```java
/**
 * Java Helper Code for Mahat Exams (2023–2025)
 * Updated with solution-based logic from official exams and teacher keys
 */

import java.util.Scanner;
import java.util.Random;

public class JavaMahatHelper {

    static Scanner in = new Scanner(System.in);
    static Random rand = new Random();

    // === Input Examples ===
    public static void inputExamples() {
        int a = in.nextInt();
        double d = in.nextDouble();
        String s = in.next();
    }

    // === Output Example ===
    public static void outputExamples() {
        System.out.print("output");
    }

    // === Random Number ===
    public static int getRandom(int x, int y) {
        return x + rand.nextInt(y - x + 1);
    }

    // === Digit Sum (3-digit) ===
    public static int sumDigits(int num) {
        int sum = 0;
        while (num > 0) {
            sum += num % 10;
            num /= 10;
        }
        return sum;
    }

    // === Check if string is valid (same start/mid/end char) ===
    public static boolean isValid(String s) {
        if (s.length() % 2 != 0) return false;
        char a = s.charAt(0), b = s.charAt(s.length() / 2), c = s.charAt(s.length() - 1);
        return a == b && b == c;
    }

    // === Check if value appears again after index ===
    public static boolean isInArray(int[] arr, int ind, int value) {
        for (int i = ind + 1; i < arr.length; i++) {
            if (arr[i] == value) return true;
        }
```

```
    return false;
}

public static boolean isUniqueArray(int[] arr) {
    for (int i = 0; i < arr.length; i++) {
        if (isInArray(arr, i, arr[i])) return false;
    }
    return true;
}

// === String filter logic (e.g., includes X but not Y) ===
public static boolean includesXNotY(String str) {
    return str.contains("X") && !str.contains("Y");
}

// === Brother numbers ===
public static boolean brothers(int num1, int num2) {
    if (num1 <= 0 || num2 <= 0) return false;
    int msd1 = num1, msd2 = num2;
    while (msd1 >= 10) msd1 /= 10;
    while (msd2 >= 10) msd2 /= 10;
    return msd1 == msd2 && num1 % 10 == num2 % 10;
}

// === Palindrome checker ===
public static boolean isPalindrom(String str) {
    for (int i = 0; i < str.length() / 2; i++)
        if (str.charAt(i) != str.charAt(str.length() - 1 - i)) return false;
    return true;
}

public static boolean isLetter(char c) {
    return (c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z');
}

public static boolean isLetterPalindrom(String str) {
    int left = 0, right = str.length() - 1;
    while (left < right) {
        while (left < str.length() && !isLetter(str.charAt(left))) left++;
        while (right >= 0 && !isLetter(str.charAt(right))) right--;
        if (left >= right) break;
        if (str.charAt(left) != str.charAt(right)) return false;
        left++;
        right--;
    }
    return true;
}

public static String[] getPalArr(String[] arr) {
    int count = 0;
    for (String s : arr)
        if (isLetterPalindrom(s)) count++;
```

```
    String[] res = new String[count];
    int idx = 0;
    for (String s : arr)
        if (isLetterPalindrom(s)) res[idx++] = s;
    return res;
}

// === Class: Box ===
static class Box {
    private int width, height, length;
    private String color;
    private double weight;

    public Box(String color, double weight) {
        this.width = 1 + rand.nextInt(10);
        this.length = 1 + rand.nextInt(10);
        this.height = 1 + rand.nextInt(10);
        this.color = color;
        this.weight = weight;
    }

    public String getColor() { return color; }
    public double getWeight() { return weight; }
}

public static String heaviestBox(Box[] arr) {
    double max = 0;
    String color = "";
    for (Box b : arr) {
        if (b.getWeight() > max) {
            max = b.getWeight();
            color = b.getColor();
        }
    }
    return color;
}

public static Box heaviestColorBox(Box[] arr, String color) {
    double max = -1;
    int index = -1;
    for (int i = 0; i < arr.length; i++) {
        if (arr[i].getColor().equals(color) && arr[i].getWeight() > max) {
            max = arr[i].getWeight();
            index = i;
        }
    }
    return (index == -1) ? null : arr[index];
}

public static void main(String[] args) {
    System.out.println("Updated Mahat Java Helper Ready");
}
```

}