using System;

## Count the number of Occurrences

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

using System.Text;

namespace level\_2\_practise

{

class Program

{

static void Main(string[] args)

{

int count=0;

string s1 = Console.ReadLine(); string s2 = Console.ReadLine(); string st1 = s1.ToLower(); string st2 = s2.ToLower(); string[] arr1 = st1.Split(' '); string[] arr2 = st2.Split(' ');

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

{

if (arr1[i] == arr2[1]) count++;

}

if(count==0)

Console.WriteLine(0);

else

}

}

}

Console.WriteLine(count);

**Day of Week**

using System;

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

using System.Text;

namespace day\_of\_week\_next\_year

{

class userprogramcode

{

public static string nextyearday(string s)

{

DateTime dt; DateTime dt1;

bool b = DateTime.TryParseExact(s, "dd/MM/yyyy", null, System.Globalization.DateTimeStyles.None, out dt);

if (b)

{

dt1 = dt.AddYears(1);

string ou = dt1.DayOfWeek.ToString(); return ou;

}

else

return "-1";

}

}

}

## Reverse and Format

using System;

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

using System.Text;

namespace reverse\_and\_format

{

class user

{

public static string reverse(string s, char c)

{

char[] ch = s.ToCharArray(); Array.Reverse(ch);

StringBuilder sb = new StringBuilder(); foreach (char item in ch)

{

sb.Append(item); sb.Append(c);

}

string output1 = sb.ToString();

string output2 = output1.Remove(output1.Length - 1); return output2;

}

}

}

##### Finding common Elements in multiples of 3

**Calculate Cost Unique Counter**

(do it)

**Calculate Telephone Bill**

using System;

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

using System.Text;

namespace telephone\_bill

{

class userprogramcode

{

public static double user(int r)

{

double b=0.00;

//double r1 = (double)r;

//Console.WriteLine(r1); if (r <= 300)

{

b = 200;

}

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

{

b=(double)(((r - 300) \* 0.60) + 200);

}

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

{

}

else

b =(double)(((r - 350) \* 0.50 )+ (50\* 0.60) + 200.00);

b = (double)(((r-400)\*0.40)+(50 \*0.50) + (50\*0.60) + 200.00);

//double output= Math.Round(b, 2);

//Console.WriteLine(output.ToString("0.00")); return b;

}

}

}

**Vowels**

### removeTens

**Validating the pan**

##### Validate Password

using System;

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

using System.Text;

using System.Text.RegularExpressions;

namespace validate\_password

{

class userprogramcode

{

public static int user(string st)

{

Regex r = new Regex(@"^[A-Za-z](?=.\*[A-Za-z])(?=.\*[0-9])(?=.\*[#@\_])([a-zA-Z0- 9@#\_]{8,})[A-Za-z0-9]$");

//Regex r = new Regex(@"^((?=.\*[A-Za-z])(?=.\*[0-9])(?=.\*[@#$])([A-Za-z0- 9@#$]{6,20}))$");

//if (Regex.IsMatch(st, @"^((?=.\*[\d])(?=.\*[a-zA-z])(?=.\*[@#$])([a-zA-z0- 9$#@]{6,20}))"))

if (r.IsMatch(st)) return 1;

else

return -1;

//if (Regex.IsMatch(st, @"(^(([a-zA-Z]))(?=.\*[\d])(?=.\*[a-zA-

z])(?=.\*[@#\_])([a-zA-z0-9\_#@]{8,})([A-Za-z0-9])$)"))

// return 1;

//else

// return -1;

}

}

}

## Sort the list

using System;

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

using System.Text;

using System.Text.RegularExpressions;

namespace sort\_list

{

class userprogramcode

{

public static List<string> GetlongestString(List<string> list, char c)

{

// foreach (string item in list)

// {

// item.ToLower();

// }

// List<string> output=new List<string>(1);

// Regex r = new Regex(@"^[a-zA-Z]{1,}$");

// foreach (string item in list)

//{

// if (!r.IsMatch(item))

// {

// output.Add("-2");

// return output;

// System.Environment.Exit(0);

// }

//}

// string s = c.ToString();

// string st = s.ToLower();

// List<string> ou = (from p in list

// where !p.StartsWith(st)

// select p).ToList();

// List.sort(ou);

// if (ou.Count == 0)

// {

// output.Add("-1");

// return output;

// System.Environment.Exit(0);

// }

// else

// return ou;

string g=c.ToString(); List<string> l = new List<string>(); List<string> b = new List<string>();

foreach (var item in list)

{

if(!Regex.IsMatch(item,@"^([a-zA-Z]{1,})$"))

{

return b;

}

}

b.Add("-2");

System.Environment.Exit(0);

var q = from z in list where !z.StartsWith(g) select z;

l = q.ToList(); if (l.Count == 0)

{

return b;

}

b.Add("-1");

System.Environment.Exit(0);

foreach (var item in q)

{

b.Add(item);

}

return b;

}

}

}

**Convert Format**

using System;

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

using System.Text;

using System.Text.RegularExpressions;

namespace convert\_format

{

class userprogramcode

{

public static string format(string st)

{

//string s = Console.ReadLine();

//StringBuilder sb = new StringBuilder();

//Regex r = new Regex(@"(([0-9]{3})+[-]+([0-9]{3})+[-]+([0-9]{4}))");

//if (r.IsMatch(s))

//{

// sb.Append(s.Substring(0, 2) + '-' + s[2] + s[4] + '-' + s.Substring(5, 2) + s[8] + '-' + s.Substring(9));

//}

//else

//{

// Console.WriteLine("noo");

//}

//Console.WriteLine(sb.ToString());

//return sb.ToString();

string op = "";

//Regex r = new Regex(@"^[0-9]{3}[-][0-9]{3}[-][0-9]{4}$");

//if (!r.IsMatch(st))

//{

// op = "-1";

// return op;

// System.Environment.Exit(0);

//}

//else

//{

op = st.Substring(0, 2) + "-" + st.Substring(2, 1) + st.Substring(4, 1) + "-"

+ st.Substring(5, 2) + "-" + st.Substring(8, 1) + "-" + st.Substring(9, 3); return op;

//}

}

}

}

##### Common Characters(try own logic)

using System;

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

using System.Text;

namespace count\_common\_elements

{

class userprogramcode

{

public static int format(string st1,string st2)

{

char[] c1 = st1.ToCharArray(); char[] c2 = st2.ToCharArray(); string str1 = "";

string str2 = "";

StringBuilder sb = new StringBuilder(); StringBuilder sb2 = new StringBuilder(); for (int i = 0; i < c1.Length; i++)

{

for (int j = 0; j < c2.Length; j++)

{

if (c1[i] == c2[j] && c1[i] != ' ')

{

sb.Append(c1[i]);

}

}

}

str1 = sb.ToString(); var v = str1.Distinct(); foreach (var item in v)

{

sb2.Append(item);

}

str2 = sb2.ToString(); return str2.Length;

}

}

}

# Sum of Odd Even Positioned(try owm logic)

## Dash Check

using System;

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

using System.Text;

namespace dash\_count

{

class userprogramcode

{

public static int format(string st1,string st2)

{

int count=0;

string[] arr1 = st1.Split('\_'); string[] arr2 = st2.Split('\_');

for (int i = 0; i < arr1.Length-1; i++)

{

if (arr1[i].Length == arr2[i].Length) count++;

}

if (count == arr1.Length - 1) return 1;

else

}

}

}

return 2;

##### Calculate Bill Amount

using System;

##### Sum Non Prime Numbers

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

using System.Text;

namespace sum\_of\_non\_prime\_numbers

{

class userprogramcode

{

public static int nonprime(int a)

{

int count=0,sum=0;

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

{

count=0;

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

{

if (i % j == 0) count++;

}

if (count != 2) sum = sum + i;

}

return sum;

}

}

}

##### Calculate VAT

using System;

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

using System.Text;

namespace calculate\_VAT

{

class userprogramcode

{

public static double vat(char ch, double cost)

{

double tax=0; if (cost < 0) tax=-1;

else if (ch != 'M' && ch != 'V' && ch != 'C' && ch != 'E') tax =-1;

else if(ch=='M')

tax = 0.09 \* cost; else if(ch=='V')

tax = 0.05 \* cost;

else if(ch=='C')

tax = 0.12 \* cost;

else if(ch=='E')

tax = 0.625 \* cost; return tax;

}

}

}

##### Calculate Take Home Salary

using System;

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

using System.Text;

namespace calculate\_take\_home\_salary

{

class userprogramcode

{

public static int cal(int a)

{

int sal=0; if (a < 0)

sal = -1;

else if (a > 0 && a <= 15000) sal = a - 750 - 678;

else if (a >= 15001 && a <= 22000) sal = a - 850 - 678;

else if (a >=22001 && a <= 30000) sal = a - 925 - 678;

else if (a > 30000)

sal = a - 1000 - 678;

return sal;

}

}

}

##### Odd Even Sum

using System;

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

using System.Text;

namespace oddevensum

{

class userprogramcode

{

public static int oddeven(int a)

{

int[] arr=new int[10];

int b=0,os=0,es=0; while (a != 0)

{

arr[b] = a % 10; b++;

a = a / 10;

}

Array.Resize(ref arr, b); Array.Reverse(arr);

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

{

if (i % 2 == 0)

os = os + arr[i]; else

es = es + arr[i];

}

if (os == es)

return 1;

else

}

}

}

return -1;

##### Extract Max Substring

using System;

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

using System.Text;

namespace extract\_max\_substring

{

class userprogramcode

{

public static string max(string s, string d)

{

//string max = "";

//int m = 0;

//char c = Convert.ToChar(d);

//string[] arr = s.Split(c);

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

//{

// if (arr[i].Length > m)

// {

// m = arr[i].Length;

// max = arr[i];

// }

//}

//return max;

char c = Convert.ToChar(d); List<string> l = new List<string>(); string[] st = s.Split(c);

l = (from z in st

orderby st.Length descending select z).ToList();

foreach(string item in l)

{

l.Add(item);

}

string a = l[0].ToString(); return a;

}

}

}

## Arrange After Cubing(ask doubt try it)

using System;

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

using System.Text;

namespace arrange\_after\_cubing

{

class userprogramcode

{

public static List<int> cube(List<int> a)

{

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

for (int i = 0; i < a.Count-1; i++)

{

l.Add(a[i]);

if (a[i] \* a[i] == a[i + 1])

{

l.Add(a[i]\*a[i]\*a[i]);

}

}

l.Add(a[a.Count - 1]); foreach (int item in l)

{

Console.WriteLine(item);

}

return l;

}

}

}

## Find Leaders

using System;

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

using System.Text; namespace find\_leaders

{

class userprogramcode

{

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

{

int[] output=new int[20]; int k=0,count=0;

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

{

if (a[i] < 0)

{

output[k] = -1; return output;

}

}

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

{

output[k] = -2; return output;

}

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

//{

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

// {

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

// {

// output[k] = a[i];

// k++;

// }

// }

//}

//if (a.Length<output.Length)

//{

// output[0] = -3;

// return output;

//}

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

{

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

{

if (a[i] > a[j]) count++;

}

if (count == i + 1)

{

output[k] = a[i]; k++;

}

}

Array.Resize(ref output, k); Array.Sort(output);

return output;

}

}

}

## Validate String(check pc) Digit Sum in String Array Symmetric Difference(do it)

**Repeated Words(check)**

(do it)

**Count Subsets**

using System;

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

using System.Text;

namespace count\_subsets

{

class userprogramcode

{

public static int subset(int[] a)

{

int count = 0,sum=0;

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

{

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

{

if (a[i] == a[j]) return -3;

sum = a[i] + a[j];

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

{

if (a[k] < 0)

return -2;

else if (sum == a[k]) count++;

}

}

}

if (count == 0) return -1;

return count;

}

}

}

## Identify Perfect Numbers

using System;

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

using System.Text;

namespace perfect\_numbers

{

class userprogramcode

{

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

{

int sum=0,k=0;

int[] output = new int[20];

if (a.Length == 1 || a.Length > 7)

{

output[0] = -3;

Array.Resize(ref output, 1); return output; System.Environment.Exit(0);

}

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

{

if (a[i] < 0)

{

output[0] = -1;

Array.Resize(ref output, 1); return output; System.Environment.Exit(0);

}

for (int b = i+1; b < a.Length; b++)

{

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

{

output[0] = -2;

Array.Resize(ref output, 1); return output;

}

}

}

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

{

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

{

if (a[i] % j == 0) sum = sum + j;

}

if (sum != a[i])

{

output[k] = a[i]; k++;

}

}

Array.Resize(ref output, k); return output;

}

}

}

## Quadratic Equation

using System;

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

using System.Text;

namespace quadratic\_equation

{

class userprogramcode

{

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

{

int y = 0, z = 0, k = 0; int[] output = new int[20];

if (a.Length == 1 || a.Length > 10)

{

output[0] = -3;

Array.Resize(ref output, 1); return output;

}

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

{

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

{

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

{

output[0] = -2;

Array.Resize(ref output, 1); return output;

}

}

}

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

{

if (a[i] < 0)

{

output[0] = -1;

Array.Resize(ref output, 1); return output;

}

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

z = (2 \* y)-(a[i] \* a[i]); output[k] = y;

output[k + 1] = z; k = k + 2;

}

Array.Resize(ref output, k); return output;

}

}

}

## Triplets(only 1 set?)

using System;

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

using System.Text;

namespace triplets

{

class userprogramcode

{

public static int[] triplets(int[] a,int d)

{

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

//int sum = 0,e=0;

//int[] output=new int[20];

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

//{

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

// {

// for (int k = i+2; k < a.Length; k++)

// {

// sum = a[i] + a[j] + a[k];

// if (sum == d)

// {

// output[e] = a[i];

// output[e+1] = a[j];

// output[e+2] = a[k];

// e=e+3;

// }

// }

// }

//}

//Array.Resize(ref output, e);

//return output;

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

{

if (a[i] < 0)

{

op.Add(-1);

return op.ToArray();

}

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

{

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

{

op.Add(-3);

return op.ToArray();

}

}

}

int[] op1 = new int[3];

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

{

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

{

for (int k = j + 1; k < a.Length; k++)

{

if (a[i] + a[j] + a[k] == d)

{

op.Add(a[i]);

op.Add(a[j]);

op.Add(a[k]);

}

}

}

}

if (op.Count == 0)

{

op.Add(-2);

return op.ToArray();

}

return op.ToArray();

}

}

}

## Max Diff in Array(doubt)

#### Password Encryption(do it)

using System;

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

using System.Text;

namespace password\_encryption

{

class userprogramcode

{

public static string pass(string s, char c)

{

string[] arr = s.Split(' '); StringBuilder sb = new StringBuilder(); foreach (string item in arr)

{

char[] ch = item.ToCharArray(); if (ch[0] == c)

{

if (ch[0] == 'Z' || ch[0] == 'z')

{

char x = (char)(ch[0] - 25); sb.Append(x); sb.Append('#');

}

else

{

}

}

else

{

char x1 = (char)(ch[0] + 1); sb.Append(x1); sb.Append('#');

sb.Append(c);

}

sb.Append(item.Substring(1)); sb.Append(' ');

}

return sb.ToString();

}

}

}

## GCD – Array

using System;

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

using System.Text;

namespace gcd\_array

{

class userprogramcode

{

public static int gcd(int[] a)

{

//Array.Sort(a);

//Array.Reverse(a);

//foreach (int item in a)

//{

// Console.WriteLine(item);

//}

//int max=0;

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

//{

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

// {

// if (a[i] % j==0)

// {

// if (j > max)

// max = j;

// }

// }

//}

//return max;

int flag = 0;

List<int> l = new List<int>(); Array.Sort(a);

int b = a[0];

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

{

flag = 0;

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

{

if (a[j] % i != 0)

{

flag = 1;

}

}

if (flag == 1)

{

}

else

{

l.Add(i);

}

}

Console.WriteLine(l[l.Count-1]); return l[l.Count-1];

}

}

}

## Sort String(do it)

using System;

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

using System.Text;

namespace sort\_string

{

class userprogramcode

{

public static string[] sort(string[] a)

{

List<string> li = new List<string>(); StringBuilder sb = new StringBuilder(); foreach (string item in a)

{

string s = item.ToLower(); char[] c = s.ToCharArray();

char[] c1= (from r in c

orderby r

select r).Distinct().ToArray();

string str = new string(c1); li.Add(str);

}

li.Sort();

return li.ToArray();

}

}

}

## Duplicate Date Elements

**Train tariff:’**

using System;

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

using System.Text;

namespace train\_tariff\_calculation

{

class userprogramcode

{

public static int train(string b, string j, string c)

{

int cost = 0; DateTime dt; DateTime dt1;

bool x = DateTime.TryParseExact(b, "yyyy.MM.dd", null, System.Globalization.DateTimeStyles.None, out dt);

bool y = DateTime.TryParseExact(j, "yyyy.MM.dd", null, System.Globalization.DateTimeStyles.None, out dt1);

if (!x && !y)

{

return -1;

}

int z = (dt - dt1).Days; Console.WriteLine(z);

if (z < 3)

return -2; else if (z > 90)

return -3;

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

return -4;

if (z > 30 && z <= 90)

{

if (c == "SL")

cost = 1000; else if (c == "1AC")

cost = 2500; else if (c == "2AC")

cost = 2000; else if (c == "3AC")

cost = 1500;

}

else if (z >=21 && z <= 30)

{

if (c == "SL")

cost = (int)(1.10\*1000); else if (c == "1AC")

cost = (int)(1.10\*2500); else if (c == "2AC")

cost = (int)(1.10\*2000); else if (c == "3AC")

cost = (int)(1.10\*1500);

}

else if (z >= 11 && z <= 20)

{

if (c == "SL")

cost = (int)(1.20 \* 1000); else if (c == "1AC")

cost = (int)(1.20 \* 2500); else if (c == "2AC")

cost = (int)(1.20 \* 2000); else if (c == "3AC")

cost = (int)(1.20 \* 1500);

}

else if (z >= 4 && z <= 10)

{

if (c == "SL")

cost = (int)(1.30 \* 1000); else if (c == "1AC")

cost = (int)(1.30 \* 2500); else if (c == "2AC")

cost = (int)(1.30 \* 2000);

else if (c == "3AC")

cost = (int)(1.30 \* 1500);

}

else if (z ==3)

{

if (c == "SL")

cost = (int)(1.40 \* 1000); else if (c == "1AC")

cost = (int)(1.40 \* 2500); else if (c == "2AC")

cost = (int)(1.40 \* 2000); else if (c == "3AC")

cost = (int)(1.40 \* 1500);

}

return cost;

}

}

}

## Calculate Grade

using System;

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

using System.Text;

namespace calculate\_grade

{

class userprogramcode

{

public static string[] grade(int[] a)

{

int max=0,rollno=0;

List<string> li = new List<string>(); string[] ou = new string[20];

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

{

if (a[i] < 0)

{

ou[0] = "-1";

Array.Resize(ref ou, 1); return ou;

}

}

if (a.Length < 2)

{

ou[0] = "-2";

Array.Resize(ref ou, 1); return ou;

}

if (a.Length % 2!=0)

{

ou[0] = "-3";

Array.Resize(ref ou, 1); return ou;

}

for (int i = 1; i < a.Length; i=i+2)

{

if (a[i] > max)

{

max = a[i];

rollno = a[i - 1];

}

}

if (max >= 80)

{

li.Add(rollno.ToString()); li.Add("DISTINCTION");

}

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

{

li.Add(rollno.ToString()); li.Add("FIRST CLASS");

}

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

{

li.Add(rollno.ToString()); li.Add("SECOND CLASS");

}

else if (max <=0)

{

li.Add(rollno.ToString()); li.Add("FAIL");

}

return (li.ToArray());

}

}

}

## Array Median(check)(use )

**Student Score**

using System;

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

using System.Text;

namespace student\_score

{

class userprogramcode

{

public static string[] score(string[] a,string b)

{

string[] ou=new string[20]; List<string> li=new List<string>(); string max="";

string name = "";

if (a.Length % 2 != 0)

{

ou[0] = "-1";

Array.Resize(ref ou, 1); return ou;

}

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

{

char[] c = a[i].ToCharArray(); foreach (char item in c)

{

if (!char.IsLetterOrDigit(item))

{

ou[0] = "-2";

Array.Resize(ref ou, 1); return ou;

}

}

}

for (int i = 0; i < a.Length; i=i+2)

{

if (a[i] ==b )

{

name = a[i]; max = a[i +1];

}

}

int m = int.Parse(max); if (m >= 80)

{

li.Add(name); li.Add(m.ToString()); li.Add("OUTSTANDING");

}

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

{

li.Add(name); li.Add(m.ToString());

li.Add("GOOD");

}

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

{

li.Add(name); li.Add(m.ToString());

li.Add("AVERAGE");

}

if (m <50)

{

li.Add(name); li.Add(m.ToString());

li.Add("FAIL");

}

return li.ToArray();

}

}

}

## Relative Order

using System;

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

using System.Text;

using System.Text.RegularExpressions;

namespace l2ndl3

{

class Usercode

{

public static int[] counter(int[] a, int[] a1)

{

int flag = 0;

List<int> li = new List<int>(); List<int> li1 = new List<int>(); List<int> li2 = new List<int>(); for (int i = 0; i < a1.Length; i++)

{

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

{

if (a[j]==a1[i])

{

li.Add(a[j]);

}

}

}

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

{

flag = 0;

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

{

if (a1[l]==a[k])

{

flag = 1;

}

}

if (flag==1)

{

}

else

{

li1.Add(a[k]); li1.Sort();

}

}

foreach (var item in li1)

{

li.Add(item);

}

int[] op = li.ToArray(); return op;

}

}

}

## Berth Type

using System;

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

using System.Text;

namespace berth\_type

{

class userprogramcode

{

public static string berth(string f, string gf, string r)

{

string ou = "";

List<string> s = new List<string>(); s.Add(f);

s.Add(gf);

s.Add(r);

foreach (string item in s)

{

char[] c = item.ToCharArray(); foreach (char i in c)

{

if (!char.IsLetterOrDigit(i)) return "Invalid Input";

}

}

int fa = int.Parse(f); int gfa = int.Parse(gf); int ra = int.Parse(r);

List<int> li = new List<int>(); List<string> seat = new List<string>(); li.Add(fa);

li.Add(gfa);

li.Add(ra);

foreach(int item in li)

{

if (item <= 0 || item > 1000) return "Invalid seat number";

}

foreach (int item in li)

{

if (item % 8 == 1 || item % 8 == 4) seat.Add("L");

if (item % 8 == 2 || item % 8 == 5) seat.Add("M");

if (item % 8 == 3 || item % 8 == 6) seat.Add("U");

if (item % 8 == 7) seat.Add("SL");

if (item % 8 == 0) seat.Add("SU");

}

request";

if (seat[1] == "L")

ou="Lower berth provided as per request"; if (seat[2] == "L")

ou= "Your seat has been swapped from "+gfa+ " to "+ra+" as per preference

if (seat[0] == "L")

ou = "Your seat has been swapped from " + gfa + " to " + fa + " as per

preference request";

else

ou = "Your seat no will be changed on the date of travel"; return ou;

}

}

}

**Unique Even Sum**

using System;

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

using System.Text;

using System.Text.RegularExpressions;

namespace l2ndl3

{

class Usercode

{

public static int counter(int[] a)

{

int flag = 0;

List<int> li = new List<int>(); List<int> li1 = new List<int>(a);

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

{

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

{

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

{

li.Add(a[i]);

}

}

}

li1 = li1.Distinct().ToList();

int[] z = li1.Except(li).ToArray(); fo {

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

{

flag = flag + z[i];

}

}

return flag;

}

}

}

## Largest Span

using System;

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

using System.Text;

namespace unique\_even\_sum

{

class Program

{

static void Main(string[] args)

{

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

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

{

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

}

int ou = userprogramcode.unique(arr);

Console.WriteLine(ou);

}

}

}

using System;

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

using System.Text;

namespace unique\_even\_sum

{

class userprogramcode

{

public static int unique(int[] ar)

{

List<int> li = new List<int>(); int a = 0, b = 0,c=0;

foreach (int item in ar)

{

a = Array.IndexOf(ar, item);

b = Array.LastIndexOf(ar, item);

c = (b - a) + 1;

li.Add(c);

}

if (li.Count == 0) return 0;

else

return li[0];

}

}

}

#### Reimbursement:

using System;

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

using System.Text;

namespace reimbursement

{

class userprogramcode

{

public static int reim(double f, int p, bool b)

{

int refund=0; if (b)

return -4; if (!b)

{

if (f < 25000)

return -1; if (p < 80)

return -2;

if (p >= 80 && p <= 85)

refund = (int)((.40 \* f) + 3000); if (p >= 86 && p <= 90)

refund = (int)((.50 \* f) + 5000); if (p > 90)

refund = (int)((.60 \* f) + 7000);

}

return refund;

}

}

}

using System;

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

using System.Text;

namespace insurance\_guide

{

## Insurance Guide

class userprogramcode

{

public static int[] ins(char h, int age, char gender, char loc)

{

int[] ou=new int[2]; if (age > 60)

{

ou[0] = -1;

Array.Resize(ref ou, 1); return ou;

}

if (h == 'E' && age >= 25 && age <= 35 && gender == 'M' && loc == 'C')

{

ou[0] = 4;

ou[1] = 200000;

}

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

{

ou[0] = 3;

ou[1] = 100000;

}

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

{

}

else

{

}

ou[0] = 6;

ou[1] = 10000;

ou[0] = -2;

Array.Resize(ref ou, 1); return ou;

return ou;

}

}

}

## Display Students Exam Eligibility status

using System;

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

using System.Text;

namespace exam\_eligibility\_status

{

class userprogramcode

{

public static string eli(int a, int b)

{

if (a >100 || b > 100)

{

return "Invalid Input";

}

if (a >= 55 && b >= 45) return "P";

if (a >= 45 && a<55 && b >= 45) return "P";

if (a >= 65 && b < 45) return "R";

else

return "F";

}

}

}

## Calculate Charge

using System;

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

using System.Text;

namespace calculate\_charge

{

class userprogramcode

{

public static int charge(string s1, string s2)

{

DateTime dt;

DateTime dt1;int res=0;

bool a = DateTime.TryParseExact(s1, "yyyy-MM-dd:HH:mm:ss", null, System.Globalization.DateTimeStyles.None, out dt);

bool b = DateTime.TryParseExact(s2, "yyyy-MM-dd:HH:mm:ss", null, System.Globalization.DateTimeStyles.None, out dt1);

if(!a)

return -1; if (!b)

return -1;

int time = (int)dt1.Subtract(dt).TotalHours; if (time < 0)

return -2; if (time > 24) return -3;

if (time <= 3) return 20;

if (time == 24) return 100;

if (time > 3 && time < 24)

res=20+(time\*5); return res;

}

}

}

# Get word with Maximum Vowels

using System;

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

using System.Text;

namespace password\_encryption

{

class userprogramcode

{

public static string pass(string s)

{

int count = 0, max = 0;

string r = "";

string[] st = s.Split(' '); foreach (var item1 in st)

{

count = 0;

char[] ch = item1.ToCharArray(); foreach (var item in ch)

{

== 'u' ||

== 'U')

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

{

count++;

}

}

if (count > max)

{

max = count; r = item1;

}

}

return r;

}

}

}

## Check Palindrome

using System;

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

using System.Text;

namespace check\_palindrome

{

class userprogramcode

{

public static int palindrome(string s)

{

char[] a = s.ToCharArray(); StringBuilder sb = new StringBuilder(); int count = 0;

List<char> li = new List<char>(); foreach (char i in a)

{

if (i == 'a' || i == 'e' || i == 'i' || i == 'o' || i == 'u' || i == 'A'

|| i == 'E' || i == 'I' || i == 'O' || i == 'U')

{

if (li.Contains(i))

{

}

else

{

li.Add(i); count++;

}

}

}

if (count >= 2)

{

Array.Reverse(a); foreach (char item in a)

{

sb.Append(item);

}

}

string res = sb.ToString(); if (res == s)

return 1; else

return -1;

}

}

}

## Reverse the adjacent pairs of letters

using System;

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

using System.Text;

namespace reverse\_the\_adjacent\_pair\_of\_letters

{

class userprogramcode

{

public static string reverse(string s)

{

int l = s.Length;

char[] c = s.ToCharArray(); StringBuilder sb = new StringBuilder(); List<char> li = new List<char>();

if (l % 2 == 0)

{

for (int i = 0; i < l; i=i+2)

{

sb.Append(c[i + 1]);

sb.Append(c[i]);

}

}

else if (l % 2 != 0)

{

for (int i = 0; i < l-1; i=i+2)

{

sb.Append(c[i + 1]);

sb.Append(c[i]);

}

sb.Append(c[l - 1]);

}

return sb.ToString();

}

}

}

**Duplicate Characters**

using System;

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

using System.Text;

namespace duplicate\_characters

{

class userprogramcode

{

public static string duplicate(string s)

{

char[] c = s.ToCharArray(); List<char> li = new List<char>(); string sa = "";

StringBuilder sb = new StringBuilder(); foreach (char item in c)

{

if (li.Contains(item))

{

}

else

{

li.Add(item);

}

}

foreach (char item in li)

{

sb.Append(item);

}

sa = sb.ToString(); return sa;

}

}

}

##### All Vowels

using System;

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

using System.Text;

namespace all\_vowels

{

class userprogramcode

{

public static int vowels(string s)

{

string a = "aeiou";

char[] c = s.ToCharArray(); StringBuilder sb=new StringBuilder(); foreach (char item in c)

{

'u')

if (item == 'a' || item == 'e' || item == 'i' || item == 'o' || item ==

sb.Append(item);

}

string b=sb.ToString(); if (a == b)

return 1;

else

}

}

}

return -1;

##### String Occurences

using System;

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

using System.Text;

namespace String\_Occurences

{

class userprogramcode

{

public static int occurence(string s1, string s2)

{

int count = 0;

string[] arr1 = s1.Split(' ');

string[] arr2 = s2.Split(' '); foreach (string item in arr1)

{

if (item == arr2[1]) count++;

}

return count;

}

}

}

##### Check Anagrams

using System;

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

using System.Text;

namespace check\_anagrams

{

class userprogramcode

{

public static bool anagram(string s1, string s2)

{

int count=0;

string st1 = s1.ToLower(); string st2 = s2.ToLower(); char[] c1 = st1.ToCharArray(); char[] c2 = st2.ToCharArray();

List<char> li = new List<char>(); List<char> li2 = new List<char>(); foreach (char item in c1)

{

if (!char.IsWhiteSpace(item)) li.Add(item);

}

foreach (char item in c2)

{

if (!char.IsWhiteSpace(item)) li2.Add(item);

}

li.Sort();

li2.Sort();

if (li.Count != li2.Count) return false;

else

{

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

{

if (li[i] != li2[i]) count = 1;

}

if (count == 1) return false;

else

return true;

}

}

}

}

##### Repeat Characters

using System;

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

using System.Text;

namespace repeat\_characters

{

class userprogramcode

{

public static string repeat(string s,int n)

{

if (n < 0)

return "-1"; if (n > 10)

return "-2"; if (s.Length < 2)

return "-3";

char[] arr = s.ToCharArray(); StringBuilder sb=new StringBuilder(); StringBuilder sb1 = new StringBuilder(); if(arr.Length%2!=0)

{

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

{

if (i % 2 == 0) sb.Append(arr[i]);

}

}

if (arr.Length % 2 == 0)

{

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

{

if (i % 2 != 0) sb.Append(arr[i]);

}

}

string re = sb.ToString(); for (int i = 0; i <n ; i++)

{

sb1.Append(re);

}

return sb1.ToString();

}

}

}

##### Check Batch Code

using System;

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

using System.Text;

namespace check\_batch\_code

{

class userprogramcode

{

public static string batchcode(string s)

{

char[] c=s.ToCharArray(); string exp = "";

//string dm = "";

string loc = s.Substring(0, 3); string year = s.Substring(3, 2); string domain = s.Substring(5, 2); string batchno = s.Substring(7, 3);

//if(domain=="DN")

// dm="DotNet";

if (loc != "CHN" && loc != "CBE" && loc != "KOC" && loc != "PUN" && loc != "BGL" && loc != "HYD" && loc != "KOL")

return "-1";

if( (!char.IsDigit(c[3])) && (!char.IsDigit(c[4])) && (!char.IsDigit(c[7])) && (!char.IsDigit(c[8])) && (!char.IsDigit(c[9])) )

return "-2";

if (domain != "DN")

return "-3"; if (loc == "CHN")

exp = "Chennai"; else if (loc == "CBE")

exp = "Coimbatore"; else if (loc == "KOC")

exp = "Kochi"; else if (loc == "PUN")

exp = "pune";

else if (loc == "BGL") exp = "Bangalore"; else if (loc == "HYD") exp = "Hyderabad"; else if (loc == "KOL")

exp = "Kolkata";

return "DotNet batch "+batchno+" has joined in "+"20"+year+ " year and is at "+exp+" location";

}

}

}

## Image Types (refer mail)

**Calculate New Salary**

using System;

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

using System.Text;

namespace calculate\_new\_salary

{

class userprogramcode

{

public static int newsalary(int y, string d, int s)

{

int salary=0;

if (y < 0 || y > 25) return -1;

if (d != "RS" && d != "CS") return -2;

if (s < 0 || s > 100000) return -3;

if (d == "CS")

{

if (y > 0 && y >= 3)

salary = (int)(1.3 \* s); else if (y > 3 && y >= 5)

salary = (int)(1.35 \* s); else if (y > 5 && y >= 8)

salary = (int)(1.40 \* s); else if (y > 8)

salary = (int)(1.45 \* s);

}

else if (d == "RS")

{

if (y > 0 && y >= 3)

salary = (int)(1.35 \* s); else if (y > 3 && y >= 5)

salary = (int)(1.40 \* s); else if (y > 5 && y >= 8)

salary = (int)(1.45 \* s); else if (y > 8)

salary = (int)(1.50 \* s);

}

return salary;

}

}

}

### Calculate Discount

using System;

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

using System.Text;

namespace calculate\_discount

{

class userprogramcode

{

public static int[] discount(int[] a, int[] b)

{

int dis=0;

List<int> li = new List<int>(); for (int i = 0; i < a.Length; i++)

{

if (b[i + 1] >= 500000)

{

if (a[i + 1] >= 1 && a[i + 1] <= 4)

{

dis = (int)(b[i] \* 0.25);

li.Add(a[i]); li.Add(dis);

}

if (a[i + 1] >= 5 && a[i + 1] <= 8)

{

dis = (int)(b[i] \* 0.20);

li.Add(a[i]); li.Add(dis);

}

if (a[i + 1] >= 9 && a[i + 1] <= 12)

{

dis = (int)(b[i] \* 0.15);

li.Add(a[i]); li.Add(dis);

}

}

else if (b[i + 1] >= 1000000 && b[i]<=5000000)

{

if (a[i + 1] >= 1 && a[i + 1] <= 4)

{

dis = (int)(b[i] \* 0.15);

li.Add(a[i]); li.Add(dis);

}

if (a[i + 1] >= 5 && a[i + 1] <= 8)

{

dis = (int)(b[i] \* 0.10);

li.Add(a[i]); li.Add(dis);

}

if (a[i + 1] >= 9 && a[i + 1] <= 12)

{

dis = (int)(b[i] \* 0.05);

li.Add(a[i]); li.Add(dis);

}

}

}

foreach (int item in li)

{

Console.WriteLine(item);

}

return li.ToArray();

}

}

}

## EMI Calculation

using System;

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

using System.Text;

namespace EMI

{

class Userprogramcode

{

public static int inst(string dob, int month)

{

if ((month != 12) && (month != 24) && (month != 36) && (month != 48))

{

return -2;

}

DateTime dt1; int inst = 0; double amount1;

bool i = DateTime.TryParseExact(dob, "dd-MM-yyyy", null, System.Globalization.DateTimeStyles.None, out dt1);

if (i)

{

int year = DateTime.Now.Year - dt1.Year;

//int mont = DateTime.Now.Month - dt1.Month;

//DateTime dt3 = DateTime.Now.Date;

//DateTime dt4 = dt3.AddMonths(month);

//if (mont < 0)

//{

// year = year - 1;

// mont = mont + 12;

//}

if (year <= 22)

{

amount1 = (double)(200000 \* 1.03); inst = (int)amount1 / month;

}

if (year > 22 && year <= 45)

{

amount1 = (double)(300000 \* 1.05); inst = (int)amount1 / month;

}

if (year > 45 && year <= 100)

{

amount1 = (double)(500000 \* 1.07); inst = (int)amount1 / month;

}

}

else

}

return inst; return -1;

}

}

## Permutations(refer mail)