CSA0980 -- PROGRAMMING IN JAVA FOR IDL TECHNOLOGY :-

M . NAGA KESHAVA

REG NO :-192125019

1. Write a program to given an integer n, return true if it is a power of three. Otherwise, return false.

Input =27

Output= true

Explanation: 27=33

Test cases:

12

abc@45

1827

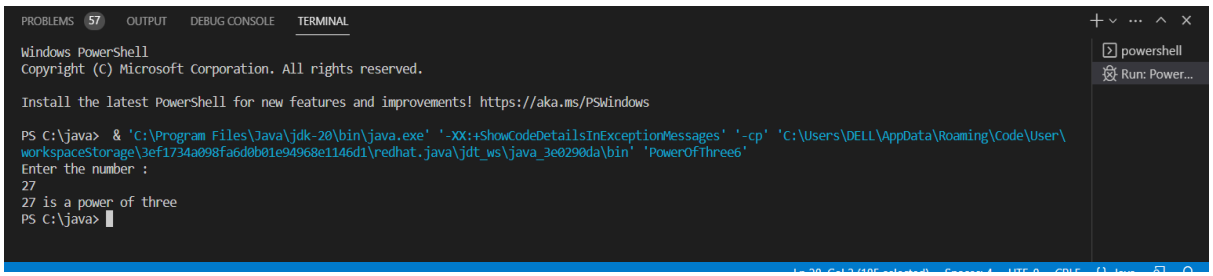
-100

0

PROGRAM:-

```
import java.util.*;
public class PowerOfThree6
{
     public static boolean isPowerOfThree6(int n)
        {
            if (n <= 0)
            {
                return false;
            }
            while (n % 3 == 0)
            {
                n /= 3;
            }
            return n == 1;
        }
        public static void main(String[] args)
        {
            Scanner v = new Scanner (System.in);
            System.out.println("Enter the number : ");
            int n = v.nextInt();
            if (isPowerOfThree6(n))
            {
                System.out.println(n + " is a power of three");
            } else {
                System.out.println(n + " is not a power of three");
            }
        }
}
```

OUTPUT :-



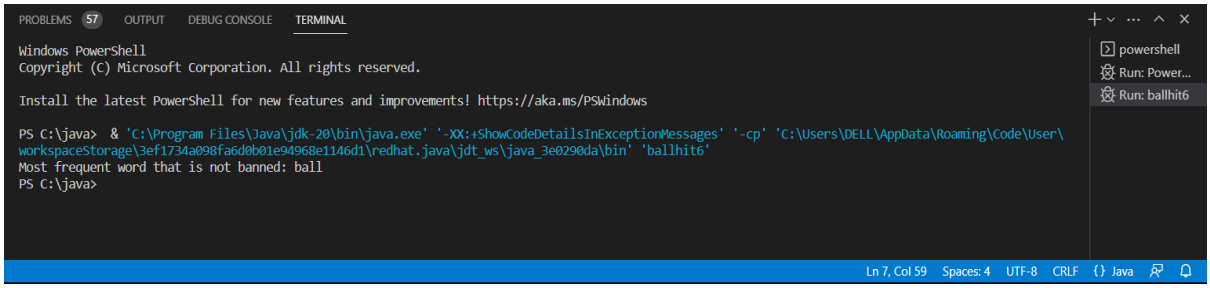
2. Write a program to given a string paragraph and a string array of the banned words banned, return the most frequent word that is not banned. It is guaranteed there is at least one word that is not banned, and that the answer is unique.
Input Paragraph=”Ram hit a ball, the hit ball flew far after it was hit”,
Banned = [hit]
Output=”Ball”

PROGRAM:-

```
import java.util.*;
public class ballhit6 {
    public static String mostCommonWord(String paragraph, String[] banned)
    {
        String cleanParagraph = paragraph.replaceAll("[^a-zA-Z0-9 ]",
"").toLowerCase();
        Map<String, Integer> wordCounts = new HashMap<>();
        String[] words = cleanParagraph.split("\\s+");
        for (String word : words) {
            if (!Arrays.asList(banned).contains(word)) {
                wordCounts.put(word, wordCounts.getOrDefault(word, 0) + 1);
            }
        }
        String mostCommonWord = null;
        int maxCount = 0;
        for (String word : wordCounts.keySet()) {
            int count = wordCounts.get(word);
            if (count > maxCount) {
                mostCommonWord = word;
                maxCount = count;
            }
        }
        return mostCommonWord;
    }
    public static void main(String[] args) {
```

```
        String paragraph = "Ram hit a ball, the hit ball flew far after it was
hit";
        String[] banned = {"hit"};
        String mostCommonWord = mostCommonWord(paragraph, banned);
        System.out.println("Most frequent word that is not banned: " +
mostCommonWord);
    }
}
```

OUTPUT :-



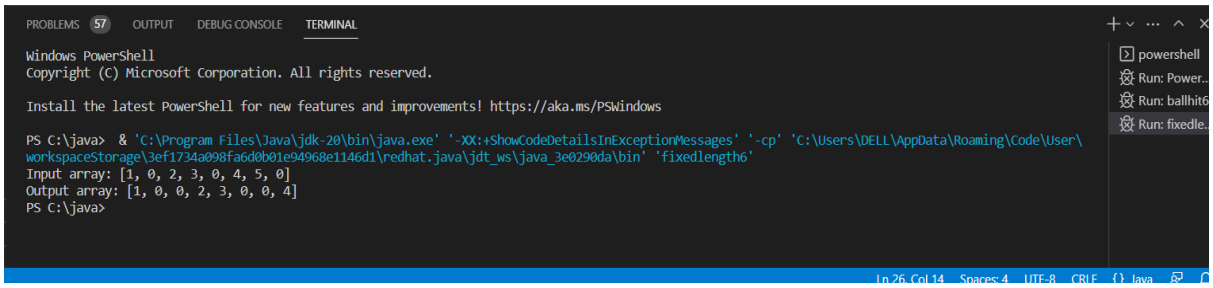
3. Write a program to given a fixed-length integer array arr, duplicate each occurrence of zero, shifting the remaining elements to the right.
Input: arr = [1, 0, 2, 3, 0, 4, 5, 0]
Output: [1, 0, 0, 2, 3, 0, 0, 4]
Explanation:
After calling your function, the input array is modified to [1, 0, 0, 2, 3, 0, 0, 4]

PROGRAM:-

```
import java.util.*;
public class fixedlength6
{
    public static void duplicateZeros(int[] arr)
    {
        int zeros = 0;
        int n = arr.length;
        for (int i = 0; i < n; i++) {
            if (arr[i] == 0)
            {
                zeros++;
            }
        }
        for (int i = n - 1, j = n + zeros - 1; i >= 0; i--, j--)
        {
            if (arr[i] == 0)
            {
                if (j < n) {
                    arr[j] = 0;
```

```
                }
                j--;
            }
            if (j < n)
            {
                arr[j] = arr[i];
            }
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {1, 0, 2, 3, 0, 4, 5, 0};
        System.out.println("Input array: " + Arrays.toString(arr));
        duplicateZeros(arr);
        System.out.println("Output array: " + Arrays.toString(arr));
    }
}
```

OUTPUT :-



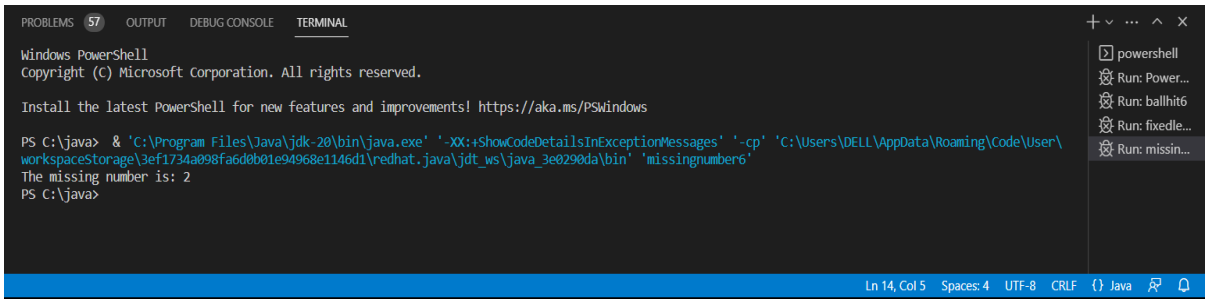
4. Write a program to given an array nums containing n distinct numbers in the range [0, n], return the only number in the range that is missing from the array.
Input nums = [3, 0, 1]
Output: 2
Explanation: n = 3 since there are 3 numbers, so all numbers are in the range [0, 3]. 2 is the missing number in the range since it does not appear in nums.

PROGRAM:-

```
public class missingnumber6
{
        public int missingNumber(int[] nums)
        {
            int n = nums.length;
            int sum = (n * (n + 1)) / 2;
            int actualSum = 0;
            for (int i = 0; i < n; i++)
            {
                actualSum += nums[i];
```

```
            }
            return sum - actualSum;
        }

        public static void main(String[] args) {
            int[] nums = {3, 0, 1};
            missingnumber6 mn = new missingnumber6();
            int missingNum = mn.missingNumber(nums);
            System.out.println("The missing number is: " + missingNum);
        }
    }
```

OUTPUT :-



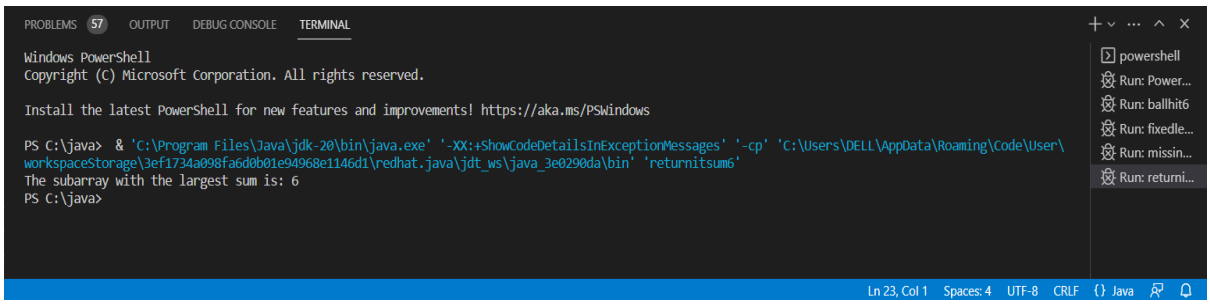
5. Write a program to given an integer array nums, find the subarray with the largest sum, and return its sum.
Input nums = [-2, 1,-3, 4,-1, 2, 1,-5, 4]
Output: 6
Explanation: The subarray [4,-1, 2, 1] has the largest sum 6.

PROGRAM:-

```
public class returnitsum6
{
        public static int maxSubArray(int[] nums)
        {
            int maxSum = nums[0];
            int currentSum = nums[0];
            for (int i = 1; i < nums.length; i++)
            {

                currentSum = Math.max(nums[i], currentSum + nums[i]);
                maxSum = Math.max(maxSum, currentSum);
            }
            return maxSum;
        }
```

```
        public static void main(String[] args)
        {
            int[] nums = {-2, 1, -3, 4, -1, 2, 1, -5, 4};
            int largestSum = maxSubArray(nums);
            System.out.println("The subarray with the largest sum is: " +
largestSum);
        }
}
```

OUTPUT :-



6. Write Java programs to implement multiple threads and apply join method for thread and thread has to be started after 500ms using sleep ().

PROGRAM:-

```
public class sleepthread6
{
        public static void main(String[] args)
        {
            Thread t1 = new Thread(new Runnable()
            {
                public void run() {
                    try {
                        System.out.println("Thread 1 started");
                        Thread.sleep(2000);
                        System.out.println("Thread 1 ended");
                    } catch (InterruptedException e)
                    {
                        e.printStackTrace();
                    }
                }
            });
            Thread t2 = new Thread(new Runnable()
            {
                public void run() {
                    try {
                        System.out.println("Thread 2 started");
                        Thread.sleep(3000);
```

```
                    System.out.println("Thread 2 ended");
                } catch (InterruptedException e)
                {
                    e.printStackTrace();
                }
            }
        });
        try {
            Thread.sleep(500);
            t1.start();
            t2.start();
        } catch (InterruptedException e)
        {
            e.printStackTrace();
        }
        try {
            t1.join();
            t2.join();
        } catch (InterruptedException e)
        {
            e.printStackTrace();
        }
        System.out.println("Threads have completed");
    }
}
```

OUTPUT :-

7. Generate a Java code that implements java selection and iteration statements. Use do while loop to process a menu selection. When a menu is selected, it should display the syntax of the selected statements.

PROGRAM:-

```
import java.util.*;
public class menuselectedjava6 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int choice;
```

```
        do {
            System.out.println("Menu:");
            System.out.println("1. if statement");
            System.out.println("2. switch statement");
            System.out.println("3. for loop");
            System.out.println("4. while loop");
            System.out.println("5. do-while loop");
            System.out.println("0. Exit");
            System.out.print("Enter your choice: ");

            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.println("Syntax of if statement:");
                    System.out.println("if (condition) {");
                    System.out.println("    // code to be executed if
condition is true");
                    System.out.println("}");
                    break;
                case 2:
                    System.out.println("Syntax of switch statement:");
                    System.out.println("switch (expression) {");
                    System.out.println("    case value1:");
                    System.out.println("        // code to be executed if
expression matches value1");
                    System.out.println("        break;");
                    System.out.println("    case value2:");
                    System.out.println("        // code to be executed if
expression matches value2");
                    System.out.println("        break;");
                    System.out.println("    // ...");
                    System.out.println("    default:");
                    System.out.println("        // code to be executed if none
of the cases match");
                    System.out.println("}");
                    break;
                case 3:
                    System.out.println("Syntax of for loop:");
                    System.out.println("for (initialization; condition;
update) {");
                    System.out.println("    // code to be executed");
                    System.out.println("}");
                    break;
                case 4:
                    System.out.println("Syntax of while loop:");
                    System.out.println("while (condition) {");
                    System.out.println("    // code to be executed");
```

```
                    System.out.println("}");
                    break;
                case 5:
                    System.out.println("Syntax of do-while loop:");
                    System.out.println("do {");
                    System.out.println("    // code to be executed");
                    System.out.println("} while (condition);");
                    break;
                case 0:
                    System.out.println("Exiting...");
                    break;
                default:
                    System.out.println("Invalid choice, try again.");
                    break;
            }
        } while (choice != 0);

        scanner.close();
    }
}
```

OUTPUT :-

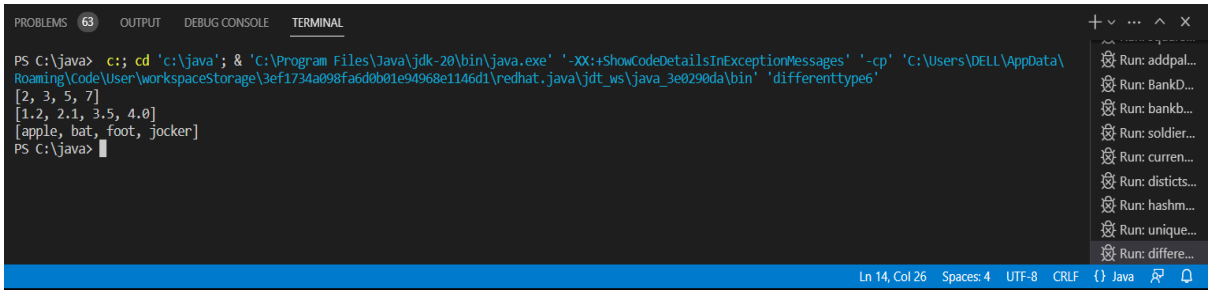
8. Create a simple generics class with type parameters for sorting values of different types.

PROGRAM:-

```
import java.util.Arrays;

class Sorter<T extends Comparable<T>> {
    private T[] arr;

    public Sorter(T[] arr) {
        this.arr = arr;
    }

    public void sort() {
```

```
        Arrays.sort(arr);
    }

    public void print() {
        System.out.println(Arrays.toString(arr));
    }
}

public class differenttype6 {
        public static void main(String[] args) {
            Integer[] intArr = { 5, 2, 7, 3 };
            Double[] doubleArr = { 1.2, 3.5, 2.1, 4.0 };
            String[] strArr = { "foot", "bat", "apple", "jocker" };

            Sorter<Integer> intSorter = new Sorter<>(intArr);
            intSorter.sort();
            intSorter.print();

            Sorter<Double> doubleSorter = new Sorter<>(doubleArr);
            doubleSorter.sort();
            doubleSorter.print();

            Sorter<String> strSorter = new Sorter<>(strArr);
            strSorter.sort();
            strSorter.print();
        }
    }
```

OUTPUT :-



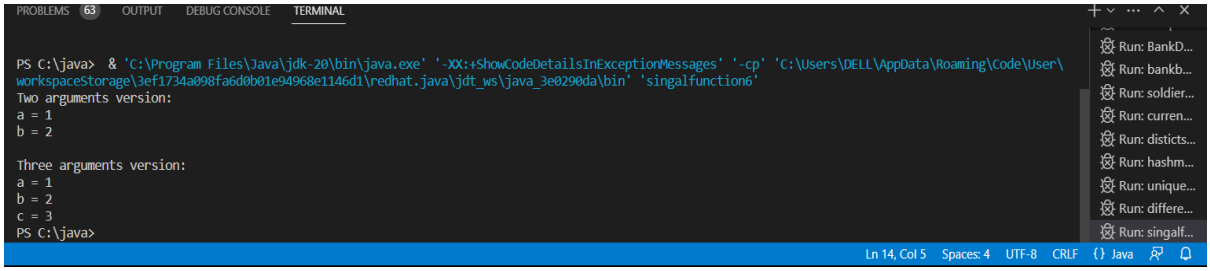
9. Create a class name ‘overload’. Write a program to assign the values for two values by different number of arguments using a single function.

PROGRAM:-

```
public class singalfunction6 {
        public void assignValues(int a, int b) {
            System.out.println("Two arguments version:");
```

```
            System.out.println("a = " + a);
            System.out.println("b = " + b);
        }

        public void assignValues(int a, int b, int c) {
            System.out.println("Three arguments version:");
            System.out.println("a = " + a);
            System.out.println("b = " + b);
            System.out.println("c = " + c);
        }

        public static void main(String[] args) {
            singalfunction6 obj = new singalfunction6();
            obj.assignValues(1, 2);
            System.out.println();
            obj.assignValues(1, 2, 3);
        }
}
```

OUTPUT :-



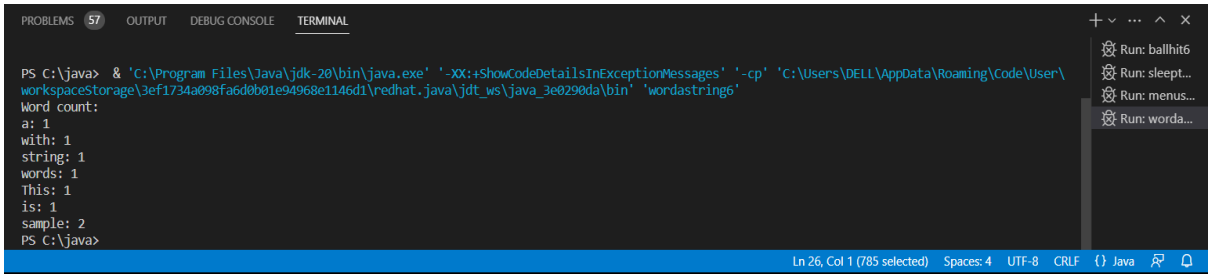
10. Write a Java Program to count the number of words in a string using Hash Map.

PROGRAM:-

```
import java.util.HashMap;
import java.util.Map;
public class wordastring6 {

    public static void main(String[] args) {
        String str = "This is a sample string with sample words";
        String[] words = str.split(" ");

        Map<String, Integer> wordCount = new HashMap<>();

        for (String word : words) {
            if (wordCount.containsKey(word)) {
                int count = wordCount.get(word);
                wordCount.put(word, count + 1);
            } else {
```

```
                wordCount.put(word, 1);
            }
        }

        System.out.println("Word count:");
        for (String word : wordCount.keySet()) {
            System.out.println(word + ": " + wordCount.get(word));
        }
    }
}
```

OUTPUT :-



11. Write a Java Program to read an email and password from excel sheet by retrieving the cell using getRow() and getCell() method.

PROGRAM:-

```
import java.io.FileInputStream;
import java.io.IOException;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
public class excelsheet6 {

    public static void main(String[] args) {
        try {
            FileInputStream file = new
FileInputStream("path/to/excel/file.xlsx");
            Workbook workbook = new XSSFWorkbook(file);
            Sheet sheet = workbook.getSheetAt(0); // Accessing the first
worksheet
            Row row = sheet.getRow(0); // Accessing the first row
            Cell emailCell = row.getCell(0); // Accessing the first cell
            Cell passwordCell = row.getCell(1); // Accessing the second cell
            String email = emailCell.getStringCellValue();
            String password = passwordCell.getStringCellValue();
```

```
            System.out.println("Email: " + email);
            System.out.println("Password: " + password);
            workbook.close();
            file.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

12. Write a Java program to sorts the given value using Hash Map.

PROGRAM:-

```
import java.util.*;
public class hashmap6 {
          public static void main(String[] args) {
            HashMap<String, Integer> map = new HashMap<>();
            map.put("Alice", 27);
            map.put("Bob", 32);
            map.put("Charlie", 19);
            map.put("David", 45);
            TreeMap<String, Integer> sortedMap = new TreeMap<>(map);
            for (Map.Entry<String, Integer> entry : sortedMap.entrySet()) {
                System.out.println(entry.getKey() + " : " + entry.getValue());
            }
        }
    }
```

OUTPUT :-

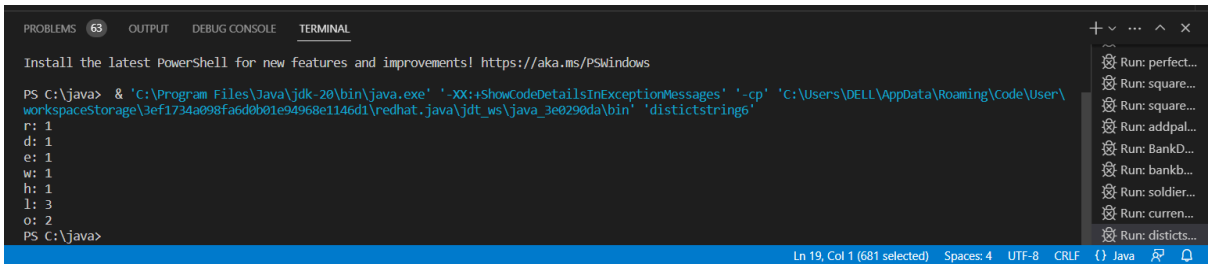
13. Write a Java program to find distinct characters and their count in a string.

PROGRAM:-

```
import java.util.*;
public class distictstring6 {
     public static void main(String[]args)
     {
```

```
         String input = "Hello, World!";
        Map<Character, Integer> characterCount = new HashMap<>();
        for (int i = 0; i < input.length(); i++) {
            char ch = input.charAt(i);
            if (Character.isLetter(ch)) {
                ch = Character.toLowerCase(ch);
                characterCount.put(ch, characterCount.getOrDefault(ch, 0) +
1);
            }
        }
        for (Map.Entry<Character, Integer> entry : characterCount.entrySet())
{
            System.out.println(entry.getKey() + ": " + entry.getValue());
        }
    }
}
```

OUTPUT :-



14. Write a program to print all the unique characters in a String. For instance, if the input string is “abcb”, the output will be the characters ‘a’ and ‘c’ as they are unique. The character ‘b’ repeats twice and so it will not be printed.

PROGRAM:-

```
import java.util.HashSet;
public class uniquecharacter6 {
    public static void main(String[] args) {
        String input = "abcb";
        HashSet<Character> uniqueChars = new HashSet<Character>();

        for (int i = 0; i < input.length(); i++) {
            char c = input.charAt(i);
            if (input.indexOf(c) == input.lastIndexOf(c)) {
                uniqueChars.add(c);
            }
        }

        System.out.println("Unique characters in the string \"" + input + "\"
are:");
```

```
        for (char c : uniqueChars) {
            System.out.print(c + " ");
        }
    }
}
```

OUTPUT :-

15. Write a Program to create a list of all numbers in a range which are perfect squares and the sum of the digits of the number is less than 10.
Sample Input & Output:
Enter lower range: 1
Enter upper range: 40
[1, 4, 9, 16, 25, 36]

PROGRAM:-

```
import java.util.ArrayList;
import java.util.List;
public class perfectrange6
{
    public static void main(String[] args)
    {
        int lowerRange = 1;
        int upperRange = 40;
        List<Integer> perfectSquareList = new ArrayList<>();
        for (int i = lowerRange; i <= upperRange; i++)
        {
            if (isPerfectSquare(i) && digitSum(i) < 10)
            {
                perfectSquareList.add(i);
            }
        }
        System.out.println(perfectSquareList);
    }
    public static boolean isPerfectSquare(int n) {
        int sqrt = (int) Math.sqrt(n);
        return sqrt * sqrt == n;
    }
    public static int digitSum(int n) {
        int sum = 0;
        while (n > 0) {
```

```
            sum += n % 10;
            n /= 10;
        }
        return sum;
    }
}
```

OUTPUT :-

16. Write a Program to create an array with the First Element as the Number and Second Element as the Square of the Number.
Sample Input:
Enter the lower range:45
Enter the upper range:49
Sample Output:
[(45, 2025), (46, 2116), (47, 2209), (48, 2304), (49, 2401)]

PROGRAM:-

```
import java.util.*;
public class squareofnumbers6Arrays {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the lower range: ");
        int lowerRange = input.nextInt();
        System.out.print("Enter the upper range: ");
        int upperRange = input.nextInt();

        int[][] numberSquareArray = new int[upperRange - lowerRange + 1][2];
        int i = 0;

        for (int num = lowerRange; num <= upperRange; num++) {
            numberSquareArray[i][0] = num;
            numberSquareArray[i][1] = num * num;
            i++;
        }

        System.out.println(Arrays.deepToString(numberSquareArray));
    }
```

```
}
```

OUTPUT :-

17. Develop a code to Reverse and Add a Number until you get a Palindrome.

Sample Input If 7325 is input number, then
7325 (Input Number) + 5237 (Reverse Of Input Number) = 12562
12562 + 26521 = 39083
39083 + 38093 = 77176
77176 + 67177 = 144353
144353 + 353441 = 497794 (Palindrome)

PROGRAM:-

```
import java.util.*;
public class addpalindrome6 {

    public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                System.out.print("Enter a number: ");
                int num = sc.nextInt();
                int count = 0;

                while (!isPalindrome(num)) {
                    int reverseNum = reverse(num);
                    num += reverseNum;
                    count++;
                    System.out.println(num + " (number after " + count + "
iterations)");
                }

                System.out.println("Final Palindrome: " + num);
                System.out.println("Total iterations: " + count);
            }
            public static int reverse(int num) {
                int rev = 0;
                while (num != 0) {
                    int digit = num % 10;
```

```
            rev = rev * 10 + digit;
            num /= 10;
        }
        return rev;
    }
    public static boolean isPalindrome(int num) {
        return num == reverse(num);
    }
}
```

OUTPUT :-

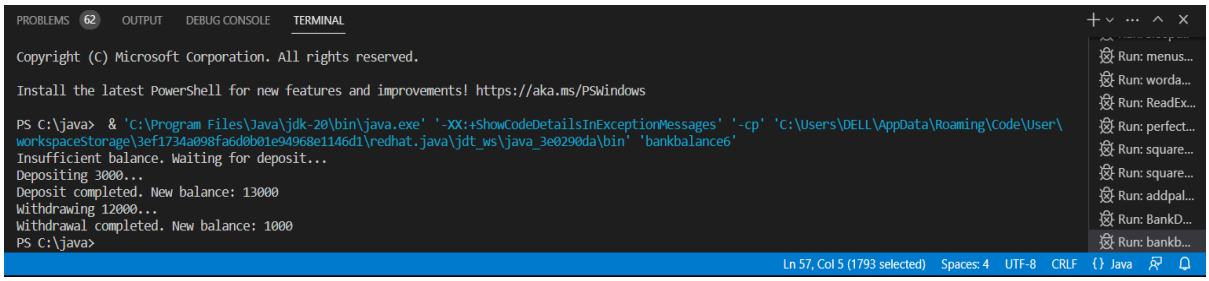
18. Create Customer class with deposit() and withdraw() as synchronized methods. Declare AccountNo, AccName and Balance as Instance Variables inside the class. From the main class, Input the amount for withdraw() operation and if requested amount is not available in existing Balance amount, withdraw() method should be temporarily suspended using wait() method until deposit() method receives the input for amount, to be added in the existing Balance amount and then withdraw() would be completed in a successful manner. Develop the above scenario using Synchronization and Inter-Thread Communication.
Note : existing Bank balance amount 10000
Sample Input : 12000, 3000
Sample Output : Withdraw operation success, balance amount 1000

PROGRAM:-

```
import java.util.*;
class customer {
        private int accountNo;
        private String accName;
        private int balance = 10000;

        public customer(int accountNo, String accName) {
            this.accountNo = accountNo;
            this.accName = accName;
        }

        public synchronized void deposit(int amount) {
```

```
            System.out.println("Depositing " + amount + "...");
            balance += amount;
            System.out.println("Deposit completed. New balance: " + balance);
            notify();
        }

        public synchronized void withdraw(int amount) {
            if (balance < amount) {
                System.out.println("Insufficient balance. Waiting for
deposit...");
                try {
                    wait();
                } catch (InterruptedException e) {
                    System.out.println(e.getMessage());
                }
            }
            System.out.println("Withdrawing " + amount + "...");
            balance -= amount;
            System.out.println("Withdrawal completed. New balance: " +
balance);
        }
    }

    public class bankbalance6 {
        public static void main(String[] args) {
            customer customer = new customer(12345, "John Doe");

            Thread t1 = new Thread(() -> {
                customer.withdraw(12000);
            });

            Thread t2 = new Thread(() -> {
                customer.deposit(3000);
            });

            t1.start();
            t2.start();

            try {
                t1.join();
                t2.join();
            } catch (InterruptedException e) {
                System.out.println(e.getMessage());
            }
        }
    }
```

OUTPUT :-



19. You are given an m x n binary matrix mat of 1's (representing soldiers) and 0's (representing civilians). The soldiers are positioned in front of the civilians. That is, all the 1's will appear to the left of all the 0's in each row.
A row i is weaker than a row j if one of the following is true:
The number of soldiers in row i is less than the number of soldiers in row j.
Both rows have the same number of soldiers and i< j. Return the indices of the k weakest rows in the matrix ordered from weakest to strongest.
Example 1:
Input: mat =
[[1,1,0,0,0],
[1,1,1,1,0],
[1,0,0,0,0],
[1,1,0,0,0],
[1,1,1,1,1]],
k = 3
Output: [2,0,3]

PROGRAM:-

```java
import java.util.*;
public class soldiersinrow6 {
    public static void main(String[] args) {
        int[][] mat = {{1,1,0,0,0},
                       {1,1,1,1,0},
                       {1,0,0,0,0},
                       {1,1,0,0,0},
                       {1,1,1,1,1}};
        int k = 3;
        int[] weakestRows = kWeakestRows(mat, k);
        System.out.println(Arrays.toString(weakestRows));
    }
    public static int[] kWeakestRows(int[][] mat, int k) {
        Map<Integer, Integer> soldierCount = new HashMap<>();
        for (int i = 0; i < mat.length; i++) {
            int count = 0;
            for (int j = 0; j < mat[0].length; j++) {
```

```
                if (mat[i][j] == 1) {
                    count++;
                } else {
                    break;
                }
            }
            soldierCount.put(i, count);
        }
        List<Map.Entry<Integer, Integer>> list = new
LinkedList<>(soldierCount.entrySet());
        Collections.sort(list, new Comparator<Map.Entry<Integer, Integer>>() {
            public int compare(Map.Entry<Integer, Integer> o1,
Map.Entry<Integer, Integer> o2) {
                int c = o1.getValue().compareTo(o2.getValue());
                if (c == 0) {
                    c = o1.getKey().compareTo(o2.getKey());
                }
                return c;
            }
        });
        int[] weakestRows = new int[k];
        for (int i = 0; i < k; i++) {
            weakestRows[i] = list.get(i).getKey();
        }

        return weakestRows;
    }
}
```

OUTPUT :-

20. Given an integer num, return the number of steps to reduce it to zero. In one step, if the current number is even, you have to divide it by 2, otherwise, you have to subtract 1 from it.

Example 1:

Input: num = 14

Output: 6

Explanation:

Step 1) 14 is even; divide by 2 and obtain 7.

Step 2) 7 is odd; subtract 1 and obtain 6.

Step 3) 6 is even; divide by 2 and obtain 3.
Step 4) 3 is odd; subtract 1 and obtain 2.
Step 5) 2 is even; divide by 2 and obtain 1.
Step 6) 1 is odd; subtract 1 and obtain 0.

PROGRAM:-

```
import java.util.Scanner;
public class currennumber6 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = input.nextInt();
        int steps = reduceToZero(num);
        System.out.println("Number of steps to reduce " + num + " to zero: " +
steps);
    }

    public static int reduceToZero(int num) {
        int steps = 0;
        while (num != 0) {
            if (num % 2 == 0) {
                num /= 2;
            } else {
                num -= 1;
            }
            steps++;
        }
        return steps;
    }
}
```

OUTPUT :-