Array Programs

By

Praveen Oruganti

Blog: https://praveenoruganti.blogspot.com

Facebook Group: https://www.facebook.com/groups/2340886582907696/

Github repo: https://github.com/praveenoruganti

Program to find second largest element in an Array

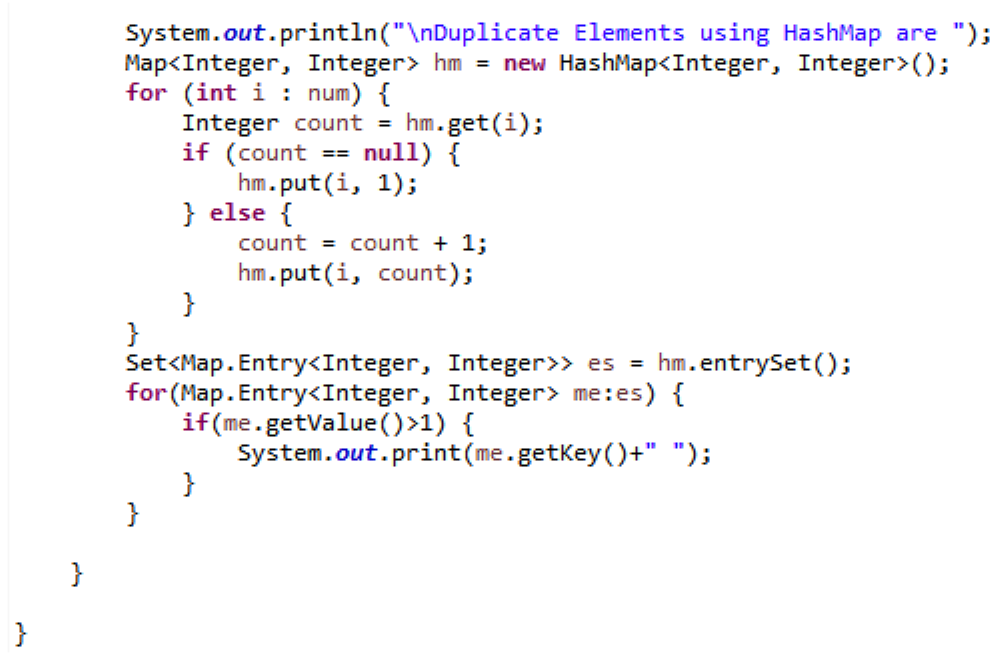
```
public class ArraySecondLargest {

    public static void main(String[] args) {
        int[] num = { 90, 24, 46, 35, 32, 12, 98, 2 };
        int largest = Integer.MIN_VALUE;
        int secondLargest = Integer.MIN_VALUE;
        for (int i = 0; i < num.length; i++) {
            if (num[i] > largest) {
                secondLargest = largest;
                largest = num[i];
            } else if (num[i] > secondLargest && num[i] != largest) {
                secondLargest = num[i];
            }

        }

        System.out.println("Second Largest Element "+ secondLargest);
    }

}
```

Output

```
Second Largest Element 90
```

Program to find duplicate elements in an Array

```
public class ArrayDuplicateElements {

    public static void main(String[] args) {
        int[] num = { 90, 24, 46, 35, 32, 12, 98, 2, 90, 16, 24, 30, 32 };

        System.out.println("Duplicate Elements using brute force method are ");
        for (int i = 0; i < num.length; i++) {
            for (int j = i + 1; j < num.length; j++) {
                if (num[i] == num[j] && i != j) {
                    System.out.print(num[j] + " ");
                }
            }
        }

        System.out.println("\nDuplicate Elements using HashSet are ");
        Set<Integer> hs = new HashSet<Integer>();
        for (int i : num) {
            if (hs.add(i) == false) {
                System.out.print(i + " ");
            }
        }
```

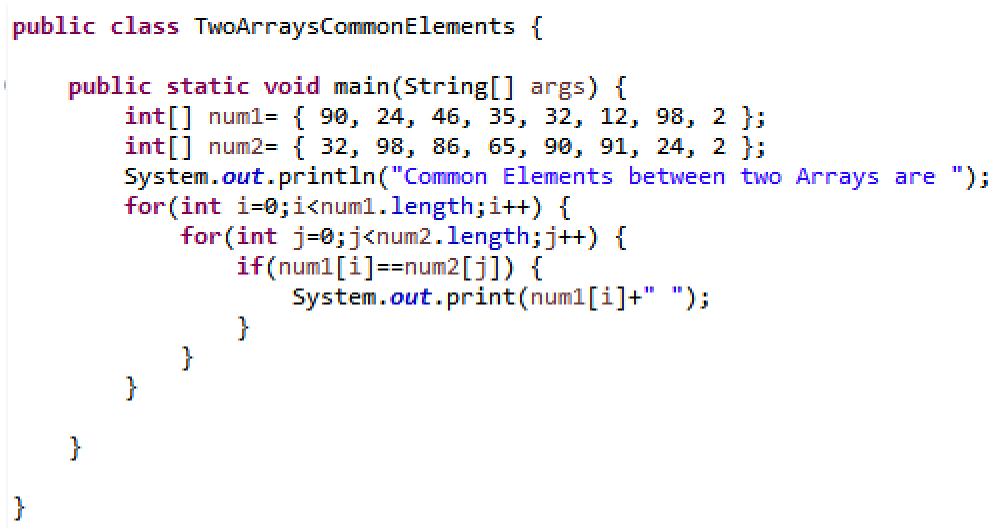


```
        System.out.println("\nDuplicate Elements using HashMap are ");
        Map<Integer, Integer> hm = new HashMap<Integer, Integer>();
        for (int i : num) {
            Integer count = hm.get(i);
            if (count == null) {
                hm.put(i, 1);
            } else {
                count = count + 1;
                hm.put(i, count);
            }
        }
        Set<Map.Entry<Integer, Integer>> es = hm.entrySet();
        for(Map.Entry<Integer, Integer> me:es) {
            if(me.getValue()>1) {
                System.out.print(me.getKey()+" ");
            }
        }

    }

}
```

Output

```
Duplicate Elements using brute force method are
90 24 32
Duplicate Elements using HashSet are
90 24 32
Duplicate Elements using HashMap are
32 24 90
```

<u>Program to find common elements between two Arrays</u>

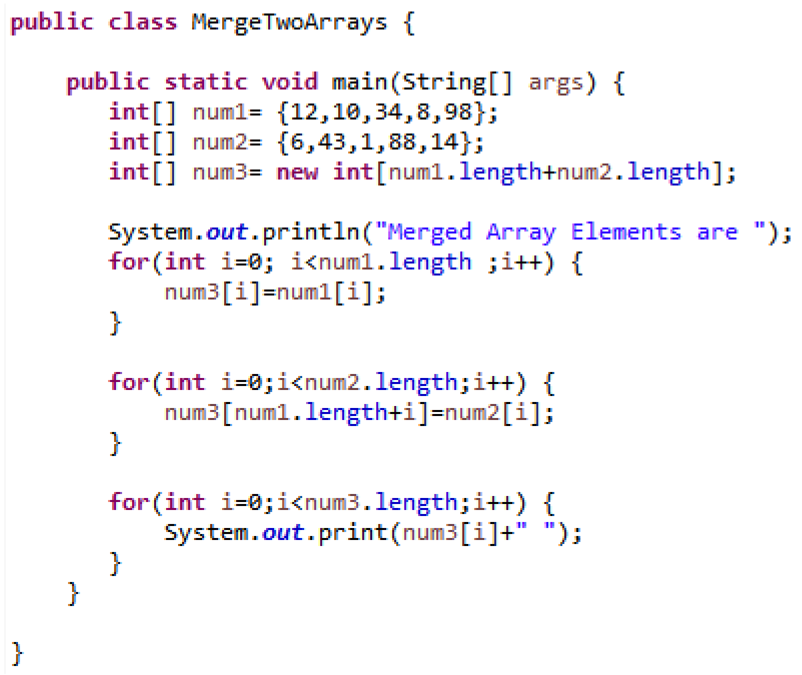


```
public class TwoArraysCommonElements {

    public static void main(String[] args) {
        int[] num1= { 90, 24, 46, 35, 32, 12, 98, 2 };
        int[] num2= { 32, 98, 86, 65, 90, 91, 24, 2 };
        System.out.println("Common Elements between two Arrays are ");
        for(int i=0;i<num1.length;i++) {
            for(int j=0;j<num2.length;j++) {
                if(num1[i]==num2[j]) {
                    System.out.print(num1[i]+" ");
                }
            }
        }

    }

}
```

Output

```
Common Elements between two Arrays are
90 24 32 98 2
```

Program to merge two Arrays

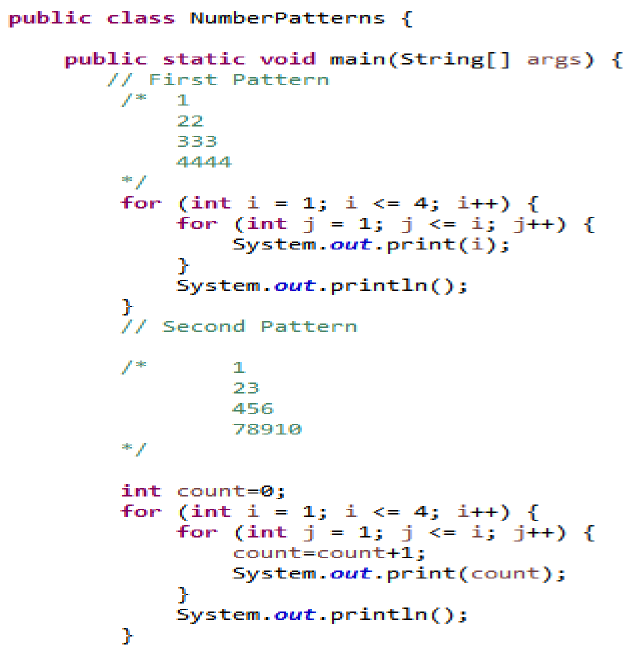


```
public class MergeTwoArrays {

    public static void main(String[] args) {
       int[] num1= {12,10,34,8,98};
       int[] num2= {6,43,1,88,14};
       int[] num3= new int[num1.length+num2.length];

       System.out.println("Merged Array Elements are ");
       for(int i=0; i<num1.length ;i++) {
           num3[i]=num1[i];
       }

       for(int i=0;i<num2.length;i++) {
           num3[num1.length+i]=num2[i];
       }

       for(int i=0;i<num3.length;i++) {
           System.out.print(num3[i]+" ");
       }
    }

}
```

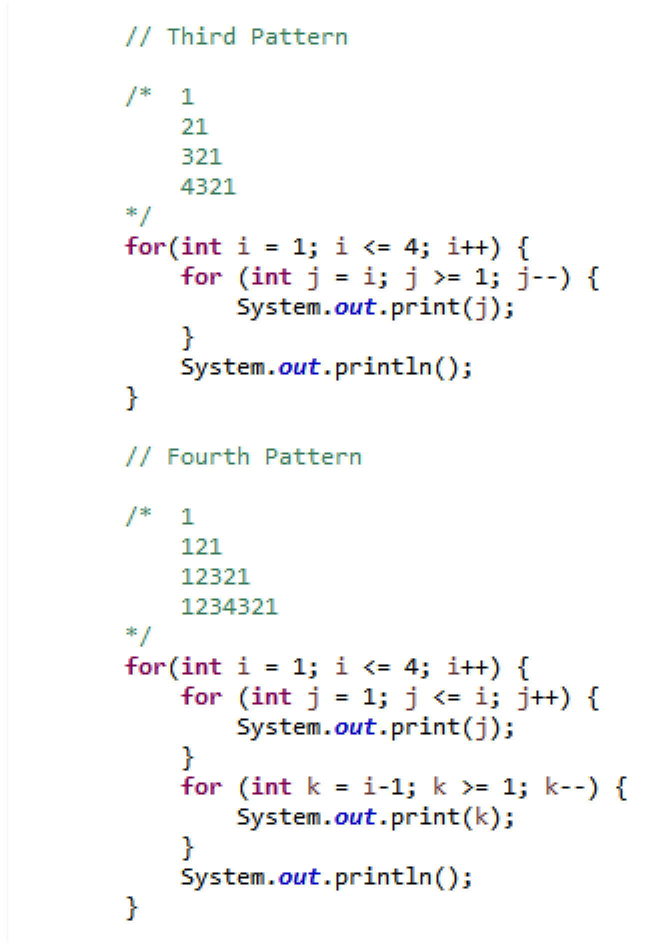
Output

```
Merged Array Elements are
12 10 34 8 98 6 43 1 88 14
```

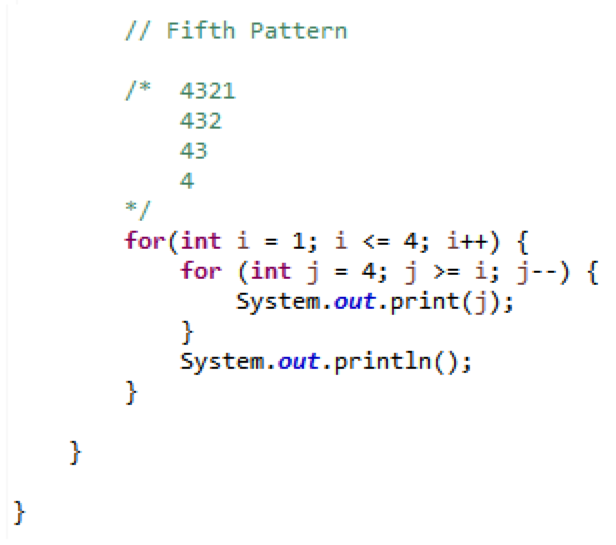
Number Patterns



```
public class NumberPatterns {

    public static void main(String[] args) {
       // First Pattern
        /*  1
            22
            333
            4444
        */
        for (int i = 1; i <= 4; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print(i);
            }
            System.out.println();
        }
        // Second Pattern

        /*      1
                23
                456
                78910
        */

        int count=0;
        for (int i = 1; i <= 4; i++) {
            for (int j = 1; j <= i; j++) {
                count=count+1;
                System.out.print(count);
            }
            System.out.println();
        }
```

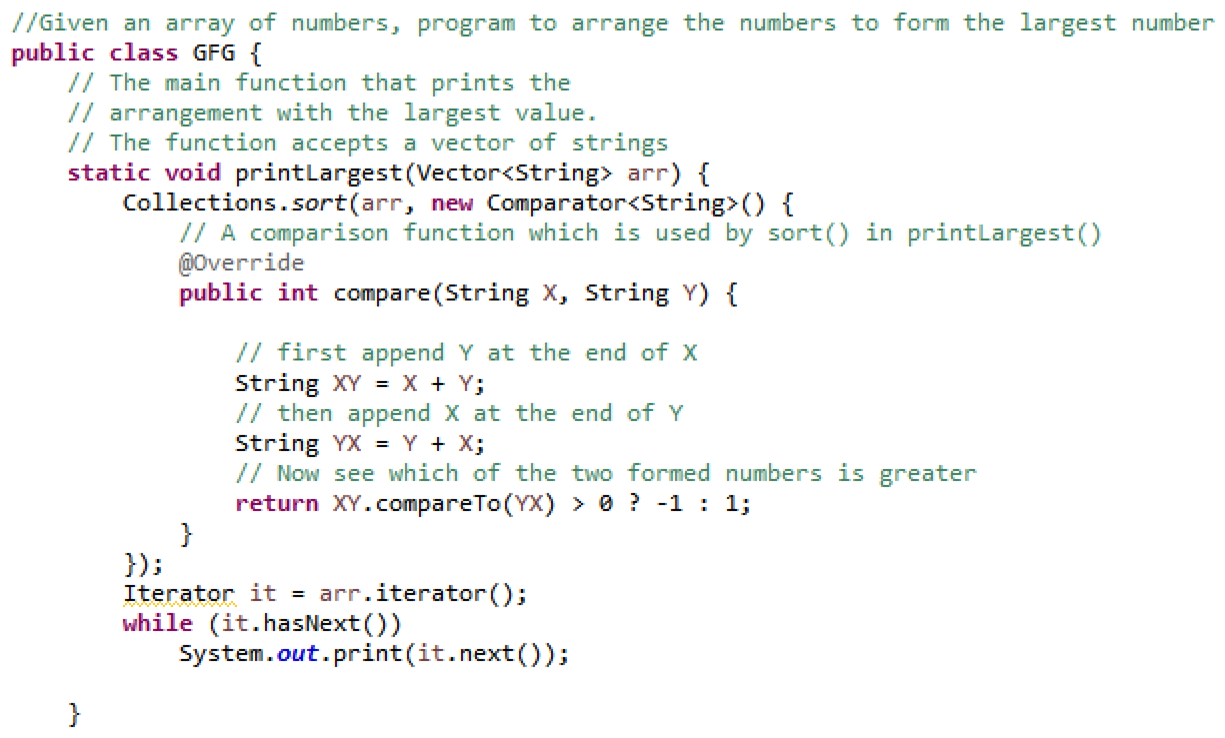


```
        // Third Pattern

        /*  1
            21
            321
            4321
        */
        for(int i = 1; i <= 4; i++) {
            for (int j = i; j >= 1; j--) {
                System.out.print(j);
            }
            System.out.println();
        }

        // Fourth Pattern

        /*  1
            121
            12321
            1234321
        */
        for(int i = 1; i <= 4; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print(j);
            }
            for (int k = i-1; k >= 1; k--) {
                System.out.print(k);
            }
            System.out.println();
        }
```



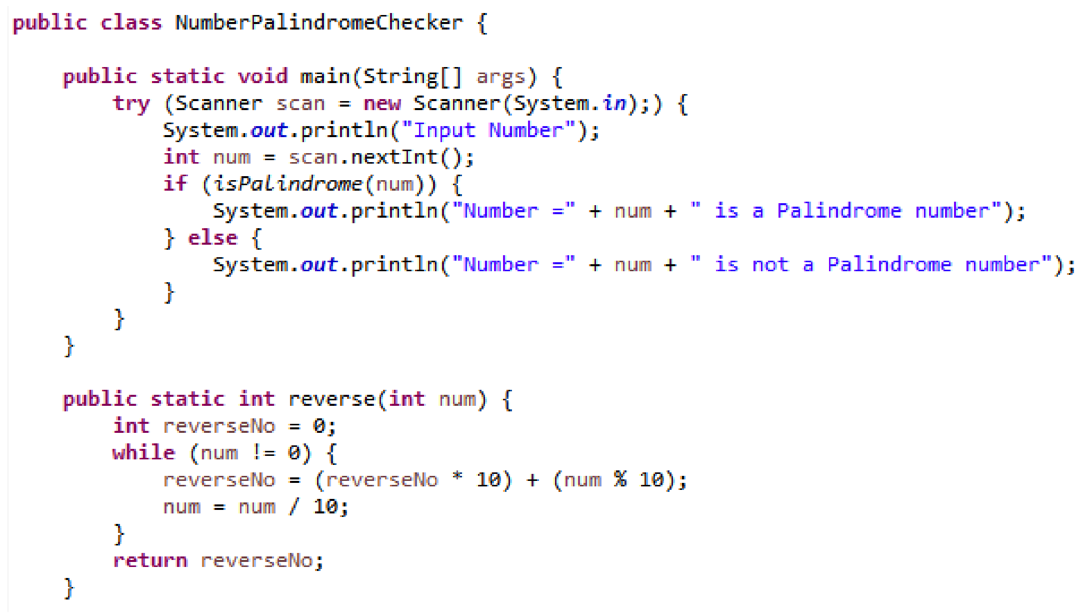
```
        // Fifth Pattern

        /*  4321
            432
            43
            4
        */
        for(int i = 1; i <= 4; i++) {
            for (int j = 4; j >= i; j--) {
                System.out.print(j);
            }
            System.out.println();
        }

    }

}
```

Given an array of numbers, arrange them in a way that yields the largest value. For example, if the given numbers are {54, 546, 548, 60}, the arrangement 6054854654 gives the largest value. And if the given numbers are {1, 34, 3, 98, 9, 76, 45, 4}, then the arrangement 998764543431 gives the largest value.



```java
//Given an array of numbers, program to arrange the numbers to form the largest number
public class GFG {
    // The main function that prints the
    // arrangement with the largest value.
    // The function accepts a vector of strings
    static void printLargest(Vector<String> arr) {
        Collections.sort(arr, new Comparator<String>() {
            // A comparison function which is used by sort() in printLargest()
            @Override
            public int compare(String X, String Y) {

                // first append Y at the end of X
                String XY = X + Y;
                // then append X at the end of Y
                String YX = Y + X;
                // Now see which of the two formed numbers is greater
                return XY.compareTo(YX) > 0 ? -1 : 1;
            }
        });
        Iterator it = arr.iterator();
        while (it.hasNext())
            System.out.print(it.next());

    }
```

Check whether given number is palindrome or not



```java
public class NumberPalindromeChecker {

    public static void main(String[] args) {
        try (Scanner scan = new Scanner(System.in);) {
            System.out.println("Input Number");
            int num = scan.nextInt();
            if (isPalindrome(num)) {
                System.out.println("Number =" + num + " is a Palindrome number");
            } else {
                System.out.println("Number =" + num + " is not a Palindrome number");
            }
        }
    }

    public static int reverse(int num) {
        int reverseNo = 0;
        while (num != 0) {
            reverseNo = (reverseNo * 10) + (num % 10);
            num = num / 10;
        }
        return reverseNo;
    }
    public static boolean isPalindrome(int num) {
        if (reverse(num) == num) {
            return true;
        }
        return false;
    }
}
```

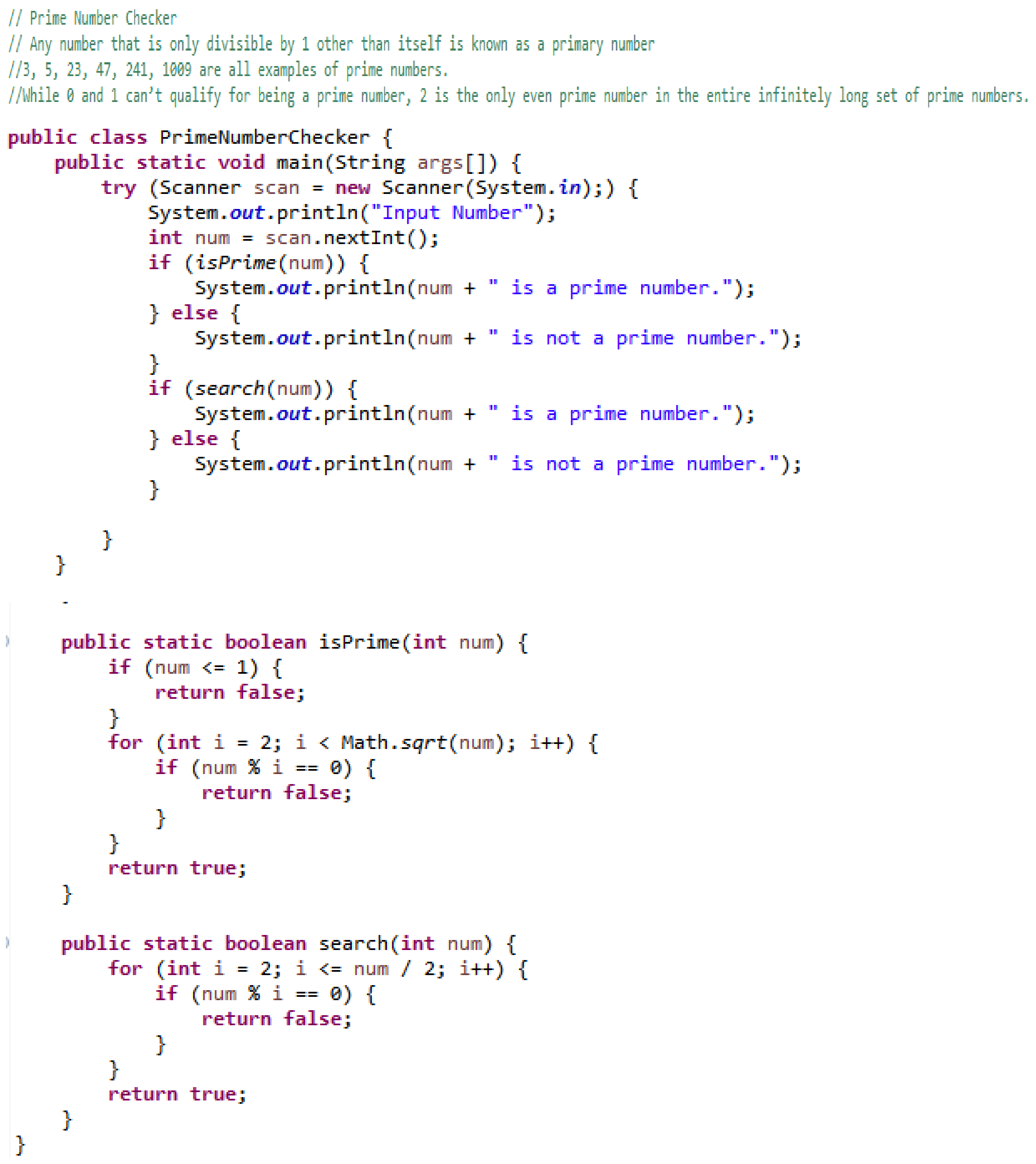
Check whether given string is palindrome or not

```
public class StringPalindromeChecker {
    public static void main(String[] args) {
        try (Scanner scan = new Scanner(System.in);) {
            System.out.println("Input String");
            String word = scan.next();
            if (isPalindrome(word)) {
                System.out.println("Word " + word + " is a Palindrome");
            } else {
                System.out.println("Word " + word + " is not a Palindrome");
            }
        }
    }

    public static boolean isPalindrome(String word) {
        boolean isPalindrome = false;
        String reverse = "";
        for (int i = word.length() - 1; i >= 0; i--) {
            reverse = reverse + word.charAt(i);
        }
        if (reverse.equals(word)) {
            isPalindrome = true;
        }

        return isPalindrome;

    }
}
```

Find longest word in a given sentence

```
public class LongestWordInAString {
    public static void main(String[] args) {
        String sentence="This is Praveen Oruganti I am Senior Technical Lead in Birlasoft";
        String[] words= sentence.split(" ");
        String largestWord= words[0];
        for(int i=1;i<words.length;i++) {
            if(words[i].length()>largestWord.length()) {
                largestWord=words[i];
            }
        }
        System.out.println("Longest word in Sentence \n"+sentence +" \nis \n"+largestWord);
    }
}
```

Check whether given number is prime number or not



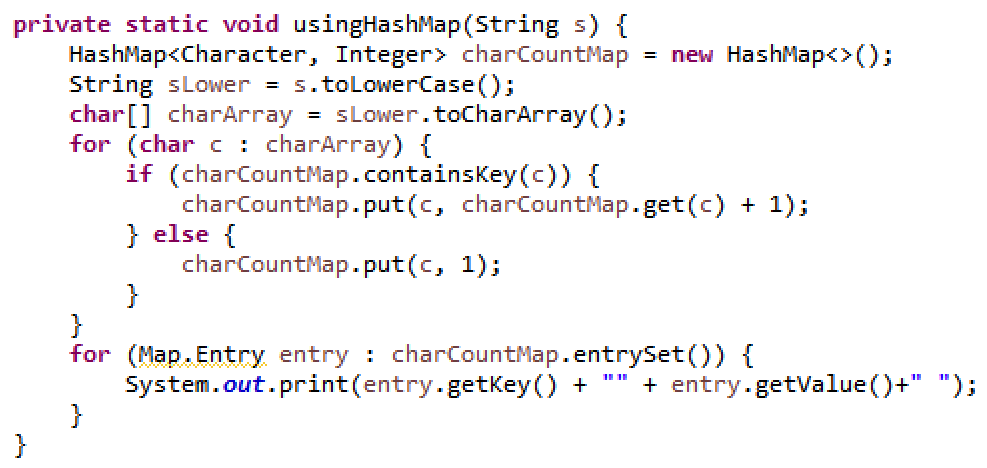
```java
// Prime Number Checker
// Any number that is only divisible by 1 other than itself is known as a primary number
//3, 5, 23, 47, 241, 1009 are all examples of prime numbers.
//While 0 and 1 can't qualify for being a prime number, 2 is the only even prime number in the entire infinitely long set of prime numbers.

public class PrimeNumberChecker {
    public static void main(String args[]) {
        try (Scanner scan = new Scanner(System.in);) {
            System.out.println("Input Number");
            int num = scan.nextInt();
            if (isPrime(num)) {
                System.out.println(num + " is a prime number.");
            } else {
                System.out.println(num + " is not a prime number.");
            }
            if (search(num)) {
                System.out.println(num + " is a prime number.");
            } else {
                System.out.println(num + " is not a prime number.");
            }

        }
    }

    public static boolean isPrime(int num) {
        if (num <= 1) {
            return false;
        }
        for (int i = 2; i < Math.sqrt(num); i++) {
            if (num % i == 0) {
                return false;
            }
        }
        return true;
    }

    public static boolean search(int num) {
        for (int i = 2; i <= num / 2; i++) {
            if (num % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

Swap two numbers

```java
public class SwapTwoNumbers {
    public static void main(String args[]) {
        swapUsingTemp(4, 5);
        swapWithoutTemp(4, 5);
    }

    public static void swapUsingTemp(int num1, int num2) {
        System.out.println("swapUsingTemp Method Entry");
        int temp = 0;
        System.out.println("Before Swapping :" + " num1 is " + num1 + " num2 is " + num2);
        temp = num1;
        num1 = num2;
        num2 = temp;
        System.out.println("After Swapping :" + " num1 is " + num1 + " num2 is " + num2);
        System.out.println("swapUsingTemp Method Exit");

    }

    public static void swapWithoutTemp(int num1, int num2) {
        System.out.println("swapWithoutTemp Method Entry");
        System.out.println("Before Swapping :" + " num1 is " + num1 + " num2 is " + num2);
        num1 = num1 + num2;
        num2 = num1 - num2;
        num1 = num1 - num2;
        System.out.println("After Swapping :" + " num1 is " + num1 + " num2 is " + num2);
        System.out.println("swapWithoutTemp Method Exit");

    }
}
```

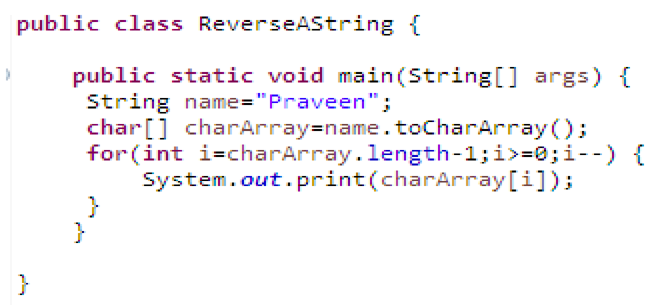
Program to find frequency of characters in a string

```java
private static void usingArrays(String s) {
    char ch;
    String sLower = s.toLowerCase();
    for (char c = 'A'; c <= 'z'; c++) {
        int count = 0;
        for (int j = 0; j < s.length(); j++) {
            ch = sLower.charAt(j);
            if (ch == c) {
                count++;
            }
        }
        if (count > 0) {
            System.out.print(c + "" + count+" ");
        }
    }
}
```



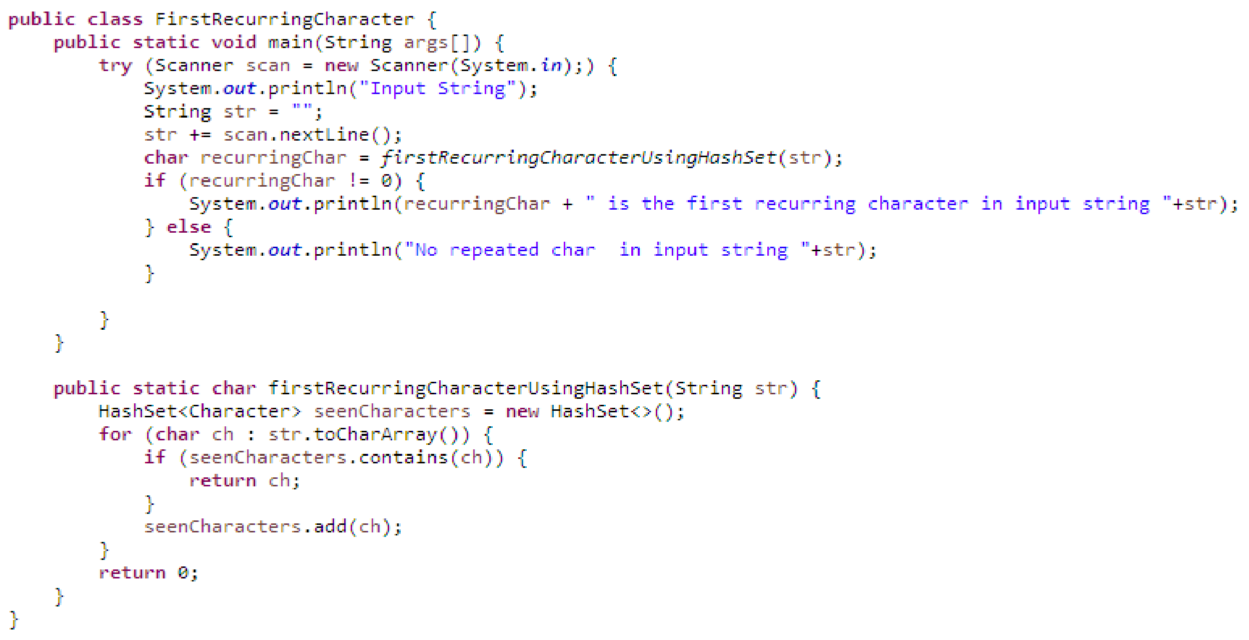
```java
private static void usingHashMap(String s) {
    HashMap<Character, Integer> charCountMap = new HashMap<>();
    String sLower = s.toLowerCase();
    char[] charArray = sLower.toCharArray();
    for (char c : charArray) {
        if (charCountMap.containsKey(c)) {
            charCountMap.put(c, charCountMap.get(c) + 1);
        } else {
            charCountMap.put(c, 1);
        }
    }
    for (Map.Entry entry : charCountMap.entrySet()) {
        System.out.print(entry.getKey() + "" + entry.getValue()+" ");
    }
}
```

How to reverse a string in java without using any API



```java
public class ReverseAString {

    public static void main(String[] args) {
     String name="Praveen";
     char[] charArray=name.toCharArray();
     for(int i=charArray.length-1;i>=0;i--) {
         System.out.print(charArray[i]);
     }
    }

}
```

How to find first recurring character in a given String



```java
public class FirstRecurringCharacter {
    public static void main(String args[]) {
        try (Scanner scan = new Scanner(System.in);) {
            System.out.println("Input String");
            String str = "";
            str += scan.nextLine();
            char recurringChar = firstRecurringCharacterUsingHashSet(str);
            if (recurringChar != 0) {
                System.out.println(recurringChar + " is the first recurring character in input string "+str);
            } else {
                System.out.println("No repeated char  in input string "+str);
            }

        }
    }

    public static char firstRecurringCharacterUsingHashSet(String str) {
        HashSet<Character> seenCharacters = new HashSet<>();
        for (char ch : str.toCharArray()) {
            if (seenCharacters.contains(ch)) {
                return ch;
            }
            seenCharacters.add(ch);
        }
        return 0;
    }
}
```

How to check if two strings are Anagram

An anagram is a word which is formed by rearranging or shuffling of letters in another word, the most important property

In Anagram is that all the letters have to be used only once. For example, abcd and dcba (or) abcd and cabd (or) abcd and abdc are Anagrams

Example like LISTEN and SILENT (or) TRAINGLE and INTEGRAL are Anagrams

```java
static boolean isAnagram(String a, String b) {
    if (a.length() != b.length()) {
        return false;
    } else {
        for (int i = 0; i < a.length(); i++) {
            char ch = a.toLowerCase().charAt(i);
            b = b.toLowerCase();
            if (b.indexOf(ch) != -1) {
                b = b.replaceFirst(ch + "", "");
            } else {
                return false;
            }
        }
        return b.length() == 0;
    }
}
```

Program to find max and min values in an Array

```java
public class ArrayMaxMin {
    public static void main(String[] args) {
        int[] num = { 90, 24, 46, 35, 32, 12, 98, 2 };
        int max = num[0];
        int min = num[1];

        for (int i = 0; i < num.length; i++) {
            if (num[i] > max) {
                max = num[i];
            } else if (num[i] < min) {
                min = num[i];
            }
        }

        System.out.println("Max Element is " + max);
        System.out.println("Min Element is " + min);
    }
}
```

You can view the complete code in my repository (https://github.com/praveenoruganti/praveen-java-datastructure-algorithm)