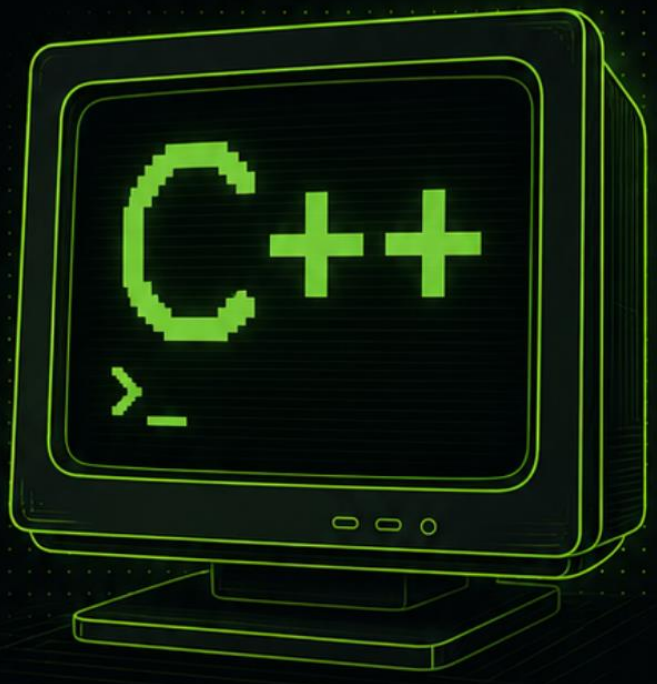
C++

// Șiruri de caractere
// pentru Bacalaureat

EDITURA EVOMIND

ȘIRURI DE CARACTERE
pentru Bacalaureat

Ghid pentru rezolvarea în C++
a problemelor cu șiruri
de caractere
de la subiectele II și III



TEORIE
Explicații clare și concise, esențiale pentru examen.

EXEMPLE
Exemple rezolvate pas cu pas, cu cod comentat.

SUBIECTE REZOLVATE (2026–2020)
Toate subiectele II și III rezolvate și explicate.

siruri.cpp

```cpp
#include <iostream>
#include <cstring>
using namespace std;

int main() {
    char s[256];
    cin >> s;
    // prelucrare șiruri de caractere
    return 0;
}

```

Cristina Constantinescu

TEORIE | EXEMPLE | SUBIECTE REZOLVATE | (2026–2020)

Cristina Constantinescu

Șiruri de caractere pentru Bacalaureat

Ghid pentru rezolvarea în C++ a problemelor cu șiruri de caractere de la subiectele II și III • Teorie • Exemple • Subiecte rezolvate (2026–2020)

Editura EVOMIND

ISBN: 978-606-9734-65-0

2026

CUVÂNT ÎNAINTE

Această carte a apărut din dorința de a oferi elevilor de liceu un instrument clar și practic pentru pregătirea exercițiilor privind șirurile de caractere pentru proba de Bacalaureat la informatică, în special pentru Subiectul al III-lea – Problema 2- tip obiectiv, dar și pentru exercițiile de la Subiectul II, itemi obiectivi sau semiobiectiv.

Materialul reunește funcțiile utilizate pentru prelucrarea șirurilor de caractere și exemple de utilizare a acestora în limbajul C++. Fiecare noțiune teoretică este însoțită de cod sursă comentat și de exemple concrete, astfel încât trecerea de la teorie la rezolvarea efectivă a unei probleme să fie cât mai firească și ușor de urmărit.

Ultima parte a cărții reunește un număr mare de subiecte rezolvate, cuprinzând variantele oficiale și testele de antrenament din perioada 2026–2020. Parcurgerea lor, în ordine sau selectiv, în funcție de nevoile fiecărui elev, oferă exercițiul necesar pentru a aborda cu încredere proba din ziua examenului.

Recomand ca această carte să fie folosită activ: studiați codul problemelor cu atenție, testați-l pe calculator și încercați și alte implementări ale problemelor. Înainte de rezolvarea propriu-zisă a unei probleme, este preferat să se simuleze pe hârtie efectul fiecărei instrucțiuni asupra șirului, urmărind, indice cu indice.

Mult succes la Bacalaureat!

Cristina Constantinescu

Cuprins

1. NOȚIUNI TEORETICE — ȘIRURI DE CARACTERE

1.1 Definiție și declarare

Un șir de caractere (string) este, în limbajul C/C++, un vector de tip char ale cărui elemente sunt caractere, terminat obligatoriu cu caracterul special NULL, notat '\0' (cod ASCII 0). Acest caracter marchează sfârșitul șirului.

Declarare: char s[101]; — un șir care poate reține cel mult 100 de caractere util pe pozițiile 0, 1, …, 99, poziția 100 rămâne rezervată pentru '\0'.

Pentru: *s[0]='b'; s[1]='a'; s[2]='c'; s[3]='\0'; => s-a format în variabila s șirul "bac"*

Lungimea logică a șirului (numărul de caractere utile) este dată de funcția *strlen* și este întotdeauna mai mică decât dimensiunea vectorului declarat.
Pentru șirul s construit mai sus, strlen(s) este 3.

1.2 Citirea și scrierea unui șir

cin >> s; citește un șir fără spații (se oprește la primul spațiu, tab sau Enter).

cin.get(s, 100); citește o linie întreagă, inclusiv spațiile, până la Enter sau până se ating 99 de caractere.

cin.get(); este folosit de obicei înaintea lui cin.get(s,100) pentru a „consuma" caracterul Enter rămas în buffer de la o citire anterioară cu >>.

cout << s; afișează șirul până la întâlnirea caracterului '\0'.

1.3 Funcțiile uzuale din biblioteca <string.h> (sau <cstring>)

1.3.1 strlen(s) — lungimea șirului

Returnează numărul de caractere utile din s (nu numără '\0').

```
cin>>s;
cout<<strlen(s);   // afișează lungimea lui s
```

Exemplu:

```
char s[] = "Ana are mere";
//s: Ana are mere; s+1: na are mere; s+2: a are mere;….
//s[0] este A, s[1] este n, s[2] este a, s[3] este spațiu
cout << s << " are " << strlen(s) << "caractere";
```

Textul afișat pe ecran este: Ana are mere 12 caractere

1.3.2 strcpy(dest, sursa) — copiere

Copiază conținutul din *sursa* în *dest*, inclusiv '\0'. Se poate folosi și cu adrese de forma *s+k* pentru a copia dintr-o poziție dată sau pentru a insera/lipi un șir începând de la o anumită poziție *k* dată.

```
char s[] = "Ana are mere";
strcpy(s+3, s+7); //s+7 are conținutul: mere care va înlocui //textul păstrat în
s+3, adică: are mere
cout << s;
```

Textul afișat pe ecran este: Ana mere

1.3.3 strncpy(dest, sursa, n) — copiază cel mult n caractere;

Dacă sursa are mai puține caractere decât n, restul este completat cu '\0'.
Atenție: dacă sursa are exact n sau mai multe caractere, *strncpy* NU adaugă automat sfârșitul de șir, acesta trebuie pus manual (*dest[n]='\0'*;).

```
strcpy(s,"informatica");     // s devine "informatica"
strncpy(s,"creion",4); s[4]='\0';   // s devine "crei"
```

1.3.4 strcat(dest, sursa) — concatenare (lipire)

Adaugă (lipește) conținutul lui *sursa* la finalul lui *dest*, după caracterul '\0' al lui *dest*, care este suprascris. Rezultatul se termină din nou cu '\0'.

```
strcpy(s1,"bac"); strcat(s1,"2026");  // s1 devine "bac2026"
```

1.3.5 strcmp(s1, s2) — comparare

Compară s1 cu s2, caracter cu caracter (ordine lexicografică) și returnează: 0 dacă șirurile sunt identice, o valoare negativă dacă s1 < s2, o valoare pozitivă dacă s1 > s2. Compararea se face automat, caracter cu caracter, atât timp cât 2 caractere de pe aceeași poziție sunt la fel. Când pe o poziție dată găsește două caractere diferite, se compară codul lor ASCII. Dacă codul ASCII al caracterului din primul șir este mai mic decât codul ASCII al caracterului de pe aceeași poziție din cel de-al doilea șir, rezultatul funcției e negativ, iar dacă e mai mare, rezultatul este pozitiv. Dacă până la final toate caracterele au avut același cod și aceeași lungime, atunci rezultatul funcției este 0.

```
 cout << strcmp("ban","bac") << endl; /*determină afișarea valorii 1 (codul ASCII al lui ‘n’ este 110 >  decat codul ASCII al lui ‘c’ care este 99, restul caracterlor aflate la stanga acestora fiind identice) */

cout << strcmp("bac","ac") << endl; /* determină afișarea valorii 1 (codul ASCII al lui ‘b’ este 98>
decat codul ASCII al lui ‘a’ care este 97*/

cout << strcmp("bac", "informatica") << endl; /* determină afișarea valorii -1 (codul ASCII al lui ‘b’ este  98 < decat codul ASCII al lui ‘i’ care este 105)*/

cout << strcmp("bac","bac") << endl; // determină afișarea valorii 0
```

1.3.6 strchr(s, c) — căutarea unui caracter

Returnează adresa primei apariții a caracterului c în șirul s, sau NULL dacă c nu se găsește.

```
cout << strchr("bacalaureat", 'a') << endl; // determina afisarea sirului “acalaureat”
```

Este folosită frecvent pentru a testa dacă un caracter este vocală, cifră, literă dintr-o mulțime dată etc.

```
if (strchr("aeiouAEIOU", c) != NULL)  // înseamnă că variabila c este vocală
        cout << “caracterul pastrat in variabila c este vocala”;
else
       cout << “caracterul pastrat in variabila c nu este vocala”;
```

1.3.7 strrchr(s, c) — căutare la dreapta

cout << strrchr("bacalaureat", 'a') << endl; //determina afisarea sirului “at”

Returnează adresa ultimei apariții a caracterului c în s (căutare de la dreapta la stânga).

1.3.8 strstr(s, t) — căutarea unui subșir

Returnează adresa de unde începe prima apariție a șirului *t în șirul s*, sau *NULL* dacă t nu se găsește în s.

cout << strstr("bacalaureat", "cal") << endl; //afișează sirul “calaureat”

cout << strstr("bacalaureat", "carte") << endl; / nu determină afișarea nici unei valori pe ecran, funcția returnând valoarea NULL */*

Diferența dintre adresa returnată și adresa de început a lui s reprezintă poziția de la care începe t în s.

cout << strstr("bacalaureat", "cal") - "bacalaureat" << endl;//afiseaza valoarea 2

1.3.9 strtok(s, separatori) — extragerea cuvintelor

Împarte șirul s în subșiruri (cuvinte), folosind caracterele din separatori drept delimitatori (de obicei spațiul " "). Prima apelare se face cu numele șirului: strtok(s," "); apelurile următoare, pentru a obține cuvintele următoare, se fac cu strtok(NULL," ");. Funcția returnează NULL când nu mai sunt cuvinte de extras.

```
char *p = strtok(s," ");
while (p != NULL)
{
   // prelucrarea cuvantului p
   cout << p << “-“;
   p = strtok(NULL," ");
}
```

Pentru textul “Ana are mere”, pe ecran se afișează textul: Ana-are-mere-

1.4 Parcurgerea unui șir caracter cu caracter

Cea mai frecventă structură de lucru cu șiruri este parcurgerea prin indice, de la 0 până la strlen(s)-1:

```
for (i = 0; i<strlen(s); i++)
   // prelucrarea caracterului s[i]
```

Pentru probleme de tip „palindrom” sau simetrie se folosesc doi indici, unul pornind de la capătul stâng (i = 0), celălalt de la capătul drept (j = strlen(s)-1), care se apropie unul de celălalt (i crește, j scade) până când i>=j.

1.5 Tehnici frecvente de prelucrare

1.5.1 Ștergerea unui caracter de pe o anumită poziție

strcpy(s+i, s+i+1); // toate caracterele de după poziția i se mută cu o poziție la stânga

1.5.2 Inserarea unui șir dat începând de pe o anumită poziție

Se reține mai întâi partea rămasă din s de la poziția i încolo într-un șir auxiliar, apoi se lipește t, apoi se lipește partea reținută (t, s și aux sunt șiruri de caractere):

strcpy(aux, s+i);
s[i] = '\0';
strcat(s, t);
strcat(s, aux);

1.5.3 Extragerea și prelucrarea cuvintelor unui text

Se folosește *strtok* pentru a separa textul în cuvinte după caracterul spațiu, iar fiecare cuvânt obținut poate fi analizat (lungime, apariția unor litere, simetrie etc.) și, eventual, adăugat cu strcat la un nou șir rezultat. De multe ori, pentru probleme în care se solicită modificarea în memorie a unui șir, pentru claritate, se poate construi un șir nou care va fi copiat în șirul initial.

1.5.4 Verificarea vocalelor / consoanelor

Se testează apartenența unui caracter la mulțimea vocalelor cu *strchr("aeiouAEIOU", s[i]) != NULL*. Analog, pentru cifre se poate folosi *strchr("0123456789", s[i]*).
Soluția următoare poate fi folosită pentru citirea unui șir de caractere și modificarea acestuia în memorie. Se construiește un nou șir și apoi se copiază peste șirul înițial. Se rețin cuvintele formate doar din vocale. Pentru aceasta, *strtok* se aplică șirului inițial. Se testează dacă există doar vocale în cuvântul accesat prin parcurgerea acestuia caracter cu caracter și utilizarea funcției *strchr*. Înainte de parcurgere, variabila ok este 1, iar dacă un caracter nu se află în șirul de vocale, ok devine 0. Dacă ok rămâne cu valoarea 1 pană la finalul cuvântului, acesta se adaugă în noul șir format. După fiecare cuvânt se adaugă și spațiu.

În situația în care textul initial are cuvintele separate prin unul sau mai multe spații, e recomandat să fie copiat textul initial într-un nou șir și acestuia să i se aplice *strtok*. Dacă cuvântul nu îndeplinește condiția dată, va fi căutat în șirul initial și se va șterge.

Aplicație 1 (textul are cuvintele separate printr-un spațiu și vreau să rămână cuvintele formate doar din vocale):

```cpp
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    char s[] = "Ana are oua si o ie", sn[100] = "";
    // s = sirul initial
    // sn = sirul nou
    char *p;
    p = strtok(s, " ");
    while (p != NULL)
    {
        /// verifică dacă p conține doar vocale
        int ok = 1;
        for (int i = 0; i < strlen(p); i++)
        {
            if (strchr("aeiou", p[i]) == NULL)
            {
                ok = 0;
                break;
            }
        }
        /// dacă este format doar din vocale, îl adaugă în sn
        if (ok == 1)
        {
            strcat(sn, p);
            strcat(sn, " ");
        }
        p = strtok(NULL, " ");
    }
    strcpy(s, sn);
    cout << s;
    return 0;
}
```

```
D:\Mate\sircă\bin\Debug\sirc
oua o ie
Process returned 0 (0x0)   execution time
```

Aplicație 2 (textul are cuvintele separate printr-unul sau mai multe spații și vreau să rămână doar cuvintele formate din vocale):

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    char s[] = "Ana are    oua    si o      ie", cs[100], sn[100] = "", *q;
    // s = sirul initial
    // sn = sirul nou
    char *p;
    strcpy(cs,s);
    p = strtok(cs, " ");
    while (p != NULL)
    {
        /// verifică dacă p contine doar vocale
        int ok = 1;
        for (int i = 0; i < strlen(p); i++)
        {
            if (strchr("aeiou", p[i]) == NULL)
            {
                ok = 0;
                break;
            }
        }
        /// dacă este format si din consoane îl caut in s si-l sterg
        if (ok == 0)
        {
            q = strstr(s,p);
            strcpy (q, q+strlen(p));
        }
        p = strtok(NULL, " ");
    }
    cout << s;
    return 0;
}
```

```
D:\Mate\sircă\bin\Debug\sirc
oua o ie
```

1.5.5 Interschimbarea a două caractere / două subșiruri

Pentru caractere se poate folosi funcția *swap(s[i], s[j])* sau o variabilă auxiliară de tip char:
c = s[i]; s[i] = s[j]; s[j] = c;
La fel procedăm și pentru șiruri de caractere.

```
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
   char s1[10] = "Ana", s2[10] = "Ioana";
   swap(s1,s2);
   cout << s1 << " " << s2;
   return 0;
}
```

Se afișează: Ioana Ana.
Sau utilizând variabilă auxiliară vom obține același lucru:
#include <iostream>

```
#include <string.h>
using namespace std;
int main()
{
   char s1[10]="Ana", s2[10]="Ioana", s3[10];
   strcpy(s3, s1);
   strcpy(s1, s2);
   strcpy(s2, s3);
   cout << s1 << " " << s2;
   return 0;
}
```

1.5.6 Observații importante pentru rezolvarea subiectelor de bacalaureat

- Dacă *s* este un șir, *s+k* reprezintă adresa caracterului aflat pe poziția k; *cout<<s+k; afișează șirul începând cu poziția k*, iar strcpy/strcat pot primi astfel de adrese ca argumente, ceea ce permite lucrul direct pe o „porțiune” a șirului.
- Diferența dintre două adrese returnate de strchr/strstr (de forma p - s, unde p este rezultatul funcției, iar s este șirul original) reprezintă poziția (indicele) de la care a fost găsit caracterul/ subșirul căutat.
- La atribuirea unui caracter '\0' pe o poziție k a șirului (s[k]='\0';), șirul este „scurtat” la lungimea k — toate caracterele de după poziția k devin invizibile pentru funcțiile de bibliotecă, deși rămân fizic în memorie.
- Pentru determinarea valorii corespunzătoare unui caracter cifră se calculează diferența dintre codul ASCII al cracterului cifră și codul ASCII al caracterului 0. Exemplu: pentru caracterul ‘3’, valoarea 3 este obținută ‘3’ - ‘0’ ceea ce corespunde valorii 51 – 48 = 3, știind că 51 este codul ASCII al lui ‘3’, iar 48 este codul ASCII al lui ‘0’.
- Pentru transformarea caracterului literă mare în caracterul literă mică, se adună 32. Exemplu: pentru a obține ‘a’ din ‘A’, adun la codul ASCII al lui ‘A’ valoarea 32. Cum codul ASCII al lui ‘A’ este 65 => 65 + 32 = 97 => am ajuns la ‘a’. Pentru a obține literă mica din literă mare se scade 32.
  Sacă șirul s este “bAc”, iar în program există instrucțiunile s[0] = s[0] - 32; s[1] = s[1] + 32; s[2] = s[2] - 32; => s devine: “BaC”

2. PROBLEME DATE LA BACALAUREAT — ȘIRURI DE CARACTERE (2026–2020)

2.1 Subiect 2026. Varianta 4. Sub II. 2

Variabilele **s** și **t** permit memorarea câte unui șir de maximum **20** de caractere.

Scrieți valorile afișate în urma executării secvenței C/C++ alăturate. **(6p.)**

```
strcpy(s,"anatoliana"); strcpy(t,"ana");
cout<<strlen(s)<<' '; | printf("%d ",strlen(s));
if(strstr(s,t)==s) cout<<"DA "; | printf("DA ");
else cout<<"NU "; | printf("NU ");
if(strcmp(strstr(s+1,t),t)==0) cout<<"DA"; | printf("DA");
else cout<<"NU"; | printf("NU");
```

s: anatoliana, t: ana

Lungimea lui s este 10. strstr (s,t) reține șirul de unde începe t în s, adică chiar de la început => este anatoliana, adică este chiar s. => Afișează DA

strstr(s+1, t) va fi șirul de unde începe t (ana) în natoliana => va fi ana. Deci strcmp are valoarea 0 deoarece cele două șiruri sunt egale . => Afișează DA

10 DA DA

2.2 Subiect 2026. Varianta SIMULARE

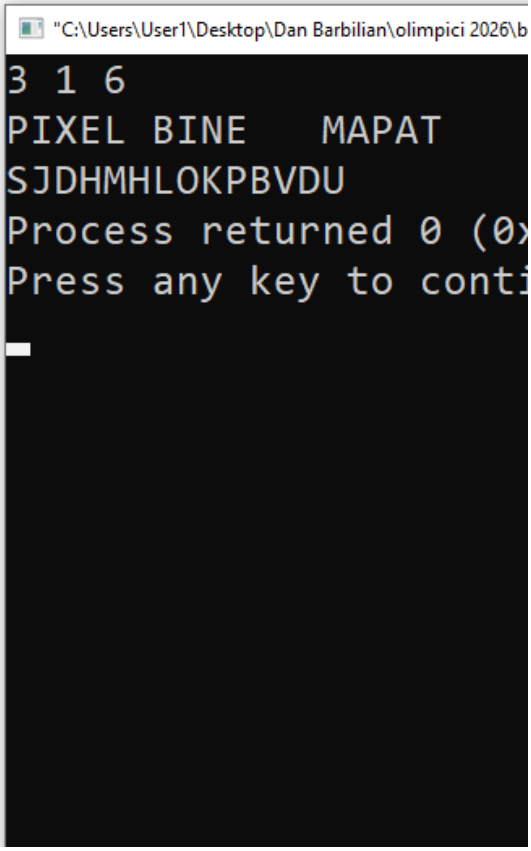
O metodă de criptare a unui text se bazează pe un cifru format din mai multe coduri, numere naturale. Prin criptare se elimină spațiile, apoi la fiecare pas se ia în considerare o pereche formată dintr-o literă a textului și un cod al cifrului, iar această literă se înlocuiește cu cea obținută prin deplasarea circulară, spre dreapta, în alfabetul limbii engleze, cu un număr de poziții egal cu codul din pereche. Prima pereche este formată din prima literă a textului și primul cod al cifrului, iar după fiecare pereche urmează cea formată din litera următoare a textului, parcurs de la stânga la dreapta, respectiv din codul următor din cifru, parcurs circular, spre dreapta, ca în exemplu.

Într-un text de cel mult **100** de caractere cuvintele sunt formate din litere mari ale alfabetului limbii engleze și sunt separate prin unul sau mai multe spații. Alfabetul limbii engleze are **26** de litere.

Scrieţi un program C/C++ care citeşte de la tastatură trei numere naturale din intervalul **[0,25]**, reprezentând, în această ordine, codurile unui cifru, apoi un text de tipul precizat. Programul modifică textul în memorie, criptându-l cu cifrul dat, apoi afişează pe ecran textul obţinut.

Exemplu: pentru cifrul cu codurile **3**, **1**, **6** și textul **PIXEL BINE MAPAT** se obține textul **SJDHMHLOKPBVDU** (primele patru perechi sunt **P-3**, **I-1**, **X-6**, **E-3**). **(10p.)**

```
char s[101]; int c1,c2,c3;
int main()
{
    int i,k;
    cin>>c1>>c2>>c3;cin.get();cin.get(s,100);
    ///sterg spatiile
    for (i=0; i<strlen(s); i++)
        if (s[i]==' ') {strcpy(s+i, s+i+1);i--;}
    ///parcurg  sirul si modific caracterele din sir
    ///Prima litera mare din alfabet are codul ASCII 65.
    ///Sunt 26 de litere => Ultimul cod este 65+26-1=90.
    ///Apoi, litera cu codul 91 tb sa fie cea care corespunde codului 65,
    ///92 cea care corespunde lui 66...
    for (i=0; i<strlen(s); i++)
        if (i%3==0)
        {
            if (s[i]+c1 <= 'Z' )
                s[i] =  (s[i]+c1) ;
            else
                s[i]='A'+ (s[i]+c1-'Z')-1;
        }
        else if (i%3==1)
        {
            if (s[i]+c2 <= 'Z' )
                s[i] = (s[i]+c2) ;
            else
                s[i]='A'+ (s[i]+c2-'Z')-1;
        }
        else
        {
            if (s[i]+c3 <= 'Z' )
                s[i] =  (s[i]+c3) ;
            else
                s[i]='A'+ (s[i]+c3-'Z')-1;
        }
    cout<<s;
    return 0;
}
```



2.3 Subiect 2026. Varianta MODEL. Sub. II.3

Variabilele **tI**, **pN** și **tL** permit accesul la câte un șir de maximum **20** de caractere. Inițial, șirul accesat prin **tL** este vid, șirul accesat prin **tI** reprezintă un număr de telefon în format internațional, iar șirul accesat prin **pN** reprezintă un prefix național sau este un șir vid.
Numărul de telefon în format internațional conține codul de țară, scris între paranteze rotunde, urmat de cifrele numărului propriu-zis. Codul de țară este format din cifre sau din simbolul **+** (plus), urmat de cifre. În multe țări, pentru a forma local un număr de telefon, se înlocuiește secvența formată din paranteze și codul de țară cu un prefix național format din cifre (de exemplu 0, în România) sau cu șirul vid (de exemplu în SUA).
Scrieți o secvență de instrucțiuni C/C++ astfel încât, în urma executării acesteia, șirul accesat prin **tL** să reprezinte numărul de telefon format local. Declarați corespunzător eventualele alte variabile utilizate.
Exemplu: dacă prin variabila **tI** se accesează șirul **(+254)722123456**, iar prin variabila **pN** se accesează șirul **0**, atunci numărul format local este **0722123456**, iar dacă prin variabila **tI** se accesează șirul **(+1)2121234567**, iar prin variabila **pN** se accesează șirul vid, atunci numărul format local este **2121234567**. **(6p.)**

Trebuie să reținem șirul ce urmează parantezei rotunde închise, să lipim la tL pe pN și apoi șirul reținut.
*char *p = strchr (tI, ')') + 1 ;*
strcat (tL,pN); strcat (tL, p);

```
#include <iostream>
#include <string.h>
using namespace std;
char tI[101]="(+254)722123456", pN[10]="0", tL[101]="";
int main()
{

    char *p = strchr ( tI, ')' ) + 1 ;
    strcat(tL,pN);
    strcat (tL, p);
    cout<<tL;
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\verif_111\bin\Deb
0722123456
Process returned 0 (0x0)   execut
```

2.4 Subiect 2025. Varianta 1

Un **cuvânt semioglindit** se obţine dintr-un cuvânt cu $2 \cdot k$ ($k \in [1, 10^2]$) litere, prin interschimbarea în acesta a secvenței formate din primele **k** litere cu secvența formată din ultimele **k** litere.
Exemplu: din cuvântul **platim** se obţine cuvântul semioglindit **timpla**.
Într-un text de cel mult **200** de caractere, cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spaţiu. Scrieţi un program C/C++ care citeşte de la tastatură un text de tipul precizat, pe care îl transformă în memorie, prin înlocuirea fiecărui cuvânt cu număr par de litere, cu cel semioglindit obținut din acesta, ca în exemplu. Programul afișează pe ecran textul obținut, sau mesajul **nu exista**, dacă toate cuvintele au număr impar de litere.
Exemplu: pentru textul **am facut fotografii unei flori mari**
se afișează pe ecran textul **ma facut rafiifotog eiun flori rima** **(10p.)**

*Vom accesa cuvintele din text cu **strtok** și dacă numarul de caractere e par vom forma un nou cuvânt în care punem mai întâi caracterele de la dreapta mijlocului, apoi le lipim pe cele de la stânga. Formăm un nou șir în care punem cuvintele.*

```cpp
#include <iostream>
#include <string.h>
using namespace std;
//Caut cuvintele de lungime para
/// am facut fotografii unei flori mari !
//sn e sirul nou
char s[202], sn[250]=" ", *p, aux[250];
int main()
{
    cin.get(s,200);
    p = strtok(s," ");
    while (p!=NULL)
    {
        int j = strlen(p);
        if (j%2 == 0)
        {
            //pastrez prima jumatate a.i. sa devina ultima
            strncpy(aux, p, j/2);
            aux[j/2] = NULL;
            strcpy(p, p+j/2);
            //lipesc  q la p
            strcat(p,aux);
        }
        strcat (sn,p);
        strcat (sn, " ");
        p=strtok(NULL," ");
    }
    //sterg spatiul adaugat la sfarsit
    strcpy(s, sn);
    strcpy(s,s+1);
    s[strlen(s)-1]=NULL;// a sters spatiu de la final!
    cout<<s;
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2025_ Var 1_ Sirc\bin\Debug\Bac 2025_ Var 1_ Sir
am facut fotografii unei flori mari
ma facut rafiifotog eiun flori rima
```

2.5 Subiect 2025. Varianta 7. Sub. II.3

Variabilele **s1** și **s2** permit accesarea câte unui șir de cel mult **50** de caractere, iar variabila **n** este de tip întreg. Scrieți șirul accesat prin variabila **s1**, precum și valoarea lui **n**, în urma executării secvenței alăturate. **(6p.)**

```cpp
strcpy(s1,"parcarea");
strcpy(s2,strstr(s1,"car"));
n=strlen(s2);
strcpy(s1+n-2, s2+n-2);
```

s1: parcarea

s2: carea

n = 5

s1+3: carea va deveni s2 + 3, adică => ea

*=> **s1: parea***

*Șirul s1 va fi **parea**, iar n va fi **5***

2.6 Subiect 2025. Varianta 6.

Într-un text de cel mult **100** de caractere cuvintele sunt formate doar din litere mici ale alfabetului englez și sunt separate prin unul sau mai multe spații. Textul are cel puțin două cuvinte.
Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat mai sus și îl transformă în memorie prin eliminarea sau inserarea unor spații și a unor cratime (simbolul –), astfel încât între oricare două cuvinte consecutive în text să fie câte o cratimă, încadrată la stânga și la dreapta de câte un spațiu, ca în exemplu. Programul afișează pe ecran textul obținut.
Exemplu: pentru textul
anul acesta devin student
se obține
anul – acesta – devin – student **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
//Extrag cuvintele
/// anul    acesta        devin              student
//sn e sirul nou
char s[102], sn[102]="", *p;
int main()
{
    cin.get(s,200);
    p = strtok(s," ");
    while (p!=NULL)
    {
        strcat (sn,p);
        strcat (sn, " - ");
        p=strtok(NULL," ");
    }
    strcpy(s, sn);
    //sterg spatiul -  adaugat la sfarsit
    strcpy(s+strlen(s)-3,s + strlen(s));
    cout<<s;
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2025 _ V6_ sir\bin\Debug\Bac 2025 _ V6_ sir.exe"
anul    acesta        devin              student
anul - acesta - devin - student
```

2.7 Subiect 2025. Varianta SIMULARE

Două cuvinte se numesc **asemenea** dacă sunt distincte și au același număr de vocale. Se consideră vocale literele **a**, **e**, **i**, **o**, **u**.
Scrieți un program C/C++ care citește de la tastatură un număr natural **n** ($n \in [1, 10^2]$), apoi **n** cuvinte, separate prin Enter. Fiecare cuvânt este format din cel mult **20** de caractere, numai litere mici ale alfabetului englez. Programul afișează pe ecran, separate prin câte un spațiu, toate cuvintele asemenea cu ultimul cuvânt citit, sau mesajul **nu exista** dacă nu există astfel de cuvinte.
Exemplu: dacă se citesc datele alăturate, se afișează pe ecran, nu neapărat în această ordine, cuvintele:
lalelele brandusele **(10p.)**

```
9
mereu
primavara
infloresc
ghioceii
lalelele
si
brandusele
de
primavara
```

```cpp
#include <iostream>
#include <string.h>
using namespace std;
//Citesc cuvintele si le pun intr-un vector de cuvinte
char s[101][21], voc[]="aeiou";
int main()
{
    int i, n, nrv1=0, nrv2;
    cin >> n;
    for (i=1; i<=n; i++)
        cin >> s[i];
    //determin nr vocale  ale ultimului cuvant
    ///parcurg cuvantul si numar vocalele
    for (i=0; i<strlen(s[n]); i++)
        if (strchr(voc, s[n][i]))
            nrv1++;
    ///Acum iau fiecare cuvant din primele n-1 si le afisez pe cele care au acelasi nr de voc
    //Folosesc ok pentru a trata cazul nu exista
    int ok=1;
    ///iau in calcul cuvintele distincte
    for (i=1; i<n; i++)
        if (strcmp(s[i], s[n])!=0)
        {
            nrv2=0;
            for (int j=0; j<strlen(s[i]); j++)
                if (strchr(voc, s[i][j]))
                    nrv2++;
            if (nrv2 == nrv1)
            {
                ok=1;
                cout<<s[i]<<" ";
            }
        }
    if (ok==0)
        cout<<"nu exista";
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2025 Simulare sirc\bin\Debug\Bac 2025 Simulare sirc
9
mereu
primavara
infloresc
ghioceii
lalelele
si
brandusele
de
primavara
lalelele brandusele
```

Observație:

Dacă se cere să se afișeze cuvintele care au același număr de vocale (**distinte**) cu ultimul, ultimul având 2 vocale, a și i, va trebui să parcurgem șirul de vocale și să vedem câte dintre ele se găsesc în cuvânt.

```
9
mereu
primavara
infloresc
ghioceii
lalelele
si
brandusele
de
primavara
```

Pentru același exemplu, vom avea cuvintele mereu, primavara și lalelele cu câte 2 vocale, la fel ca ultimul cuvânt citit, adică primavara.

```cpp
#include <iostream>
#include <string.h>
using namespace std;
//Citesc cuvintele si le pun intr-un vector de cuvinte
char s[101][21], voc[]="aeiou";
int main()
{
    int i, n, nrv1=0, nrv2;
    cin >> n;
    for (i=1; i<=n; i++)
        cin >> s[i];
    //determin nr vocale distincte ale ultimului cuvant
    ///parcurg sirul de vocale si vad daca le regasesc in ultimul cuvant
    //voc[i] este caracterul cautat si s[n] este ultimul cuvant
    for (i=0; i<strlen(voc); i++)
        if (strchr(s[n], voc[i]))
            nrv1++;
    ///Acum iau fiecare cuvant din primele n-1 si le afisez pe cele care au acelasi nr de voc
    //Folosesc ok pentru a trata cazul nu exista
    int ok=1;
    for (i=1; i<n; i++)
    {
        nrv2=0;
        for (int j=0; j<strlen(voc); j++)
            if (strchr(s[i], voc[j]))
                nrv2++;
        if (nrv2 == nrv1)
        {
            ok=1;
            cout<<s[i]<<" ";
        }
    }
    if (ok==0)
        cout<<"nu exista";
    return 0;
}
```

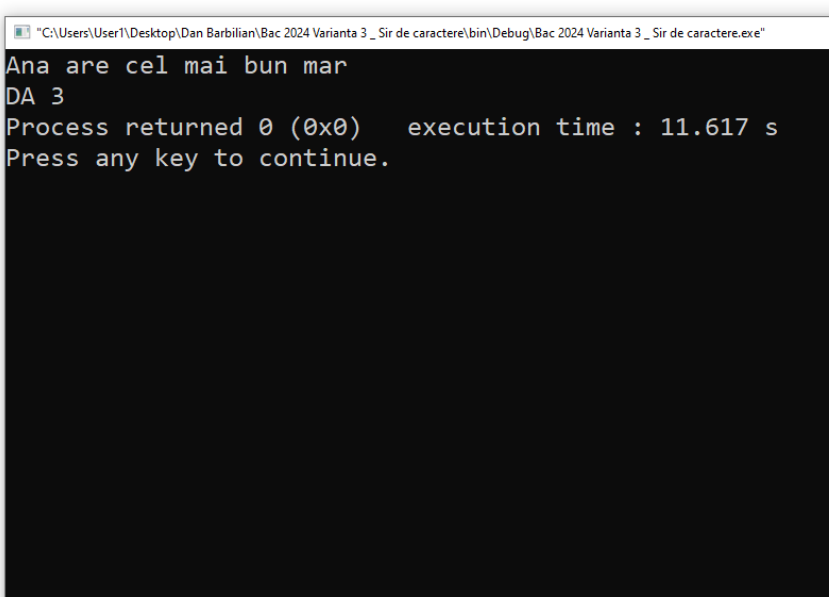
```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2025 Simulare sirc\bin\Debug\Bac 2025 Simulare sirc.ex
9
mereu
primavara
infloresc
ghioceii
lalelele
si
brandusele
de
primavara
mereu primavara lalelele
Process returned 0 (0x0)   execution tim
```

2.8 Subiect 2024. Varianta 3

Într-un text, de cel mult **100** de caractere, cuvintele sunt formate din litere ale alfabetului englez și sunt separate prin câte un spațiu. Textul are cel puțin două cuvinte.
Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat mai sus și afișează pe ecran mesajul **DA** și un număr natural **n**, separate printr-un spațiu, dacă toate cuvintele din text au câte **n** litere, sau mesajul **NU** în cazul în care nu toate cuvintele au același număr de litere.
Exemplu: dacă textul citit este **Ana are cel mai bun mar** se afișează pe ecran **DA 3**
iar dacă textul citit este **Ana are cel mai dulce mar** se afișează pe ecran **NU** **(10p.)**

```cpp
#include <iostream>
#include <string.h>
using namespace std;
//Extrag cuvintele
/// Ana are cel mai bun mar
//Extrag cuvintele cu strtok.
//Retin lungimea primului cuvant si verific daca toate celelalte au aceeasi lungime.

char s[102], *p;
int n, ok=1;
int main()
{
    cin.get(s,100);
    p = strtok(s," ");
    n = strlen (p);
    while (p!=NULL)
    {
        if (strlen (p) != n)
        {
            ok=0;
            break;
        }
        p=strtok(NULL," ");
    }
    if (ok==1)
        cout<<"DA "<<n;
    else
        cout<<"NU";
    return 0;
}
```



using namespace std;
//Extrag cuvintele
/// Ana are cel mai bun mar
//Extrag cuvintele cu strtok.
//Retin lungimea primului cuvant si verific daca toate celelalte au aceeasi lungime.

2.9 Subiect 2024. Varianta 4. Sub. II. 3

Variabilele **c** și **i** sunt de tip întreg, iar variabila **s** permite memorarea unui șir de cel mult **20** de caractere. Se citesc de la tastatură **10** cuvinte, formate din litere mici ale alfabetului englez și separate prin Enter. Scrieți secvența de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila **c** să memoreze valoarea **1** dacă există printre cuvintele citite cel puțin unul format din două litere și care să conțină o vocală și o consoană, sau valoarea **0** altfel. Se consideră vocale literele **a**, **e**, **i**, **o**, **u**.
Exemplu: dacă se citesc cuvintele alăturate, variabila **c** are valoarea **1**.

daca
au
plecat
el
nu
primeste
si
inghetata
de
fragi

```
c=.....;
for(i=1;i<=10;i++) {  cin>>s;  | scanf("%s",s);
                      ..........
                   }
```

(6p.)

c= 0;
for (i=1; i<=10; i++)
{ cin >> s; if (strlen(s)==2 && ((strchr ("aeiou", s[0])!= NULL && strchr ("aeiou", s[1])== NULL)) || (strchr ("aeiou", s[0])== NULL && strchr ("aeiou", s[1])!= NULL))
c=1; }

2.10 Subiect 2024. Varianta SIMULARE

Un **șablon** este un text în care cuvintele sunt separate prin câte un spațiu și sunt formate fie numai din litere mici și mari ale alfabetului englez, fie numai din caractere *****, în ultimul caz numindu-se **cuvinte generice**. Lungimea unui cuvânt este egală cu numărul de caractere care îl compun.
Un computer generează o frază pe baza unui astfel de șablon, prin înlocuirea fiecărui cuvânt generic cu unul dintre cuvintele de aceeași lungime, preluat dintr-o listă dată.
Scrieți un program C/C++ care citește de la tastatură un număr natural, **n** (**n∈[1,100]**), și o listă de **n** cuvinte, urmată de un șablon de tipul precizat. Fiecare cuvânt din listă este format din maximum **10** litere mici și mari ale alfabetului englez și la citire este introdus singur pe linie. Șablonul conține maximum **100** de caractere. Programul obține în memorie și apoi afișează pe ecran una dintre frazele care pot fi generate pe baza șablonului și a listei citite sau mesajul **imposibil** dacă nu se poate genera o astfel de frază.

Exemplu: dacă **n=6**, iar lista de cuvinte este cea alăturată,

pentru șablonul	**Era o vreme ******* si ***** din belsug *******
se generează fraza	**Era o vreme placuta si soare din belsug soare**
sau fraza	**Era o vreme calduta si soare din belsug acasa**
etc., iar pentru șablonul	***** o vreme ******* si **************
se afișează mesajul **imposibil**	

(10p.)

```
rece
placuta
acasa
calduta
innorata
soare
```

*Extrag cuvintele din fraza. Daca cuvântul începe cu * caut în vector un cuvânt de aceeasi lungime. Dacă există un astfel de cuvânt, îl adaug în fraza nouă și opresc căutarea în vectorul de cuvinte. Dacă nu gasesc un cuvânt în vector, opresc algoritmul punând în variabila gasit valoarea 0. Dacă nu s-a oprit algoritmul, trec la următorul cuvânt din fraza.*

```cpp
#include <string.h>
using namespace std;

char s[101][11], cuvant[11], fraza [101], frazan[101]="";
int n, ok=1, gasit=1;
int main()
{
    char *p;
    int i,ok;
    cin >> n;
    for (i=1; i<=n; i++)
        cin >> s[i];
    cin.get();
    cin.get(fraza, 100);
    p = strtok(fraza," ");
    while (p!=NULL)
    {
        if (p[0]=='*')
        {
            int ok=0;
            for (i=1; i<=n; i++)
                if (strlen(s[i])==strlen(p))
                {
                    strcat (frazan, s[i]);
                    ok=1;
                    break;
                }
            if (ok==0)
            {
                gasit=0;
                break;
            }
        }
        else
            strcat (frazan, p);
        strcat(frazan," ");
        p=strtok(NULL," ");
    }
    strcpy(fraza, frazan);
    if (gasit==0)
        cout<<"imposibil";
    else
        cout<<frazan;
    /*Extrag cuvintele din fraza. Daca cuv incepe cu * caut in vector un cuvant
    de aceeasi lungime. Daca exista adaug cuvantul respectiv in fraza noua.
    In cazul in care nu gaseac cuvantul de aceeasi lungime voi opri algoritmul si gasit este

    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2024 Varianta4 Sir de
6
rece
placuta
acasa
calduta
innorata
soare
Era o vreme ******* si ***** din belsug *****
Era o vreme placuta si acasa din belsug acasa
Process returned 0 (0x0)   execution time : 3.1
Press any key to continue.
```

2.11 Subiect 2024. Varianta MODEL

Un text are cel mult **100** de caractere, iar cuvintele sale sunt formate numai din litere mici ale alfabetului englez, sunt distincte și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un număr natural **n** (**n∈[1,10²]**), apoi un text de tipul precizat mai sus, și afișează pe ecran cuvinte ale acestuia, pe două linii separate, astfel încât prima linie să conțină mulțimea cuvintelor care au mai puțin de **n** litere, iar a doua linie să conțină mulțimea cuvintelor care au mai mult de **n** litere. Cuvintele de pe fiecare linie sunt afișate într-o ordine oarecare, iar dacă una dintre cele două mulțimi este vidă, se afișează pe ecran doar mesajul **nu exista**.

Exemplu: pentru **n=3** și textul **era o apa rece si cu gust bun** se poate afișa pe ecran textul alăturat.

(10p.)

```
o si cu
rece gust
```

Citesc n, apoi citesc textul. Separ cuvintele din text și le pun în 2 vectori de cuvinte. Într-unul cuvintele de lungime mai mica decât n, iar în altul voi pune cuvintele cu lungime mai mare decât n. Dacă lungimea unuia este 0 afișez mesajul "nu există", altfel afișez cei 2 vectori.

```
#include <iostream>
#include <string.h>
using namespace std;
char s1[102][100], s2[102][100],  *p, s[101];
int n, nr1, nr2;
int main()
{
    cin>>n;
    cin.get();
    cin.get(s,200);
    p = strtok(s," ");
    while (p!=NULL)
    {
        if (strlen(p)<n){
            nr1++;
            strcpy(s1[nr1],p);
        }
        else
          if (strlen(p)>n){
            nr2++;
            strcpy(s2[nr2],p);
        }
        p=strtok(NULL," ");
    }
    if (nr1==0 || nr2==0)
        cout<<"nu exista";
    else
    {
        for (int i=1; i<=nr1; i++)
            cout<<s1[i]<<" ";
        cout<<endl;
        for (int i=1; i<=nr2; i++)
            cout<<s2[i]<<" ";
    }
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2024 MODEL Sir de caractere\bin\Debug\B
3
 era o apa rece si cu gust bun
o si cu
rece gust
Process returned 0 (0x0)   execut
Press any key to continue.
```

O altă soluție, va construi 2 șiruri de caractere. Intr-unul să pună cuvintele de lungime mai mică decât n, iar în celălalt să punem cuvintele de lungime mai mare decât n.

#include <iostream>
#include <string.h>

```
using namespace std;
char s1[102]="", s2[102]="",  *p, s[101];
int n, nr1, nr2;
int main()
{
   cin>>n;
   cin.get();
   cin.get(s,200);
   p = strtok(s," ");
   while (p!=NULL)
   {
      if (strlen(p)<n){
         strcat(s1,p); strcat(s1," ");
      }
      else
        if (strlen(p)>n){
          strcat(s2,p); strcat (s2," ");
      }
      p=strtok(NULL," ");
   }
   if (strlen(s1)==0 || strlen(s2)==0)
      cout<<"nu exista";
   else
         cout<<s1<<endl<<cout<<s2;
   return 0;
}
```

2.12 Subiect 2023. Varianta 6. Sub II.3

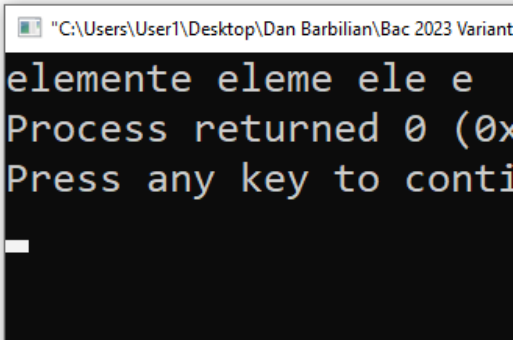
Un șir de caractere **a** este numit **prefix** al unui șir de caractere **b** dacă este identic cu **b** sau dacă **b** se obține din **a** prin adăugarea la dreapta a unor alte caractere.
Variabila **k** este de tip întreg, iar variabila **s** permite accesul la un șir de cel mult **20** de caractere. Scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, să se afișeze pe ecran, în ordinea descrescătoare a lungimii, separate prin câte un spațiu, toate prefixele șirului accesat prin variabila **s**, fiecare încheindu-se cu prima literă a șirului **s**, ca în exemplu. Declarați eventualele alte variabile utilizate.
Exemplu: pentru șirul **elemente** se afișează: **elemente eleme ele e**

```
for(k=strlen(s)-1;k>=0;k--)
{..................}
```

(6p.)

```
for(k=strlen(s)-1; k>=0; k--)
   {
      if (s[0]==s[k])
         cout<<s<<" ";
      s[k]=NULL;
   }
```

```
#include <iostream>
#include <string.h>
using namespace std;
int k;
char s[21]="elemente";
int main()
{
    for(k=strlen(s)-1; k>=0; k--)
    {
        if (s[0]==s[k])
            cout<<s<<" ";
        s[k]=NULL;
    }
    return 0;
}
```



2.13 Subiect 2023. Varianta SIMULARE

Într-un text de cel mult **100** de caractere cuvintele sunt separate prin câte un spațiu și sunt formate din litere mari ale alfabetului englez, iar dacă sunt scrise prescurtat sunt urmate de caracterul . (punct). Textul reprezintă denumirea științifică a unei păsări și doar cuvintele din mulțimea {**FAMILIA**, **GENUL**, **SPECIA**}, specifice sistemului de clasificare a organismelor, sunt mereu prescurtate, prin eliminarea ultimelor lor litere.
Scrieţi un program C/C++ care citeşte de la tastatură un text de tipul precizat și construiește în memorie, apoi afișează pe ecran denumirea științifică, în care pentru cuvintele specifice sistemului de clasificare a organismelor se păstrează doar primele trei litere, scrise cu litere mici, și urmate de punct, ca în exemplu.
Exemplu: pentru textul **FAMIL. PHASIANIDAE GEN. MELEAGRIS SP. GALLOPAVO**
sau pentru textul **FAM. PHASIANIDAE G. MELEAGRIS SPECI. GALLOPAVO**
se obține **fam. PHASIANIDAE gen. MELEAGRIS spe. GALLOPAVO** **(10p.)**

```cpp
#include <iostream>
#include <string.h>
using namespace std;
char s[101], fraza[101]="";
int n;
int main()
{
    char *p;
    cin.get(s, 100);
    p = strtok (s, " ");
    while (p!=NULL)
    {
        if (p[strlen(p)-1]=='.')
        {
            ///verific daca cuvantul se termina cu .
            //verific prima litera
            if(p[0]=='S')
                strcat(fraza,"spe.");
            else if(p[0]=='F')
                strcat(fraza,"fam.");
            if(p[0]=='G')
                strcat(fraza,"gen.");
        }
        else
            strcat(fraza,p);
        strcat(fraza," ");
        p = strtok(NULL," ");
    }
    cout<<fraza;

    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\simulare2023\bin\Debug\simulare2023.exe"
FAMIL. PHASIANIDAE GEN. MELEAGRIS SP. GALLOPAVO
fam. PHASIANIDAE gen. MELEAGRIS spe. GALLOPAVO
```

2.14 Subiect 2023. Varianta MODEL. Sub. II. 3

Variabila **p** este de tip întreg, iar variabilele **s1** şi **s2** permit memorarea câte unui şir de cel mult **30** de caractere. Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```cpp
strcpy(s1, "plantau fistic");
p=strchr(s1,' ')-s1;
strcpy(s2,s1+p+1); strcpy(s1+p-1,s2+2);
strcpy(s2+1,s1+2);
cout<<p<<s2; | printf("%d%s",p,s2);
```

strchr(s1, ' ') va reține adresa șirului de unde începe spațiul, iar s1 va reține adresa șirului ***plantau fistic****. P va fi diferența dintre cele 2 adrese, adică va fi chiar lungimea cuvântului* ***plantau****, deci va fi 7. În s2 se reține textul* ***fistic*** *și apoi s2+2, adică stic se copiază la adresa s1+6, adică acolo unde începe* ***u*** *=> s1 devine:* ***plantastic***
Prin strcpy(s2+1, s1+2), în s2 voi înlocui pe ***istic*** *cu* ***antastic****, deci s2 devine:* ***fantastic****.*
La final se afișează ***7fantastic***

2.15 Subiect 2022. Varianta 5

Se consideră o vocală oarecare a alfabetului englez, notată cu **v**, și o consoană oarecare a alfabetului englez, notată cu **c**. Litera **v** se numeşte **vocală prietenă** a lui **c** dacă în şirul literelor alfabetului englez, ordonat lexicografic, **v** îl precede pe **c**, iar între **v** și **c** nu există nicio vocală. Se consideră vocale literele **a**, **e**, **i**, **o**, **u**.
Exemplu: **e** este vocală prietenă pentru consoanele **f**, **g** și **h**, dar nu este vocală prietenă pentru consoanele **d** și **j**.
Un elev vrea să transmită unui prieten o parolă, codificată. Parola este formată dintr-un singur cuvânt de cel mult **50** de caractere, litere mici ale alfabetului englez, cel puțin una fiind consoană. Codificarea se face prin înlocuirea fiecărei consoane cu vocala sa prietenă, ca în exemplu.
Scrieţi un program C/C++ care citeşte de la tastatură un cuvânt, reprezentând o parolă de tipul precizat şi determină, în memorie, forma codificată a acesteia. Programul afişează pe ecran parola codificată obţinută.
Exemplu: pentru parola **rame** se afişează **oaie**, iar pentru parola **sport** se afișează **ooooo** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
char s[51], voc[]="aeiou";

int main()
{
    int i;
    cin >> s;
    for (i=0; i<strlen(s); i++)
        if (strchr(voc,s[i]) == NULL)
    {
        //caut vocala precedenta din alfabet inainte de s[i]
        int j=s[i];
        while (char (j)>='a' && strchr("aeiou",char (j))==NULL)
            j--;
        s[i]=j;
    }
    cout<<s;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\Bac 2022 Varianta 5_ Sirc\bin\Debug\Bac 2022 Varianta 5_ Sirc.exe"
rame
oaie
Process returned 0 (0x0)   execution time : 1
```

2.16 Subiect 2022. Varianta 1. Sub II.3

Variabilele **s** și **id** permit accesul la câte un șir de maximum **50** de caractere, șirul accesat prin **id** fiind inițial vid, iar cel accesat prin **s** memorând, în această ordine, separate printr-un spațiu, prenumele și numele unei persoane, fiecare fiind format numai din litere ale alfabetului englez. Scrieți o secvență de instrucțiuni C/C++ astfel încât, în urma executării acesteia, șirul accesat prin **id** să memoreze numele persoanei menționate, urmat de **2022**. Declarați corespunzător eventualele alte variabile utilizate.
Exemplu: dacă șirul accesat prin variabila **s** este **Ana Popescu**
atunci șirul accesat prin variabila **id** este **Popescu2022** **(6p.)**

strcat(id, strchr(s,' ') + 1); //am pastrat in id șirul aflat după spațiu.
strcat(id, "2022");

2.17 Subiect 2022. Varianta SIMULARE

Un text, de cel mult **250** de caractere, reprezintă o listă cu date de identificare ale invitaților la o petrecere; fiecare invitat are un prenume și un nume, care apar în listă în această ordine, urmate de simbolul **;** (punct și virgulă), ca în exemplu. Numele și prenumele sunt formate din câte un singur cuvânt, compus din litere mari ale alfabetului englez, și sunt separate printr-un spațiu.
Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat mai sus apoi, de pe rândul următor, un cuvânt, **x**, și afișează pe ecran, separate prin câte un spațiu, **numele** tuturor invitaților care au **prenumele x**, ca în exemplu, sau mesajul **NU** dacă nu există astfel de invitați.
Exemplu: dacă lista este **DAN MARIS; DANILA PREPELEAC; DAN POPA; EDANA DAN;**
și cuvântul **x** este **DAN** se afișează pe ecran **MARIS POPA** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;


int main()
{
    char sir[251];
    char x[10], *p;
    cin.getline(sir,1000);
    cin>>x;
    p=strtok(sir,";");
    while(p!=NULL){
          //cout<<p<<endl;
          //verific daca x este prenume pentru p
          if (strcmp(strstr(p,x),p)==0 && p[strlen(x)]==' ')
              cout<<strchr(p,' ')+1<<" ";
          p=strtok(NULL,";");
    }
     return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\sircaracter\bin\Debug\sircaracter.exe"
DAN MARIS;DANILA PREPELEAC;DAN POPA;EDANA DAN;
DAN
MARIS POPA
Process returned 0 (0x0)   execution time : 2.872 s
Press any key to continue.
```

2.18 Subiect 2021. Varianta 4. Sub II.2

Variabila **s** poate memora un șir de cel mult **20** de caractere, variabila **aux** este de tip **char**, iar celelalte variabile sunt de tip întreg.
Scrieți șirul memorat prin intermediul variabilei **s** în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"ROMANIA");
i=strlen(s)-1;
for(j=3;j>=0;j--)
{ aux=s[i]; s[i]=s[i-j]; s[i-j]=aux;
  i=i-j;
}
```

i	j	i-j	S[i]	S[i-j]	0123456
					ROMANIA
~~6~~ 3	3	3	~~A~~ A	~~A~~ A	~~ROMANIA~~
~~3~~ 1	2	1	~~N~~ A	~~A~~ N	~~RAMONIA~~
~~1~~ 0	1	0	~~A~~ R	~~R~~ A	~~ARMONIA~~
0 0	0	0	~~A~~ A	~~A~~ A	ARMONIA

Șirul memorat în variabila s este: ARMONIA

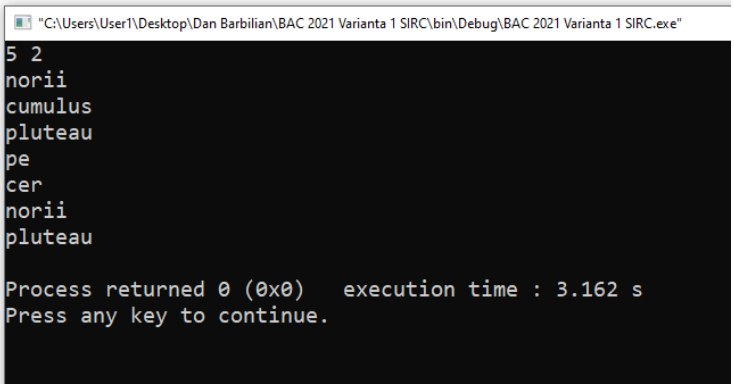
2.19 Subiect 2021. Varianta 1

Scrieți un program C/C++ care citește de la tastatură două numere naturale **n** și **k**, apoi **n** cuvinte, separate prin Enter. Fiecare cuvânt este format din cel mult **10** caractere, numai litere mici ale alfabetului englez, iar numerele citite sunt din intervalul **[1,20]**.

Programul afișează pe ecran, pe linii separate, primele **k** cuvinte dintre cele citite pentru care ultima literă este o vocală, sau **doar** mesajul **nu exista** dacă nu există **k** astfel de cuvinte. Se consideră vocale literele **a**, **e**, **i**, **o**, **u**.
Exemplu: dacă se citesc datele alăturate, se afișează pe ecran:
norii
pluteau **(10p.)**

```
5 2
norii
cumulus
pluteau
pe
cer
```

```
#include <iostream>
#include <string.h>
using namespace std;
//Citesc  cuvintele si le pun in vector pe cele care au ultima litera vocala
char s[21][11], *p, cuv[11];
int n,  k, nr;
int main()
{
    cin>>n>>k;
    for (int i=1; i<=n; i++)
    {
        cin>>cuv;
        if (strchr("aeiou", cuv[strlen(cuv)-1])!=NULL)
        {
            nr++;
            strcpy(s[nr], cuv);
        }
    }
    if (nr<k)
        cout<<"nu exista";
    else
        for (int i=1; i<=k; i++)
         cout<<s[i]<<endl;
    return 0;
}
```



2.20 Subiect 2021. Varianta 7

3. Variabila **i** este de tip întreg, iar variabila **x** permite memorarea unui şir cu cel mult **100** de caractere. Scrieți ce se afişează în urma executării secvenţei alăturate. **(6p.)**

```
strcpy(x,"bac2021");
cout<<x+3<<endl;   |  printf("%s\n",x+3);
for(i=0;i<strlen(x);i++)
  if(strchr("0123456789",x[i])==0)
    cout<<x[i]<<'!'; | printf("%c! ",x[i]);
```

x devine BAC2021. Se afișează x+3: ***2021***
Se afișează toate caracterele care nu sunt cifre din sirul x și apoi semnul exclamării: ***B!A!C!***

⇨ *În urma executării algoritmului, se afișează*
2021
B!A!C!

2.21 Subiect 2021. Varianta SIMULARE

Variabila **i** este de tip întreg, iar variabila **s** permite memorarea unui șir de cel mult **20** de caractere. Scrieți șirul accesat prin variabila **s** în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"ELITIST");
for(i=2;i<6;i++)
  if(i%2==0) s[i]=s[0];
  else s[i]=s[1]+i/2;
```

i	S[i]	0123456 ELITIST
~~2~~	E	~~ELETIST~~
~~3~~	'L'+3/2 => 'L'+1 => M	~~ELEMIST~~
~~4~~	E	~~ELEMEST~~
5	'L'+5/2 => 'L'+2 => N	ELEMEN

2.22 Subiect 2021. Test antrenament 1. Sub II.3

Variabilele **s1** și **s2** pot memora câte un șir de cel mult **50** de caractere. Scrieți ce se afișează în urma executării secvenței alăturate.

```
strcpy(s1,"bac2021");
cout<<strlen(s1)<<endl; | printf("%d\n",lengtf(s1));
strcpy(s2,s1+3); strcpy(s2+2,"20-");
strcat(s2,s1+3);
cout<<s2; | printf("%s",s2);
```

Pentru s1: bac2021 se afișează lungimea sa, adică 7.
s1+3: 2021 => s2 este 2021. Prin strcpy(s2+2, "20-"), s2 devine 2020-.
Prin strcat(s2,s1+3) => s2: 2020-2021.
=> Se vor afișa:
7
2020-2021

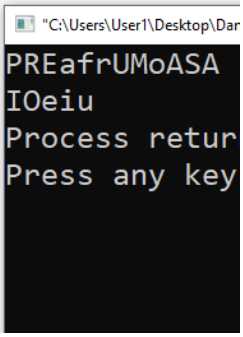
2.23 Subiect 2021. Test antrenament 1. Sub II.3

Variabila **s** memorează un șir de cel mult **20** de caractere (litere mari și mici ale alfabetului englez). Declarați eventuale alte variabile necesare și scrieți o secvență de instrucțiuni în urma executării căreia se afișează pe ecran vocalele care **NU** apar în șirul menționat.
Se consideră vocale literele **a**, **e**, **i**, **o**, **u**, **A**, **E**, **I**, **O**, **U**.
Exemplu: dacă se citește textul **PREafrUMoASA** se afișează pe ecran, nu neapărat în această ordine, vocalele: **euOiI** **(6p.)**

```
#include <iostream>
#include <string.h>
using namespace std;

char s[21], voc[]="AEIOUaeiou";

int main()
{
    int i;
    cin >> s;
    for (i=0; i< strlen(voc); i++)
        if (strchr(s,voc[i])==0)
            cout<<voc[i];
    return 0;
}
```



2.24 Subiect 2021. Test antrenament 2. Sub II.3

Variabila **s** permite memorarea unui șir de cel mult 20 de caractere.

Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"muzeu");
s[0]=s[0]+1;
cout<<s[1]<<s[0]<<endl; | printf("%c%c\n",s[1],s[0]);
strcpy(s,"muzeu"+2);
cout<<s; | printf("%s",s);
```

s: muzeu
s[0] devine n, apoi se afișează un. Lui s i se atribuie valoarea zeu și apoi se afișează s.
=> se afișează
un
zeu

2.25 Subiect 2021. Test antrenament 4. Sub II.3

In secvenţa alăturată, variabila **i** este de tip întreg, iar variabilele **s** și **t** pot memora câte un şir cu cel mult 20 de caractere.
Scrieţi ce se afișează pe ecran în urma executării secvenţei. **(6p.)**

```
strcpy(s,"sanataTEA");
cout<<strlen(s); | printf("%d", strlen(s));
i=0;
while(i<strlen(s))
  if(s[i]=='a')
  { strcpy(t, s+i+1); strcpy(s+i, t);}
  else i=i+1;
cout<<s; | printf("%s",s);
```

i 012345678	s
0	~~sanataTEA~~
1	~~snataTEA~~
2	~~sntaTEA~~
3	~~sntaTEA~~
4	sntTEA

Se afișează:
9sntTEA

2.26 Subiect 2021. Test antrenament 5. Sub II.3

Variabila `i` este de tip întreg, iar variabila `s` permite memorarea unui şir cu cel mult 10^2 caractere.
Scrieți ce se afișează pe ecran în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"informatica");
cout<<strlen(s); | printf("%d",strlen(s));
for (i=0;i<strlen(s);i++)
  if (strchr("aeiou",s[i])!=NULL)
    s[i]= '*';
cout<<s; | printf("%s",s);
```

strcpy(s,"informatica"); atribuie lui s textul informatica
cout<<strlen(s); afișează 11, lungimea șirului s
for (i=0;i<strlen(s);i++)
if (strchr("aeiou",s[i])!=NULL)
s[i]= '';*
cout<<s;
*Toate vocalele sunt înlocuite de *. => s devine *nf*rm*t*c**
Se afișează:
*11*nf*rm*t*c**

2.27 Subiect 2021. Test antrenament 6. Sub II.3

Variabila `i` este de tip întreg, iar variabilele `s` și `t` permit memorarea câte un șir de cel mult 20 de caractere. Scrieți șirul accesat prin variabila `s` în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"PRASLEA*CEL*VOINIC"); i=0;
while (i<strlen(s))
 if (strchr("ACEI",s[i])!=NULL)
 { strcpy(t,s+i+1); strcpy(s+i,t);}
 else i=i+1;
```

*strcpy(s,"PRASLEA*CEL*VOINIC");*
i=0;
while (i<strlen(s)) if (strchr("ACEI",s[i])!=NULL)
{
strcpy(t,s+i+1);
strcpy(s+i,t);
}
else i=i+1;
Secvența de program parcurge șirul de caractere și șterge toate caracterele care se printer cele ale șirului "ACEI"
*PRSL*L*VON*

2.28 Subiect 2021. Test antrenament 4. Sub II.3

Variabilele `i` și `j` sunt de tip întreg, iar variabilele `s` și `t` permit memorarea unui șir de cel mult 20 de caractere. Scrieți șirul accesat prin variabila `s` în urma executării secvenței de mai jos.

```
strcpy(s,"ABCDUECDA");
i=0; j=strlen(s)-1;
while (i<j)
  if(s[i]==s[j])
  {strcpy(t,s+j+1);strcpy(s+j,t);strcpy(t,s+i+1);strcpy(s+i,t);j=j-2;}
  else { i=i+1; j=j-1; }
```

(6p.)

I	j	s
0	8	~~ABCDUECDA~~
	6	~~BCDUECD~~
1	5	BDUED

2	4	
3	3	

Secvență de program va afișa textul: BDUED

2.29 Subiect 2021. Test antrenament 8. Sub II.3

Variabila `i` este de tip întreg, iar variabilele `s` și `aux` permit memorarea câte unui șir cu cel mult `15` caractere. Scrieți ce se afișează pe ecran în urma executării secvenței de program alăturate. **(6p.)**

```
strcpy(s,"voalata");
cout<<strlen(s); | printf("%d",strlen(s));
i=0;
while (i<strlen(s))
  if (strchr("aeiou",s[i])!=NULL)
  { strcpy(aux,s+i+1);strcpy(s+i,aux); i=i+1; }
  else i=i+2;
cout<<s; | printf("%s",s);
```

i	s
0	~~voalata~~
2	~~volata~~
3	volta
4	volt
5	

Secvență de program va afișa textul: 7volt

2.30 Subiect 2021. Test antrenament 9. Sub II.3

Într-un text cu cel mult 10^2 caractere, cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat și afișează pe ecran, pe linii separate, toate cuvintele sale care conțin o singură vocală distinctă, ca în exemplu. Dacă nu există niciun astfel de cuvânt, se afișează pe ecran mesajul `nu exista`. Se consideră vocale literele din mulțimea `a`, `e`, `i`, `o`, `u`.

Exemplu: pentru textul `a plantat cinci lalele visinii sau rosii` se afișează pe ecran, nu neapărat în această ordine, cuvintele alăturate. **(10p.)**

```
a
plantat
cinci
visinii
```

```cpp
#include <iostream>
#include <string.h>
using namespace std;
//Citesc fraza, extrag cuvintele si numar vocalele distinctechar s[101], voc[]="aeiou";
char s[101], *p, voc[]="aeiou";
int main()
{
    int i,  nrv1=0,ok=0;
    cin.get(s,100);
    p=strtok(s," ");
    while (p)
    {
        nrv1=0;
        for (i=0; i<strlen(voc); i++)
            if (strchr(p, voc[i]))
                nrv1++;
        if (nrv1==1)
        {
            cout<<p<<endl;
            ok=1;
        }
        p=strtok(NULL," ");
    }

    if (ok==0)
        cout<<"nu exista";
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2021 BAC test9 antr_ sirc\bin\Debug\2021 BAC test9 antr_ sirc.exe"
a plantat cinci lalele visinii sau rosii
a
plantat
cinci
visinii

Process returned 0 (0x0)   execution time : 3.161
Press any key to continue.
```

2.31 Subiect 2021. Test antrenament 10. Sub II.3

Scrieți ce se afișează în urma executării secvenței de mai jos, în care variabilele `s` și `t` permit memorarea câte unui șir de cel mult 50 de caractere.

```
strcpy(s,"vorbeste");
s[3]=s[0]; s[5]=s[2]; s[0]=s[1]+1; s[2]=s[1]-2; s[6]=s[4]-1;
strcpy(t,s); t[3]='\0';
cout<<t<<endl<<s+3; | printf("%s\n%s",t,s+3);
```

(6p.)

strcpy(s,"vorbeste"); ***///s: vorbeste***
s[3]=s[0]; ***///s: vorveste***
s[5]=s[2]; ***///s: vorverte***
s[0]=s[1]+1; ***///s: porverte***
s[2]=s[1]-2; ***///s: pomverte***
s[6]=s[4]-1; ***///s: pomverde***
strcpy(t,s); ***///t: pomverde***
t[3]='\0'; ***///t: pom***
cout<<t<<endl<<s+3;

⇨ *Secvența de program va afișa:*

pom
verde

2.32 Subiect 2021. Test antrenament 11. Sub II.3

Variabila `i` este de tip întreg, iar variabila `x` permite memorarea unui şir cu cel mult `100` de caractere. Scrieți ce se afişează în urma executării secvenţei alăturate. **(6p.)**

```
strcpy(x,"bac2021");
cout<<x+3<<endl;  |  printf("%s\n",x+3);
for(i=0;i<strlen(x);i++)
  if(strchr("0123456789",x[i])==0)
    cout<<x[i]<<'!'; | printf("%c! ",x[i]);
```

*strcpy(x,"bac2021"); /// **x: bac2021***
*cout<<x+3<<endl; /// **x+3: ~~bac~~2021 => se afișează 2021***
for(int i=0; i<strlen(x); i++)
if(strchr("0123456789",x[i])==0)
cout<<x[i]<<'!';
/// Pentru x cu valoarea bac2021, caută caracterele care nu sunt cifre și după fiecare afișează !

⇨ *Se afișează:*
2021
b!a!c!

2.33 Subiect 2021. Test antrenament 12. Sub II.3

Variabila `i` este de tip întreg, iar variabila `s` permite memorarea unui șir cu cel mult `50` de caractere, numai litere mari ale alfabetului englez. Scrieți secvența de mai jos înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, să se afișeze pe ecran toate consoanele din șir, iar în locul vocalelor din mulțimea {`O,A,U`} să se afișeze simbolul `*`.

Exemplu: dacă șirul este `CALCULATOARE`, se afișează `C*LC*L*T**RE` **(6p.)**

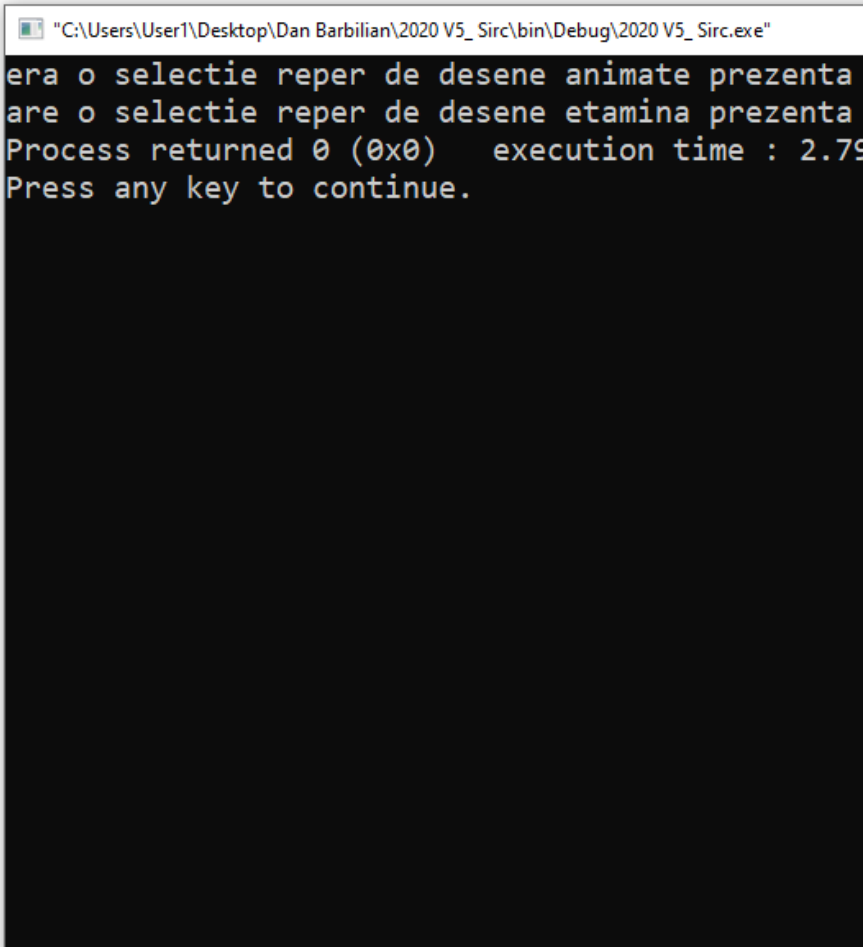
```
for(i=0;i<strlen(s);i++)
if(.............) cout<<...;  |  printf(......);
else cout<<...;  |  printf(......);
```

Secvența devine:
for(int i=0; i<strlen(s); i++) if(***strchr("OAU", s[i])!=NULL***)
cout<<***'*'***;
else cout<<***s[i]***;

2.34 Subiect 2020. Varianta 5

Două cuvinte distincte se numesc **în oglindă** dacă fiecare dintre ele se obține prin citirea literelor celuilalt de la dreapta la stânga.
Exemplu: `animate` și `etamina` sunt în oglindă, iar pentru cuvântul `reper` nu există un cuvânt cu care să fie în oglindă.
Se consideră un text cu cel mult `100` de caractere, în care cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul menționat mai sus și îl transformă în memorie, înlocuind fiecare cuvânt cu număr impar de litere cu acel cuvânt cu care el este în oglindă, dacă acesta există, ca în exemplu. Programul afișează pe ecran textul obținut sau mesajul `nu exista`, dacă în text nu s-a înlocuit niciun cuvânt.
Exemplu: pentru textul `era o selectie reper de desene animate prezenta`
se obține textul `are o selectie reper de desene etamina prezenta`
iar pentru textul `un reper pentru desene`
se afișează pe ecran mesajul `nu exista` **(10p.)**

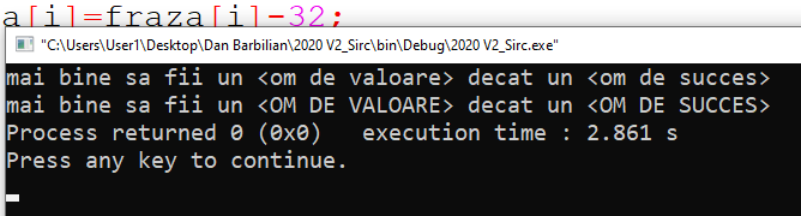
```cpp
#include <string.h>
using namespace std;
char s[101],  fraza [101], frazan[101]="";
int n,  gasit=0;
int main()
{
    char *p;
    int i,ok;
    cin.get(fraza, 100);
    p = strtok(fraza," ");
    while (p!=NULL)
    {
        n = strlen(p);
        if (n%2==1)
        {
            int ok=0;
            strcpy(s,p);
            for (i=0; i<n/2; i++)
                swap(s[i],s[n-i-1]);
            if (strcmp(s,p)!=0)
            {
                gasit=1;
                strcat (frazan, s);
            }
            else
                strcat (frazan, p);
        }
        else
            strcat (frazan, p);
        strcat(frazan," ");
        p=strtok(NULL," ");
    }
    strcpy(fraza, frazan);
    if (gasit==0)
        cout<<"nu exista";
    else
        cout<<frazan;
    return 0;
}
```



2.35 Subiect 2020. Varianta 2

Numim **citat** într-un text o secvență de caractere din acel text care începe cu un caracter < și se termină cu un caracter >, celelalte caractere ale secvenței fiind diferite de < și >.
Un text de cel mult 100 de caractere (litere mici ale alfabetului englez, spații și caracterele < și >) conține cel puțin un citat. Textul nu conține alte caractere < şi > decât cele care mărginesc citatele, și oricare două citate nu au nici caractere < şi > și nici alte caractere în comun.
Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat și îl transformă în memorie prin înlocuirea tuturor literelor mici cuprinse în citate cu literele mari corespunzătoare, celelalte rămânând nemodificate, ca în exemplu. Programul afișează pe ecran textul obținut.
Exemplu: pentru textul `mai bine sa fii un <om de valoare> decat un <om de succes>`
se afișează `mai bine sa fii un <OM DE VALOARE> decat un <OM DE SUCCES>` **(10p.)**

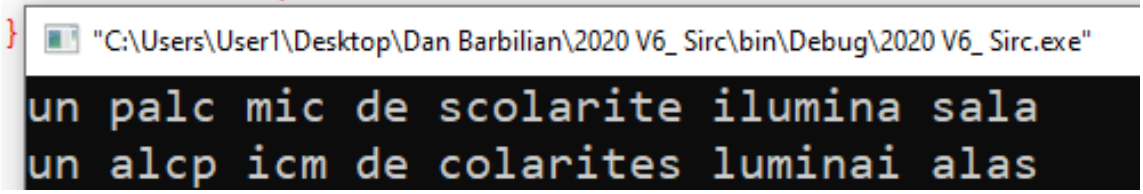
```
#include <iostream>
#include <string.h>
using namespace std;
char s[101],  fraza [101], frazan[101]="";
int n,  gasit=0;
int main()
{
    char *p;
    int i,ok;
    cin.get(fraza, 100);
    for (int i=0; i<strlen(fraza); i++)
        if (fraza[i]=='<')
            ok=1;
        else if (fraza[i]=='>')
            ok=0;
        else if (ok==1 && fraza[i]!=' ')
            fraza[i]=fraza[i]-32;
    cout<<fraza;
    return 0;
}
```



2.36 Subiect 2020. Varianta 6

Numim **rotire spre stânga** a unui cuvânt format din cel puțin trei litere operația prin care prima sa literă se mută la final, iar toate celelalte litere se mută cu o poziție spre stânga.
Exemplu: în urma rotirii spre stânga a cuvântului `ilumina` se obține cuvântul `luminai`.
Un text are cel mult 100 de caractere, iar cuvintele sale sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul menționat mai sus și îl transformă în memorie prin rotirea spre stânga a fiecărui cuvânt al său format din cel puțin trei litere, ca în exemplu. Programul afișează pe ecran textul obținut sau mesajul `nu exista`, dacă în text nu există niciun cuvânt de cel puțin trei litere.
Exemplu: pentru textul `un palc mic de scolarite ilumina sala`
se afișează pe ecran `un alcp icm de colarites luminai alas` **(10p.)**

```cpp
#include <iostream>
#include <string.h>
using namespace std;
char s[101],  fraza [101], frazan[101]="";
int n,  gasit=0;
int main()
{
    char *p;
    int i,ok;
    cin.get(fraza, 100);
    p = strtok(fraza," ");
    while (p!=NULL)
    {
        n = strlen(p);
        if (n>=3)
        {
            char c;
            c=p[0];
            for (i=1; i<n; i++)
                swap(p[i-1],p[i]);
            p[n-1]=c;
            strcat (frazan, p);
        }
        else
            strcat (frazan, p);
        strcat(frazan," ");
        p=strtok(NULL," ");
    }
    cout<<frazan;
    return 0;
}
```



2.37 Subiect 2020. Varianta MODEL

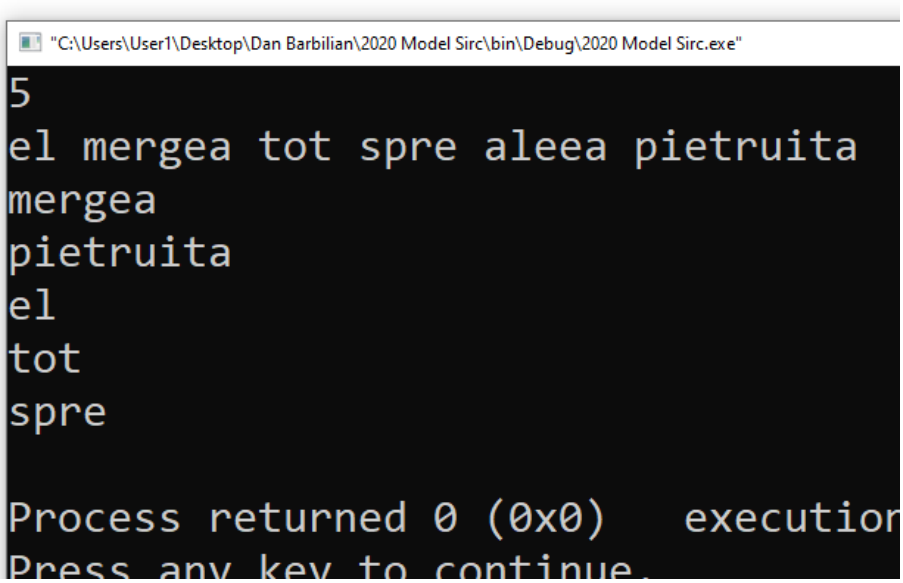
Un text are cel mult 100 de caractere, iar cuvintele sale sunt formate numai din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un număr natural n ($n \in [1, 10^2]$), apoi un text de tipul precizat mai sus, și afișează pe ecran cuvintele acestuia, pe rânduri separate, astfel încât primele poziții să fie ocupate de mulțimea formată de cele care au cel puțin n litere, iar următoarele poziții, în continuarea acestora, să fie ocupate de mulțimea celorlalte cuvinte.

Cuvintele din aceeași mulțime sunt afișate într-o ordine oarecare, iar dacă una dintre cele două mulțimi este vidă, se afișează pe ecran doar mesajul `nu exista`.

Exemplu: pentru n=5 și textul `el mergea tot spre aleea pietruita` datele afișate pot fi cele alăturate. **(10p.)**

```
mergea
aleea
pietruita
el
tot
spre
```

```
#include <string.h>
using namespace std;
char s[101][101],  fraza [101];
int n,  ok1=0, ok2=0;
int main()
{
    char *p;
    int i;
    cin>>n;
    cin.get();
    cin.get(fraza, 100);
    p = strtok(fraza," ");
    ///extrag cuvintele si le pun in vector
    ///verific daca[ am cel putin un cuvant de lungime <n sau mai mare decat n
    ///parcurg vectororul si afisez cuvintele conform cerintei
    i=0;
    while (p!=NULL)
    {
        i++;
        if (strlen(p)<n)
            ok1=1;
        else if (strlen(p)>n)
            ok2=1;
        strcpy(s[i],p);
        p=strtok(NULL," ");
    }
    if (ok1==0 || ok2==0)
        cout<<"nu exista";
    else
    {
        for (int j=1;j<=i;j++)
            if (strlen(s[j])>n)
            cout<<s[j]<<endl;
        for (int j=1;j<=i;j++)
            if (strlen(s[j])<n)
            cout<<s[j]<<endl;
    }
    return 0;
}
```



2.38 Subiect 2020. Test Antrenament 1

În secvenţa alăturată, variabila `a` memorează un şir cu cel mult `100` de caractere, iar variabilele `i` şi `k` sunt de tip întreg.
Scrieţi ce se afişează pe ecran în urma executării secvenţei.

```
k='a'-'A';
strcpy(a,"VIcToriE");
cout<<strlen(a); | printf("%d", strlen(a));
for(i=0;i<strlen(a);i++)
  if(a[i]>='A' && a[i]<='Z') a[i]=a[i]+k;
  else a[i]=a[i]-k;
cout<<a;  |  printf("%s",a);
```

k='a'-'A'; ///k=32
*strcpy(a,"VIcToriE"); ///**a: VIcToriE***
*cout<<strlen(a); /// **Se afiseaza 8;***
for(i=0;i<strlen(a);i++)

if(a[i]>='A' && a[i]<='Z') a[i]=a[i]+k; else a[i]=a[i]-k;
///Literele mari sunt transformate in litere mici si literele mici in litere mari prin adunarea/ scaderea lui 32 din cod
cout<<a; ***/// Se afiseaza viCtORIe***
Secvența de program va determina afișarea textului:
8viCtORIe

2.39 Subiect 2020. Test Antrenament 2

Un text are cel mult `100` de caractere și este format din cuvinte şi numere, separate prin câte un spațiu. Cuvintele sunt formate numai din litere ale alfabetului englez. Toate numerele sunt reale şi sunt formate numai din parte întreagă sau din parte întreagă şi parte fracţionară, separate prin virgulă (`,`), numerele negative fiind precedate de semnul minus (`-`).
Scrieţi un program C/C++ care citeşte de la tastatură textul, pe care îl transformă, eliminând din componenţa sa toate numerele negative. Programul afişează apoi pe ecran textul obţinut.
Exemplu: pentru textul
`2,7 minus 3,5 minus 2 egal 2,7 plus -3,5 plus -2 egal -0,2 rezultat`
se va afişa pe ecran textul:
`2,7 minus 3,5 minus 2 egal 2,7 plus  plus  egal  rezultat` **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
char  fraza [101], frazan[101]="";
int n,  gasit=0;
int main()
{
    char *p;
    int i,ok;
    cin.get(fraza, 100);
    p = strtok(fraza," ");
    while (p!=NULL)
    {
        if (p[0]!='-')
            strcat (frazan, p);
        strcat(frazan," ");
        p=strtok(NULL," ");
    }
    strcpy(fraza, frazan);
    cout<<frazan;
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 Test antr 2\bin\Debug\2020 Test antr 2.exe"
2,7 minus 3,5 minus 2 egal 2,7 plus -3,5 plus -2 egal -0,2 rezultat
2,7 minus 3,5 minus 2 egal 2,7 plus  plus  egal  rezultat
```

2.40 Subiect 2020. Test Antrenament 3

Variabila **p** este de tip întreg, iar variabila **s** memorează un şir de cel mult **20** de caractere, numai litere mari ale alfabetului englez. Fără a utiliza alte variabile, scrieţi o secvență de instrucțiuni în urma executării căreia să se afişeze pe ecran toate literele şirului memorat de variabila **s**, cu excepţia vocalei **A**, dacă în șirul inițial aceasta este alături de vocala **I**. Literele se afişează în ordinea apariţiei lor în şir.
Exemplu: dacă şirul memorat în variabila **s** este **ALIANTA** sau **ALAINTA** se va afişa **ALINTA**. **(6p.)**

```
#include <iostream>
#include <string.h>
using namespace std;

int main()
{
    char s[21]="ALIANTA";
    int p;
    ///ALIANTA sau ALAINTA se va afisa ALINTA
    if ((s[0]=='A' && s[1]!='I')|| (s[0]!='A'))
        cout << s[0];
    for (p=1; p<strlen(s)-1; p++)
        if ((s[p]=='A' && s[p+1]!='I' && s[p-1]!='I')|| (s[p]!='A'))
            cout << s[p];
    if ((s[strlen(s)-1]=='A' && s[strlen(s)-2]!='I')|| (s[strlen(s)-1]!='A'))
        cout << s[strlen(s)-1];
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 Test antr3\bin\Debug\2020 Test antr3.exe"
ALINTA
```

2.41 Subiect 2020. Test Antrenament 4

Variabilele **i** și **j** sunt de tip întreg, iar variabila **s** poate memora un şir de cel mult **20** de caractere. Scrieți șirul memorat de variabila **s** în urma executării secvenței de mai jos.

```
strcpy(s,"optsprezece"); i=0; j=strlen(s)-1;
while(i<j)
{ if(strchr("aeiou",s[i])==NULL && strchr("aeiou",s[j])!=NULL)
  {  s[i]=s[i]+1; s[j]=s[j]-1;}
  i=i+1;j=j-1;
}
```

(6p.)

*strcpy(s,"optsprezece"); ///**s: optsprezece***
i=0; j=strlen(s)-1;
while(i<j)
{ if(strchr("aeiou",s[i])==NULL && strchr("aeiou",s[j])!=NULL)
{ s[i]=s[i]+1; s[j]=s[j]-1;}
i=i+1;j=j-1;
}
Secvența de algoritm parcurge șirul s prin i de la stânga spre dreapta și prin j de la dreapta spre stânga până se întâlnesc. Dacă s[i] este consoană și s[j] este vocală s[i] devine următoarea literă din alfabet și s[j] devine litera precedentă din alfabet.
*s devine: **opusqrdzdce***

2.42 Subiect 2020. Test Antrenament 5

Un text cu cel mult **100** de caractere conține cuvinte și numere, separate prin câte un spațiu. Cuvintele sunt formate numai din litere mici ale alfabetului englez, iar numerele sunt reale, pozitive, cu partea zecimală și partea întreagă separate prin simbolul virgulă, sau numai cu partea întreagă, ca în exemplu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat și afișează pe ecran numărul de valori întregi din text.

Exemplu: pentru textul
grus leucogeranus are 1,40 m inaltime si traieste intre 30 si 40 de ani
se afișează pe ecran **2** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
char s[101], *p;

int main()
{
    int nr=0;
     cin.get(s,100);
     p=strtok(s," ");
     while(p!=NULL)
     {
         if (strspn(p,"0123456789")==strlen(p))
            nr++;
         p=strtok(NULL," ");
     }
     cout<<nr;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 Test 5 antr Sc\bin\Debug\2020 Test 5 antr Sc.exe"
grus leucogeranus are 1,40 m inaltime si traieste intre 30 si 40 de ani
2
```

2.43 Subiect 2020. Test Antrenament 6

Într-un text cu cel mult **100** de caractere, cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul menționat și afișează pe ecran, pe linii separate, toate cuvintele sale pentru care numărul de vocale este strict mai mic decât numărul de consoane. Dacă nu există niciun astfel de cuvânt, se afișează pe ecran mesajul **nu exista**. Se consideră vocale literele din mulțimea **a**, **e**, **i**, **o**, **u**.

Exemplu: pentru textul **ei au plantat tamarix ea a adus iasomie**
se afișează pe ecran, nu neapărat în această ordine, cuvintele alăturate. **(10p.)**

```
plantat
tamarix
```

```
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
    int nv=0,nc=0,ok=0;
    char s[101],*p;
    cin.get(s,100);
    p=strtok(s," ");
    while(p!=NULL)
    {
        nc=0;
        nv=0;
        for(int i=0; i<strlen(p); i++)
            if (strchr("aeiou",p[i])!=NULL)
                nv++;
            else
                nc++;
        if(nc>nv)
        {
            ok=1;
            cout<<p<<endl;
        }

        p=strtok(NULL," ");
    }
    if (ok==0)
        cout<<"nu exista";
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 Test antr 6 sir c\bin\Debug\2020 Test antr 6 sir c.exe"
ei au plantat tamarix ea a adus iasomie
plantat
tamarix

Process returned 0 (0x0)   execution tim
Press any key to continue.
```

2.44 Subiect 2020. Test Antrenament 7

Variabila **s** poate memora un șir de cel mult 20 de caractere. Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"stilou");
cout<<s+4<<endl; | printf("%s\n",s+4);
s[0]=s[0]-1; s[1]=s[0]-3;
s[2]=s[0]+1; s[3]=s[0]+3;
s[4]='\0';
cout<<s; | printf("%s",s);
```

*strcpy(s,"stilou"); ///**s: stilou***
*cout<<s+4<<endl; ///se afișează: **ou***
*s[0]=s[0]-1; ///**s: rtilou***
*s[1]=s[0]-3; ///**s: rolou***
*s[2]=s[0]+1; ///**s: rosou***
*s[3]=s[0]+3; ///**s: rosuu***
*s[4]='\0'; ///**s: rosu***
cout<<s;
Secvența determină afișarea:
ou
rosu

2.45 Subiect 2020. Test Antrenament 8

Variabila **i** este de tip întreg, iar variabila **s** poate memora un șir de cel mult **20** de caractere. Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"stilou"+4);
cout<<s<<endl; | printf("%s\n",s);
strncpy(s,"stilou",4); s[4]='\0';
for(i=0;i<4;i++)
  if(i%2==0) s[i]=s[0]+i-1;
  else s[i]=s[0]+3*(2*i/3-1);
cout<<s; | printf("%s",s);
```

strcpy(s,"stilou"+4);///s: ou
cout<<s<<endl;///ou
strncpy(s,"stilou",4);///s: stil
s[4]='\0';///s: stil
for(i=0; i<4; i++)
if(i%2==0) s[i]=s[0]+i-1;
else s[i]=s[0]+3(2*i/3-1);*
cout<<s;///rosu
Secvența determină afișarea:
ou
rosu

2.46 Subiect 2020. Test Antrenament 9

Un cuvânt este **prefix** al unui alt cuvânt dacă se obține din acesta, prin eliminarea ultimelor sale litere. Scrieți un program C/C++ care citește de la tastatură un număr natural **n** (**n∈[2,20]**) și apoi **n** cuvinte distincte, fiecare fiind format din cel mult **20** de caractere, numai litere mici ale alfabetului englez.

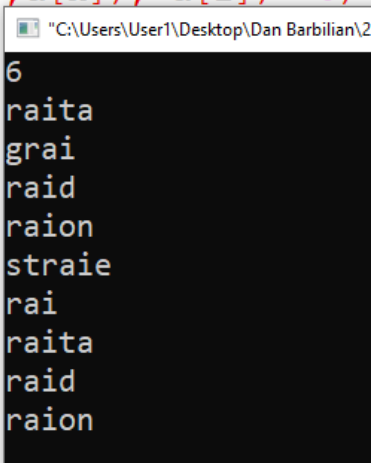
La introducerea datelor, după fiecare cuvânt se tastează Enter. Programul afișează pe ecran, separate prin câte un spațiu, cuvintele care îl au drept prefix pe ultimul cuvânt citit. Dacă nu există astfel de cuvinte, se afișează pe ecran mesajul **nu exista**.
Exemplu: dacă **n=6** și se citesc cuvintele alăturate, pe ecran se afișează
raita raid raion **(10p.)**

```
raita
grai
raid
raion
straie
rai
```

```cpp
#include <iostream>
#include <string.h>t
using namespace std;

char a[21][21];
int n,ok=0;
int main()
{
    cin>>n;
    int i;
    for (i=1; i<=n; i++)
        cin>>a[i];
    for(i=1; i<=n-1; i++)
        if (strcmp(strstr(a[i],a[n]), a[i])==0)
        {
            cout<<a[i]<<endl;
            ok=1;
        }

    if(ok==0)
        cout<<"nu exista";
}
```



2.47 Subiect 2020. Test Antrenament 10

Într-un text cu cel mult 10^2 caractere cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul menționat, pe care îl modifică în memorie, inserând cuvântul **succes** între oricare două cuvinte ale sale care se termină cu aceeași literă. Cuvântul inserat este separat prin câte un spațiu de cuvintele vecine. Textul transformat este afișat pe ecran, iar dacă nu există perechi de astfel de cuvinte, se afișează pe ecran mesajul **nu exista**.

Exemplu: dacă textul citit este **testez validez utilizez date corecte acum**

se obține textul **testez succes validez succes utilizez date succes corecte acum** **(10p.)**

```cpp
#include <iostream>
#include <string.h>
using namespace std;
char s[200],aux[200];
int main()
{
    cin.getline(s,100);
    int i,j,p1,p2,ok=0;
    ///caut primul spatiu, pastrez pozitia lui; caut urmatorul spatiu
    ///si vad daca in fata lor am aceeasi litera, in caz afirmativ inserez succes
    ///retin pozitia acestui spatiu si caut mai departe
    ///procedez la fel pana la final
    for(i=0;i<strlen(s);i++)
        if(s[i]==' ')
    {
        p1=i;
        break;
    }
    for(i=i+1;i<strlen(s);i++)
         if(s[i]==' ')
    {
        p2=i;
        if(s[p1-1]==s[p2-1])
        {
            ///inserez
            strcpy(aux,s+p1+1);
            strcpy(s+p1+1,"succes ");
            strcpy(s+p1+8,aux);
            ok=1;
            i=i+8;
            p1=p2+7;
        }
       else p1=p2;
    }
    if (ok==1)cout<<s;
    else
        cout<<"nu ";
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020_test 10 antr_sirc\bin\Debug\2020_test 10 antr_sirc.exe"
testez validez utilizez date corecte acum
testez succes validez succes utilizez date succes corecte acum
Process returned 0 (0x0)   execution time : 7.225 s
Press any key to continue.
```

///caut primul spatiu, pastrez pozitia lui; caut urmatorul spatiu
///si vad daca in fata lor am aceeasi litera, in caz afirmativ inserez succes
///retin pozitia acestui spatiu si caut mai departe
///procedez la fel pana la final

2.48 Subiect 2020. Test Antrenament 11

Într-un text cu cel mult **10²** caractere, cuvintele sunt formate din litere mici și mari ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat, pe care îl transformă, astfel încât fiecare cuvânt să aibă prima literă mare, și toate celelalte litere mici. Textul obținut se afișează pe ecran.
Exemplu: dacă de la tastatură se introduce textul **`ABIA aSTept sa Merg lA scoala`**
se obține textul **`Abia Astept Sa Merg La Scoala`** **(10p.)**

```cpp
#include <iostream>
#include <cstring>
using namespace std;
int main()
{
    char s[101];
    cin.get(s, 101);
    if(s[0]>='a'&& s[0]<='z')
        s[0] = (s[0]-32);
    for (int i = 1; i < strlen(s); ++i)
        if(s[i]>='a'&& s[0]<='z' && s[i-1]==' ')
            s[i] = (s[i]-32);
        else if  (s[i]>='A'&& s[i]<='Z')
            s[i]= (s[i]+32);
    cout << s << " ";
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 test 11 antr src\bi
ABIA aSTept sa Merg lA scoala
Abia Astept Sa merg La Scoala
```

2.49 Subiect 2020. Test Antrenament 12

Într-un text cu cel mult 10^2 caractere, cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat, pe care îl transformă în memorie, eliminând numai ultima vocală care apare în text, ca în exemplu. Programul afișează pe ecran textul obținut sau mesajul **nu exista**, dacă în text nu există nicio vocală. Se consideră vocale literele **a**, **e**, **i**, **o**, **u**.

Exemplu: dacă se citește textul: **cuvantul ritm poate fi tradus rhythm**
se obține textul **cuvantul ritm poate fi trads rhythm** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;

int main()
{

    char s[101], t[101],  voc[] = {"aeiou"};
    cin.get(s, 101);
    for (int i = strlen(s) - 1; i >= 0; --i)
    {
        if (strchr(voc,s[i]))
        {
            strcpy(t, s + i + 1);
            strcpy(s + i, t);
            cout << s;
            break;
        }
    }
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 test 12 antr_src\bin\Debug\2020 test 12 antr_src.exe"
cuvantul ritm poate fi tradus rhythm
cuvantul ritm poate fi trads rhythm
```

2.50 Subiect 2020. Test Antrenament 13

Variabila **i** este de tip întreg, iar variabila **s** poate memora un șir de cel mult **20** de caractere. Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s,"stilou");
cout<<s+4<<endl; | printf("%s\n",s+4);
for(i=0;i<4;i++)
  s[i]=s[0]+(i-1)*(1-i%2)+3*(2*i/3-1)*(i%2);
s[4]='\0';
cout<<s; | printf("%s",s);
```

strcpy(s,"stilou");///s: stilou
cout<<s+4<<endl;///s+4: ou
for(i=0; i<4; i++)
s[i]=s[0]+(i-1)(1-i%2)+3*(2*i/3-1)*(i%2);*
///s[0]:r, s[1]:o, s[2]:s, s[3]:u
s[4]='\0';
cout<<s;///afiseaza rosu
Secvența determină afișarea:
ou
rosu

2.51 Subiect 2020. Test Antrenament 14

Un text are cel mult `100` de caractere și este format din cuvinte și numere, separate prin câte un spațiu. Cuvintele sunt formate numai din litere ale alfabetului englez. Toate numerele sunt reale și sunt formate numai din parte întreagă sau din parte întreagă și parte fracționară, separate prin virgulă (`,`), numerele negative fiind precedate de semnul minus (`-`). Cel puțin unul dintre numerele reale este negativ. Scrieți un program C/C++ care citește de la tastatură textul, pe care îl transformă în memorie, înlocuind fiecare număr negativ cu valoarea sa absolută. Programul afișează apoi pe ecran textul obținut.
Exemplu: pentru textul **`Modul de -3,24 este 3,24 si modul de -15 este 15`**
se va afișa pe ecran textul: **`Modul de 3,24 este 3,24 si modul de 15 este 15`** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
char  fraza [101], frazan[101]="";
int n,  gasit=0;
int main()
{
    char *p;
    int i,ok;
    cin.get(fraza, 100);
    p = strtok(fraza," ");
    while (p!=NULL)
    {
        if (p[0]=='-')
            strcat (frazan, p+1);
        else
            strcat(frazan,p);
        strcat(frazan," ");
        p=strtok(NULL," ");
    }
    strcpy(fraza, frazan);
    cout<<frazan;
    return 0;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 test antr src 14\bin\Debug\2020 test antr src 14.exe"
Modul de -3,24 este 3,24 si modul de -15 este 15
Modul de 3,24 este 3,24 si modul de 15 este 15
```

2.52 Subiect 2020. Test Antrenament 16

Variabilele **`s1`** și **`s2`** pot memora câte un șir cu cel mult `20` de caractere. Scrieți ce se afișează în urma executării secvenței alăturate. **(6p.)**

```
strcpy(s1,"bacalaureat2020");
cout<<strlen(s1); | printf("%d",strlen(s1));
strcpy(s2,s1+11); strcpy(s1+3,s2);
cout<<s1; | printf("%s",s1);
```

*strcpy(s1,"bacalaureat2020");///**s1: bacalaureat2020***
*cout<<strlen(s1); ///**lungimea lui s1 este 15***
*strcpy(s2,s1+11);///**s1: 2020***
*strcpy(s1+3,s2);///**s1 devine bac2020***
*cout<<s1;///**bac2020***
Secvența determină afișarea:
15bac2020

2.53 Subiect 2020. Test Antrenament 17

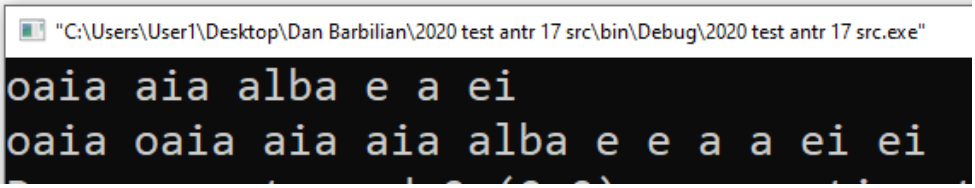
Într-un text cu cel mult 10^2 caractere cuvintele sunt formate din litere mici ale alfabetului englez și sunt separate prin câte un spațiu. Scrieți un program C/C++ care citește de la tastatură un text de tipul menționat, pe care îl modifică în memorie, duplicând fiecare cuvânt format numai din vocale. Cuvântul duplicat este separat prin câte un spațiu de cuvintele vecine. Textul transformat este afișat pe ecran, iar dacă nu există astfel de cuvinte, se afișează pe ecran mesajul **nu exista**.

Exemplu: dacă textul citit este **oaia aia alba e a ei**

se obține textul **oaia oaia aia aia alba e e a a ei ei** **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
char a[101],b[200]="";
int main()
{
    cin.getline(a,100);
    char *p;
    int ok,k=0,i;
    p=strtok(a," ");
    while(p)
    {
        ok=1;
        for(i=0; i<strlen(p); i++)
            if(strchr("aeiou",p[i])==NULL)
                ok=0;
        //cout<<ok<<" "<<p<<endl;
        if(ok==0)
        {

            strcat(b,p);
            strcat(b," ");
        }
        else
        {
            strcat(b,p);
            strcat(b," ");
            strcat(b,p);
            strcat(b," ");
            k=1;
        }
        p=strtok(NULL," ");
    }
   strcpy(a,b);
    if(k==1)
        cout<<a;
    else
        cout<<"nu exista";
}
```



2.54 Subiect 2020. Test Antrenament 18

Un cuvânt este **sufix** al unui alt cuvânt dacă se obține din acesta, prin eliminarea primelor sale litere. Scrieți un program C/C++ care citește de la tastatură două numere naturale **n** și **k** (**n∈[2,20]**, **k∈[1,n]**) și apoi **n** cuvinte distincte, fiecare fiind format din cel mult **20** de caractere, numai litere mici ale alfabetului englez.

La introducerea datelor, după fiecare cuvânt se tastează Enter. Programul afișează pe ecran, separate prin câte un spațiu, cuvintele care îl au drept sufix pe al **k**-lea cuvânt citit, ca în exemplu. Dacă nu există astfel de cuvinte, se afișează pe ecran mesajul **nu exista**.

Exemplu: dacă **n=7**, **k=3** și se citesc cuvintele alăturate, pe ecran se afișează

paratirisi hiritisi **(10p.)**

```
isihast
paratirisi
isi
meremetisire
acolisitor
hiritisi
paraponisit
```

```
#include <iostream>
#include<string.h>
using namespace std;
int n,k;
char a[21][21],b[21];
int main()
{
    cin>>n>>k;
    int i,j,ok,x=0;
    for(i=1; i<=n; i++)
    {
        cin>>a[i];
        if(i==k)
            strcpy(b,a[i]);
    }
    for(i=1; i<=n; i++)
    {
        j=strlen(a[i])-strlen(b);
        if(strcmp(a[i]+j,b)==0&&i!=k)
        {
            cout<<a[i]<<" ";
            x=1;
        }
    }

    if(x==0)
        cout<<"nu exista";
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 test antr 18 src\bin\Debug\2020 test antr 18 src.exe"
7 3
isihast paratirisi isi meremetisire acolisitor hiritisi paraponisit
paratirisi hiritisi
Process returned 0 (0x0)   execution time : 12.719 s
```

2.55 Subiect 2020. Test Antrenament 19

Variabilele **i** și **j** sunt de tip întreg, iar variabila **s** poate memora un șir de cel mult **20** de caractere. Scrieți șirul memorat de variabila **s** în urma executării secvenței de mai jos.

```
strcpy(s,"informatie"); n=strlen(s)-1;
for(i=0;i<n/2;i++)
  if(strchr("aeiou",s[i])!=NULL && strchr("aeiou",s[n-i])!=NULL)
  {  s[i]=s[i+1]; s[n-i]=s[n-i-1];}
```

(6p.)

*strcpy(s,"informatie");///**s: informatie***
*n=strlen(s)-1; ///**n: 10-1=9***
for(i=0; i<n/2; i++)
if(strchr("aeiou",s[i])!=NULL && strchr("aeiou",s[n-i])!=NULL)
{
s[i]=s[i+1];
s[n-i]=s[n-i-1];

}

Instructiunea for ia elementele simetrice, s[0] cu s[9], s[1] cu s[8], s[2] cu s[7], s[3] cu s[6], s[4] cu s[5]. Dacă ambele sunt vocale, prima valoarea din pereche ia valoarea urmatorului element din s, iar al doilea element din pereche ia ca valoare caracterul aflat în stanga sa, în șir.

Secvența determină formarea șirului s:

Nnfrrmmtii

2.56 Subiect 2020. Test Antrenament 20

Un text cu cel mult `100` de caractere conține cuvinte și numere, separate prin câte un spațiu. Cuvintele sunt formate numai din litere mici ale alfabetului englez, iar numerele sunt reale, pozitive, cu partea întreagă și partea zecimală separate prin simbolul virgulă, sau numai cu partea întreagă, ca în exemplu. Scrieți un program C/C++ care citește de la tastatură un text de tipul precizat și îl transformă în memorie, înlocuind fiecare număr real cu partea întreagă a acestuia.

Exemplu: pentru textul

`partea intreaga a lui 5,75 este egala cu a lui 5,25 si cu a lui 5 si este 5`

se afișează pe ecran

`partea intreaga a lui 5 este egala cu a lui 5 si cu a lui 5 si este 5` **(10p.)**

```
#include <iostream>
#include <string.h>
using namespace std;
int main()
{
    char a[101];
    int i,j;
    cin.getline(a,100);
    for (i=1; i<strlen(a); i++)
        if (a[i]==',' &&  strchr("0123456789",a[i-1])!=NULL)
        {
            ///parcurg cu j pana dau de spatiu sau ajung la final
            j=i;
            while(j<strlen(a)&& a[j]!=' ')
                j++;
            strcpy(a+i,a+j);
        }
    cout<<a;
}
```

```
"C:\Users\User1\Desktop\Dan Barbilian\2020 test antr 20 src\bin\Debug\2020 test antr 20 src.exe"
partea intreaga a lui 5,75 este egala cu a lui 5,25 si cu a lui 5 si este 5
partea intreaga a lui 5 este egala cu a lui 5 si cu a lui 5 si este 5
```