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
Ans:

```
public class Armstrong {
   // Method to calculate the power of a number
   public static int power(int base, int exp) {
      int result = 1;
      for (int i = 0; i < exp; i++) {
         result *= base;
      }
      return result;
   }

   // Method to check if a number is an Armstrong number
   public static boolean isArmstrong(int number) {
      int originalNumber = number;
      int sum = 0;
      int digits = 0;

      // Calculate the number of digits
      while (originalNumber != 0) {
         originalNumber /= 10;
         digits++;
      }

      originalNumber = number;
```

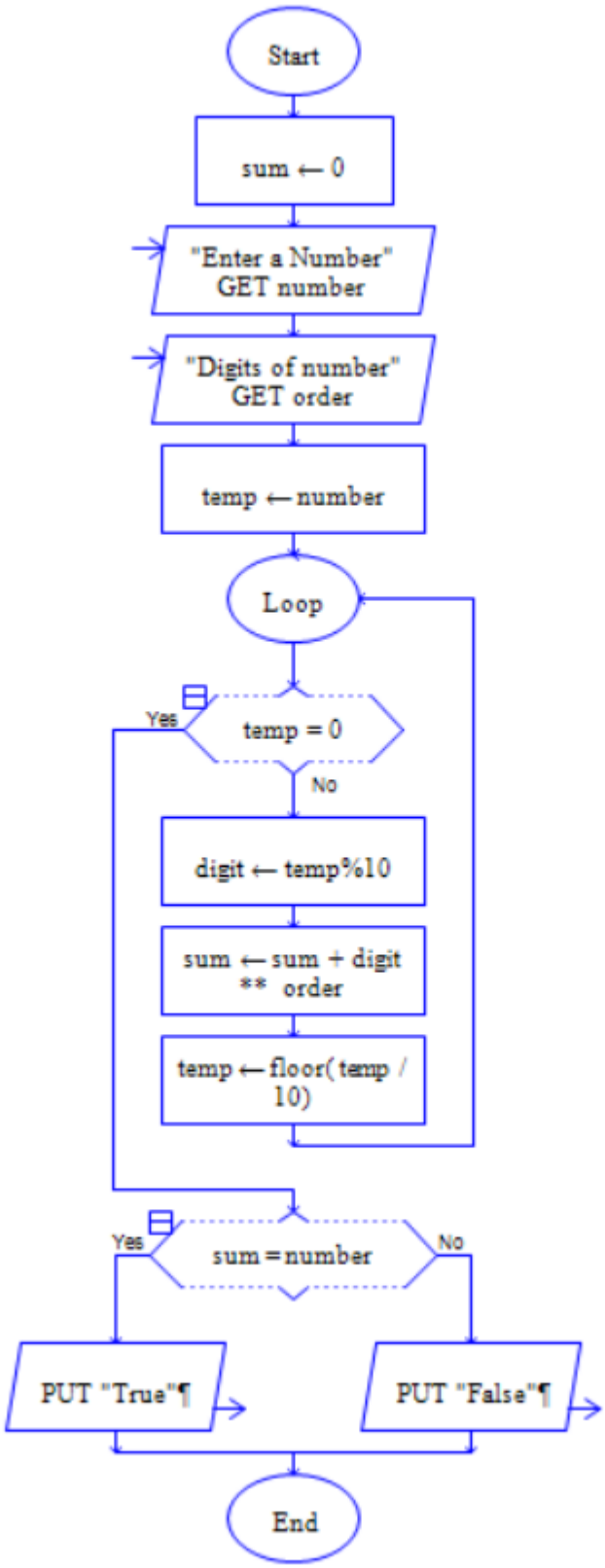
```
        // Calculate the sum of the digits raised to the power of the number of digits
        while (originalNumber != 0) {
            int remainder = originalNumber % 10;
            sum += power(remainder, digits);
            originalNumber /= 10;
        }

        // Check if the sum is equal to the original number
        return sum == number;
    }

    public static void main(String[] args) {
        int number1 = 153;
        int number2 = 123;

        System.out.println(number1 + " is an Armstrong number: " + isArmstrong(number1));
        System.out.println(number2 + " is an Armstrong number: " + isArmstrong(number2));
    }
}
```

Armstrong.java

Armstrong.java > ...

```
1   public class Armstrong {
2       // Method to calculate the power of a number
3       public static int power(int base, int exp) {
4           int result = 1;
5           for (int i = 0; i < exp; i++) {
6               result *= base;
7           }
8           return result;
9       }
10
11      // Method to check if a number is an Armstrong number
12      public static boolean isArmstrong(int number) {
13          int originalNumber = number;
14          int sum = 0;
15          int digits = 0;
16
17          // Calculate the number of digits
18          while (originalNumber != 0) {
19              originalNumber /= 10;
20              digits++;
21          }
22
23          originalNumber = number;
24
25          // Calculate the sum of the digits raised to the power of the number of digits
26          while (originalNumber != 0) {
27              int remainder = originalNumber % 10;
28              sum += power(remainder, digits);
29              originalNumber /= 10;
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1> javac Armstrong.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java Armstrong
153 is an Armstrong number: true
123 is an Armstrong number: false
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```


Start
sum ← 0
"Enter a Number"
GET number
"Digits of number"
GET order
temp ← number
Loop
Yes
temp = 0
No
digit ← temp%10
sum ← sum + digit
** order
temp ← floor(temp /
10)
Yes
sum = number
No
PUT "True"¶
PUT "False"¶
End

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
Ans:

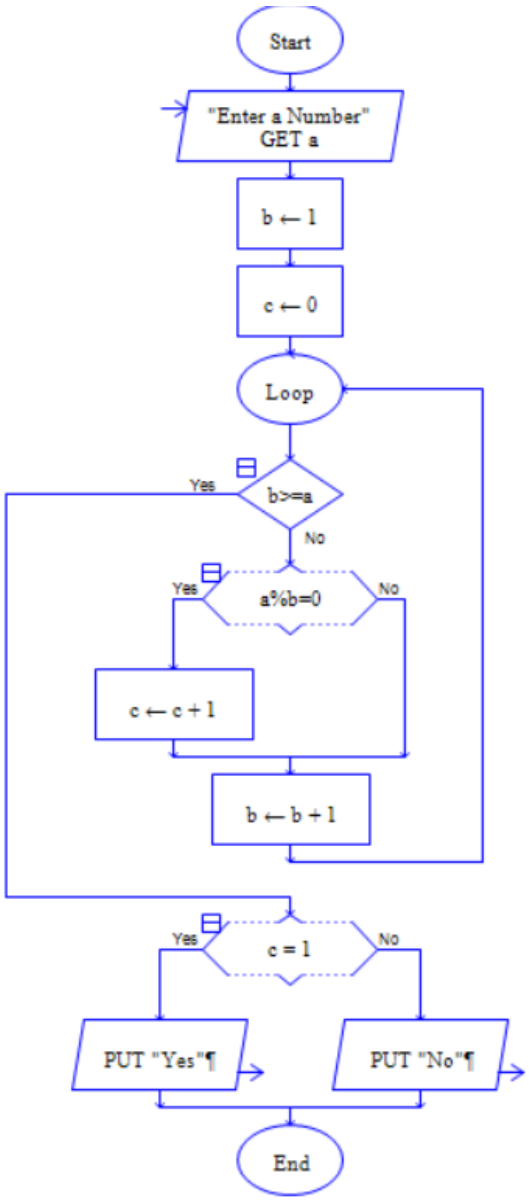
```
public class PrimeNumber {
   // Method to check if a number is prime
   public static boolean isPrime(int number) {
      // Corner case
      if (number <= 1) {
         return false;
      }
      // Check from 2 to the square root of the number
      for (int i = 2; i <= Math.sqrt(number); i++) {
         if (number % i == 0) {
            return false;
         }
      }
      return true;
   }

   public static void main(String[] args) {
      int number1 = 29;
      int number2 = 15;

      System.out.println(number1 + " is a prime number: " + isPrime(number1));
      System.out.println(number2 + " is a prime number: " + isPrime(number2));
   }
}
```

Armstrong.java | PrimeNumber.java

PrimeNumber.java > PrimeNumber > main(String[])

```java
public class PrimeNumber {
    // Method to check if a number is prime
    public static boolean isPrime(int number) {
        // Corner case
        if (number <= 1) {
            return false;
        }
        // Check from 2 to the square root of the number
        for (int i = 2; i <= Math.sqrt(number); i++) {
            if (number % i == 0) {
                return false;
            }
        }
        return true;
    }

    Run | Debug
    public static void main(String[] args) {
        int number1 = 29;
        int number2 = 15;

        System.out.println(number1 + " is a prime number: " + isPrime(number1));
        System.out.println(number2 + " is a prime number: " + isPrime(number2));
    }
}
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac PrimeNumber.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java PrimeNumber
29 is a prime number: true
15 is a prime number: false
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```



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

Ans:

```
import java.util.Scanner;

public class Factorial {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int num = scanner.nextInt();
      scanner.close();

      long factorial = calculateFactorial(num);
      System.out.println("Factorial of " + num + " is: " + factorial);
   }

   public static long calculateFactorial(int num) {
      if (num == 0) {
         return 1;
      }
      long result = 1;
      for (int i = 1; i <= num; i++) {
         result *= i;
      }
      return result;
   }
```

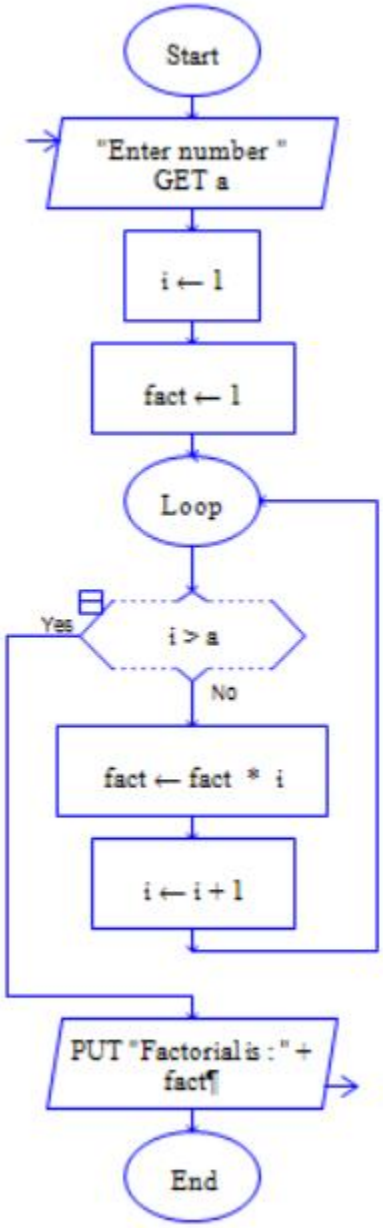
}

Armstrong.java | Factorial.java | PrimeNumber.java

Factorial.java > Factorial > calculateFactorial(int)

```java
import java.util.Scanner;

public class Factorial {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print(s:"Enter a number: ");
        int num = scanner.nextInt();
        scanner.close();

        long factorial = calculateFactorial(num);
        System.out.println("Factorial of " + num + " is: " + factorial);
    }

    public static long calculateFactorial(int num) {
        if (num == 0) {
            return 1;
        }
        long result = 1;
        for (int i = 1; i <= num; i++) {
            result *= i;
        }
        return result;
    }
}
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac Factorial.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java Factorial
Enter a number: 100
Factorial of 100 is: 0
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```


Start
"Enter number "
GET a
i ← 1
fact ← 1
Loop
Yes
i > a
No
fact ← fact * i
i ← i + 1
PUT "Factorial is : " +
fact¶
End

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]
Ans:

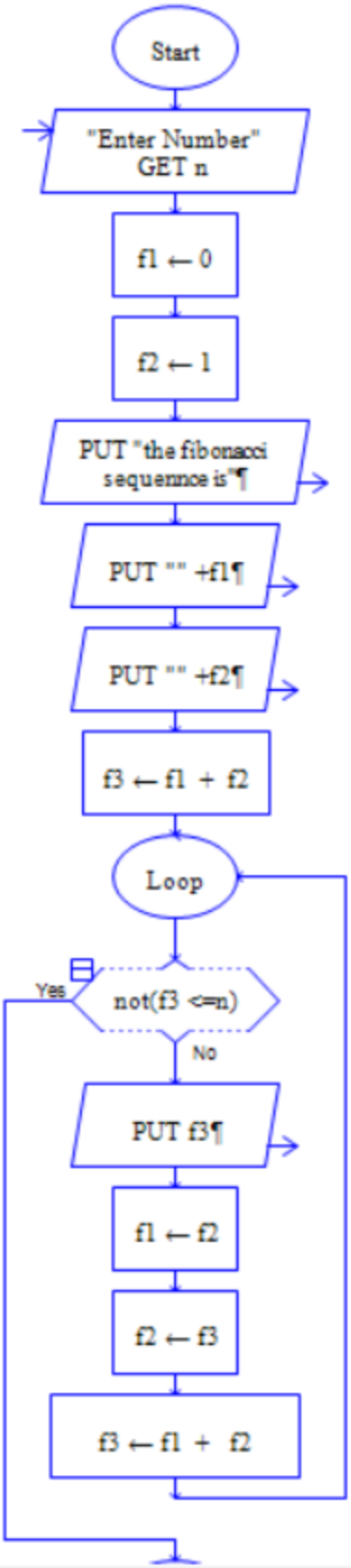
```
import java.util.Scanner;

public class Fibonacci {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the number of terms: ");
      int n = scanner.nextInt();
      scanner.close();

      printFibonacciSeries(n);
   }

   public static void printFibonacciSeries(int n) {
      int firstTerm = 0, secondTerm = 1;
      System.out.print("Fibonacci Series: ");

      for (int i = 1; i <= n; ++i) {
         System.out.print(firstTerm + " ");

         // Compute the next term
         int nextTerm = firstTerm + secondTerm;
         firstTerm = secondTerm;
         secondTerm = nextTerm;
      }
   }
}
```

Armstrong.java | Factorial.java | Fibonacci.java | PrimeNumber.java

Fibonacci.java > Fibonacci > printFibonacciSeries(int)

```java
import java.util.Scanner;

public class Fibonacci {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print(s:"Enter the number of terms: ");
        int n = scanner.nextInt();
        scanner.close();

        printFibonacciSeries(n);
    }

    public static void printFibonacciSeries(int n) {
        int firstTerm = 0, secondTerm = 1;
        System.out.print(s:"Fibonacci Series: ");

        for (int i = 1; i <= n; ++i) {
            System.out.print(firstTerm + " ");

            // Compute the next term
            int nextTerm = firstTerm + secondTerm;
            firstTerm = secondTerm;
            secondTerm = nextTerm;
        }
    }
}
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1> javac Fibonacci.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java Fibonacci
Enter the number of terms: 25
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 4
6368
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```


Start
"Enter Number"
GET n
f1 ← 0
f2 ← 1
PUT "the fibonacci sequennce is"¶
PUT "" +f1¶
PUT "" +f2¶
f3 ← f1 + f2
Loop
Yes
not(f3 <=n)
No
PUT f3¶
f1 ← f2
f2 ← f3
f3 ← f1 + f2

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
Ans:

```
public class GCD {
   public static void main(String[] args) {
      int a = 54;
      int b = 24;
      System.out.println("GCD of " + a + " and " + b + " is " + findGCD(a, b));

      a = 17;
      b = 13;
      System.out.println("GCD of " + a + " and " + b + " is " + findGCD(a, b));
   }

   public static int findGCD(int a, int b) {
      if (b == 0) {
         return a;
      }
      return findGCD(b, a % b);
   }
```

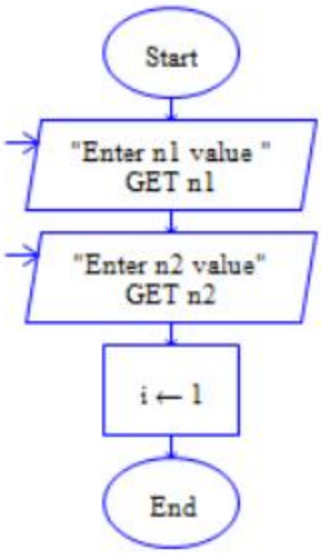
}

Armstrong.java | Factorial.java | Fibonacci.java | Exception.java 3 | PrimeNumber.java | GCD.java

GCD.java > ...

```java
public class GCD {
    Run | Debug
    public static void main(String[] args) {
        int a = 54;
        int b = 24;
        System.out.println("GCD of " + a + " and " + b + " is " + findGCD(a, b));

        a = 17;
        b = 13;
        System.out.println("GCD of " + a + " and " + b + " is " + findGCD(a, b));
    }

    public static int findGCD(int a, int b) {
        if (b == 0) {
            return a;
        }
        return findGCD(b, a % b);
    }
}
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac GCD.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java GCD
GCD of 54 and 24 is 6
GCD of 17 and 13 is 1
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```



6. Find Square Root
Problem: Write a Java program to find the square root of a given number (using integer approximation).

Test Cases:

Input: x = 16
Output: 4
Input: x = 27
Output: 5
Ans:

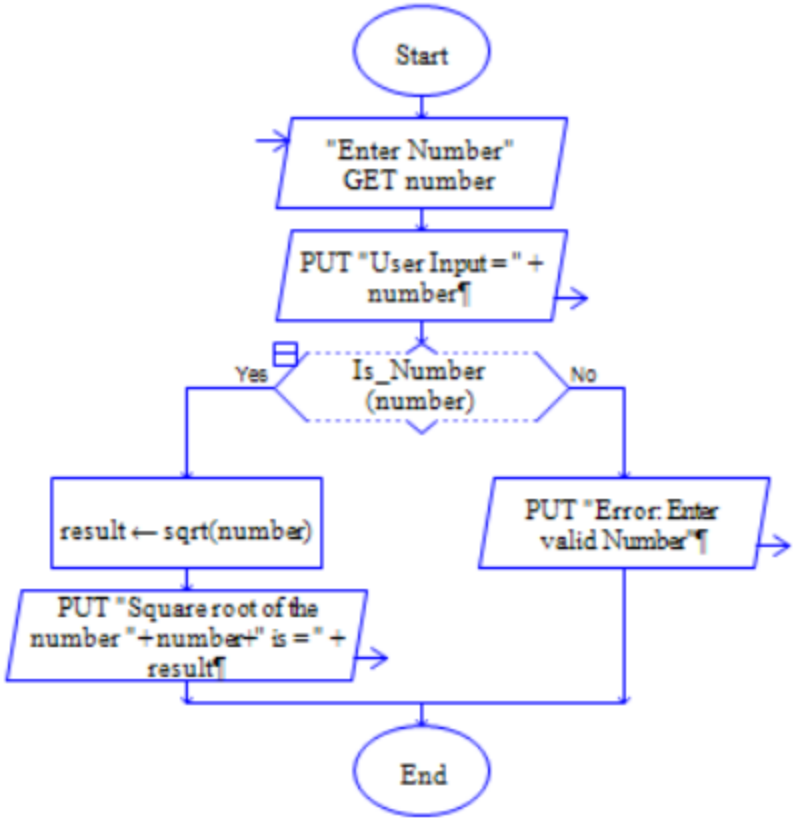
```
import java.util.Scanner;

public class SquareRoot {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int x = scanner.nextInt();
      System.out.println("The integer approximation of the square root of " + x + " is: " +
integerSquareRoot(x));
   }

   public static int integerSquareRoot(int x) {
      if (x == 0 || x == 1) {
         return x;
      }
      int start = 1, end = x, result = 0;
      while (start <= end) {
         int mid = (start + end) / 2;
```

```
        if (mid * mid == x) {
            return mid;
        }
        if (mid * mid < x) {
            start = mid + 1;
            result = mid;
        } else {
            end = mid - 1;
        }
    }
    return result;
  }
}
```

Factorial.java | Fibonacci.java | Exception.java 3 | PrimeNumber.java | GCD.java | SquareRoot.java 1

SquareRoot.java > ...

```java
import java.util.Scanner;

public class SquareRoot {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print(s:"Enter a number: ");
        int x = scanner.nextInt();
        System.out.println("The integer approximation of the square root of " + x + " is: " + integerSquar
    }

    public static int integerSquareRoot(int x) {
        if (x == 0 || x == 1) {
            return x;
        }
        int start = 1, end = x, result = 0;
        while (start <= end) {
            int mid = (start + end) / 2;
            if (mid * mid == x) {
                return mid;
            }
            if (mid * mid < x) {
                start = mid + 1;
                result = mid;
            } else {
                end = mid - 1;
            }
        }
        return result;
    }
```



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']
Ans:

```
import java.util.HashMap;
import java.util.Map;
import java.util.ArrayList;
import java.util.List;

public class RepeatedString {
   public static List<Character> findRepeatedCharacters(String str) {
      Map<Character, Integer> charCountMap = new HashMap<>();
      List<Character> repeatedChars = new ArrayList<>();
```

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

        // Collect characters that appear more than once
        for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
            if (entry.getValue() > 1) {
                repeatedChars.add(entry.getKey());
            }
        }

        return repeatedChars;
    }

    public static void main(String[] args) {
        String input1 = "programming";
        String input2 = "hello";

        System.out.println("Input: " + input1 + " Output: " + findRepeatedCharacters(input1));
        System.out.println("Input: " + input2 + " Output: " + findRepeatedCharacters(input2));
    }
}
```

bonacci.java | Exception.java 3 | PrimeNumber.java | GCD.java | SquareRoot.java 1 | RepeatedString.java

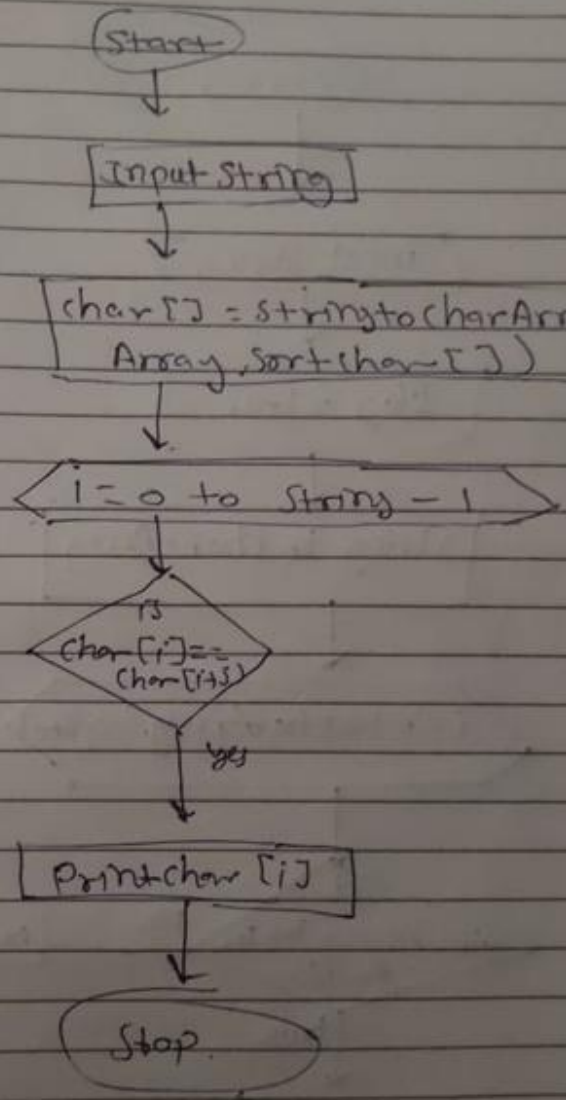
RepeatedString.java > RepeatedString > findRepeatedCharacters(String)

```java
public class RepeatedString {
    public static List<Character> findRepeatedCharacters(String str) {
        Map<Character, Integer> charCountMap = new HashMap<>();
        List<Character> repeatedChars = new ArrayList<>();

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

        // Collect characters that appear more than once
        for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
            if (entry.getValue() > 1) {
                repeatedChars.add(entry.getKey());
            }
        }

        return repeatedChars;
    }

    Run | Debug
    public static void main(String[] args) {
        String input1 = "programming";
        String input2 = "hello";

        System.out.println("Input: " + input1 + " Output: " + findRepeatedCharacters(input1));
        System.out.println("Input: " + input2 + " Output: " + findRepeatedCharacters(input2));
    }
}
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac RepeatedString.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java RepeatedString
Input: programming Output: [r, g, m]
Input: hello Output: [l]
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```

(Q7) Find Repeated character in string



Steps :

1 → input String
2 → Convert to char array
3 → Sort array
4 → if charAt(i) == charA(i+1)
5 → print charAt(i)

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
Ans:

```
import java.util.LinkedHashMap;
import java.util.Map;

public class NonRepeatedCharacter {
   public static Character findFirstNonRepeatedCharacter(String str) {
      Map<Character, Integer> charCountMap = new LinkedHashMap<>();

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

      // Find the first character with a count of 1
      for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
         if (entry.getValue() == 1) {
            return entry.getKey();
         }
      }

      return null; // Return null if no non-repeated character is found
   }

   public static void main(String[] args) {
      String input1 = "stress";
      String input2 = "aabbcc";

      System.out.println("Input: " + input1 + " Output: " + findFirstNonRepeatedCharacter(input1));
      System.out.println("Input: " + input2 + " Output: " + findFirstNonRepeatedCharacter(input2));
   }
```

}

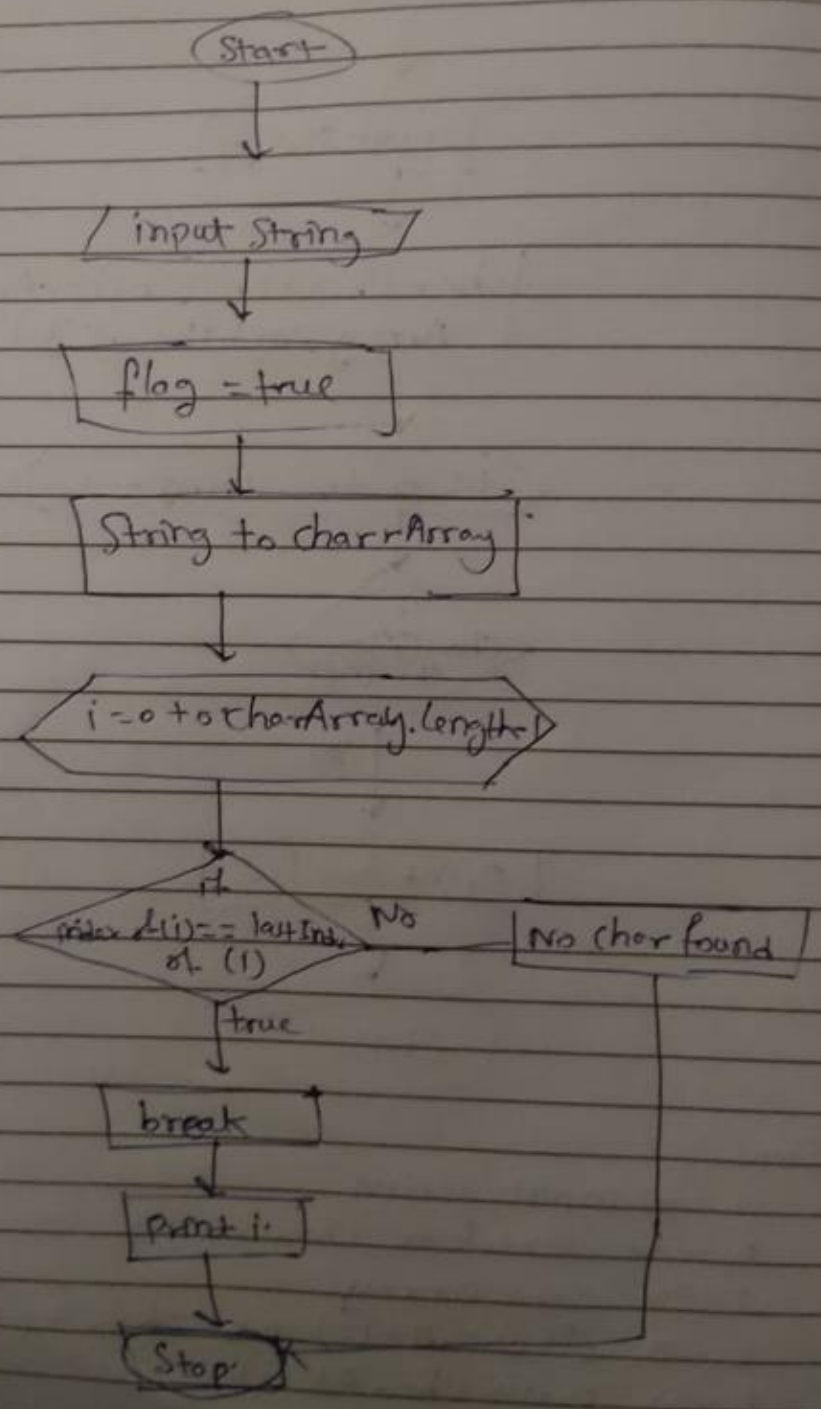
PrimeNumber.java | GCD.java | SquareRoot.java 1 | RepeatedString.java | NonRepeatedCharacter.java

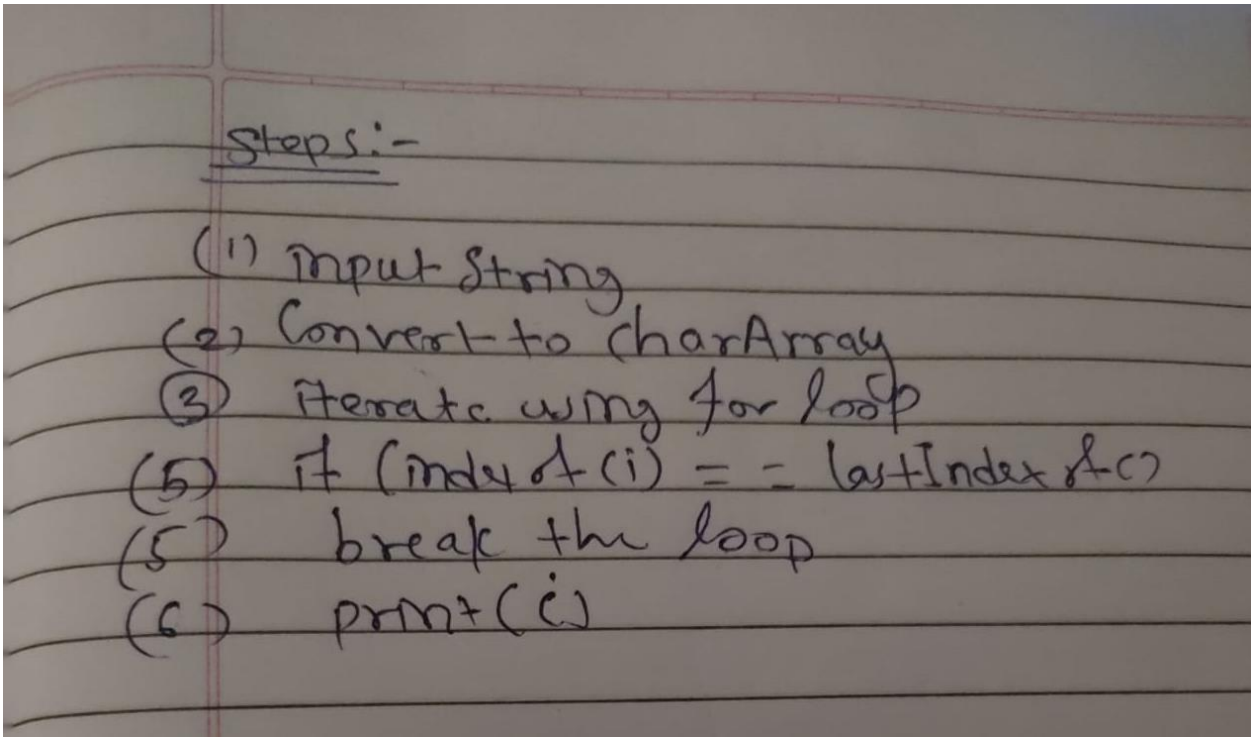
NonRepeatedCharacter.java > NonRepeatedCharacter > findFirstNonRepeatedCharacter(String)

```java
import java.util.LinkedHashMap;
import java.util.Map;

public class NonRepeatedCharacter {
    public static Character findFirstNonRepeatedCharacter(String str) {
        Map<Character, Integer> charCountMap = new LinkedHashMap<>();

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

        // Find the first character with a count of 1
        for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
            if (entry.getValue() == 1) {
                return entry.getKey();
            }
        }

        return null; // Return null if no non-repeated character is found
    }

    Run | Debug
    public static void main(String[] args) {
        String input1 = "stress";
        String input2 = "aabbcc";

        System.out.println("Input: " + input1 + " Output: " + findFirstNonRepeatedCharacter(input1));
        System.out.println("Input: " + input2 + " Output: " + findFirstNonRepeatedCharacter(input2));
    }
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac NonRepeatedCharacter.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java NonRepeatedCharacter
Input: stress Output: t
Input: aabbcc Output: null
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```

(Q8) First Non-Repeatative character to String





9. Integer Palindrome
Problem: Write a Java program to check if a given integer is a palindrome.

Test Cases:

Input: 121
Output: true
Input: -121
Output: false
Ans:

```
public class IntegerPalindrome {
   public static boolean isPalindrome(int number) {
      // Negative numbers are not palindromes
      if (number < 0) {
         return false;
      }

      int originalNumber = number;
      int reversedNumber = 0;

      // Reverse the number
      while (number != 0) {
         int digit = number % 10;
         reversedNumber = reversedNumber * 10 + digit;
         number /= 10;
      }
```

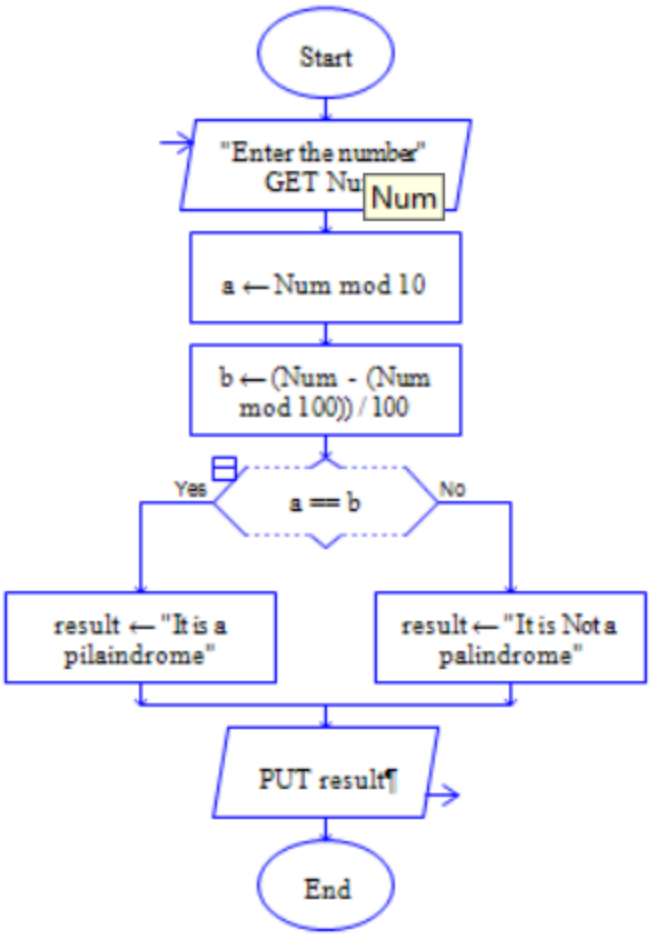
```
    // Check if the original number is equal to the reversed number
    return originalNumber == reversedNumber;
  }

  public static void main(String[] args) {
    int input1 = 121;
    int input2 = -121;

    System.out.println("Input: " + input1 + " Output: " + isPalindrome(input1));
    System.out.println("Input: " + input2 + " Output: " + isPalindrome(input2));
  }
}
```

ava | GCD.java | SquareRoot.java 1 | RepeatedString.java | NonRepeatedCharacter.java | Palindrome.java

Palindrome.java > Palindrome

```
1   public class Palindrome {
2       public static boolean isPalindrome(int number) {
3           // Negative numbers are not palindromes
4           if (number < 0) {
5               return false;
6           }
7
8           int originalNumber = number;
9           int reversedNumber = 0;
10
11          // Reverse the number
12          while (number != 0) {
13              int digit = number % 10;
14              reversedNumber = reversedNumber * 10 + digit;
15              number /= 10;
16          }
17
18          // Check if the original number is equal to the reversed number
19          return originalNumber == reversedNumber;
20      }
21
        Run | Debug
22      public static void main(String[] args) {
23          int input1 = 121;
24          int input2 = -121;
25
26          System.out.println("Input: " + input1 + " Output: " + isPalindrome(input1));
27          System.out.println("Input: " + input2 + " Output: " + isPalindrome(input2));
28      }
29  }
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac Palindrome.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java Palindrome
Input: 121 Output: true
Input: -121 Output: false
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```



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
Ans:

```
public class LeapYear {
   public static boolean isLeapYear(int year) {
      // A year is a leap year if it is divisible by 4 but not by 100,
      // except if it is also divisible by 400.
      if (year % 4 == 0) {
         if (year % 100 == 0) {
            return year % 400 == 0;
```

```
        } else {
            return true;
        }
    } else {
        return false;
    }
}

public static void main(String[] args) {
    int input1 = 2020;
    int input2 = 1900;

    System.out.println("Input: " + input1 + " Output: " + isLeapYear(input1));
    System.out.println("Input: " + input2 + " Output: " + isLeapYear(input2));
}
}
```

SquareRoot.java 1 | RepeatedString.java | NonRepeatedCharacter.java | Palindrome.java | LeapYear.java

LeapYear.java > LeapYear

```
1   public class LeapYear {
2       public static boolean isLeapYear(int year) {
3           // A year is a leap year if it is divisible by 4 but not by 100,
4           // except if it is also divisible by 400.
5           if (year % 4 == 0) {
6               if (year % 100 == 0) {
7                   return year % 400 == 0;
8               } else {
9                   return true;
10              }
11          } else {
12              return false;
13          }
14      }
15
        Run | Debug
16      public static void main(String[] args) {
17          int input1 = 2020;
18          int input2 = 1900;
19
20          System.out.println("Input: " + input1 + " Output: " + isLeapYear(input1));
21          System.out.println("Input: " + input2 + " Output: " + isLeapYear(input2));
22      }
23  }
24
25
```

```
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
javac LeapYear.java
PS C:\Users\Sumit\Downloads\ADS Assignment 1> java LeapYear
Input: 2020 Output: true
Input: 1900 Output: false
PS C:\Users\Sumit\Downloads\ADS Assignment 1>
```