**151. Remove Duplicates**    
 Given a string as input, write a program to remove duplicate characters from the string. Note - Only the first occurrence of each character should be retained. Retain all blank spaces. Business Rule: If there is no duplicate found, print -1.  
    
 Create a class named UserProgramCode that has the following static method   
 public static string removeDuplicates(string 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 a string.  
 Refer business rules and sample output for formatting specifications.

Sample Input :  
 hi this is sample test  
    
 Sample Output : hi ts ample

using System;

using System.Linq;

using System.Collections.Generic;

using System.Text;

class Program

{

public static void Main(string[] args)

{

string input;

input = Console.ReadLine();

string outputStr = UserProgramCode.removeDuplicates(input);

Console.WriteLine(outputStr);

Console.ReadLine();

}

}

############################################

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

class UserProgramCode

{

public static string removeDuplicates(string input1)

{

//char[] ch = input1.ToCharArray();

//char[] res = ch.Distinct().ToArray();+

int i = 0,j=0;

//int len = input1.Length;

//char c='\*';

for (i = 0; i < input1.Length; i++)

{

for (j = i + 1; j < input1.Length; j++)

{

if (!(input1[j] == ' '))

{

if (input1[i] == input1[j])

{

input1 = input1.Remove(j, 1);

}

}

}

}

//input1.Remove(4, 1);

string output = input1;

//Console.Read();

return output;

}

}

**152.Sum Of Squares Of Even Digits**  
 Write a program to read a positive integer and to calculate the sum of squares of even digits available in the given number. Print the output.  
    
 Example:   
 input = 56895  
 output = 6\*6 + 8\*8 = 100  
    
 Include a class UserProgramCode with a static method sumOfSquaresOfEvenDigits which accepts an Integer. The return type (Integer) should return the sum of squares of even digits available in the given number.  
 Create a Class Program which would be used to accept an Integer, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of an Integer.  
 Output consists of an Integer, the sum of squares of even digits available in the given number .  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 56895  
 Sample Output 1:  
 100

class Program

{

static void Main(string[] args)

{

int i=Convert.ToInt32(Console.ReadLine());

int s1 = UserProgramCode.sumOfSquareofEvenDigits(i);

Console.WriteLine(s1);

Console.ReadLine();

}

}

class UserProgramCode

{

public static int sumOfSquareofEvenDigits(int a)

{

int sum=0;

string a1 = a.ToString();

int a2 = 0;

for (int i = 0; i < a1.Length; i++)

{

a2 = (int)(a1[i])-48;

if (a2 % 2 == 0)

{

sum = sum + (a2 \* a2);

}

}

return sum;

}

}

**153 .Validate ID Locations**  
 Given a method with two string inputs. First string is ID and second string is location. ID is in the format CTS-LLL-XXXX where LLL is the first three letters of given location and XXXX is a four digit number. If the given ID meets the given format return 1 else return -1 . Example: Input1 = CTS-hyd-1234 Input2 = hyderabad output = 1  
 Create a class named UserProgramCode that has the following static method   
 public static int validateIDLocations(string input1, string input2)   
 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 2 strings.  
 Output is an integer.  
    
 Sample Input : CTS-hyd-1234  
 hyderabad    
 Sample Output :  
 1  
    
  using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace idValidation

{

class Program

{

static void Main(string[] args)

{

string s1 = Console.ReadLine();

string s2 = Console.ReadLine();

int i = UserProgramCode.validateIDlocations(s1, s2);

if (i == 1)

{

Console.WriteLine("valid");

}

else

{

Console.WriteLine("Invalid");

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace idValidation

{

class UserProgramCode

{

public static int validateIDlocations(string s1, string s2)

{

int output = 0;

Regex reg = new Regex(@"^([CTS]+[-]+([A-Za-z]{3})+[-]+([0-9]{4}))$");

if (reg.IsMatch(s1))

{

string res = s2.ToLower();

if (s1.Contains(res.Substring(0, 3)))

{

output = 1;

}

else

{

output = -1;

}

}

else

{

output = -2;

}

return output;

}

}

}

**154.Vowels**  
 A string is said to be valid if it contains exactly five vowels in any order. Assume there is no repetition of any vowel in the given string.  
    
 Example:  
 Input : acbisouzze  
 Output: Valid  
    
 Include a class UserProgramCode with a static method testVowels that accepts a string and returns an integer. The method returns 1 if the string is valid. Else it returns -1.  
    
 Create a class Program which would get the input and call the static method testVowels() present in the UserProgramCode.  
    
 If there are exactly five vowels present in the string then print "Valid" else print as "Invalid".  
 Sample Input 1: education Sample Output 1: Valid Sample Input 2: vowels Sample Output 2: Invalid  
    
    
  Vowels

using System;

using System.Text;

using System.Linq;

public class UserMainCode

{

public static int testVowels(string str)

{

if ((str.Contains('a') || str.Contains('A')) && (str.Contains('e') || str.Contains('E')) && (str.Contains('i') || str.Contains('I')) && (str.Contains('o') || str.Contains('O')) && (str.Contains('u') || str.Contains('U')))

{

return 1;

}

else

{

return -1;

}

}

}

using System;

public class Program {

public static void Main(){

string st = Console.ReadLine();

int result = UserMainCode.testVowels(st);

if (result == 1)

Console.WriteLine("Valid");

else

Console.WriteLine("Invalid");

Console.Read();

}

}

**155.Count the number of Occurrences** Write code to read two lines(String) and count the number of occurrences of second word of second sentence in the first sentence. Print the count.   
    
 Note - Consider case.  
    
 Example:  
 Input1: Hi this is cognizant Academy  
 Input2: Hello this is a trainee  
 Output: 1  
    
 Business Rule:   
 If there is no occurrence of the second word of second sentence with respect to the first sentence, return 0.  
    
 Include a class UserProgramCode with a static method countNoOfWords which accepts two strings. The return type (integer) should return the count.  
 Create a Class Program which would be used to accept two strings and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of two strings.  
 Output consists of an integer (the count).  
 Refer sample output for formatting specifications.  
 Sample Input 1:  
 Hi this is cognizant Academy  
 Hello this is a trainee  
 Sample Output 1:  
 1

m using System;

class UserProgramCode

{

public static int countNoOfWords(string str1,string str2)

{

string[] sub1=str1.Split(' ');

string[] sub2 = str2.Split(' ');

int ctr=0;

string str = sub2[1];

for (int i = 0; i < sub1.Length; i++)

{

if (sub1[i] == str)

{

ctr++;

}

}

return ctr;

}

}

using System;

class Program

{

public static void Main( string[] args )

{

string str1=Console.ReadLine();

string str2=Console.ReadLine();

Console.WriteLine(UserProgramCode.countNoOfWords(str1,str2));

Console.ReadLine();

}

}

**156.Index power array** Write code to read an integer array and to find the power of each individual element according to its position index, add them up and print as output.  
    
 Example :   
 input = {7,6,2,1}   
 output = (7 power 0)+(6 power 1)+(2 power 2)+(1 power 3) = 1+6+4+1=12  
    
 Include a class UserProgramCode with a static method getSumOfPower which accepts an integer that corresponds to the size of the array  and an integer array. The return type (Integer) should return the final output.  
 Create a Class Program which would be used to accept Input array and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of n+1 integers, where the first integer corresponds to the number of elements, followed by the array elements.  
 Output consists of an Integer(final output).  
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 4  
 7  
 6  
 2  
 1  
 Sample Output 1:  
 12

6)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Reflection;

namespace ConsoleApplication2

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

if (n > 0)

{

int[] a = new int[n];

for(int i=0;i<n;i++)

{

a[i] = int.Parse(Console.ReadLine());

}

int sum = UserProgramCode.getSumOfPower(n, a);

Console.WriteLine(sum);

}

}

}

}

class UserProgramCode

{

public static int getSumOfPower(int size,int[] a)

{

int sum=0;

for(int i=0;i<size; i++)

{

sum+=(int)Math.Pow(a[i],i);

}

return sum;

}

}

**157.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

2>>using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace TestPractice

{

public class UserMainCode

{

public static int output1;

public static int nthLargest(int[] input1, int input2)

{

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 Program

{

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=UserMainCode.nthLargest(input1,input2);

if(res==-1)

{

Console.WriteLine("invalid input");

}

else

{

Console.WriteLine(res);

}

Console.ReadLine();

}

}

}

**158.Validate Phone Number**  
 Write a program to read a phone number as a string input and to verify the phone number using following business rules:   
 -it should contain only numbers or dashes (-)  
 -dashes may appear at any position  
 -should have exactly 10 digits  
 If the Phone number is valid print “Valid” otherwise print “Invalid”.  
    
 Example:  
 input = 265-265-7777  
 output = Valid  
    
 Include a class UserProgramCode with a static method validatePhoneNumber which accepts a String. The return type (Integer) should return 1 if valid, else return 2.  
 Create a Class Program which would be used to accept a String, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of a String, which corresponds to the phone number.  
 Output consists of a String, “Valid” if the phone number is valid, else “Invalid”.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 265-265-7777  
 Sample Output 1:  
 Valid  
    
 Sample Input 2:  
 1111-111-1111  
 Sample Output 2:  
 Invalid

47.Validate phone number

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication18

{

class UserProgramCode

{

public static int validatephonenumber(string s)

{

int len = s.Length;

int digit = 0, flag = 0;

char[] a = s.ToCharArray();

for (int i = 0; i < len; i++)

{

if ((a[i] == '-') || char.IsDigit(a[i]))

flag++;

}

if (flag == len)

{

for (int i = 0; i < len; i++)

{

if (char.IsNumber(a[i]))

{

digit++;

}

}

}

if (digit == 10)

return 1;

else

return 2;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication18

{

class Program

{

static void Main(string[] args)

{

UserProgramCode u = new UserProgramCode();

string s;

int result;

s = Console.ReadLine();

result = UserProgramCode.validatephonenumber(s);

if(result==1)

Console.WriteLine("valid");

else if(result==2)

Console.WriteLine("Invalid");

}

}

}

=========================================================

S

**159.Get Big Diff**   
    
 Write a program that reads an integer array and finds the difference between the largest element and smallest element in the array.  
 Include a class UserProgramCode with a static method getBigDiff that accepts an integer array and returns an integer.  
    
 Create a Class Program which would be used to read the integer array and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array.  
 The next 'n' integers correspond to the elements in the array.  
 Output consists of an integer.  
    
 Sample Input 1: 4 10 3 5 6 Sample Output 1: 7 Sample Input 2: 4 2 -10 7 -2 Sample Output 2: 17

50.Get Big Diff

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication18

{

class Program

{

static void Main(string[] args)

{

UserProgramCode u=new UserProgramCode();

int n = int.Parse(Console.ReadLine());

int[] a = new int[n];

int result;

for (int i = 0; i < n; i++)

a[i] = int.Parse(Console.ReadLine());

result = UserProgramCode.getBigDiff(a);

Console.WriteLine(result);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication18

{

class UserProgramCode

{

public static int getBigDiff(int[] a)

{

int min = a[0],max=a[0],diff=0;

for (int i = 0; i < a.Length; i++)

{

if (a[i] >= max)

max = a[i];

if (a[i] <= min)

min = a[i];

}

diff= max - min;

return diff;

}

}

}

**160.Sum Common Elements**  
 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

18)

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;

}

}

class Program

{

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);

}

}

}

**161.Quadratic Equation**  
 Consider two equations x^2 - 2y + z = 0 and x^2+y=40 . Given an input integer array input1 containing values for the variable 'x', write a program to apply the input values to the equations and find out the corresponding y and z values to store them in the output in the following format (y1,z1,y2,z2,....} and so on. Print the output array Output array elements can contain negative values. Business rule: 1) If any of the elements in input1 array is negative, then print -1. 2) If there are any duplicates found in input1 array, then print -2. 3) If size of the input1 array is 1 or greater than 10, then print -3. Example 1: input1 : {1,2,4,5} output1: {39,77,36,68,24,32,15,5} Example 2: input1 : {5,8,3,-4,6} output1: {-1}    
 Create a class named UserProgramCode that has the following static method   
 public static int[] quadEquation(int[] 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 consist of elements in the input array.  
 Refer business rules and sample output for formatting specifications. Sample Input 1 :  
 4  
 1  
 2  
 4  
 5  
 Sample Output 1 :  
 39  
 77  
 36  
 68  
 24  
 32  
 15  
 5 Sample Input 2 : 5  
 5  
 8  
 3  
 -4  
 6    
 Sample Output 2 :  
 -1

using System;

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

namespace ConsoleApplication9

{ class Program

{ static void Main(string[] args)

{

int[] arr=new int[5];

Console.WriteLine("Enter the numbers of the array");

for (int i = 0; i < arr.Length; i++)

{

arr[i] = Convert.ToInt32(Console.ReadLine());

}

int[] result=sourav(arr);

for (int i = 0; i < result.Length; i++)

{

Console.WriteLine(result[i]+",");

}

Console.ReadLine();

}

public static int[] sourav(int[] arr)

{

List<int> li = new List<int>(); int[] arr1 = new int[]{-1}; int[] arr2 = new int[] { -2 }; List<int> li1 = new List<int>();

for (int i = 0; i < arr.Length; i++)

{

if (arr[i] < 0)

{

return arr1;

}

}

for (int i = 0; i < arr.Length; i++)

{

if (!li1.Contains(arr[i]))

{

li1.Add(arr[i]);

} else {

return arr2;

}

}

int y = 0,z=0;

for (int i = 0; i < arr.Length; i++)

{

y = (int)(40 - Math.Pow(arr[i], 2)); li.Add(y);

z = (int)((2 \* y) - Math.Pow(arr[i], 2)); li.Add(z);

} return li.ToArray();

}

}

}

Method 2:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace quadratic

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

int[] xarr = new int[n];

for (int i = 0; i < n; i++)

{

xarr[i] = int.Parse(Console.ReadLine());

}

int[] op = UserProgramCode.quadEquation(xarr);

int len = op.Length;

if (op[0] == -1)

{

Console.WriteLine("-1");

}

else if (op[0] == -2)

{

Console.WriteLine("-2");

}

else if (op[0] == -3)

{

Console.WriteLine("-3");

}

else

for (int i = 0; i < op[len - 1]; i++)

{

Console.WriteLine(op[i]);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace quadratic

{

class UserProgramCode

{

public static int[] quadEquation(int[] input1)

{

int z= 0;

int y = 0;

int k = 0;

int flag = 1;

int[] temp = new int[30];

for (int i = 0; i < input1.Length; i++)

{

if (input1[i] < 0)

{

temp[0] = -1;

flag = 0;

}

}

if (flag == 1)

{

if (input1.Length < 2 || input1.Length > 10)

{

temp[0] = -3;

flag = 0;

}

}

if (flag == 1)

{

for (int i = 0; i < input1.Length; i++)

{

for (int j = i + 1; j < input1.Length; j++)

{

if (input1[i] == input1[j])

{

temp[0] = -2;

flag = 0;

}

}

}

}

if (flag == 1)

{

for (int i = 0; i < input1.Length; i++)

{

z = 0;

y = 0;

y = 40 - (input1[i] \* input1[i]);

z = (2 \* y) - (input1[i] \* input1[i]);

temp[k] = y;

temp[k + 1] = z;

k = k + 2;

}

}

if(flag==1)

temp[29] = k;

return temp;

}

}

}

**162. Number System** Given a number system having numbers which is a combination of digits 3 and 4 only. First few numbers in the number system are: 3, 4, 33, 34, 43, 44, 333, 334, 343, 344, 433, 434, 443, 444, 3333, 3334, 3343, 3344, 3433, 3434, 3443, 3444, … Find the nth number in the number system where n is an integer index given as input . Business Rule: 1. If the input1 is less than 1, then print -1 Create a class named UserProgramCode that has the following static method   
 public static void findNumber(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 is an integer.  
 Output is an integer.  
    
 Sample Input 1 : 10

Sample Output1 :  
 344

Sample Input 2 :  
 -8

Sample Output 2 :  
 -1

**163.File Extension** Write a program to read a file name as a string and find out the file extension and return it as output. For example, the file sun.gif has the extension gif.  
    
 Include a class UserProgramCode with a static method fileIdentifier which accepts a string. The return type (string) should return the extension of the input string (filename).  
 Create a Class Program which would be used to accept Input String and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of a string that corresponds to a file name.  
 Output consists of a string(extension of the input string (filename)).  
 Refer sample output for formatting specifications.  
 Sample Input 1:  
 sun.gif  
 Sample Output 1:  
 gif

class Program

{

static void Main(string[] args)

{

string s;

s = Console.ReadLine();

userprogramcode obj = new userprogramcode();

Console.WriteLine(obj.fileIdentifier(s));

}

}

public class userprogramcode

{

public string fileIdentifier(string s)

{

string[] str;

str=s.Split('.');

return str[1];

}

}

164.Calculate Bill Amount

A Electrical shop has announced the following seasonal discount for the purchase of certain items. IInclude a class UserProgramCode with a static method calculateBillAmount which accepts double and char value as input and returns an integer that corresponds to the net amount. Compute the net amount to be paid by the customer based on the below criteria, Purchase Amount(Rs) Discount on Tv Discount on MusicSystem 1-25000 5% 10% 25001-50000 10% 20% More than 50000 15% 30% [Hint: DiscountPrice=(DiscountRate/100)\*Amount of Purchase. Net Amount=Amount of Purchase-DiscountPrice.] If the purchase item is other than TV and MusicSystem return -2, if the purchase amount is a negative value return -1.Otherwise return the net amount. Create a Main class which gets double and char as an input and call the static method calculateBillAmount present in the UserProgramCode. If the method returns -1, then print 'Negative Values'. If the method returns -2, then print 'No Items'. Input and  output format: The input will be a double and char values. If the input character is 'T', it corresponds to TV. If the input character is 'M', it corresponds to Music System.

Refer sample output for formatting specifications.

Sample Input 1: 20000 X

Sample Output 2: No Items

Sample Input 2: -5000 M

Sample Output 2: Negative Values

Sample Input 3: 70000 T

Sample Output 3: 59500

calculate bill amount

using System;

class Program

{

public static void Main( string[] args )

{

double input1;

char input2;

input1 = Convert.ToDouble(Console.ReadLine());

input2 = Convert.ToChar(Console.ReadLine());

int value = UserProgramCode.calculateBillAmount(input1,input2);

if(value == -2)

Console.Write("No Items");

else if(value == -1)

Console.Write("Negative Values");

else

Console.Write(value);

Console.ReadLine();

}

}

using System;

class UserProgramCode

{

public static int calculateBillAmount(double input1, char input2)

{

double dp;

double na = 0;

char[] ch = new char[] { 'T', 'M' };

if (input1 < 0)

{

return -1;

}

if ((input2 != 'T') && input2 != 'M')

{

return -2;

}

if (input1 >= 1 && input1 <= 25000)

{

if (input2 == 'T')

{

dp = (0.05 \* input1);

na = (input1 - dp);

return (int)na;

}

else if (input2 == 'M')

{

dp = (0.1 \* input1);

na = (input1 - dp);

return (int)na;

}

else

{

Console.WriteLine("not valid");

}

}

else if (input1 >= 25001 && input1 <= 5000)

{

if (input2 == 'T')

{

dp = (0.1 \* input1);

na = (input1 - dp);

return (int)na;

}

else if (input2 == 'M')

{

dp = (0.2 \* input1);

na = (input1 - dp);

return (int)na;

}

else

{

Console.WriteLine("not valid");

}

}

if (input1 >= 50000)

{

if (input2 == 'T')

{

dp = (0.15 \* input1);

na = (input1 - dp);

return (int)na;

}

else if (input2 == 'M')

{

dp = (0.3 \* input1);

na = (input1 - dp);

return (int)na;

}

else

{

Console.WriteLine("not valid");

}

}

return (int)na;

}

}

**165.Calculate Frequency**  
 Given two string inputs input1 and input2, write a program to find the number of times the complete string in input1 occurs in input2 and print the count. Ignore case sensitiveness in the input strings. Business Rules: 1)If input1 has repeated words, print -1. 2)If the count of occurrence is zero then print -2 . Example1: input1: A good place input2: It is a good place to be in and a good place to have fun. output: 2 Example: input1 :Does he have to have a car ? input2: Yes he should. output: -1 Create a class named UserProgramCode that has the following static method  
 public static int calcFrequency(string input1, string input2)  
 Create a class named Program that accepts the input and calls the static method present in the UserProgramCode. Input and Output Format: Input consists of 2 strings.  
 Output consists of an integer.  
 Sample Input : A good place It is a good place to be in and a good place to have fun.  
 Sample Output : 2

77.

//program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace oops2

{

class Program

{

static void Main(string[] args)

{

String input1 = Console.ReadLine();

String input2 = Console.ReadLine();

int ret = UserProgramCode.calcFrequency(input1, input2);

Console.WriteLine(ret);

//Console.WriteLine(input1.ToLower());

}

}

}

//UserProgramCode.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace oops2

{

class UserProgramCode

{

public static int calcFrequency(string input1, string input2)

{

StringBuilder sb = new StringBuilder();

int val = 0;

String[] arr = input1.Split(' ');

foreach (String a in arr)

{

sb.Append(a.ToLower());

}

int len = arr.Length;

for (int i = 0; i < len; i++)

{

String s = arr[i];

for (int j = i + 1; j < len; j++)

{

if (s.Equals(arr[j]))

{

val++;

return -1;

}

}

}

if (val == 0)

{

int count = 0;

StringBuilder sb1 = new StringBuilder();

String linput1 = input1.ToLower();

String linput2 = input2.ToLower().Replace('.',' ');

// Console.WriteLine(linput2);

String[] l = linput2.Split(' ');

foreach (String a in l)

{

sb1.Append(a);

}

len = sb.Length;

int len2 = sb1.Length;

// Console.WriteLine(len + "" + len2);

for (int i = 0; i < len2; i++)

{

if(i<len2-len)

{

// Console.WriteLine(i+","+len);

// Console.WriteLine(sb);

// Console.WriteLine(sb1);

String sub = sb1.ToString().Substring(i, len);

if (sub.Equals(sb.ToString()))

{

count++;

}

}

}

if (count == 0)

return -2;

else

return count;

}

return 0;

}

}

}  
    
**166.Power of 2**    
 Write a program to check whether an integer number is a power of 2 or not. If it is a power of 2 print the power else print -1 . Business Rule: 1. If the given input integer is a negative number/not a power of 2, print -1. Create a class named UserProgramCode that has the following static method   
 public static int twoPower(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.  
 Output consists of an integer. Refer business rules and sample output for the format.  
    
 Sample Input 1 : 1024  
 Sample Output 1 : 10 Sample Input 2 : 6 Sample Output 2 :  
 -1

 94.Power of 2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Collections;

namespace ConsoleApplication23

{

class UserProgramCode

{

public static int twoPower(int input1)

{

int i1 = input1;

int n=2;

int i = 1;

int sum=1;

int val=0;

ArrayList al = new ArrayList();

while (i < i1)

{

if ((sum \* n) == i1)

{

val = i;

break;

}

sum = sum \* n;

al.Add(sum);

i++;

}

return val;

}

static void Main(string[] args)

{

int i;

i = int.Parse(Console.ReadLine());

int val1;

val1 = twoPower(i);

if (val1 > 0)

Console.WriteLine(val1);

else

Console.WriteLine(-1);

}

}

}

**167. Shortest Word Length**  
 Given a string array as input, write a program to find the length of the shortest word in the array..  
 Create a class named UserProgramCode that has the following static method  
 public static int shortestWordLength(string[] input1)  
 Create a class named Program that accepts a string array as input 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 string array.  
 The next 'n' lines of the input consists of strings that correspond to the elements in the string array.  
 Output consists of a single integer that corresponds to the length of the shortest word in the array.  
 Sample Input :  
 3 cherry apple blueberry Sample Output : 5

Program 72:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace ConsoleApplication18

{

class Program

{

static void Main(string[] args)

{

int n = Convert.ToInt32(Console.ReadLine());

string[] a = new string[n];

for (int i = 0; i < a.Length; i++)

a[i] = Console.ReadLine();

int r = userProgramCode.shortestWordLength(a);

Console.WriteLine(r);

Console.ReadLine();

}

}

class userProgramCode

{

public static int shortestWordLength(string[] a)

{

int min=1000;

for (int i = 0; i < a.Length; i++)

{

if (i == 0)

min = a[i].Length;

else

{

if (min > a[i].Length)

min = a[i].Length;

}

}

return min;

}

}

}

**168. Calculate Simple Interest**  
 Write a customized Simple Interest Calculator to calculate simple interest for any given inputs of Principal P and years N for a private bank in Europe.  
    
 Business Rules:  
 1) If principal is greater than or equal to 99999 & less than 500001 and years > 5,  the interest rate is 8.75%.  
 2) If principal is greater than or equal to 500000 & less than 1000001 and years > 3, the interest rate is 9.25%.  
 3) Any other principal amount and years will attract the standard interest rate of 8.25%.  
 4) The bank can accept max of 1000000 for a deposit. For amount greater than 1000000 the calculator should return -1.  
 5) Round off the interest to 2 decimal places.  
 6) Simple interest formula: (P\*N\*R) /100.  
 7) All the inputs should be non negative values. Else return -1.  
    
 Include a class UserProgramCode with a static method calculateSimpleInterest which accepts two integers and returns a double. The first input parameter refers to the principal P and the second input parameter refers to the years N.  
 The return type (Double) should return Simple Interest amount. Refer Business Rules and return -1 accordingly.  
 Create a Class Program which would be used to accept two Integers, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of two integers, the principal amount and the number of years, respectively.  
 Output consists of an Integer (the gift voucher amount) or a String “Interest cannot be calculated” if any of the inputs is invalid.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 100000  
 1  
 Sample Output 1:  
 8250  
    
 Sample Input 2:  
 10000000  
 1  
 Sample Output 2:  
 Interest cannot be calculated

simple interest

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Simple\_Interest

{

class Program

{

static void Main(string[] args)

{

int pin = int.Parse(Console.ReadLine());

int year = int.Parse(Console.ReadLine());

double op=UserProgramCode.calculatesimpleinterest(pin, year);

if(op==-1)

Console.WriteLine("Interest cannot be calculated");

else

Console.WriteLine(op);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Simple\_Interest

{

class UserProgramCode

{

public static double calculatesimpleinterest(int pin, int year)

{

double final = 0;

if (pin > 1000000)

return -1;

if (pin >= 99999 && pin <= 500001)

{

if (year > 5)

{

final = Math.Round((pin \* year \* 8.75) / 100, 2);

}

}

if (pin <= 1000001 && pin >= 500001)

{

if (year == 3 || year == 4)

{

final = Math.Round((pin \* year \* 9.25) / 100, 2);

}

}

if (year < 2 && pin < 1000000)

{

final = Math.Round((pin \* year \* 8.25) / 100, 2);

}

return final;

}

}

}

**169. Sum of Odd Even Positioned**

Write a program to find whether the sum of digits at even indexes and sum of digits at odd indexes in the given number are equal. Include a class UserProgramCode with a static method sumOfOddEvenPositioned that accepts an integer . The return type (integer) should return 1 if it a valid, else return 2. Create a Class Program which would be used to read an integer and call the static method present in UserProgramCode. Input and Output Format:  
 Input consists of an integer.  
 Output consists of a String(“Valid” or “Not Valid”).  
 Refer sample output for formatting specifications.  
 Sample Input 1: 1221  
 Sample Output 1:  
 Valid  
    
 Sample Input 2:  
 2000204    
 Sample Output 2:  
 Not Valid

SUM OF ODD EVEN POSITIONED

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp8

{

class Program

{

static void Main(string[] args)

{

int input = Convert.ToInt32(Console.ReadLine());

int value = UserMainCode.sumOfOddEvenPositioned(input);

if (value == 1)

Console.WriteLine("Valid");

else

Console.WriteLine("Not Valid");

}

}

class UserMainCode

{

public static int sumOfOddEvenPositioned(int input)

{

int even = 0, odd = 0,i=0,j=0,value=0;

string str = input.ToString();

for (i = 0; i < str.Length; i++)

{

if (i%2==0)

{

even = even + str[i];

}

else

{

odd = odd + str[i];

}

}

if (even==odd)

{

value = 1;

}

else

{

value = -1;

}

return value;

}

}

}

**170. Student Score**  
    
 Given a string array Input (Input 1) containing Student name and percentage of marks in the below format Input = {StudentName1, Mark1, StudentName2, Mark2, StudentName3, Mark3,......etc}, write a program to determine the student grade based on below condition, and print the output in the below format.  
    
 AAA has scored BBB marks with CCC scores  
    
 where AAA - Input StudentName (Input 2), BBB - Mark and CCC - Grade in Upper case. Grade calculation: If the mark is greater than or equal to 80, then OUTSTANDING If the mark is less than 80 and greater than or equal to 60, then GOOD If the mark is less than 60 and greater than or equal to 50, then AVERAGE If the mark is less than 50, then FAIL Business rule: 1) If any of the StudentName in Input1 or Input2 contains any special characters, then print “Invalid Input”. 2) If the Input2 string value is not present in Input1 array, then print “Invalid Student” 3) If the Input1 array length is odd, then print “No corresponding Student or Mark”    
 Create a class named UserProgramCode that has the following static method   
 public static string studentScore(string[] input1, string input2) 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 consist of elements in the input array.  
 The next line of the input consists of a string that corresponds to the student name.  
 Refer business rules and sample output for formatting specifications. Sample Input 1 : 4 Ram 55 Vignesh 89 Vignesh    
 Sample Output 1 : Vignesh has scored 89 marks with OUTSTANDING grade

Sample Input 2 : 5 Anil 76 Sunil 68 Raja Vignesh    
 Sample Output 2 : No corresponding Student or Mark

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace practice1

{

class Program

{

static void Main(string[] args)

{

int size = Int32.Parse(Console.ReadLine());

string[] str = new string[size];

for (int i = 0; i < size; i++)

{

str[i] = Console.ReadLine();

}

string input2 = Console.ReadLine();

Console.WriteLine(UserProgramCode.studentscore(str, input2));

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace practice1

{

class UserProgramCode

{

public static string studentscore(string[] s, string i2)

{

string[] str = s; string out1 = ""; int marks = 0, f1 = 0, f2 = 0; if (str.Length % 2 == 0)

{

for (int i = 0; i < str.Length; i++)

{

if (i % 2 == 0)

{

foreach (char c in str[i])

{

if (char.IsLetter(c))

{ f1 = 1; }

else

{

f1 = -1;

}

}

foreach (char c1 in i2.ToCharArray())

{

if (char.IsLetter(c1))

{ f2 = 1; }

else

{

f2 = -1;

}

}

if (f1 == 1 && f2 == 1)

{

if (s[i] == i2)

{

marks = Int32.Parse(str[i + 1]); if (marks >= 80)

out1 = s[i] + " has scored " + str[i + 1] + " marks with OUTSTANDING grade";

if (marks < 80 && marks >= 60)

out1 = s[i] + " has scored " + str[i + 1] + " marks with GOOD grade";

if (marks < 60 && marks >= 50)

out1 = s[i] + " has scored " + str[i + 1] + " marks with AVERAGE grade";

if (marks < 50) out1 = "fail";

}

else

{

out1 = "invalid student";

}

}

else if (f1 == -1 || f2 == -1)

{

out1 = "invalid input";

}

}

}

}

else

{

out1 = "no marks or student";

}

return out1;

}

}

}

**171. Sum of Squares**    
 Write a program to find the sum of the squares of first n natural numbers. If n less than 0, return -1.  
 Include a class UserProgramCode with a static method sumSquare which accepts an integer. The return type is an integer as given in the above statement.  
 Create a Class Program which would be used to accept Input and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of the value n.   
 Output consists of a integer as mentioned in the problem statement.  
 Refer sample output for formatting specifications.  
 Sample Input 1:  
 3  
 Sample Output 1:  
 14  
 Sample Input 2:  
 -5  
 Sample Output 2:  
 -1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Reflection;

namespace ConsoleApplication2

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

int result=UserProgramCode.sumSquare(n);

if ( result== -1)

{

Console.WriteLine(-1);

}

else

{

Console.WriteLine(result);

}

}

}

}

class UserProgramCode

{

public static int sumSquare(int n)

{

int sum = 0;

if (n < 0)

return -1;

for (int i = 1; i <= n; i++)

{

sum += (int)Math.Pow(i, 2);

}

return sum;

}

}

**172. Insurance Guide**    
 An Insurance company follows the following rules to calculate premium.   
    
 (1) If a person's health is excellent and the person's age is in the range [25,35] (both 25 and 35 inclusive)  and lives in a city and is a male then the premium is 4 Rs. per thousand and his policy amount cannot exceed Rs. 2 lakhs rupees.  
 (2) If a person satisfies all the above conditions except that the sex is female then the premium is 3 Rs. per thousand and her policy amount cannot exceed Rs. 1 lakh rupees.   
 (3) If a person's health is poor and the person's age is in the range [25,35] (both 25 and 35 inclusive) and lives in a village and is a male then the premium is 6 Rs. per thousand and his policy cannot exceed Rs. 10,000 rupees.   
 (4) In all other cases the person cannot be insured.   
    
 Write a program to display premium and maximum policy amount for given inputs.  
    
 Input1 - Health condition ( 'E' for excellent and 'P' for poor health)   
 input2 - Age  
 input3 - Gender('F' for female,'M' for Male)  
 input4 - Location('C' for City,'V' for Village)  
    
 Output1 - Premium per Thousand  
 Output2 - Maximum Insurance Amount the person can avail.  
    
 Example:  
    
 input1: E  
 input2: 30  
 input3: F  
 input4: C  
    
 Output1: 3  
 Outpu2: 100000  
    
 Business Rule:  
    
 1. If the person can't be insured then print “The person cannot be insured”.  
 2. If the person's age is more than 60 then print “Age limit Exceeded”.  
    
 Include a class UserProgramCode with a static method InsuranceGuide which accept three characters and an integer. The return type is void.  
 Create a Class Program which would be used to accept three characters and an integer , and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of a character, which corresponds to the health condition, an Integer, which corresponds to the age, a character, which corresponds to the gender, a character, which corresponds to the location.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 E  
 30  
 F  
 C  
 Sample Output 1:  
 3  
 100000  
    
 Sample Input 2:  
 E  
 70  
 F  
 C  
 Sample Output 2:  
 Age limit Exceeded  
    
 Sample Input 3:  
 P  
 50  
 F  
 V  
 Sample Output 3:  
 The person cannot be insured

insuranceguide

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

class Program

{

public static void Main(string[] args)

{

char health = Convert.ToChar(Console.ReadLine());

int age = Convert.ToInt32(Console.ReadLine());

char gender = Convert.ToChar(Console.ReadLine());

char location = Convert.ToChar(Console.ReadLine());

UserProgramCode.InsuranceGuide(health, age, gender, location);

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

class UserProgramCode

{

public static void InsuranceGuide(char health, int age, char gender, char location)

{

//Fill your code here

if (health == 'E' && age >= 25 && age <= 35 && location == 'C' && gender == 'M')

{

Console.WriteLine(4);

Console.WriteLine(200000);

}

else if (health == 'E' && age >= 25 && age <= 35 && location == 'C' && gender == 'F')

{

Console.WriteLine(3);

Console.WriteLine(100000);

}

else if (health == 'P' && age >= 25 && age <= 35 && location == 'V' && gender == 'M')

{

Console.WriteLine(6);

Console.WriteLine(10000);

}

else if(age>=60)

Console.WriteLine("Age limit Exceeded");

else

Console.WriteLine("The person cannot be insured");

}

}

**173. Calculate Cost**

Samira Florists take orders for flower decoration in social events and functions. They charge their customers based on:  
    
 The weight of flowers (kg) and   
 Decoration Options – Simple (S), Customized (C).  
 Flower type - Normal flowers (N) cost Rs.400 per kg   
 Exotic flowers (E) cost Rs. 700 per kg  
    
 They charge an additional Rs.15000 for simple decoration and Rs.25000 for a Customized decoration, apart from the cost of flowers. Also, they take only a minimum order of Rs. 20000 or more (including the flower cost and the decoration cost).   
    
 Business Rule:  
    
 If the cost calculated is less Rs 20000, then return -1.  
 If the type of flowers given is other than (N) or (E), then return -2.  
 If the type of decorations given is other than (S) or (C), then return -3.  
    
 Write a program to read an Integer and two characters, and calculate the total amount the customer pays to the florists. The values are: weight of flowers(in kg), the type of flowers (N or E)and the type of decoration(S or C). Print the final cost, or print “Too low cost”, if method returns -1, print “Invalid type of flower”, if method returns -2, print “Invalid decoration type”, if the method returns -3.  
    
 (Total Amount customer pays = Cost of flowers + Cost of Decoration)  
    
 Include a class UserProgramCode with a static method calculateCost which accept an integer and two characters . The return type (integer) should return output according to the business rules.  
 Create a Class Program which would be used to accept an integer and two characters, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of an integer, which corresponds to the weight, and two characters, which correspond to the flowers type and decoration type respectively.  
 Out put consists of an integer or a string.  
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 10  
 N  
 B  
 Sample Output 1:  
 Invalid decoration type  
    
 Sample Input 2:  
 30  
 A  
 S  
 Sample Output 2:  
 Invalid type of flower  
    
 Sample Input 3:  
 10  
 N  
 S  
 Sample Output 3:  
 Too low cost  
    
 Sample Input 4:  
 20  
 N  
 S  
 Sample Output 4:  
 23000

24.CALCULATE COST

using System;

class Program

{

public static void Main(string[] args)

{

int cost = Convert.ToInt32(Console.ReadLine());

char fltype = Convert.ToChar(Console.ReadLine());

char dctype = Convert.ToChar(Console.ReadLine());

int result = UserProgramCode.calculateCost(cost, fltype, dctype);

if (result == -1)

Console.WriteLine("Too low cost");

else if (result == -2)

Console.WriteLine("Invalid type of flower");

else if (result == -3)

Console.WriteLine("Invalid decoration type");

else

Console.WriteLine(result);

}

}

using System;

class UserProgramCode

{

public static int calculateCost(int cost, char fltype, char dctype)

{

if (dctype != 'S' && dctype != 'C')

return (-3);

if (fltype != 'N' && fltype != 'E')

return (-2);

int value = 0;

if (dctype == 'S')

value = value + 15000;

if (dctype == 'C')

value = value + 25000;

if (fltype == 'N')

value = value + (cost \* 400);

if (fltype == 'E')

value = value + (cost \* 700);

if (value < 20000)

return -1;

else

return (value);

}

}

**174. Concatenate Characters**    
 Given an input string array, write a program to get the second character of each string and form a new String by concatenating the fetched characters together. Print the new string formed.    
 Business Rules : 1. If the given input array element contains numbers, then print -1. 2. If the given input array element contains special characters , then print -2. 3. If the input array contains only one string, then print -3.    
 Create a class named UserProgramCode that has the following static method   
 public static string concatCharacter(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 :  
 3  
 ab  
 aaa  
 adbcd  
 Sample Output 1 :  
 bad Sample Input 2 :  
 4  
 ban  
 b%a  
 ssm  
 tea    
 Sample Output 2 :  
 -2

----Concatenate Characters---

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace concatcharacters

{

class Program

{

static void Main(string[] args)

{

int i,n;

n=Convert.ToInt32(Console.ReadLine());

string[]s=new string[n];

for (i = 0; i < n; i++)

{

s[i]=Console.ReadLine();

}

concat c = new concat();

string output=c.concatenation(s);

Console.WriteLine(output);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace concatcharacters

{

class concat

{

public string concatenation(string[] s)

{

StringBuilder sb = new StringBuilder();

if (s.Length == 1)

{

return "-3";

}

foreach (string i in s)

{

foreach (char c in i)

{

if(Char.IsDigit(c))

{

return "-1";

}

if (!Char.IsLetter(c))

{

return "-2";

}

}

sb.Append(i[1]);

}

return sb.ToString();

}

}

}

**175.Validate ID Locations**    
 Write a program to read two string inputs and check whether the first string is in valid format. First string is ID and second string is location. A valid ID  should be in the format CTS-LLL-XXXX where LLL is the first three letters of given location and XXXX is a four digit number. If the given ID is as per the given format, print “valid” else print “invalid”. Example: Input1 = CTS-hyd-1234 Input2 = hyderabad output = valid    
 Include a class UserProgramCode with a static method validateIDLocations which accepts two Strings. The return value (Integer) should be 1 if the first string is valid, else return -1.  
 Create a Class Program which would be used to read 2 strings and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of 2 strings. Output consists of a string, “valid” or  “invalid”.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 CTS-hyd-1234 hyderabad Sample Output 1:  
 valid

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace idValidation

{

class Program

{

static void Main(string[] args)

{

string s1 = Console.ReadLine();

string s2 = Console.ReadLine();

int i = UserProgramCode.validateIDlocations(s1, s2);

if (i == 1)

{

Console.WriteLine("valid");

}

else

{

Console.WriteLine("Invalid");

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace idValidation

{

class UserProgramCode

{

public static int validateIDlocations(string s1, string s2)

{

int output = 0;

Regex reg = new Regex(@"^([CTS]+[-]+([A-Za-z]{3})+[-]+([0-9]{4}))$");

if (reg.IsMatch(s1))

{

string res = s2.ToLower();

if (s1.Contains(res.Substring(0, 3)))

{

output = 1;

}

else

{

output = -1;

}

}

else

{

output = -2;

}

return output;

}

}

}

**176. Next Year Day**  
 Write a program to read a date String in dd/mm/yyyy format and to calculate the day which falls on the same date next year and print it. Note - return the output in small case.  
    
 Example:  
 Input = 13/07/2012  
 output = saturday  
    
 Include a class UserProgramCode with a static method nextYearDay which accepts a String. The return type (String) should return the day which falls on the same date next year. Return -1 in case the format of the date is incorrect.  
 Create a Class Program which would be used to accept an Integer, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of a String, date in dd/mm/yyyy format.  
 Output consists of a String, the the day which falls on the same date next year.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input:  
 13/07/2012  
 Sample Output:  
 saturday

72.Next year day

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication27

{

class UserProgramCode

{

public static string nextYearDay(string date)

{

string output = "";

DateTime day;

bool check = DateTime.TryParseExact(date, "dd/MM/yyyy", null, System.Globalization.DateTimeStyles.None, out day);

if (check == true)

{

day = day.AddYears(1);

//Console.WriteLine(day);

//day = day.DayOfWeek;

output = day.DayOfWeek.ToString();

output = output.ToLower();

return output;

}

else

{

return output;

}

}

class Program

{

static void Main(string[] args)

{

string input, output;

input = Console.ReadLine();

output = UserProgramCode.nextYearDay(input);

if (output == null)

{

Console.WriteLine("Invalid date");

}

Console.WriteLine(output);

Console.ReadKey();

}

}

}

}

177. Count Vowels   
 Write a program to count the character which comes under the vowels sound from the given string .  
 The string value should have only the alphabet values.  
 Business Rules:  
 1. If the input string consists of any other character than the alphabets, return -1 from the method and print "Other characters found" in Main.  
 Include a class UserProgramCode with static method countVowels() that accepts a string and returns an integer.  
    
 Create a class Program which would get the input and call the static method countVowels() present in the UserProgramCode.  
 Input and Output Format: Input is a string. Output is an integer if the method returns the count, else a String "Other characters found" if the method returns '-1'. Sample Input 1: sang-gee- Sample Output 1: Other characters found Sample Input 2: god Sample Output 2: 1

53.CountVowels

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace CountVowels

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

int rslt = UserProgramCode.countVowels(str);

if (rslt == -1)

Console.WriteLine("Other character found");

else

Console.WriteLine(rslt);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace CountVowels

{

class UserProgramCode

{

public static int countVowels(string str)

{

string str1=str.ToLower();

int rs=0;

Regex reg = new Regex("^[a-z]+$");

if (reg.IsMatch(str1))

{

foreach (char c in str1)

{

if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')

{

rs++;

}

}

// return rs;

}

else

{

rs = -1;

}

return rs;

}

}

}

Program 2:

12)/////count vowels///

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace CountVowells

{

class UserProgramCode

{

public static int countVowels(string input)

{

int output = 0;

foreach (char c in input)

{

if (char.IsLetter(c) == false)

{

output = -1;

return output;

}

else

{

switch (c)

{

case 'a':

case 'e':

case 'i':

case 'o':

case 'u':

case 'A':

case 'E':

case 'I':

case 'O':

case 'U':

output++;

break;

}

}

}

return output;

}

}

}

**178.Repeated Integers**    
 Write code to pick all the repeated integers in a given integer array, sort them in ascending order and put them in the output list. Print the output list.  
    
 Include a class UserProgramCode with a static method findRepeatedIntegers which accepts the size of an integer array and an integer array. The return type is void. Print the repeated integers in sorted order if present. If there are no repeated numbers, then print “No repeated numbers”. If there are negative numbers in the array, print “Array contains negative numbers” . Create a Class Program which would be used to accept Input array and call the static method present in UserProgramCode.  
    
 Input and Output Format:  
 Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.  
    
 Refer sample output for formatting specifications.  
    
 Assume that the maximum number of elements in the array is 30.  
    
 Sample Input 1:  
 4  
 3  
 3  
 2  
 10  
 Sample Output 1:  
 3  
    
 Sample Input 2:  
 4  
 3  
 1  
 2  
 10  
 Sample Output 2:  
 No repeated numbers  
    
 Sample Input 3:  
 4  
 3  
 -11  
 2  
 10  
 Sample Output 3:  
 Array contains negative numbers

98.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Collections;

namespace ConsoleApplication98

{

class Program

{

static void Main(string[] args)

{

int l=1;

int[] a = new int[30];

for (int i = 0; i <= l; i++)

{

a[i] = Convert.ToInt16(Console.ReadLine());

l = a[0];

}

UserProgramCode.findRepeatedIntegers(a);

}

}

class UserProgramCode

{

public static void findRepeatedIntegers(int[] a)

{

ArrayList a1 = new ArrayList();

int flag = 0;

for (int i = 1; i <= a[0]; i++)

{

if (a[i] >= 0)

{

int c = 0;

for (int j = 1; j <= a[0]; j++)

{

if (a[i] == a[j])

c++;

}

if (c > 1)

{

if (!(a1.Contains(a[i])))

a1.Add(a[i]);

}

}

else

{

flag = 1;

break;

}

}

if (flag == 0)

{

a1.Sort();

int c1 = 0;

foreach (int i in a1)

c1++;

if (c1 == 0)

Console.WriteLine("No repeated numbers");

else

{

foreach (int i in a1)

{

Console.WriteLine(i);

}

}

}

else

Console.WriteLine("Array contains negative numbers");

}

}

}

**179. Number Validation**  
 Write a program to read a string of 10 digit number and to check whether the string contains a 10 digit number in the format XXX-XXX-XXXX where 'X' is a digit.   
    
 Include a class UserProgramCode with a static method validateNumber which accepts a string as input and returns an Integer .  
 The method returns 1 if the string meets the above specified format . If the string input does not meet the specified format the method returns -1.  
    
 Create a class Program which would get the input as a String and call the static method validateNumber present in the UserProgramCode.  
    
 Input and Output Format:  
 Input consists of a string.  
 Output is a string specifying the given string is valid ("Valid number format") or not ("Invalid number format") .  
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 123-456-7895  
 Sample Output 1:  
 Valid number format  
 Sample Input 2:  
 -123-12344322  
 Sample Output 2:  
 Invalid number format

62

class Program

{

static void Main(string[] args)

{

int f = 0;

string s;

s = Console.ReadLine();

userprogramcode obj = new userprogramcode();

f=obj.validatenumber(s);

if(f==1)

Console.WriteLine("Valid number format");

if(f==-1)

Console.WriteLine("Invalid number format");

}

}

public class userprogramcode

{

public int validatenumber(string s)

{

if (Regex.IsMatch(s, @"^\d{3}[-]\d{3}[-]\d{4}$"))

{

return 1;

}

else

return -1;

}

}

**180. Validate String**    
 For a given String apply the following validations. 1. The given input String should be only four characters long. If not print -1. 2. First character can be an alphabet or digit. If not print -2 . 3. Second character must be uppercase alphabet. (eg 'M','R'.. any alpbabet A - Z). If not print -3 . 4. Third character must be a number and also between 5-9. If not print -4 . If all the conditions are satisfied print 1. Example 1: input='vM7u3' output = -1 Example 2: input='&Mau' output = -2 Example 3: input='vrau' output = -3 Example 4: input='vR3a' output = -4 Example 5: input='vR5a' output = 1    
 Create a class named UserProgramCode that has the following static method   
 public static int validateString(string 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 a string.  
 Output is an integer.  
 Sample Input :  
 vR5a  
    
 Sample Output :  
 1  
    
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Validatestring

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

int op = UserProgramCode.validate(str);

Console.WriteLine(op);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Validatestring

{

class UserProgramCode

{

public static int validate(string str)

{

if (str.Length > 4)

{

return -1;

}

else if (!Char.IsLetterOrDigit(str[0]))

{

return -2;

}

else if (!(str[1] >= 65 && str[1] <= 90))

{

return -3;

}

else if (!((str[2] == '5') || (str[2] == '6') || (str[2] == '7') || (str[2] == '8') || (str[2] == '9')))

{

return -4;

}

else

return 1;

}

}

}

**181. IP Validator**  
    
 Write code to read an IP address in a String variable and validate the IP address. Print “Valid” if it is a valid IP address else print “Invalid”.  
    
 Note: An IP address has the format a.b.c.d where a,b,c,d are numbers between 0-255  
    
 Include a class UserProgramCode with a static method ipValidator which accepts a string. The return type (integer) should return 1 if it a valid IP, else return 2.  
 Create a Class Program which would be used to accept a string and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of a String.  
 Output consists of a String(“Valid” or “Invalid”).  
 Refer sample output for formatting specifications.  
 Sample Input 1:  
 132.145.184.210  
 Sample Output 1:  
 Valid  
    
 Sample Input 2:  
 132.145.184.290  
 Sample Output 2:  
 Invalid

Q3.IP Validator

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace level2

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

int no = UserProgramCode.ipValidator(str);

if(no==1)

Console.WriteLine("Valid");

else

Console.WriteLine("Invalid");

}

}

class UserProgramCode

{

public static int ipValidator(string str)

{

Int32 i = 0, len, no,flag=0;

len = str.Length;

Int32 start = 0, count = 0;

while (i < len)

{

if (str.ElementAt(i) != '.')

{

count++;

}

else

{

no = Int32.Parse(str.Substring(start, count));

if (no < 0 || no > 255)

return 2;

start = start + count+1;

flag++;

count = 0;

}

i++;

}

no = Int32.Parse(str.Substring(start, count));

flag++;

if (no < 0 || no > 255)

return 2;

else if (flag > 4)

return 2;

else

{

if (i == len)

{

return 1;

}

else

return 0;

}}

}

}

182. Fibonacci Series  
 Write method to generate fibonacci series and calculate the sum of first n numbers in the series and return it as output.  
    
 Example:  
 input = 5  
 output = 0 + 1 + 1 + 2 + 3 = 7  
    
 Include a class UserProgramCode with a static method getSumOfNfibos that accepts an integer as input and returns an integer.  
    
 Create a class Program which would get the input and call the static method getSumOfNfibos present in the UserProgramCode.  
    
 Input and Output Format: Input consists of an integer that corresponds to n. Output consists of an integer which corresponds to the sum of the first n terms in the fibonocci series.  
    
 Note: First two numbers in a Fibonacci series are 0, 1 and all other subsequent numbers are sum of its previous two numbers. Example - 0, 1, 1, 2, 3, 5...  
 Sample Input 1: 15 Sample Output 1: 986 Sample Input 2: 4 Sample Output 2: 4  
  

n..27

class Program

{

static void Main(string[] args)

{

int n = Convert.ToInt16(Console.ReadLine());

int i = UserProgramCode.getSumOfNfibos(n);

Console.WriteLine(i);

Console.ReadLine();

}

}

class UserProgramCode

{

public static int getSumOfNfibos(int n)

{

int f = 0, f1 = -1, f2 = 1, sum = 0;

for (int i = 0; i < n; i++)

{

f = f1 + f2;

f1 = f2;

f2 = f;

sum = sum + f;

}

return sum;

}

}

**183. Rearrange Case**    
 Given a string input, write a program to form a new string provided with the the below limitations 1. Check for the alphabet which has maximum number of Upper case and lower case in the input string value. 2. Uppercase alphabets would be moved to the start of the Output string and lowercase alphabets should be moved to the end of the string. 3. Remaining other alphabets will remain the same in between the start and end of the output variable irrespective of the case. Business rule: 1) If the Input string contains any special characters, then print 'Invalid Input'. 2) If the Input string does not contain Uppercase at all, then print 'Condition does not meet' . 3) If two or more alphabets has maximum upper and lower case, then print 'Re-arranging is not possible' . Create a class named UserProgramCode that has the following static method   
 public static string rearrangeCase(string 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 a string.  
 Output consists of a string.  
 Refer business rules and sample output for the format.    
 Sample Input 1 :  
 CancelPolicy Sample Output 1 :  
 CanelPoliycc Sample Input 2 :   
 XYZbossxyz Sample Output 2 :  
 Re-arranging is not possible

 97.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Collections;

namespace ConsoleApplication97

{

class Program

{

static void Main(string[] args)

{

string a, b;

a = Console.ReadLine();

b=UserProgramCode.rearrangeCase(a);

Console.WriteLine(b);

}

}

class UserProgramCode

{

static char cf;

public static string rearrangeCase(string input1)

{

int l = input1.Length;

// string a=Convert.ToString();

int c1 = 0,c3=0;

StringBuilder a1 = new StringBuilder();

StringBuilder a2 = new StringBuilder();

StringBuilder a3 = new StringBuilder();

for (int i = 0; i < l; i++)

{

if(char.IsLower(input1[i]))

{

c1++;

}

}

if (c1 == l)

{

return ("Condition does not meet");

}

else

{

for (int i = 0; i < l; i++)

{

int c2 = 0;

if (char.IsUpper(input1[i]))

{

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

{

if (input1[j] == char.ToLower(input1[i]))

c2++;

}

}

if (c2 > c3)

{

c3 = c2;

cf = input1[i];

}

else if (c2 == c3)

{

return ("Re-arranging is not possible");

}

}

for (int i = 0; i < l; i++)

{

if (input1[i] == char.ToUpper(cf))

{

a1.Append(cf);

}

else if (input1[i] == char.ToLower(cf))

{

a2.Append(char.ToLower(cf));

}

else

{

a3.Append(input1[i]);

}

}

}

return (a1.ToString() + "" + a3.ToString() + "" + a2.ToString());

}

}

}

**184. Bill Amount Calculation** Raviraj group is having guest houses at different locations. Write a program to calculate bill amount as per the Room type & number of days of stay at a particular location. Rent for Non-AC room: Chennai(C): 1000 per day Hyderabad(H): 800 per day Bangalore(B): 1100 per day Rent for AC room: Chennai(C): 1300 per day Hyderabad(H): 1100 per day Bangalore(B): 1400 per day Input1: Location Input2: Room Type Input3 :Number of days The input values should exactly be the same as 'C', 'H',' B' for the given locations Chennai, Hyderabad and Banglore respectively and AC, NAC for the AC and non AC room type respectively. If input for location and room type is not given as per above criteria, print -1. Only positive number should be given for number of days. If not, print -1.  
 Create a class named UserProgramCode that has the following static method  
 public static int calculateBillAmount(char input1, string input2, int input3)   
 The first input parameter refers to the location, the second parameter refers to the room type and the third refers to the number of days of stay. Create a class named Program that accepts the inputs and calls the static method present in the UserProgramCode.  
    
 Input and Output Format: The first input corresponds to the location, the second input corresponds to the room type and the third input corresponds to the number of days of stay. Output consists of an integer.  
    
 Sample Input: C AC 5    
 Sample Output :  
 6500

10.BILL AMOUNT

using System;

class Program

{

public static void Main( string[] args )

{

char input1;

string input2;

int input3;

input1 = Convert.ToChar(Console.ReadLine());

input2 = Console.ReadLine();

input3 = Convert.ToInt32(Console.ReadLine());

Console.WriteLine(UserProgramCode.calculateBillAmount(input1,input2,input3));

}

}

using System;

class UserProgramCode

{

public static int calculateBillAmount(char input1,string input2,int input3)

{

if(input3>0)

{

if(input1=='C' && input2.Equals("NAC"))

return input3\*1000;

else if(input1=='C' && input2.Equals("AC"))

return input3\*1300;

else if(input1=='H' && input2.Equals("NAC"))

return input3\*800;

else if(input1=='H' && input2.Equals("AC"))

return input3\*1100;

else if(input1=='B' && input2.Equals("NAC"))

return input3\*1100;

else if(input1=='B' && input2.Equals("AC"))

return input3\*1400;

else

return -1;

}

else

return -1;

Console.ReadLine();

}

}

**185. Valid Negative Number** Write a program to read a negative number as a String variable and to validate the number. If the given string contains a valid negative number print corresponding positive number  else print “Invalid number” .  
    
 Example:  
 input = "-94923"  
 output = "94923"  
    
    
 Include a class UserProgramCode with a static method validateNumber which accepts a String. The return type (String) should return the corresponding output. If the input string is not a valid negative number, the method returns "-1".  
 Create a Class Program which would be used to accept a String, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of a String( a negative number).  
 Output consists of a String(the corresponding output).  
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 -94923  
 Sample Output 1:  
 94923  
    
 Sample Input 2:  
 -13O  
 Sample Output 2:  
 Invalid number

13.Valid Negative Number

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication9

{

public class UserProgramCode

{

public static string validateNumber(string str)

{

str.ToCharArray();

int temp = 0;

if (str[0] == '-')

{

for (int i = 0; i < str.Length; i++)

if (str[i] >= 48 && str[i] <= 57)

temp = 1;

else

temp = 0;

if (temp == 1)

{

str = str.Substring(1, str.Length-1);

return str.ToString();

}

else

return "-1";

}

else

return "-1";

}

}

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

Console.WriteLine(UserProgramCode.validateNumber(str));

}

}

}

**186. Interchange Characters**  
 Write a program that accepts a string input and interchanges the first and last characters. Case sensitivity should be checked. Business rules: 1) Print 'Invalid String' when the given input string consists of any special characters or numbers. 2) Print 'No Change' when the first and last characters of the input string is same and of the same case. Example 1: Input: Execute Output: executE Example 2: Input: BoB Output: No Change Create a class named UserProgramCode that has the following static method   
 public static string interchangeFirstLast(string input1)  
    
 Create a class named Program that accepts the input and calls the static method present in the UserProgramCode.  
 Input and Output Format: Input consists of a string.  
 Output is a string. Refer sample output and business rules Sample Input 1: Execute  
    
 Sample Output 1:  
 executE  
    
 Sample Input 2: BoB  
    
 Sample Output 2:  
 No Change

84.)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication13

{

class userprogramcode

{

public static string getString(string ip1)

{

string[] final = new string[ip1.Length];

for (int j = 0; j < ip1.Length; j++)

final[j] = ip1[j].ToString();

string t1,ans="";

for (int j = 0; j < ip1.Length; j++)

if (!char.IsLetter(ip1[j]))

return "Invalid String";

if (final[0] != final[ip1.Length - 1])

{

t1 = final[0];

final[0] = final[ip1.Length - 1];

final[ip1.Length - 1] = t1;

for (int i = 0; i < ip1.Length; i++)

ans=ans+final[i];

}

else

ans = "No Change";

return ans;

}

}

class Program

{

static void Main(string[] args)

{

String x, y;

x = Console.ReadLine();

y = userprogramcode.getString(x);

Console.WriteLine(y);

}

}

}

**187. Common Characters**

Write a method to count the common character from the given two Strings. Rule: - Space should not be counted as a letter. - Consider letters to be case sensitive. ie, 'a' is not equal to 'A'. Example: input1 = ""a black cow"" input2 =""battle ship"" output = 3 Include a class UserProgramCode with static method commonChars which accepts two String values.The return type is an interger. Create a class Program which would get the input and call the static method commonChars present in the UserProgramCode. Input Output format: The input consists of two strings. The output is an interger which counts the common characters.

Sample input 1: a black cow battle ship

Sample Output 1: 3

Sample input 2: australia sri lanka

Sample Output 2: 5

COMMON CHARACTER

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

class Program

{

public static void Main()

{

String input1 = Console.ReadLine();

String input2 = Console.ReadLine();

Console.WriteLine(UserProgramCode.commonChars(input1, input2));

Console.ReadLine();

}

}

class UserProgramCode

{

public static int commonChars(String input1, String input2)

{

char[] s1a = input1.ToCharArray();

char[] s2a = input2.ToCharArray();

int l1 = input1.Length;

int l2 = input2.Length;

Array.Sort(s1a);

Array.Sort(s2a);

List<char> s1c = new List<char>();

List<char> s2c = new List<char>();

for (int i = 0; i < l1; i++)

{

if (s1a[i] == ' ')

continue;

else if (!s1c.Contains(s1a[i]))

{

s1c.Add(s1a[i]);

}

}

for (int i = 0; i < l2; i++)

{

if (s2a[i] == ' ')

continue;

else if (!s2c.Contains(s2a[i]))

{

s2c.Add(s2a[i]);

}

}

int c = 0;

foreach (var i1 in s1c)

{

foreach (var i2 in s2c)

{

if (i1 == i2)

{

c++;

}

}

}

// Console.WriteLine(c);

return c;

}

}

**188. Repeated Words**  
    
 Given two string inputs input1 and input2, write a program to calculate the number of times each word in input1 occurs in input2 and print the words which has occurred the maximum number of times in the output with empty spaces between them and also in the same order as given in input1. Output should be printed in lower case.Ignore case sensitiveness in both the input strings. Business Rules: 1)If any of the words is repetitive in input1, then print -1. 2)If none of the words from input1 has occurred in input2, then print -2. Create a class named UserProgramCode that has the following static method   
 public static string repeatedWords(string input1, string input2)  
 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 2 strings, input1 and input 2.  
 Output consists of a string.  
 Refer business rules and sample output for formatting specifications.  
    
 Sample Input 1 : Apple is a fruit apple is a nice fruit and apple is available in all seasons  
 Sample Output 1 :  
 apple is Sample Input 2 : Does he have to have a car ? Yes he should.  
 Sample Output 2 :  
 -1

REPEATED WORDS

using System;

class Program

{

public static void Main( string[] args )

{

string input1,input2,output;

input1 = Console.ReadLine();

input2 = Console.ReadLine();

output = UserProgramCode.repeatedWords(input1,input2);

Console.WriteLine(output);

Console.ReadKey();

}

}

using System;

using System.Collections;

class UserProgramCode {

public static string repeatedWords(string input1,string input2)

{

input1 = input1 + " ";

input2 = input2 + " ";

ArrayList s1 = new ArrayList();

ArrayList s2 = new ArrayList();

ArrayList intS1 = new ArrayList();

string[] a1 = input1.Split(' ');

string[] a2 = input2.Split(' ');

int i,c=0,j,max=0,flag=0;

string str="";

s1.Add(a1[0]);

for (i = 1; i < a1.Length-1; i++)

{

if (s1.Contains(a1[i]))

return "-1";

else

s1.Add(a1[i]);

}

for (i = 0; i < a2.Length-1; i++)

s2.Add(a2[i]);

for (i = 0; i < s1.Count; i++)

{

for (c=0,j = 0; j < s2.Count; j++)

{

if (a1[i].Equals((string)s2[j],StringComparison.OrdinalIgnoreCase))

{

c++;

}

}

intS1.Add(c);

}

for (i = 0; i < intS1.Count; i++)

{

if ((int)intS1[i] == 0)

flag++;

}

if (flag == intS1.Count)

return "-2";

max = (int)intS1[0];

for (i = 0; i < intS1.Count; i++)

{

if ((int)intS1[i] > max)

{

max = (int)intS1[i];

}

}

for (i = 0; i < s1.Count; i++)

{

if ((int)intS1[i] == max)

{

str = str + a1[i] + " ";

}

}

//Console.ReadKey();

return str.ToLower();

}

}

**189. Get All Elements**  
 Write a program to get all the elements that are greater than 5 from a given input integer list. Display it in the order as present in the array.    
 Print the elements.  
 Example:   
 Input1: [1,3,7,8,5,13]  
 Output1:[7,8,13]  
    
 Business Rule:  
 If any of the element in the input list is greater than 500 then store -1 in the oth index of the output list.  
    
 Include a class UserProgramCode with a static method GetAllElements which accepts an integer List and its size. The return type (integer list) should return output according to the business ruless  
 Create a Class Program which would be used to accept a list, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of n+1 integers, where first integer corresponds to the size of the list, followed by the corresponding list elements.  
 Output consists of an Integer list, or a String “Array element greater than 500” if any of the elements is greater than 500.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 6  
 1  
 3  
 7  
 8  
 5  
 13  
 Sample Output1:  
 7  
 8  
 13  
    
 Sample Input 2:  
 6  
 1  
 3  
 7  
 8  
 501  
 13  
 Sample Output 2:  
 Array element greater than 500

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace get\_all\_the\_elements

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

int[] arr = new int[n];

for (int i = 0; i < n; i++)

{

arr[i] = int.Parse(Console.ReadLine());

}

int[] op = UserProgramCode.getElements(arr);

int len=op.Length;

for (int i = 0; i < op[len - 1]; i++)

{

if (op[0] == -1)

{

Console.WriteLine("Array element greater than 500");

}

else

Console.WriteLine(op[i]);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace get\_all\_the\_elements

{

class UserProgramCode

{

public static int[] getElements(int[] arr)

{

int k=0;

int[] temp =new int[30];

for (int i = 0; i < arr.Length; i++)

{

if (arr[i] > 500)

temp[0] = -1;

}

for (int i = 0; i < arr.Length; i++)

{

if (arr[i] > 5)

{

temp[k] = arr[i];

k++;

}

}

temp[29] = k ;

return temp;

}

}

}

**190. Calculate Telephone Bill**

Write a program which reads the number of calls as input and calculates the monthly telephone bills as per the following rules. Print the bill amount.  
    
 Minimum Rs. 200 for upto 300 calls.  
 Plus Rs. 0.60 per call for next 50 calls.   
 Plus Rs. 0.50 per call for next 50 calls.   
 Plus Rs. 0.40 per call for any call beyond 400 calls.  
    
 Example :   
 If the calls is 720, the calculation would be as follows :  
 1. First 300 calls the charge is Rs 200.  
 2. Next 50 the charge is 50 \*.60=30.  
 3. Next 50 the charge is 50 \*.50=25.  
 4. Balance calls ( 720-300-50-50) 320 the charge is 320 \*.40=128.  
 5. Total charge = 200 + 30+25+128 = 383.  
    
 The calculated charge should be in double datatype which is rounded to 2 decimal places.  
    
 Include a class UserProgramCode with a static method calculateTelephoneBill which accepts an Integer. The return type (Double) should return the final bill amount.  
 Create a Class Program which would be used to accept an Integer, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of an Integer, which corresponds to number of calls.  
 Output consists of a Double (The final bill amount).  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 720  
 Sample Output 2:  
 383.00

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace calculate\_telephone\_bill

{

class UserProgramCode

{

public static double Telephone(int input1)

{

double output1;

if (input1 <= 300)

{

output1 = 200;

return output1;

}

else if (input1 > 300 && input1 <= 350)

{

output1 = 200 + ((input1 - 300) \* 0.60);

return output1;

}

else if (input1 > 350 && input1 <= 400)

{

output1 = 200 + (50 \* 0.60) + ((input1 - 350) \* 0.50);

return output1;

}

else if (input1 > 400)

{

output1 = 200 + (50 \* 0.60) + (50 \* 0.50) + ((input1 - 400) \* 0.40);

return output1;

}

return 0;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace calculate\_telephone\_bill

{

class Program

{

static void Main(string[] args)

{

int Calls = int.Parse(Console.ReadLine());

double op = UserProgramCode.Telephone(Calls);

Console.WriteLine(op.ToString("F"));

Console.ReadLine();

}

}

}

**191. Calculate Take Home Salary**

TMB Software solution is under developing software company in KolKatta. The company is providing the Provident fund for all the employees based on the salary criteria as given below. Write a method to find the take home salary for the employee. Medical Insurance for each employee is Rs 678.   Salary Range------------ PF Lesser than 15000 ------- 750  15001 - 22000 -------- 850  22001-30000 -------- 925  Above 30000 -------- 1000    Take Home Salary = Salary - pf - MedicalInsurance   Include a class UserProgramCode with a static method calculateHomeSalary which accepts an integer and returns an integer that corresponds to the Take Home Salary.  
 The method returns -1 when the input integer is negative. Create a class Program which would get the input and call the static method calculateHomeSalary present in the UserProgramCode. If the method returns -1, print 'Invalid Input'.    
 Input and Output format : Input consists of a integer that represents a salary. Output is an integer that corresponds to 'take home salary' or a string 'Invalid Input'.

Sample Input 1 : 13500

Sample Output 1 : 12072

Sample Input 2 : -10000

Sample Output 2 : Invalid Input

20.CALCULATE TAKE HOME SALARY

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace takehomesalary

{

class Program

{

static void Main(string[] args)

{

int rslt;

Console.WriteLine("enter the salary");

int Salary= Convert.ToInt32(Console.ReadLine());

rslt=UserProgramCode.calculateHomeSalary(Salary);

if (rslt == -1)

{

Console.WriteLine("Invalid Input");

}

else

{

Console.WriteLine(rslt);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace takehomesalary

{

class UserProgramCode

{

public static int calculateHomeSalary(int Salary)

{

int PF=0;

int MedicalInsurance=678;

int HomeSal=0;

if (Salary > 0)

{

if (Salary < 15000)

{

PF = 750;

HomeSal = (Salary - PF - MedicalInsurance);

}

else if (Salary >= 15001 && Salary <= 22000)

{

PF = 850;

HomeSal = Salary - PF - MedicalInsurance;

}

else if (Salary >= 22001 && Salary <= 30000)

{

PF = 925;

HomeSal = Salary - PF - MedicalInsurance;

}

else if (Salary > 30000)

{

PF = 1000;

HomeSal = Salary - PF - MedicalInsurance;

}

return HomeSal;

}

else

{

return -1;

}

}

}

}

**192. Count the number of odd integers**

Write a code to count the number of odd integers in the given integer array. Include a class UserProgramCode with static method countOddIntegers  that accepts an integer array and the return type should be int (count of odd Integers). Return -1 if the array contains negative values. Create a class Program which would get the input and call the static method countOddIntegers present in the UserProgramCode. In Program  display “The Array consists non-positive value(s)” if -1 is returned, else print the count. Input and Output Format:  
 Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.  
 Output consists of an integer or a string. Refer sample output for formatting specification. SAMPLE INPUT 1: 2 -1 2 SAMPLE OUTPUT 1: The Array consists non-positive value(s) SAMPLE INPUT 2: 2 1 3 SAMPLE OUTPUT 2: 2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace count\_odd\_integers

{

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

int[] arr = new int[n];

for (int i = 0; i < n; i++)

{

arr[i] = int.Parse(Console.ReadLine());

}

int op = UserProgramCode.countOdd(arr,n);

if (op == -1)

{

Console.WriteLine("The Array consists non-positive value(s)");

}

else

Console.WriteLine(op);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace count\_odd\_integers

{

class UserProgramCode

{

public static int countOdd(int[] arr,int n)

{

int c = 0;

for (int i = 0; i < n;i++ )

{

if (arr[i] < 0)

return -1;

}

for (int i = 0; i < n; i++)

{

if (arr[i] % 2 != 0)

{

c++;

}

}

return c;

}

}

}

**193.Calculate New Salary**

The HR of an IT company fixes the salary of a new joinee who joins the company with the previous experience. Given the year of experience, Technology expertise and the previous salary drawn, the HR fixes the current salary as follows: 1. Straight 30% hike from the previous salary drawn. 2. a. Add Extra 5% if the year of experience is more than 3 and less than or equal to 5 years. b. Add Extra 10% if the year of experience is more than 5 and less than or equal to 8 years. c. Add Extra 15% if the year of experience is more than 8 years 3. Technology expertise is classified broadly into two namely common skills (CS) and rare skills (RS) People having rare skills get an extra 5% increase from their previous salary drawn. Write a program to calculate the salary the HR fixes for the new joinee given the experience (input1), technology expertise classification (input2) and the previous salary drawn (input3) and print the output in the given format  
 Your Salary is fixed as Rs YYYYYY where YYYYYY is the calculated new salary. (The digits depends upon the salary calculated.) Salary is rounded off and displayed as an integer.  
 Business rules: 1. If the experience is given more than 25 or less than 0, then print “Invalid Experience” 2. If the technology expertise classification is given other than CS or RS, then print “Invalid Technology expertise classification” 3. If the previous salary is given more than 100000 or less than 0, then print “Invalid Salary”    
 Create a class named UserProgramCode that has the following static method   
 public static int calculateNewSalary(int,string,int)  
 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 that corresponds to experience (input 1).  
 The second line of the input consists of a string that corresponds to technology expertise (input 2).  
 The third line of the input consists of an integer that corresponds to previous salary drawn (input 3).  
 Refer business rules and sample output for formatting specifications. Sample Input 1 : 7 RS 30000    
 Sample Output 1 :  
 Your Salary is fixed as Rs 43500 Sample Input 2 : 7 RSS 30000  
 Sample Output 2 :  
 Invalid Technology expertise classification

class Program

{

public static void Main(string[] args)

{

int experience, oldSalary, salary;

string expertise;

experience = Convert.ToInt32(Console.ReadLine());

expertise = Console.ReadLine();

oldSalary = Convert.ToInt32(Console.ReadLine());

salary = UserProgramCode.calculateNewSalary(experience, expertise, oldSalary);

if (salary == -1)

Console.WriteLine("Invalid Experience");

else if (salary == -2)

Console.WriteLine("Invalid Technology expertise classification");

else if (salary == -3)

Console.WriteLine("Invalid Salary");

else

Console.WriteLine("Your Salary is fixed as Rs {0}", salary);

Console.ReadLine();

}

}

class UserProgramCode

{

public static int calculateNewSalary(int experience,string expertise,int oldSalary)

{

double sal = 0,ctr=0,sa=0;

if (experience > 25 || experience < 0)

{

sa = -1;

ctr++;

}

if (expertise != "RS" && expertise != "CS")

{

sa = -2;

ctr++;

}

if (oldSalary > 100000 || oldSalary < 0)

{

sa = -3;

ctr++;

}

if(ctr==0)

{

sal = sal + (oldSalary \* 0.3);

if (experience > 3 && experience <= 5)

{

sal = sal + (oldSalary \* 0.05);

}

else if (experience > 5 && experience <= 8)

{

sal = sal + (oldSalary \* 0.10);

}

else if (experience > 8)

{

sal = sal + (oldSalary \* 0.15);

}

if(expertise=="RS")

{

sal=sal+(oldSalary\*0.05);

}

sal = sal + oldSalary;

sa = Math.Round(sal);

}

return (int)sa;

}

}

**194. Count Sequential Characters**    
 Write a program to count the number of characters which gets repeated 3 times consecutively. If no character gets repeated 3 times consecutively, then print -1. Create a class named UserProgramCode that has the following static method   
 public static int countSequentialChars(string 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 a string.  
 Output is an integer.  
    
 Sample Input 1 : abcXXXabc    
 Sample Output 1 :  
 1 Sample Input 2 :  
 aaaxxyzAAAx    
 Sample Output 2 :  
 2

66)using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication24

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

int a = UserProgramCode.countSequentialChars(s);

if(a==-1)

Console.WriteLine("No Repeated Words Found");

else

Console.WriteLine(a);

Console.ReadLine();

}

}

class UserProgramCode

{

public static int countSequentialChars(string s)

{

int l = s.Length;

string[] st = new string[50];

for (int k = 0; k < l; k++)

{

st[k] = s.Substring(k, 1);

}

int count = 0;

int c=0;

for (int k = 0; k < l - 1; k++)

{

if (st[k] == st[k+1])

count++;

else

count = 0;

if (count == 2)

c++;

}

if(c==0)

return -1;

else

return c;

}

}

}

**195. Find Total number of days in given month**  Write code to find out total number of days in the given month for the given year.  
 Month is coded as: Jan=0, Feb=1 ,Mar=2 ...  
 Include a class UserProgramCode with static method  getNumberOfDays that accepts two integers and return type should be int.  
 Create a class Program which would get the input and call the static method getNumberOfDays(int year, int month) present in the UserProgramCode. Return the  result from getNumberOfDays and dispaly the result in Program class.    
  Input and Output Format : The first integer represent the year. The second integer represents the month The output is an interger which is number of days in the given month. SAMPLE INPUT 1:  
 2000 1 SAMPLE OUTPUT 1: 29

44. Find total number of days in given month

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Workout44

{

class UserProgramCode

{

public static int getNumberOfDays(int y,int m)

{

int d;

d = System.DateTime.DaysInMonth(y,m+1);

return d;

}

}

class Program

{

static void Main(string[] args)

{

//UserProgramCode u = new UserProgramCode();

int y,m,n;

y = int.Parse(Console.ReadLine());

m= int.Parse(Console.ReadLine());

n = UserProgramCode.getNumberOfDays(y,m);

Console.WriteLine(n);

}

}

}

**196. ValidateNumber**

Write a code to check the if the given number is validate or not. This string is expected to contain a 10 digit number in the format XXX-XXX-XXXX where 'X' is a digit. Include a class UserProgramCode with static method validateNumber which  accepts String value. The validateNumber function return type is an interger.Return 1 if the given string meets this format else return -1.if the function returns 1 then print as "Valid Number" and if it returns -1 then print as "Invalid Number".  
 Create a class Program which would get the input and call the static method validateNumber present in the UserProgramCode.  
 Input Output Format: The input consists of String. The output consists of a String "Valid Number" or "Invalid Number".

Sample input 1: 452-789-4568

Sample output 1: Valid Number

Sample input 2: 1234-234-123

Sample output 2: Invalid Number

62

class Program

{

static void Main(string[] args)

{

int f = 0;

string s;

s = Console.ReadLine();

userprogramcode obj = new userprogramcode();

f=obj.validatenumber(s);

if(f==1)

Console.WriteLine("Valid number format");

if(f==-1)

Console.WriteLine("Invalid number format");

}

}

public class userprogramcode

{

public int validatenumber(string s)

{

if (Regex.IsMatch(s, @"^\d{3}[-]\d{3}[-]\d{4}$"))

{

return 1;

}

else

return -1;

}

}

**197. Print Digit Sum** Write a program that accepts a string input and finds the sum of all numeric digits present in the string.  
 Example 1: input : abc12de4 output : 7 Example 2: input : udjc&23er output : -1 Business Rules : 1. If the given input string contains any special characters, then print -1. 2. f the given input string contains no numbers,then print -2. Create a class named UserProgramCode that has the following static method  
 public static int getdigits(string 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 a String  
 Output is an integer.  
 Sample Input 1:  
 abc12de4  
 Sample Output 1:  
 7  
    
 Sample Input 2:  
 udjc&23er  
 Sample Output 2:  
 -1

Program 75:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace ConsoleApplication18

{

class Program

{

static void Main(string[] args)

{

string s=Console.ReadLine();

int i=userProgramCode.getdigits(s);

Console.WriteLine(i);

Console.ReadLine();

}

}

class userProgramCode

{

public static int getdigits(string i)

{

int s=0;

for (int j = 0; j < i.Length; j++)

{

if ((char.IsLetterOrDigit(i[j])))

{

if (char.IsDigit(i[j]))

s += (Convert.ToInt32(i[j])-48);

}

else

return -1;

}

if (s == 0)

return -2;

else

return s;

}

}

}

**198. Strong Number**    
 Write a program to find whether the given integer input is a strong number or not. If the sum of each digits factorial is the same as the given input value then it is a strong number. If the Input1 is strong number then print "Input1 is a Strong Number" where Input1 is the input integer value. (Refer Example) Business rule: 1) If the Input1 value is not a strong number then print "Sum of all digits factorial is XX" where XX is the total of each digits factorial value. 2) Print "Invalid Input" when given input number is a negative number. Example:1 Input1: 145 1!+4!+5! = 1+24+120 = 145 Output1: 145 is a Strong Number Example:2 Input1: 25 2!+5! = 2+120 = 122 Output1: Sum of all digits factorial is 122    
 Create a class named UserProgramCode that has the following static method  
 public static String checkStrongNumber(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 a single integer.  
 Output is a string.  
    
 Sample Input 1:  
 145  
 Sample Output 1:  
 145 is a Strong Number  
    
 Sample Input 2:  
 25  
 Sample Output 2:  
 Sum of all digits factorial is 122  
    
    
 Program 74:

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

namespace ConsoleApp8

{

class Program

{

static void Main(string[] args)

{

int n = Convert.ToInt32(Console.ReadLine());

string r = UserMainCode.checkStrongNumber(n);

Console.WriteLine(r);

Console.ReadLine();

}

}

class UserMainCode

{

public static String checkStrongNumber(int number)

{

int fact = 1, temp = 0, ans = 0;

int res = number;

while (number > 0)

{

temp = number % 10;

for (int i = 1; i <= temp; i++)

{

fact = fact \* i;

}

ans = ans + fact;

number = number / 10;

fact = 1;

}

if (res == ans)

{

return ans.ToString() +" is a Strong Number";

}

else

{

return "Sum of all digits factorial is " + ans.ToString();

}

}

}

}

**199.Find Span**    
 Given an integer array as input, write a program to find the size of the largest Span in the given array,  
 Note: Span is the number of elements between two repeated numbers including both numbers. Assume an array with single element has a span of 1. Business rule: If there is no number repeated in an array, print -1. If there are two repeated integers in the input array, consider the first number and return the span.    
 Create a class named UserProgramCode that has the following static method   
 public static int getMaxSpan(int[] 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 : 5 1 2 1 1 3    
 Sample Output 1:  
 4 Sample Input 2 : 7 1 4 2 1 4 1 5    
 Sample Output 2 :  
 6

 using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

namespace ConsoleApp8

{

class Program

{

static void Main(string[] args)

{

int size;

size = Convert.ToInt32(Console.ReadLine());

int[] arr = new int[size];

for (int i = 0; i < size; i++)

{

arr[i] = Convert.ToInt32(Console.ReadLine());

}

int res = UserMainCode.getMaxSpan(size, arr);

Console.WriteLine(res);

Console.ReadLine();

}

}

class UserMainCode

{

public static int getMaxSpan(int size, int[] arr)

{

int span = 0, temp = 0, flag = 0 ;

for (int i = 0; i < arr.Length; i++)

{

for (int j =i+1; j < arr.Length; j++)

{

if (arr[i]==arr[j])

{

flag++;

}

}

}

if (flag==0)

{

span = -1;

}

else

{

for (int i = 0; i < arr.Length; i++)

{

for (int j = 0; j < arr.Length; j++)

{

if (arr[i]==arr[j])

{

temp = j - i+1;

}

if (temp>span)

{

span = temp;

}

}

}

}

return span;

}

}

}

**200. Longest Palindrome**  
 Given an input string input1, write a program to find the length of the longest substring which is a palindrome. Palindrome is a word, phrase, or sequence that reads the same backwards as forwards e.g. madam Ignore case sensitivity for the input strings. Business Rule: 1) If the input string contains any number,then print -1. 2) If the input string contains any special characters,then print -2. 3) If the input string does not contain a string palindrome,then print -3. Please note that a single character is not considered to be palindrome. Create a class named UserProgramCode that has the following static method   
 public static int longestPalindrome(string 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 a string.  
 Output is an integer.  
 Refer business rules and sample output for output format.  
 Sample Input 1: seaesstringnirts  
 Sample Output 1: 11 Sample Input 2: sea34esstringnirts Sample Output 2 :  
 -1

91.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Collections;

namespace ConsoleApplication23

{

class UserProgramCode

{

public static int longestPalindrome(string s1)

{

string s = s1;

s = Regex.Replace(s, @"\s+", " ");

string[] s2 = s.Split(' ');

char[] a2=new char[100];

for(int i=0;i<s2.Length;i++)

{

for(int j=0;j<s2[i].Length;j++)

{

if(char.IsDigit(s2[i][j]))

{

return -1;

}

}

}

for (int i = 0; i < s2.Length; i++)

{

for (int j = 0; j < s2[i].Length; j++)

{

if (!char.IsLetter(s2[i][j]))

{

return -2;

}

}

}

for (int i = 0; i < s2.Length; i++)

{

int flag = 0;

string r=s2[i];

char[] a1=r.ToCharArray();

int k = a1.Length;

for (int i1 = 0; i < k; i1++)

{

a2[i1] = a1[k-1];

k--;

}

for (int j = 0; j < s2[i].Length; j++)

{

if (s2[i][j] == a2[j])

{

flag = 1;

}

else

{

break;

}

}

if (flag == 1)

{

return flag;

}

}

return 0;

}

static void Main(string[] args)

{

string a;

a=Console.ReadLine();

int flag;

flag =longestPalindrome(a);

Console.WriteLine(flag);

}

}

}

**201. Find Occurence**  
 Write a method to find the occurence (number of times) of a given character  in a given input string. Include a class UserProgramCode with a static method findOccurence which accepts a string and character as input and returns an integer. Business Rules: 1. Search criteria is irrespective of cases. 2. The input string should consists of only alphabets, no special characters or numbers should be there in the string. If present, the method returns -1.. Create a class Program which would get the input and call the static method findOccurence present in the UserProgramCode. If the method returns -1, print 'Invalid Input'. Input and Output format :  Input consists of string and character.  Refer sample output for formatting specifications. Sample Input 1 : HELLO friends Welcome to CSharp wonderful world char=’L’ Sample Output 1 : 5 Sample Input 2 : Gr8...I am fine. 8 Sample Output 2 : Invalid Input

69)using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

char c = Convert.ToChar(Console.ReadLine());

int a = UserProgramCode.findOccurence(s,c);

if (a == -1)

Console.WriteLine("Invalid Input");

else

Console.WriteLine(a);

Console.ReadLine();

}

}

class UserProgramCode

{

public static int findOccurence(string a,char b)

{

int count = 0;

if (a.Any(ch => !(Char.IsLetter(ch) || Char.IsWhiteSpace(ch))))

{

count = 0;

}

else

{

foreach (char item in a)

{

if (item == Char.ToUpper(b) || item == Char.ToLower(b))

{

count++;

}

}

}

return count;

}

}

}