Write a java program to Implement Queue?

=================================

1. Write a java program to move all the mail subjects to an Excel sheet using selenium for a specific date?
2. Write a java program to for given a scenario which includes DB connection also?
3. what is testng class?
4. how to read and write data from excel?
5. database connectivity with java?
6. what is functional and non functional?
7. what is second highest salary?
8. what is joins?
9. what is locators ans which locators u want to use?
10. What is the prime number ?
11. What is  fibocanai series?
12. which is better implicit wait or explicit wait?
13. **Write a java program to find common elements between two arrays?**

public class HelloWorld {

public static void main(String a[]) {

int[] arr1 = {4,7,3,9,2};

int[] arr2 = {3,2,12,9,40,32,4};

for(int i = 0;i<arr1.length;i++) {

for(int j = 0;j<arr2.length;j++) {

if(arr1[i]==arr2[j]) {

System.out.println(arr1[i]);

}

}

}

}

}

4

3

9

1. **How to swap two numbers without using temporary variable in Java?**

// Program to swap two numbers  without

// using temporary variable

import java.\*;

class Geeks {

    public static void main(String a[])

    {

        int x = 10;

        int y = 5;

        x = x + y;

        y = x - y;

        x = x - y;

        System.out.println("After swaping:"

             + " x = " + x + ", y = " + y);

    }

}

After Swapping: x = 5, y = 10

1. **Write a java program to find out duplicate characters in a string?**
2. import java.util.HashMap;
3. import java.util.Set;
4. class DuplicateCharactersInString
5. {
6. static void **duplicateCharCount**(**String** inputString)
7. {
8. //Creating a HashMap containing char as key and it's occurrences as value
9. HashMap<Character, Integer> charCountMap = **new** HashMap<Character, Integer>();
10. //Converting given string to char array
11. **char**[] strArray = inputString.toCharArray();
12. //checking each char of strArray
13. **for** (**char** c : strArray)
14. {
15. **if**(charCountMap.containsKey(c))
16. {
17. //If char is present in charCountMap, incrementing it's count by 1
18. charCountMap.put(c, charCountMap.get(c)+1);
19. }
20. **else**
21. {
22. //If char is not present in charCountMap,
23. //putting this char to charCountMap with 1 as it's value
24. charCountMap.put(c, 1);
25. }
26. }
27. //Getting a Set containing all keys of charCountMap
28. Set<Character> charsInString = charCountMap.keySet();
29. System.out.println("Duplicate Characters In "+inputString);
30. //Iterating through Set 'charsInString'
31. **for** (Character ch : charsInString)
32. {
33. **if**(charCountMap.get(ch) > 1)
34. {
35. //If any char has a count of more than 1, printing it's count
36. System.out.println(ch +" : "+ charCountMap.get(ch));
37. }
38. }
39. }
40. public static void **main**(**String**[] args)
41. {
42. **duplicateCharCount**("JavaJ2EE");
43. **duplicateCharCount**("Fresh Fish");
44. **duplicateCharCount**("Better Butter");
45. }
46. }

**OUTPUT :**

Duplicate Characters In JavaJ2EE  
E : 2  
a : 2  
J : 2  
Duplicate Characters In Fresh Fish  
F : 2  
s : 2  
h : 2  
Duplicate Characters In Better Butter  
t : 4  
e : 3  
r : 2  
B : 2

----------------------------------------------------------------

public class DuplStr {

public static void main(String argu[]) {

String str = "w3schools";

char[] inp = str.toCharArray();

System.out.println("Duplicate Characters are:");

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

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

if (inp[i] == inp[j]) {

System.out.println(inp[j]);

break;

}

}

}

}

}

Duplicate Characters are: s o

1. **Write a java program to find palindrome number?**

A **palindrome number** is a number that is same after reverse. For example 545, 151, 34543, 343, 171, 48984 are the palindrome numbers. It can also be a string like LOL, MADAM etc.

import java.util.Scanner;

public class Example24 {

public static void main(String args[])

{

Scanner in = new Scanner(System.in);

System.out.print("Input a number: ");

int n = in.nextInt();

int sum = 0, r;

int temp = n;

while(n>0)

{

r = n % 10;

sum = (sum\*10)+r;

n = n/10;

}

if(temp==sum)

System.out.println("It is a Palindrome number.");

else

System.out.println("Not a palindrome");

}

}

Input a number: 5

It is a Palindrome number.

1. **Write a java program to print Fibonacci series?**

In fibonacci series, next number is the sum of previous two numbers for example 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 etc. The first two numbers of fibonacci series are 0 and 1.

**class** FibonacciExample1{

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

{

**int** n1=0,n2=1,n3,i,count=10;

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

**for**(i=2;i<count;++i)//loop starts from 2 because 0 and 1 are already printed

{

n3=n1+n2;

System.out.print(" "+n3);

n1=n2;

n2=n3;

}

}}

1. **Write a java program to find the given number is Armstrong number or not?**

A positive number is called **Armstrong number** if it is equal to the sum of cubes of its digits for example 0, 1, 153, 370, 371, 407 etc.

Let's try to understand why **153** is an Armstrong number.

153 = (1\*1\*1)+(5\*5\*5)+(3\*3\*3)

where:

(1\*1\*1)=1

(5\*5\*5)=125

(3\*3\*3)=27

So:  1+125+27=153

public class JavaExample {

public static void main(String[] args) {

int num = 370, number, temp, total = 0;

number = num;

while (number != 0)

{

temp = number % 10;

total = total + temp\*temp\*temp;

number /= 10;

}

if(total == num)

System.out.println(num + " is an Armstrong number");

else

System.out.println(num + " is not an Armstrong number");

}

}

Output:

370 is an Armstrong number

1. **Write a java program to reverse a string?**

**Following are some interesting facts about String and StringBuffer classes :**  
1. Objects of String are immutable.  
2. String class in Java does not have reverse() method, however StringBuilder class has built in reverse() method.  
3. StringBuilder class do not have toCharArray() method, while String class does have toCharArray() method.

**1st Way : Using built in reverse() method of the StringBuilder class:** String class does not have reverse() method, we need to convert the input string to StringBuilder, which is achieved by using the append method of StringBuilder. After that, print out the characters of the reversed string by scanning from the first till the last index.

// Java program to ReverseString using StringBuilder

import java.lang.\*;

import java.io.\*;

import java.util.\*;

// Class of ReverseString

class ReverseString

{

    public static void main(String[] args)

    {

        String input = "Geeks for Geeks";

        StringBuilder input1 = new StringBuilder();

        // append a string into StringBuilder input1

        input1.append(input);

        // reverse StringBuilder input1

        input1 = input1.reverse();

        // print reversed String

        System.out.println(input1);

    }

}

skeeG rof skeeG

**2nd Way :** **Converting String to character array:** The user input the string to be reversed.  
**Method:**

1. First, convert String to character array

by using the built in Java String class

method toCharArray().

2. Then, scan the string from end to start,

and print the character one by one.

// Java program to Reverse a String  by

// converting string to characters  one

// by one

import java.lang.\*;

import java.io.\*;

import java.util.\*;

// Class of ReverseString

class ReverseString

{

    public static void main(String[] args)

    {

        String input = "GeeksForGeeks";

        // convert String to character array

        // by using toCharArray

        char[] try1 = input.toCharArray();

        for (int i = try1.length-1; i>=0; i--)

            System.out.print(try1[i]);

    }

}

skeeGrofskeeG

**3rd Way :** **Using ArrayList object:**Convert the input string into the character array by using toCharArray() built in method. Then, add the characters of the array into the ArrayList object. Java also has built in reverse() method for the Collections class. Since Collections class reverse() method takes a list object , to reverse the list , we will pass the LinkedList object which is a type of list of characters.

1. We copy String contents to an object

of ArrayList.

1. We create a ListIterator object by using

the listIterator() method on the LinkedList

object.

2. ListIterator object is used to iterate over

the list.

3. ListIterator object helps us to iterate

over the reversed list and print it one

by one to the output screen.

// Java program to Reverse a String using ListIterator

import java.lang.\*;

import java.io.\*;

import java.util.\*;

// Class of ReverseString

class ReverseString

{

    public static void main(String[] args)

    {

        String input = "Geeks For Geeks";

        char[] hello = input.toCharArray();

        List<Character> trial1 = new ArrayList<>();

        for (char c: hello)

            trial1.add(c);

        Collections.reverse(trial1);

        ListIterator li = trial1.listIterator();

        while (li.hasNext())

            System.out.print(li.next());

    }

}

skeeG roF skeeG

**Reverse a String without changing Special character position?**

class GFG

{

    public static void reverse(char str[])

    {

        // Initialize left and right pointers

        int r = str.length - 1, l = 0;

        // Traverse string from both ends until

        // 'l' and 'r'

        while (l < r)

        {

            // Ignore special characters

            if (!Character.isAlphabetic(str[l]))

                l++;

            else if(!Character.isAlphabetic(str[r]))

                r--;

            // Both str[l] and str[r] are not spacial

            else

            {

                char tmp = str[l];

                str[l] = str[r];

                str[r] = tmp;

                l++;

                r--;

            }

        }

    }

    // Driver Code

    public static void main(String[] args)

    {

        String str = "a!!!b.c.d,e'f,ghi";

        char[] charArray = str.toCharArray();

        System.out.println("Input string: " + str);

                            reverse(charArray);

        String revStr = new String(charArray);

        System.out.println("Output string: " + revStr);

    }

}

1. **Write a java program to count the number of word in a given string?**

***Below program is without using String.split() method.***

class StringDemo

{

public static void main(String...s)

{

  int word=1;

  String str="count number of words and sapces";

  for(int i=0;i<str.length();++i)

  {

   if(str.charAt(i)==' ')

    word++;

  }

  System.out.println("Number of words="+word);

  System.out.println("Number of spaces="+(word-1));

}

}

Output :

Number of words =6

Number of spaces=5

***Below program is without using String.split() method.***

In this solution, we will use the split() method of java.lang.String class to count the number of words in a given sentence. This solution uses the regular expression "\\s+" to split the String on whitespace. The split method returns an array, the length of array is your number of words in given String.

If you are new to regular expression in Java, the \s is a character class to detect space including tabs, since \ needs to be escaped in Java, it becomes \\s and because there could be multiple spaces between words we made this regular expression greedy by adding +, hence **\\s+** will find one more space and split the String accordingly.

**public** **class** PracticeProgram {

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

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

String str = "Count the number of words";

**if**(str == **null** || str.isEmpty())

{

**return**;

}

String[] words = str.split("\\s+");

System.***out***.println("number of words are : " +words.length);

}

}

Output :

number of words are : 5

1. **Write a java program to find a missing number in an array?**

1. Get the sum of numbers

total = n\*(n+1)/2

2 Subtract all the numbers from sum and

you will get the missing number.

import java.util.\*;

public class Exercise24 {

public static void main(String[] args) {

int total\_num;

int[] numbers = new int[]{1,2,3,4,6,7};

total\_num = 7;

int expected\_num\_sum = total\_num \* ((total\_num + 1) / 2);

int num\_sum = 0;

for (int i: numbers) {

num\_sum += i;

}

System.out.print( expected\_num\_sum - num\_sum);

System.out.print("\n");

}

}

Sample Data: 1,2,3,4,6,7

Sample Output:

5

1. **Write a java program to remove repetitive character?**

Clear the doubt :

// Java prigram to remove duplicates

import java.util.\*;

class RemoveDuplicates

{

    /\* Function removes duplicate characters from the string

    This function work in-place \*/

    void removeDuplicates(String str)

    {

        LinkedHashSet<Character> lhs = new LinkedHashSet<>();

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

            lhs.add(str.charAt(i));

        // print string after deleting duplicate elements

        for(Character ch : lhs)

            System.out.print(ch);

    }

    /\* Driver program to test removeDuplicates \*/

    public static void main(String args[])

    {

        String str = "geeksforgeeks";

        RemoveDuplicates r = new RemoveDuplicates();

        r.removeDuplicates(str);

    }

}

geksfor

**without using Set. By using indexOf() method**

**public** **class** Demo {

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

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

String s = "GREEKSFORGREEKS";

*removeDuplicates*(s);

}

**public** **static** **void** removeDuplicates(String s)

{

String str = **new** String();

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

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

{

**char** c =s.charAt(i);

**if**(str.indexOf(c)<0)

{

str = str+c;

}

}

System.***out***.print(str);

}

}

1. **Write a java program to implement ArrayList?**

class ArrayListDemo

{

public static void main(String[] args)

{

ArrayList<String> list = new ArrayList<String>();

list.add("ONE"); //Adds "ONE" at the end of the list

list.add("TWO"); //Adds "TWO" at the end of the list

list.add("THREE"); //Adds "THREE" at the end of the list

list.add("FOUR"); //Adds "FOUR" at the end of the list

System.out.println(list); //Output : [ONE, TWO, THREE, FOUR]

list.add(3, "INSERTED"); //Inserts "INSERTED" at position 3

System.out.println(list); //Output : [ONE, TWO, THREE, INSERTED, FOUR]

list.add(1, "INSERTED"); //Inserts "INSERTED" at position 1

System.out.println(list); //Output : [ONE, INSERTED, TWO, THREE, INSERTED, FOUR]

list.remove("INSERTED"); //Removes first occurence of "INSERTED"

System.out.println(list); //Output : [ONE, TWO, THREE, INSERTED, FOUR]

list.remove(3); //Removes an element at position 3

System.out.println(list); //Output : [ONE, TWO, THREE, FOUR]

list.set(3, "REPLACED"); //Replaces an element at position 3 with "REPLACED"

System.out.println(list); //Output : [ONE, TWO, THREE, REPLACED]

}

}

1. **Write a java program to Search word in a sentence?**

This example shows how we can search a word within a String object using indexOf() method which returns a position index of a word within the string if found. Otherwise it returns -1.

public class SearchStringEmp{

public static void main(String[] args) {

String strOrig = "Hello readers";

int intIndex = strOrig.indexOf("Hello");

if(intIndex == - 1) {

System.out.println("Hello not found");

} else {

System.out.println("Found Hello at index " + intIndex);

}

}

}

Result

The above code sample will produce the following result.

Found Hello at index 0

This example shows how we can search a word within a String object

public class HelloWorld {

public static void main(String[] args) {

String text = "The cat is on the table";

System.out.print(text.contains("the"));

}

}

## Result

The above code sample will produce the following result.

true

1. **Write a java program to compare two strings using compareTo() method?**

Here we have three Strings and we are comparing them with each other using compareTo() method.

public class CompareToExample {

public static void main(String args[]) {

String str1 = "String method tutorial";

String str2 = "compareTo method example";

String str3 = "String method tutorial";

int var1 = str1.compareTo( str2 );

System.out.println("str1 & str2 comparison: "+var1);

int var2 = str1.compareTo( str3 );

System.out.println("str1 & str3 comparison: "+var2);

int var3 = str2.compareTo("compareTo method example");

System.out.println("str2 & string argument comparison: "+var3);

}

}

Output:

str1 & str2 comparison: -16

str1 & str3 comparison: 0

str2 & string argument comparison: 0

1. **Write a java program for Bubble Sort?**

import java.util.Arrays;

class BubbleSort

{

void bubbleSort(int nums[])

{

int n = nums.length;

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

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

if (nums[j] > nums[j+1])

{

// swap temp and nums[i]

int temp = nums[j];

nums[j] = nums[j+1];

nums[j+1] = temp;

}

}

// Method to test above

public static void main(String args[])

{

BubbleSort ob = new BubbleSort();

int nums[] = {7, -5, 3, 2, 1, 0, 45};

System.out.println("Original Array:");

System.out.println(Arrays.toString(nums));

ob.bubbleSort(nums);

System.out.println("Sorted Array");

System.out.println(Arrays.toString(nums));

}

}

Copy

Sample Output:

Original Array:

[7, -5, 3, 2, 1, 0, 45]

Sorted Array

[-5, 0, 1, 2, 3, 7, 45]

1. **Write a java program to create deadlock between two threads?**

Deadlock describes a situation where two or more threads are blocked forever, waiting for each other. Deadlocks can occur in Java when the synchronized keyword causes the executing thread to block while waiting to get the lock, associated with the specified object. Since the thread might already hold locks associated with other objects, two threads could each be waiting for the other to release a lock. In such case, they will end up waiting forever.

package com.java2novice.algos;

public class MyDeadlock {

    String str1 = "Java";

    String str2 = "UNIX";

    Thread trd1 = new Thread("My Thread 1"){

        public void run(){

            while(true){

                synchronized(str1){

                    synchronized(str2){

                        System.out.println(str1 + str2);

                    }

                }

            }

        }

    };

    Thread trd2 = new Thread("My Thread 2"){

        public void run(){

            while(true){

                synchronized(str2){

                    synchronized(str1){

                        System.out.println(str2 + str1);

                    }

                }

            }

        }

    };

    public static void main(String a[]){

        MyDeadlock mdl = new MyDeadlock();

        mdl.trd1.start();

        mdl.trd2.start();

    }

}

1. **Write a java program to find top two maximum numbers in a array?**
2. package arraysInterviewPrograms.instanceofjava;
3. public class FindTopTwo {
5. public void findTwoMaxNumbers(int[] array){
7. int maxOne = 0;
8. int maxTwo = 0;
10. for(int i:array){
12. if(maxOne < i){
13. maxTwo = maxOne;
14. maxOne =i;
15. } else if(maxTwo < i){
16. maxTwo = i;
17. }
18. }

21. System.out.println("First Maximum Number: "+maxOne);
22. System.out.println("Second Maximum Number: "+maxTwo);
23. }
25. public static void main(String a[]){
27. int num[] = {4,23,67,1,76,1,98,13};
28. FindTopTwo obj = new FindTopTwo();
29. obj.findTwoMaxNumbers(num);
30. obj.findTwoMaxNumbers(new int[]{4,5,6,90,1});
32. }
34. }
35. **Write a java program to Find longest substring without repeating characters?**
36. import java.util.LinkedHashMap;
37. public class MainClass
38. {
39. static void **longestSubstring**(**String** inputString)
40. {
41. //Convert inputString to charArray
42. **char**[] charArray = inputString.toCharArray();
43. //Initialization
44. **String** longestSubstring = null;
45. **int** longestSubstringLength = 0;
46. //Creating LinkedHashMap with characters as keys and their position as values.
47. LinkedHashMap<Character, Integer> charPosMap = **new** LinkedHashMap<Character, Integer>();
48. //Iterating through charArray
49. **for** (**int** i = 0; i < charArray.length; i++)
50. {
51. **char** ch = charArray[i];
52. //If ch is not present in charPosMap, adding ch into charPosMap along with its position
53. **if**(!charPosMap.containsKey(ch))
54. {
55. charPosMap.put(ch, i);
56. }
57. //If ch is already present in charPosMap, reposioning the cursor i to the position of ch and clearing the charPosMap
58. **else**
59. {
60. i = charPosMap.get(ch);
61. charPosMap.clear();
62. }
63. //Updating longestSubstring and longestSubstringLength
64. **if**(charPosMap.size() > longestSubstringLength)
65. {
66. longestSubstringLength = charPosMap.size();
67. longestSubstring = charPosMap.keySet().toString();
68. }
69. }
70. System.out.println("Input String : "+inputString);
71. System.out.println("The longest substring : "+longestSubstring);
72. System.out.println("The longest Substring Length : "+longestSubstringLength);
73. }
74. public static void **main**(**String**[] args)
75. {
76. **longestSubstring**("javaconceptoftheday");
77. System.out.println("==========================");
78. **longestSubstring**("thelongestsubstring");
79. }
80. }

**Output :**

Input String : javaconceptoftheday  
The longest substring : [o, f, t, h, e, d, a, y]  
The longest Substring Length : 8  
==========================  
Input String : thelongestsubstring  
The longest substring : [u, b, s, t, r, i, n, g]  
The longest Substring Length : 8

1. **Write a java program to check the given number is a prime number or not?**
2. **public** **class** PrimeExample{
3. **public** **static** **void** main(String args[]){
4. **int** i,m=0,flag=0;
5. **int** n=3;//it is the number to be checked
6. m=n/2;
7. **if**(n==0||n==1){
8. System.out.println(n+" is not prime number");
9. }**else**{
10. **for**(i=2;i<=m;i++){
11. **if**(n%i==0){
12. System.out.println(n+" is not prime number");
13. flag=1;
14. **break**;
15. }
16. }
17. **if**(flag==0)  { System.out.println(n+" is prime number"); }
18. }//end of else
19. }
20. }