31.Relative Order  
    
 Given two input integer arrays input1 and input2, write a program to sort input1 in such a way that the relative order among the elements will be same as those are in input2. For the elements not present in input2,append them at last in sorted order. Business Rules : 1. If any of the given inputs contains any negative number, then print -1. 2. If any of the elements in input 2 array is not available in input 1 array, then print -2. 3. If there are less than 3 elements or more than 15 elements in the input1 array, print -3.    
 Example  
 Input Array 1 = {2,1,2,5,7,1,9,3,6,8,8} Input Array 2 = {2,1,8,3| Output Array = {2,2,1,1,8,8,3,5,6,7,9} Create a class named UserProgramCode that has the following static method   
 public static int[] relativeOrder(int[] input1,int[] 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 1.  
 The next 'n' lines of input consist of elements in the input array 1.  
 The next line of the input consists of an integer, m that corresponds to the number of elements in the input array 2.  
 The next 'm' lines of input consist of elements in the input array 1.  
 Refer business rules and sample output for formatting specifications. Sample Input 1 : 11 2 1 2 5 7 1 9 3 6 8 8 4 2 1 8 3 Sample Output 1 : 2 2 1 1 8 8 3 5 6 7 9    
 Sample Input 2 : 8 2 1 5 7 9 3 6 8 4 2 1 8 3    
 Sample Output 2 : 2 1 8 3 5 6 7 9 Sample Input 3 : 11 2 1 2 -5 7 1 9 3 6 8 8 4 2 1 8 3    
 Sample Output 3 :  
 -1

program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Relative\_Order

{

class Program

{

static void Main(string[] args)

{

int n1 = int.Parse(Console.ReadLine());

int[] arr1 = new int[n1];

for (int i = 0; i < n1; i++)

{

arr1[i] = int.Parse(Console.ReadLine());

}

int n2 = int.Parse(Console.ReadLine());

int[] arr2 = new int[n2];

for (int i = 0; i < n2; i++)

{

arr2[i] = int.Parse(Console.ReadLine());

}

int[] op = UserProgramCode.Relative\_order(arr1, arr2);

{

foreach (var item in op)

{

Console.WriteLine(item);

}

}

}

}

}

userprogram.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Relative\_Order

{

class UserProgramCode

{

public static int[] Relative\_order(int[] arr1, int[] arr2)

{

List<int> list=new List<int>();

List<int> list1 = new List<int>();

for (int i = 0; i < arr2.Length; i++)

{

for (int j = 0; j < arr1.Length; j++)

{

if (arr1[j] == arr2[i])

{

list.Add(arr1[j]);

}

}

}

for (int i = 0; i < arr1.Length; i++)

{

if(!list.Contains(arr1[i]))

{

list1.Add(arr1[i]);

}

}

list1.Sort();

List<int> finalList = new List<int>(list);

for (int i = 0; i < list1.Count; i++)

{

finalList.Add(list1[i]);

}

return finalList.ToArray();

}

}

}

32. Repeat Characters  
 Write a program to repeat the string multiple times provided with the below limitations. a. Consider the index position of the input word starts with 1. b. If the Input1 string length is odd, then the even index position characters should be removed from the input string and the remaining characters should be repeated based on Input2 value. c. If the Input1 string length is even then the odd index position characters should be removed from the input string and the remaining characters should be repeated based on Input2 value. Business Rules : 1. If the Input2 value is negative, then print “Invalid Input”. 2. If the Input2 value is greater than 10, then print “Input value is too long”. 3. If the length of Input1 is less than 2, then print “Input value is insufficient”.    
 Create a class named UserProgramCode that has the following static method   
 public static string repeatRemoveString(string input1, int 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 a string and an integer.  
 Output is a string.  
 Refer business rules and sample output for formatting specifications. Sample Input 1 : Price 4  
 Sample Output 1 :  
 PiePiePiePie Sample Input 2 :  
 A 8    
 Sample Output 2 :  
 Input value is insufficient

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Repeated\_Characcters

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

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

int len = str.Length;

string op = UserProgramCode.repeatRemoveString(str, n);

if (op == "-1")

Console.WriteLine("nvalid Input");

else if (op == "-2")

Console.WriteLine("Input Value is too long");

else if (op == "-3")

Console.WriteLine("Input value is insufficient");

else

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

{

Console.Write(op);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Repeated\_Characcters

{

class UserProgramCode

{

public static string repeatRemoveString(string input1, int input2)

{

int len = input1.Length;

string s = "";

if (input2 < 0)

{

return "-1";

}

else if (input2 > 10)

{

return "-2";

}

else if (len < 2)

{

return "-3";

}

else if(len % 2 != 0)

{

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

{

if (i % 2 == 0)

{

s = s + input1[i];

}

}

}

else

{

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

{

if (i % 2 != 0)

{

s = s + input1[i];

}

}

}

return s;

}

}

}

33. Get the longest string  
 Write a program to get the longest string from the list which starts with the given character.Assume that input comparison  is done  irrespective of case. ie case insensitive.  
    
 Include a class UserProgramCode with a static method  getLongestString which accepts a String list and a character. The return type  is a string.  
 Create a Class Program which would be used to accept the size of the string list, the list elements and the search character and calls the static method present in UserProgramCode.  
    
 In getLongestString  
 1. If there is no element found list, then return the string "No elements found "  
 2. Only alphabets should be given in the list. Otherwise return the string, "String contains non alphabetic characters. "  
 3.I f the two or more strings start with the given character ,the longest string should be returned. Assume that the longest string will be unique. Input Output format  
 First line points to the size of the string list as n.  
 The next n lines points to elements of the string list.  
 The last input points to the character.  
 Output consists of a string. SAMPLE INPUT 1:  
 4 Yellow Red Black Blue b SAMPLE OUTPUT 1: Black SAMPLE  INPUT 2: 3 Black White 45 W SAMPLE OUTPUT 2: String contains non alphabetic characters.  

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Get\_Logest\_String

{

class UserProgramCode

{

public static string LOngeststing(string[] arr, char s)

{

List<string> list = new List<string>();

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

{

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

{

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

{

return "Only alphabets should be";

}

}

}

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

{

if (arr[i].StartsWith(s.ToString()))

{

list.Add(arr[i]);

}

}

if (list.Count == 0)

return "No elements Found";

else

{

int max = 0;

foreach (string str in list)

{

if (str.Length > max)

max = str.Length;

}

foreach (string str in list)

{

if (str.Length == max)

{

return str;

}

}

}

return "";

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Get\_Logest\_String

{

class Program

{

static void Main(string[] args)

{

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

string[] arr = new string[n];

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

{

arr[i] = Console.ReadLine();

}

Char s = Convert.ToChar(Console.ReadLine());

string op = UserProgramCode.LOngeststing(arr, s);

Console.WriteLine(op);

Console.ReadLine();

}

}

}

34. Special Characters  
 Write a program that accepts a string input and removes all the alphabetic characters from input and stores only the special characters and digits. Note: Special characters are #, $,%,& Business Rules : 1. if the given input string contains no numbers or special characters,then print -1. Create a class named UserProgramCode that has the following static method  
 public static string getSpecialChar(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 either '-1' or the processed string.  
    
 Sample Input 1:  
 cogniz$#45Ant Sample Output 1:  
 $#45 Sample Input 2: Treasure  
 Sample Output 2: -1  
  

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Collections;

namespace trial

{

class Program

{

static void Main(string[] args)

{

String str = Console.ReadLine();

String res = UserProgramCode.getSpecialChar(str);

Console.WriteLine(res);

}

}

}

//UserProgramCode.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace trial

{

class UserProgramCode

{

public static string getSpecialChar(string input1)

{

int num = 0;

StringBuilder sb=new StringBuilder();

int sp = 0;

int len = input1.Length;

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

{

char c = input1[i];

if (char.IsDigit(c))

{

num++;

sb.Append(c);

}

if (c == '#' || c == '$' || c == '%' || c == '&')

{

sp++;

sb.Append(c);

}

//else if (!char.IsLetter(c))

//{

// sb.Append(c);

//}

}

if (num == 0 || sp == 0)

{

return "-1";

}

else

return sb.ToString();

}

}

}

35. Arithmetic Operation  
 Write a program to perform the user specified arithmethic operation. The program will consist of 3 parameters. First one for specifying the type of operation (+,-,\*,/) and the other two are the operands upon which the operation has to be performed. Print the final output.  
    
 Business Rules:  
    
 1. The first parameter should have the values as 1,2,3 or 4. If it has any other value other than this, return -1.  
 2. The Second and the third parameter should be only positive numbers, else return -2.  
 3. If the first parameter is   
    
 1 --------------- Add the second and third parameter. (second+third)  
    
 2 --------------- Subtract the second and third parameter.(second-third)  
    
 3 ---------------- Multiply second and third parameter. .(second\*third)  
    
 4 ---------------- Divide second by third parameter.(second/third)  
    
 Include a class UserProgramCode with a static method arithmeticOperation which accepts three integers. The return type (Integer) should return the result of the operation performed. Return -1 or -2 according to the Business rules.  
 Create a Class Program which would be used to accept three integers, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of three integers, where the first integer corresponds to the type of operator, the second and third integer corresponds to the operands.  
 Output consists of an Integer or, a String “Invalid operator” if the -1 is returned, “Invalid operands” if -2 is returned.  
    
 Assume all outputs are Integers.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 1  
 2  
 3  
 Sample Output 1:  
 5  
    
 Sample Input 2:  
 5  
 2  
 3  
 Sample Output 2:  
 Invalid operator  
    
 Sample Input 3:  
 1  
 -2  
 3  
 Sample Output 3:  
 Invalid operands  
  

qn..22

class Program

{

static void Main(string[] args)

{

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

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

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

int s1 = UserProgramCode.arithmeticOperation(a,b,c);

if (s1 == -1)

Console.WriteLine("Invalid Operator");

else if (s1 == -2)

Console.WriteLine("Invalid Operands");

else

Console.WriteLine(s1);

Console.ReadLine();

}

}

class UserProgramCode

{

public static int arithmeticOperation(int a,int b,int c)

{

if (a > 0 && a < 5)

{

if (b < 0 || c < 0)

return -2;

else

if (a == 1)

return b + c;

else if (a == 2)

return b - c;

else if (a == 3)

return b \* c;

else

return b / c;

}

else

return -1;

}

}

36. Get Middle Characters  
 Write a program to read a string of even length and to fetch two middle most characters. Print the output.  
    
 Example:  
 Input = this  
 Output1 = hi  
    
 Include a class UserProgramCode with a static method getMiddleChars which accepts a String. The return type (String) should return the output String.  
 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 of even length.  
 Output consists of a String, the middle two letters  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 This  
 Sample Output 1:  
 hi

qn..23

class Program

{

static void Main(string[] args)

{

string str=Console.ReadLine();

string s1 = UserProgramCode.getMiddleChars(str);

Console.WriteLine(s1);

Console.ReadLine();

}

}

class UserProgramCode

{

public static string getMiddleChars(string s)

{

int i = 0;

StringBuilder sb = new StringBuilder();

if (s.Length % 2 == 0)

{

i = s.Length / 2;

sb.Append(s[i - 1]);

sb.Append(s[i]);

}

return sb.ToString();

}

}

37. Sum Of Digits  
 Write a program to read a String and to get the sum of all the digits present in the given string. Print the sum. If there is no digit in the given string print “No digit present”.  
    
 Example1:  
 Input = good23bad4  
 Output = 2 + 3 + 4 = 9  
    
 Include a class UserProgramCode with a static method sumOfDigits which accepts a String. The return type (Integer) should return the sum, or return -1 if no digits are present.  
 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 an Integer or a String “No digit present “ ..  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 good23bad4  
 Sample Output 1:  
 9  
    
 Sample Input 2:  
 good@#bad$  
 Sample Output 2:  
 No digit present

20)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Fwd\_Prgs

{

public class UserProgramCode

{

public static string sumOfDigits(string str)

{

char[] s = str.ToCharArray();

int sum=0;

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

if ((s[i] > 47) && (s[i] < 58))

{

sum = sum + (int)s[i]-48;

}

string res = sum.ToString();

if (sum == 0)

return "-1";

else

return res;

}

}

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

string res = UserProgramCode.sumOfDigits(s);

if(res=="-1")

Console.WriteLine("No digit present");

else

Console.WriteLine(res);

}

}

}

38. Unique Counter  
 Write a program that reads a string and finds the number of unique characters in the string (ie the number of characters in the string that appear only once in the string). If the given string does not contain any unique characters print “No unique characters”.  
    
 Example -   
 Input: “HelloWorld”   
 Output: 5   
    
 Input: “coco”  
 Output: “No unique characters”  
    
 Include a class UserProgramCode with a static method uniqueCounter which accepts a String. The return type (Integer) should return the number of unique characters or -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.  
 Output consists of an Integer(number of unique characters) or a String (“No unique characters” if no unique characters are present).  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 HelloWorld  
 Sample Output 1:  
 5  
    
 Sample Input 2:  
 coco  
 Sample Output 2:  
 No unique characters

UNIQue COUNTER

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace ConsoleApplication20

{

class Program

{

static void Main(string[] args)

{

string str1;

int x;

str1 = Console.ReadLine();

x = UserProgramCode.uniqueCounter(str1);

if (x == -1)

Console.WriteLine("No unique characters");

else

Console.WriteLine(x);

Console.Read();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication20

{

class UserProgramCode

{

public static int uniqueCounter(string str)

{

int count = 0;

int count1 = 0;

char[] ch=str.ToCharArray();

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

{

count1 = 0;

char c = ch[i];

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

{

char cd = ch[j];

if (cd == c)

{

count1++;

}

}

if (count1 == 1)

{

count++;

}

}

if (count == 0)

return -1;

else

return(count);

}

}

}

39. Test Vowels Order  
    
 Write a program that takes a string as input and checks whether the given string contains exactly five vowels and the vowels should be in alphabetical order. Assume there is no repetition of any vowel in the given string. If the above condition is satisfied, print 1. If the above condition is not satisfied, print 2. Business rule: If there is a repetition of vowels in the given string, print -1. Create a class named UserProgramCode that has the following static method   
 public static int testOrderVowels (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 formatting specifications. Sample Input :  
 acebisouzz Sample Output :  
 1

TestVowelsOrder

using System;

class Program

{

public static void Main( string[] args )

{

string input=Console.ReadLine();

int result = UserProgramCode.testOrderVowels(input);

Console.WriteLine(result);

Console.Read();

}

}

using System;

using System.Linq;

using System.Text;

class UserProgramCode

{

public static int testOrderVowels(string input)

{

int res=0;

string result="";

input.ToLower();

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

{

if (input[i] == 'a' || input[i] == 'e' || input[i] == 'i' || input[i] == 'o' || input[i] == 'u')

result += input[i];

}

if (result.Length>5)

{

res =- 1;

}

else if (result.Equals("aeiou"))

res = 1;

else

res = 2;

return res;

}

}

40. Symmetric Difference  
    
 Given two integer arrays Input1 and Input2, write a program to calculate the Symmetric Difference of the two input arrays. Symmetric difference is the difference of A Union B and A Intersection B ie. [(A U B) - (A ^ B)] where A is the Input1 array and B is the Input2 array. Union operation(U) merges the two arrays and makes sure that common elements appear only once. Intersection(^) operation includes common elements from both the arrays. Finally sort the output and print the array. Business Rules : 1. If any/all of the Input values in the Input1 array is negative, then print -1. 2. If any/all of the input values in the Input2 array is negative, then print -2. 3. If all the integers in Input1 array is common to Input2 array, then print -3. 4. If none of the integers in Input1 array is common to Input2 array, then print -3. Example 1: input: 5 Input1 : 11 5 14 26 3 input : 3 Input2 : 5 3 1 Output: 1 11 14 26 AUB: {11,5,14,26,3, 1} A^B: {5,3} AUB - A^B = {1, 11, 14, 26} Example 2: Input : 3 Input1 : 2 16 -9 Input : 3 Input2 : 53 43 31 Output1: -1    
 Create a class named UserProgramCode that has the following static method   
 public static int[] symmetricDifference(int[] input1, int[] 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 input1 array.  
 The next 'n' lines of input consist of elements in the input1 array.  
 The next line of the input consists of an integer, m that corresponds to the number of elements in the input2 array.  
 The next 'm' lines of input consist of elements in the input2 array.  
 Refer business rules and sample output for output formatting specifications.    
 Sample Input : 5 11 5 14 26 3 3 5 3  
 1 Sample Output : 1 11 14 26  

using System;

using System.Linq;

using System.Collections.Generic;

using System.Text;

class UserProgramCode

{

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

{

var union = input1.Union(input2);

var intersect = input1.Intersect(input2);

var symmetric = union.Except(intersect);

int[] result = new int[symmetric.Count()];

// Write intersection to screen.

int i = 0;

foreach (int value in symmetric)

{

result[i] = value;

i++;

}

return result;

}

}

---------

Program

---------

using System;

using System.Linq;

using System.Collections.Generic;

using System.Text;

class Program

{

public static void Main(string[] args)

{

int input1, input2;

input1 = Convert.ToInt32(Console.ReadLine());

int[] inputArr1 = new int[input1];

//int[] output = new int[10];

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

{

inputArr1[i] = Convert.ToInt32(Console.ReadLine());

}

input2 = Convert.ToInt32(Console.ReadLine());

int[] inputArr2 = new int[input2];

for (int i = 0; i < input2; i++)

{

inputArr2[i] = Convert.ToInt32(Console.ReadLine());

}

int[] output = UserProgramCode.symmetricDifference(inputArr1, inputArr2);

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

{

Console.WriteLine(output[i]);

}

Console.Read();

}

41. Gyrating Numbers  
 Write a program to find whether every integer in a given input integer array is in Gyrating form. Note: Gyrating numbers are numbers whose digits increase and decrease in a continuous repetitive cycle. Every integer of each element should increase or decrease in a continuous sequence.    
 Business rule: 1) Print 1 if the given input integer array is in Gyrating sequence.  
 2) Print -1 if the given input integer array is not in Gyrating sequence. 3) Print -2 if the given input integer array consists of a negative number. Example:1 Input: 4 12 321 235 532 Output: 1 Example:2 Input: 4  
 75 12 531 45 Output: 1 Create a class named UserProgramCode that has the following static method  
 public static int gyRating(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 in the input correspond to the elements in the array.  
 Output is an integer. Refer business rules. Sample Input :  
 4 12 321 235 532  
 Sample Output :  
 1

class UserProgramCode

{

public int check(int[] a,int len)

{

int r,cmax=0,cmin=0,count=0,i=0,rem=0,j=0;

StringBuilder sb = new StringBuilder();

foreach (int val in a)

{

i = 0; cmax = 0; cmin = 0;

int v = val;

if (val < 0)

return -2;

else

{

while (v > 0)

{

++j;

r = v % 10;

if (i == 0)

{

rem = r;

++i;

}

if (r == rem)

{

++cmin;

++cmax;

}

else if (r < rem)

{

++cmin;

rem = r;

}

else if (r > rem)

{

++cmax;

rem = r;

}

v = v / 10;

}

if (cmin == 1)

sb.Append("m");

if (cmax == 1)

sb.Append("M");

if (cmin == 1 || cmax == 1 )

{

++count;

}

else

return -1;

}

}

if (count == len)

{

//Console.WriteLine("true");

int counter = 0;

char odd = sb[1];

char even = sb[0];

if (!odd.Equals(even))

{

for (int ii = 0; ii < sb.Length; ii++)

{

if (ii % 2 == 0)

{

//Console.WriteLine("inside even pos");

if (sb[ii].Equals(even))

counter++;

else

return -1;

}

else

{

//Console.WriteLine("inside odd pos");

if (sb[ii].Equals(odd))

counter++;

else

return -1;

}

}

if (counter == count)

return 1;

}

else

return -1;

}

return 0;

}

}

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

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using ConsoleApplication23;

using ConsoleApplication23;

//using ConsoleApplication9;

namespace ConsoleApplication23

{

class Program

{

static void Main(string[] args)

{

UserProgramCode ob = new UserProgramCode();

int len;

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

int[] a = new int[len];

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

{

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

}

int res = ob.check(a,len);

Console.WriteLine(res);

Console.ReadLine();

}

}

}

42. Unique Even Sum  
    
 Write a program to remove all duplicate elements from an input array and return the sum of all even numbers. Example : Input: {1,2,7,2,4,8,9,6,8}  After removing duplicates : {1,7,4,9,6} Output: 4+6 = 10 Exception Rules:  
 If there is no even number in the input, return -1.  
 if input contains negative numbers, then return -2.  
 Include a class UserProgramCode with a static method addUniqueEven which accepts an integer array. The return type is an integer as given in the above statement.  
 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 values. The first value corresponds to size of the array. The next n numbers contains the integer values.  
 Output consists of a integer as mentioned in the problem statement.  
 Refer sample output for formatting specifications.  
 Sample Input 1:  
 9  
 1  
 2  
 7  
 2  
 4  
 8  
 9  
 6  
 8  
 Sample Output 1:  
 10  
    
 Sample Input 2:  
 5  
 1  
 3  
 5  
 7  
 9  
 Sample Output 2:  
 -1  
    
    
 Sample Input 3:  
 4  
 1  
 -2  
 6  
 8  
 Sample Output 3:  
 -2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Unique\_Even\_Sum

{

class UserProgramCode

{

public static int uniqueevensum(int[] a)

{

int flag = 0;

int len=a.Length;

int sum = 0;

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

{

if (a[i] < 0)

return -2;

}

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

{

if (a[i] % 2 == 0)

{

flag = 1;

break;

}

}

if (flag == 0)

return -1;

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

{

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

{

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

{

a[i] = 0;

a[j] = 0;

}

}

}

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

{

if (a[i] % 2 == 0)

{

sum = sum + a[i];

}

}

return sum;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Unique\_Even\_Sum

{

class Program

{

static void Main(string[] args)

{

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

int[] a =new int[n];

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

{

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

}

int op = UserProgramCode.uniqueevensum(a);

Console.WriteLine(op);

Console.ReadLine();

}

}

}

43. Check Palindrome  
 Write a program to read a string input and to check that given string is a palindrome and contains at least two distinct vowels.   
 The vowels can be repetitive. Even if the same vowel occurs more than once, it should be considered as one vowel only. If the above condition is satisfied, print “Palindrome”, else print “Not Palindrome”.  
    
 Include a class UserProgramCode with a static method checkPalindrome which accepts a String. The return type (Integer) should return 1 if Palindrome, else -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.  
 Output consists of a String, “Palindrome” or “Not Palindrome”.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 himesh  
 Sample Output 1:  
 Not Palindrome  
    
 Sample Input 2:  
 ABEBA  
 Sample Output 2:  
 Palindrome

33.CHECK\_PALLINDROME

using System;

class Program

{

public static void Main(string[] args)

{

string str = Console.ReadLine();

int output = UserProgramCode.checkPalindrome(str);

if (output == 1)

Console.WriteLine("Palindrome");

else

Console.WriteLine("Not Palindrome");

}

}

using System;

using System.Globalization;

using System.Text;

public class UserProgramCode

{

public static int checkPalindrome(string str)

{

char[] a = str.ToCharArray();

char[] b = str.ToCharArray();

Array.Reverse(b);

string stra = new string(a);

string strb = new string(b);

if (stra.Equals(strb))

{

char[] d = str.ToCharArray();

foreach (char item in stra)

{

if (item == 'a' || item == 'A' || item == 'e' || item == 'E' || item == 'i' || item == 'I' || item == 'o' || item == 'O' || item == 'u' || item == 'U')

{

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

{

if (d[i] == item)

{

}

else

if (d[i] == 'a' || d[i] == 'A' || d[i] == 'e' || d[i] == 'E' || d[i] == 'i' || d[i] == 'I' || d[i] == 'o' || d[i] == 'O' || d[i] == 'u' || d[i] == 'U')

{

return 1;

}

}

}

}

}

return -1;

}

}

44. String Array Sorting  
    
 Given a string array, write a function to remove the duplicate values from a String Array, sort the strings in ascending and display the string array.  
 The values 'AA' and 'aa' are NOT the same elements or duplicates. The case sensitive check should be implemented. While sorting, words starting with upper case letters should be considered first.  
    
 Business rules:  
 1) Print 'Invalid String' when the given input integer array consists of any special character or numbers.  
 2) All the elements in the array should be of same length. If not, then print 'Invalid String'.  
    
 Create a class named UserProgramCode that has the following static method   
 public static string[] orderStringElements(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 string array.  
 The next 'n' lines in the input correspond to the elements in the string array.  
 Output is a string array. Refer sample output and business rules Sample Input 1:  
 6  
 AAA  
 BBB  
 AAA  
 AAA  
 CCC  
 CCC  
 Sample Output 1:  
 AAA  
 BBB  
 CCC  
    
 Sample Input 2:  
 7  
 AAA  
 BBB  
 aaa  
 AAA  
 Abc  
 A  
 b  
 Sample Output 2:  
 Invalid String  
    
  83.

//program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace trial

{

class Program

{

static void Main(string[] args)

{

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

String[] ar = new String[n];

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

{

ar[i] = Console.ReadLine();

}

String[] ret = UserProgramCode.orderStringElements(ar);

foreach (String a in ret)

{

Console.WriteLine(a);

}

}

}

}

//UserProgramCode.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace trial

{

class UserProgramCode

{

public static string[] orderStringElements(string[] ar)

{

int len = ar[0].Length;

String pat = @"^[a-zA-Z]{"+len+"}$";

// Regex reg = new Regex(@"^[a-zA-Z]+$");

Regex reg1 = new Regex(pat);

String[] res = new String[1];

StringBuilder sb = new StringBuilder();

int n = ar.Length;

foreach (String aa in ar)

{

if (!reg1.IsMatch(aa))

{

res[0] = "Invalid String";

return res;

}

}

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

{

String a = ar[i];

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

{

if (a.Equals(ar[j]) && !a.Equals(""))

{

ar[j] = "-1";

}

}

if (!a.Equals("-1"))

{

if (i > 0)

sb.AppendLine();

sb.Append(a);

}

}

String[] array = sb.ToString().Split('\n');

Array.Sort(array, StringComparer.Ordinal);

return array;

}

}

}

45. Count Between Numbers   Write a program to find the count of elements in the range [l, h] (both inclusive) in an integer array. l corresponds to the lower value and h corrresponds to the higher value. Include a class UserProgramCode with a static method countBetweenNumbers which accepts an integer array and two other integers ( l and h). The return type is int. If there are negative numbers in the array or if the value of l or h is negative, return -1. Create a Class Program which would be used to read the inputs  and call the static method present in UserProgramCode. If the method returns -1, then print "Negative value Present"    
 Input and Output Format:  
 Input consists of n+3 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. The last  integers correspond to the lower value and higher value. Output consists of an integer or a string.  
 Refer sample output for formatting specifications.  
    
    
 Sample Input 1:  
 9 2 13 6 19 25 65 34 1 20 5 20  
 Sample Output 1:  
 4  
    
 Sample Input 2:  
 2  
 3 -5  
 2 3    
 Sample Output 2:  
 Negative value Present

Program 53:

=========--

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace progm53

{

class Program

{

static void Main(string[] args)

{

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

int[] a = new int[n];

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

{

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

}

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

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

int output=UserProgramCode.countBetweenNumbers(a, l, h);

if (output != -1)

{

Console.WriteLine(output);

}

else

{

Console.WriteLine("Negative value Present");

}

}

}

class UserProgramCode

{

public static int countBetweenNumbers(int[] inp,int x,int y)

{

int max1=inp.Length;

int count = 0;

for (int i = 0; i < max1; i++)

{

if ((inp[i] < 0) || (x < 0) || (y < 0))

{

count= -1;

}

else

{

if ((inp[i] >= x) && (inp[i] <= y))

{

count++;

}

}

}

return count;

}

}

}

46. String Manipulation  
 Write a program which can read two strings and add the reverse of the second string in the middle of the first string.  
 Print "Special character found" if the string consists of special characters, else print the final String.  
    
 Examples :  
 1)  
 String1 : Arun  
 String2: Ram  
    
 Output : ArmaRun  
 2)  
 String1 : Aruns  
 String2: Ram   
    
 Output : ArumaRns  
 Hint: If the first string contains odd number of characters   
 e.g. String1 is having 7 characters, then add the second reverse string after the 4 characters of the first string.  
    
 Include a class UserProgramCode with a static method stringManipulation which accepts two Strings. The return type (String) should return the final String.  
 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 a String( the final String), or “Special character found” if the string consists of special characters.  
    
 Refer sample output for formatting specifications.  
    
 Sample Input 1:  
 Arun  
 Ram  
 Sample Output 1:  
 ArmaRun  
    
 Sample Input 2:  
 Aruns  
 Ram   
 Sample Output 2:  
 ArumaRns  
    
 Sample Input 3:  
 Arun$  
 Ram   
 Sample Output 3:  
 Special character found

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication19

{

class Program

{

static void Main(string[] args)

{

string ans,str1,str2;

str1 = Console.ReadLine();

str2 = Console.ReadLine();

ans = UserProgramCode.stringmanipulation(str1,str2);

Console.WriteLine(ans);

Console.Read();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

namespace ConsoleApplication19

{

class UserProgramCode

{

public static string stringmanipulation(string str1, string str2)

{

string res1, res2, res3, rev = "";

int len, len2, len3;

len = str1.Length;

if (len % 2 == 0)

{

len2 = len / 2;

res3 = str1.Substring(len2);

}

else

{

len2 = (len / 2) + 1;

res3 = str1.Substring(len2);

}

len3 = str2.Length - 1;

while (len3 >= 0)

{

rev = rev + str2[len3];

len3--;

}

res1 = str1.Substring(0, len2);

res2 = res1 + rev + res3;

return res2;

}

}

}

47. Day of Week  
    
 Write a program to find out the day of week for given input date which is in string format (MM-dd-yyyy). The output should be in titlecase.  
 Include a class UserProgramCode with a static method getDay which accepts a string . The return type is string 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 a string.   
 Output consists of a string as mentioned in the problem statement.  
 Refer sample output for formatting specifications.  
 Sample Input 1:  
 07-13-2012  
 Sample Output 1:  
 Friday

Program.Cs

\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace dateTime

{

class Program

{

static void Main(string[] args)

{

string date1 ;

date1 = Console.ReadLine();

UserMainCode umc = new UserMainCode();

Console.WriteLine(umc.timecheck(date1));

Console.ReadKey();

}

}

}

UserMainCode.Cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Globalization;

namespace dateTime

{

class UserMainCode

{

public string timecheck(string date1)

{

DateTime dt1;

String result = "";

DateTime.TryParseExact(date1, "MM-dd-yyyy", null, System.Globalization.DateTimeStyles.None, out dt1);

result=Convert.ToString(dt1.DayOfWeek);

return result;

}

}

}

48. Find Leaders  
    
 Given an array of integer values as input,write a program that will fetch all the leaders in the array and print them after sorting them in ascending order.An element is a leader if it is greater than all the elements to its right side. Consider that the rightmost element is always a leader. Business Rules : 1. If the given input contains any negative number, then print -1. 2. If there are less than 2 elements or more than 10 elements in the input array,print -2. 3. If any of the elements in the input array are repetitive, then print -3. Create a class named UserProgramCode that has the following static method   
 public static List<int> findLeadersArray(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 sample output for formatting specifications.    
 Sample Input 1: 6 6 7 4 3 5 2    
 Sample Output 1: 2 5 7    
 Sample Input 2:  
 6 6 7 -4 3 5 2    
 Sample Output 2: -1

173)Leader number

Find leader

class UserProgramCode

{

public static List<int> findLeadersArray(int[] input)

{

List<int> l= new List<int>();

List<int> l1 = new List<int>();

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

{

if (input[i] < 0)

{

l.Add(-1);

return l;

}

}

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

{

l.Add(-2);

return l;

}

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

{

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

{

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

{

l.Add(-3);

return l;

}

}

}

l.Add(input[input.Length - 1]);

int temp = 0;

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

{

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

{

if (input[i] > input[j])

{

temp = input[i];

}

else

{

temp = 0;

break;

}

}

if (temp != 0)

{

l.Add(temp);

}

temp = 0;

}

l.Sort();

return l;

}

class Program

{

public static void Main( string[] args )

{

int size;

List<int> opli = new List<int>();

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

int[] arr = new int[size];

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

{

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

}

opli = UserProgramCode.findLeadersArray(arr);

for(int i=0;i<opli.Count;i++)

{

Console.WriteLine(opli[i]);

}

Console.ReadLine();

}

}

49. SortList Write a code to sort the given array of integers in ascending order.  Business Rules: Only positive numbers should be given as inputs to the integer array. Include a class UserProgramCode with static method sortList which accepts interger array The return type is a interger array. If the input consists of negative numbers,return -1. If the input array size is 0, return -2.  
 Create a class Program which would get the input and call the static method sortList present in the UserProgramCode.  
 If the sortList method returns -1 print "Invalid Input". If the sortList method returns -2 print "Input is Empty". Input Output Format: Input consists of a n+1. n represents the size of the array followed by the elements in the array. Output consists of a array which is sorted in ascending order.   Sample Input 1: 3 45 12 36 Sample Output 1 : Sorted Array : 12 36 45 Sample Input 2: 4 -1 56 89 45 Sample Output 2 : Invalid Input

Program 55:

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

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace progm55

{

class UserProgramCode

{

public static int[] sortList(int[] a)

{

int temp;

int maxi = a.Length;

if (maxi == 0)

{

a[0] = -1;

}

else

{

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

{

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

{

if (a[i] < 0)

{

a[0] = -1;

}

else

{

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

{

temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}

}

}

}

return a;

}

}

class Program

{

static void Main(string[] args)

{

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

int[] a=new int[n];

int[] output=new int[n];

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

{

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

}

output=UserProgramCode.sortList(a);

if (output[0] != -1)

{

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

{

Console.WriteLine(output[i]);

}

}

else

{

Console.WriteLine("Invalid Input");

}

}

50. Add Series  
 Write a program that takes an odd positive integer number as input and evaluates the following series: 1+3-5+7-9…+/-n. Example: input = 9 series = 1+3-5+7-9 output = -3  
 Create a class named UserProgramCode that has the following static method  
 public static int addSeries(int input1)  
 Create a class named Program that accepts an integer as input and calls the static method present in the UserProgramCode.  
 Input and Output Format:  
 Input consists of a single integer that corresponds to n.  
 Output consists of a single integer that corresponds to the sum of the series.  
 Sample Input: 9  
    
 Sample Output : -3

PROGRAM 59.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace level1\_59

{

class Program

{

static void Main(string[] args)

{

int n,x;

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

x = UserProgramCode.addSeries(n);

Console.WriteLine(x);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace level1\_59

{

class UserProgramCode

{

public static int addSeries(int a)

{

int t = 0, k = 1;

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

{

if (i == 0)

{

t = t + k;

}

else if (i == 1)

{

t = t + k;

}

else if (i % 2 != 0)

{

t = t + k;

}

else

{

t = t - k;

}

k = k + 2;

}

return t;

}

}

}

51. Word Form  
 Write a program that accepts an integer input and displays the given number in word form.The word form should include only billions,millions,thousands,hundreds wherever applicable. The starting alphabet of each word should be in capital except the word "and". Business Rules: 1)If the given integer is negative convert that to a positive number and append "Minus" before the word then dispaly the result. Create a class named UserProgramCode that has the following static method  
 public static string wordForm(int number)  
 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 an integer.  
 Output is a string.  
    
 Sample Input 1:  
 364576567 Sample Output1:  
 Three Hundred and Sixty Four Million Five Hundred and Seventy Six Thousand Five Hundred and Sixty Seven Sample Input 2: -1234 Sample Output 2:  
 Minus One Thousand Two Hundred and Thirty Four

79.

//program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace oops2

{

class Program

{

static void Main(string[] args)

{

int num;

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

String ret = UserProgramCode.wordForm(num);

Console.WriteLine(ret);

}

}

}

//UserProgramCode.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace oops2

{

class UserProgramCode

{

public static string wordForm(int n)

{

return \_toText(n, true);

}

private static string \_toText(long n, bool isFirst = false)

{

string result;

if (isFirst && n == 0)

{

result = "Zero";

}

else if (n < 0)

{

result = "Negative " + \_toText(-n);

}

else if (n == 0)

{

result = "";

}

else if (n <= 9)

{

result = new[] { "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", "Nine" }[n - 1] + " ";

}

else if (n <= 19)

{

result = new[] { "Ten", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen", "Nineteen" }[n - 10] + (isFirst ? null : " ");

}

else if (n <= 99)

{

result = new[] { "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy", "Eighty", "Ninety" }[n / 10 - 2] + (n % 10 > 0 ? "-" + \_toText(n % 10) : null);

}

else if (n <= 999)

{

result = \_toText(n / 100) + "Hundred " + \_toText(n % 100);

}

else if (n <= 999999)

{

result = \_toText(n / 1000) + "Thousand " + \_toText(n % 1000);

}

else if (n <= 999999999)

{

result = \_toText(n / 1000000) + "Million " + \_toText(n % 1000000);

}

else

{

result = \_toText(n / 1000000000) + "Billion " + \_toText(n % 1000000000);

}

if (isFirst)

{

result = result.Trim();

}

return result;

}

}

}

52.

Repeat Characters  
 Write a program to repeat the string multiple times provided with the below limitations. a. If Input1 string length is five or less than five, then the first three characters should be repeated based on Input2 value. b. If the Input1 string length is more than five then the last three characters should be repeated based on Input2 value Business Rules : 1. If the length of Input1 is less than 3, then print 'Input value is insufficient' 2. If the Input2 value is negative, then print 'Invalid Input' 3. If the Input2 value is greater than 10, then print 'Input value is too long' Example 1: Input1: Price Input2: 3 Output : PriPriPri Example 2: Input1: Sunday Input2: 4 Output: daydaydayday Example 3: Input1: So Input2: 5 Ouput: Input value is insufficient    
 Create a class named UserProgramCode that has the following static method  
 public static string repeatManipulateString(string input1, int 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 a string.  
 The second line of input consists of an integer.  
 Output is a string. Refer sample output and business rules for output formatting specifications.  
    
 Sample Input:  
 Price  
 3  
 Sample Output:  
 PriPriPri

80.

//program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace oops2

{

class Program

{

static void Main(string[] args)

{

String val = Console.ReadLine();

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

String ret = UserProgramCode.repeatManipulateString(val, num);

Console.WriteLine(ret);

}

}

}

//UserProgramCode.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace oops2

{

class UserProgramCode

{

public static string repeatManipulateString(string input1, int input2)

{

StringBuilder sb = new StringBuilder();

int len = input1.Length;

if (len < 3)

return "Input value is insufficient";

if (input2 < 0)

return "Invalid Input";

if (len <= 5)

{

for (int i = 0; i < input2; i++)

{

sb.Append(input1.Substring(0,3));

}

return sb.ToString();

}

else if (len > 10)

{

return "Input value is too long";

}

else if (len >5)

{

for (int i = 0; i < input2; i++)

{

sb.Append(input1.Substring(len-3, 3));

}

return sb.ToString();

}

return " ";

}

}

}

53. Shipping Cost  
 Write a program to compute the Cost of Book Shipping. The Shipping Cost is computed according to the shipping type and the package weight. The shipping rate is given below. Shipping types - Weight Rate (bahts/gram) Regular for first 2000 - 0.25 (basic charge) Regular exceeding 2000 - 0.35 for each Express uses the same rate as Regular + 50 bahts fee Note that the Shipping cost is computed from the possible valid minimum rate. Input1- Weight in grams Input2- Type of delivery ('R' Regular and 'X' Express) Create a class named UserProgramCode that has the following static method  
 public static float CalcShippingCost(float input1, char 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 a float that corresponds to the weight in grams.  
 The second line of the input consists of a character ('R' or 'X') that corresponds to the type of service.  
 Output consists of a single float that corresponds to the shipping cost. Output is displayed correct to 2 decimal places.  
 Sample Input:  
 4500  
 R  
 Sample Output:  
 1375.00  

Program 73:

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)

{

float f=float.Parse(Console.ReadLine());

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

float p=userProgramCode.CalcShippingCost(f,c);

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

Console.ReadLine();

}

}

class userProgramCode

{

public static float CalcShippingCost(float i, char c)

{

float p;

if (c == 'X')

i += 50;

if (i > 2000)

{

float d = i - 2000;

p =Convert.ToSingle(2000 \* 0.25);

p +=Convert.ToSingle( (d \* 0.35));

}

else

p = Convert.ToSingle(i \* 0.25);

return p;

}

}

}

54. String Processing II  
 Given a string input input1 ,form another string with the given input string using the following rules. Form the output string with only the last letter of each word of the given input sentence in capital letters separated by $. Dont store $ after the last letter in the output string. Example 1: Input1:This is a cat Output1:S$S$A$T Example 2: Input1:This7 is a cat Output1: -1 Business Rules : 1. If the given input string contains any number, print -1. 2. If the input contains any special characters, print -2. 3. If there is only one word in input1, then print -3.  
 Create a class named UserProgramCode that has the following static method  
 public static string formWordwithLastLetters(string input1)  
 Create a class named Program that accepts the inputs and calls the static method present in the UserProgramCode. Input and Output Format: Input consists of a string.  
 Output consists of a string or '-1' or '-2' or '-3'. Sample Input 1:  
 This is a cat  
 Sample Output 1:  
 S$S$A$T  
 Sample Input 2:  
 This7 is a cat  
 Sample Output 2:  
 -1  
    
  

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace String\_Process2**

**{**

**class UserProgramCode**

**{**

**public static string StringProcess2(string str)**

**{**

**string s = "";**

**int flag = 0;**

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

**{**

**if (Char.IsDigit(str[i]))**

**{**

**flag = 1;**

**return "-1";**

**}**

**}**

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

**{**

**if (flag == 0)**

**{**

**if (!(Char.IsLetterOrDigit(str[i])))**

**{**

**if (!char.IsWhiteSpace(str[i]))**

**{**

**flag = 1;**

**return "-2";**

**}**

**}**

**}**

**}**

**if (flag == 0)**

**{**

**Char[] ch = str.ToCharArray();**

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

**{**

**if(ch[i]==' ')**

**{**

**flag = 2;**

**}**

**}**

**}**

**if (flag != 2)**

**return "-3";**

**for (int i = 0; i < str.Length - 1; i++)**

**{**

**if (str[i + 1] == ' ')**

**{**

**s = s + str[i].ToString().ToUpper() + '$';**

**}**

**}**

**s = s + str[str.Length - 1].ToString().ToUpper();**

**return s;**

**}**

**}**

**}**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace String\_Process2**

**{**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**string str = Console.ReadLine();**

**string op = UserProgramCode.StringProcess2(str);**

**Console.WriteLine(op);**

**Console.ReadLine();**

**}**

**}**

**}**

**55.** Calculate Grade    
 Given an input as integer array with Student\_ID and marks as the array element for multiple students in the format. {Student\_ID\_1, Mark1, Student\_ID\_2, Mark2, Student\_ID\_3, Mark3, etc...}, write a program to calculate the grade of the student who has scored the maximum marks and print the output in the following format.  
    
 Student\_ID XXX has passed in YYY  
    
 where XXX is the Student\_ID and YYY is the grade. 1) If Mark is greater than or equal to 80, then store the grade as "DISTINCTION" 2) If Mark is less than 80 and greater than or equal to 60, then store the grade as "FIRST CLASS" 3) If Mark is less then to 60 and greater than or equal to 45 then store the grade as "SECOND CLASS" 4) If Mark is less than 45 and greater than or equal to 0, then store the grade as "FAIL". Business rules: 1) If the Input contains any negative numbers, then print “Invalid Input”. 2) If the number of elements in Input array is less than or equal to 2, then print “Grading is not possible”. 3) If the number of elements in the Input array is odd, then print “Scores not provided for all Students”. Create a class named UserProgramCode that has the following static method   
 public static string getGrade(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 : 10 101 80 102 75 103 50 104 60 100 40    
 Sample Output 1 : Student\_ID 101 has passed in DISTINCTION Sample Input 2 : 4 22 9 -5 6    
 Sample Output 2 :  
 Invalid Input  

**17.Calculate Grade**

**using System;**

**class Program**

**{**

**public 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());**

**}**

**string result = UserProgramCode.getGrade(arr);**

**Console.WriteLine(result);**

**Console.ReadLine();**

**}**

**}**

**using System;**

**class UserProgramCode**

**{**

**public static string getGrade(int[] arr)**

**{**

**int max = arr[1], i;**

**int id=arr[0];**

**string grade="";**

**for (i = 1; i < arr.Length; i=i+2)**

**{**

**if (arr[i] > 0 && arr[i-1]>0 && arr.Length>2 && arr.Length%2==0)**

**{**

**if (max < arr[i])**

**{**

**max = arr[i];**

**id = arr[i - 1];**

**}**

**}**

**else if(arr[i]<0||arr[i-1]<0)**

**{**

**grade = "Invalid Input";**

**return grade;**

**}**

**else if (arr.Length<= 2)**

**{**

**grade = "Grading is not possible";**

**return grade;**

**}**

**else if (arr.Length % 2 != 0)**

**{**

**grade = "Scores not provided for all Students";**

**return grade;**

**}**

**}**

**if (max >= 80)**

**{**

**grade = "Student\_ID "+id+" has passsed in DISTINCTION";**

**}**

**else if (max >= 60 && max < 80)**

**{**

**grade = "Student\_ID "+id+" has passed in FIRST CLASS";**

**}**

**else if (max >= 45 && max < 60)**

**{**

**grade = "Student\_ID "+id+" has passed SECOND CLASS";**

**}**

**else if (max >= 0 && max < 45)**

**{**

**grade = "FAIL";**

**}**

**return grade;**

**}**

**}**

**56.** Remove Vowels from Even Position  
 Write code to remove vowels from even position in the string. Return final string as output. Assume the first character is at position 1 in the given string.   
    
 Include a class UserProgramCode with static method removeEvenVowels that accepts the String .The return type is a string.  
 Create a class Program which would get the input and call the static method removeEvenVowels present in the UserProgramCode.  
    
 Input output format Input consists of a string. Output consists of a string. Sample Input 1: commitment Sample Output 1: cmmitmnt Sample Input 2: rythm Sample Output 2: rythm

**Program 52:**

**============**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace progm52**

**{**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**string inpt;**

**inpt = Convert.ToString(Console.ReadLine());**

**string outp=UserProgramCode.removeEvenVowels(inpt);**

**Console.WriteLine(outp);**

**}**

**}**

**class UserProgramCode**

**{**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace final\_f**

**{**

**class UserProgramCode**

**{**

**public static string removeEvenVowels(string a)**

**{**

**int maxi = a.Length;**

**for (int i = 1; i < maxi; i++)**

**{**

**if ((a[i] == 'a') || (a[i] == 'e') || (a[i] == 'i') || (a[i] == 'o') || (a[i] == 'u'))**

**{**

**a = a.Remove(i, 1);**

**maxi--;**

**}**

**}**

**return a;**

**}**

**}**

**}**

**}**

**}**

**57.** Shipping Cost  
 Write a program to compute the Cost of Booking for Shipping. The Shipping Cost is computed according to the shipping type and the package weight. The shipping rate is given below.   
 Shipping types - Weight Rate (bahts/gram)   
 Regular for first 2000 - 0.25 (basic charge)   
 Regular exceeding 2000 - 0.35   
 For each Express, use the same rate as Regular + 50 bahts fee  
 Note that the Shipping cost is computed from the possible valid minimum rate.   
    
 Input1- Weight in grams   
 Input2- Type of delivery ('R' Regular and 'X' Express)  
    
 Example:  
 Input1: 4500  
 Input2: R  
 Output1: 1375  
    
 Include a class UserProgramCode with a static method CalcShippingCost which accepts an integer(weight) and a character (type of delivery). The return type (integer) should return the shipping cost.  
 Create a Class Program which would be used to accept a integer value and a character, and call the static method present in UserProgramCode.  
 Input and Output Format:  
 Input consists of an integer and a character.  
 Output consists of an integer(the shipping cost).  
 Refer sample output for formatting specifications.  
    
 Sample input 1:  
 4500  
 R  
 Sample Output 1:  
 1375  
    
 Sample Input 2:  
 1800  
 X  
 Sample Output 2:  
 500

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace Shipping\_Cost**

**{**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**int f = int.Parse(Console.ReadLine());**

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

**int p = UserProgramCode.CalcShippingCost(f, c);**

**Console.WriteLine(p);**

**Console.ReadLine();**

**}**

**}**

**}**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace Shipping\_Cost**

**{**

**class UserProgramCode**

**{**

**public static int CalcShippingCost(int i, char c)**

**{**

**double p;**

**if (c == 'X')**

**i += 50;**

**if (i > 2000)**

**{**

**int d = i - 2000;**

**p = (2000 \* 0.25);**

**p =p+ ((d \* 0.35));**

**}**

**else**

**p = (i \* 0.25);**

**int a =(int)Math.Round(p,0);**

**return a;**

**}**

**}**

**}**

**58.** Train Tariff Calculation  
    
 Ram has to book his train tickets for travelling from Chennai to Pune through an online portal. The tariffs vary based on parameters such as the Date of booking(DOB) and the Date of Travel(DOT). Consider the number of days between DOB and DOT as NOD. The normal ticket cost from Chennai to Pune when the ticket booking is done minimum one month before is Rs 1000 in Sleeper class(SL). The normal ticket cost for AC class ratings are as follows : First Class AC (1AC):Rs 2500 Second Class AC(2AC): Rs 2000 Third Class AC(3AC) : Rs 1500 A. If NOD is from 21 days upto 30 days ,then the tariff is 10% more than normal ticket cost B. If NOD is from 11 days upto 20 days ,then the tariff is 20% more than normal ticket cost C. If NOD is from 4 days upto 10 days ,then the tariff is 30% more than normal ticket cost D.If NOD is upto 3 days ,then the tariff is 40% more than normal ticket cost. Write a program to calculate the total cost a person has to pay for their booking given their date of booking , the date of travel and the class of travel as input1, input2 and input3 respectively and print the output in the following format. Your ticket is confirmed and the booking cost is Rs YYYYYY where YYYYYY is the calculated booking cost. Print the booking cost as an integer. The date of booking and the date of travel are given as string in the format yyyy.mm.dd Business rules: 1. If the date of travel is less than 3 days from the date of booking, then the tickets cannot be booked and print “Short Notice and hence Tickets cannot be booked” . 2. If the date of booking or the date of travel are not in the date format, then print “Improper Date format in the input” . 3. If the date of travel is more than 90 days from the date of booking, then the tickets cannot be booked and print “Long Notice period and hence Tickets cannot be booked” . 4. If the class of travel is other than SL,1AC,2AC or 3AC, then print “Improper class of Travel” . Create a class named UserProgramCode that has the following static method   
 public static int calculateTrainTariff(string input1, string input2, string input3) 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 3 strings --- input1 (Date of Booking), input2 (Date of Travel) and input3(class of travel).  
 Refer business rules and sample output for formatting specifications. Sample Input 1 : 2014.05.15 2014.05.25 SL  
 Sample Output 1 :  
 1200 Sample Input 2 : 201405.15 2014.05.25 SL  
 Sample Output 2 :  
 Improper Date format in the input

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace ConsoleApplication1**

**{**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**string input1, input2, input3;**

**input1 = Console.ReadLine();**

**input2 = Console.ReadLine();**

**input3 = Console.ReadLine();**

**int output=UserProgramCode.calculateTrainTariff(input1, input2, input3);**

**//output = Math.Round(output, 0);**

**if (output > 5)**

**{**

**Console.WriteLine(output);**

**}**

**else**

**{**

**if (output == 1)**

**{**

**Console.WriteLine("Short Notice and hence Tickets cannot be booked");**

**}**

**if (output == 2)**

**{**

**Console.WriteLine("Improper Date format in the input");**

**}**

**if (output == 3)**

**{**

**Console.WriteLine("Long Notice period and hence Tickets cannot be booked");**

**}**

**if (output == 4)**

**{**

**Console.WriteLine("Improper class of Travel");**

**}**

**}**

**Console.ReadLine();**

**}**

**}**

**}**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace ConsoleApplication1**

**{**

**class UserProgramCode**

**{**

**public static int calculateTrainTariff(string input1, string input2, string input3)**

**{**

**double total = 0;**

**try**

**{**

**DateTime d1 = Convert.ToDateTime(input1);**

**DateTime d2 = Convert.ToDateTime(input2);**

**int year = d1.Year - d2.Year;**

**int month = d1.Month - d2.Month;**

**int days = d1.Day - d2.Day;**

**int totaldays = Convert.ToInt32(((d2 - d1).TotalDays)+1);**

**if (totaldays < 3)**

**{**

**return 1;**

**}**

**if (totaldays > 90)**

**{**

**return 3;**

**}**

**if ((input3 != "SL") && (input3 != "1AC") && (input3 != "2AC") && (input3 != "3AC"))**

**{**

**return 4;**

**}**

**if (totaldays >= 4 && totaldays <= 10)**

**{**

**if (input3 == "SL")**

**{**

**total = 1000 \* 1.30;**

**}**

**if (input3 == "1AC")**

**{**

**total = 2500 \* 1.30;**

**}**

**if (input3 == "2AC")**

**{**

**total = 2000 \* 1.30;**

**}**

**if (input3 == "3AC")**

**{**

**total = 1500 \* 1.30;**

**}**

**return (int)total;**

**}**

**if (totaldays >= 11 && totaldays <= 20)**

**{**

**if (input3 == "SL")**

**{**

**total = 1000 \* 1.20;**

**}**

**if (input3 == "1AC")**

**{**

**total = 2500 \* 1.20;**

**}**

**if (input3 == "2AC")**

**{**

**total = 2000 \* 1.20;**

**}**

**if (input3 == "3AC")**

**{**

**total = 1500 \* 1.20;**

**}**

**return (int)total;**

**}**

**if (totaldays >= 21 && totaldays <= 30)**

**{**

**if (input3 == "SL")**

**{**

**total = 1000 \* 1.10;**

**}**

**if (input3 == "1AC")**

**{**

**total = 2500 \* 1.10;**

**}**

**if (input3 == "2AC")**

**{**

**total = 2000 \* 1.10;**

**}**

**if (input3 == "3AC")**

**{**

**total = 1500 \* 1.10;**

**}**

**return (int)total;**

**}**

**if (totaldays == 3)**

**{**

**if (input3 == "SL")**

**{**

**total = 1000 \* 1.40;**

**}**

**if (input3 == "1AC")**

**{**

**total = 2500 \* 1.40;**

**}**

**if (input3 == "2AC")**

**{**

**total = 2000 \* 1.40;**

**}**

**if (input3 == "3AC")**

**{**

**total = 1500 \* 1.40;**

**}**

**return (int)total;**

**}**

**return 0;**

**}**

**catch(Exception ex)**

**{**

**return 2;**

**}**

**}**

**}**

**}**

**59.** Count Vowels Write code to check total number of vowels in the given string. Example: input = ""avinash"" output = 3 Include a class UserProgramCode with static method countVowles which accepts string and return an interger value. Create a class Program which would get the input and call the static method countVowles present in the UserProgramCode.  
    
 Input and Output Format: Input consists of a string Output consists of a integer which corresponds to the number of vowels in the input string. Sample Input 1: suraj Sample Output 1: 2 Sample Input 2: why Sample Output 2: 0

**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;**

**}**

**}**

**}**

**60.** Image Types    
 Given a string array input which consists of image file names along with their respective image type extensions in the format ("filename.extensiontype",..so on). The image file name and the extension are seperated by a dot (.)operator. Write a program to calculate the count of image files having same extension type and store the values in the output string array variable in the below format. output (Key,Value) = (ExtensionType1,count1,ExtensionType2,count2,...so on) . Output should be stored in descending order based on the count of image files having the same extension type. Note: jpeg,jfif,exif,tiff,raw,gif,bmp,png are the various types of image file extensions Business Rules: 1)If all the elements of the input array do not have image type extension, then print -1. 2)If any of the file name doesn't contain extension type or if the extension is not an image type then it will be treated as other type, take the count of all such files and store as the last element in the sorted output array with key element as "others" and value element as the calculated count. 3)If more than one key element have same count, then store the key and their respective value element in the order given in input.  
    
 Create a class named UserProgramCode that has the following static method   
 public static List<string> countImageTypes(string[] input1)  
 Create a class named Program that accepts the inputs and calls the static method present in the UserProgramCode.    
 Input and Output Format:  
 The first line of the input consists of an integer, n that corresponds to the number of elements in the input array .  
 The next 'n' lines of input correspond to elements in the input array.  
 Refer business rules and sample output for formatting specifications. Sample Input 1 :  
 4 Employee.jpeg Purchase.jpeg stock.jpeg book.gif    
 Sample Output 1 : jpeg 3 gif 1 Sample Input 2 :  
 7  
 Sales.doc Employee.jpeg Purchase.jpeg image.png stock.jpeg book.gif pen    
 Sample Output 2 : jpeg 3 png 1 gif 1 others 2

**36)image types**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace image**

**{**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**int a = int.Parse(Console.ReadLine());**

**List<string> image = new List<string>();**

**for (int i = 0; i < a; i++)**

**{**

**image.Add(Console.ReadLine());**

**}**

**List<string> ouyp = userProgramcode.imagescount(image);**

**foreach (string s in ouyp)**

**{**

**Console.WriteLine(s);**

**}**

**Console.ReadLine();**

**}**

**}**

**}**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**namespace image**

**{**

**class userProgramcode**

**{**

**public static List<string> imagescount(List<string> input1)**

**{**

**int k = 0,ctr=0;**

**string[] ar=new string[8]{"jpeg","jfif","exif","tiff","raw","gif","bmp","png"};**

**List<string> outp = new List<string>();**

**string[] st = new string[input1.Count];**

**int[] sco = new int[input1.Count];**

**for (int j = 0; j < input1.Count; j++)**

**{**

**string[] arr = input1[j].Split('.');**

**if (arr.Length==2 && !st.Contains(arr[1]) && ar.Contains(arr[1]) )**

**{**

**st[k] = arr[1];**

**sco[k] = sco[k] + 1;**

**k++;**

**}**

**else if (arr.Length == 2 && st.Contains(arr[1]) && ar.Contains(arr[1]) )**

**{**

**for (int p = 0; p < st.Length; p++)**

**{**

**if (st[p] == arr[1])**

**{**

**sco[p] = sco[p] + 1;**

**break;**

**}**

**}**

**}**

**else**

**{**

**ctr++;**

**}**

**}**

**if (ctr != input1.Count)**

**{**

**int[] co = new int[k];**

**co = sco.ToArray();**

**sco = co.Distinct().ToArray();**

**Array.Sort(sco);**

**Array.Reverse(sco);**

**for (int m = 0; m < sco.Length - 1; m++)**

**{**

**for (int n = 0; n < co.Length; n++)**

**{**

**if (sco[m] == co[n])**

**{**

**outp.Add(st[n]);**

**outp.Add(sco[m].ToString());**

**}**

**}**

**}**

**}**

**if (ctr != 0 && ctr!=input1.Count)**

**{**

**outp.Add("Others");**

**outp.Add(ctr.ToString());**

**}**

**else if (ctr == input1.Count)**

**{**

**outp.Add("-1");**

**return outp;**

**}**

**return outp;**

**}**

**}**

**}**