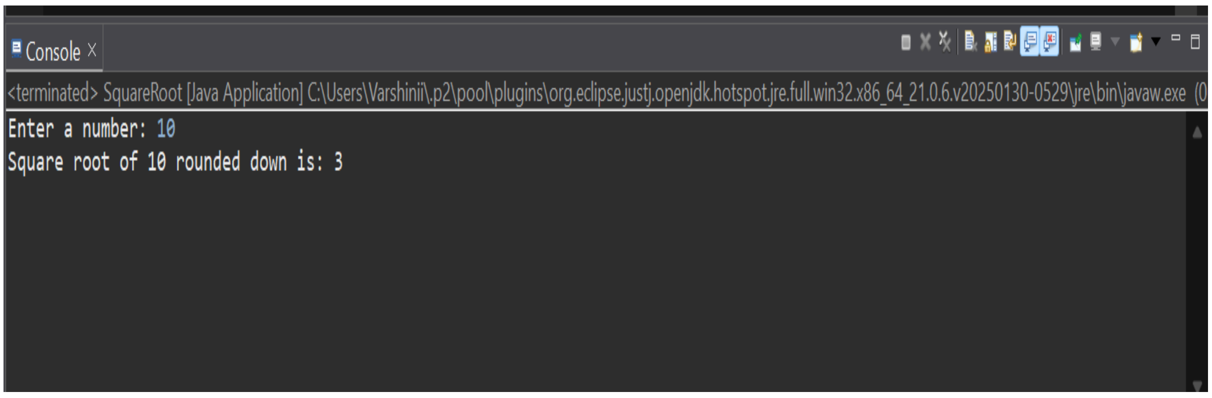
TASK 1 – MATHEMATICS – WHY MATHS IN CODING?

1.A

PROGRAM:

```
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();
        int result = 0;
        for (int i = 1; i <= x; i++) {
            if (i * i <= x) {
                result = i;
            } else {
                break;
            }
        }
       System.out.println("Square root of " + x + " rounded down is: " + result);
       scanner.close();
    }
}
```

OUTPUT:



```
<terminated> SquareRoot [Java Application] C:\Users\Varshinii\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw.exe (0
Enter a number: 10
Square root of 10 rounded down is: 3
```

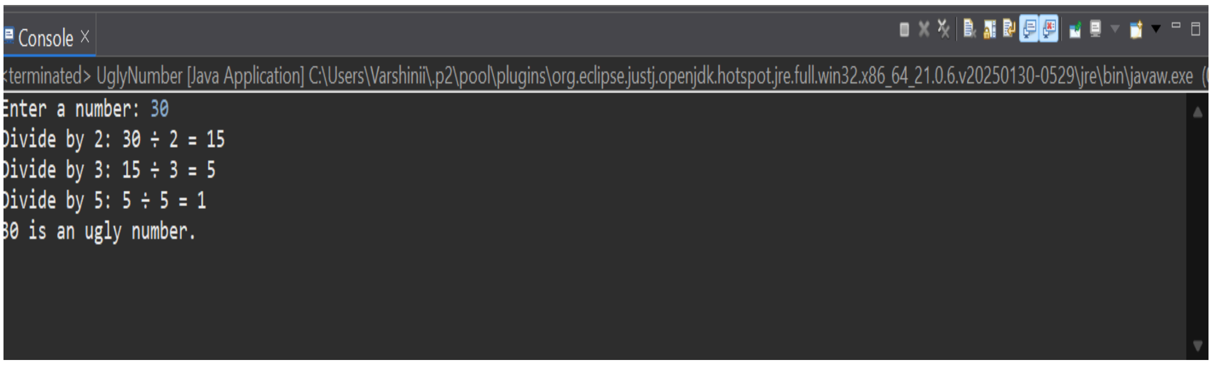
1.B

PROGRAM:

```
import java.util.Scanner;
public class UglyNumber {
   public static boolean isUgly(int num) {
        if (num <= 0) return false;
       while (num % 2 == 0) {
           System.out.println("Divide by 2: " + num + " ÷ 2 = " + (num / 2));
           num /= 2;
        }
        while (num % 3 == 0) {
           System.out.println("Divide by 3: " + num + " ÷ 3 = " + (num / 3));
           num /= 3;
        }
        while (num % 5 == 0) {
           System.out.println("Divide by 5: " + num + " ÷ 5 = " + (num / 5));
           num /= 5;
        }
       return num == 1;
   }
public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
       System.out.print("Enter a number: ");
       if (isUgly(input)) {
           System.out.println(input + " is an ugly number.");
        } else {
           System.out.println(input + " is not an ugly number.");
        }
        scanner.close()
```

```
  }
}
```

OUTPUT:



Console

```
<terminated> UglyNumber [Java Application] C:\Users\Varshinii\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw.exe (
Enter a number: 30
Divide by 2: 30 ÷ 2 = 15
Divide by 3: 15 ÷ 3 = 5
Divide by 5: 5 ÷ 5 = 1
30 is an ugly number.
```

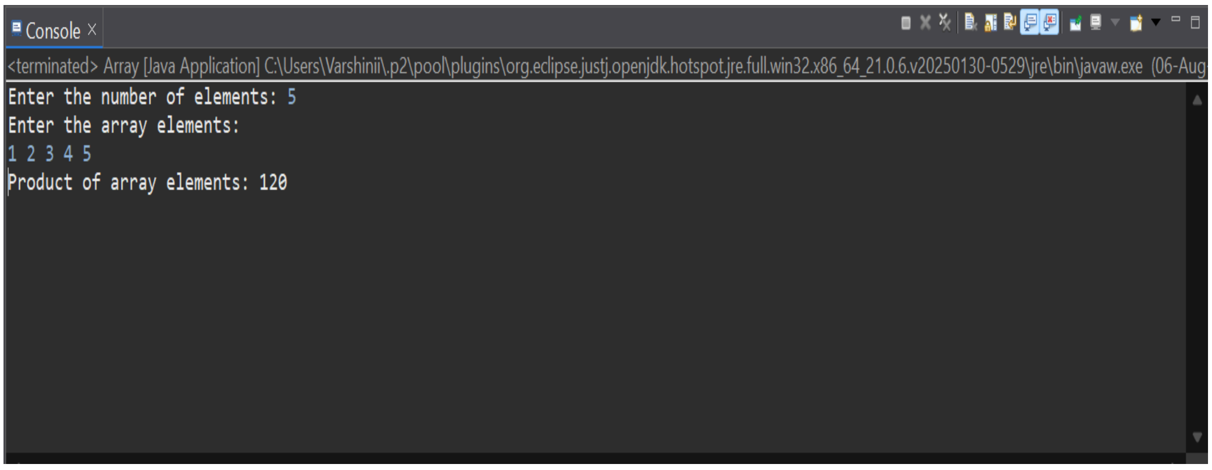
1.C

PROGRAM:

```
import java.util.Scanner;
public class CC2 {

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

     int[] ar = new int[n];
     System.out.println("Enter the array elements:");
     for (int i = 0; i < n; i++) {
        ar[i] = scanner.nextInt();
     }
     int result = 1;
```

```
        for (int i = 0; i < n; i++) {
            result *= ar[i];
        }
        System.out.println("Product of array elements: " + result);


        scanner.close();
    }
}
```

OUTPUT:



```
<terminated> Array [Java Application] C:\Users\Varshini\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw.exe (06-Aug
Enter the number of elements: 5
Enter the array elements:
1 2 3 4 5
Product of array elements: 120
```

TASK NO: 2 :- ADVANCED CONCEPTS ON ARRAY :

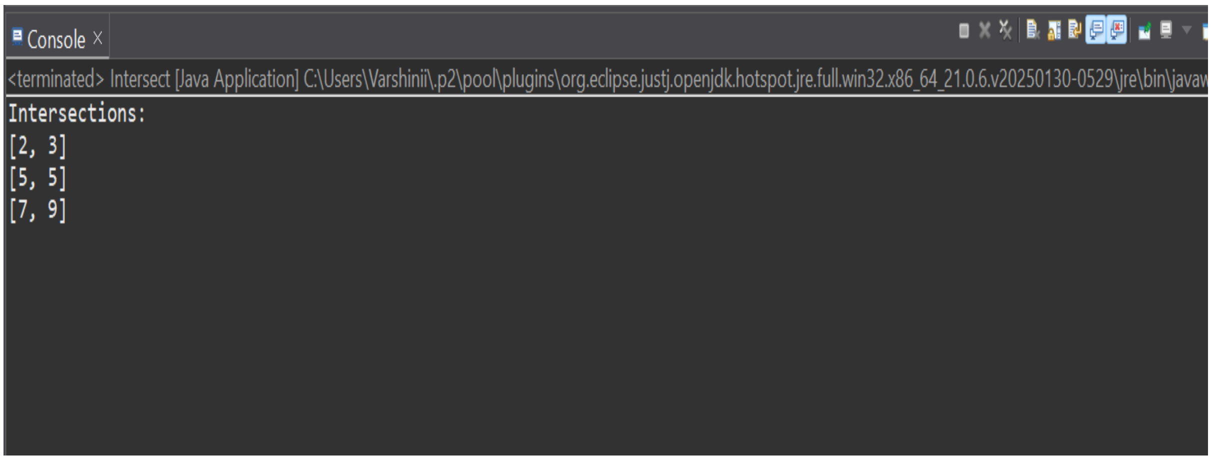
2.A

PROGRAM:

```
import java.util.*;

public class Intersect {

  public static int[][] get(int[][] A, int[][] B) {
    List<int[]> res = new ArrayList<>();
    int i = 0, j = 0;

    while (i < A.length && j < B.length) {
      int start = Math.max(A[i][0], B[j][0]);
      int end = Math.min(A[i][1], B[j][1]);

      if (start <= end) {
        res.add(new int[]{start, end});
      }

      if (A[i][1] < B[j][1]) i++;
      else j++;
    }

    return res.toArray(new int[res.size()][]);
  }

  public static void main(String[] args) {
    int[][] A = {{1, 3}, {5, 9}};
```

```
        int[][] B = {{2, 5}, {7, 10}};

        int[][] out = get(A, B);

        System.out.println("Intersections:");
        for (int[] in : out) {
            System.out.println("[" + in[0] + ", " + in[1] + "]");
        }
    }
}
```

OUTPUT:



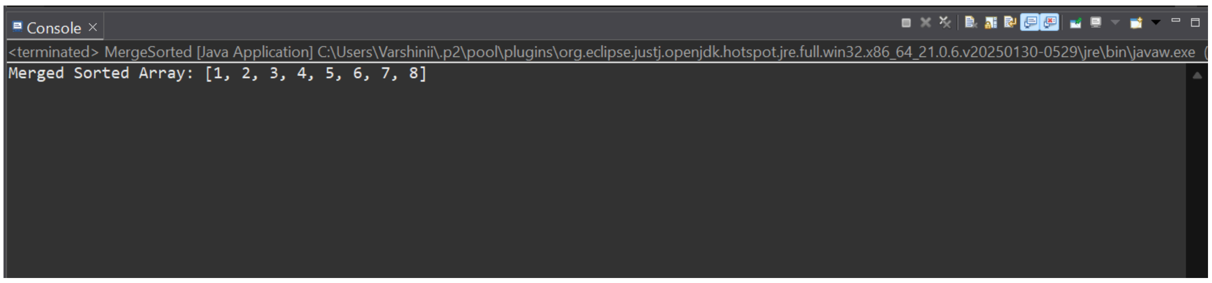
```
<terminated> Intersect [Java Application] C:\Users\Varshinii\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw
Intersections:
[2, 3]
[5, 5]
[7, 9]
```

2.B

PROGRAM:

```
import java.util.*;
public class MergeSorted {
public static int[] merge(int[] a, int[] b) {
        int n = a.length, m = b.length;
        int[] res = new int[n + m];
        int i = 0, j = 0, k = 0;

        while (i < n && j < m) {
            if (a[i] < b[j]) res[k++] = a[i++];
            else res[k++] = b[j++];
        }

        while (i < n) res[k++] = a[i++];
        while (j < m) res[k++] = b[j++];

        return res;
    }

    public static void main(String[] args) {
        int[] arr1 = {1, 3, 5, 7};
        int[] arr2 = {2, 4, 6, 8};

        int[] merged = merge(arr1, arr2);

        System.out.println("Merged Sorted Array: " + Arrays.toString(merged));
    }
}
```

OUTPUT:



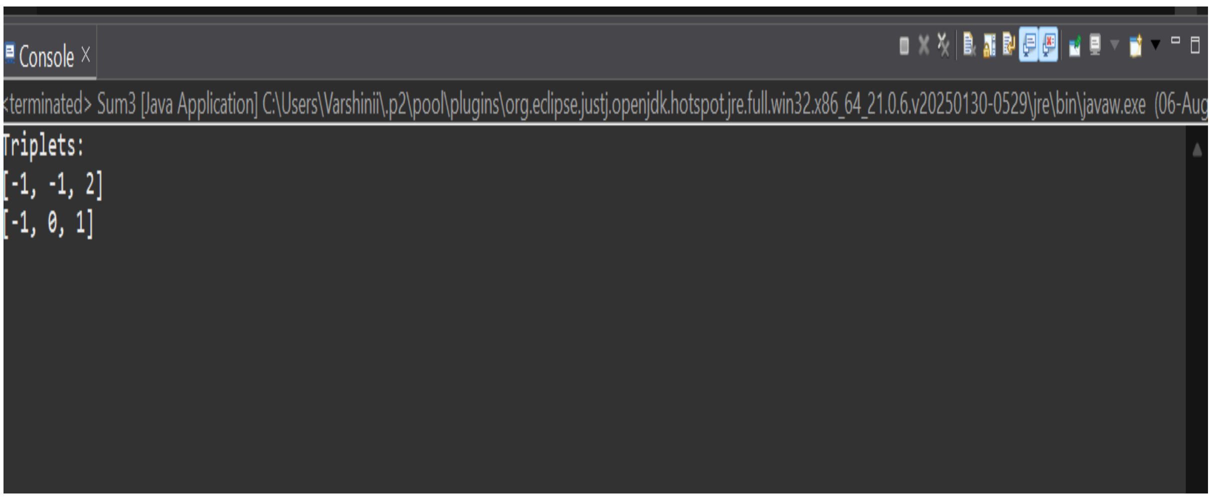
```
<terminated> MergeSorted [Java Application] C:\Users\Varshinii\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw.exe
Merged Sorted Array: [1, 2, 3, 4, 5, 6, 7, 8]
```

2.C

PROGRAM:

```
package cc2.task;

import java.util.*;

public class Sum3 {

    public static List<List<Integer>> trip(int[] a) {
        List<List<Integer>> res = new ArrayList<>();
        Arrays.sort(a);

        for (int i = 0; i < a.length - 2; i++) {
            if (i > 0 && a[i] == a[i - 1]) continue;

            int l = i + 1, r = a.length - 1;

            while (l < r) {
                int s = a[i] + a[l] + a[r];

                if (s == 0) {
                    res.add(Arrays.asList(a[i], a[l], a[r]));
                    while (l < r && a[l] == a[l + 1]) l++;
                    while (l < r && a[r] == a[r - 1]) r--;
                    l++;
                    r--;
                } else if (s < 0) {
                    l++;
                } else {
                    r--;
                }
            }
        }
        return res;
    }

    public static void main(String[] args) {
        int[] nums = {-1, 0, 1, 2, -1, -4};
        List<List<Integer>> ans = trip(nums);

        System.out.println("Triplets:");
        for (List<Integer> t : ans) {
            System.out.println(t);
        }
    }
}
```

OUTPUT:



```
<terminated> Sum3 [Java Application] C:\Users\Varshinii\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.6.v20250130-0529\jre\bin\javaw.exe (06-Aug
Triplets:
[-1, -1, 2]
[-1, 0, 1]
```

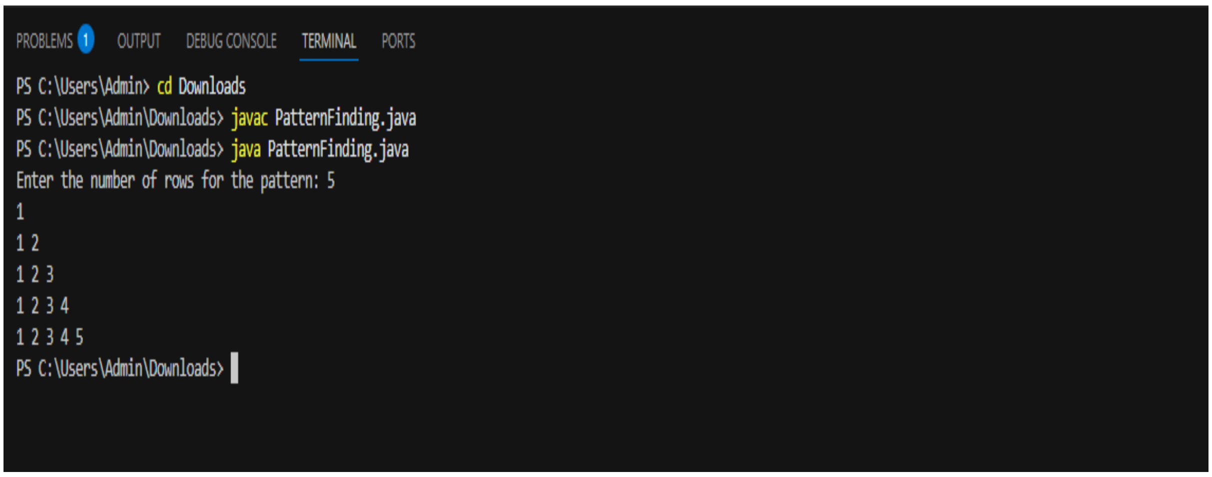
TASK 3 : ADVANCED CONCEPTS ON STRINGS

3.A

PROGRAM:

```
import java.util.Scanner;
public class Pattern {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of rows for the pattern: ");
        int n = sc.nextInt();
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print(j + " ");
            }
            System.out.println();
        }
        sc.close();
    }
}
```

OUTPUT:



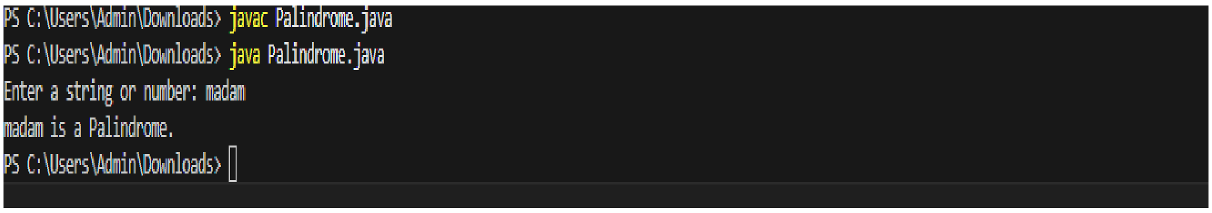
```
PS C:\Users\Admin> cd Downloads
PS C:\Users\Admin\Downloads> javac PatternFinding.java
PS C:\Users\Admin\Downloads> java PatternFinding.java
Enter the number of rows for the pattern: 5
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
PS C:\Users\Admin\Downloads>
```

3.B

PROGRAM:

```
mport java.util.Scanner;

public class Palindrome {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter a string or number: ");
        String input = sc.nextLine();

        String str = input.toLowerCase().replace(" ", "");

        String reversed = "";
        // Reverse the string manually
        for (int i = str.length() - 1; i >= 0; i--) {
            reversed += str.charAt(i);
        }

        if (str.equals(reversed)) {
            System.out.println(input + " is a Palindrome.");
        } else {
            System.out.println(input + " is NOT a Palindrome.");
        }

        sc.close();
    }
}
```

OUTPUT:

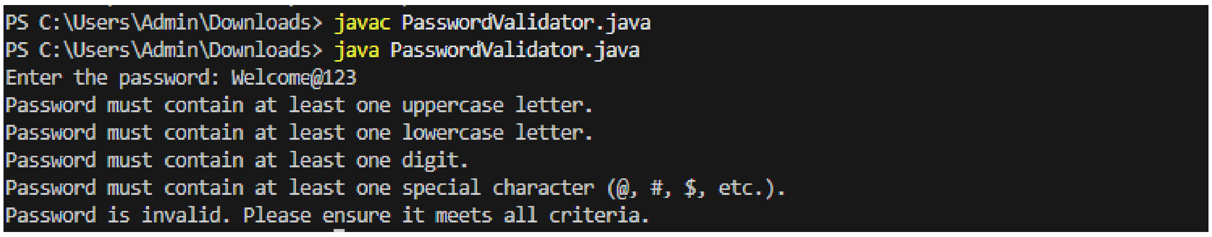

```
PS C:\Users\Admin\Downloads> javac Palindrome.java
PS C:\Users\Admin\Downloads> java Palindrome.java
Enter a string or number: madam
madam is a Palindrome.
PS C:\Users\Admin\Downloads>
```

3.C

PROGRAM:

```
import java.util.Scanner;

public class PasswordValidator {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the password: ");
        String password = scanner.nextLine();

        boolean isValid = true;

        if (password.length() < 8) {
            System.out.println("Password must be at least 8 characters long.");
            isValid = false;
        }


        if (!password.matches(".[A-Z].")) {
            System.out.println("Password must contain at least one uppercase letter.");
            isValid = false;
        }



        if (!password.matches(".[a-z].")) {
            System.out.println("Password must contain at least one lowercase letter.");
            isValid = false;
        }

      java
        if (!password.matches(".\\d.")) {
            System.out.println("Password must contain at least one digit.");
            isValid = false;
        }

        if (!password.matches(".[@#!$%^&].*")) {
            System.out.println("Password must contain at least one special character (@, #, $,
etc.).");
            isValid = false;
        }

        if (isValid) {
            System.out.println("Password is valid.");
        } else {
            System.out.println("Password is invalid. Please ensure it meets all criteria.");
        }

        scanner.close();
    }
}
```

OUTPUT:



```
PS C:\Users\Admin\Downloads> javac PasswordValidator.java
PS C:\Users\Admin\Downloads> java PasswordValidator.java
Enter the password: Welcome@123
Password must contain at least one uppercase letter.
Password must contain at least one lowercase letter.
Password must contain at least one digit.
Password must contain at least one special character (@, #, $, etc.).
Password is invalid. Please ensure it meets all criteria.
```

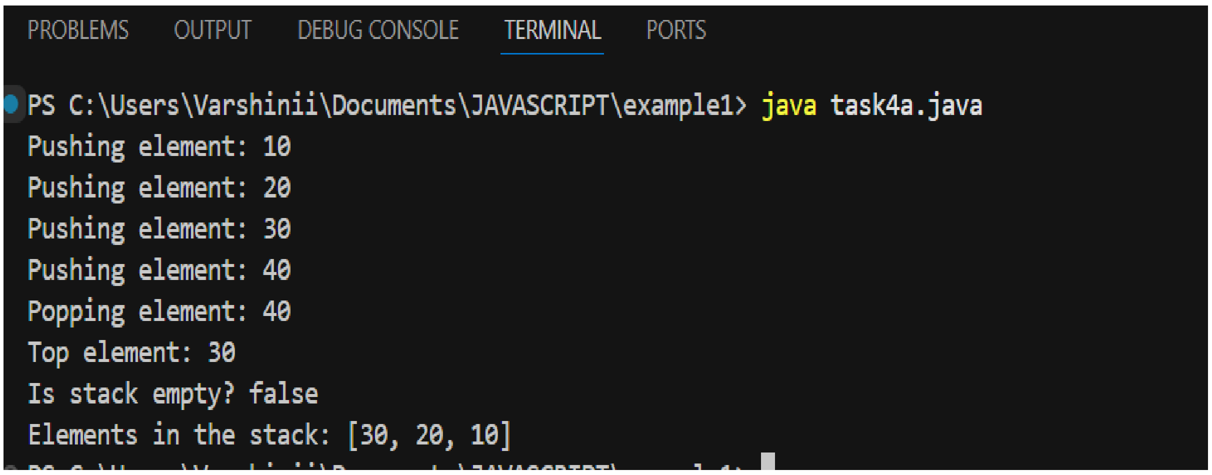
TASK 4 : STACK AND QUEUES

4.a

PROGRAM:

```
import java.util.*;
public class task4a {
    Queue<Integer> q1 = new LinkedList<>();
    Queue<Integer> q2 = new LinkedList<>();
     public void push(int x) {
        System.out.println("Pushing element: " + x);
        q2.add(x);
        while (!q1.isEmpty()) {
            q2.add(q1.poll());
        }
        swap();
    }
    private void swap() {
        Queue<Integer> temp = q1;
        q1 = q2;
        q2 = temp;
    }
   public void pop() {
        if (q1.isEmpty()) {
            System.out.println("Stack is empty. Cannot pop.");
            return;
        }
        System.out.println("Popping element: " + q1.poll());
    }
   public void top() {
        if (q1.isEmpty()) {
            System.out.println("Stack is empty. No top element.");
            return;
        }
        System.out.println("Top element: " + q1.peek());
    }
    public void isEmpty() {
        System.out.println("Is stack empty? " + q1.isEmpty());
    }
  public static void main(String[] args) {
        task4a stack = new task4a();
        stack.push(10);
        stack.push(20);
        stack.push(30);
        stack.push(40);
        stack.pop();
        stack.top();
        stack.isEmpty();
        System.out.println("Elements in the stack: " + stack.q1);
    }
}
```

OUTPUT:



```
PS C:\Users\Varshinii\Documents\JAVASCRIPT\example1> java task4a.java
Pushing element: 10
Pushing element: 20
Pushing element: 30
Pushing element: 40
Popping element: 40
Top element: 30
Is stack empty? false
Elements in the stack: [30, 20, 10]
```

4b

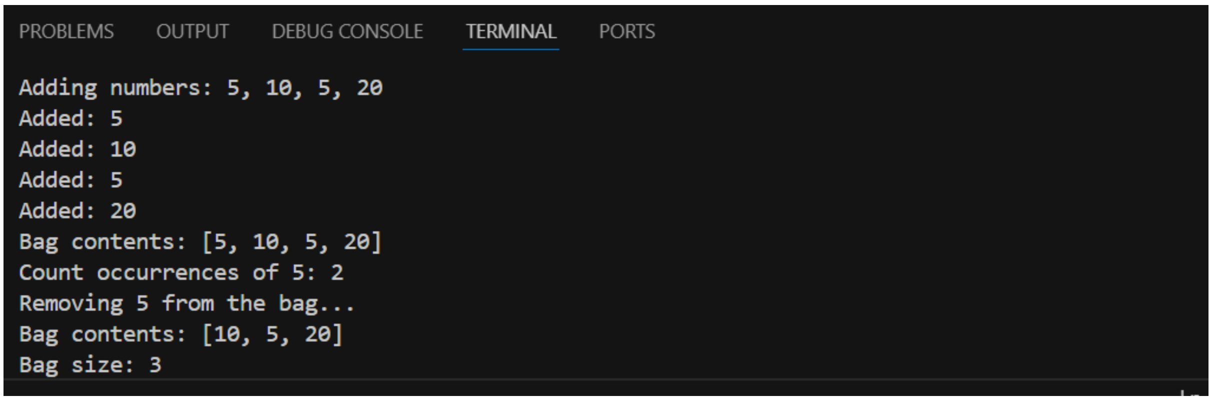
PROGRAM:

```
import java.util.ArrayList;

class BagOfNumbers {
    private ArrayList<Integer> bag = new ArrayList<>();

    public void add(int x) {
        bag.add(x);
        System.out.println("Added: " + x);
    }

    public void remove(int x) {
        if (bag.remove((Integer) x)) {
            System.out.println("Removing " + x + " from the bag...");
        } else {
            System.out.println("Number " + x + " not found in the bag.");
        }
    }

    public void countOccurrences(int x) {
        int count = 0;
        for (int num : bag) {
            if (num == x) count++;
        }
        System.out.println("Count occurrences of " + x + ": " + count);
    }

    public void isEmpty() {
        System.out.println("Is the bag empty? " + bag.isEmpty());
```

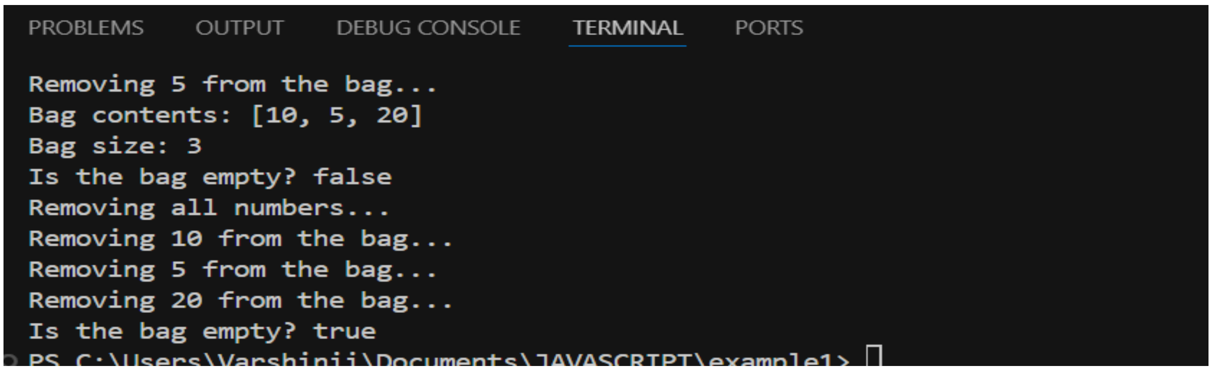
```
}

public void size() {
    System.out.println("Bag size: " + bag.size());
}

public void display() {
    System.out.println("Bag contents: " + bag);
}

public static void main(String[] args) {
    BagOfNumbers bag = new BagOfNumbers();

    System.out.println("Adding numbers: 5, 10, 5, 20");
    bag.add(5);
    bag.add(10);
    bag.add(5);
    bag.add(20);
    bag.display();

    bag.countOccurrences(5);
    bag.remove(5);
    bag.display();
    bag.size();
    bag.isEmpty();

    System.out.println("Removing all numbers...");
    bag.remove(10);
    bag.remove(5);
    bag.remove(20);
```

```
        bag.isEmpty();
    }
}
```

OUTPUT:



```
Adding numbers: 5, 10, 5, 20
Added: 5
Added: 10
Added: 5
Added: 20
Bag contents: [5, 10, 5, 20]
Count occurrences of 5: 2
Removing 5 from the bag...
Bag contents: [10, 5, 20]
Bag size: 3
```



```
Removing 5 from the bag...
Bag contents: [10, 5, 20]
Bag size: 3
Is the bag empty? false
Removing all numbers...
Removing 10 from the bag...
Removing 5 from the bag...
Removing 20 from the bag...
Is the bag empty? true
PS C:\Users\Varshinii\Documents\JAVASCRIPT\example1>
```

4.C

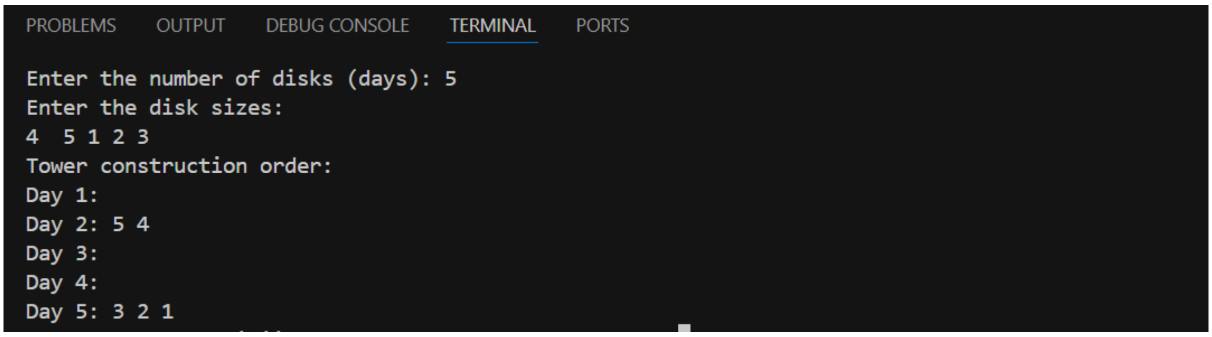
PROGRAM:

```
import java.util.*;


public class DiskTowerRecursion {
    static PriorityQueue<Integer> maxHeap = new
PriorityQueue<>(Collections.reverseOrder());
```

```
static int currentMax;

public static void solveTower(int[] disks, int day, int n) {
    if (day == n) return; // Base case: all days processed

    maxHeap.add(disks[day]);
    System.out.print("Day " + (day + 1) + ": ");

    while (!maxHeap.isEmpty() && maxHeap.peek() == currentMax) {
        System.out.print(maxHeap.poll() + " ");
        currentMax--;
    }
    System.out.println();

    solveTower(disks, day + 1, n);
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    System.out.print("Enter the number of disks (days): ");
    int n = sc.nextInt();
    int[] disks = new int[n];

    System.out.println("Enter the disk sizes:");
    for (int i = 0; i < n; i++) {
        disks[i] = sc.nextInt();
    }

    currentMax = Arrays.stream(disks).max().getAsInt();
```

```
        System.out.println("Tower construction order:");
        solveTower(disks, 0, n);
    }
}
```

OUTPUT:



```
Enter the number of disks (days): 5
Enter the disk sizes:
4  5 1 2 3
Tower construction order:
Day 1:
Day 2: 5 4
Day 3:
Day 4:
Day 5: 3 2 1
```