1)Add Non Common Element:

Write a program to read two integer arrays and to add all the non common elements from the 2 integer arrays. Print the final output.

Example:
input1: [7,9,1,0]
input2: [10,6,5]
Output1:38

Business Rules:
Only positive numbers should be given to the input Lists.
1. If the input1 List consists of negative numbers, return -1.
2. If the input2 List consists of negative numbers, return -2.
3. If the both the input lists consists of negative numbers, return -3.

Include a class UserProgramCode with a static method sumNonCommonElement which accepts the inputs in the following order (input1, size1, input2, size2). The return type (integer) should return output according to the business rules.
Create a Class Program which would be used to accept two lists, and call the static method present in UserProgramCode.
Input and Output Format:
Input consists of n+m+2 integers, where first two integers corresponds to the size of the two array lists, respectively, followed by the corresponding array elements.
Output consists of an Integer(the corresponding output), or a String "Input 1 has negative numbers" if the first array contains negative numbers, "Input 2 has negative numbers" if the second array contains negative numbers, or "Both inputs has negative numbers" if both array has negative numbers.
Refer sample output for formatting specifications.

Sample Input 1:
4

3

6

9

2

1

10

7

5

Sample Output 1:

40

Sample Input 2:

4

3

-6

9

2

1

10

7

5

Sample Output 2:

Input 1 has negative numbers

Sample Input 3:

4

3

6

9

2

1

10

-7

5

Sample Output 3:

Input 2 has negative numbers

Sample Input 3:

4

3

6

9

-2

1

10

-7

5

Sample Output 3:

Both inputs has negative numbers

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication32
{

       public class UserProgramCode
       {
        public static int sumNonCommonElement(int[] ar1, int n, int[] ar2, int m)
        {
            int a = 0, b = 0;
            int[] temp = new int[m + n];
            for (int i = 0; i < n; i++)
            {
                if (ar1[i] < 0)
                    a = 1;

            }
            for (int j = 0; j < m; j++)
            {
                if (ar2[j] < 0)
                    b = 1;

            }
            if (a == 1 && b == 0)
            {
                return -1;
            }
            else if (a == 0 && b == 1)
            {
                return -2;
            }
            else if (a == 1 && b == 1)
            {
                return -3;
            }
            else if (a == 0 && b == 0)
            {
                for (int i = 0; i < n; i++)

                    for (int j = 0; j < m; j++)

                        if (ar1[i] == ar2[j])
                        {
                            ar1[i] = 0;
                            ar2[j] = 0;
                        }
            }
            return ar1.Sum() + ar2.Sum();
        }
    }

    class NonCommonElement
    {
        static void Main(string[] args)
        {

            int n = int.Parse(Console.ReadLine());
            int m = int.Parse(Console.ReadLine());
            int[] ar1 = new int[n];
            int[] ar2 = new int[m];
            for (int i = 0; i < n; i++)
                ar1[i] = int.Parse(Console.ReadLine());
            for (int i = 0; i < m; i++)
                ar2[i] = int.Parse(Console.ReadLine());
            int flag = UserProgramCode.sumNonCommonElement(ar1, n, ar2, m);
            if (flag == -1)
                Console.WriteLine("Input 1 has negative numbers");
            else if (flag == -2)
                Console.WriteLine("Input 2 has negative numbers");
            else if (flag == -3)
                Console.WriteLine("Both inputs has negative numbers");
            else
                Console.WriteLine(flag);
            Console.ReadLine();
        }
    }
}
```

2)Bonus Calculation.

Write a program to calculate the bonus of the employee given the basic salary of the employee. The bonus will be calculated based on the below category. If Basic Salary>15000 and less than 20001 calculate bonus as 17% of basic+1500 If Basic Salary>10000 and less than 15001 calculate bonus as 15% of basic+1200 If Basic Salary<10001 calculate bonus as 12% of basic+1000 for rest calculate bonus as 8%of basic+500 Business rule: 1) If the salary given is a negative number, then print -1. 2) If the salary given is more than 1000000, then print -2 . 3) All the test cases has the calculated bonus as integer value only. Create a class named UserProgramCode that has the following static method

public static int calculateBonus(int input1)

Create a class named Program that accepts the inputs and calls the static method present in the UserProgramCode.

Input and Output Format:

Input consists of an integer that corresponds to the salary.

Output is an integer.

Refer sample output and business rule for output formatting specifications.

Sample Input 1 :

10000

Sample Output1 :

2200

Sample Input 2 :
2000000
Sample Output 2 :
-2

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;


namespace bonus

{
        public class UserProgramCode
        {
         public static int CalculateBonus(int basic)
         {
             int bonus = 0;
             if (basic < 0)
                 return -1;
             if (basic > 1000000)
                 return -2;
             if (basic < 20001 && basic > 15000)
             {
                 bonus = Convert.ToInt32(basic * 0.17) + 1500;
             }
             if (basic > 10000 && basic < 15001)
                 bonus = Convert.ToInt32(basic * 0.15) + 1200;
             if (basic < 10001)
                 bonus = Convert.ToInt32(basic * 0.12) + 1000;
             else

                 bonus = Convert.ToInt32(basic * 0.08) + 500;
             return bonus;
         }
    }
}
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;


namespace bonus

{

    class Program
    {
        static void Main(string[] args)
        {
            int basic = int.Parse(Console.ReadLine());
            int op = UserProgramCode.CalculateBonus(basic);
            Console.WriteLine(op);
            Console.ReadLine();
        }
    }
}
```

3)Find nth Largest Number:

Write a method to find the nth largest number in an input integer array. Include a class UserProgramCode with a static method findNthLargestNumber which accepts 2 inputs, an integer array and an integer (n) and returns an integer. If the input consists of any negative numbers, the method returns -1. Else the method returns the nth largest element in the array.

Create a class Program which would get the input and call the static method findNthLargestNumber present in the UserProgramCode. If the method returns -1, then print 'Invalid Input'.

Input and Output Format: The first line of the input consists of an integer that corresponds to m, the size of the array. The next m lines of input consists of integers that correspond to the elements in the array. The next line of input consists of integer that corresponds to 'n'. Refer sample output for formatting specifications.

Sample Input 1: 7 2 1 67 10 55 12 7 -2

Sample Output 1: Invalid Input Sample Input 2: 7 100 300 150 450 650 50 25 4

Sample Output 2: 150

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace TestPractice
{
public class UserProgramCode
    {
        public static int output1;
        public static int nthLargest(int[] input1, int input2)
        {
            if (input2 < 1)
            {
                return -1;
            }
            foreach (var item in input1)
            {
                if (item < 0)
                {
                    output1 = -1;
                    return output1;
                }
            }
            if (output1 != -1)
            {
                Array.Sort(input1);
                Array.Reverse(input1);
                input1 = input1.Distinct().ToArray();
                output1 = input1[input2 - 1];
            }
            return output1;
        }
    }
```

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;


namespace TestPractice

{

class Program1
    {
        static void Main(string[] args)
        {
            int n;
            n = Convert.ToInt32(Console.ReadLine());
            int[] input1 = new int[n];
            for (int i = 0; i < n; i++)
            {
                input1[i] = Convert.ToInt32(Console.ReadLine());
            }
            int input2 = Convert.ToInt32(Console.ReadLine());
            int res = UserProgramCode.nthLargest(input1, input2);
            if (res == -1)
            {
                Console.WriteLine("invalid input");
            }
            else
            {
                Console.WriteLine(res);
            }
            Console.ReadLine();
        }
    }

}
```

4)Sum Common Element:

Write a program to read two int arrays, eg. A{2,3,5,1} and B{1,3,9}, and to find out sum of common elements in given arrays. Print the sum, or print “No common elements found” if there are no common elements.

Assume the common element appears only once in each array.

Include a class UserProgramCode with a static method getSumOfIntersection which accept the size of two integer arrays and the two integer arrays. The return type (integer) should return the sum, or -1, accordingly.

Create a Class Program which would be used to accept two integer arrays, and call the static method present in UserProgramCode.

Input and Output Format:

Input consists of n+m+2 integers, where first two integers corresponds to the size of the two array lists, respectively, followed by the corresponding array elements.

Output consists of an Integer(the corresponding output) or string - (“No common elements found”).

Refer sample output for formatting specifications.

Sample Input 1:

4

3

2

3

5

1

1

3

9

Sample Output 1:

4

Sample Input 2:

4

3

2

31

5

14

1

3

9

Sample Output 2:

No common elements found

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;


namespace Fwd_Prgs

{

   public class UserProgramCode

   {

public static int getSumOfIntersection(int n1, int n2, int[] a, int[] b)
        {
            int sum = 0;
            for (int i = 0; i < n1; i++)
            {
                for (int j = 0; j < n2; j++)
                    if (a[i] == b[j])
                        sum = sum + a[i];

            }
            if (sum == 0)
                return -1;
            else
                return sum;
        }

    }

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;


namespace Fwd_Prgs
```

```
{
class Program3
    {
        static void Main(string[] args)
        {
            int n1 = int.Parse(Console.ReadLine());
            int n2 = int.Parse(Console.ReadLine());
            int[] a = new int[n1];
            int[] b = new int[n2];
            for (int i = 0; i < n1; i++)
                a[i] = int.Parse(Console.ReadLine());
            for (int i = 0; i < n2; i++)
                b[i] = int.Parse(Console.ReadLine());
            int res = UserProgramCode.getSumOfIntersection(n1, n2, a, b);
            if (res == -1)
                Console.WriteLine("No common elements found");
            else
                Console.WriteLine(res);
            Console.ReadLine();
        }
    }

}
```

5)Sort the list:

Write a program which reads an Integer(size of the list), a String List and a character, and to get the strings that will not start with the given character irrespective of case. Sort the elements in ascending order based on its length. Print the output list. If the elements are having the same length, then display the elements in alphabetical order.
Include a class UserProgramCode with static method GetTheElements which accepts a String list and a character. The return type is List<String>.
Create a Class Program which would be used to get the inputs and call the static method present in UserProgramCode. In GetTheElements method
Only alphabets should be given in list , otherwise return "Invalid Input". When the output list is empty, then return "List is Empty". Otherwise return the appropriate result.
In Program class Print the result which is return by GetTheElements method in UserProgramCode. Input output format The first line of the input is an integer that corresponds to n, the size of the list. The next n lines of input correspond to the elements in the string list. The line of the input contains the character. The output is the List type List<String>. Sample Input 1: 3 read write edit e Sample Output 1: read write Sample Input2 : 2 Elegent event e Sample Output2 : List is Empty Sample Input 3: 2 Eleg$ent e^ent e Sample Output 3 : Invalid Input

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Text.RegularExpressions;


namespace ConsoleApplication22
```

```
{
  class UserProgramCode
    {
        public static List<string> GetTheElements(string[] t, char f)
        {
            Regex r = new Regex(@"^([a-zA_Z]{1,})$");
            List<string> m = new List<string>();
            foreach (string c in t)
            {
                if (!r.IsMatch(c))
                {
                    m.Add("-1");
                    break;
                }
                else
                {
                    if (!c.StartsWith(f.ToString()))
                    {
                        m.Add(c);
                    }
                }
            }
            for (int i = 0; i < m.Count; i++)
            {
                for (int j = i + 1; j < m.Count; j++)
                {
                    if (m[i].Length == m[j].Length)
                    {
                        m.Sort();
                    }
                    else
                    {
                        if (m[j].Length < m[i].Length)
                        {
                            m.Reverse();
                        }
                    }
                }
            }
            return m;
        }
    }

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication22
{
class Program2
    {
        static void Main(string[] args)
        {
            int n = int.Parse(Console.ReadLine());
            string[] p = new string[n];
            for (int i = 0; i < n; i++)
            {
                p[i] = Console.ReadLine();
            }
            char e = char.Parse(Console.ReadLine());
            List<string> l = new List<string>();
            l = UserProgramCode.GetTheElements(p, e);
            if (l.Contains("-1"))
            {
                Console.WriteLine("Invalid Input");
            }
            else if (l.Count > 0)
            {
                foreach (string x in l)
                {
                    Console.WriteLine(x);
                }
            }
            else
                Console.WriteLine("list is empty");
            Console.WriteLine();
        }
    }
}
```

6)Image Types:

Given a string array input which consists of image file names along with their respective image type extensions in the format ("filename.extensiontype",..so on). The image file name and the extension are seperated by a dot (.)operator. Write a program to calculate the count of image files having same extension type and store the values in the output string array variable in the below format. output (Key,Value) = (ExtensionType1,count1,ExtensionType2,count2,...so on) . Output should be stored in descending order based on the count of image files having the same extension type. Note: jpeg,jfif,exif,tiff,raw,gif,bmp,png are the various types of image file extensions Business Rules: 1)If all the elements of the input array do not have image type extension, then print -1. 2)If any of the file name doesn't contain extension type or if the extension is not an image type then it will be treated as other type, take the count of all such files and store as the last element in the sorted output array with key element as "others" and value element as the calculated count. 3)If more than one key element have same count, then store the key and their respective value element in the order given in input.

Create a class named UserProgramCode that has the following static method

public static List<string> countImageTypes(string[] input1)

Create a class named Program that accepts the inputs and calls the static method present in the UserProgramCode.

Input and Output Format:

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

The next 'n' lines of input correspond to elements in the input array.

Refer business rules and sample output for formatting specifications. Sample Input 1 :

4 Employee.jpeg Purchase.jpeg stock.jpeg book.gif

Sample Output 1 : jpeg 3 gif 1 Sample Input 2 :

7

Sales.doc Employee.jpeg Purchase.jpeg image.png stock.jpeg book.gif pen

Sample Output 2 : jpeg 3 png 1 gif 1 others 2

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;



namespace image

{

 class userProgramcode

    {

            public static List<string> imagescount(List<string> input1)
            {
                int k = 0, ctr = 0;
                string[] ar = new string[8] { "jpeg", "jfif", "exif", "tiff", "raw",
"gif", "bmp", "png" };
                List<string> outp = new List<string>();
                string[] st = new string[input1.Count];
                int[] sco = new int[input1.Count];
                for (int j = 0; j < input1.Count; j++)
                {
                    string[] arr = input1[j].Split('.');
                    if (arr.Length == 2 && !st.Contains(arr[1]) && ar.Contains(arr[1]))
                    {

                        st[k] = arr[1];
                        sco[k] = sco[k] + 1;
                        k++;
                    }
                    else if (arr.Length == 2 && st.Contains(arr[1]) &&
ar.Contains(arr[1]))
                    {
                        for (int p = 0; p < st.Length; p++)
                        {
                            if (st[p] == arr[1])
                            {
                                sco[p] = sco[p] + 1;
                                break;
                            }
                        }
                    }
                    else
                    {
                        ctr++;

                    }
                }

                if (ctr != input1.Count)
                {
                    int[] co = new int[k];
                    co = sco.ToArray();
                    sco = co.Distinct().ToArray();
                    Array.Sort(sco);
                    Array.Reverse(sco);

                    for (int m = 0; m < sco.Length - 1; m++)
                    {
                        for (int n = 0; n < co.Length; n++)
                        {
                            if (sco[m] == co[n])
                            {
                                outp.Add(st[n]);
                                outp.Add(sco[m].ToString());
                            }
                        }
                    }
                }
                if (ctr != 0 && ctr != input1.Count)
                {
                    outp.Add("Others");
                    outp.Add(ctr.ToString());
                }
                else if (ctr == input1.Count)
                {
                    outp.Add("-1");
                    return outp;
                }

                return outp;
            }
        }
```

using System;

```
using System.Collections.Generic;

using System.Linq;

using System.Text;



namespace image

{

class Program4
    {
        static void Main(string[] args)
        {
            int a = int.Parse(Console.ReadLine());
            List<string> image = new List<string>();
            for (int i = 0; i < a; i++)
            {
                image.Add(Console.ReadLine());
            }
            List<string> ouyp = userProgramcode.imagescount(image);

            foreach (string s in ouyp)
            {

                Console.WriteLine(s);
            }
            Console.ReadLine();

        }

    }

}
```