Java Programs

Q. Write a function to find out duplicate words in a given string?

Approach

1. Define a string.
2. Convert the string into lowercase to make the comparison insensitive.
3. Split the string into words.
4. Two loops will be used to find duplicate words. Outer loop will select a word and Initialize variable count to 1. Inner loop will compare the word selected by outer loop with rest of the words.
5. If a match found, then increment the count by 1 and set the duplicates of word to '0' to avoid counting it again.
6. After the inner loop, if count of a word is greater than 1 which signifies that the word has duplicates in the string.

```
public class DuplicateWord {

    public static void main(String[] args) {

        String string = "Big black bug bit a big black dog on his big black nose";
        int count;

        //Converts the string into lowercase
        string = string.toLowerCase();

        //Split the string into words using built-in function
        String words[] = string.split(" ");

        System.out.println("Duplicate words in a given string : ");
        for(int i = 0; i < words.length; i++) {
            count = 1;
            for(int j = i+1; j < words.length; j++) {
                if(words[i].equals(words[j])) {
                    count++;
                    //Set words[j] to 0 to avoid printing visited word
                    words[j] = "0";
                }
            }

            //Displays the duplicate word if count is greater than 1
            if(count > 1 && words[i] != "0")
                System.out.println(words[i]);
        }
    }
}
```

Output

```
Duplicate words in a given string :
big
black
```

Q. Find the missing number in an array?

Approach

1. Calculate `A = n*(n+1)/2` where n is largest number in series 1…N.
2. Calculate B = Sum of all numbers in given series
3. Missing number = A – B

```
// Java program to find missing Number

public class Test {

 public static void main(String[] args) {

  int total;
  int[] numbers = new int[]{1, 2, 3, 4, 6, 7};
  total = 7;
  int expected_sum = total * ((total + 1) / 2);
```

```
   int num_sum = 0;

   for (int i: numbers) {
    num_sum += i;
   }

   System.out.print( expected_sum - num_sum );
  }
}
```

Output

```
5
```

↥ back to top

Q. Write a program to generate random numbers between the given range?

ToDo

Q. Write a java program to swap two string variables without using temp variable?

Approach

1. Append second string to first string and store in first string: a = a + b
2. call the method substring(int beginindex, int endindex) by passing beginindex as 0 and endindex as, a.length() - b.length(): b = substring(0,a.length()-b.length());
3. call the method substring(int beginindex) by passing b.length() as argument to store the value of initial b string in a a = substring(b.length());

```
/**
* Java program to swap two strings without using a temporary
* variable.
**/
import java.util.*;

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

   // Declare two strings
   String a = "Hello";
   String b = "World";

   // Print String before swapping
   System.out.println("Strings before swap: a = " + a + " and b = "+b);

   // append 2nd string to 1st
   a = a + b;

   // store intial string a in string b
   b = a.substring(0, a.length() - b.length());

   // store initial string b in string a
   a = a.substring(b.length());

   // print String after swapping
   System.out.println("Strings after swap: a = " + a + " and b = " + b);
  }
}
```

Output

```
Strings before swap: a = Hello and b = World
Strings after swap: a = World and b = Hello
```

Q. Write a java program to Move all zeroes to end of array?

```
Input:  arr[] = {1, 2, 0, 4, 3, 0, 5, 0};
Output: arr[] = {1, 2, 4, 3, 5, 0, 0, 0};
```

```
public class Test
{
    static void pushZerosToEnd(int arr[], int n) {

        int count = 0;  // Count of non-zero elements

        // Traverse the array. If element encountered is
        // non-zero, then replace the element at index 'count'
        // with this element
        for (int i = 0; i < n; i++)
            if (arr[i] != 0)
                arr[count++] = arr[i];

        // Now all non-zero elements have been shifted to
        // front and 'count' is set as index of first 0.
        // Make all elements 0 from count to end.
        while (count < n)
            arr[count++] = 0;
    }


    public static void main (String[] args) {

        int arr[] = {1, 9, 8, 4, 0, 0, 2, 7, 0, 6, 0, 9};
        int n = arr.length;
        pushZerosToEnd(arr, n);
        System.out.println("Array after pushing zeros to the back: ");
        for (int i=0; i
```

Output

```
Array after pushing all zeros to end of array:
1 9 8 4 2 7 6 9 0 0 0 0
```

↥ back to top

Q. Write a multi-threading program to print odd number using one thread and even number using other?

```
class TaskEvenOdd implements Runnable {

    private int max;
    private Printer print;
    private boolean isEvenNumber;

    TaskEvenOdd(Printer print, int max, boolean isEvenNumber) {
        this.print = print;
        this.max = max;
        this.isEvenNumber = isEvenNumber;
    }

    @Override
    public void run() {

        int number = isEvenNumber == true ? 2 : 1;
        while (number <= max) {

            if (isEvenNumber) {
                //System.out.println("Thread Even: "+ Thread.currentThread().getName());
                print.printEven(number);
            } else {
                //System.out.println("Thread Odd: "+ Thread.currentThread().getName());
                print.printOdd(number);
            }
            number += 2;
        }
    }
}

class Printer {

    boolean isOdd = false;

    synchronized void printEven(int number) {

        while (isOdd == false) {
            try {
                wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println("Thread Even: " + number);
        isOdd = false;
        notifyAll();
    }

    synchronized void printOdd(int number) {
        while (isOdd == true) {
            try {
                wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        System.out.println("Thread Odd: " + number);
        isOdd = true;
        notifyAll();
    }
}

public class Test
{
 static int MAX = 5;
 public static void main(String... args) {
        Printer print = new Printer();
        Thread t1 = new Thread(new TaskEvenOdd(print, MAX, false));
        Thread t2 = new Thread(new TaskEvenOdd(print, MAX, true));
        t1.start();
        t2.start();
    }
}
```

Output

```
Thread Odd: 1
Thread Even: 2
Thread Odd: 3
Thread Even: 4
Thread Odd: 5
```

Q. How to print all permutations of a String in Java?

Approach

1. Define a string.
2. Fix a character and swap the rest of the characters.
3. Call the generatePermutation() for rest of the characters.
4. Backtrack and swap the characters again.

Recursive Approach

Recursive Approach

```
public class PermuteString
{
    // Function for swapping the characters
    public static String swapString(String a, int i, int j) {
        char[] b =a.toCharArray();
        char ch;
        ch = b[i];
        b[i] = b[j];
        b[j] = ch;
        return String.valueOf(b);
    }

    // Function for generating different permutations of the string
    public static void generatePermutation(String str, int start, int end) {

        //Prints the permutations
        if (start == end-1)
            System.out.println(str);
        else {

            for (int i = start; i < end; i++) {

                //Swapping the string by fixing a character
                str = swapString(str,start,i);
                //Recursively calling function generatePermutation() for rest of the characters
                generatePermutation(str,start+1,end);
                //Backtracking and swapping the characters again.
                str = swapString(str,start,i);
            }
        }
    }
    public static void main(String[] args) {

        String str = "ABC";
        int len = str.length();
        System.out.println("All the permutations of the string are: ");
        generatePermutation(str, 0, len);
    }
}
```

Output

```
All the permutations of the string are:

ABC
ACB
BAC
BCA
CBA
CAB
```

Q. Reverse the string with preserving the position of spaces

Approach

1. Create a string to store result. Mark the space position of the given string in this string.
2. Insert the character from input string into the result string in reverse order.
3. While inserting the character check if the result string already contains a space at index 'j' or not. If it contains, we copy the character to the next position.

```
// Java program to reverse a string
// preserving spaces.
public class ReverseStringPreserveSpace
{
 static void reverses(String str) {

   char[] inputArray = str.toCharArray();
   char[] result = new char[inputArray.length];
   // Mark spaces in result
   for (int i = 0; i < inputArray.length; i++) {
     if (inputArray[i] == ' ') {
      result[i] = ' ';
     }
   }
   // Traverse input string from beginning
   // and put characters in result from end
   int j = result.length - 1;
   for (int i = 0; i < inputArray.length; i++) {
     // Ignore spaces in input string
     if (inputArray[i] != ' ') {
       // ignore spaces in result.
       if (result[j] == ' ') {
        j--;
       }
       result[j] = inputArray[i];
       j--;
     }
   }
   System.out.println(String.valueOf(result));
 }
 public static void main(String[] args) {
  reverses("India Is my country");
 }
}
```

Output

```
India Is my country --> yrtnu oc ym sIaidnI
```

Q. How do you find longest substring without repeating characters in a string?

Approach

1. Start traversing the string from left to right and maintain track
2. Check the non-repeating characters in current substring with the help of a start and end index

```
public class Test
{
 public static String getUniqueCharacterSubstring(String input) {
     Map visited = new HashMap<>();
     String output = "";
     for (int start = 0, end = 0; end < input.length(); end++) {
         char currChar = input.charAt(end);
         if (visited.containsKey(currChar)) {
             start = Math.max(visited.get(currChar) + 1, start);
         }
         if (output.length() < end - start + 1) {
             output = input.substring(start, end + 1);
         }
         visited.put(currChar, end);
     }
     return output;
 }

 public static void main(String[] args) {

     String input = "LongestSubstringFindOut";
     System.out.println(getUniqueCharacterSubstring(input));
 }
}
```

Output

```
LongestSubstringFindOut --> LongestSub
```

Q. A Program to check if strings are rotations of each other or not?

Approach

1. Create a temp string and store concatenation of str1 to str1 in temp. temp = str1.str1
2. If str2 is a substring of temp then str1 and str2 are rotations of each other.

Example:
str1 = "ABACD" str2 = "CDABA"

```
    temp = str1.str1 = "ABACDABACD"
```

Since str2 is a substring of temp, str1 and str2 are rotations of each other.

```
class StringRotation
{

 static boolean areRotations(String str1, String str2) {
   // There lengths must be same and str2 must be
   // a substring of str1 concatenated with str1.
   return (str1.length() == str2.length()) && ((str1 + str1).indexOf(str2) != -1);
 }

 public static void main (String[] args) {
   String str1 = "AACD";
   String str2 = "ACDA";

   if (areRotations(str1, str2))
     System.out.println("Strings are rotations of each other");
   else
     System.out.printf("Strings are not rotations of each other");
 }
}
```

Output

```
Strings are rotations of each other
```

Q. Can you write a regular expression to check if String is a number?

```
public class StringTest
{
 public static void main (String[] args) {
   String regex = "[0-9]+";
   // String regex = "\\d+";
   String data = "23343453";
   System.out.println("Is Number: "+ data.matches(regex));
 }
}
```

Output

```
Is Number: true
```

Q. Write a program to find top two maximum numbers in a array?

```
public class TwoMaxNumbers {

    public void printTwoMaxNumbers(int[] nums) {
        int maxOne = 0;
        int maxTwo = 0;
        for(int n:nums) {
            if(maxOne < n) {
                maxTwo = maxOne;
                maxOne = n;
            } else if(maxTwo < n) {
                maxTwo = n;
            }
        }
        System.out.println("First Max Number: "+maxOne);
        System.out.println("Second Max Number: "+maxTwo);
    }

    public static void main(String a[]) {
        int num[] = {5,34,78,2,45,1,99,23};
        TwoMaxNumbers tmn = new TwoMaxNumbers();
        tmn.printTwoMaxNumbers(num);
    }
}
```

Output

```
First Max Number: 99
Second Max Number: 78
```

Q. How to find all the leaders in an integer array in java?

An element is leader if it is greater than all the elements to its right side. And the rightmost element is always a leader. For example int the array {16, 17, 4, 3, 5, 2}, leaders are 17, 5 and 2.

```
public class Test
{
    void printLeaders(int arr[], int size) {
        for (int i = 0; i < size; i++) {
            int j;
            for (j = i + 1; j < size; j++) {
                if (arr[i] <= arr[j])
                    break;
            }
            if (j == size) // the loop didn't break
                System.out.print(arr[i] + " ");
        }
    }

    public static void main(String[] args) {
     Test lead = new Test();
        int arr[] = new int[]{25, 10, 2, 4, 1, 3};
        int n = arr.length;
        lead.printLeaders(arr, n);
    }
}
```

Output

```
25 10 4 3
```

Q. Write a java program to find number of characters, number of words and number of lines in a text file?

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

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

        int charCount = 0;
        int wordCount = 0;
        int lineCount = 0;

        try (
         BufferedReader reader = new BufferedReader(new FileReader("C:\\file.txt"));
        ) {
            // Reading the first line into currentLine
            String currentLine = reader.readLine();
            while (currentLine != null) {
                // Updating the lineCount
                lineCount++;

                // Getting number of words in currentLine
                String[] words = currentLine.split(" ");

                // Updating the wordCount
                wordCount = wordCount + words.length;

                for (String word : words) {
                    charCount = charCount + word.length();
                }

                // Reading next line into currentLine
                currentLine = reader.readLine();
            }

            System.out.println("Number Of Chars In A File : "+charCount);
            System.out.println("Number Of Words In A File : "+wordCount);
            System.out.println("Number Of Lines In A File : "+lineCount);
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Q. Find all pairs of elements whose sum is equal to given number?

```
public class FindPairs
{
 // Prints number of pairs in arr[0..n-1] with sum equal
    // to 'sum'
    public static void getPairsCount(int[] arr, int sum) {
        // Consider all possible pairs and check their sums
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if ((arr[i] + arr[j]) == sum) {
                  System.out.printf("(%d, %d) %n", arr[i], arr[j]);
                }
            }
        }
    }
 public static void main(String args[]) {
        int[] arr = { 1, 5, 7, -1, 5 };
        int sum = 12;
        getPairsCount(arr, sum);
    }
}
```

Output

```
(5, 7)
(7, 5)
```

Q. Program to convert lower to upper case without using toUppercase()?

```
public class LowerToUpperCase
{
    public static void toLowerCase(String a) {
        for (int i = 0; i< a.length(); i++) {
            char character = a.charAt(i);
            if (65 <= character && character <= 90) {
              character = (char)( (character + 32) );
            }
            System.out.print(character);
         }
     }
    public static void main(String[] args) {
  String str = "HELLO WORLD";
        toLowerCase(str);
    }
}
```

Output

```
HELLO WORLD --> hello world
```

Q. Write a program to create deadlock between two threads?

```
public class DeadlockExample
{
 // Creating Object Locks
    static Object ObjectLock1 = new Object();
    static Object ObjectLock2 = new Object();

    private static class ThreadName1 extends Thread {
      public void run() {
         synchronized (ObjectLock1) {
            System.out.println("Thread 1: Has  ObjectLock1");
            /* Adding sleep() method so that
               Thread 2 can lock ObjectLock2 */
            try {
                Thread.sleep(100);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread 1: Waiting for ObjectLock 2");
            /*Thread 1 has ObjectLock1
              but waiting for ObjectLock2*/
            synchronized (ObjectLock2) {
               System.out.println("Thread 1: No DeadLock");
            }
         }
      }
   }
   private static class ThreadName2 extends Thread {
      public void run() {
         synchronized (ObjectLock2) {
            System.out.println("Thread 2: Has  ObjectLock2");
            /* Adding sleep() method so that
               Thread 1 can lock ObjectLock1 */
            try {
                Thread.sleep(100);
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread 2: Waiting for ObjectLock 1");
            /*Thread 2 has ObjectLock2
              but waiting for ObjectLock1*/
            synchronized (ObjectLock1) {
               System.out.println("Thread 2: No DeadLock");
            }
         }
      }
   }

   public static void main(String args[]) {
      ThreadName1 thread1 = new ThreadName1();
      ThreadName2 thread2 = new ThreadName2();
      thread1.start();
      thread2.start();
   }
}
```

Output

```
Thread 1: Has  ObjectLock1
Thread 2: Has  ObjectLock2
Thread 2: Waiting for ObjectLock 1
Thread 1: Waiting for ObjectLock 2
```

To get the Deadlocak details

```
To get the Thread PID
cmd > jps   // 9004 Test
cmd > jcmd 9004 Thread.print


9004:
2019-12-30 20:39:13
Full thread dump Java HotSpot(TM) 64-Bit Server VM (25.121-b13 mixed mode):

"DestroyJavaVM" #12 prio=5 os_prio=0 tid=0x000000000261d800 nid=0x25a8 waiting on condition [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"Thread-1" #11 prio=5 os_prio=0 tid=0x000000001d746000 nid=0xe78 waiting for monitor entry [0x000000001de9e000]
   java.lang.Thread.State: BLOCKED (on object monitor)
        at Test$ThreadName2.run(Test.java:45)
        - waiting to lock <0x000000076b3eae68> (a java.lang.Object)
        - locked <0x000000076b3eae78> (a java.lang.Object)

"Thread-0" #10 prio=5 os_prio=0 tid=0x000000001d745000 nid=0x468c waiting for monitor entry [0x000000001dd9e000]
   java.lang.Thread.State: BLOCKED (on object monitor)
        at Test$ThreadName1.run(Test.java:24)
        - waiting to lock <0x000000076b3eae78> (a java.lang.Object)
        - locked <0x000000076b3eae68> (a java.lang.Object)
```

```
"Service Thread" #9 daemon prio=9 os_prio=0 tid=0x000000001d6c1000 nid=0x2c08 runnable [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"C1 CompilerThread2" #8 daemon prio=9 os_prio=2 tid=0x000000001bd6c000 nid=0x265c waiting on condition [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"C2 CompilerThread1" #7 daemon prio=9 os_prio=2 tid=0x000000001bd46000 nid=0x461c waiting on condition [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"C2 CompilerThread0" #6 daemon prio=9 os_prio=2 tid=0x000000001bd3c800 nid=0x2bb8 waiting on condition [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"Attach Listener" #5 daemon prio=5 os_prio=2 tid=0x000000001bd3b000 nid=0x2b2c waiting on condition [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"Signal Dispatcher" #4 daemon prio=9 os_prio=2 tid=0x000000001bd39800 nid=0x55e4 runnable [0x0000000000000000]
   java.lang.Thread.State: RUNNABLE

"Finalizer" #3 daemon prio=8 os_prio=1 tid=0x000000001bd29000 nid=0x32fc in Object.wait() [0x000000001d09f000]
   java.lang.Thread.State: WAITING (on object monitor)
        at java.lang.Object.wait(Native Method)
        - waiting on <0x000000076b388ec8> (a java.lang.ref.ReferenceQueue$Lock)
        at java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:143)
        - locked <0x000000076b388ec8> (a java.lang.ref.ReferenceQueue$Lock)
        at java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:164)
        at java.lang.ref.Finalizer$FinalizerThread.run(Finalizer.java:209)

"Reference Handler" #2 daemon prio=10 os_prio=2 tid=0x0000000002751000 nid=0x39e4 in Object.wait() [0x000000001cf9f000]
   java.lang.Thread.State: WAITING (on object monitor)
        at java.lang.Object.wait(Native Method)
        - waiting on <0x000000076b386b68> (a java.lang.ref.Reference$Lock)
        at java.lang.Object.wait(Object.java:502)
        at java.lang.ref.Reference.tryHandlePending(Reference.java:191)
        - locked <0x000000076b386b68> (a java.lang.ref.Reference$Lock)
        at java.lang.ref.Reference$ReferenceHandler.run(Reference.java:153)

"VM Thread" os_prio=2 tid=0x000000001bd06800 nid=0x5990 runnable

"GC task thread#0 (ParallelGC)" os_prio=0 tid=0x0000000002676800 nid=0x51d8 runnable

"GC task thread#1 (ParallelGC)" os_prio=0 tid=0x0000000002678000 nid=0x489c runnable

"GC task thread#2 (ParallelGC)" os_prio=0 tid=0x000000000267a000 nid=0x2e5c runnable

"GC task thread#3 (ParallelGC)" os_prio=0 tid=0x000000000267b800 nid=0x52ec runnable

"VM Periodic Task Thread" os_prio=2 tid=0x000000001d6cb000 nid=0x45ec waiting on condition

JNI global references: 5


Found one Java-level deadlock:
=============================
"Thread-1":
  waiting to lock monitor 0x00000000027572b8 (object 0x000000076b3eae68, a java.lang.Object),
  which is held by "Thread-0"
"Thread-0":
  waiting to lock monitor 0x0000000002759d58 (object 0x000000076b3eae78, a java.lang.Object),
  which is held by "Thread-1"

Java stack information for the threads listed above:
===================================================
"Thread-1":
        at Test$ThreadName2.run(Test.java:45)
        - waiting to lock <0x000000076b3eae68> (a java.lang.Object)
        - locked <0x000000076b3eae78> (a java.lang.Object)
"Thread-0":
        at Test$ThreadName1.run(Test.java:24)
        - waiting to lock <0x000000076b3eae78> (a java.lang.Object)
        - locked <0x000000076b3eae68> (a java.lang.Object)

Found 1 deadlock.
```

Q. How to find the word with the highest frequency from a file in Java?

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

   int count = 0;
   String mostRepeatedWord = null;

   // Creating wordCountMap which holds words as keys and their occurrences as values
   HashMap wordCountMap = new HashMap();

   try (
     BufferedReader reader = new BufferedReader(new FileReader("C:\\file.txt"));
   ) {
     // Reading the first line into currentLine
```

```
    String currentLine = reader.readLine();

    while (currentLine != null) {
      // splitting the currentLine into words
      String[] words = currentLine.toLowerCase().split(" ");

      for (String word : words) {
        // If word is already present in wordCountMap, updating its count
        if(wordCountMap.containsKey(word)) {
          wordCountMap.put(word, wordCountMap.get(word)+1);
        } else {
          wordCountMap.put(word, 1);
        }
      }
      // Reading next line into currentLine
      currentLine = reader.readLine();
    }

    // Getting the most repeated word and its occurrence
    Set> entrySet = wordCountMap.entrySet();

    for (Entry entry : entrySet) {
      if(entry.getValue() > count) {
        mostRepeatedWord = entry.getKey();
        count = entry.getValue();
      }
    }

    System.out.println("The most repeated word in input file is: "+mostRepeatedWord);
    System.out.println("Number Of Occurrences: "+count);
   }
   catch (IOException e) {
    e.printStackTrace();
   }
  }
}
```

Output

```
The most repeated word in input file is: java
Number Of Occurrences: 5
```

Q. How to sort a text file in java?

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

   //Create an ArrayList object to hold the lines of input file
   ArrayList lines = new ArrayList();

   try (
    BufferedReader reader = new BufferedReader(new FileReader("C:\\file.txt"));
    BufferedWriter writer = new BufferedWriter(new FileWriter("C:\\output.txt"));
   )
   {
    //Reading all the lines of input file one by one and adding them into ArrayList
    String currentLine = reader.readLine();

    while (currentLine != null) {
     lines.add(currentLine);
     currentLine = reader.readLine();
    }

    //Sorting the ArrayList
    Collections.sort(lines);

    //Writing sorted lines into output file
    for (String line : lines) {
     writer.write(line);
     writer.newLine();
    }
   }
   catch (IOException e) {
    e.printStackTrace();
   }
  }
}
```

Output

```
20
50
Enterprise JavaBeans (EJB)
J2EE Connector Architecture (J2EE-CA, or JCA)
Java Database Connectivity (JDBC)
Java Message Service (JMS)
Java Naming and Directory Interface (JNDI)
Java Servlets
```

Java Transaction API (JTA)
Java Transaction Service (JTS)
JavaMail
JavaServer Pages (JSP)

Q. Find middle index of array where both ends sum is equal?

```
public class FindMiddleIndex
{
    public static int findMiddleIndex(int[] array) throws Exception {

        int endIndex = array.length - 1;
        int startIndex = 0;
        int leftSum = 0;
        int rightSum = 0;
        while (true) {
            if (leftSum > rightSum) {
                rightSum += array[endIndex--];
            } else {
                leftSum += array[startIndex++];
            }
            if (startIndex > endIndex) {
                if (leftSum == rightSum) {
                    break;
                } else {
                    throw new Exception("No such combination found in the array.");
                }
            }
        }
        return endIndex;
    }

    public static void main(String[] args) {
        int[] array = {1, 7, 5, 2, 8, 3};
        try {
            int index = findMiddleIndex(array);
            System.out.println("Sum preceding the index " + index + " is equal to sum succeeding the index " + index);
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

Output

```
Sum preceding the index 2 is equal to sum succeeding the index 2
```

Q. What do the expression 1.0 / 0.0 will return? will it throw Exception? any compile time error?

- Output: Infinity, No Exception

Q. How do you check the equality of two arrays in java?

- Arrays.equals()

Q. How to remove duplicate elements from ArrayList in java?

Using Java 8 Stream.distinct()

Approach

1. Get the ArrayList with duplicate values.
2. Create a new List from this ArrayList.
3. Using Stream().distinct() method which return distinct object stream.
4. convert this object stream into List

```
/**
* Java program to remove duplicates from ArrayList
*
**/
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;

class ArrayListExample
{
 public static void main(String[] args) {
   // input list with duplicates
   List list = new ArrayList<>(Arrays.asList(1, 10, 1, 2, 2, 3, 10, 3, 3, 4, 5, 5));
   System.out.println("ArrayList with duplicates: "+ list);

   // Construct a new list from the set constucted from elements
   // of the original list
   List newList = list.stream().distinct().collect(Collectors.toList());

   System.out.println("ArrayList with duplicates removed: "+ newList);
  }
}
```

Output

```
ArrayList with duplicates: [1, 10, 1, 2, 2, 3, 10, 3, 3, 4, 5, 5]
ArrayList with duplicates removed: [1, 10, 2, 3, 4, 5]
```

Q. Write a java program to append text to a file?

```
public class Test
{
 public static void main(String[] args) {
        String path = "C:\\file.txt";
        String text = "Append new text";
        try (
         FileWriter fw = new FileWriter(path, true);
        ) {
            fw.write(text);
        }
        catch(IOException e) {
         e.printStackTrace();
        }
    }
}
```

Q. Write a program to sort a map by value?

```
public class SortMapExample
{
 public static Map sortMap(Map map) {

        List> capitalList = new LinkedList<>(map.entrySet());
        Collections.sort(capitalList, (o1, o2) -> o1.getValue().compareTo(o2.getValue()));
        LinkedHashMap result = new LinkedHashMap<>();

        for (Map.Entry entry : capitalList) {
            result.put(entry.getKey(), entry.getValue());
        }
        return result;
    }
    public static void main(String[] args) {

     LinkedHashMap capitals = new LinkedHashMap<>();
        capitals.put("Nepal", "Kathmandu");
        capitals.put("India", "New Delhi");
        capitals.put("United States", "Washington");
        capitals.put("England", "London");
        capitals.put("Australia", "Canberra");

        Map result = sortMap(capitals);

        for (Map.Entry entry : result.entrySet()) {
            System.out.print("Key: " + entry.getKey());
            System.out.println(", Value: " + entry.getValue());
        }
    }
}
```

Output

```
Key: Australia,     Value: Canberra
Key: Nepal,         Value: Kathmandu
Key: England,       Value: London
Key: India,         Value: New Delhi
Key: United States, Value: Washington
```

Q. How to find duplicate number on Integer array in Java?

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

        int[] array = {4, 2, 4, 5, 2, 3, 1};
        Set set = new HashSet<>();

        for(int i = 0; i < array.length ; i++) {
            //If same integer is already present then add method will return FALSE
            if(set.add(array[i]) == false) {
                System.out.println("Duplicate Element Found: " + array[i]);
            }
        }
    }
}
```

Output

```
Duplicate Element: 4
Duplicate Element: 2
```

Q. How to find largest and smallest number in unsorted array?

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

        int numbers[] = {85, 91, 7, 98, 71, 57, 20, 38, 97, 6};
        int smallest = numbers[0];
        int biggest = numbers[0];

        for(int i = 1; i < numbers.length; i++) {
                if(numbers[i] > biggest)
                        biggest = numbers[i];
                else if (numbers[i] < smallest)
                        smallest = numbers[i];
        }
        System.out.println("Smallest Number is : " + smallest);
        System.out.println("Largest Number is : " + biggest);
    }
}
```

Output

```
Smallest Number is: 6
Largest Number is: 98
```

Q. FizzBuzz problem:- Write a program which return "fizz" if the number is a multiplier of 3, return "buzz" if its multiplier of 5 and return "fizzbuzz" if the number is divisible by both 3 and 5. If the number is not divisible by either 3 or 5 then it should just return the number itself?

```
public class FizzBuzzExample
{
 public static String fizzBuzz(int number) {
        if (number % 3 == 0) {
            if (number % 5 == 0) {
                return "fizzbuzz";
            } else {
                return "fizz";
            }
        } else if (number % 5 == 0) {
            return "buzz";
        }
        return String.valueOf(number);
    }
 public static void main(String[] args) {
   System.out.println(fizzBuzz(15));
  }
}
```

Output

```
3   ---> fizz
15  ---> fizzbuzz
23  ---> 23
```

Q. Write a Comparator in Java to compare two employees based upon their name, departments and age?

Name Sorter

```
import java.util.Comparator;

public class NameSorter implements Comparator{

    @Override
    public int compare(Employee o1, Employee o2) {
        return o1.getName().compareTo(o2.getName());
    }
}
```

Department Sorter

```
import java.util.Comparator;

public class DepartmentSorter implements Comparator {

    @Override
    public int compare(Employee o1, Employee o2) {
        return o1.getDepartment().compareTo(o2.getDepartment());
    }
}
```

Age Sorter

```
import java.util.Comparator;

public class AgeSorter implements Comparator {
    @Override
    public int compare(Employee o1, Employee o2) {
        return o1.getAge() - o2.getAge();
    }
}
```

compare with Comparator

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

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

        Employee e1 = new Employee(1, "aTestName", "dLastName", 34);
        Employee e2 = new Employee(2, "nTestName", "pLastName", 30);
        Employee e3 = new Employee(3, "kTestName", "sLastName", 31);
        Employee e4 = new Employee(4, "dTestName", "zLastName", 25);

        List employees = new ArrayList();
        employees.add(e2);
        employees.add(e3);
        employees.add(e1);
        employees.add(e4);

        // UnSorted List
        System.out.println(employees);

        Collections.sort(employees);

        // Default Sorting by employee id
        System.out.println(employees);

        Collections.sort(employees, new NameSorter());

        // Sorted by Employee Name
        System.out.println(employees);

        Collections.sort(employees, new DepartmentSorter());

        // Sorted by Department Name
        System.out.println(employees);

        Collections.sort(employees, new AgeSorter());

        // Sorted by Employee Age
        System.out.println(employees);
    }
}
```

Output:

```
//Unsorted

[Employee : 2 - nTestName - pLastName - 30
, Employee : 3 - kTestName - sLastName - 31
, Employee : 1 - aTestName - dLastName - 34
, Employee : 4 - dTestName - zLastName - 25]

//Default sorting based on employee id

[Employee : 1 - aTestName - dLastName - 34
, Employee : 2 - nTestName - pLastName - 30
, Employee : 3 - kTestName - sLastName - 31
, Employee : 4 - dTestName - zLastName - 25]
```

Q. Write a program to convert Decimal To Binary, Decimal To Octal and Decimal to HexaDecimal in Java?

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

        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the Decimal Number: ");
        int inputNumber = sc.nextInt();

        int copyOfInputNumber = inputNumber;
        String binary = "";
        int rem = 0;

        while (inputNumber > 0) {
            rem = inputNumber % 2;
            binary =  rem + binary;
            inputNumber = inputNumber/2;
        }
        System.out.println("Binary Equivalent of "+copyOfInputNumber+" is "+binary);
    }
}
```

Output

```
Enter the Decimal Number: 5

Binary Equivalent of 5 is 101
```

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

        Scanner sc = new Scanner(System.in);
        System.out.println("Enter The Decimal Number: ");

        int inputNumber = sc.nextInt();
        int copyOfInputNumber = inputNumber;
        String octal = "";
        int rem = 0;

        while (inputNumber > 0) {
            rem = inputNumber % 8;
            octal =  rem + octal;
            inputNumber = inputNumber / 8;
        }
        System.out.println("Octal Equivalent of "+copyOfInputNumber+" is "+octal);
    }
}
```

Output

```
Enter The Decimal Number: 8

Octal Equivalent of 8 is 10
```

```
public class DecimalToHexaDecimal
{
 public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter The Decimal Number: ");

        int inputNumber = sc.nextInt();
        int copyOfInputNumber = inputNumber;
        String hexa = "";

        //Digits in HexaDecimal Number System
        char hexaDecimals[] = {'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};
        int rem = 0;

        while (inputNumber > 0) {
            rem = inputNumber % 16;
            hexa =  hexaDecimals[rem] + hexa;
            inputNumber = inputNumber / 16;
        }
        System.out.println("HexaDecimal Equivalent of "+copyOfInputNumber+" is "+hexa);
    }
}
```

Output

```
Enter The Decimal Number: 100

HexaDecimal Equivalent of 100 is 64
```

Q. How to rearrange array in alternating positive and negative number?

Objective: Given an array arrA[] which has negative and positive elements, rearrange the array in such a manner that positive and negative elements occupy the alternate positions and if there are extra positive or negative elements are left then append it to the end.

Example:

```
int[] arrA = { 1, 2, -3, -4, -5, 6, -7, -8, 9, 10, -11, -12, -13, 14 };
Output: -13 9 -3 10 -5 6 -7 2 -12 1 -11 14 -4 -8
```

Quick Sort Technique

- Take the pivot element as 0 and do the first round of Quick Sort.
- After above step you will have all the negative elements on left and all the positive elements on the right.
- Then just the every alternate element in the left half (negative elements) with the elements in the right (positive elements)

```
public class RearragePostiveNegativeAlternatively
{
 public void rerrange(int[] arrA) {
  int pivot = 0;
  int left = 0;
  int right = arrA.length - 1;
  while (right > left) {
   while (arrA[left] < 0 && left < right)
    left++;
   while (arrA[right] > 0 && left < right)
    right--;
   if (left < right) {

    int temp = arrA[left];
    arrA[left] = arrA[right];
    arrA[right] = temp;
    left++;
    right--;
   }
  }
  // At the point all the negative elements on the left half and
  // positive elements on the right half of the array
  // swap the every alternate element in the left half (negative
  // elements) with the elements in the right (positive elements)
  left = 1;
  int high = 0;
  while (arrA[high] < 0)
   high++;
  right = high;
  while (arrA[left] < 0 && right < arrA.length) {
   int temp = arrA[left];
   arrA[left] = arrA[right];
   arrA[right] = temp;
   left = left + 2;
   right++;
  }
  for (int i = 0; i < arrA.length; i++) {
   System.out.print("  " + arrA[i]);
  }
 }

 public static void main(String[] args) throws java.lang.Exception {
  int[] arrA = { 1, 2, -3, -4, -5, 6, -7, -8, 9, 10, -11, -12, -13, 14 };
  RearragePostiveNegativeAlternatively i = new RearragePostiveNegativeAlternatively();
  i.rerrange(arrA);
 }
}
```

Output

```
-13 9 -3 10 -5 6 -7 2 -12 1 -11 14 -4 -8
```

Q. How Convert lower to upper case without using toUppercase() in java?

```
/**
* Java program to Convert lower to upper case
*
**/
public class toLowerCase {
    public static void main(String[] args) {
        toLowerCase(args[0]);
    }

    public static void toLowerCase(String a) {

        for (int i = 0; i< a.length(); i++) {
            char aChar = a.charAt(i);
            if (65 <= aChar && aChar<=90) {
                aChar = (char)( (aChar + 32) );
            }
            System.out.print(aChar);
         }
      }
}
```

↥ back to top

Q. Explain deadlock condition in-between two threads with example?

```
public class DeadLockSimulator {

    public static Object Lock1 = new Object();
    public static Object Lock2 = new Object();

    private static class FirstThread extends Thread {
        public void run() {
            synchronized (Lock1) {
            System.out.println("Thread 1: Holding lock 1...");
            try { Thread.sleep(10); } catch (Exception e) {}
            System.out.println("Thread 1: Waiting for lock 2...");
            synchronized (Lock2) {
                System.out.println("Thread 1: Holding lock 1 & 2...");
            }
            }
        }
    }

    private static class SecondThread extends Thread {
        public void run() {
            synchronized (Lock2) {
            System.out.println("Thread 2: Holding lock 2...");
            try { Thread.sleep(10); } catch (Exception e) {}
            System.out.println("Thread 2: Waiting for lock 1...");
            synchronized (Lock1) {
                System.out.println("Thread 2: Holding lock 1 & 2...");
            }
            }
        }
    }

    public static void main(String args[]) {

        new FirstThread().start();
        new SecondThread().start();
    }
}
```

Output:

```
"Thread-1" prio=6 tid=0x0000000007319000 nid=0x7cd3c waiting for monitor entry [0x0000000008a3f000]
   java.lang.Thread.State: BLOCKED (on object monitor)
    at com.tier1app.DeadLockSimulator$SecondThread.run(DeadLockSimulator.java:29)
    - waiting to lock 0x00000007ac3b1970 (a java.lang.Object)
    - locked 0x00000007ac3b1980 (a java.lang.Object)

   Locked ownable synchronizers:
    - None

"Thread-0" prio=6 tid=0x0000000007318800 nid=0x7da14 waiting for monitor entry [0x000000000883f000]
   java.lang.Thread.State: BLOCKED (on object monitor)
    at com.tier1app.DeadLockSimulator$FirstThread.run(DeadLockSimulator.java:16)
    - waiting to lock 0x00000007ac3b1980 (a java.lang.Object)
    - locked 0x00000007ac3b1970 (a java.lang.Object)

   Locked ownable synchronizers:
    - None
```

Q. How do you sort out items in ArrayList in reverse direction?

Reverse order of all elements of Java ArrayList

```
import java.util.ArrayList;
import java.util.Collections;
```

```
public class MainClass {
  public static void main(String[] args) {
    ArrayList arrayList = new ArrayList();

    arrayList.add("A");
    arrayList.add("B");
    arrayList.add("C");
    arrayList.add("D");
    arrayList.add("E");

    System.out.println(arrayList);
    Collections.reverse(arrayList);
    System.out.println(arrayList);
  }
}
```

↥ back to top

Q. Find the shortest-path weights d(s, v) from given source s for all vertices v present in the graph.

Dijkstras Algorithm

For Example:

```
Path from vertex A to vertex B has min cost of 4 & the route is [ A -> E -> B ]

Path from vertex A to vertex C has min cost of 6 & the route is [ A -> E -> B -> C ]

Path from vertex A to vertex D has min cost of 5 & the route is [ A -> E -> D ]

Path from vertex A to vertex E has min cost of 3 & the route is [ A -> E ]
```

Dijkstra's Algorithm is an algorithm for finding the shortest paths between nodes in a graph. For a given source node in the graph, the algorithm finds the shortest path between that node and every other node. It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined.

Dijkstra's Algorithm is based on the principle of relaxation, in which an approximation to the correct distance is gradually replaced by more accurate values until shortest distance is reached. The approximate distance to each vertex is always an overestimate of the true distance, and is replaced by the minimum of its old value with the length of a newly found path. It uses a priority queue to greedily select the closest vertex that has not yet been processed and performs this relaxation process on all of its outgoing edges.

```
import java.util.*;

// Data structure to store graph edges
class Edge
{
 int source, dest, weight;

 public Edge(int source, int dest, int weight) {
   this.source = source;
   this.dest = dest;
   this.weight = weight;
 }
};

// data structure to store heap nodes
class Node {
 int vertex, weight;

 public Node(int vertex, int weight) {
   this.vertex = vertex;
   this.weight = weight;
 }
};

// class to represent a graph object
class Graph
{
 // A List of Lists to represent an adjacency list
 List> adjList = null;

 // Constructor
 Graph(List edges, int N) {

   adjList = new ArrayList<>(N);
```

```
    for (int i = 0; i < N; i++) {
      adjList.add(i, new ArrayList<>());
    }

    // add edges to the undirected graph
    for (Edge edge: edges) {
      adjList.get(edge.source).add(edge);
    }
  }
}

class Main
{
  private static void printRoute(int prev[], int i) {
    if (i < 0)
      return;

    printRoute(prev, prev[i]);
    System.out.print(i + " ");
  }

  // Run Dijkstra's algorithm on given graph
  public static void shortestPath(Graph graph, int source, int N) {

    // create min heap and push source node having distance 0
    PriorityQueue minHeap = new PriorityQueue<>((lhs, rhs) -> lhs.weight - rhs.weight);
    minHeap.add(new Node(source, 0));

    // set infinite distance from source to v initially
    List dist = new ArrayList<>(Collections.nCopies(N, Integer.MAX_VALUE));

    // distance from source to itself is zero
    dist.set(source, 0);

    // boolean array to track vertices for which minimum
    // cost is already found
    boolean[] done = new boolean[N];
    done[0] = true;

    // stores predecessor of a vertex (to print path)
    int prev[] = new int[N];
    prev[0] = -1;

    // run till minHeap is not empty
    while (!minHeap.isEmpty()) {

      // Remove and return best vertex
      Node node = minHeap.poll();

      // get vertex number
      int u = node.vertex;

      // do for each neighbor v of u
      for (Edge edge: graph.adjList.get(u)) {

        int v = edge.dest;
        int weight = edge.weight;

        // Relaxation step
        if (!done[v] && (dist.get(u) + weight) < dist.get(v)) {
          dist.set(v, dist.get(u) + weight);
          prev[v] = u;
          minHeap.add(new Node(v, dist.get(v)));
        }
      }

      // marked vertex u as done so it will not get picked up again
      done[u] = true;
    }

    for (int i = 1; i < N; ++i) {
      System.out.print("Path from vertex 0 to vertex " + i + " has minimum cost of "
           + dist.get(i) + " and the route is [ ");
      printRoute(prev, i);
      System.out.println("]");
    }
  }

  public static void main(String[] args) {

    // initialize edges as per above diagram
    // (u, v, w) triplet represent undirected edge from
    // vertex u to vertex v having weight w
    List edges = Arrays.asList(
        new Edge(0, 1, 10), new Edge(0, 4, 3),
        new Edge(1, 2, 2), new Edge(1, 4, 4),
        new Edge(2, 3, 9), new Edge(3, 2, 7),
        new Edge(4, 1, 1), new Edge(4, 2, 8),
        new Edge(4, 3, 2)
    );

    // Set number of vertices in the graph
    final int N = 5;
    // construct graph
    Graph graph = new Graph(edges, N);

    shortestPath(graph, 0, N);
  }
}
```

Output

```
Path from vertex 0 to vertex 1 has minimum cost of 4 and the route is [ 0 4 1 ]
Path from vertex 0 to vertex 2 has minimum cost of 6 and the route is [ 0 4 1 2 ]
Path from vertex 0 to vertex 3 has minimum cost of 5 and the route is [ 0 4 3 ]
Path from vertex 0 to vertex 4 has minimum cost of 3 and the route is [ 0 4 ]
```

↑ back to top

Q. How to find if there is a sub array with sum equal to zero?

Q. How to remove a given element from array in Java?

Q. How to find trigonometric values of an angle in java?

Q. Reverse and add until you get a palindrome

Q. Selection sort in java

Q. Write a singleton class in java?

Q. How to calculate complexity of algorithm?

Q. Implement Producer Consumer design Pattern in Java using wait, notify and notifyAll method in Java?

Q. Find Minimum numbers of platforms required for railway station in java

Q. How to check whether given number is binary or not?

Q. Armstrong number program in java

Q. How to find sum of all digits of a number in java?

Q. How to find largest number less than a given number and without a given digit?

Q. Roman equivalent of a decimal number

Q. How to check whether user input is number or not in java?

Q. Write a program to convert decimal number to binary format.

Q. Write a program to find perfect number or not.

Q. Write a program to find sum of each digit in the given number using recursion.

Q. Write a program to check the given number is a prime number or not?

Q. Write a program to check the given number is binary number or not?

Q. Find prime factors of number in java

Q. Write code to avoid deadlock in Java where N threads are accessing N shared resources?

Q. How to sort an array in place using QuickSort algorithm?

Q. Write a program to find intersection of two sorted arrays in Java?

Q. How find the first repeating element in an array of integers?

Q. How to find first non-repeating element in array of integers?

Q. How to find top two numbers from an integer array?

Q. How to find the smallest positive integer value that cannot be represented as sum of any subset of a given array?

Q. How to merge sorted array?

Q. How to find sub array with maximum sum in an array of positive and negative number?

Q. How to find sub array with largest product in array of both positive and negative number?

Q. Write a program to find length of longest consecutive sequence in array of integers?

Q. How to find minimum value in a rotated sorted array?

Q. Given an array of of size n and a number k, find all elements that appear more than n/k times?

Q. Returns the largest sum of contiguous integers in the array

Q. Return the sum two largest integers in an array

Q. How to search an array to check if an element exists there?

Q. How to copy array in Java?

Q. Write a function to find out longest palindrome in a given string?

Q. Can you make array volatile in Java?

Q. How to find top two numbers from an integer array in Java?

Q. Can you pass the negative number as an array size?

Q. What is an anonymous array? Give example?

Q. What is the difference between int[] a and int a[] ?

Q. What are jagged arrays in java? Give example?

Q. While creating the multidimensional arrays, can you specify an array dimension after an empty dimension?

Q. How to reverse Singly Linked List?

Q. Create a Java program to find middle node of linked list in Java in one pass?

Q. How to find if a linked list contains cycle or not in Java?

Q. How to find nth element from end of linked list

Q. How to check if linked list is palindrome in java

Q. Add two numbers represented by linked list in java

Q. How to sort a Stack using a temporary Stack?

Q. Implement Binary Search Tree (BST)

Q. Find min and max value from Binary Search Tree (BST)

Q. Find height of a Binary Search Tree (BST)

Q. Implement Binary Search Tree (BST) Level order traversal (breadth first).

Q. Implement Binary Search Tree (BST) pre-order traversal (depth first).

Q. Implement Binary Search Tree (BST) in-order traversal (depth first).

Q. Implement Binary Search Tree (BST) post-order traversal (depth first).

Q. How to check the given Binary Tree is Binary Search Tree (BST) or not?

Q. How to delete a node from Binary Search Tree (BST)?

Q. Binary tree level order traversal

Q. Binary tree spiral order traversal

Q. Binary tree reverse level order traversal

Q. Binary tree boundary traversal

Q. Print leaf nodes of binary tree

Q. Count leaf nodes in binary tree

Q. Get maximum element in binary tree

Q. Print all paths from root to leaf in binary tree

Q. Print vertical sum of binary tree in java

Q. Get level of node in binary tree in java

Q. Lowest common ancestor(LCA) in binary tree in java

Q. Search element in row wise and column wise sorted matrix

Q. Stock buy and sell to maximize profit.

Q. How to implement merge sort in java

Q. How to implement bubble sort in java

Q. How to implement insertion sort in java

Q. Write a program to implement hashcode and equals.

Q. Write wait-notify code for producer-consumer problem?

Q. Write a program to implement ArrayList.

Q. A maximal subarray

Q. Sort a linked list

Q. Iterative Quick sort

Q. Bucket sort

Q. Counting sort

Q. Square root of number

Q. Printing patterns

Q. Leap year

Q. Design a Vending Machine

Q. Transpose a matrix

Q. Adding two matrices in Java

Q. Matrix multiplication

Q. Write a java program to print Floyd's Triangle?

Q. Spiral Matrix Program.

Q. Anagram program in java

Q. Write a program to print fibonacci series.

Q. How to you calculate the difference between two dates in Java?

Q. Java Program to find gcd and lcm of two numbers

Q. Write a program to find the sum of the first 1000 prime numbers?

Q. How to perform matrix operations in java?

↥ back to top