**Aplicatii suplimentare**

**Functii**

**Problema 1**

using System;

namespace P1

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a subprogramului sub cu doi parametri:

/// n (număr natural, 0<n≤50) şi k (număr natural, 0<k≤20).

/// Subprogramul determină afişarea pe o linie nouă a ecranului,

/// în ordine descrescătoare, a primelor n numere naturale nenule divizibile cu k.

/// Numerele vor fi separate prin câte spaţiu.

/// Exemplu: dacă n=3 şi k=5 la apelul subprogramului se va afişa pe ecran: 15 10 5.

/// </summary>

static void Main(string[] args)

{

int n, k;

Console.WriteLine("Introduceti n: ");

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

Console.WriteLine("Introduceti k: ");

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

sub(n, k);

}

private static void sub(int n, int k)

{

//primele n numere naturale nenule divizibile cu k <=> primii n multipli ai lui k

//acesti multipli pot fi obtinuti prin inmultirea lui k cu 1, 2, ..., n

//pentru a-i putea afisa descrescator, incep inmultirea lui k cu n, n-1, ..., 1

Console.WriteLine("Primele n numere divizibile cu k, afisate descrescator, sunt: ");

for(int i=n;i>=1;i--)

Console.Write(k\*i+" ");

}

}

}

**Problema 2**

using System;

namespace P2

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a subprogramului sub cu trei parametri:

/// n (număr natural, 5<n≤30000), a şi b; subprogramul furnizează prin intermediul

/// parametrilor a şi b cele mai mari două numere prime distincte mai mici decât n.

/// Exemplu: dacă n= 28 la apelul subprogramului se va furniza prin parametrul a valoarea 23 şi prin parametrul b valoarea 19.

/// </summary>

static void Main(string[] args)

{

int n,a=-1,b=-1;

Console.WriteLine("Introduceti n: ");

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

sub(n, ref a, ref b);

Console.WriteLine("Primele doua numere prime mai mici decat n sunt: " + a +" "+ b);

}

private static void sub(int n, ref int a, ref int b)

{

bool prim=false; //prin intrmediul acestei variabile vom retine daca un nr este sau nu prim

n--; //deoarece numerele trebuie sa fie mai mici decat n

if (n % 2 == 0)

n--; //deoarece un numar par sigur nu e prim

for (int i=n;i>0 && prim==false;i-=2) //i merge din 2 in 2, descrescator, pt a sari peste numerele pare

{

prim = true; //presupunem ca i este nr prim, iar daca se va dovedi ca nu este, prim va redeveni false

for (int j = 3; j \* j <= i; j += 2)

if (i % j == 0)

{

prim = false; //daca i se imparte exact la j, inseamna ca nu este prim break; //putem merge mai departe

}

if (prim==true) //daca prim a ramas true inseamna ca nu s-a gasit niciun divizor pt i si se iese din for deoarece prim != false

a = i;

}

prim = false;

for(int i=a-2;i>0 && prim==false; i-=2) //cautam al doilea nr prim incepand cu a-2; comentariile de mai sus raman valabile si aici

{

prim = true;

for (int j = 3; j \* j <= i; j += 2)

if (i % j == 0)

{

prim = false;

break;

}

if (prim == true)

b = i;

}

}

}

}

**Problema** **3**

using System;

namespace P3

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a subprogramului multiplu care are 3 parametri:

/// a, prin care primeşte un tablou unidimensional cu maximum 100 de numere naturale mai mici decât 1000,

/// n, numărul efectiv de elemente ale tabloului şi k, un număr natural (k≤9).

/// Subprogramul returnează numărul de elemente din tablou care sunt multipli ai numărului k şi au ultima cifră egală cu k.

/// Exemplu: dacă n=6, a=(9,273,63,83,93,123), iar k=3, subprogramul va returna valoarea 4.

/// </summary>

static void Main(string[] args)

{

int n, k;

Console.WriteLine("Introduceti n (nr. de elemente): ");

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

int[] a = new int[n];

Console.WriteLine("Introduceti pe o singura linie elementele tabloului: ");

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

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

a[i] = int.Parse(buffer[i]);

Console.WriteLine("Introduceti k: ");

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

Console.WriteLine("Nr de elemente care respecta cerinta este: "+multiplu(a,n,k));

}

private static int multiplu(int[] a, int n, int k)

{

int nrElemente = 0; //stocheaza nr de elemente care respecta cerinta

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

if (a[i] % k == 0 && a[i] % 10 == k) //verifica daca elementul de pe pozitia i este divizibil cu k si are ultima cifra k

nrElemente++;

return nrElemente;

}

}

}

**Problema 4**

using System;

namespace P4

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a subprogramului interval care are doi parametri a şi n,

/// prin care primeşte un tablou unidimensional cu maximum 100 de numere naturale mai mici decât 1000

/// şi respectiv numărul efectiv de elemente din tabloul unidimensional.

/// Subprogramul returnează numărul de elemente din tabloul unidimensional

/// care aparţin intervalului închis determinat de primul şi respectiv ultimul element al tabloului.

/// Exemplu: dacă tabloul are 6 elemente şi este de forma (12,27,6,8,9,2), subprogramul va returna valoarea 5.

/// </summary>

static void Main(string[] args)

{

int n;

Console.WriteLine("Introduceti n: ");

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

int[] a = new int[n];

Console.WriteLine("Introduceti elementele tabloului pe o singura linie: ");

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

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

a[i] = int.Parse(buffer[i]);

Console.WriteLine("Nr de elemente care apartin intervalului este: " + interval(a,n));

}

private static int interval(int[] a, int n)

{

int nr1, nr2; //variabile in care se vor memora capetele tabloului

int nrElemente=0; //variabila in care se va stoca nr de elemente care verifica conditia din cerinta

//verific care dintre cele doua numere din capete este mai mare si stochez in nr1 numarul mai mic, iar in nr2 numarul mai mare

if(a[0]>a[n-1])

{

nr1 = a[n - 1];

nr2 = a[0];

}

else

{

nr1 = a[0];

nr2 = a[n - 1];

}

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

if (a[i] >= nr1 && a[i] <= nr2) //verific apartenenta la interval

nrElemente++;

return nrElemente;

}

}

}

**Problema 5**

using System;

namespace P5

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a subprogramului count care are doi parametri, a şi n,

/// prin care primeşte un tablou unidimensional cu maximum 100 de numere reale

/// şi respectiv numărul efectiv de elemente din tablou. Subprogramul returnează

/// numărul de elemente din tabloul a care sunt mai mari sau cel puţin egale

/// cu media aritmetică a tuturor elementelor din tablou.

/// Exemplu: dacă tabloul are 6 elemente şi este de forma (12, 7.5, 6.5, 3, 8.5, 7.5),

/// subprogramul va returna valoarea 4 (deoarece media tuturor elementelor este 7.5 şi

/// numerele subliniate sunt cel puţin egale cu această medie).6.

/// </summary>

static void Main(string[] args)

{

int n;

Console.WriteLine("Introduceti n: ");

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

float[] a = new float[n];

Console.WriteLine("Introduceti elementele tabloului pe o singura linie: ");

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

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

a[i] = float.Parse(buffer[i]);

Console.WriteLine("Nr de elemente este: "+ count(a,n));

}

private static int count(float[] a, int n)

{

float suma = 0, media; //in suma se va stoca suma elementelor pentru a putea face media lor aritmetica,n in medie se va stoca media

int nrElemente = 0; //in nrElemente se va stoca nr de elemente care verifica consitia

for (int i = 0; i < n; i++) //parcurgem vectorul pentru a face suma elementelor

suma += a[i];

media = suma / n;

for (int i = 0; i < n; i++) //parcurgem vectorul pentru a verifica conditia din cerinta

if (a[i] >= media)

nrElemente++;

return nrElemente;

}

}

}

**Problema 6**

using System;

namespace P6

{

class Program

{

/// <summary>

/// Subprogramul aranjare are doi parametri: a prin care primeşte un tablou unidimensional

/// cu maximum 100 de numere reale nenule şi n, numărul de elemente din tablou.

/// Subprogramul rearanjează elementele tabloului unidimensional astfel încât toate

/// valorile negative să se afle pe primele poziţii, iar valorile pozitive în continuarea celor negative.

/// Ordinea în cadrul secvenţei de elemente pozitive, respectiv în cadrul secvenţei de elemente negative, poate fi oricare.

/// Tabloul modificat va fi furnizat tot prin intermediul parametrului a.

/// Exemplu: dacă tabloul are 6 elemente şi este de forma (12, -7.5, 6.5, -3, -8, 7.5), după apel, acesta ar putea fi: (-7.5, -3, -8, 12, 6.5, 7.5). Scrieţi definiţia completă a subprogramului aranjare

/// </summary>

static void Main(string[] args)

{

int n;

Console.WriteLine("Introduceti n: ");

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

double[] a = new double[n];

Console.WriteLine("Introduceti elementele tabloului pe o singura linie: ");

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

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

a[i] = double.Parse(buffer[i]);

aranjare(ref a, n);

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

Console.Write(a[i]+" ");

}

private static void aranjare(ref double[] a, int n)

{

double temp;

for (int i = 0; i < n - 1; i++) //verifica mai intai primul element cu cele care urmeaza dupa el si, daca e pozitiv, le interschimba, ducand spre capat numerele pozitive; procedeaza la fel cu al doilea element si restul etc.

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

if (a[i] > 0)

{

temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}

}

}

**Problema 7**

using System;

namespace P7

{

class Program

{

/// <summary>

/// Subprogramul nule are doi parametri: a, prin care primeşte un tablou unidimensional

/// cu maximum 100 de numere întregi, cu cel mult 4 cifre fiecare şi n, numărul de elemente din tablou.

/// Subprogramul rearanjează elementele tabloului unidimensional astfel încât toate valorile nule

/// să se afle la sfârşitul tabloului. Ordinea în cadrul secvenţei de elemente nenule poate fi oricare.

/// Tabloul modificat este furnizat tot prin parametrul a. Exemplu: dacă n=6, a=(12,0,0,-3,-8,0),

/// după apel, acesta ar putea fi: a=(12,-3,-8,0,0,0). Scrieţi definiţia completă a subprogramului nule.

/// </summary>

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

static void Main(string[] args)

{

int n;

Console.WriteLine("Introduceti n: ");

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

double[] a = new double[n];

Console.WriteLine("Introduceti elementele tabloului pe o singura linie: ");

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

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

a[i] = double.Parse(buffer[i]);

nule(ref a, n);

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

Console.Write(a[i] + " ");

}

private static void nule(ref double[] a, int n)

{

double temp;

for (int i = 0; i < n - 1; i++) //verifica mai intai primul element cu cele care urmeaza dupa el si, daca e 0, le interschimba, ducand spre capat numerele nule; procedeaza la fel cu al doilea element si restul etc.

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

if (a[i] == 0)

{

temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}

}

}

**Problema 8**

using System;

namespace P8

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a unui subprogram i\_prim care primeşte prin singurul său parametru, n, un număr natural din intervalul [2,30000] şi returnează diferenţa minimă

/// p2-p1 în care p1 şi p2 sunt numere prime şi p1≤n≤p2.

/// Exemplu: dacă n=20 atunci i\_prim(n)=4, valoare obţinută pentru p1=19 şi p2=23. Functii + Program1.

/// </summary>

static void Main(string[] args)

{

int n;

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

Console.WriteLine(i\_prim(n));

}

private static int i\_prim(int n)

{

int p1 = n, p2 = n; //p1 si p2 se initializeaza la n deoarece cele doua numere pot fi si egale cu n

while (prim(p1) != true) //cat timp p1 nu este prim, se face p1-1 pana cand p1 devine prim, deoarece p1<=n

p1--;

while (prim(p2) != true) //cat timp p2 nu este prim, se face p2+1 pana cand p2 devine prim, deoarece p2>=n

p2++;

return p2 - p1;

}

private static bool prim(int nr)

{

if (nr < 2) //numerele mai mici decat 2 nu sunt prime

return false;

if (nr == 2) //2 este singurul numar par prim

return true;

for (int d = 2; d \* d <= nr; d++) //se verifica daca numarul are vreun divizor

if (nr % d == 0)

return false;

return true;

}

}

}

**Problema 9**

using System;

namespace P9

{

class Program

{

/// <summary>

/// Subprogramul sub, cu trei parametri, primeşte prin intermediul parametrilor: v un tablou unidimensional cu cel mult 100 de componente

/// ce memorează numere întregi de cel mult 4 cifre fiecare, n un număr natural nenul mai mic sau egal cu 100 ce reprezintă numărul

/// efectiv de componente ale tabloului primit prin intermediul parametrului v, a un număr întreg cu cel mult 4 cifre.

/// Subprogramul sub returnează numărul componentelor tabloului primit prin intermediul parametrului v ale căror valori sunt

/// strict mai mici decât valoarea parametrului a.Exemplu: pentru valorile n=5, v=(1,21,9,21,403), a=20 ale parametrilor, în urma apelului,

/// subprogramului sub va returna valoarea 2. Scrieţi definiţia completă a subprogramului sub.

/// Să se scrie un program C# care să citească de la tastatură un număr natural nenul n (n≤100) şi n numere întregi,

/// fiecare având cel mult 4 cifre, şi care, folosind apeluri utile ale subprogramului sub, să afişeze pe ecran mesajul DA dacă oricare două dintre

/// cele n numere întregi citite sunt distincte două câte două, sau mesajul NU în caz contrar.

/// Exemplu: pentru n=6 şi cele n numere citite de la tastatură: 47 183 69 8 134 -56 se va afişa pe ecran mesajul DA.

/// </summary>

static void Main(string[] args)

{

int n, s=0;

Console.WriteLine("Introduceti n: ");

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

int[] v = new int[n];

Console.WriteLine("Introduceti elementele tabloului pe o singura linie: ");

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

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

v[i] = int.Parse(buffer[i]);

for (int i = 0; i < n; i++) //parcurge vectorul

s += sub(n, v, v[i]); //in s se aduna rezultatul apelului functiei care verifica cate numere sunt mai mici decat v[i]

if (s==n\*(n-1)/2) //atunci cand numerele sunt distincte, in s se afla suma numerelor de la 0 la n-1 care se calculeaza cu suma lui Gauss, prezenta in if

Console.WriteLine("DA");

else

Console.WriteLine("NU");

}

private static int sub(int n, int[] v, int a)

{

int contor = 0; //numara cate numere verifica conditia din cerinta

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

if (v[i] < a)

contor++;

return contor;

}

}

}

**Problema 10**

using System;

namespace P10

{

class Program

{

/// <summary>

/// Un număr n se numeşte extraprim dacă atât el, cât şi orice număr obţinut prin permutarea cifrelor lui n, sunt numere prime. De exemplu, numărul 113 este un număr extraprim deoarece 113, 311, 131 sunt numere prime. Scrieţi definiţia completă a unui subprogram e\_prim, cu un parametru, subprogram care primeşte prin intermediul parametrului a număr natural cu cel mult 3 cifre (a>1) si returnează suma tuturor exponenţilor din descompunerea în factori primi a valorii parametrului a. Exemplu: pentru a=90 subprogramul va returna valoarea 4, deoarece a=2\*3^2\*5 şi 1+2+1=4. Scrieţi un program C# care citeşte de la tastatură un număr natural n, 2≤n≤999 şi, folosind apeluri utile ale subprogramului e\_prim, verifică dacă n este un număr extraprim. În caz afirmativ, programul afişează pe ecran mesajul DA, în caz contrar afişând mesajul NU.

/// </summary>

static void Main(string[] args)

{

int n;

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

//ca un nr sa fie prim, suma exponentilor din descompunerea in factori primi trebuie sa fie 1 astfel, ca un nr sa fie extraprim, atat suma exponentilor din descompunerea in factori primi a numarului n, cat si a numerelor obtinute din permutarea cifrelor lui n, trebuie sa fie 1

if(n<10) //daca n are o singura cifra, e de ajuns sa verificam daca e\_prim(n) este 1

if(e\_prim(n)==1)

Console.WriteLine("DA");

else

Console.WriteLine("NU");

else if (n>9 && n<100) //daca n are 2 cifre, trebuie sa verificam atat pt n, cat si pentru n inversat

if(e\_prim(n)==1 && e\_prim(n%10\*10+n/10)==1)

Console.WriteLine("DA");

else

Console.WriteLine("NU");

else if (n>99 && n<1000) //daca n are 3 cifre, trebuie sa verificam atat pt n, cat si pt permutarile lui n

{//pt a nu ne incurca, iau 3 variabile in care sa retin fiecare cifra a lui n, astfel putand sa verific mai usor permutarile

int a, b, c;

a = n / 100;

b = n / 10 % 10;

c = n % 10;

//reformam numerele in functie de a,b,c a.i. sa obtinem permutarile lui n

if(e\_prim(n)==1 && e\_prim(a\*100+c\*10+b)==1 && e\_prim(b\*100+a\*10+c)==1 && e\_prim(b\*100+c\*10+a)==1 && e\_prim(c\*100+a\*10+b)==1 && e\_prim(c\*100+b\*10+a)==1)

Console.WriteLine("DA");

else

Console.WriteLine("NU");

}

}

private static int e\_prim(int n)

{

int p, d = 2, sum=0; //in p se va tine minte puterea la care se afla factorul prim in descompunerea lui n, d este factorul prim cu care se va incerca divizarea lui n, iar sum este variabila in care se memoreaza suma puterilor factorilor primi din descompunerea lui n

while (n!=1)

{

p = 0;

while (n%d==0) //cat timp n se divide cu n, se face aceasta divizare si se aduna 1 la p, adica puterea factorului d creste cu 1

{

n /= d;

p++;

}

if (p > 0)

sum += p;

d++;

}

return sum;

}

}

}

**Problema 11 + Problema 12**

using System;

namespace P11

{

class Program

{

/// <summary>

/// Subprogramul cif, cu doi parametri, primeşte prin intermediul parametrului a un număr natural cu cel mult 8 cifre şi prin intermediul parametrului b o cifră; subprogramul returnează numărul de apariţii ale cifrei b în scrierea numărului a. Exemplu: pentru a=125854 şi b=5, subprogramul va returna valoarea 2. Scrieţi definiţia completă a subprogramului cif.Scrieţi un program C# care citeşte de la tastatură un număr natural n cu exact 8 cifre, fiecare cifră fiind nenulă, şi care determină şi afişează pe ecran, folosind apeluri utile ale subprogramului cif, cel mai mic număr palindrom ce poate fi obţinut prin rearanjarea tuturor cifrelor numărului n. Dacă nu se poate obţine un palindrom din toate cifrele numărului n, programul va afişa pe ecran numărul 0. Un număr natural este palindrom dacă este egal cu numărul obţinut prin scrierea cifrelor sale în ordine inversă. Exemplu: dacă n=21523531 atunci se va afişa pe ecran numărul 12355321, iar dacă n=12272351 atunci se va afişa pe ecran numărul 0.

/// </summary>

static void Main(string[] args)

{

int n;

Console.WriteLine("Introduceti n: ");

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

//intr-un palindrom de 8 cifre, cifrele numarului au un nr par de aparitii; astfel, vom folosi un vector de frecventa care sa numere aparitiile cifrelor

int[] frecv = new int[10];

bool ok = true; //variabila care va deveni false in caz ca frecventa unei cifre va fi impara

for(int i=0;i<10;i++)

{

frecv[i] = cif(n, i);

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

{

Console.WriteLine(0);

ok = false;

break;

}

}

if (ok==true)

{

int[] nr = new int[8]; //in acest vector formez noul numar despre care se stie ca are 8 cifre din cerinta

int poz = 0;

for(int i=0;i<10;i++)

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

{

while(frecv[i]!=0)

{

nr[poz] = nr[7 - poz] = i; //pt poz=0, la aceasta instructiune, prima si ultima cifra devin egale cu i; pt poz=1, a doua si penultima cifra devin egale cu i

poz++;

frecv[i] -= 2;

}

}

for(int i=0;i<8;i++)

Console.Write(nr[i]);

}

}

private static int cif(int a, int b)

{

int contor = 0; //variabila in care se va retine nr de aparitii a lui b in a

while(a!=0)

{

if (a % 10 == b) //daca ultima cifra a lui a este b, se incrementeaza contor cu 1

contor++;

a /= 10; //se taie ultima cifra din a, fiind deja verificata

}

return contor;

}

}

}

**Problema 13**

using System;

namespace P13

{

class Program

{/// <summary>

/// Subprogramul f, cu un parametru primeşte prin intermediul parametrului a un număr natural cu cel mult 8 cifre (a>1) si returnează cel mai mic divizor prim al valorii parametrului a.Exemplu: pentru valoarea 45 a parametrului a, subprogramul va returna valoarea 3 deoarece a=32\*5, iar cel mai mic divizor prim al său este 3. Scrieţi definiţia completă a subprogramului f. Scrieţi un program C# care să citească de la tastatură un număr natural nenul n (n≤100) şi apoi un şir de n numere naturale de cel mult 8 cifre fiecare, toate numerele din şir fiind strict mai mari decât 1. Folosind apeluri utile ale subprogramului f, programul va determina va afişa pe ecran toate numerele prime din şirul citit. Numerele determinate se vor afişa pe ecran, separate prin câte un spaţiu, în ordine crescătoare a valorii lor. Dacă nu există astfel de numere se va afişa pe ecran mesajul NU EXISTA. Exemplu: pentru n=7, şirul: 1125, 2, 314, 101, 37, 225, 12 pe ecran se va afişa: 2 37 101.

/// </summary>

static void Main(string[] args)

{

int n, nr;

Console.WriteLine("Introduceti n: ");

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

int[] v = new int[n]; //declaram un vector in care se vor stoca numerele prime

int contor = 0;//numaram cate numere prime am gasit si punem pe pozitia contor din v numarul prim gasit

Console.WriteLine("Introduceti pe o singura linie numerele: ");

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

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

{

nr = int.Parse(buffer[i]);

if(f(nr)==nr) //daca functia returneaza acelasi nr inseamna ca nu a gasit divizori pt nr, deci nr este prim

{

v[contor] = nr;

contor++;

}

}

//sortam vectorul cu valori prime

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

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

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

{

int aux = v[i];

v[i] = v[j];

v[j] = aux;

}

//afisam numerele prime daca exista

if (contor!=0)

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

Console.Write(v[i]+" ");

else

Console.WriteLine("NU EXISTA");

}

private static int f(int n)

{

for (int i = 2; i <= n / 2; i++) //cauta, de la 2, un divizor prim pentru n; daca nu gaseste, returneaza n

if (n % i == 0)

return i;

return n;

}

}

}

**Problemele 14, 15, 16 sunt sunt enuntate si ca Problema 9, 11/12, 13: Problema 14 = Problema 9,**

**Problema 15 = Problema 11/12, Problema 16 = Problema 13**

**Problema 17**

using System;

namespace P17

{

class Program

{/// <summary>

///Scrieţi definitia funcţiei sum care primeşte ca parametru un număr natural nenul x cu maximum 9 cifre şi returnează suma divizorilor

///numărului x.Exemplu: sum(6) are valoarea 12 (=1+2+3+6). Să se scrie un program C# care citeşte de la tastatură un număr natural n (0<n<25)

///şi apoi n numere naturale nenule cu maximum 9 cifre fiecare. Programul calculează, folosind apeluri ale funcţiei sum,

///şi afişează pe ecran câte numere prime conţine şirul citit. Exemplu: pentru n=5 şi valorile 12 3 9 7 1 se va afişa pe ecran valoarea 2

///(în şirul dat există două numere prime şi anume 3 şi 7).

/// </summary>

static void Main(string[] args)

{

int n;

Console.WriteLine("Introduceti n: ");

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

int contor = 0; //numara numerele prime

Console.WriteLine("Introduceti sirul pe o singura linie: ");

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

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

{

int nr = int.Parse(buffer[i]);

if (sum(nr) == 1 + nr) //un numar este prim daca suma divizorilor sai este egala cu 1+nr, deoarece divizorii unui numar prim sunt 1 si el insusi

contor++;

}

Console.WriteLine(contor);

}

private static int sum(int n)

{

if (n == 1)//suma divizorilor lui 1 este 1

return 1;

int suma = 1+n; //se va memora suma divizorilor; primul divizor al unui nr este 1, iar ultimul este mereu el insusi

for (int i = 2; i <= n / 2; i++)

if (n % i == 0) //daca n se divide cu i, adaugam i la suma

suma += i;

return suma;

}

}

}

**Problema 18**

using System;

using System.IO;

namespace P18

{

class Program

{/// <summary>

///Se consideră subprogramul P care are doi parametri:

/// - n, prin intermediul căruia primeşte un număr natural de cel mult 9 cifre

/// - c, prin intermediul căruia primeşte o cifră. Subprogramul va furniza tot prin intermediul parametrului n numărul

/// obţinut din n prin eliminarea tuturor apriţiilor cifrei c. Dacă, după eliminare, numărul nu mai conţine nicio cifră sau

/// conţine doar cifre 0, rezultatul returnat va fi 0. Pe prima linie a fişierului text BAC.IN se găsesc, separate prin câte

/// un spaţiu, mai multe numere naturale de cel mult 9 cifre fiecare. Scrieţi programul C# care citeşte numerele din acest fişier,

/// utilizând apeluri ale subprogramului P elimină toate cifrele impare din fiecare dintre aceste numere şi apoi scrie în fişierul text BAC.OUT

/// numerele astfel obţinute, separate prin câte un spaţiu. Dacă un număr din fişierul BAC.IN nu conţine nicio cifră pară nenulă,

/// acesta nu va mai apărea deloc în fişierul de ieşire.

/// Exemplu: dacă fişierul BAC.IN conţine numerele 25 7 38 1030 45127 0 35 60 15, atunci BAC.OUT va avea conţinutul: 2 8 42 60.

/// </summary>

static void Main(string[] args)

{

int nr;

StreamReader sr = new StreamReader(@"..\..\BAC.IN");

StreamWriter sw = new StreamWriter(@"..\..\BAC.OUT");

string[] buffer = sr.ReadLine().Split(' ');

for(int i=0;i<buffer.Length;i++) //se parcurge prima linie citita din fisier

{

nr = int.Parse(buffer[i]); //in nr se retine numarul de pe pozitia i din fisier

for (int j = 1; j <= 9; j += 2) //cu acest for se parcurg cifrele impare

P(ref nr, j); //apeleaza subprogramul pentru nr si fiecare cifra impara

if (nr != 0) //daca nr este diferit de 0, il afiseaza

sw.Write(nr + " ");

}

sr.Close();

sw.Close();

}

private static void P(ref int n, int c)

{

int a=0, p=1;//in a se va retine numarul nou, iar p se va folosi pentru a pastra ordinea cifrelor

while (n!=0)

{

if (n % 10 != c) //daca ultima cifra a lui n este diferita de c o adauga la a

{

a += n % 10 \* p;

p \*= 10;

}

n /= 10;

}

n = a;

}

}

}

**Problema 19**

using System;

namespace P19

{

class Program

{

/// <summary>

/// Scrieţi definiţia completă a unui subprogram, nz, cu un parametru întreg n (0<n≤32000), care returnează

/// numărul zerourilor de la sfârşitul numărului n!. Scrieţi programul C# care citește de la tastatură un număr

/// natural k (0<k≤3) și determină, folosind apeluri ale subprogramului nz, cel mai mic număr natural n pentru care n!

/// are cel puţin k zerouri la sfârşit. Numărul determinat se afișează pe ecran.

/// </summary>

static void Main(string[] args)

{

int k;

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

bool ok = false; //variabila care ne va spune cand se va gasi numarul cautat

for(int i=0;ok==false;i++)

if(nz(i)==k) //verificam conditia din cerinta pana cand ne convine rezultatul

{

Console.WriteLine(i);

ok = true;

}

}

private static int nz(int n)

{

int s = 0, p = 5; //in s se va retine numarul de zerouri, iar la p se va imparti numarul pentru a-i afla nr de zerouri

while(p<=n)

{

s += n / p;

p \*= 5;

}

return s;

}

}

}

**Vectori**

**Problema 1**

using System;

using System.IO;

namespace P1

{

class Program

{

/// <summary>

/// Să se formeze vectorul primelor n, unde n este un număr natural dat, elemente ale şirului lui Fibonacci:

/// 0, 1, 1, 2, 3, 5, 8, 13, 21,.... (f[1]=0, f[2]=1, f[i]=f[i-1]+f[i-2]) și să se afișeze în fișierul fibonacci.out

/// </summary>

static void Main(string[] args)

{

StreamWriter sw = new StreamWriter(@"..\..\fibonacci.out");

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

int[] f = new int[n];

int nr1 = 0, nr2 = 1;

if (n == 1) //daca n este 1, vectorul il va contine doar pe 0, acesta fiind primul element al sirului lui Fibonacci

f[0] = nr1;

else if(n==2) //daca n este 2, vectorul va contine 0 si 1, fiind primele 2 elemente ale sirului

{

f[0] = nr1;

f[1] = nr2;

}

else //daca n este mai mare decat 2, incepand cu elementul al 3-lea, elementele se vor forma prin adunarea celor 2 numere precedente

{

f[0] = nr1;

f[1] = nr2;

int nr3;

for(int i=2;i<n;i++)

{

nr3 = nr1 + nr2;

f[i] = nr3;

nr1 = nr2;

nr2 = nr3;

}

}

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

sw.Write(f[i] + " ");

sw.Close();

}

}

}

**Problema 2**

using System;

using System.IO;

namespace P2

{

class Program

{

/// <summary>

/// Fișierul date.in conține numere naturale mai mici decât 100. Se cere să se afișeze, în ordine crescătoare elementele aflate

/// în fișier. Indicație: se va crea un tablou, inițializat cu zero, care “numără” elementele din fișier. Astfel, fiecare număr din

/// fișier este considerat indice al tabloului și, de fiecare data când este extrase, valoarea din tablou este incrementată (v[x]=v[x]+1)

/// </summary>

static void Main(string[] args)

{

StreamReader sr = new StreamReader(@"..\..\date.in");

string[] buffer = sr.ReadLine().Split(' ');

int lungime = buffer.Length; //lungimea vectorului este egala cu lungimea stringului citit de pe prima linie

int[] v\_initial = new int[lungime];

int maxim = -1; //avem nevoie sa gasim maximul pentru a stabili lungimea vectorului ce trebuie creat

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

{

v\_initial[i] = int.Parse(buffer[i]);

if (v\_initial[i] > maxim)

maxim = v\_initial[i];

}

int[] v = new int[maxim + 1];

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

v[v\_initial[i]]++; //numaram aparitiile numerelor din vectorul initial

for(int i=0;i<maxim+1;i++)

if(v[i]!=0)

while (v[i]!=0) //pentru numerele care apar de mai multe ori, le vom afisa de cate ori apar

{

Console.Write(i+" ");

v[i]--;

}

}

}

}

**Metode de sortare**

**Problema 1**

using System;

using System.IO;

namespace P1

{

class Program

{

/// <summary>

/// Fişierul text NR.TXT conţine pe o singură linie, separate prin câte un singur spaţiu, cel mult 100 de numere naturale,

/// fiecare număr având cel mult 4 cifre. Scrieţi un program C# care citeşte toate numerele din fişierul NR.TXT şi afişează pe ecran,

/// separate prin câte un spaţiu, în ordine crescătoare, toate numerele din fişier care au cel puţin 3 cifre.

/// Dacă fişierul nu conţine astfel de numere se va afişa pe ecran mesajul NU EXISTA.

/// </summary>

static void Main(string[] args)

{

StreamReader sr = new StreamReader(@"..\..\NR.txt");

string[] buffer = sr.ReadLine().Split(' ');

int lungime = buffer.Length; //numarul de numere aflate in fisier

int[] v = new int[lungime]; //creez un vector de aceeasi lungime ca si numarul numerelor din fisier, presupunand ca toate au cel putin 3 cifre

int contor = 0;//numar numerele care verifica conditia

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

{

int nr = int.Parse(buffer[i]);

if(nr>99)

{

v[contor] = nr; //punem in v numerele care verifica conditia

contor++;

}

}

if (contor!=0) //daca contor!=0 inseamna ca avem numere care sa verifice conditia si sortam crescator vectorul

{

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

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

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

{

int aux = v[i];

v[i] = v[j];

v[j] = aux;

}

for(int i=0;i<contor;i++) //afisam numerele

Console.Write(v[i]+" ");

}

else

Console.WriteLine("NU EXISTA");

}

}

}

**Problema 2**

using System;

using System.IO;

namespace P2

{

class Program

{

/// <summary>

/// Fişierul text NR.TXT conţine pe o singură linie, separate prin câte un singur spaţiu, cel mult 100 de numere naturale,

/// fiecare număr având cel mult 4 cifre. Scrieţi un program C# care citeşte numerele din fişierul NR.TXT şi afişează pe ecran,

/// separate prin câte un spaţiu, în ordine descrescătoare, toate numerele din fişier care au cel mult 2 cifre. Dacă fişierul nu conţine

/// astfel de numere se va afişa pe ecran mesajul NU EXISTA.

/// </summary>

static void Main(string[] args)

{

StreamReader sr = new StreamReader(@"..\..\NR.txt");

string[] buffer = sr.ReadLine().Split(' ');

int lungime = buffer.Length; //numarul de numere aflate in fisier

int[] v = new int[lungime]; //creez un vector de aceeasi lungime ca si numarul numerelor din fisier, presupunand ca toate au cel mult 2 cifre

int contor = 0;//numar numerele care verifica conditia

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

{

int nr = int.Parse(buffer[i]);

if (nr < 100)

{

v[contor] = nr; //punem in v numerele care verifica conditia

contor++;

}

}

if (contor != 0) //daca contor!=0 inseamna ca avem numere care sa verifice conditia si sortam crescator vectorul

{

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

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

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

{

int aux = v[i];

v[i] = v[j];

v[j] = aux;

}

for (int i = 0; i < contor; i++) //afisam numerele

Console.Write(v[i] + " ");

}

else

Console.WriteLine("NU EXISTA");

}

}

}

**Problema 3**

using System;

using System.IO;

namespace P3

{

class Program

{

/// <summary>

/// Se consideră fişierul BAC.TXT ce conţine cel mult un milion de numere naturale separate prin spatii, fiecare număr având

/// cel mult nouă cifre. Scrieţi un program C# care citeşte toate numerele din fişierul BAC.TXT şi determină, folosind un algoritm

/// eficient din punct de vedere timpului de executare, cele mai mari două numere de trei cifre care nu se află în fişier.

/// Cele două numere vor fi afişate pe ecran în ordine descrescătoare, cu un spaţiu între ele. Dacă nu pot fi determinate două astfel

/// de numere, programul va afişa pe ecran valoarea 0.

/// Exemplu: dacă fişierul BAC.TXT conţine numerele: 12 2345 123 67 989 6 999 123 67 989 999, atunci programul va afişa 998 997

/// </summary>

static void Main(string[] args)

{

StreamReader sr = new StreamReader(@"C:\Users\CRISTI\Desktop\AplicatiiCarte\MetodeSortare\P3\BAC.txt");

int[] v = new int[1000]; //vector de frecventa pt numerele de 3 cifre din fisier

string[] buffer = sr.ReadLine().Split(' ');

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

{

int nr = int.Parse(buffer[i]);

if(nr>99 && nr<1000) //daca nr are 3 cifre, v[nr] creste cu 1

v[nr]++;

}

int contor = 0; //variabila menita sa numere cate numere se afiseaza, iar cand/daca ajunge la 2 programul se va opri

for(int i=999;i>=100 && contor<2;i--) //parcurge vectorul de la 999 pana la 100, cat timp contor<2, adica numerele nu au fost inca determinate

if(v[i]==0 && contor==0) //daca v[i] si contor sunt 0, inseamna ca i nu apartine numerelor din fisier si este primul numar care verifica conditia

{

Console.Write(i+" ");

contor++;

}

else if(v[i]==0 && contor==1) //daca v[i] este 0 si contor este 1, inseamna ca i nu apartine numerelor din fisier si este al doilea numar care verifica conditia

{

Console.Write(i+" ");

contor++;

}

if(contor==0)

Console.WriteLine(0);

}

}

}

**Problema 4**

using System;

using System.IO;

namespace P4

{

class Program

{

/// <summary>

/// În fişierul numere.txt sunt memorate maximum 10000 de numere naturale cu cel mult 9 cifre fiecare.

/// Fiecare linie a fişierului conţine câte un număr. Se cere afişarea pe ecran, în ordine descrescrescătoare, a tuturor cifrelor

/// care apar în numerele din fişier. Alegeţi un algoritm de rezolvare eficient din punct de vedere al timpului de executare.

/// Exemplu: dacă fişierul numere.txt conţine:267 39628 79, se va tipări 9987766322.

/// </summary>

static void Main(string[] args)

{

int[] cif = new int[10];//vector de frecventa pt cifrele numerelor din fisier

foreach (string line in File.ReadLines(@"..\..\numere.txt"))

{

int nr = int.Parse(line);

while (nr!=0)

{

cif[nr % 10]++; //stocheaza frecventa aparitiei cifrelor din nr, unde nr este numarul citit din fisier

nr /= 10;

}

}

for(int i=9;i>=0;i--)

if(cif[i]!=0)

while(cif[i]!=0) //pentru a afisa fiecare cifra de cate ori apare

{

Console.Write(i);

cif[i]--;

}

}

}

}

**Problema 5**

using System;

using System.IO;

namespace P5

{

class Program

{/// <summary>

/// În fişierul numere.txt pe prima linie este memorat un număr natural n (n≤10000), iar pe linia următoare un şir de n numere naturale

/// distincte două câte două, separate prin câte un spaţiu, cu maximum 4 cifre fiecare. Se cere afişarea pe ecran a poziţiei pe care s-ar

/// găsi primul element din şirul aflat pe linia a doua a fişierului, în cazul în care şirul ar fi ordonat crescător.

/// Numerotarea poziţiilor elementelor în cadrul şirului este de la 1 la n. Alegeţi un algoritm de rezolvare eficient din punct de vedere

/// al memoriei utilizate şi al timpului de executare. Exemplu: dacă fişierul numere.txt conţine: 6 267 13 45 628 7 79, se va afişa 5,

/// deoarece primul element din şirul iniţial, 267, s-ar găsi pe poziţia a cincea în şirul ordonat crescător (7 13 45 79 267 628).

/// </summary>

static void Main(string[] args)

{

StreamReader sr = new StreamReader(@"..\..\Numere.txt");

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

int[] v = new int[n];

string[] buffer = sr.ReadLine().Split(' ');

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

v[i] = int.Parse(buffer[i]);

int nr1 = v[0]; //retin primul numar din sir

//sortez crescator sirul

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

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

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

{

int aux = v[i];

v[i] = v[j];

v[j] = aux;

}

//caut nr1 in vectorul ordonat si afisez pozitia la care a fost gasit

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

if(v[i]==nr1)

{

Console.WriteLine(i+1); //adaug 1 la i pentru a pastra numerotarea de la 1 la n

break;

}

}

}

}