Weekly Assessment 5

Problem Statement 1:

Shift_Left_And_Fill_Zeros

Print a version of the given array where all the 10'shave been removed. The remaining elements should shift left towards the start of the array as needed, and the empty spaces at the end of the array should be 0. So {1,10,10,2} yields {1,2,0,0} You may modify and display the given array or make a new array.

Input Format

{1,10,10,2}

Constraints

Array Size > 0

Output Format

{1,2,0,0}

Sample Input 0

```
4
1 10 10 2
```

Sample Output 0

```
1 2 0 0
```

Code:

Problem Statement 2:

The method accepts two positive integers ‘r’ and ‘unit’ and a positive integer array ‘arr’ of size ‘n’ as its argument ‘r’ represents the number of rats present in an area, ‘unit’ is the amount of food each rat consumes and each ith element of array ‘arr’ represents the amount of food present in ‘i+1’ house number, where 0 <= i.

Note: Return -1 if the array is null Return 0 if the total amount of food from all houses is not sufficient for all the rats.

Input Format

7 – Number of rats 2 – Food needed for each rat 8 – Number of houses {2,8,3,5,7,4,1,2} – Food available in each house

Constraints

r > 0, unit > 0, n > 0

Output Format

4

Sample Input 0

```
7
2
8
2 8 3 5 7 4 1 2
```

Sample Output 0

```
4
```

Explanation of the Sample Input:

- We have 7 rats, and each rat requires 2 units of food, so the total food needed is 7 * 2 = 14 units.
- The food available in the houses is: [2, 8, 3, 5, 7, 4, 1, 2].
- Sorting this array in descending order gives: [8, 7, 5, 4, 3, 2, 2, 1].
- The total food collected from the first 4 houses is 8 + 7 + 5 + 4 = 24 units, which is sufficient to feed all the rats. Therefore, the answer is 4.

Explanation:
Total amount of food required for all rats = r * unit
= 7 * 2 = 14.
The amount of food in 1st houses = 2+8+3+5 = 18. Since, the amount of food in 1st 4 houses is sufficient for all the rats. Thus, output is 4.

Problem Statement 3:

Write a method to remove all repeated elements from an array. The resultant array should have only unique elements.

```
    int [] removeDuplicates (int [] array)      {
      // CODE

}
```

Input Format

Sample Input: 5 5 10 15 20 5 30

Constraints

n should be positive

Output Format

sample output: 5 10 15 20 30

Sample Input 0

```
6
5 10 15 20 5 30
```

Sample Output 0

```
5 10 15 20 30
```

Sample Input 1

```
5
1 2 3 4 5
```

Sample Output 1

```
1 2 3 4 5
```

Problem Statement 1:

Code:

```
import java.util.Scanner;

public class BubbleSort {

   public static void main(String[] args) {

       Scanner sc = new Scanner(System.in);

       // Read the number of elements

       int n = sc.nextInt();
```

```
int[] arr = new int[n];
// Read the array elements
for (int i = 0; i < n; i++) {
    arr[i] = sc.nextInt();
}
// Bubble Sort algorithm
for (int i = 0; i < n - 1; i++) {
    for (int j = 0; j < n - 1 - i; j++) {
        if (arr[j] > arr[j + 1]) {
            // Swap arr[j] and arr[j + 1]
            int temp = arr[j];
            arr[j] = arr[j + 1];
            arr[j + 1] = temp;
        }
    }
}
// Replace all 10s with 0
for (int i = 0; i < n; i++) {
    if (arr[i] == 10) {
        arr[i] = 0;
    }
}
// Print the modified array
for (int i = 0; i < n; i++) {
    System.out.print(arr[i] + " ");
```

```
        }

    }

}
```

Problem Statement 2:

Code:

```
import java.util.*;
class Main
{
    public static int solve (int r, int unit, int arr[], int n)
    {
        if (arr == null)
            return -1;
        int res = r * unit;
        int sum = 0;
        int count = 0;
        for (int i = 0; i < n; i++)
        {
            sum = sum + arr[i];
            count++;
            if (sum >= res)
                break;
        }
        if(sum<res)
            return 0;
        return count;
    }
    public static void main (String[]args)
    {
        Scanner sc = new Scanner (System.in);
        int r = sc.nextInt ();
        int unit = sc.nextInt ();
        int n = sc.nextInt ();
        int arr[] = new int[n];
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt ();
        System.out.println (solve (r, unit, arr, n));
    }
}
```

Problem Statement 3:

Code:

```
public class Main {
    // Bubble Sort and Remove Duplicates combined
    public static int remove(int[] a) {
        int n = a.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (a[j] > a[j + 1]) {
                    int temp = a[j];
                    a[j] = a[j + 1];
                    a[j + 1] = temp;
                }
            }
        }
        int j = 0;
        for (int i = 1; i < n; i++) {
            if (a[i] != a[j]) {
                a[++j] = a[i];
            }
        }
        return j + 1;
    }
    public static void main(String[] args) {
        int[] a = {1, 2, 3, 1, 4, 2, 1, 5};
        int n = remove(a);
        for (int i = 0; i < n; i++) {
            System.out.print(a[i] + " ");
        }
    }
}
```