C-DAC Mumbai

Date 25/09/2024

Subject: Algorithm and Data Structure
Assignment 1

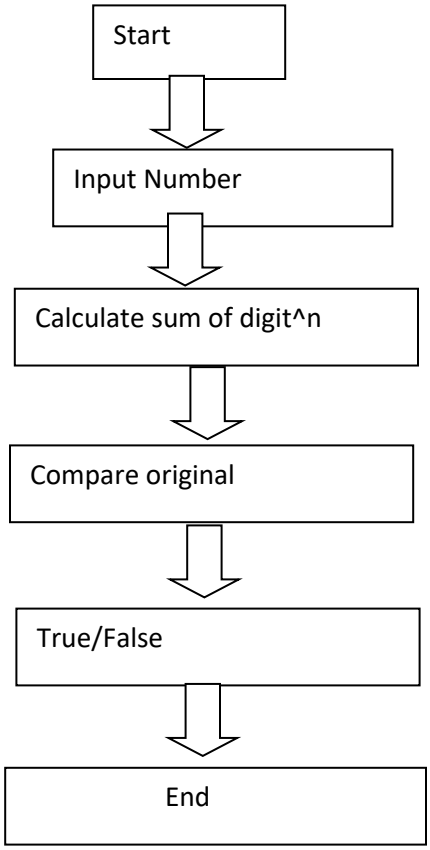
Solve the assignment with following thing to be added in each question.

-Program
-Flow chart
-Explanation
-Output
-Time and Space complexity

1. Armstrong Number
Problem: Write a Java program to check if a given number is an Armstrong number.

Test Cases:

Input: 153
Output: true
Input: 123
Output: false

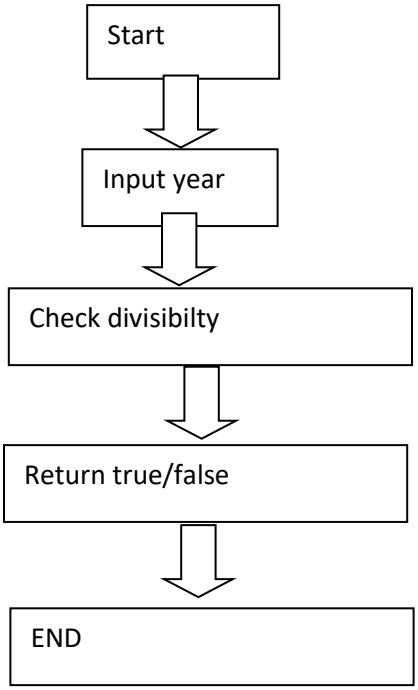


```
public class ArmstrongNumber {
      public static boolean isArmstrong(int number) {
            int original = number;
            int sum = 0;
   int digits = String.valueOf(number).length();

   while (number != 0) {
       int digit = number % 10;
        sum += Math.pow(digit, digits);
         number /= 10;
   }
   return sum == original;
}

public static void main(String[] args) {
   System.out.println(isArmstrong(153));  // Output: true
   System.out.println(isArmstrong(123));  // Output: false
}
}
```

2. Prime Number
Problem: Write a Java program to check if a given number is prime.

Test Cases:

Input: 29
Output: true
Input: 15
Output: false




```
import java.util.Scanner;

public class PrimeNumber {
    public static boolean isPrime(int number) {
        if (number <= 1) {
            return false;
        }


        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) {
                return false;
            }
        }

        return true;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = sc.nextInt();
        System.out.println(isPrime(number));

        sc.close();
    }
```

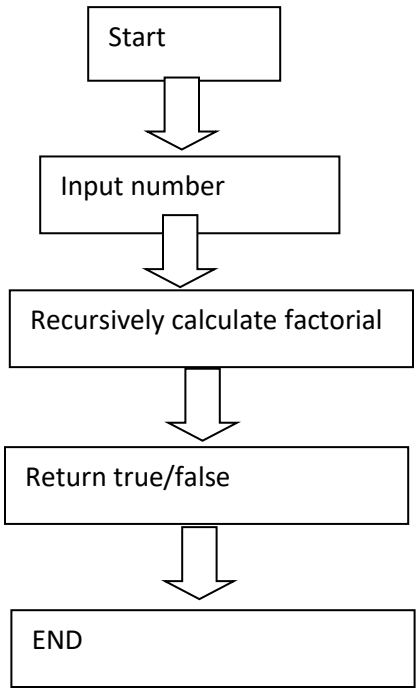
}

```
C:\Users\RISHI\Downloads\ADS>java PrimeNumber
Enter a number: 29
true

C:\Users\RISHI\Downloads\ADS>java PrimeNumber
Enter a number: 15
false

C:\Users\RISHI\Downloads\ADS>
```

3. Factorial
Problem: Write a Java program to compute the factorial of a given number.

Test Cases:

Input: 5
Output: 120
Input: 0
Output: 1



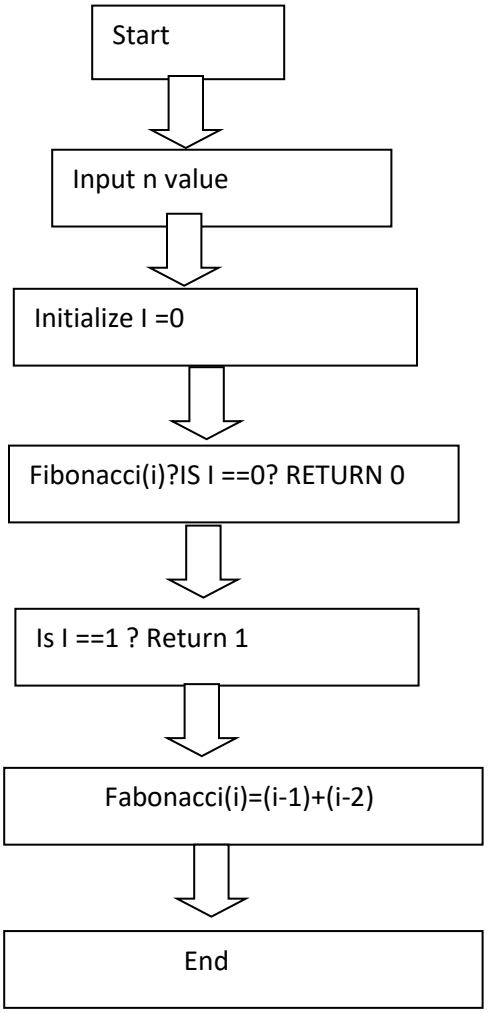
```
import java.util.Scanner;
public class Factorial{
   public static int factorial(int number){
      int result = 1;
      for(int i =2;i<=number;i++){
         result = result *i;
      }
      return result;
   }
   public static void main(String [] args){
      Scanner sc = new Scanner(System.in);

   System.out.print("Enter a number:");
   int number = sc.nextInt();
   System.out.println(factorial(number));

sc.close();
  }
}
```

```
PS C:\Users\RISHI\Downloads\ADS> javac Factorial.java
PS C:\Users\RISHI\Downloads\ADS> java Factorial
Enter a number:5
120
PS C:\Users\RISHI\Downloads\ADS> java Factorial
Enter a number:0
1
PS C:\Users\RISHI\Downloads\ADS>
```

4. Fibonacci Series
Problem: Write a Java program to print the first n numbers in the Fibonacci series.

Test Cases:

Input: n = 5
Output: [0, 1, 1, 2, 3]
Input: n = 8
Output: [0, 1, 1, 2, 3, 5, 8, 13]




```
import java.util.*;
public class FibonacciSeriesR {
   public static void main(String []args){
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter the value of n:");
      int n = sc.nextInt();
      for(int i =0;i<n;i++){
         System.out.println(fibonacci(i)+"");
      }
      sc.close();
      }
      public static int fibonacci(int n){
         if (n<=1){
            return n;
         }else{
```

```
        return fibonacci(n-1)+fibonacci(n-2);
    }
}
```

```
PS C:\Users\RISHI\Downloads\ADS> java FibonacciSeriesR
Enter the value of n:
5
0
1
1
2
3
PS C:\Users\RISHI\Downloads\ADS> java FibonacciSeriesR
Enter the value of n:
8
0
1
1
2
3
5
8
13
```

5. Find GCD

Problem: Write a Java program to find the Greatest Common Divisor (GCD) of two numbers.

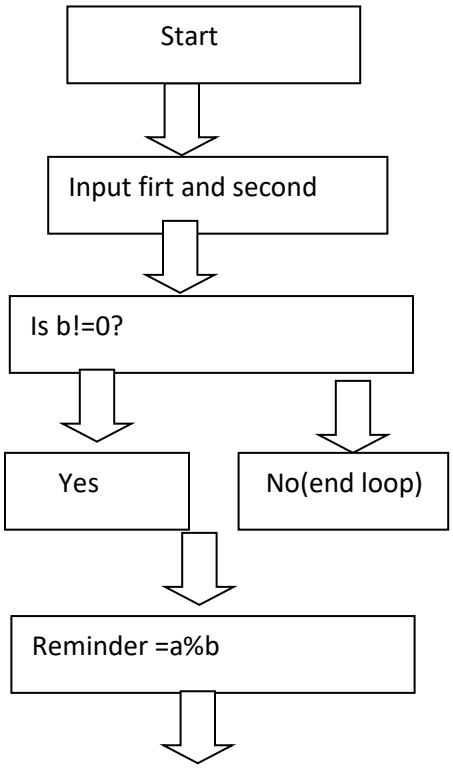
Test Cases:

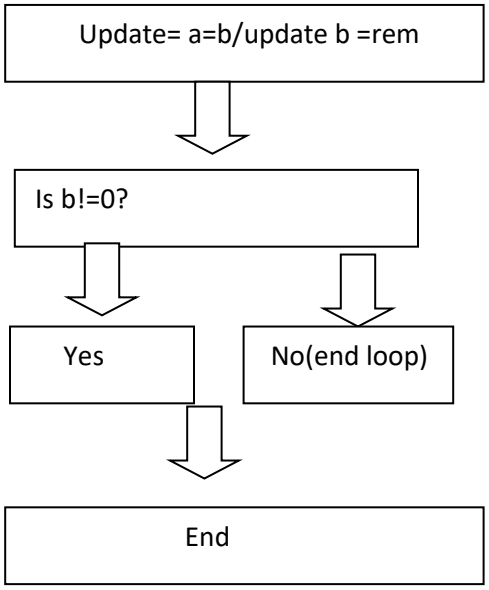
Input: a = 54, b = 24

Output: 6

Input: a = 17, b = 13

Output: 1



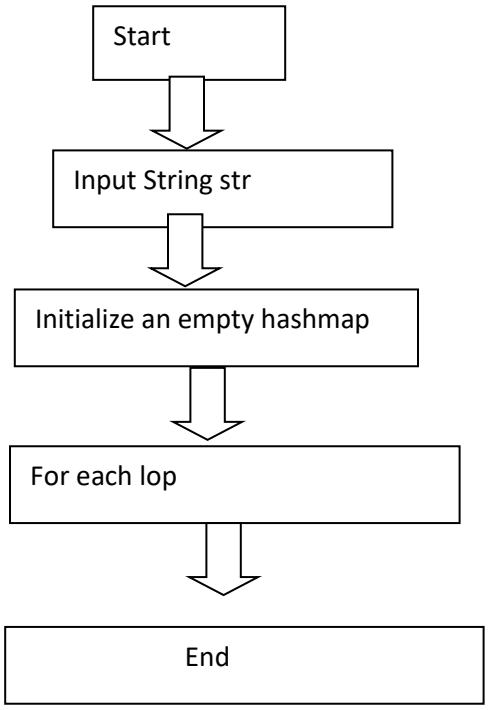



```
import java.util.*;
public class Gcd {
   public static void main(String args[]){
      Scanner sc =new Scanner(System.in);
      System.out.println("Enter the first number");
      int a = sc.nextInt();
      System.out.print("Enter the second number");
      int b = sc.nextInt();
      int gcd = findGcd(a,b);
      System.out.println("The GCD of " + a + " and " + b + " is: " + gcd );

      }
      public static int findGcd(int a,int b){
         while (b != 0){
            int temp = b;
            b = a%b;
            a =temp;
         }
         return a;
      }
   }
```

```
PS C:\Users\RISHI\Downloads\ADS> javac Gcd.java
PS C:\Users\RISHI\Downloads\ADS> java Gcd
Enter the first number
54
Enter the second number24
The GCD of 54 and 24 is: 6
PS C:\Users\RISHI\Downloads\ADS> java Gcd
Enter the first number
17
Enter the second number13
The GCD of 17 and 13 is: 1
PS C:\Users\RISHI\Downloads\ADS>
```

7. Find Repeated Characters in a String
Problem: Write a Java program to find all repeated characters in a string.

Test Cases:
Input: "programming"
Output: ['r', 'g', 'm']
Input: "hello"
Output: ['l']




```
import java.util.*;
public class RepeatedCharacters {
    public static List<Character> findRepeatedChars(String str) {
```

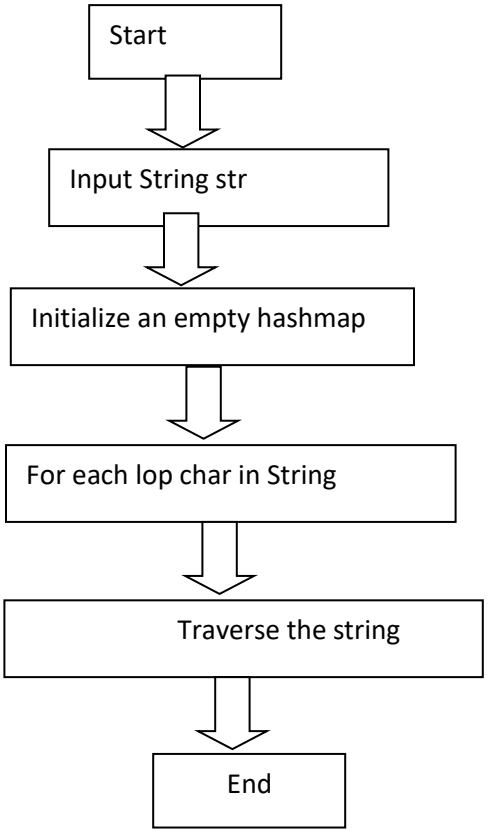
```
        Map<Character, Integer> charCount = new HashMap<>();
        List<Character> repeatedChars = new ArrayList<>();

        // Count occurrences of each character
        for (char c : str.toCharArray()) {
            charCount.put(c, charCount.getOrDefault(c, 0) + 1);
        }

        // Find characters with more than one occurrence
        for (Map.Entry<Character, Integer> entry : charCount.entrySet()) {
            if (entry.getValue() > 1) {
                repeatedChars.add(entry.getKey());
            }
        }

        return repeatedChars;
    }

    public static void main(String[] args) {
        System.out.println(findRepeatedChars("programming"));
        System.out.println(findRepeatedChars("hello"));
    }
}
```

```
PS C:\Users\RISHI\Downloads\ADS> javac RepeatedCharacters.java
PS C:\Users\RISHI\Downloads\ADS> java RepeatedCharacters
[r, g, m]
[l]
PS C:\Users\RISHI\Downloads\ADS>
```

8. First Non-Repeated Character
Problem: Write a Java program to find the first non-repeated character in a string.

Test Cases:

Input: "stress"
Output: 't'
Input: "aabbcc"
Output: null




```
import java.util.*;
public class Nonrepeated {
   public static Character findFirstNonRepeatedChar(String str) {
      Map<Character, Integer> charCount = new LinkedHashMap<>();

      for (char c : str.toCharArray()) {
         charCount.put(c, charCount.getOrDefault(c, 0) + 1);
      }

      for (Map.Entry<Character, Integer> entry : charCount.entrySet()) {
         if (entry.getValue() == 1) {
            return entry.getKey();
         }
      }

      return null;
   }
```

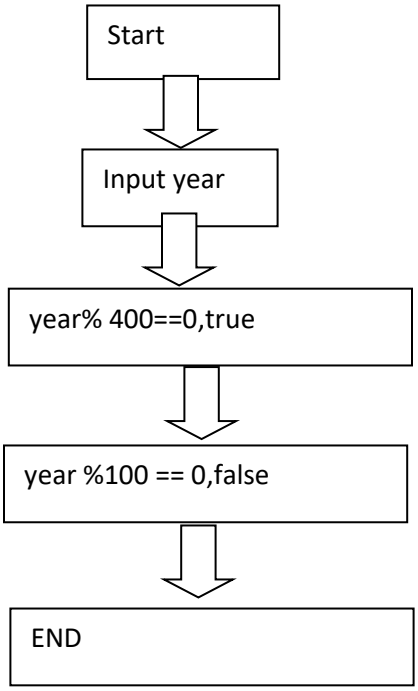
```
    public static void main(String[] args) {
        System.out.println(findFirstNonRepeatedChar("stress"));
        System.out.println(findFirstNonRepeatedChar("aabbcc"));
    }
}
```

```
PS C:\Users\RISHI\Downloads\ADS> javac Nonrepeated.java
PS C:\Users\RISHI\Downloads\ADS> java Nonrepeated
t
null
PS C:\Users\RISHI\Downloads\ADS>
```

10. Leap Year
Problem: Write a Java program to check if a given year is a leap year.

Test Cases:

Input: 2020
Output: true
Input: 1900
Output: false



```
public class Leapyear {
    public static boolean isLeapyear(int year){
        if(year% 400==0){
            return true;
        }else if(year %100 == 0){
            return false;
        }else{
            return year% 4 ==0;
        }
}
public static void main(String[] args){
    System.out.println(isLeapyear(2020));
    System.out.println(isLeapyear(1900));
}
}
```