**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**

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.

/// </summary>

static void Main(string[] args)

{

int a,b;

Console.WriteLine("Introduceti a: ");

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

Console.WriteLine("Introduceti b: ");

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

Console.WriteLine(cif(a,b));

}

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 12**