

## Theorie

Instruktionen se import in 2 Kategorien:

- Instruktionen condit/ode: if, else, switch
- u - u repetitive strukturen (for, while, do while),

Struktur Programmier:

#include < stdio.h >

int main( )

{

// code block or some arch.

return 0;

}



## Un program în C - structura:

structura preprocesor

declaratii de variabile globale

->  $\downarrow$  defunctie  
 main() -> defunctie utilizator  
 { -> interbel functiei principale  
 } -> main() -> n  
 declaratii de variabile -> variabile

Functii

Instrucții

y

def. de functii

-> ale limbajului C  
 definiri functiei principale

-> defunctie utilizator

- > nu avem variabile declarate implicit, de aceea trebuie declarate variabilele care vor fi folosite în program. Acestea sunt în RAY. Right-side
- > datele codificate în formă binară (0, 1)

Variabile & constante

Date numerice intreg (int)  
real - float (mantisa precizare), double  
de tip caracter (simboluri) - char (litera, cifre sau simboluri)

Functii pt. introducere datelor (putchar(), getch(), getche(), gets(), scortf(),  
 -> extogere -> (read); putch(), puts(), printf())

Instrucții simple compuse, de adunare (for, while, do-while), de decizie (if), de multe linii (switch), continu, goto, else, switch).

If -> If (expresie)  
 instrucții

if -> se utilizează ca o condiție

Le următoare instrucții If se execută numai dacă expresia din paranteze (expresie) este adevarată (true), se execută instrucțiunea continută în instrucție, care urmează după if. Dacă expresie = 0 (false), nu se execută niciunul din continutul instrucției, care urmează după if.

if      if (expresie)  
 else     instrucții  
 else     :  
 :        n -> 1, 2

le fel. Dacă expresie ≠ 0 (adesea să se execute instrucția 1), se execută expresie = 0 (false), se execută instrucția 2 și se continuă.

Instrucții de decizie multiple -> instrucții de decizie în cadrul

if (expresie-1)  
 instrucții  
 else if (expresie2)  
 :  
 :  
 else if (expresie-n-1)  
 instrucții

Le următor se evaluatează expresia 1. Dacă este adevarată (true), se execută instrucțiile 1, și dacă este falsă se trece la evaluarea expresiei 2. Dacă expresie 2 este adevarată, se execută instrucțiile 2, și dacă este falsă se trece la evaluarea expresiei 3. Dacă nu există nicio expresie mai multă, se execută instrucțiile.



Instrucție de cicle (for, while, do-while) - executare în mod repetat a unei comenzi

For - pt. for(expresie1; expresie2; expresie3)  
                  instructione

expresie 1 - se evaluează și apoi datează în tristăț. For, pentru  
primul iterativ = se initializează variabilele condusă de cicle  
expresie 2 - testul cu care se controlează execuția. Se evaluează de fiecare  
dată înaintea fiecărui iterativ.

— n — 3 - expresie prin care se modifică valoarea variabilei condusă de cicle  
Se evaluează de fiecare dată după executarea instrucției, ca formă  
completă cicleului (instructione). E o expresie de lucru (H) sau  
decrementează (—) a valorii variabilei condusă de cicle

La initializare lini for, se evaluează expresie 1; se evaluează expresie 2. Dacă valoarea  
el este nemulțumită (adesea 0) se execută buclă și în continuare se evaluează  
expresie 3, după ce se revine evaluarea expresiei 2. Procedoul următor este  
timp expresiei 1 este ≠ 0 (adesea 0). Dacă val. expresiei 2 = 0 (fals) se pornește  
timp expresiei 1 este ≠ 0 (adesea 0). Dacă val. expresiei 2 ≠ 0 (în cazul cind cicleul for  
ciclel, executare programul continuând cu lista care urmează după ciclel for



Examination of the following names

1. Produtividade negativa: Dofrida, droga de uso recreativo.
  2. Vectais com 3 medidumentos: Definitiva, catadentica.
  3. Se cíclope é um dia de desabafos para quem de fato gosta

am sin. 5) de determinante este ultima mici (a, s, -z etc) nu sunt am



Problema 2 Se citește un str de la tastatura pînă se întâlnește ".!"

```
#include <stdio.h>
#include <string.h>
int main()
{
    char[1000] str;
    printf("Introdu un str de caractere: ");
    fgets(str, sizeof(str), stdin);
    char[1000] vector;
    for(int i=0; i<strlen(str); i++)
    {
        if(str[i] ? = ".!")
            vector[i]=str[i];
        else
        {
            break;
        }
    }
    printf("%s", vector);
    return 0;
}
```



```

#include <stdio.h>
#include <string.h> → doar folosește metoda strlen

int main()
{
    char nr_introduse[1000];
    char txt_fnl[1000];

    printf("Introdu un sir de caractere: ");
    fgets(nr_introduse, sizeof(nr_introduse), stdin);

    for (int i = 0; i < strlen(nr_introduse); i++)
    {
        if (nr_introduse[i] == '.' || nr_introduse[i] == ',' || nr_introduse[i] == ':' || nr_introduse[i] == ';' || nr_introduse[i] == '!' || nr_introduse[i] == '?') {
            txt_fnl[i] = ' ';
        }
        else
        {
            txt_fnl[i] = nr_introduse[i];
        }
    }

    txt_fnl[strlen(nr_introduse) - 1] = '\0'; // Elimină caracterul newline adăugat de fgets

    FILE *pointerfisier;
    pointerfisier = fopen("Exercitiul1.txt", "w");

    if (pointerfisier == NULL)
    {
        printf("Eroare la deschiderea fisierului.\n");
    }
}

```

```
    return 1;
}

Else
{
    fputs(txt_fnl, pointerfisier);
    printf("gata!\n");
}

// Închide calea către fisier
fclose(pointerfisier);

return 0;
}
```

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

/* am adaugat si biblioteca ctype pentru ca in ea gasim functiile isupper, islower si isdigit. Ca noi sa nu scriem conditii pentru fiecare litera a alfabetului exista aceste metode. De asemenea pentru numere exista isdigit. Daca nu ar fii existat aceste metode, ar fi trebuit sa scriem cel putin 50 de conditii. Adica pentru fiecare majuscule, fiecare minuscula si fiecare numar*/
int main()
{
    // acum citesc din fisier si copies continutul in variabila txt_citit
    FILE *pointer;
    pointer = fopen("Exercitiul1.txt", "r");
    char txt_citit[1000];
    //Daca fisierul e gol primim eroare si mesaj
    if (pointer == NULL)
    {
        printf("Nu s-a gasit fisierul\n");
        return 1;
    }
    //urmatoarea linie copiaza textul din fisier in variabila txt_citit, dupa cum spuneam sim ai sus
    fgets(txt_citit, sizeof(txt_citit), pointer);
    //am inchis conexiunea la fisier
    fclose(pointer);
    //declar trei variabile care sa memorize numarul de litere mari, mici si numere
    int majuscule = 0, minuscule = 0, numere = 0;
    int lungime = strlen(txt_citit);
    /*pe linia anterioara am stocat in variabila numita lungime, dimensiunea vectorului de char: txt_fnl pentru ca este conditia de oprire a forului*/
    for (int i = 0; i < lungime; i++)
    {
        if (isupper(txt_citit[i]))
    }
```

```
    majuscule++;
}

else if (islower(txt_citit[i]))
{
    minuscule++;
}

else if (isdigit(txt_citit[i]))
{
    numere++;
}

/*
In for se verifica fiecare pozitie a vectorului txt_fnl si daca intruneste una din cele trei, atunci se
incrementeaza cu 1 variabilele care Numara caracterele*/
printf("Majuscule: %d\n", majuscule);
printf("Minuscule: %d\n", minuscule);
printf("Numere: %d\n", numere);

//am dat mesaje cu variabilele si gata problema numarul 2

return 0;
}
```

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

char Met_vocala_ret(char consoana)
{
    int variabilalocala = 0;
    char vocala;

    if (isalpha(consoana) && isupper(consoana))
    {
        variabilalocala++;
        consoana = tolower(consoana);
    }

    switch (consoana)
    {
        case 'b':
        case 'c':
            vocala = 'a';
            break;
        case 'd':
        case 'f':
        case 'g':
            vocala = 'e';
            break;
        case 'h':
        case 'j':
        case 'k':
        case 'l':
            vocala = 'i';
    }
}
```

```
        break;

    case 'm':
    case 'n':
    case 'p':
    case 'q':
    case 'r':
        vocala = 'o';
        break;

    case 's':
    case 't':
    case 'v':
    case 'w':
    case 'x':
    case 'y':
    case 'z':
        vocala = 'u';
        break;

    default:
        vocala = consoana;
    }

if (variabilalocala == 1)
{
    vocala = toupper(vocala);
}

else
{
    return vocala;
}

}
```

```
int main()
{
    char txt_citit[1000];
    char txt_fnl[1000];
    FILE *pointer = fopen("Exercitiul1.txt", "r");

    if (pointer == NULL)
    {
        printf("Nu s-a gasit fisierul\n");
        return 1;
    }

    else
    {
        fgets(txt_citit, sizeof(txt_citit), pointer);
    }

    fclose(pointer);

    for (int i = 0; i < strlen(txt_citit); i++)
    {
        txt_fnl[i] = Met_vocala_ret(txt_citit[i]);
    }

    FILE *pointer2 = fopen("Exercitiul1.txt", "w");

    if (pointer2 == NULL)
    {
        printf("Nu s-a gasit fisierul\n");
    }

    else
    {
```

```
fputs(txt_fnl, pointer2);

}

fclose(pointer2);

printf("S-a terminat intreg procesul!\n");

return 0;

}
```

```

#include <stdio.h>
#include <string.h>

int main()
{
    char nr[2];
    char text[100];
    printf("Introdu un numar cuprins intre 0 si 9 inclusiv: ");
    fgets(nr, sizeof(nr), stdin);

/* Dupa ce am afisat mesajul: „Introdu un numar cuprins intre 0 si 9 inclusiv”, cu ajutorul lui fgets
preluam datele introduse de utilizator si le stocam in variabila nr in masura dimensiunii ei de un
caracter*/
    if (nr[0] < '0' || nr[0] > '9')
    {
        printf("Te rog introdu un numar valid intre 0 si 9.\n");
        return 1;
    }

/* Iful anterior spune ca daca numarul introdus de utilizator si stocat in variabila nr este mai mic de 0
si mai mare decat 9 se va executa blocul de cod din accolade, adica se va afisa mesaj si vom primii si o
eroare*/
    printf("Introdu un sir de numere intregi: ");
    fgets(text, sizeof(text), stdin);

/* Am stocat in variabila text numerele introduse de la tastatura*/
    int nr_mai_mici = 0;

    for (int i = 0; i < strlen(text); i++)
    {
        if (text[i] >= '0' && text[i] <= '9')
        {
            if (text[i] - '0' < litera[0] - '0')
                nr_mai_mici++;
        }
    }
}

```

```
    }  
}  
}
```

*/\* In for avem pe i care pleaca de la 0 si creste cu cate 1 pana ajunge la numarul maxim de pozitii al variabilei text. Aici gasim un if care spune ca daca pozitia din vector contine un numar >= 0 si acelasi numar <=9 atunci se poate executa urmatorul bloc de cod aflat intre accolade. In cazul in care conditia este indeplinita, apare un alt if care spune ca daca aceeasi pozitie a vectorului text, anterior verificata daca e cuprinsa intre 0 si 9, daca este mai mica decat variabila nr. Daca da, atunci variabila nr\_mai\_mici se va incrementa, ea memorand cate numere mai mici decat nr au fost intalnite\*/*

```
printf("Din numerele introduse, %d sunt mai mici ca %c\n", nr_mai_mici, litera[0]);  
return 0;  
}  
//mesaj cu rezultatul si apoi inchiderea programului
```

```
#include <stdio.h>
#include <string.h>

int main()
{
    char text[100];

    printf("Introdu un sir de numere intregi: ");
    fgets(text, sizeof(text), stdin);

    /* Utilizatorului i se cere sa introduca un sir de numere intregi si apoi cu ajutorul lui fgets variabila text va stoca caracterele in limita dimensiunii sale*/

    int nr_pare= 0;
    int nr_impare=0;

    for (int i = 0; i < strlen(text); i++)
    {
        if (text[i] >= '0' && text[i] <= '9')
        {
            if ((text[i] - '0') %2 ==0)
            {
                nr_pare++;
            }
            else
            {
                nr_impare++;
            }
        }
    }

    /*In for este verificata conditia care spune ca pozitia din vectorul text este cuprinsa intre 0 si 9. Daca da, se executa blocul de cod. Intalnim alt if unde spune ca daca aceeasi pozitie a vectorului se
```

imparte exact la 2 atunci se va incrementa variabila nr\_pare, dar daca nu se va incrementa variabila nr\_impare\*/

```
printf("Din numerele introduse, %d sunt pare si%d sunt impare \n", nr_pare, nr_impare);

return 0;
}

//mesaj si inchidere program
```

```

#include <stdio.h>
#include <string.h>

int main()
{
    char text[100];
    printf("Introdu un sir de numere intregi: ");
    fgets(text, sizeof(text), stdin);
    /* mesaj si stocarea caracterelor introduse de utilizator in variabila txt*/
    char fnl[100];
    int j = 0;
    for (int i = 0; i < strlen(text); i++)
    {
        if (text[i] >= '0' && text[i] <= '9')
        {
            if ((text[i] - '0') % 3 == 0)
            {
                fnl[j++] = text[i];
            }
        }
    }
    /* In for se verifica daca pozitia vectorului contine numar >0 si <9 si daca da se executa urmatorul
    bloc de cod. Aici intalnim un alt if in care spune ca daca numarul continut de pozitia vectorului se
    imparte exact la 3 va fi stocat de variabila fnl. De observat ca am mai folosit si variabila j, dar e mult
    de explicat va spun la telefon*/
    fnl[j] = '\0';
    printf("Sirul de caractere divizibile cu 3 este: %s\n", fnl);

    return 0;
}
// mesaj si inchidere

```



```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main()
{
    char text[1000];
    FILE *pointer = fopen("Exercitiul 1.txt", "r");

    if (pointer == NULL)
    {
        printf("Documentul nu poate fi gasit\n");
        return 1;
    }

    fgets(text, sizeof(text), pointer);

    int lit_mari = 0;
    int ctrl = 0;

    for (int i = 0; i < strlen(text); i++)
    {
        if (isalpha(text[i]) && toupper(text[i]) && isalpha(text[i + 1]) && ctrl == 0)
        {
            lit_mari++;
            ctrl = 1;
        }
        if (text[i] == ' ')
        {
            ctrl = 0;
        }
    }
}
```

```
}

printf("Numarul de cuvinte care incepe cu litera mare este %d\n", lit_mari);

fclose(pointer); // Închide fișierul

return 0;

}
```

h.2.

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main()
{
    char text[1000];
    FILE *pointer = fopen("Exercitiul 1.txt", "r");

    if (pointer == NULL)
    {
        printf("Documentul nu poate fi gasit\n");
        return 1;
    }

    fgets(text, sizeof(text), pointer);

    int numere = 0;
    int ctrl = 0;

    for (int i = 0; i < strlen(text); i++)
    {
        if (isdigit(text[i]) && ctrl == 0) { // Folosește isdigit() pentru a verifica dacă caracterul este o cifră
            numere++;
            ctrl = 1;
        }
        if (text[i] == ' ') {
            ctrl = 0;
        }
    }
}
```

```
printf("In document sunt intalnite %d numere!\n", numere);  
fclose(pointer);  
  
return 0;  
}
```

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int main()
{
    char text[1000];
    FILE *pointer = fopen("Exercitiul 1.txt", "r");

    if (pointer == NULL)
    {
        printf("Documentul nu poate fi gasit\n");
        return 1;
    }

    fgets(text, sizeof(text), pointer);

    int numere = 0;
    int majuscule = 0;
    int minuscule = 0;
    int ctrl = 0;

    for (int i = 0; i < strlen(text); i++)
    {
        if (isupper(text[i]))
        {
            majuscule++;
        } else if (islower(text[i])) {
            minuscule++;
        }
        if (isdigit(text[i]) && ctrl == 0)
```

```
{  
    numere++;  
    ctrl = 1;  
}  
  
if (text[i] == ' ')  
{  
    ctrl = 0;  
}  
}  
  
printf("In document sunt %d litere mari, %d litere mici si %d cifre.\n", majuscule, minuscule,  
numere);  
fclose(pointer);  
return 0;  
}
```

# Problema 1 Adunare

```
#include <stdio.h>
#include <string.h>

int main()
{
    char nr_introduse[1000];
    char txt_fnl[1000];

    printf("Introdu un sir de caractere: ");
    fgets(nr_introduse, sizeof(nr_introduse), stdin);

    for (int i = 0; i < strlen(nr_introduse); i++)
    {
        if (nr_introduse[i] == '.' || nr_introduse[i] == ',' || nr_introduse[i] == ':' || nr_introduse[i] == ';' || nr_introduse[i] == '!' || nr_introduse[i] == '?') {
            txt_fnl[i] = ' ';
        }
        else
        {
            txt_fnl[i] = nr_introduse[i];
        }
    }

    txt_fnl[strlen(nr_introduse) - 1] = '\0'; // Elimină caracterul newline adăugat de fgets

    FILE *pointerfisier;
    pointerfisier = fopen("Exercitiul1.txt", "w");

    if (pointerfisier == NULL)
    {
        printf("Eroare la deschiderea fisierului.\n");
    }
```

```
    return 1;  
}  
  
Else  
{  
    fputs(txt_fnl, pointerfisier);  
    printf("gata!\n");  
}  
  
// Închide calea către fisier  
fclose(pointerfisier);  
  
return 0;  
}
```

Problema 2 Aditan

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

/* am adaugat si biblioteca ctype pentru ca in ea gasim functiile isupper, islower si isdigit. Ca noi sa nu scriem conditii pentru fiecare litera a alfabetului exista aceste metode. De asemenea pentru numere exista isdigit.Daca nu ar fii existat aceste metode, ar fi trebuit sa scriem cel putin 50 de conditii. Adica pentru fiecare majuscule, fiecare minuscula si fiecare numar*/
int main()
{
    // acum citesc din fisier si copies continutul in variabila txt_citit
    FILE *pointer;
    pointer = fopen("Exercitiul1.txt", "r");
    char txt_citit[1000];
    //Daca fisierul e gol primim eroare si mesaj
    if (pointer == NULL)
    {
        printf("Nu s-a gasit fisierul\n");
        return 1;
    }
    //urmatoarea linie copiaza textul din fisier in variabila txt_citit, dupa cum spuneam sim ai sus
    fgets(txt_citit, sizeof(txt_citit), pointer);
    //am inchis conexiunea la fisier
    fclose(pointer);
    //declar trei variabile care sa memorize numarul de litere mari, mici si numere
    int majuscule = 0, minuscule = 0, numere = 0;
    int lungime = strlen(txt_citit);
    /* pe linia anterioara am stocat in variabila numita lungime, dimensiunea vectorului de char: txt_fnl pentru ca este conditia de oprire a forului*/
    for (int i = 0; i < lungime; i++)
    {
        if (isupper(txt_citit[i]))
    }
```

```
    majuscule++;
}

else if (islower(txt_citit[i]))
{
    minuscule++;

}

else if (isdigit(txt_citit[i]))
{
    numere++;
}

}

/*In for se verifica fiecare pozitie a vectorului txt_fnl si daca intruneste una din cele trei, atunci se
incrementeaza cu 1 variabilele care Numara caracterele*/

printf("Majuscule: %d\n", majuscule);
printf("Minuscule: %d\n", minuscule);
printf("Numere: %d\n", numere);

//am dat mesaje cu variabilele si gata problema numarul 2

return 0;
}
```

### Problema 3 Adrián

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

char Met_vocala_ret(char consoana)

{
    int variabilalocala = 0;
    char vocala;

    if (isalpha(consoana) && isupper(consoana))
    {
        variabilalocala++;
        consoana = tolower(consoana);
    }

    switch (consoana)
    {
        case 'b':
        case 'c':
            vocala = 'a';
            break;
        case 'd':
        case 'f':
        case 'g':
            vocala = 'e';
            break;
        case 'h':
        case 'j':
        case 'k':
        case 'l':
            vocala = 'i';
    }
}
```

```
        break;

    case 'm':
    case 'n':
    case 'p':
    case 'q':
    case 'r':
        vocala = 'o';
        break;

    case 's':
    case 't':
    case 'v':
    case 'w':
    case 'x':
    case 'y':
    case 'z':
        vocala = 'u';
        break;

    default:
        vocala = consoana;
    }

if (variabilalocala == 1)
{
    vocala = toupper(vocala);
}

else
{
    return vocala;
}

}
```

```
fputs(txt_fnl, pointer2);

}

fclose(pointer2);

printf("S-a terminat intreg procesul!\n");

return 0;

}
```

```
int main()
{
    char txt_citit[1000];
    char txt_fnl[1000];
    FILE *pointer = fopen("Exercitiul1.txt", "r");

    if (pointer == NULL)
    {
        printf("Nu s-a gasit fisierul\n");
        return 1;
    }

    else
    {
        fgets(txt_citit, sizeof(txt_citit), pointer);

        fclose(pointer);

        for (int i = 0; i < strlen(txt_citit); i++)
        {
            txt_fnl[i] = Met_vocala_ret(txt_citit[i]);
        }
    }

    FILE *pointer2 = fopen("Exercitiul1.txt", "w");

    if (pointer2 == NULL)
    {
        printf("Nu s-a gasit fisierul\n");
    }

    else
    {
```

Subiecte Programarea Algoritmilor.txt

- 1)a) Instrucțiunea switch. Descriere. Detaliați.  
b) Instrucțiuni repetitive. Descriere. Detaliați. Diferențe.
- 2) Să se scrie un program în C pentru următoarea problema: Se citește un sir de la tastatură până se întâlnește ".":
  - i) Să se copieze într-un fișier, dar fără semne de punctuație;
  - ii) Câte litere mari, mici, cifre sunt în fișier?
  - iii) Să se înlocuiască în fișierul din text fiecare consoană cu vocala cea mai apropiată. Vocalele sunt A, E, I, O, U.
- 3) Să se scrie un program în C pentru următoarea problemă: Se citește un număr întreg k de la tastatură, apoi se citește un sir de k numere întregi de la tastatură.
  - i) Să se determine câte numere din sirul respectiv sunt mai mici decât k
  - ii) Să se determine câte numere sunt pare și câte numere sunt impare
  - iii) Să se creeze o listă simplu înlanțuită formată numai din numerele divizibile cu 3 din sirul respectiv.
- 4) Să se scrie un program în C pentru următoarea problemă: Se citește un text dintr-un fișier.
  - i) Să se determine câte cuvinte încep cu literă mare
  - ii) Să se determine câte numere sunt în text (un număr este o succesiune de cifre)
  - iii) Să se determine câte litere mari, mici și cifre sunt în text.

Pentru problemele 2, 3 și 4 descrieți în cuvinte ideea principală a algoritmului (un pseudocod) și apoi scrieți implementarea. Comentați cât mai mult din cod pentru a se

înțelege exact.



1. Instrucțiunee switch. Descrivi, detaliile.

1. Instrucțiuni repetitive. Descrivere diferențe.

Se scrie în propria ptt ~~un~~ transcriere\_subiect\_2\_PP (2).txt

2) Se citește un sir de la tastatura până se întâlnește '.'.

i) Să se copieze într-un fișier, dar fără semne de punctuație;

ii) Câte litere mari, mici, cifre sunt în fișier?

iii) Să se înlocuiască în fișier fiecare consoană cu vocala cea mai apropiată.

Vocalele sunt A, E, I, O, U. ~~ul din text~~

3. Propozitie - Se citește un nr. între K de la tastatură și se cită un nr. de K numere între 0 și K.

i) să se determine care numere din sirul respectiv sunt multipli exact K.

ii) să se determine care numere sunt pare și care numere sunt impare.

iii) să se creeze o listă simplă reprezentând numerele din numerele divizibile cu 3 din sirul respectiv.

4. Se scrie un program - ptt problema.

Se citește un text dintr-un fișier.

a) să se determine care caractere încep cu litere mari.

b) să se determine care numere sunt în text (în nr. este o succesiune de cifre)

c) să se determine care litere mici, ~~mari~~ și cifre sunt în text.

Pt. problemele 3 și 4 - Pseudocod (Descriere algoritmului).





## Re: Programare procedural...



Buna ziua, Domnule profesor  
se poate sa faceti disponibil un



[Vizualizare mesaj](#)



Liviu P. Dinu  
către Eu și încă 2  
⌚ Azi, 12:25



exemplu de subiecte:

1. Instructiunea switch. Descriere
2. Instructiuni repetitive. Descriere, diferente.
3. Sa se scrie un program (in C sau C++) pentru urmatoarea problema.  
Se citeste un numar intreg k de la tastatura, apoi se citeste un sir de k numere intregi de la tastatura.
  - a. aa se determine cate numere din sirul respectiv sunt mai mici decat k
  - b. Sa se determine cate numere sunt pare si cate numere sunt impare.
  - c. Sa se creeze o lista simplu inlantuita formata numai din numerele divizibile cu 3 din sirul



Stergere



Arhivare



Mutare



Răsp. tut.



Mai multe





3 din sirul  
respectiv

4. Sa se scrie un program (in C sau C++) pentru urmatoarea problema.

Se citeste un text dintr-un fisier.

a) Sa se determine cate cuvinte incep cu litera mare

b) Sa se determine cate numere sunt in text (un numar este o succesiune de cifre)

c) Sa se determine cate litere mari, mici, si cifre sunt in text.

Nota: pentru problemele 3 si 4 descrieti in cuvinte ideea principala a algoritmului folosit (un

pseudocod) si apoi scrieti implementarea; comentati cat mai mult din cod pentru a se

intelege exact.

cursul e aici:

<https://nlp.unibuc.ro/courses/PP.pdf>



$$\boxed{12:05} + 10^1 \rightarrow \boxed{12 \cdot 15}$$

① Instruction repetitive: détaillé.

② Se citeste un sir de la tastatura pînă să intâmneze:

- (i) Să se copieze într-un fișier, din fișierul său de punctuație;
- (ii) căte litere sunt, milii, cifre sunt în fișier;
- (iii) să se înțeleagă în fișierul său textul care să conțină cele patru variabile sunt A, E, i, O, și

