1. **Remove 10's**

Write a program to read an integer array and remove all 10s from the array, shift the other elements towards left and fill the trailing empty positions by 0 so that the modified array is of the same length of the given array.

Include a class **UserMainCode** with a static method **removeTens**which accepts the number of elements and an integer array. The return type (Integer array) should return the final array.

Create a Class Main which would be used to read the number of elements and the input array, and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of n+1 integers, where n corresponds to size of the array followed by n elements of the array.

Output consists of an integer array (the final array).

Refer sample output for formatting specifications.

**Sample Input :**

5

1

10

20

10

2

**Sample Output :**

1

20

1. **Add and Reverse**

Given an int array and a number as input, write a program to add all the elements in the array greater than the given number. Finally reverse the digits of the obtained sum and print it.

Include a class **UserMainCode** with a static method “**addAndReverse**” that accepts 2 arguments and returns an integer.The first argument corresponds to the integer array and the second argument corresponds to the number.

Create a class **Main** which would get the required input and call the static method **addAndReverse** present in the UserMainCode.

**Example:**

Input Array = {10,15,20,25,30,100}

Number = 15

sum = 20 + 25 + 30 + 100 = 175

output = 571

**Input and Output Format:**

The first line of the input consists of an integer that corresponds to the number of elements in the array.

The next n lines of the input consists of integers that correspond to the elements in the array.

The last line of the input consists of an integer that corresponds to the number.

Output consists of a single integer.

1. **Triplets**

Given an integer array, Write a program to find if the array has any triplets. A triplet is a value if it appears 3 consecutive times in the array.  
  
Include a class UserMainCode with a static method **checkTripplets** which accepts an integer array. The return type is boolean stating whether its a triplet or not.  
  
Create a Class Main which would be used to accept the input arrayand call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of n+1 integers. The first integer would represent the size of array and the next n integers would have the values.  
  
Output consists of a string stating TRUE or FALSE.  
  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
7  
3  
3  
5  
5  
5  
2  
3  
  
**Sample Output 1:**  
TRUE  
  
**Sample Input 2:**  
7  
5  
3  
5  
1  
5  
2  
3  
  
**Sample Output 2:**  
FALSE

1. **Median Calculation**

Write a program to accept an int array as input, and calculate the median of the same.  
  
Median Calculation Procedure:

1. Sort the sequence of numbers.
2. The total number count is odd, Median will be the middle number.

    The total number count is even, Median will be the average of two middle numbers, After calculating the average, round the number to nearest integer.  
  
Include a class UserMainCode with a static method **calculateMedian** which accepts the int array. The return type is the integer which would be the median.  
  
Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a an integer which denotes the size of the array followed by the array of integers.  
Output consists of a integer.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
7  
1  
2  
1  
4  
7  
1  
2  
  
**Sample Output 1:**  
2  
  
**Sample Input 2:**  
6  
52  
51  
81  
84  
60  
88  
  
**Sample Output 2:**  
71

1. **Sequence in Array**

Write a program to accept an int array as input, and check if [1,2,3] appears somewhere in the same sequence.  
  
Include a class UserMainCode with a static method **searchSequence** which accepts the int array. The return type is a boolean which returns true or false.  
  
Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a an integer which denotes the size of the array followed by the array of integers.  
Output should print true or false.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
9  
11  
-2  
5  
1  
2  
3  
4  
5  
6  
  
**Sample Output 1:**  
TRUE  
  
**Sample Input 2:**  
6  
-2  
5  
1  
3  
2  
6  
  
**Sample Output 2:**  
FALSE

1. **Perfect Number**

Write a program to that takes a positive integer and returns true if the number is perfect number.  
  
A positive integer is called a perfect number if the sum of all its factors (excluding the number itself, i.e., proper divisor) is equal to its value.  
  
For example, the number 6 is perfect because its proper divisors are 1, 2, and 3, and 6=1+2+3; but the number 10 is not perfect because its proper divisors are 1, 2, and 5, and 1+2+5 is not equal to 10  
  
Include a class UserMainCode with a static method **getPerfection** which accepts the number. The return type is boolean (true / false).  
  
Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a integer.  
Output consists of TRUE / FALSE.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
28  
  
**Sample Output 1:**  
TRUE

1. **Find Digits**

For a given double number with atleast one decimal value, Write a program to compute the number of digits before and after the decimal point in the following format – noOfDigitsBeforeDecimal:noOfDigitsAfterDecimal.  
  
Note: Ignore zeroes at the end of the decimal (Except if zero is the only digit after decimal. Refer Example 2 and 3)  
  
Include a class UserMainCode with a static method **findNoDigits** which accepts the decimal value. The return type is string.  
  
Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a double.  
Output consists of string.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
843.21  
  
**Sample Output 1:**  
3:2  
  
**Sample Input 2:**  
20.130  
  
**Sample Output 2:**  
2:2  
  
**Sample Input 3:**  
20.130  
  
**Sample Output 3:**  
2:2

**8.Largest Span**

Write a program to read a integer array, find the largest span in the array.  
Span is the count of all the elements between two repeating elements including the repeated elements.  
Include a class UserMainCode with a static method **getLargestSpan** which accepts the integer array. The return type is integer.  
  
Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of an integer n which is the number of elements followed by n integer values.  
Output consists of integer.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
6  
4  
2  
1  
4  
5  
7  
**Sample Output 1:**  
4

**10. Even Sum & Duplicate Elements**

Write a program to read a integer array, Remove the duplicate elements and display sum of even numbers in the output. If input array contain only odd number then return -1.  
Include a class UserMainCode with a static method **sumElements** which accepts the integer array. The return type is integer.  
  
Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of an integer n which is the number of elements followed by n integer values.  
Output consists of integer.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
7  
2  
3  
54  
1  
6  
7  
7  
**Sample Output 1:**  
62  
  
**Sample Input 2:**  
6  
3  
7  
9  
13  
17  
21  
**Sample Output 2:**  
-1