**1.Remove 10's**

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

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

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

Input and Output Format:

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

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

Refer sample output for formatting specifications.

Sample Input :

5

1

10

20

10

2

Sample Output :

1

20

2

o

o

class pgm1

{

static int[] removeTens(int[] a)

{

ArrayList al = new ArrayList();

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

{

if (a[i] != 10)

{

al.Add(a[i]);

}

}

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

al.Add(0);

int[] b = (int[])al.ToArray(typeof(int)); return b;

}

static void Main(string[] args)

{

Console.WriteLine("enter the array range");

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[] c=removeTens(a);

foreach (var v in c)

Console.WriteLine(v);

Console.Read();

}

}

**2.Programming Logic**

Write a Program that accepts three integer values (a,b,c) and returns their sum. However, if one of the values is 13 then it does not count towards the sum and the next number also does not count. So for example, if b is 13, then both b and c do not count.

Include a class UserMainCode with a static method getLuckySum which accepts three integers. The return type is integer representing the sum.

Create a Class Main which would be used to accept the input integers and call the static method present in UserMainCode.

Input and Output Format:

Input consists of three integers.

Output consists of a single integer.

Refer sample output for formatting specifications.

Sample Input 1:

1

2

3

Sample Output 1:

6

Sample Input 2:

1

2

13

Sample Output 2:

3

Sample Input 3:

13

3

8

Sample Output 3:

8

class pgm2

{

static int getLuckySum(int n,int n1,int n2)

{

int sum = 0;

if (n2 == 13)

{

sum = n + n1;

}

else if (n == 13)

{

sum = n2;

}

else if (n1 == 13)

{

sum = n;

}

else

sum = n + n1 + n2;

return sum;

}

public static void Main(string[] args)

{

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

int n1 = Convert.ToInt32(Console.ReadLine());

int n2 = Convert.ToInt32(Console.ReadLine());

int a=getLuckySum(n, n1, n2);

Console.WriteLine(a);

Console.Read();

}

}

**3.Simple String Manipulation**

Write a program to read a string and return a modified string based on the following rules.

Return the String without the first 2 chars except when

1. keep the first char if it is 'j'

2. keep the second char if it is 'b'.

Include a class UserMainCode with a static method getString which accepts a string. The return type (string) should be the modified string based on the above rules. Consider all letters in the input to be small case.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string with maximum size of 100 characters.

Output consists of a string.

Refer sample output for formatting specifications.

Sample Input 1:

hello

Sample Output 1:

llo

Sample Input 2:

java

Sample Output 2:

jva

class Pgm3

{

static string getstring(string s)

{

s = s.ToLower();

string s1 = "";

StringBuilder sb = new StringBuilder(s);

if (sb[0] == 'j' && sb[1] == 'b')

{

s1 = sb.ToString();

}

if (sb[0] == 'j' && sb[1] != 'b')

{

sb.Remove(1, 1);

s1 = sb.ToString();

}

if (sb[0] != 'j' && sb[1] == 'b')

{

sb.Remove(0, 0);

s1 = sb.ToString();

}

if (sb[0] != 'j' && sb[1] != 'b')

{

sb.Remove(0, 2);

s1 = sb.ToString();

}

return s1;

}

public static void Main()

{

string s = Console.ReadLine();

string a=getstring(s);

Console.WriteLine(a);

Console.Read();

}

}

**4. Color Code**

Write a program to read a string and validate whether the given string is a valid color code based on the following rules:

- Must start with "#" symbol

- Must contain six characters after #

- It may contain alphabets from A-F or digits from 0-9

Include a class UserMainCode with a static method validateColorCode which accepts a string. The return type (integer) should return 1 if the color is as per the rules else return -1.

Create a Class Main which would be used to accept a String and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of a string (Valid or Invalid).

Refer sample output for formatting specifications.

Sample Input 1:

#FF9922

Sample Output 1:

Valid

Sample Input 2:

#FF9(22

Sample Output 2:

Invalid

static int validcolorcode(string s)

{

Regex r = new Regex("#{1}([0-9A-Fa-f]{6})", RegexOptions.IgnoreCase);

int n = 0;

Match m = r.Match(s);

if (m.Success)

n = 1;

else

n = -1;

return n;

}

public static void Main()

{

string s = Console.ReadLine();

int n1=validcolorcode(s);

Console.WriteLine(n1);

Console.Read();

}

}

**5.Digits - II**

Write a program to read a non-negative integer n, compute the sum of its digits. If sum is greater than 9 repeat the process and calculate the sum once again until the final sum comes to single digit.Return the single digit.

Include a class UserMainCode with a static method getDigitSum which accepts the integer value. The return type is integer.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

9999

Sample Output 1:

9

Sample Input 2:

698

Sample Output 2:

5

using System;

using System.Collections;

namespace Remove10s

{

class Program

{

static void Main(string[] args)

{

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

int sum = UserMainCode.getDigitSum(n);

Console.WriteLine(sum);

Console.Read();

}

}

class UserMainCode

{

public static int getDigitSum(int n)

{

int sum = 0,output=0,temp=0,temp1=0;

while (n>0)

{

temp = n % 10;

sum = sum + temp;

n = n / 10;

}

while (sum>0)

{

output = sum % 10;

temp1 = output + temp1;

sum = sum / 10;

}

return temp1;

}

}

}

**6. Add and Reverse**

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

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

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

Example:

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

Number = 15

sum = 20 + 25 + 30 + 100 = 175

output = 571

Input and Output Format:

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

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

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

Output consists of a single integer.

Sample Input

6

10

15

20

25

30

100

15

Sample Output

571

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace AddandReverse

{

class Program

{

static void Main(string[] args)

{

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

int[] input = new int[size];

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

{

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

}

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

int output = UserMainCode.addAndReverse(input, number);

Console.WriteLine(output);

}

}

class UserMainCode

{

public static int addAndReverse(int[] input,int number)

{

int res = 0, output = 0,j=1;

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

{

if (input[i]>number)

{

res = res + input[i];

}

}

while (res > 0)

{

int temp = res % 10;

output = (output\*10) + temp;

res = res / 10;

}

return output;

}

}

}

}

**7.String Processing - VII**

Write a program to read a two strings and one int value(N). check if Nth character of first String from start and Nth character of second String from end are same or not. If both are same return true else return false.

Check need not be Case sensitive

Include a class UserMainCode with a static method isEqual which accepts the two strings and a integer n. The return type is the TRUE / FALSE.

Create a Class Main which would be used to read the strings and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings and an integer.

Output consists of TRUE / FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

AAAA

abab

2

Sample Output 1:

TRUE

Sample Input 2:

MNOP

QRST

3

Sample Output 2:

FALSE

class Pgm7

{

static bool isEqual(string s1,string s2,int n)

{

s1 = s1.ToLower();

s2 = s2.ToLower();

char[] ch1 = s2.ToCharArray();

Array.Reverse(ch1);

string s3 = new string(ch1);

bool a = false;

if (s1[n - 1] == s3[n - 1])

{

a = true;

}

return a;

}

public static void Main()

{

string s1 = Console.ReadLine();

string s2 = Console.ReadLine();

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

bool b=isEqual(s1, s2, n);

Console.WriteLine(b);

Console.Read();

}

}

**8.Month : Number of Days**

class Pgm8

{

static int getNumberOfDays(int year, int month)

{

int n = 0;

if (year % 4 == 0 || year % 400 == 0)

{

if (month == 0 || month == 2 || month == 4 || month == 6 || month == 7 || month == 9 || month == 11)

n = 31;

else if (month == 3 || month == 5 || month == 8 || month == 10)

n = 30;

else

n = 29;

}

else

{

if (month == 0 || month == 2 || month == 4 || month == 6 || month == 7 || month == 9 || month == 11)

n = 31;

else if (month == 3 || month == 5 || month == 8 || month == 10)

n = 30;

else

n = 28;

}

return n;

}

static void Main()

{

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

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

int a = getNumberOfDays(year, month);

Console.WriteLine(a);

Console.ReadLine();

}

}

**9.SumOdd**

Given two inputs year and month (Month is coded as: Jan=0, Feb=1 ,Mar=2 ...), write a program to find out total number of days in the given month for the given year.

Include a class UserMainCode with a static method “getNumberOfDays” that accepts 2 integers as arguments and returns an integer. The first argument corresponds to the year and the second argument corresponds to the month code. The method returns an integer corresponding to the number of days in the month.

Create a class Main which would get 2 integers as input and call the static method getNumberOfDays present in the UserMainCode.

Input and Output Format:

Input consists of 2 integers that correspond to the year and month code.

Output consists of an integer that correspond to the number of days in the month in the given year.

Sample Input:

2000

1

Sample Output:

29

class pgm9

{

static int addOddNumbers(int n)

{

int sum = 0;

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

{

if(i%2!=0)

{

sum = sum + i;

}

}

return sum;

}

public static void Main()

{

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

int sum=addOddNumbers(n);

Console.WriteLine(sum);

Console.Read();

}

}

**10.Even Sum & Duplicate Elements**

Write a program to read a integer array, Remove the duplicate elements and display sum of even numbers in the output. If input array contain only odd number then return -1.

Include a class UserMainCode with a static method sumElements which accepts the integer array. The return type is integer.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

7

2

3

54

1

6

7

7

Sample Output 1:

62

Sample Input 2:

6

3

7

9

13

17

21

Sample Output 2:

-1

class Pgm10

{

static int sumElements(int[] a)

{

List<int> l1 = a.ToList();

List<int> l2=l1.Distinct().ToList();

int[] b = l2.ToArray();

int sum = 0,flag=0;

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

{

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

{

sum += b[i];

flag ++;

}

}

if(flag==0)

{

sum = -1;

}

return sum;

}

public static void Main()

{

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

int[] n1 = new int[n];

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

{

n1[i] = Convert.ToInt32(Console.ReadLine());

}

int sum=sumElements(n1);

Console.WriteLine(sum);

Console.Read() }}

**Question 11:**

11.   **ArrayList to String Array**

Write a program that performs the following actions:  
    Read n strings as input.  
    Create an arraylist to store the above n strings in this arraylist.  
    Write a function convertToStringArray which accepts the arraylist as input.  
    The function should sort the elements (strings) present in the arraylist and convert them into a string array.  
    Return the array.  
Include a class UserMainCode with the static method **convertToStringArray** which accepts an arraylist and returns an array.  
  
Create a Class Main which would be used to read n strings and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of n+1 integers. The first integer denotes the size of the arraylist, the next n strings are values to the arraylist.  
Output consists of an arrayas per step 4.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
4  
a  
d  
c  
b  
**Sample Output 1:**  
a  
b  
c  
d

using System;

using System.Collections;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_11

{

static void ConvertToStringArray(ArrayList al,int n)

{

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

al.Add(Console.ReadLine());

al.Sort();

string[] str = new string[n];

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

{

str[i] = (string)al[i];

}

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

Console.WriteLine(str[i]);

}

static void Main(string[] args)

{

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

ArrayList al = new ArrayList();

ConvertToStringArray(al, n);

Console.Read();

}

}

}

**Question 12:**

Write a program to read a string from the user and remove all the alphabets and spaces from the String, and**only store special characters and digit** in the output String. Print the output string.

Include a class **UserMainCode** with a static method **getSpecialChar**which accepts a string. The return type (String) should return the character removed string.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of a strings.

Output consists of an String (character removed string).

Refer sample output for formatting specifications.

**Sample Input :**

cogniz$#45Ant

**Sample Output :**

$#45

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_12

{

static void getSpecialChar(string s)

{

string s1 = "";

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

{

if (!(s[i] >= 'a' && s[i] <= 'z' || s[i] >= 'A' && s[i] <= 'Z'))

{

s1 = s1 + s[i];

}

}

Console.WriteLine(s1);

}

static void Main()

{

Console.WriteLine("Enter the string: ");

string s = Console.ReadLine();

getSpecialChar(s);

Console.Read();

}

}

}

**Question 13:**

Write a Program that accepts four int inputs(x1,y1,x2,y2) as the coordinates of two points. Calculate the distance between the two points using the below formula.  
Formula : square root of((x1-x2)\*(x1-x2)+(y1-y2)\*(y1-y2))  
Then, Round the result to return an int  
Include a class UserMainCode with a static method **findDistance** which accepts four integers. The return type is integer representing the formula.  
Create a Class Main which would be used to accept the input integers and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of four integers.  
Output consists of a single integer.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
3  
4  
5  
2  
**Sample Output 1:**  
3  
  
**Sample Input 2:**  
3  
1  
5  
2  
**Sample Output 2:**  
2

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_13

{

static void findDistance(int x1,int y1,int x2,int y2)

{

double d = Math.Sqrt((Math.Pow(x2 - x1, 2) + Math.Pow(y2 - y1, 2)));

Console.WriteLine(Math.Round(d));

}

static void Main()

{

int x1 =Convert.ToInt32(Console.ReadLine());

int y1 = Convert.ToInt32(Console.ReadLine());

int x2 = Convert.ToInt32(Console.ReadLine());

int y2 = Convert.ToInt32(Console.ReadLine());

findDistance(x1, y1, x2, y2);

Console.Read();

}

}

}

**Question 14:**

14.   **Find common characters and unique characters in string**

Given a method with two strings as input. Write code to count the common and unique letters in the two strings.

Note:

- Space should not be counted as a letter.

- Consider letters to be case sensitive. ie, "a" is not equal to "A".

Include a class **UserMainCode** with a static method **commonChars** which accepts two strings as input.

The return type of the output is the count of all common and unique characters in the two strings.

Create a class **Main** which would get the inputs and call the static method **commonChars** present in the UserMainCode.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_14

{

static void commonChars(string s1,string s2)

{

int count = 0;

char[] ch = string.Join(" ", s1.Split(' ')).ToString().ToCharArray();

char[] ch1 = string.Join(" ", s2.Split(' ')).ToString().ToCharArray();

Dictionary<char, int> d1 = new Dictionary<char, int>();

Dictionary<char, int> d2 = new Dictionary<char, int>();

StringBuilder sb1 = new StringBuilder();

StringBuilder sb2 = new StringBuilder();

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

{

if (d1.ContainsKey(ch[i]))

d1[ch[i]]++;

else

d1.Add(ch[i], 1);

}

foreach (var v in d1)

if (v.Value == 1)

sb1.Append(v.Key);

for (int i = 0; i < ch1.Length; i++)

{

if (d2.ContainsKey(ch1[i]))

d2[ch1[i]]++;

else

d2.Add(ch1[i], 1);

}

foreach (var v in d2)

if (v.Value == 1)

sb2.Append(v.Key);

for (int i = 0; i < sb1.Length; i++)

{

for (int j = 0; j < sb2.Length; j++)

{

if (sb1[i] == sb2[j])

count++;

}

}

Console.WriteLine(count);

}

static void Main()

{

string s1 = Console.ReadLine();

string s2 = Console.ReadLine();

commonChars(s1, s2);

Console.Read();

}

}

}

**Question 15:**

15.   **Concatenate Characters**

Given an array of Strings, write a program to take the last character of each string and make a new String by concatenating it.

Include a class **UserMainCode** with a static method **“concatCharacter”**that accepts a String array as input and returns the new String.

Create a class **Main** which would get the String array as input and call the static method **concatCharacter**present in the UserMainCode.

**Input and Output Format:**

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

The next n lines of the input consist of the strings in the input string array.

Output consists of a string.

**Sample Input:**

3

ab

a

abcd

**Sample Output:**

bad

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp7

{

class Program

{

static void Main(string[] args)

{

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

string[] str = new string[size];

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

{

str[i] = Console.ReadLine();

}

string output= UserMainCode.concatCharacter(str);

Console.WriteLine(output);

Console.Read();

}

}

class UserMainCode

{

public static string concatCharacter(string[] str)

{

string s1 = "";

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

{

s1=s1+ str[i].Substring((str[i].Length) - 1);

}

return s1;

}

}

}

**Question 16:**

16.   **ArrayList to String Array**

Write a program that performs the following actions:  
  
1.Read m strings as input (fruit names).  
2.Create an arraylist to store the above m strings in this arraylist.  
3.Read n strings as input (fruit names).  
4.Create an arraylist to store the above n strings in this arraylist.  
5.Write a function fruitSelector which accepts the arraylists as input.  
6.Remove all fruits whose name ends with 'a' or 'e' from first arrayList and remove all fruits whose name begins  with 'm' or 'a' from second arrayList then combine the two lists and return the final output as a String array.  
7.If the array is empty the program will print as “No fruit found”  
Include a class UserMainCode with the static method **fruitSelector** which accepts the two arraylists and returns an array.  
  
Create a Class Main which would be used to read n strings and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of an integer (m) denoting the size of first arraylist. The next m elements would be the values of the first arraylist. The next input would be n denoting the size of the second arraylist. The next n elements would be the values of the second arraylist.  
  
Output consists of an array as per step 6. Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
3  
Apple  
Cherry  
Grapes  
4  
Orange  
Mango  
Melon  
Apple  
**Sample Output 1:**  
Cherry  
Grapes  
Orange

using System;

using System.Collections;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_16

{

static void fruitSelector(ArrayList a1,ArrayList a2)

{

ArrayList a3 = new ArrayList();

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

{

if (!(a1[i].ToString().StartsWith("a") || (a1[i].ToString().StartsWith("e"))))

a3.Add(a1[i]);

}

for (int i = 0; i < a2.Count; i++)

{

if (!(a2[i].ToString().StartsWith("m") || (a2[i].ToString().StartsWith("a"))))

a3.Add(a2[i]);

}

string[] str = new string[a3.Count];

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

str[i] = (string)a3[i];

if (str.Length == 0)

Console.WriteLine("No Fruits found ");

else

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

{

Console.WriteLine(str[i]);

}

}

static void Main()

{

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

ArrayList a1 = new ArrayList();

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

{

a1.Add(Console.ReadLine());

}

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

ArrayList a2 = new ArrayList();

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

{

a2.Add(Console.ReadLine());

}

fruitSelector(a1, a2);

Console.Read();

}

}

}

**Question 17:**

**17. Elements in ArrayList**

Use Collection Methods.  
Write a program that takes two ArrayLists as input and  finds out all elements present either in A or B, but not in both.

Include a class UserMainCode with the static method arrayListSubtractor which accepts the two arraylists and returns an array.  
  
Create a Class Main which would be used to read the inputs and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of an integer (m) denoting the size of first arraylist. The next m elements would be the values of the first arraylist. The next input would be n denoting the size of the second arraylist. The next n elements would be the values of the second arraylist.  
  
Output consists of an array. The elements in the output array need to be printed in sorted order.  
  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
4  
1  
8  
3   
5  
2  
3  
5  
**Sample Output 1:**  
1  
8  
  
**Sample Input 2:**  
4  
9  
1  
3  
5  
4  
1  
3  
5  
6  
**Sample Output 2:**  
6  
9

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_17

{

//static void arrayListSubtractor(ArrayList a1,ArrayList a2)

//{

// ArrayList a3 = new ArrayList();

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

// {

// for(int j=0;j<a2.Count;j++)

// {

// if (a1[i] != a2[j])

// {

// a3.AddRange(a1);

// a3.AddRange(a2);

// }

// }

// }

// foreach(var v in a3)

// {

// Console.WriteLine(v);

// }

//}

static void Main()

{

Console.WriteLine("Enter Size1: ");

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

Console.WriteLine("Enter Elements into list1: ");

ArrayList a1 = new ArrayList();

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

{

a1.Add(Convert.ToInt32(Console.ReadLine()));

}

Console.WriteLine("Enter Size2: ");

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

ArrayList a2 = new ArrayList();

Console.WriteLine("Enter elemnts into list2: ");

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

{

a2.Add(Convert.ToInt32(Console.ReadLine()));

}

ArrayList a3 = new ArrayList();

StringBuilder sb = new StringBuilder();

var v = from int x in a1 select x;

var w = from int y in a2 select y;

var z = v.Except(w);

var e = w.Except(v);

foreach (var c in z)

sb.Append(c);

foreach (var b in e)

sb.Append(b);

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

{

Console.WriteLine(sb[i]);

}

Console.Read();

}

}

}

**Question 18:**

**18.  Sum of Digits in a String**

Write code to get the sum of all the digits present in the given string.

Include a class **UserMainCode** with a static method **sumOfDigits** which accepts string input.

Return the sum as output. If there is no digit in the given string return -1 as output.

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

**Input and Output Format:**

Input consists of a string.

Output is a single integer which is the sum of digits in a given string.

Refer sample output for formatting specifications.

**Sample Input 1:**

good23bad4

**Sample Output 1:**

9

**Sample Input 2:**

good

**Sample Output 2:**

-1

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp7

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

int sum = UserMainCode.SumOfDigits(s);

Console.WriteLine(sum);

Console.Read();

}

}

class UserMainCode

{

public static int SumOfDigits(string s)

{

int sum = 0;

char[] ch = s.ToCharArray();

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

{

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

{

sum = sum + (ch[i])-48;

}

else

{

sum = -1;

}

}

return sum;

}

}

}

**Question 19:**

**19.  Word Count**

Given a string array (s) and non negative integer (n) and return the number of elements in the array which have same number of characters as the givent int N.  
Include a class UserMainCode with a static method **countWord** which accepts the string array and integer. The return type is the string formed based on rules.  
Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of a an integer indicating the number of elements in the string array followed the elements and ended by the non-negative integer (N).  
Output consists of a integer .  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
4  
a  
bb  
b  
ccc  
1  
**Sample Output 1:**  
2  
  
**Sample Input 2:**  
5  
dog  
cat  
monkey  
bear  
fox  
3  
**Sample Output 2:**  
3

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace JavDumps

{

class Prog\_19

{

static void Main()

{

int count = 0;

Console.WriteLine("Enter the size: ");

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

string[] s = new string[n];

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

{

s[i] = Console.ReadLine();

}

Console.WriteLine("Enter the desired length: ");

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

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

{

if (s[i].Length == len)

{

count++;

}

}

Console.WriteLine(count);

Console.Read();

}

}

}

**Question 20:**

**20.  IP Validator**

Write a program to read a string and validate the IP address. Print “Valid” if the IP address is valid, else print “Invalid”.

Include a class **UserMainCode** with a static method **ipValidator** which accepts a string. The return type (integer) should return 1 if it is a valid IP address else return 2.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of a string that corresponds to an IP.

Output consists of a string(“Valid” or “Invalid”).

Refer sample output for formatting specifications.

**Note**: An IP address has the format a.b.c.d where a,b,c,d are numbers between 0-255.

**Sample Input 1:**

132.145.184.210

**Sample Output 1:**

Valid

**Sample Input 2:**

132.145.184.290

**Sample Output 2:**

Invalid

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ipValidator

{

class Program

{

static void Main(string[] args)

{

string input = Console.ReadLine();

int output = UserMainCode.ipValidator(input);

if (output == 1)

{

Console.WriteLine("Valid");

}

else if (output == -1)

{

Console.WriteLine("Invalid");

}

}

}

class UserMainCode

{

public static int ipValidator(string input)

{

int count = 0,output=0;

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

int len = s1.Length;

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

{

if ((int.Parse(s1[i])>=0) && (int.Parse(s1[i]) <= 255))

{

count++;

}

}

if (count==4)

{

output = 1;

}

else

{

output = -1;

}

return output;

}

}

}

**c21.  Anagram**

Write a program to check whether the two given strings are anagrams.

Note: Rearranging the letters of a word or phrase to produce a new word or phrase, using all the original letters exactly once is called Anagram."

Include a class **UserMainCode** with a static method **“getAnagram”** that accepts 2 strings as arguments and returns an int. The method returns 1 if the 2 strings are anagrams. Else it returns -1.

Create a class **Main** which would get 2 Strings as input and call the static method **getAnagram** present in the UserMainCode.

**Input and Output Format:**

Input consists of 2 strings. Assume that all characters in the string are lower case letters.

Output consists of a string that is either “Anagrams” or “Not Anagrams”.

**Sample Input 1:**

eleven plus two

twelve plus one

**Sample Output 1:**

Anagrams

**Sample Input 2:**

orchestra

carthorse

**Sample Output 2:**

Anagrams

**Sample Input 3:**

cognizant

technologies

**Sample Output 3:**

Not Anagrams

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class prg21

{

public static void Main()

{

string s = Console.ReadLine().ToLower();

string s1 = Console.ReadLine().ToLower();

int a = getAnagram(s, s1);

if (a == 0)

{

Console.WriteLine("anagram");

}

if(a==1)

Console.WriteLine("not anagram");

Console.ReadLine();

}

public static int getAnagram(string s,string s1)

{

int n=0;

char[] ch = s.ToCharArray();

char[] ch1 = s1.ToCharArray();

Array.Sort(ch);

Array.Sort(ch1);

string s2 = new string(ch);

string s3 = new string(ch1);

if(s2==s3)

{

n= 0;

}

else

{

n = 1;

}

return n;

}

}

}

**22.  String processing – Long + Short + Long**

Obtain two strings S1,S2 from user as input. Your program should form a string of  “long+short+long”, with the shorter string inside of the longer String.  
Include a class UserMainCode with a static method **getCombo** which accepts two string variables. The return type is the string.  
Create a Class Main which would be used to accept two Input strings and call the static method present in UserMainCode.  
**Input and Output Format:**  
Input consists of two strings with maximum size of 100 characters.  
Output consists of an string.  
Refer sample output for formatting specifications.  
**Sample Input 1:**  
Hello  
Hi  
**Sample Output 1:**  
HelloHiHello

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class prg22

{

public static void Main()

{

string s = Console.ReadLine();

string s1 = Console.ReadLine();

string s3 = getCombo(s, s1);

Console.WriteLine(s3);

Console.ReadLine();

}

public static string getCombo(string s,string s1)

{

int x = s.Length;

int y = s1.Length;

string s4 = "";

if (x > y)

{

s4 = s + s1 + s;

}

else

s4 = s1 + s + s1;

return s4;

}

}

}

**23.  Odd Digit Sum**

Write a program to input a String array. The input may contain digits and alphabets (“de5g4G7R”). Extract odd digits from each string and find the sum and print the output.

For example, if the string is "AKj375A" then take 3+7+5=15 and not as 375 as digit.

Include a class **UserMainCode** with a static method **oddDigitSum** which accepts a string array and the size of the array. The return type (Integer) should return the sum.

Create a Class Main which would be used to accept Input Strings and call the static method present in UserMainCode.

Assume maximum length of array is 20.

**Input and Output Format:**

Input consists of an integer n, corresponds to the number of strings, followed by n Strings.

Output consists of an Integer.

Refer sample output for formatting specifications.

**Sample Input :**

3

cog2nizant1

al33k

d2t4H3r5

**Sample Output :**

15

**(1+3+3+3+5)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Collections;

namespace java\_dumps

{

class prg23

{

public static void Main()

{

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

string[] s = new string[20];

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

{

s[i] = Console.ReadLine();

}

int res = oddDigitSum(s,n);

Console.WriteLine(res);

Console.ReadLine();

}

public static int oddDigitSum(string[] s,int n)

{

int sum = 0;

string s1 = "";

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

{

s1 += s[i];

}

char[] ch = s1.ToCharArray();

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

{

if(char.IsNumber(ch[j]))

{

if(ch[j]%2!=0)

{

sum += ch[j];

}

}

}

return sum;

}

}

}

**24.  Forming New Word from a String**

Write a program to read a string and a positive integer n as input and construct a string with first n and last n characters in the given string.

Include a class **UserMainCode** with a static method **formNewWord** which accepts a string and positive integer .

The return type of the output should be a string (value) of first n character and last n character.

Create a class **Main** which would get the input as a string and integer n and call the static method**formNewWord** present in the UserMainCode.

**Input and Output Format:**

Input consists of a string of even length.

Output is a string .

Note: The given string length must be >=2n.

Refer sample output for formatting specifications.

**Sample Input 1:**

California

3

**Sample Output 1:**

Calnia

**Sample Input2:**

this

1

**Sample Output 2:**

Ts

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp8

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

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

string s1 = UserMainCode.formNewWord(s, n);

Console.WriteLine(s1);

Console.ReadLine();

}

}

class UserMainCode

{

public static string formNewWord(string s, int n)

{

string s1 = s.Substring(0, n);

string s2 = s.Substring(s.Length - n);

string s3 = s1 + s2;

return s3;

}

}

}

**25.  Decimal to Binary Conversion**

Write a Program that accepts a decimal number n, and converts the number to binary.  
  
Include a class UserMainCode with a static method **convertDecimalToBinary** which accepts an integer. The return type is long representing the binary number.  
  
Create a Class Main which would be used to accept the input integer and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of single integer.  
Output consists of a single long.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
5  
**Sample Output 1:**  
101

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class prg25

{

public static void Main()

{

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

long l = convertdecimaltobinary(n);

Console.WriteLine(l);

Console.ReadLine();

}

public static long convertdecimaltobinary(int n)

{

long l;

string s = Convert.ToString(n, 2);

l = Convert.ToInt64(s);

return l;

}

}

}

**26.  Palindrome & Vowels**

Write a program to check if a given string is palindrome and contains at least two different vowels.

Include a class UserMainCode with a static method **checkPalindrome** which accepts a string. The return type (integer) should be 1 if the above condition is satisfied, otherwise return -1.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Note – Case Insensitive while considering vowel, i.e a & A are same vowel, But Case sensitive while considering palindrome i.e abc CbA are not palindromes.

**Input and Output Format:**

Input consists of a string with maximum size of 100 characters.

Output consists of a single Integer.

Refer sample output for formatting specifications.

**Sample Input 1:**

Abceecba

**Sample Output 1:**

valid

**Sample Input 2:**

abcd

**Sample Output 2:**

Invalid

namespace java113

{

    class prog26

    {

        static int palindrome(string s1)

        {

            int n = 0;

            char[] ch1 = s1.ToLower().ToCharArray();

            Array.Reverse(ch1);

            string s2 = new string(ch1);

            Regex r = new Regex(@"^([A/E/I/O/U/a/e/i/o/u]{2})");

            Match m = r.Match(s1);

            if (m.Success)

            {

https://ssl.gstatic.com/ui/v1/icons/mail/images/cleardot.gif

                if (s1 == s2)

                {

                    n = 1;

                }

            }

            else

            {

                n = -1;

            }

            return n;

        }

        static void Main()

        {

            string s1 = Console.ReadLine();

            int num = palindrome(s1);

            if (num == 1)

            {

                Console.WriteLine("palindrome");

            }

            if (num == -1)

            {

                Console.WriteLine("not palindrome");

            }

            Console.ReadLine();

        }

    }

}

**27.  States and Capitals**

Write a program that construts a hashmap with “state” as key and “capital” as its value. If the next input is a state, then it should return capital$state in lowercase.

Include a class UserMainCode with a static method **getCapital** which accepts a hashmap. The return type is the string as given in the above statement

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of 2n+2 values. The first value corresponds to size of the hashmap. The next n pair of numbers contains the state and capital. The last value consists of the “state” input.

Output consists of a string as mentioned in the problem statement.

Refer sample output for formatting specifications.

**Sample Input 1:**

3

Karnataka

Bangaluru

Punjab

Chandigarh

Gujarat

Gandhinagar

Punjab

**Sample Output 1:**

chandigarh$punjab

namespace java113

{

    class prog27

    {

        static string statescapitals(string[] sa,int n)

        {

            Hashtable h = new Hashtable(n);

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

            {

                h.Add(sa[i], sa[i + 1]);

            }

            foreach(DictionaryEntry v in h)

            {

                Console.WriteLine("key{0}-value{1}", v.Key, v.Value);

            }

            string s = Console.ReadLine();

            string op = "";

            foreach (DictionaryEntry v in h)

            {

                if(v.Value.Equals(s))

                {

                    op = v.Key + "$" + v.Value;

                }

            }

            return op;

        }

        public static void Main()

        {

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

            string[] sa = new string[2\*n];

            for(int i=0;i<6;i++)

            {

                sa[i] = Console.ReadLine();

            }

            Console.WriteLine(statescapitals(sa,n));

            Console.ReadLine();

        }

    }

}

**28.  Leap Year**

Write a program to read a string containing date in DD/MM/YYYY format and check if its a leap year. If so, return true else return false.  
  
Include a class UserMainCode with a static method **isLeapYear** which accepts the string. The return type is the boolean indicating TRUE / FALSE.  
  
Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a string.  
  
Output consists of TRUE / FALSE.  
  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
  
23/02/2012  
  
**Sample Output 1:**  
  
TRUE  
  
**Sample Input 2:**  
  
12/12/2011  
  
**Sample Output 2:**  
  
FALSE

namespace java113

{

    class prog28

    {

        static bool leapyear(string s)

        {

            DateTime d = Convert.ToDateTime(s);

            int y = d.Year;

            if ((y % 4 == 0 && y % 100 != 0) || (y % 400 == 0))

            {

                return true;

            }

            else

            {

                return false;

            }

        }

        public static void Main()

        {

            string s = Console.ReadLine();

            Console.WriteLine(leapyear(s));

            Console.ReadLine();

        }

    }

}

**29.  Vowel Check**

Write a program to read a String and check if that String contains all the vowels. Print “yes” if the string contains all vowels else print “no”.

Include a class **UserMainCode** with a static method **getVowels** which accepts a string. The return type (integer) should return 1 if the String contains all vowels else return -1.

Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of a string.

Output consists of a string(“yes” or “no”).

Refer sample output for formatting specifications.

**Sample Input 1:**

abceiduosp

**Sample Output 1:**

yes

**Sample Input 2:**

bceiduosp

**Sample Output 2:**

no

namespace java113

{

    class prog29

    {

        static int vowels(string s1)

        {

            int n = 0;

                if (s1.Contains("a")&&s1.Contains("e") && s1.Contains("i")

                && s1.Contains("o") && s1.Contains("u"))

                {

                    n = 1;

                }

                else

                {

                    n = -1;

                }

            return n;

        }

        static void Main()

        {

            string s1 = Console.ReadLine();

            int num = vowels(s1);

            if (num == 1)

            {

                Console.WriteLine

("yes");

            }

            if (num == -1)

            {

                Console.WriteLine("no");

            }

            Console.ReadLine();

        }

    }

}

41.

Write a program to get two string inputs and validate the ID as per the specified

format.

Include a class UserMainCode with a static method validateIDLocations which

accepts two strings as input.

The return type of the output is a string Valid Id or Invalid Id.

Create a class Main which would get the input and call the static

method validateIDLocations present in the UserMainCode.

static void Main(string[] args)

{

Console.WriteLine("enter id");

string[] s = Console.ReadLine().Split('-');

Console.WriteLine("enter location");

string c = Console.ReadLine();

string s2 = s[2];

char[] a = new char[s2.Length];

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

{

if(s2[i]>='0' && s2[i]<='9')

{

a[i] = s2[i];

}

}

if (s.Length == 3)

{

if (s[0] == "CTS" && s[1] == c.Substring(0, 3) && a.Length == 4)

{

Console.WriteLine("valid");

}

else

{

Console.WriteLine("invalid");

}

}

else

{

Console.WriteLine("invalid");

}

Console.Read();

}

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

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

42.

You have recently learnt about hashmaps and in order to master it, you try and use it

in all of your programs.

Your trainer / teacher has given you the following exercise:

1.   Read 2n numbers as input where the first number represents a key and second one

as value. Both the numbers are of type integers.

2.   Write a function getAverageOfOdd to find out average of all values whose keys

are represented by odd numbers. Assume the average is an int and never a decimal

number. Return the average as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read 2n numbers and build the hashmap.

Call the static method present in UserMainCode.

public static void Main()

{

Console.WriteLine("enter number of elements in hashmap");

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

Console.WriteLine("enter key and values");

Hashtable h = new Hashtable();

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

{

h.Add(Convert.ToInt32(Console.ReadLine()),Convert.ToInt32(Console.ReadLine()));

}

int sum = 0;

int count = 0;

ICollection c = h.Keys;

foreach (var v in c)

{

if (((int)v) % 2 != 0)

{

sum += Convert.ToInt32(h[v]);

count++;

}

}

Console.WriteLine("after avg");

Console.WriteLine(sum/count);

Console.Read();

}

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

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

43. Test Vowels

Write a program to read a string and check if given string contains exactly five vowels in any order. Print “Yes” if the condition satisfies, else print “No”.

Assume there is no repetition of any vowel in the given string and all characters are lowercase.

Include a class UserMainCode with a static method testVowels which accepts a string. The return type (Integer) should return 1 if all vowels are present, else return 2.

Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

static void Main()

{

string s = Console.ReadLine().ToLower();

string s1 = new String(s.Distinct().ToArray());

int x = UserMainCode.testVowels(s1);

if (x == 1)

Console.WriteLine("true");

else

Console.WriteLine("false");

Console.Read();

}

class UserMainCode

{

public static int testVowels(string s1)

{

int count = 0;

for (int i = 0; i < s1.Length; i++)

{

if (s1[i] == 'a' || s1[i] == 'e' || s1[i] == 'i' || s1[i] == 'o' || s1[i] == 'u')

count++;

}

if (count == 5)

return 1;

else

return 2;

}

}

}

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

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

44. Regular Expression - III

Given a string (s) apply the following rules.

I)At least 8 characters must be present

II)At least one capital letter must be present

III)At least one small letter must be present

Iv)At least one special symbol must be present

V)At least one numeric value must be present

If the condition is satisifed then print valid else print invalid.

Include a class UserMainCode with a static method passwordValidation which accepts the string. The return type is the string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of string (valid / invalid) .

Refer sample output for formatting specifications.

Sample Input 1:

Technology$1213

Sample Output 1:

valid

public static string passwordvalidation(string p)

{

Regex r = new Regex(@"^(?=.\*[A-Z])(?=.\*[a-z])(?=.\*[a-z])(?=.\*\W).{8,}$");

if (r.IsMatch(p))

{

return "valid";

}

else

return "not valid";

}

public static void Main()

{

Console.WriteLine("enter the string");

string s = Console.ReadLine();

string res = prog44.passwordvalidation(s);

Console.Write(res);

Console.Read();

}

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

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

45. Average of Prime Locations

Write a program to read an integer array and find the average of the numbers located on the Prime location(indexes).

Round the avarage to two decimal places.

Assume that the array starts with index 0.

Include a class UserMainCode with a static method averageElements which accepts a single integer array. The return type (double) should be the average.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Double value.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 20.

Sample Input 1:

8

4

1

7

6

5

8

6

9

Sample Output 1:

7.5

static void Main()

{

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

int[] a = new int[n];

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

{

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

}

double x = UserMainCode.averageElements(n, a);

Console.WriteLine(x);

Console.Read();

}

}

class UserMainCode

{

public static double averageElements(int n, int[] a)

{

int count = 0, sum = 0, c = 0;

double avg = 0;

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

{

count = 0;

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

{

if (i % j == 0)

count++;

}

if (count == 2)

{

sum = sum + a[i];

c++;

}

}

avg = (double)sum / c;

return avg;

}

}

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

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

46. Middle of Array

Write a program to read an integer array and return the middle element in the array. The size of the array would always be odd.

Include a class UserMainCode with a static method getMiddleElement which accepts a single integer array. The return type (integer) should be the middle element in the array.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next 'n' integers correspond to the elements in the array.

Output consists of a single Integer value.

Refer sample output for formatting specifications.

Assume that the maximum number of elements in the array is 19.

Sample Input 1:

5

1

5

23

64

9

Sample Output 1:

23

static void Main()

{

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

int x = UserMainCod.getMiddleElement(n);

if (x == 1)

Console.WriteLine("enter only odd number");

else

Console.WriteLine(x);

Console.Read();

}

}

class UserMainCod

{

public static int getMiddleElement(int n)

{

int b = 0;

if (n % 2 == 0)

{

return 1;

}

else

{

int[] a = new int[n];

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

{

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

}

b = a[n / 2];

return b;

}

}

}

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

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

47. Negative String

Given a string input, write a program to replace every appearance of the word "is" by "is not".

If the word "is" is immediately preceeded or followed by a letter no change should be made to the string .

Include a class UserMainCode with a static method “negativeString” that accepts a String arguement and returns a String.

Create a class Main which would get a String as input and call the static method negativeString present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String.

Sample Input 1:

This is just a misconception

Sample Output 1:

This is not just a misconception

static void Main(string[] args)

{

string s = Console.ReadLine();

string res = prog47.Negativestring(s);

Console.Write(res);

Console.Read();

}

public static string Negativestring(string s)

{

string p = s.Replace("is", "is not");

return p;

}

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

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

48. Sum of Common Elements

Write a program to find out sum of common elements in given two arrays. If no common elements are found print - “No common elements”.

Include a class UserMainCode with a static method getSumOfIntersection which accepts two integer arrays and their sizes. The return type (integer) should return the sum of common elements.

Create a Class Main which would be used to accept 2 Input arrays and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2+m+n integers. The first integer corresponds to m (Size of the 1st array), the second integer corresponds to n (Size of the 2nd array), followed by m+n integers corresponding to the array elements.

Output consists of a single Integer corresponds to the sum of common elements or a string “No common elements”.

Refer sample output for formatting specifications.

Assume the common element appears only once in each array.

Sample Input 1:

4

3

2

3

5

1

1

3

9

Sample Output 1:

4

Sample Input 2:

4

3

2

3

5

1

12

31

9

Sample Output 2:

No common elements

public static void Main()

{

Console.WriteLine("enter the length of first array");

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

Console.WriteLine("enter the length of second array");

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

Console.WriteLine("enter first array elements");

int[] a = new int[m];

int[] b = new int[n];

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

{

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

}

Console.WriteLine("enter first array elements");

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

{

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

}

int res = prog48.sumofintersection(a, b);

if (res != 0)

{

Console.WriteLine(res);

}

else

Console.WriteLine("no common elements");

Console.Read();

}

public static int sumofintersection(int[] x,int[] y)

{

int sum = 0;

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

{

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

{

if(x[i]==y[j])

{

sum = sum + x[i];

break;

}

}

}

return sum;

}

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

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

49.

Given a string (s) apply the following rules.

1. String should not begin with a number.

If the condition is satisifed then print TRUE else print FALSE.

Include a class UserMainCode with a static method validateString which accepts the string.

The return type is the boolean formed based on rules.

Create a Class Main which would be used to accept the string and call the static method

present in UserMainCode.

public static void Main()

{

Console.WriteLine("enter the string");

string s = Console.ReadLine();

if((s[0]>='a' && s[0]<='z') || (s[0] >= 'A' && s[0] <= 'Z'))

{

Console.WriteLine("true");

}

else

{

Console.WriteLine("false");

}

Console.Read();

}

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

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

50.

Write a program to read a string and return the length of the largest &quot;chunk&quot; in the string.

A chunk is a repetition of same character 2 or more number of times. If the given string doest

not contain any repeated chunk of characters return -1.

Include a class UserMainCode with a static method getLargestSpan which accepts the

string. The return type is the integer.

Create a Class Main which would be used to accept the string and call the static method

present in UserMainCode.

public static void Main()

{

Console.WriteLine("enter the string");

string[] s = Console.ReadLine().Split(' ');

int max = 0;

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

{

string s1 = s[i];

for(int j=0;j<s1.Length;j++)

{

int count = 0;

for(int k=j+1;k<s1.Length;k++)

{

if(s1[j]==s1[k])

{

count++;

}

}

if (count > max)

max = count+1;

}

}

Console.WriteLine(max);

Console.Read();

}

**Question 51:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace \_113javadumps

{

class \_51

{

/// <summary>

/// For a given double number with atleast one decimal value, Write a program to compute the number of

/// digits before and after the decimal point in the following format –

/// noOfDigitsBeforeDecimal:noOfDigitsAfterDecimal.

// Note: Ignore zeroes at the end of the decimal (Except if zero is the only digit after decimal.

//Refer Example 2 and 3)

//Include a class UserMainCode with a static method findNoDigits which accepts the decimal value.

///The return type is string.

//Create a Class Main which would be used to accept the string and call the static method present in

//UserMainCode.

/// </summary>

/// <param name="args"></param>

static void Main(string[] args)

{

double d = Convert.ToDouble(Console.ReadLine());

string s = UserMainCode51.findNoDigits(d);

Console.WriteLine(s);

Console.Read();

}

}

class UserMainCode51

{

public static string findNoDigits(double d)

{

string s1 = d.ToString();

string[] s2 = s1.Split('.');

int count1 = s2[0].Length;

int count2 = 0;

int temp = 0;

char[] ch1 = s2[1].ToCharArray();

for (int i = ch1.Length-1; i > 0; i--)

{

if (ch1[i] != '0')

{

temp = 1;

break;

}

else ch1[i] = '\0';

}

if (temp == 1)

count2 = ch1.Length;

string s = count1 + ":" + count2;

return s;

}

}

}

**Question 52:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_52

{

/// <summary>

/// Write a program to read a string and an integer and return a string based on the below rules.

// If input2 is equal or greater than 3 then repeat the first three character of the String by given input2 times,

//separated by a space.

// If input2 is 2 then repeat the first two character of String two times separated by a space,

// If input2 is 1 then return the first character of the String.

//Include a class UserMainCode with a static method repeatString which takes a string & integer and returns

///a string based on the above rules.

//Create a Class Main which would be used to accept Input string and call the static method present in

//UserMainCode.

/// </summary>

public static void Main()

{

string s = Console.ReadLine();

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

string s1=UserMainCode52.repeatstring(s, n);

Console.WriteLine(s1);

Console.Read();

}

}

class UserMainCode52

{

public static string repeatstring(string s,int n)

{

string s1 = string.Empty;

string s2 = string.Empty;

if (n >= 3)

{

s1 = s.Substring(0, 3);

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

{

if (i != n && i!=0)

{

s2 += " ";

}

s2 += s1;

}

}

else if (n == 2)

{

s1 = s.Substring(0, 2);

s2 = s1 + " " + s1;

}

else

s2 = s[0].ToString();

return s2;

}

}

}

**Question 53:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_53

{

/// <summary>

/// Write a program to check whether the given input number is a Kaprekar number or not.

// Note : A positive whole number ‘n’ that has ‘d’ number of digits is squared and split into two

//pieces, a right-hand piece that has ‘d’ digits and a left-hand piece that has remaining ‘d’ or ‘d-1’ digits

//.If the sum of the two pieces is equal to the number, then ‘n’ is a Kaprekar number.

// If its Kaprekar number assign to output variable 1 else -1.

//Example 1:

//Input1:9

//9^2 = 81, right-hand piece of 81 = 1 and left hand piece of 81 = 8

//Sum = 1 + 8 = 9, i.e.equal to the number.Hence, 9 is a Kaprekar number.

//Example 2:

//Input1:45

//Hint:

//45^2 = 2025, right-hand piece of 2025 = 25 and left hand piece of 2025 = 20

//Sum = 25 + 20 = 45, i.e.equal to the number. Hence, 45 is a Kaprekar number."

//Include a class UserMainCode with a static method “getKaprekarNumber” that accepts an integer argument

//and returns an integer.The method returns 1 if the input integer is a Kaprekar number.Else the method returns -1.

//Create a class Main which would get the an Integer as input and call the static method getKaprekarNumber

///present in the UserMainCode.

/// </summary>

public static void Main()

{

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

int outval=UserMainCode53.getKaprekarNumber(n);

if (outval == 1)

Console.WriteLine("Given number is Kaprekar");

else Console.WriteLine("Given Number is not a kaprekar");

Console.ReadLine();

}

}

class UserMainCode53

{

public static int getKaprekarNumber(int n)

{

int num =(int)Math.Pow(n, 2);

string s1=num.ToString();

long one=Convert.ToInt64(s1.Substring(0, s1.Length / 2));

long two = Convert.ToInt64(s1.Substring(s1.Length / 2));

long num1 = one + two;

if (n == num1)

return 1;

else

return -1;

}

}

}

**Question 54:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_54

{

/// <summary>

/// Write a program to read a sentence in string variable and convert the first letter of each word to capital case. Print the final string.

// Note: - Only the first letter in each word should be in capital case in final string.

//Include a class UserMainCode with a static method printCapitalized which accepts a string. The return type(String) should return the capitalized string.

//Create a Class Main which would be used to accept a string and call the static method present in UserMainCode.

//Input and Output Format:

//Input consists of a strings.

//Output consists of a String(capitalized string).

//Refer sample output for formatting specifications.

//Sample Input:

//Now is the time to act!

//Sample Output:

//Now Is The Time To Act!

/// </summary>

public static void Main()

{

string s = Console.ReadLine();

string s1=UserMainCode54.printCapitalized(s);

Console.WriteLine(s1);

Console.Read();

}

}

class UserMainCode54

{

public static string printCapitalized(string s)

{

char[] ch = s.ToCharArray();

if (char.IsUpper(ch[0]))

{

}

else

{

ch[0] = (char)(ch[0] - 32);

}

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

{

if (ch[i]==' ')

{

ch[i + 1] = (char)(ch[i+1]-32);

//ch[i + 1]= Char.Parse(ch[i + 1].ToString().ToUpper());

}

}

string s1 = new string(ch);

return s1; }

}

}

**Question 55:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_55

{

/// <summary>

/// Write code to get two strings as input and If strings are of same length simply append them together and return the final string. If given strings are of different length, remove starting characters from the longer string so that both strings are of same length then append them together and return the final string.

// Include a class UserMainCode with a static method concatstring which accepts two string input.

// The return type of the output is a string which is the concatenated string.

//Create a class Main which would get the input and call the static method concatstring present in the UserMainCode.

/// </summary>

public static void Main()

{

Console.WriteLine(UserMainCode55.concatstring(Console.ReadLine(), Console.ReadLine()));

Console.Read();

}

}

class UserMainCode55

{

public static string concatstring(string s1,string s2)

{

if (s1.Length == s2.Length)

{

return string.Concat(s1, s2);

}

else

{

StringBuilder sb = new StringBuilder();

if (s1.Length > s2.Length)

{

for (int i = s1.Length-s2.Length; i < s1.Length; i++)

{

sb.Append(s1[i]);

}

return string.Concat(sb.ToString(), s2);

}

else

{

for (int i = s2.Length - s1.Length; i <s2.Length; i++)

{

sb.Append(s2[i]);

}

return string.Concat(sb.ToString(), s1);

}

}

}

}

}

**Question 56:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_56

{

/// <summary>

/// Write a program to read a string and count the number of words present in it.

//Include a class UserMainCode with a static method countWord which accepts the string. The return type is the integer giving out the count of words.

//Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

/// </summary>

public static void Main()

{

Console.WriteLine(UserMainCode56.countWord(Console.ReadLine()));

Console.Read();

}

}

class UserMainCode56

{

public static int countWord(string s)

{

int count = 0;

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

{

if(s[i]==' ')

{

count++;

}

}

return count + 1;

}

}

}

**Question 57:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_57

{

/// <summary>

/// Write a program to read a integer array, find the largest difference between adjacent elements and display the index of largest difference.

// EXAMPLE:

//input1: {2,4,5,1,9,3,8}

//output1: 4 (here largest difference 9-1=8 then return index of 9 ie,4)

//Include a class UserMainCode with a static method checkDifference which accepts the integer array.The return type is integer.

//Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

/// </summary>

public static void Main()

{

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

int[] arr = new int[n];

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

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

Console.WriteLine(UserMainCode57.checkDifference(arr));

Console.Read();

}

}

class UserMainCode57

{

public static int checkDifference(int[] arr)

{

int x=0, y=0, z = 0;

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

{

x = Math.Abs(arr[i] - arr[i + 1]);

if (x > y)

{

y = x;

z = i+1;

}

}

return z;

}

}

}

**Question 58:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace \_113javadumps

{

class \_58

{

/// <summary>

/// Write a program to validate the Date of Birth given as input in String format (MM/dd/yyyy) as per the validation rules given below. Return true for valid dates else return false.

//1. Value should not be null

//2. month should be between 1-12, date should be between 1-31 and year should be a four digit number.

//Include a class UserMainCode with a static method ValidateDOB which accepts the string. The return type is TRUE / FALSE.

//Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

//Input and Output Format:

//Input consists of a string.

//Output consists of TRUE / FALSE.

//Refer sample output for formatting specifications.

/// </summary>

public static void Main()

{

Console.WriteLine(UserMainCode58.validateDOB(Console.ReadLine()));

Console.Read();

}

}

class UserMainCode58

{

public static bool validateDOB(string s)

{

Regex r = new Regex(@"^(0?[0-9]|1[0-2])[/](0?[0-9]|[0-2][0-9]|3[0-1])[/]([0-9][0-9][0-9][0-9])$");

if (r.IsMatch(s))

{

return true;

}

else

{

return false;

}

}

}

}

**Question 59:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_59

{

/// <summary>

/// GIven three integers (a,b,c) find the sum. However, if one of the values is the same as another, both the numbers do not count

/// towards the sum and the third number is returned as the sum.

// Include a class UserMainCode with a static method getDistinctSum which accepts three integers and returns integer.

// Create a Class Main which would be used to accept three integers and call the static method present in UserMainCode.

// Input and Output Format:

//Input consists of three integers.

//Output consists of a integer.

//Refer sample output for formatting specifications.

//Sample Input 1:

//1

//2

//1

//Sample Output 1:

//2

//Sample Input 2:

//1

//2

//3

//Sample Output 2:

//6

/// </summary>

public static void Main()

{

int[] arr = new int[3];

for(int i = 0; i < 3; i++)

{

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

}

int sum=userMainCode59.getDistinctSum(arr);

Console.WriteLine(sum);

Console.Read();

}

}

class userMainCode59

{

public static int getDistinctSum(int[] arr)

{

int sum = 0;

int val1 = arr[0];

int val2 = arr[1];

int val3 = arr[2];

if (val1 != val2 && val1 != val3 && val2 != val3)

sum = val1 + val2 + val3;

else if (val1 == val2)

sum = val3;

else if (val1 == val3)

sum = val2;

else if (val2 == val3)

sum = val1;

else if (val1 == val2 && val1 == val3 && val2 == val3)

sum = val1;

return sum;

}

}

}

**Question 60:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_113javadumps

{

class \_60

{

/// <summary>

/// Write a program to input a person's name in the format "FirstName LastName" and return the person name in the following format -

/// "LastName, InitialOfFirstName".

/// Include a class UserMainCode with a static method nameFormatter which accepts a string. The return type(string) should return

/// the expected format.

/// Create a Class Main which would be used to accept Input String and call the static method present in UserMainCode.

// Input and Output Format:

//Input consists of a string that corresponds to a Person's name.

//Output consists of a string (person's name in expected format).

//Refer sample output for formatting specifications.

//Sample Input :

//Jessica Miller

//Sample Output:

//Miller, J

/// </summary>

public static void Main()

{

string s = Console.ReadLine();

char initial = s[0];

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

StringBuilder sb = new StringBuilder();

sb.Append(s1[1]);

sb.Append(',');

sb.Append(initial);

string sout = sb.ToString();

Console.Write(sout);

Console.Read();

}

}

}

61. Remove Elements

Write a program to remove all the elements of the given length and return the size of the final array as output. If there is no element of the given length, return the size of the same array as output.

Include a class UserMainCode with a static method removeElements which accepts a string array, the number of elements in the array and an integer. The return type (integer) should return the size of the final array as output.

Create a Class Main which would be used to accept Input String array and a number and call the static method present in UserMainCode.

Assume maximum length of array is 20.

Input and Output Format:

Input consists of a integers that corresponds to n, followed by n strings and finally m which corresponds to the length value.

Output consists of a single Integer.

Refer sample output for formatting specifications.

Sample Input 1:

5

a

bb

b

ccc

ddd

2

Sample Output 1:

4

namespace Dump\_practice

{

class \_61\_removing\_elements

{

static void Main()

{

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

string[] p = new string[m];

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

{

p[i] = Console.ReadLine();

}

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

Console.WriteLine(usermain\_61.countElements(p, n));

Console.Read();

}

}

class usermain\_61

{

public static int countElements(string []p,int n)

{

int count = 0;

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

{

if (p[i].Length == n)

p[i] = null;

}

foreach (var item in p)

{

if (item != null)

count++;

}

return count;

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

63. Reverse Split

Write a program to read a string and a character, and reverse the string and convert it in a format such that each character is separated by the given character. Print the final string.

Include a class UserMainCode with a static method reshape which accepts a string and a character. The return type (String) should return the final string.

Create a Class Main which would be used to accept a string and a character, and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and a character.

Output consists of a string (the final string).

Refer sample output for formatting specifications.

Sample Input:

Rabbit

-

Sample Output:

t-i-b-b-a-R

namespace Dump\_practice

{

class \_63\_Reverse\_Split

{

static void Main()

{

string h = Console.ReadLine();

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

Console.WriteLine(usermain\_63.reverseElemnts(h, c));

Console.Read();

}

}

class usermain\_63

{

public static string reverseElemnts(string h ,char c)

{

string s =String.Empty;

for(int i=h.Length-1;i>=0;i--)

{

s += h[i];

if (i != 0)

s += c;

}

return s;

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

64. Largest Key in HashMap

Write a program that construts a hashmap and returns the value corresponding to the largest key.

Include a class UserMainCode with a static method getMaxKeyValue which accepts a string. The return type (String) should be the value corresponding to the largest key.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the hashmap. The next n pair of numbers equals the integer key and value as string.

Output consists of a string which is the value of largest key.

Refer sample output for formatting specifications.

Sample Input 1:

3

12

amron

9

Exide

.0

7

SF

Sample Output 1:

Amron

namespace Dump\_practice

{

class \_64\_Largest\_Key\_in\_HashMap

{

static void Main()

{

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

Dictionary<int, string> di = new Dictionary<int, string>();

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

{

di.Add(Convert.ToInt32(Console.ReadLine()), Console.ReadLine());

}

Console.WriteLine(usermain64.keySort(di));

Console.Read();

}

}

class usermain64

{

public static string keySort(Dictionary<int,string> dic)

{

var li = dic.Keys.ToList();

li.Sort();

li.Reverse();

int n = li[0];

return dic[n];

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

65. Scores

Write a program to read a integer array of scores, if 100 appears at two consecutive locations return true else return false.

Include a class UserMainCode with a static method checkScores which accepts the integer array. The return type is boolean.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of a string that is either 'TRUE' or 'FALSE'.

Refer sample output for formatting specifications.

Sample Input 1:

3

1

100

100

Sample Output 1:

TRUE

Sample Input 2:

3

100

1

100

Sample Output 2:

FALSE

namespace Dump\_practice

{

class \_65\_scores

{

static void Main()

{

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

int[] p = new int[n];

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

{

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

}

Console.WriteLine(usermain65.checkScore(p));

Console.ReadLine();

}

}

class usermain65

{

public static string checkScore(int[]p)

{

int flag = 0;

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

{

if (i < p.Length - 1)

if (p[i] == 100 && p[i + 1] == 100)

flag++;

}

if (flag != 0)

return "true";

else

return "false";

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

66. Fetching Middle Characters from String

Write a program to read a string of even length and to fetch two middle most characters from the input string and return it as string output.

Include a class UserMainCode with a static method getMiddleChars which accepts a string of even length as input . The return type is a string which should be the middle characters of the string.

Create a class Main which would get the input as a string and call the static method getMiddleChars present in the UserMainCode.

Input and Output Format:

Input consists of a string of even length.

Output is a string .

Refer sample output for formatting specifications.

Sample Input 1:

this

Sample Output 1:

hi

Sample Input 1:

Hell

Sample Output 1:

el

namespace Dump\_practice

{

class \_66\_

{

static void Main()

{

string h = Console.ReadLine();

if (h.Length % 2 == 0)

Console.WriteLine(usermain66.middle(h));

else

Console.WriteLine("-1");

Console.ReadLine();

}

}

class usermain66

{

public static string middle(string c)

{

string res = "";

res += c[c.Length / 2 - 1];

res += c[c.Length / 2];

return res;

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

67. Password Validation

Given a method with a password in string format as input. Write code to validate the password using following rules:

- Must contain at least one digit

- Must contain at least one of the following special characters @, #, $

# Length should be between 6 to 20 characters.

Include a class UserMainCode with a static method validatePassword which accepts a password string as input.

If the password is as per the given rules return 1 else return -1.If the return value is 1 then print valid password else print as invalid password.

Create a Main class which gets string as an input and call the static method validatePassword present in theUserMainCode.

Input and Output Format:

Input is a string .

Output is a string .

Sample Input 1:

%Dhoom%

Sample Output 1:

Invalid password

Sample Input 2:

#@6Don

Sample Output 2:

Valid password

namespace Dump\_practice

{

class \_67\_Password\_Validation

{

static void Main()

{

string s = Console.ReadLine();

int n=usermain67.validate(s);

if (n == 1)

Console.WriteLine("valid");

else

Console.WriteLine("invalid");

Console.ReadLine();

}

}

class usermain67

{

public static int validate(string s)

{

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

int n = r.IsMatch(s) ? 1 : -1;

return n;

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

68. Anagrams

Write a program to read two strings and checks if one is an anagram of the other.

An anagram is a word or a phrase that can be created by rearranging the letters of another given word or phrase. We ignore white spaces and letter case. All letters of 'Desperation' can be rearranged to the phrase 'A Rope Ends It'.

Include a class UserMainCode with a static method checkAnagram which accepts the two strings. The return type is boolean which is TRUE / FALSE.

Create a Class Main which would be used to accept the two strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

tea

eat

Sample Output 1:

TRUE

Sample Input 2:

Desperation

A Rope Ends It

Sample Output 2:

TRUE

namespace Dump\_practice

{

class \_68\_Anagrams

{

static void Main()

{

string s = Console.ReadLine().ToLower();

string h = Console.ReadLine().ToLower();

Console.WriteLine(usermain68.anagram(s, h));

Console.ReadLine();

}

}

class usermain68

{

public static bool anagram(string s,string h)

{

char[] p = s.ToCharArray();

char[] l = h.ToCharArray();

Array.Sort(p);

Array.Sort(l);

string one = "";

string two = "";

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

{

if (p[i] != ' ')

one += p[i];

}

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

{

if (l[i] != ' ')

two += l[i];

}

if(one.Length==two.Length)

if (one.Equals(two))

return true;

else

return false;

return true;

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

69. Pattern Matcher

Write a program to read a string and check if it complies to the pattern 'CPT-XXXXXX' where XXXXXX is a 6 digit number. If the pattern is followed, then print TRUE else print FALSE.

Include a class UserMainCode with a static method CheckID which accepts the string. The return type is a boolean value.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output should print TRUE or FALSE .

Refer sample output for formatting specifications.

Sample Input 1:

CPT-302020

Sample Output 1:

TRUE

Sample Input 2:

CPT123412

Sample Output 2:

FALSE

namespace Dump\_practice

{

class \_69\_pattern\_matcher

{

static void Main()

{

string s = Console.ReadLine();

Console.WriteLine(usermain69.pattern(s));

Console.ReadLine();

}

class usermain69

{

public static bool pattern(string s)

{

Regex r = new Regex(@"([CPT-]{4}[0-9]{6})");

// ("[CPT-]{4}[0-9]{6}

int n = r.IsMatch(s) ? 1 : 0;

if (n == 1)

return true;

else

return false;

}

}

}

}

------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

70. Max Admissions

Write a program that reads details about number of admissions per year of a particular college, return the year which had maximum admissions. The details are stored in an arraylist with the first index being year and next being admissions count.

Include a class UserMainCode with a static method getYear which accepts a arraylist. The return type is an integer indicating the year of max admissions.

Create a Class Main which would be used to accept Input string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 2n+1 values. The first value corresponds to size of the data (year & admissions). The next n pair of numbers contains the year and admissions count.

Output consists of an integer as mentioned in the problem statement.

Refer sample output for formatting specifications.

Sample Input 1:

4

2010

200000

2011

300000

2012

45000

2013

25000

Sample Output 1:

2011

namespace Dump\_practice

{

class \_70\_Max\_admissions

{

static void Main()

{

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

Dictionary<int, int> dic = new Dictionary<int, int>();

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

{

dic.Add(Convert.ToInt32(Console.ReadLine()), Convert.ToInt32(Console.ReadLine()));

}

Console.WriteLine(usermain70.Greater(dic));

Console.ReadLine();

}

}

class usermain70

{

public static int Greater(Dictionary<int, int> disc)

{

var li = disc.Values.ToList();

li.Sort();

li.Reverse();

foreach (var boo in disc.Keys)

{

if (disc[boo] == li[0])

{

return boo;

}

}

return -1;

}

}

}

**71.  Grade Calculator**

A School wants to assign grades to its students based on their marks. You have been assigned as the programmer to automate this process. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:  
    Read student details from the User. The details would include name, mark in the given order. The datatype for name is string, mark is float.  
    You decide to build a hashmap. The hashmap contains name as key and mark as value.  
  
BUSINESS RULE:  
1. If Mark is less than 60, then grade is FAIL.  
2. If Mark is greater than or equal to 60, then grade is PASS.  
Note: FAIL/PASS should be in uppercase.  
Store the result in a new Hashmap with name as Key and grade as value.  
4. You decide to write a function **calculateGrade** which takes the above hashmap as input and returns the hashmap as output. Include this function in class UserMainCode.  
Create a Class Main which would be used to read student details in step 1 and build the hashmap. Call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of student details. The first number indicates the size of the students. The next two values indicate the name, mark.  
  
Output consists of a name and corresponding grade for each student.  
  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
3  
Avi  
76.36  
Sunil  
68.42  
Raja  
36.25  
  
**Sample Output 1:**  
Avi  
PASS  
Sunil  
PASS  
Raja  
FAIL

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Collections;

namespace java\_dumps

{

class Prog71

{

static void Main()

{

Console.WriteLine("Enter the num of students:");

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

Hashtable ht = new Hashtable();

Console.WriteLine("Enter student nam and marks:");

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

{

ht.Add(Console.ReadLine(), Convert.ToSingle(Console.ReadLine()));

}

Hashtable output= UserMainCode71.calculateGrade(ht,n);

foreach (var x in output.Keys)

{

Console.WriteLine(x + " " + output[x]);

}

Console.ReadLine();

}

}

class UserMainCode71

{

public static Hashtable calculateGrade(Hashtable ht,int n)

{

Hashtable output = new Hashtable();

foreach(var x in ht.Keys)

{

if ((float)ht[x] > 60)

{

output.Add(x, "PASS");

}

else

{

output.Add(x, "FAIL");

}

}

return output;

}

}

}

**72.  Count Vowels**

Given a string input, write a program to find the total number of vowels in the given string.

Include a class **UserMainCode** with a static method “**countVowels**” that accepts a String argument and returns an int that corresponds to the total number of vowels in the given string.

Create a class **Main** which would get the String as input and call the static method **countVowels** present in the UserMainCode.

**Input and Output Format:**

Input consists of a string.

Output consists of an integer..

**Sample Input:**

avinash

**Sample Output:**

3

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace java\_dumps

{

class Prog72

{

static void Main()

{

Console.WriteLine("Enter the string:");

string str= Console.ReadLine();

Console.WriteLine(UserMainCode72.countVowels(str));

Console.ReadLine();

}

}

class UserMainCode72

{

public static int countVowels(string str)

{

// return str.ToLower().Count(x => "aeiou".Contains(x));

return Regex.Matches(str, @"[aeiouAEIOU]").Count;

}

}

}

**73.  Validate Number**

Given a negative number as string input, write a program to validate the number and to print the corresponding positive number.

Include a class **UserMainCode** with a static method “**validateNumber**” that accepts a string argument and returns a string. If the argument string contains a valid negative number, the method returns the corresponding positive number as a string. Else the method returns -1.

Create a class **Main** which would get a String as input and call the static method **validateNumber** present in the UserMainCode.

**Input and Output Format:**

Input consists of a String.

Output consists of a String.

**Sample Input 1:**

-94923

**Sample Output 1:**

94923

**Sample Input 2:**

-6t

**Sample Output 2:**

-1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace java\_dumps

{

class Prog73

{

static void Main()

{

Console.WriteLine("Enter the string:");

string str = Console.ReadLine();

Console.WriteLine(UserMainCode73.validate(str));

Console.ReadLine();

}

}

class UserMainCode73

{

public static string validate(string str)

{

string s;

Match m= Regex.Match(str, @"^[-]?[0-9]{1,}$");

if (m.Success)

{

s = str.Replace("-", "");

}

else

s= "-1";

return s;

}

}

}

**74.  Experience Calculator**

Write a program to read Date of Joining and current date as Strings and Experience as integer and validate whether the given experience and calculated experience are the same. Print “true” if same, else “false”.

Include a class **UserMainCode** with a static method **calculateExperience**which accepts 2 strings and an integer. The return type is boolean.

Create a Class Main which would be used to accept 2 string (dates) and an integer and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of 2 strings and an integer, where the 2 strings corresponds to the date of joining and current date, and the integer is the experience.

Output is either “true” or “false”.

Refer sample output for formatting specifications.

**Sample Input 1:**

11/01/2010

01/09/2014

4

**Sample Output 1:**

true

**Sample Input 2:**

11/06/2009

01/09/2014

4

**Sample Output 2:**

false

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ExperienceCalculator

{

class Program

{

static void Main(string[] args)

{

DateTime dt1 = Convert.ToDateTime(Console.ReadLine());

DateTime dt2 = Convert.ToDateTime(Console.ReadLine());

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

bool res = UserMainCode74.calculateExperience(dt1, dt2, exp);

Console.WriteLine(res);

}

}

class UserMainCode74

{

public static bool calculateExperience(DateTime dt1, DateTime dt2, int n)

{

bool res = true;

if (Math.Abs((dt1.Year-dt2.Year))==n)

{

res = true;

}

else

{

res = false;

}

return res;

}

}

}

**75.  Retirement**

Given an input as HashMap which contains key as the ID and dob as value of employees, write a program to find out employees eligible for retirement. A person is eligible for retirement if his age is greater than or equal to 60.

Assume that the current date is 01/01/2014.

Include a class **UserMainCode** with a static method “retirementEmployeeList” that accepts a HashMap<String,String> as input and returns a ArrayList<String>. In this method, add the Employee IDs of all the retirement eligible persons to list and return the sorted list.

(Assume date is in dd/MM/yyyy format).

Create a class **Main** which would get the HashMap as input and call the static method **retirementEmployeeList**present in the UserMainCode.

**Input and Output Format:**

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

The next 2 lines of the input consists of strings that correspond to the id and dob of employee 1.

The next 2 lines of the input consists of strings that correspond to the id and dob of employee 2.

and so on...

Output consists of the list of employee ids eligible for retirement in sorted order.

**Sample Input :**  
4  
C1010

02/11/1987

C2020

15/02/1980

C3030

14/12/1952

T4040

20/02/1950

**Sample Output :**

[C3030, T4040]

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Collections;

namespace java\_dumps

{

class Prog75

{

static void Main()

{

Console.WriteLine("Enter the num of employees: ");

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

Console.WriteLine("Enter Emp id and DOB:");

Hashtable ht = new Hashtable();

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

{

ht.Add(Console.ReadLine(), Console.ReadLine());

}

ArrayList al= UserMainCode75.retriementEmployeeList(ht);

al.Sort();

foreach(string s in al)

{

Console.WriteLine(s);

}

Console.ReadLine();

}

}

class UserMainCode75

{

public static ArrayList retriementEmployeeList(Hashtable ht)

{

ArrayList al = new ArrayList();

foreach(var x in ht.Keys)

{

double x1=Convert.ToDateTime("01 / 01 / 2014").Subtract(Convert.ToDateTime(ht[x])).TotalDays;

if((x1/365)>60)

{

al.Add(x);

}

}

return al;

}

}

}

**76.  Repeating set of characters in a string**

Get a string and a positive integer n as input .The last n characters should repeat the number of times given as second input.Write code to repeat the set of character from the given string.

Include a class **UserMainCode** with a static method **getString** which accepts a string and an integer n as input.

The return type of the output is a string with repeated n characters.

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

**Input and Output Format:**

Input consists a string and a positive integer n.

Output is a string with repeated characters.

Refer sample output for formatting specifications.

**Sample Input 1:**

Cognizant

3

**Sample Output 1:**

Cognizantantantant

**Sample Input 2:**

myacademy

2

**Sample Output 2:**

myacademymymy

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Repeatingsetofcharactersinastring

{

class Program

{

static void Main(string[] args)

{

string str = Console.ReadLine();

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

string output = UserMainCode76.getString(str, n);

Console.WriteLine(output);

Console.ReadLine();

}

}

class UserMainCode76

{

public static string getString(string str, int n)

{

string res = str.Substring((str.Length) - n);

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

{

str = str + res;

}

return str;

}

}

}

**77.  Sum of Squares of Even Digits**

Write a program to read a number , calculate the sum of squares of even digits (values) present in the given number.

Include a class **UserMainCode** with a static method **sumOfSquaresOfEvenDigits** which accepts a positive integer . The return type (integer) should be the sum of squares of the even digits.

Create a class **Main** which would get the input as a positive integer and call the static method sumOfSquaresOfEvenDigits present in the UserMainCode.

**Input and Output Forma:**

Input consists of a positive integer n.

Output is a single integer .

Refer sample output for formatting specifications.

**Sample Input 1:**

56895

**Sample Output 1:**

100

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SumOfSquaresOfEvenDigits

{

class Program

{

static void Main(string[] args)

{

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

Console.WriteLine(UserMainCode77.sumOfSquaresOfEvenDigits(num));

Console.ReadLine();

}

}

class UserMainCode77

{

public static int sumOfSquaresOfEvenDigits(int num)

{

int temp = 0, sum = 0;

while (num > 0)

{

temp = num % 10;

if (temp % 2 == 0)

{

sum = sum + (temp \* temp);

}

num = num / 10;

// temp = 0;

}

return sum;

}

}

}

**78.  Regular Expression - 1**

Given a string (s) apply the following rules.  
  
1. String should be only four characters long.  
2. First character can be an alphabet or digit.  
3. Second character must be uppercase 'R'.  
4. Third character must be a number between 0-9.  
  
If all the conditions are satisifed then print TRUE else print FALSE.  
  
Include a class UserMainCode with a static method **validate** which accepts the string. The return type is the boolean formed based on rules.  
  
Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a string.  
  
Output consists of TRUE or FALSE .  
  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
vR4u  
  
**Sample Output 1:**  
TRUE  
  
**Sample Input 2:**  
vRau  
  
**Sample Output 2:**  
FALSE  
  
**Sample Input 3:**  
vrau  
  
**Sample Output 3:**  
FALSE

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace java\_dumps

{

class Prog78

{

static void Main()

{

Console.WriteLine("Enter the string:");

string str = Console.ReadLine();

Console.WriteLine(UserMainCode78.validate(str));

Console.ReadLine();

}

}

class UserMainCode78

{

public static bool validate(string str)

{

Regex r = new Regex(@"^([A-Za-z0-9][R][0-9][\w|\W])$");

if (r.IsMatch(str))

{

return true;

}

else

return false;

}

}

}

**79.  Reversing a Number**

Write a program to read a positive number as input and to get the reverse of the given number and return it as output.

Include a class **UserMainCode** with a static method **reverseNumber** which accepts a positive integer .

The return type is an integer value which is the reverse of the given number.

Create a **Main** class which gets the input as a integer and call the static method **reverseNumber** present in the **UserMainCode**

**Input and Output Format:**

Input consists of a positive integer.

Output is an integer .

Refer sample output for formatting specifications.

**Sample Input 1:**

543

**Sample Output 1:**

345

**Sample Input 1:**

1111

**Sample Output 1:**

1111

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ReversingaNumber

{

class Program

{

static void Main(string[] args)

{

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

Console.WriteLine(UserMainCode79.reverseNumber(num));

Console.ReadLine();

}

}

class UserMainCode79

{

public static int reverseNumber(int num)

{

int temp = 0, res = 0;

while (num > 0)

{

temp = num % 10;

res = (res \* 10) + temp;

num = num / 10;

}

return res;

}

}

}

**80.  Boundary Average**

Given an int array as input, write a program to compute the average of the maximum and minimum element in the array.

Include a class **UserMainCode** with a static method “**getBoundaryAverage**” that accepts an integer array as argument and returns a float that corresponds to the average of the maximum and minimum element in the array.

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

**Input and Output Format:**

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

The next n lines consist of integers that correspond to the elements in the array.  
Assume that the maximum number of elements in the array is 10.

Output consists of a single float value that corresponds to the average of the max and min element in the array.

**Sample Input :**

6

3

6

9

4

2

5

**Sample Output:**

5.5

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BoundaryAverage

{

class Program

{

static void Main(string[] args)

{

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

int[] arr = new int[size];

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

{

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

}

float avg = UserMainCode.getBoundaryAverage(arr);

Console.WriteLine(avg);

}

}

class UserMainCode

{

public static float getBoundaryAverage(int[] a)

{

float ans=a.Max() + a.Min();

float avg = ans / 2;

return avg;

}

}

}

**81.  Discount Rate Calculation**

Write a  program to calculate discount of the acccount holders based on the transaction amount and registration date using below mentioned prototype:  
1. Read account details from the User. The details would include id, DOR (date of registration) and transaction amount in the given order. The datatype for id is string, DOR is string and transaction amount is integer.  
2. You decide to build two hashmaps. The first hashmap contains employee id as key and DOR as value, and the second hashmap contains same employee ids as key and amount as value.  
3. Discount Amount as on 01/01/2015:  
    a. If the transaction amount greater than or equal to 20000 and registration greater than or equal to 5     year then discount rate is 20% of transaction amount.  
    b. If the transaction amount greater than or equal to 20000 and registration less then to 5 year then     discount rate is 10% of transaction amount.  
    c. If the transaction amount less than to 20000 and registration greater than or equal to 5 year then     discount rate is 15% of transaction amount.  
    d. If the transaction amount less than to 20000 and registration less then to 5 year then discount rate     is 5% of transaction amount.  
4. You decide to write a function **calculateDiscount** which takes the above hashmaps as input and returns the treemap  as output. Include this function in class UserMainCode.  
  
Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.  
**Input and Output Format:**  
Input consists of transaction details. The first number indicates the size of the employees. The next three values indicate the user id, user DOR and transaction amount. The DOR (Date of Registration) format is “dd-mm-yyyy”  
Output consists of a string which has the user id and discount amount one in a line for each user.  
Refer sample output for formatting specifications.  
**Sample Input 1:**  
4  
A-1010  
20-11-2007  
25000  
B-1011  
04-12-2010  
30000  
C-1012  
11-11-2005  
15000  
D-1013  
02-12-2012  
10000  
**Sample Output 1:**  
A-1010:5000  
B-1011:3000  
C-1012:2250  
D-1013:500

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Globalization;

namespace java\_dumps

{

class Prog81

{

static void Main()

{

Console.WriteLine("Enter num of employees:");

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

Dictionary<string, string> dl1 = new Dictionary<string, string>();

Dictionary<string, int> dl2 = new Dictionary<string, int>();

Console.WriteLine("Enter Empid,date-of-registration,Amount:");

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

{

string s1 = Console.ReadLine();

string s2 = string.Empty;

DateTime dt;

//date-format validation

msg: if(DateTime.TryParseExact(Console.ReadLine(), "dd-MM-yyyy", CultureInfo.InvariantCulture, DateTimeStyles.None, out dt))

{

s2 = dt.ToString();

}

else

{

Console.WriteLine("Wrong date format...[ex:dd-mm-yyyy]");

goto msg;

}

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

dl1.Add(s1, s2);

dl2.Add(s1, amount);

}

Dictionary<string,int> dl3= UserMainCode81.calculateDiscount(dl1, dl2);

foreach(var x in dl3.Keys)

{

Console.WriteLine(x + ":" + dl3[x]);

}

Console.ReadLine();

}

}

class UserMainCode81

{

public static Dictionary<string,int> calculateDiscount(Dictionary<string,string>dl1, Dictionary<string, int> dl2)

{

Dictionary<string, int> dl3 = new Dictionary<string, int>();

foreach(var x in dl1.Keys)

{

foreach(var y in dl2.Keys)

{

if (x == y)

{

double d = Convert.ToDateTime("01/01/2015").Subtract(Convert.ToDateTime(dl1[x])).TotalDays;

if ((dl2[y] >= 20000) && ((d/365)>=5))

{

dl3.Add(x, dl2[y] / 5);

}

else if ((dl2[y] >= 20000) && ((d / 365) <= 5))

{

dl3.Add(x, dl2[y] / 10);

}

else if ((dl2[y] <= 20000) && ((d / 365) >= 5))

{

dl3.Add(x,(int)(dl2[y] / 6.666));

}

else if ((dl2[y] <= 20000) && ((d / 365) <= 5))

{

dl3.Add(x, dl2[y] / 20);

}

}

}

}

return dl3;

}

}

}

**82.  Largest Span**

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

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class Prog82

{

static void Main()

{

Console.WriteLine("Enter the size of the Array:");

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

Console.WriteLine("Enter the elements into the array: ");

int[] a = new int[n];

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

{

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

}

Console.WriteLine(UserMainCode82.getlargestSpan(a));

Console.ReadLine();

}

}

class UserMainCode82

{

public static int getlargestSpan(int[] a)

{

int res = 0,c=0;

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

{

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

{

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

{

c = i + j + 1;

if (c > res)

res=c;

}

}

}

return res;

}

}

}

**83.  Sum Squares of Digits**

Write a program that accepts a positive number as input and calculates the sum of squares of individual digits of the given number.

Include a class **UserMainCode** with a static method “**getSumOfSquaresOfDigits**” that accepts an integer argument and returns an integer.

Create a class **Main** which would get an integer as input and call the static method **getSumOfSquaresOfDigits** present in the UserMainCode.

**Input and Output Format:**

Input consists of an integer.

Output consists of an integer.

**Sample Input:**

321

**Sample Output:**

14

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SumSquaresofDigits

{

class Program

{

static void Main(string[] args)

{

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

Console.WriteLine(UserMainCode83.getSumOfSquaresOfDigits(num));

Console.ReadLine();

}

}

class UserMainCode83

{

public static int getSumOfSquaresOfDigits(int num)

{

int temp = 0, sum = 0;

while (num>0)

{

temp = num % 10;

sum = sum + (temp \* temp);

num = num / 10;

}

return sum;

}

}

}

**84. Validating Input Password**

102.Write a code get a password as string input and validate using the rules specified below. Apply following validations:

1. Minimum length should be 8 characters

2. Must contain any one of these three special characters @ or \_ or #

3. May contain numbers or alphabets.

4. Should not start with special character or number

5. Should not end with special character

Include a class **UserMainCode** with a static method **validatePassword** which accepts password string as input and returns an integer. The method returns 1 if the password is valid. Else it returns -1.

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

**Input and Output Format:**

Input consists of a string.

Output is a string Valid or Invalid.

Refer sample output for formatting specifications.

**Sample Input 1:**

ashok\_23

**Sample Output 1:**

Valid

**Sample Input 2:**

1980\_200

**Sample Output 2:**

Invalid

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Text.RegularExpressions;

namespace java\_dumps

{

class Prog84

{

static void Main()

{

Console.WriteLine("Enter the Password:");

string str = Console.ReadLine();

Console.WriteLine(UserMainCode84.validatePassword(str));

Console.ReadLine();

}

}

class UserMainCode84

{

public static string validatePassword(string s)

{

string s1 = string.Empty;

Regex r = new Regex(@"^[A-za-z]{1}[A-Za-z0-9@\_#]{6,}[A-Za-z0-9]{1}$");

if (r.IsMatch(s))

{

s1 = "1";

}

else

s1 = "-1";

return s1;

}

}

}

**85.Sum of cubes and squares of elements in an array**

Write a program to get an int array as input and identify even and odd numbers. If number is odd get cube of it, if number is even get square of it. Finally add all cubes and squares together and return it as output.

Include a class **UserMainCode** with a static method **addEvenOdd** which accepts integer array as input.

The return type of the output is an integer which is the sum of cubes and squares of elements in the array.

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

**Input and Output Format:**

Input consists of integer array.

Output is an integer sum.

Refer sample output for formatting specifications.

**Sample Input 1:**

5

2

6

3

4

5

**Sample Output 1:**

208

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class Prog85

{

static void Main()

{

Console.WriteLine("Enter the Size of the array:");

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

int[] n = new int[num];

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

{

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

}

Console.WriteLine(UserMainCode85.addEvenOdd(n));

Console.ReadLine();

}

}

class UserMainCode85

{

public static int addEvenOdd(int[] n)

{

int oddSum = 0, evenSum = 0,final=0;

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

{

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

{

evenSum = evenSum + (n[i]\*n[i]);

}

else

{

oddSum = oddSum + (n[i] \* n[i]\* n[i]);

}

final = evenSum + oddSum;

}

return final; }

}

}

**86.Interest Calculation**

Write a  program to calculate amount of the acccount holders based on the below mentioned prototype:  
1. Read account details from the User. The details would include id, DOB (date of birth) and amount in the given order. The datatype for id is string, DOB is string and amount is integer.  
2. You decide to build two hashmaps. The first hashmap contains employee id as key and DOB as value, and the second hashmap contains same employee ids as key and amount as value.  
3. Rate of interest as on 01/01/2015:  
    a. If the age greater than or equal to 60 then interest rate is 10% of Amount.  
    b.If the age less then to 60 and greater than or equal to 30 then interest rate is 7% of Amount.  
    v. If the age less then to 30 interest rate is 4% of Amount.  
4. Revised Amount= principle Amount + interest rate.  
5.  You decide to write a function **calculateInterestRate** which takes the above hashmaps as input and returns the treemap  as output. Include this function in class UserMainCode.  
  
Create a Class Main which would be used to read employee details in step 1 and build the two hashmaps. Call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of account details. The first number indicates the size of the acoount. The next three values indicate the user id, DOB and amount. The Employee DOB format is “dd-mm-yyyy”  
Output consists of the user id and the amount for each user one in a line.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
4  
SBI-1010  
20-01-1987  
10000  
SBI-1011  
03-08-1980  
15000  
SBI-1012  
05-11-1975  
20000  
SBI-1013  
02-12-1950  
30000  
**Sample Output 1:**  
SBI-1010:10400  
SBI-1011:16050  
SBI-1012:21400  
SBI-1013:33000

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Globalization;

namespace java\_dumps

{

class Prog86

{

static void Main()

{

Console.WriteLine("Enter num of Users:");

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

Dictionary<string, string> dl1 = new Dictionary<string, string>();

Dictionary<string, int> dl2 = new Dictionary<string, int>();

Console.WriteLine("Enter userid,date-of-birth,Amount:");

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

{

string s1 = Console.ReadLine();

string s2 = string.Empty;

DateTime dt;

//date-format validation

msg: if (DateTime.TryParseExact(Console.ReadLine(), "dd-MM-yyyy", CultureInfo.InvariantCulture, DateTimeStyles.None, out dt))

{

s2 = dt.ToString();

}

else

{

Console.WriteLine("Wrong date format...[ex:dd-mm-yyyy]");

goto msg;

}

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

dl1.Add(s1, s2);

dl2.Add(s1, amount);

}

Dictionary<string, int> dl3 = UserMainCode86.calculateInterestRate(dl1, dl2);

foreach (var x in dl3.Keys)

{

Console.WriteLine(x + ":" + dl3[x]);

}

Console.ReadLine();

}

}

class UserMainCode86

{

public static Dictionary<string, int> calculateInterestRate(Dictionary<string, string> dl1, Dictionary<string, int> dl2)

{

Dictionary<string, int> dl3 = new Dictionary<string, int>();

foreach (var x in dl1.Keys)

{

foreach (var y in dl2.Keys)

{

if (x == y)

{

double d = Convert.ToDateTime("01/01/2015").Subtract(Convert.ToDateTime(dl1[x])).TotalDays;

if ((d / 365) >= 60)

{

int Final\_Amount = dl2[y] + dl2[y] / 10;

dl3.Add(x, Final\_Amount);

}

else if (((d / 365) < 60) && ((d / 365) >= 30))

{

int Final\_Amount = dl2[y] + (int)(dl2[y] / 14.285);

dl3.Add(x, Final\_Amount);

}

else

{

int Final\_Amount = dl2[y] + (int)(dl2[y] / 25);

dl3.Add(x, Final\_Amount);

}

}

}

}

return dl3;

}

}

}

**87.String Processing - V**

Write a program to read a string and also a number N. Form a new string made up of n repetitions of the last n characters of the String. You may assume that n is between 1 and the length of the string.  
  
Include a class UserMainCode with a static method **returnLastRepeatedCharacters** which accepts the string and the number n. The return type is the string as per the problem statement.  
  
Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
Input consists of a string and integer.  
Output consists of a string.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
Hello  
2  
**Sample Output 1:**  
lolo  
  
**Sample Input 2:**  
Hello  
3  
**Sample Output 2:**  
llollollo

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class Prog87

{

static void Main()

{

Console.WriteLine("Enter the string:");

string str = Console.ReadLine();

Console.WriteLine("Enter the value between 0 and {0}: ",str.Length);

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

Console.WriteLine(UserMainCode87.returnLastRepeatedCharacters(str, n));

Console.ReadLine();

}

}

class UserMainCode87

{

public static string returnLastRepeatedCharacters(string str,int n)

{

string res = str.Substring((str.Length) - num);

string str1 = "";

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

{

str1 = str1 + res;

}

return str1;

}

}

}

**88.String Processing - III**

Write a program to read a string where all the lowercase 'x' chars have been moved to the end of the string.  
  
Include a class UserMainCode with a static method **moveX** which accepts the string. The return type is the modified string.  
  
Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.  
  
**Input and Output Format:**  
  
Input consists of a string.  
Output consists of a string.  
Refer sample output for formatting specifications.  
  
**Sample Input 1:**  
xxhixx  
  
**Sample Output 1:**  
hixxxx  
  
**Sample Input 2:**  
XXxxtest  
  
**Sample Output 2:**  
XXtestxx

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class Prog88

{

static void Main()

{

Console.WriteLine("ENter the string: ");

string str = Console.ReadLine();

Console.WriteLine(UserMainCode88.moveX(str));

Console.ReadLine();

}

}

class UserMainCode88

{

public static string moveX(string str)

{

return string.Concat(str.Where(x => x != 'x').Concat(str.Where(x => x == 'x')));

}

}

}

**89.Duplicate Characters**

Write a Program which removes duplicate characters from the string. Your program should read a sentence (string) as input from user and return a string removing duplicate characters. Retain the first occurance of the duplicate character. Assume the characters are case – sensitive.

Include a class UserMainCode with a static method **removeDuplicates** which accepts a string. The return type is the modified sentence of type string.

Create a Class Main which would be used to accept the input string and call the static method present in UserMainCode.

**Input and Output Format:**

Input consists of a string with maximum size of 100 characters.

Output consists of a single string.

Refer sample output for formatting specifications.

**Sample Input 1:**

hi this is sample test

**Sample Output 1:**

hi tsample

**Sample Input 2:**

ABC DEF

**Sample Output 2:**

ABC DEF

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class Prog89

{

static void Main()

{

Console.WriteLine("Eter the string: ");

string str = Console.ReadLine();

Console.WriteLine(UserMainCode89.removeDuplicates(str));

Console.ReadLine();

}

}

class UserMainCode89

{

public static string removeDuplicates(string str)

{

return string.Concat(str.Distinct());

}

}

}

**90.Dash Check**

Write a program to read two strings and check whether or not they have dashes in the same places. Print “Yes” if the condition satisfies, else print “No”.

Include a class **UserMainCode** with a static method **compareDashes**which accepts two strings. The return type (Integer) should return 1 if all dashes are placed correctly, else return 2.

Create a Class Main which would be used to accept two strings and call the static method present in UserMainCode.

**Note:**The strings must have exactly the same number of dashes in exactly the same positions. The strings might be of different length.

**Input and Output Format:**

Input consists of two strings.

Output consists of a string (“Yes” or “No”).

Refer sample output for formatting specifications.

**Sample Input 1:**

hi—there-you.

12--(134)-7539

**Sample Output 1:**

Yes

**Sample Input 2:**

-15-389

-xyw-zzy

**Sample Output 2:**

No

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_dumps

{

class Prog90

{

static void Main()

{

Console.WriteLine("Eter the string1: ");

string str1 = Console.ReadLine();

Console.WriteLine("Eter the string2: ");

string str2 = Console.ReadLine();

int i= UserMainCode90.compareDashes(str1, str2);

if (i == 1)

Console.WriteLine("YES");

else

Console.WriteLine("NO");

Console.ReadLine();

}

}

class UserMainCode90

{

public static int compareDashes(string str1,string str2)

{

int count1 = 0,count2 = 0;

for(int i=0;i<str1.Length;i++)

{

if (str1[i] == '-')

{

count1++;

if(str2[i]=='-')

{

count2++;

}

}

}

if (count1 == count2)

return 1;

else

return 2;

}

}

}

91)Write a program to read an integer array and find the index of larger number of the two

adjacent numbers with largest difference. Print the index.

class pro\_91

{

Console.WriteLine("enter the number of elements");

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

int[] a = new int[n];

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

{

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

}

Console.WriteLine(usermethod(a));

Console.Read();

}

public static int usermethod(int[] b)

{

int c = 0,d=0,e=0;

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

{

c = Math.Abs(b[i + 1] - b[i]);

if(c>d)

{

d = c;

e = i + 1;

}

}

return e;

}

}

92.Unique Characters in a string

Write a program that takes a string and returns the number of unique characters in the string. If the given string doest not contain any unique characters return -1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_91\_100\_programs

{

class \_92

{

static void Main()

{

int flag = 0, count = 0;

string s=Console.ReadLine();

char[] c1 = s.ToCharArray();

for (int i = 0; i < c1.Length; i++)

{

for (int j=0;j<c1.Length;j++)

{

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

flag = 1;

}

if (flag == 0)

{

count++;

}

flag = 0;

}

if (count > 0)

Console.WriteLine("count:" + count);

else

Console.WriteLine(-1);

Console.Read();

}

}

}

93)Write a program that accepts a positive number as input and calculates the sum of digits at even indexes (say evenSum) and sum of digits at odd indexes (say oddSum) in the given number. If both the sums are equal , print &#39;yes&#39;, else print no.

class pro93

{

public static void Main()

{

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

usermethod(n);

Console.Read();

}

public static void usermethod(int b)

{

int esum = 0, osum = 0, count = 0, rem = 0;

while (b != 0)

{

rem = b % 10;

if (count % 2 == 0)

{

esum += rem;

}

else

{

osum += rem;

}

count++;

b /= 10;

}

//Console.WriteLine("even sum is:" + esum + "odd sum is:" + osum);

if (esum == osum)

{

Console.WriteLine("Yes");

}

else

{

Console.WriteLine("No");

}

}

}

94)Write a program to accept a string array as input, convert all the elements into lowercase and

sort the string array. Display the sorted array.

class pro94

{

public static void Main()

{

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

string[] s = new string[n];

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

{

s[i] = Console.ReadLine().ToLower();

}

Array.Sort(s);

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

{

Console.WriteLine(s[i]);

}

Console.Read();

}

}

96)Given an input as string and write code to encrypt the given string using following rules and

return the encrypted string:

1. Replace the characters at odd positions by next character in alphabet.

2. Leave the characters at even positions unchanged.

Note:

- If an odd position charater is &#39;z&#39; replace it by &#39;a&#39;.

- Assume the first character in the string is at position 1.

class pro96

{

public static void Main()

{

string s = Console.ReadLine().ToLower();

string s1 = "";

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

{

if (i % 2 == 0)

{

s1 = s1 + (char)(((s[i] - 'a' + 1) % 26) + 'a');

}

else

s1 = s1 + s[i];

}

Console.WriteLine(s1);

Console.Read();

}

}

95)Given a method with two date strings in yyyy-mm-dd format as input. Write code to find the

difference between two dates in months.

class pro95

{

public static void Main()

{

Console.WriteLine("Enter first Date(ex:yyyy/mm/dd) ");

DateTime dt = Convert.ToDateTime(Console.ReadLine());

Console.WriteLine("Enter second Date(ex:yyyy/mm/dd) ");

DateTime ds = Convert.ToDateTime(Console.ReadLine());

int ts = Math.Abs(12 \* (dt.Year - ds.Year) + dt.Month - ds.Month);

Console.WriteLine(ts);

Console.Read();

}

}

98.Max Vowels

Write a Program which fetches the word with maximum number of vowels. Your program should read a sentence as input from user and return the word with max number of vowels. In case there are two words of maximum length return the word which comes first in the sentence.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_91\_100\_programs

{

class \_98

{

static void Main()

{

int count = 0,temp=0,l=0;string word = "";

string[] s=Console.ReadLine().Split();

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

{

//word = s[i];

char[] c = s[i].ToCharArray();

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

{

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

{

count++;

}

}

if (count >= temp)

{

temp = count;

word = s[i];

}

count = 0;

}

Console.WriteLine(word);

Console.Read();

}

}

}

99.Date Validation

Write a program to read a string representing a date. The date can be in any of the three formats

1:dd-MM-yyyy 2: dd/MM/yyyy 3: dd.MM.yyyy

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Globalization;

namespace \_91\_100\_programs

{

class \_99

{

static void Main()

{

//99.Date Validation

Console.WriteLine("Enter a date(1:dd-MM-yyyy 2: dd/MM/yyyy 3: dd.MM.yyyy) in given formats:");

string s = Console.ReadLine();

string p1 = @"^([0-2]{1}[1-9]{1}||[1-3]{1}[0-1]{1})[-]([0]{1}[1-9]{1}||[1]{1}[0-2]{1})[-]([1-9]{4,4})$";

string p2 = @"^([0-2]{1}[1-9]{1}||[1-3]{1}[0-1]{1})[/]([0]{1}[1-9]{1}||[1]{1}[0-2]{1})[/]([1-9]{4,4})$";

string p3 = @"^([0-2]{1}[1-9]{1}||[1-3]{1}[0-1]{1})[.]([0]{1}[1-9]{1}||[1]{1}[0-2]{1})[.]([1-9]{4,4})$";

if (Regex.IsMatch(s, p1))

Console.WriteLine("Valid");

else if (Regex.IsMatch(s, p2))

Console.WriteLine("Valid");

else if (Regex.IsMatch(s, p3))

Console.WriteLine("Valid");

else

Console.WriteLine("Invalid");

Console.Read();

}

}

}

100.Phone Number Validator

Given a phone number as a string input, write a program to verify whether the phone number is valid using the following business rules:

-It should contain only numbers or dashes (-)

- dashes may appear at any position

-Should have exactly 10 digits

using System;

using System.Collections;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

namespace \_91\_100\_programs

{

class \_100

{

static void Main()

{

// 100.Phone Number Validator

Console.WriteLine("Enter a PhoneNumber");

Console.WriteLine(validatePhoneNumber(Console.ReadLine()));

Console.Read();

}

static int validatePhoneNumber(string s)

{

int count = 0;

Regex r = new Regex(@"^([0-9]{1,})");

ArrayList al=new ArrayList();

if (s.Contains('-'))

{

al.AddRange(s.Split('-'));

}

else

{

if ((Regex.IsMatch(s, @"^([0-9]{10,10})")) == true)

{

return 1;

}

else

{

return 2;

}

}

foreach (var h in al)

{

if (r.IsMatch(h.ToString()))

{

count = count + h.ToString().Length;

}

else

{

return 2;

}

}

if (count == 10)

return 1;

else

return 2;

}

}

}

101.Average of Primes

Write a program to read an array and find average of all elements located at index i, where i is a prime number. Type cast the average to an int and return as output. The index starts from 0.

Include a class UserMainCode with a static method addPrimeIndex which accepts a single integer array. The return type (integer) should be the average of all elements located at index i where i is a prime number.

Create a Class Main which would be used to accept Input array and call the static method present in UserMainCode.

ans:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp15

{

class Prog101

{

static void Main()

{

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 ans=UserMainCode1.AddPrimeIndex(a);

Console.WriteLine(ans);

Console.Read();

}

}

class UserMainCode1

{

public static int AddPrimeIndex(int[] a)

{

int sum = 0, count = 0, avg = 0 ;

int[] b = new int[a.Length];

int l = 0;

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

{

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

{

if (i % j == 0)

count++;

}

if (count>1)

{

//Console.WriteLine(i);

b[l] = a[i];

l++;

}

count = 0;

}

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

{

sum += b[i];

}

avg = sum /l;

return avg;

}

}

}

-----------------------------------------------------------------------------------------------------------------------------

102.Palindrome - In Range

Write a program to input two integers, which corresponds to the lower limit and upper limit respectively, and find the sum of all palindrome numbers present in the range including the two numbers. Print the sum.

Include a class UserMainCode with a static method addPalindromes which accepts two integers. The return type (Integer) should return the sum if the palindromes are present, else return 0.

Create a Class Main which would be used to accept two integer and call the static method present in UserMainCode.

class Program

{

static void Main(string[] args)

{

Console.WriteLine(UserMainCode1.addPalindromes(int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine())));

Console.Read();

}

}

class UserMainCode1

{

public static int addPalindromes(int a,int b)

{

int rem, sum = 0,value=0;

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

{

int temp = i;

sum = 0;

while(temp>0)

{

rem = temp % 10;

sum = (sum \* 10) + rem;

temp = temp / 10;

}

if (i == sum)

value = value + i;

}

return value;

}

}

--------------------------------------------------------------------------------------------------------------------------------------

103.Math Calculator

Write a program that accepts three inputs, first two inputs are operands in int form and third one being one of the following five operators: +, -, \*, /, %. Implement calculator logic and return the result of the given inputs as per the operator provided. In case of division, Assume the result would be integer.

Include a class UserMainCode with a static method calculator which accepts two integers, one operand and returns the integer.

Create a Class Main which would be used to accept three integers and call the static method present in UserMainCode.

class \_103

{

static void Main()

{

Console.WriteLine( UserMaincode2.calculator(int.Parse(Console.ReadLine()),int.Parse(Console.ReadLine()),Convert.ToChar(Console.ReadLine())));

Console.Read();

}

}

class UserMaincode2

{

public static int calculator(int a,int b,char c)

{

int sum = 0;

switch(c)

{

case '+': sum =a + b;

break;

case '-':

sum = a - b;

break;

case '\*':

sum = a \* b;

break;

case '/':

sum = a / b;

break;

case '%':

sum = a % b;

break;

default: sum = -1;

break;

}

return sum;

}

}

--------------------------------------------------------------------------------------------------------------------------------------

104.Shift Left

Write a program to read a integer array of scores, and return a version of the given array where all the 5's have been removed. The remaining elements should shift left towards the start of the array as needed,

and the empty spaces at the end of the array should be filled with 0.

So {1, 5, 5, 2} yields {1, 2, 0, 0}.

Include a class UserMainCode with a static method shiftLeft which accepts the integer array. The return type is modified array.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

ans:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Collections;

namespace ConsoleApp15

{

class Prog104

{

static void Main()

{

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[] res = new int[a.Length];

res = UserMainCode2.ShiftLeft(a);

Console.WriteLine();

foreach(int item in res)

{

Console.WriteLine(item);

}Console.Read();

}

}

class UserMainCode2

{

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

{

ArrayList al = new ArrayList();

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

{

if (a[i] != 5)

{

al.Add(a[i]);

}

}

int s = a.Length - al.Count;

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

{

al.Add(0);

}

int[] b = new int[al.Count];

int j = 0;

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

{

b[i] = (int)al[i];

}

return b;

}

}

}

------------------------------------------------------------------------------------------------------------------

105.Repeat Front

Given a string (s) and non negative integer (n) apply the following rules.

1. Display the first three characters as front.

2. If the length of the string is less than 3, then consider the entire string as front and repeat it n times.

Include a class UserMainCode with a static method repeatFirstThreeCharacters which accepts the string and integer. The return type is the string formed based on rules.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

class \_105RepeatFront

{

static void Main()

{

Console.WriteLine(UserMaincode3.repeatFirstThreeCharaters(Console.ReadLine(), int.Parse(Console.ReadLine())));

Console.Read();

}

}

class UserMaincode3

{

public static string repeatFirstThreeCharaters(string s,int n)

{

StringBuilder sb = new StringBuilder();

while(n>0)

{

if(s.Length > 3)

for(int i=0;i<3;i++)

{

sb.Append(s[i]);

}

else

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

{

sb.Append(s[i]);

}

n--;

}

return sb.ToString();

}

}

------------------------------------------------------------------------------------------------------------------------------------

106.Regular Expression – 3 (Phone Validator)

Given a phone number as string, validate the same based on the following rules.

1. Value should contain only numbers.

2. Value should contain 10 digits.

3. Value should not start with 00.

If all the conditions are satisifed then print TRUE else print FALSE.

Include a class UserMainCode with a static method validatePhone which accepts the string. The return type is the boolean formed based on rules.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

ans:

using System;

using System.Text.RegularExpressions;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PhoneNUmberValidator3

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

bool res = UserMainCode3.ValidatePhone(s);

Console.WriteLine(res);

Console.Read();

}

}

class UserMainCode3

{

public static bool ValidatePhone(string s)

{

bool res = true;

Regex r = new Regex(@"^([1-9]{2}[0-9]{8})$");

Match m = r.Match(s);

if (m.Success)

{

res = true;

}

else

{

res = false;

}

return res;

}

}

}--------------------------------------------------------------------------------------------------------------------

107.Sum of Lowest marks

Given input as HashMap, value consists of marks and rollno as key.Find the sum of the lowest three subject marks from the HashMap.

Include a class UserMainCode with a static method getLowest which accepts a Hashmap with marks and rollno.

The return type of the output is the sum of lowest three subject marks.

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

Input and Output Format:

class \_107sumoflowestmarks

{

static void Main()

{

Hashtable h1 = new Hashtable();

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

while (n > 0)

{.

h1.Add(Convert.ToInt32(Console.ReadLine()), Convert.ToInt32(Console.ReadLine()));

n--;

}

Console.WriteLine(UserMainCode4.getLowest(h1));

Console.Read();

}

}

class UserMainCode4

{

public static int getLowest(Hashtable h)

{

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

ICollection c = h.Keys;

foreach (var v in c)

li.Add(Convert.ToInt32(h[v]));

li.Sort();

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

l.AddRange(li.Take(3));

return l.Sum();

}

}

-------------------------------------------------------------------------------------------------------------------------

108.String Processing - MixMania

Write a program to read a string and check if it starts with '\_ix' where '\_' is any one char(a-z, A-Z, 0-9).

If specified pattern is found return true else false.

Include a class UserMainCode with a static method checkPattern which accepts the string. The return type is TRUE / FALSE.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

Mix Mania

Sample Output 1:

TRUE

using System;

using System.Text.RegularExpressions;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PhoneNUmberValidator3

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

bool res = UserMainCode3.ValidatePhone(s);

Console.WriteLine(res);

Console.Read();

}

}

class UserMainCode3

{

public static bool ValidatePhone(string s)

{

bool res = true;

Regex r = new Regex(@"^([a-zA-Z0-9]{1}[i][x][\\w a-zA-Z0-9]{0,})$");

Match m = r.Match(s);

if (m.Success)

{

res = true;

}

else

{

res = false;

}

return res;

}

}

}-----------------------------------------------------------------------------------------------------------------------------------

109.Perfect Number

Write a program to that takes a positive integer and returns true if the number is perfect number.

A positive integer is called a perfect number if the sum of all its factors (excluding the number itself, i.e., proper divisor) is equal to its value.

For example, the number 6 is perfect because its proper divisors are 1, 2, and 3, and 6=1+2+3; but the number 10 is not perfect because its proper divisors are 1, 2, and 5, and 1+2+5 is not equal to 10

Include a class UserMainCode with a static method getPerfection which accepts the number. The return type is boolean (true / false).

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.

Output consists of TRUE / FALSE.

Refer sample output for formatting specifications.

Sample Input 1:

28

Sample Output 1:

TRUE

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_101\_110

{

class Program\_109

{

static bool getPerfection(int n)

{

int sum = 0;

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

{

if (n%i==0)

{

sum =sum+i;

}

}

if (sum == n)

return true;

else

return false;

}

public static void Main()

{

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

bool output = getPerfection(x);

Console.WriteLine(output);

Console.ReadLine();

}

}

}

---------------------------------------------------------------------------------------------------------------------------------

110.Check Characters in a String

Write a program to read a string and to test whether first and last character are same. The string is said to be be valid if the 1st and last character are the same. Else the string is said to be invalid.

Include a class UserMainCode with a static method checkCharacters which accepts a string as input .

The return type of this method is an int. Output should be 1 if the first character and last character are same . If they are different then return -1 as output.

Create a class Main which would get the input as a string and call the static method checkCharacterspresent in the UserMainCode.

Input and Output Format:

Input consists of a string.

Output is a string saying characters are same or not .

Refer sample output for formatting specifications.

Sample Input 1:

the picture was great

Sample Output 1:

Valid

Sample Input 1:

this

Sample Output 1:

Invalid

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace java\_101\_110

{

class Program\_110

{

static int checkCharacters(string s)

{

if (s[0] == s[s.Length - 1])

return 1;

else

return -1;

}

public static void Main(string[] args)

{

string s1 = Console.ReadLine();

int output =checkCharacters(s1);

if (output == 1)

Console.WriteLine("Valid");

else if (output == -1)

Console.Write("Invalid");

Console.ReadLine();

}

}

}

111.Max Scorer

Write a program that performs the following actions:

1. Read n strings as input and stores them as an arraylist. The string consists of student information like name and obtained marks of three subjects. Eg: name-mark1-mark2-mark3 [suresh-70-47-12] The mark would range between 0 – 100 (inclusive).

2. Write a function highestScorer which accepts these the arraylist and returns the name of the student who has scored the max marks. Assume the result will have only one student with max mark.

Include a class UserMainCode with the static method highestScorer which accepts the arraylist and returns the name (string) of max scorer.

Create a Class Main which would be used to read n strings into arraylist and call the static method present in UserMainCode.

Input and Output Format:

Input consists of 1 integer and n strings. The first integer denotes the size of the arraylist, the next n strings are score pattern described above.

Output consists of a string with the name of the top scorer.

Refer sample output for formatting specifications.

Sample Input 1:

3

sunil-56-88-23

bindul-88-70-10

john-70-49-65

Sample Output 1:

john

class pmg111\_113\_

{

static string highestscore(ArrayList al)

{

string k = "";

Dictionary<int, string> dl = new Dictionary<int, string>();

// int max = 0;

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

{

k = al[i].ToString();

string[] x = k.Split('-');

int a = Convert.ToInt32(x[1]);

int b = Convert.ToInt32(x[2]);

int c = Convert.ToInt32(x[3]);

int sum = a + b + c;

dl.Add(sum, x[0]);

}

int max = dl.Max(t => t.Key);

return dl[max];

}

public static void Main()

{

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

string[] s = new string[n];

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

{

s[i] = Console.ReadLine();

}

ArrayList al = new ArrayList();

al.AddRange(s);

string a = highestscore(al);

Console.WriteLine(a);

Console.ReadLine();

}

}

112.Valid Date

Given a date string as input, write a program to validate if the given date is in any of the following formats:

dd.mm.yyyy

dd/mm/yy

dd-mm-yyyy

Include a class UserMainCode with a static method “validateDate” that accepts a String and returns an integer. This method returns 1 if the date is valid, else return -1.

Create a class Main which would get a String as input and call the static method validateDate present in the UserMainCode.

Input and Output Format:

Input consists of a String.

Output consists of a String that is either 'Valid' or 'Invalid'.

Sample Input 1:

12.03.2012

Sample Output 1:

Valid

Sample Input 2:

27#01#1977

class pmg112\_113\_

{

static int ValidateDate(string st)

{

Regex r1 = new Regex(@"^([0-2][0-9]|[3][0-1]){1,2}([-./\\]){1,1}([0-2][0-9]){1,2}([-./\\]){1,1}([0-9]){4}$");

Match mc = r1.Match(st);

if (mc.Success)

{

return 1;

}

else

return 2;

}

static void Main()

{

string s = Console.ReadLine();

int a = ValidateDate(s);

if (a == 1)

Console.WriteLine("Valid");

else

Console.WriteLine("Invalid");

Console.ReadLine();

}

}

113.Employees & Designations

A Company wants to obtain employees of a particular designation. You have been assigned as the programmer to build this package. You would like to showcase your skills by creating a quick prototype. The prototype consists of the following steps:

Read Employee details from the User. The details would include name and designaton in the given order. The datatype for name and designation is string.

Build a hashmap which contains the name as key and designation as value.

You decide to write a function obtainDesignation which takes the hashmap and designation as input and returns a string array of employee names who belong to that designation as output. Include this function in class UserMainCode.

Create a Class Main which would be used to read employee details in step 1 and build the hashmap. Call the static method present in UserMainCode.

Input and Output Format:

Input consists of employee details. The first number indicates the size of the employees. The next two values indicate the employee name employee designation. The last string would be the designation to be searched.

Output consists of a array values containing employee names.

Refer sample output for formatting specifications.

Sample Input 1:

4

Manish

MGR

Babu

CLK

Rohit

MGR

Viru

PGR

MGR

Sample Output 1:

Manish

Rohit

class Pgm113

{

static string[] obtainDesignation(Hashtable h,string s)

{

string[] s1 = new string[100];

int i = 0;

foreach(string v in h.Keys)

{

if(s==(string)h[v])

{

s1[i] = v;

i++;

}

}

Array.Resize(ref s1, i);

return s1;

}

public static void Main()

{

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

Hashtable h = new Hashtable();

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

{

h.Add(Console.ReadLine(), Console.ReadLine());

}

string s = Console.ReadLine();

string[] s2=obtainDesignation(h, s);

foreach (var v in s2)

Console.WriteLine(v);

Console.Read();

}

}