COLEGIUL NAȚIONAL “ROMAN-VODĂ” ROMAN

ȘIRURI DE CARACTERE

LUCRARE PENTRU OBȚINEREA ATESTATULUI DE COMPETENȚE PROFESIONALE

elev: Donea Fernando-Emanuel

Coordonator,
prof. Florin Moldovanu

APRILIE 2024

Cuprins

Argument

Șirurile de caractere reprezintă o componentă fundamentală a programării fiind utilizate pentru a manipula text, date și informații. Cunoașterea aprofundată a șirurilor de caractere este esențială pentru o varietate de domenii de programare, precum dezvoltarea de aplicații web și desktop, inteligență artificială și prelucrarea datelor. În C++ există mai multe modalități de a reprezenta șirurile de caractere.

Astfel, am decis să realizez un program în limbajul C++ cu diferite aplicații despre șirurile de caractere reprezentate ca tablouri unidimensionale cu elemente de tip char.

Șiruri de caractere

I. Descriere generală. Noțiuni teoretice

O constantă de tip șir de caractere de declară între două caractere “ iar în memoria internă, o constantă de acest tip este reținută sub forma unui vector de caractere. Fiecare componentă a șirului (începând cu cea de indice 0) reține codul ASCII al caracterului pe care îl memorează.

Convenția este ca ultimul octet să rețină 0 ('\0' – caracterul cu codul ASCII 0, numit și caracter nul.). Caracterul nul este memorat automat. Totuși, trebuie rezervate *lungimea_sirului*+1 caractere char (+1 pentru caracterul nul).

1. Declararea unui șir de caractere

În C++, un șir de caractere se declară astfel:

```
char s[11];
```

S-a declarat un șir care poate memora maxim 11 caractere, cu indici 0 1 ... 10. Șirul *s* poate memora cel mult 10 caractere utile, după ultimul caracter util fiind memorat caracterul '\0'.

Limbajul C++ permite inițializarea unui tablou de caractere printr-o constanta șir, care include automat caracterul nul.

Exemplu:

```
char vect[12]=”informatica”;

char vect[]=”informatica”;// compilatorul face calculul numărului de octeți necesari

char vect[100]=”calculator”; // s-au rezervat mai mulți octeți decât era necesar)
```

Șirurile de caractere sunt de fapt *tablouri de caractere*, care au ca ultim element un terminator de șir, caracterul null.

Exemplu:

```
char tc[5] = {’a’, ’b’ , ’c’ , ’d’, ’e’};    // tablou de caractere

char sc[5] = {’a’, ’b’, ’c’, ’d’, ’\0’}; // șir de caractere cu elementele abcd
```

Ultima inițializare este echivalentă cu:

```
char sc[5] = ”abcd”;        // sau char sc[] = ”abcd”;

char sc1[5] = ”abcd”;

char s[10];

cout<<sc<<endl;  // afișează abcd

cout<<tc<<endl;   // eroare: tabloul de caractere nu
conține terminatorul de sir, deci nu poate fi afișat ca sir

cout<<s<<endl;    // eroare: tablou neinițializat

cout<<sc1[0];     // afișează primul caracter din șirul sc1

cout<<sc1[2];     // afișează al treilea element din șirul
sc1

sc1[1]=’K’;       // elementului din sir de indice 1 i se
atribuie valoarea ‘K’;
```

2. Citirea și afișarea unui șir de caractere

Pentru citirea unui șir de caractere se poate folosi operatorul **>>** de extracție din stream:

```
cin>> s;
```

În acest mod, datorită specificului operatorului **>>** nu se pot citi șiruri care conțin spații – se vor citi caracterele până la primul spațiu, fără acesta.

Pentru a citi șiruri care conțin spații, putem folosi metoda **getline** a obiectului **cin** sau alt obiect de tip **istream**:

```
istream& getline (char s*, streamsize n) ;
```

Se vor citi în șirul **s** caracterele din stream-ul de intrare (de la tastatură) până la apariția caracterului sfârșit de linie **'\n'**, dar nu mai mult de **n-1** caractere. Caracterul **'\n'** nu va fi adăugat la șirul s, dar va fi extras din stream. De exemplu:

```
cin.getline(s , 11);
```

Afișarea unui șir se poate face cu operatorul **<<** de inserție în stream:

```
cout << s << endl;
```

3. Referirea unui caracter din șir. Parcurgerea unui șir de caractere

Deoarece șirurile de caractere sunt de fapt tablouri, pentru referirea unui caracter din șir se folosește operatorul [].

Exemplu:

```
char s[]=”abac”;//șirul constă din 5 caractere: cele 4 litere și caracterul NULL '\n'

cout<<s[3]; //c

s[1]='r';

cout<<s; //arac
```

În numeroase situații este necesară analizarea fiecărui caracter din șir. Pentru aceasta este necesară o parcurgere a șirului; aceasta se face similar cu parcurgerea unui tablou oarecare. Diferența constă în faptul că, pentru șirul de caractere nu se cunoaște explicit lungimea. Ea poate fi determinată cu funcția strlen, dar putem controla parcurgerea șirului știind că după ultimul caracter valid din șir apare caracterul nul '\0'.

Exemplu1:

```
char s[11];

cin >> s; // se citește un cuvânt , fără spații

int i = 0;

while(s[i] != '\0')
{
     cout << s[i] << " ";

     i ++;
}
```

Exemplu2:

```
char s[11];

cin >> s; // se citește un cuvânt , fără spații

for(int i = 0 ; s[i] ; i ++)
{
     cout << s[i] << " ";
}
```

II. Funcții pentru caractere

Următoarele funcții au ca parametri valori numerice, reprezentând codul ASCII al unor caractere. Prototipul lor se află în header-ul **cctype**.

a) Funcția isalnum

```
int isalnum(int ch);
```

Funcția **isalnum** verifică dacă un caracter este alfanumeric (cifră, literă mare, literă mică). Returnează o valoare diferită de zero dacă parametrul este alfanumeric, 0 în caz contrar.

b) Funcția isalpha

```
int isalpha(int ch);
```

Funcția **isalpha** verifică dacă un caracter este alfabetic (literă mare, literă mică). Returnează o valoare diferită de zero dacă parametrul este alfabetic, 0 în caz contrar.

c) Funcția islower

```
int islower(int ch);
```

Funcția **islower** verifică dacă un caracter este literă mică. Returnează o valoare diferită de zero dacă parametrul este literă mică, 0 în caz contrar.

d) Funcția isupper

```
int isupper(int ch);
```

Funcția **isupper** verifică dacă un caracter este literă mare. Returnează o valoare diferită de zero dacă parametrul este literă mare, 0 în caz contrar.

e) Funcția isdigit

```
int isdigit(int ch);
```

Funcția **isdigit** verifică dacă un caracter este cifră. Returnează o valoare diferită de zero dacă parametrul este cifră, 0 în caz contrar.

f) Funcția tolower

```
int tolower(int ch);
```

Funcția tolower convertește parametrul la literă mică. Dacă parametrul este literă mare, returnează valoarea convertită, în caz contrar returnează valoarea inițială a parametrului.

g) Funcția toupper

```
int toupper(int ch);
```

Funcția toupper convertește parametrul la literă mare. Dacă parametrul este literă mică, returnează valoarea convertită, în caz contrar returnează valoarea inițială a parametrului.

III. Funcții pentru șiruri de caractere

Întrucât nu există tip de date nativ șir de caractere, limbajul nu dispune nici de operatori pentru prelucrarea șirurilor. Există însă o serie de funcții care asistă programatorul în lucrul cu șirurile de caractere. Aceste funcții necesită includerea fișierului header **cstring**.

1. Lungimea unui șir de caractere

Pentru a determina lungimea unui șir de caractere se folosește funcția **strlen(char *sir)**. Aceasta funcție returnează lungimea șirului *sir*, adică numărul de caractere din șirul al cărui prim caracter se află la adresa memorată în *sir*. Caracterul nul nu se numără.

Exemplu:

```
char sir[21];

unsigned n;

cout<<"Introduceri un sir:";
cin>>sir;//ex sir="atestat"

n=strlen(s);//n=6

cout<<"Lungimea șirului este:"<<n;
```

2. Copierea conținutului unui șir de caractere

Pentru a copia conținutul unei variabile sau constante de tip șir de caractere într-o altă variabilă de tip șir de caractere trebuie utilizată funcția **strcpy(char *dest, char *src)**.

Funcția copiază caracterele din șirul aflat la adresa *src*, inclusiv caracterul nul, în șirul al cărui prim element se află la adresa din *dest*.

Exemplu:

```
char sir1[21],sir2[21];

cout<<"Introduceri un sir:";
cin>>sir1;

strcpy(sir1,sir2); //nu este permisă atribuirea a=b

cout<<"Ati tastat:"<<b;
```

Este sarcina programatorului să se asigure că destinația are suficient spațiu alocat pentru a memora toate caracterele din variabila sursă (inclusiv terminatorul de șir).

Funcția **strncpy(char *dest, char *src, int count)**, copiază cel mult *count* caractere din șirul aflat la adresa *src*, în șirul al cărui prim element se află la adresa din *dest*. În șirul *dest* nu se va plasa caracterul nul după cele *count* caractere copiate și returnează adresa *dest*.

3. Concatenarea a două șiruri de caractere

Pentru concatenarea a două șiruri de caractere se folosește funcția **strcat(char *dest, char *src)**. Funcția adaugă (concatenează) caracterele din șirul aflat la adresa *src*, inclusiv caracterul nul, la șirul al cărui prim element se află la adresa din *dest* și returnează adresa *dest*.

Exemplu:

```
char sir1[21],sir2[21],s;

cout<<"Introduceri primul sir:";
cin>>sir1;  //ex sir1="atestat"

cout<<"Introduceri al doilea sir:";
cin>>sir2;  //ex sir2=" informatica"

strcpy(s,sir1);

strcat(s, sir2);

cout<<s;  //s="atestat informatica"
```

4. Compararea alfabetică a două șiruri de caractere

Pentru compararea a două șiruri de caractere nu se pot aplica operatorii relaționali între cele două șiruri (<.>,<=,>=,==,!=), întrucât aceștia au ca efect compararea adreselor în memorie ale celor două șiruri. De aceea trebuie utilizată funcția **strcmp(char *sir1, char *sir2)**. Funcția compară lexicografic cele două șiruri de caractere:

- dacă șirul *sir1* este lexicografi mai mic decât *sir2* funcția va returna o valoare negativă
- dacă șirul *sir1* este lexicografi mai mare decât *sir*2 funcția va returna o valoare pozitivă
- dacă cele două șiruri sunt identice funcția va returna valoarea 0

Exemplu:

```
char sir1[15],sir2[15];

cout<<"Introduceți primul șir:";cin>>sir1;

cout<<"Introduceți al doilea șir:";cin>>s2;

int x=strcmp(sir1, sir2);

if(x==0) cout<<sir1<<"="<<sir2;

else if(x<0) cout<<sir1<<"<"<<sir2;

else cout<<sir2<<">"<<sir2;
```

5. Căutarea unui caracter într-un șir de caractere

Pentru căutarea unui caracter într-un șir se folosește funcția **strchr(char *sir, char ch)**. Funcția caută caracterul *ch* în șirul al cărui prim caracter se află în memorie la adresa din *str*. Căutarea se face de la stânga la dreapta iar funcția întoarce adresa subșirului care începe cu prima apariție a caracterului *ch*.

Dacă nu este găsit caracterul, funcția returnează 0. Diferența dintre adresa șirului inițial și cea a subșirului returnat reprezintă chiar poziția caracterului căutat în șirul dat.

Exemplu:

```
char sir[15],ch,d;

cout<<"Introduceți șirul:";cin>>sir;

cout<<"Introduceți un caracter:";cin>>ch;

if(strchr(sir,ch)!=NULL)
     cout<<"Caracterul"<<ch<<" apare in sir";
else
     cout<"Caracterul<<ch<<" NU apare in sir";
```

Funcția **strrchr** are același rol cu strchr, cu deosebirea că returnează adresa ultimei apariții a caracterului (căutarea se face de la dreapta spre stânga, r=right).

6. Căutarea unui subșir într-un șir

Pentru căutarea unui subșir într-un șir se utilizează funcția **strstr(char *sir, char *subsir)**. Funcția returnează un pointer la prima apariție a conținutului variabilei subsir în conținutul variabilei sir sau valoarea NULL dacă subșirul nu apare în șirul dat.

Exemplu:

```
char sir[15],subsir[15];

char *p;

cout<<"Introduceți șirul:";
cin>>sir;

cout<<"Introduceți subșirul:";
cin>>subsir;

p=strstr(sir,subsir);

if(p!=NULL) cout<<sir<<" conține "<<subsir;

else cout<<sir<<" nu conține "<<subsir;
```

7. Separarea unui șir de caractere și obținerea cuvintelor

Funcția **strtok(*char sir, const char *sep)** extrage dintr-un șir de caractere câte un subșir (cuvânt) delimitat de caractere din șirul *sep*. Funcția se apelează în două moduri:

- primul apel are ca parametri șirul din care se face extragerea și șirul separatorilor
- la următoarele apeluri primul parametru este NULL.

Rezultatul funcției strtok este adresa de început a subșirului curent extras, sau NULL dacă nu se mai poate extrage niciun subșir din șirul dat.

Șirul din care se face extragerea se modifică în urma apelurilor. Dacă este nevoie de el mai târziu trebuie să îi facem o copie.

Exemplu:

```
char s[15], sep[]=" .,";

char *x;

cout<<"Introduceți șirul:";
cin>>s;  //ex s="Ana, are.. mere,si,.,pere"
```

```
x=strtok(s, sep);

while(x!=NULL)
{
     cout<x<<" ";
     x=strtok(NULL, sep);
}
//in urma structurii repetitive se va afisa "Ana are mere
si pere"
```

8. Transformarea caracterelor și convertirea unui șir de caractere

C++ oferă un set de funcții pentru conversii între diverse tipuri de date și șiruri de caractere.

a) Funcția strlwr

```
char strlwr(char *sir);
```

Funcția **strlwr** are rolul de a converti toate literele mari din șir în litere mici. Restul caracterelor rămân neschimbate.

b) Funcția strupr

```
char strupr(char *sir);
```

Funcția **strlwr** are rolul de a converti toate literele mici din șir în litere mari. Restul caracterelor rămân neschimbate.

c) Funcția strset

```
char *strset(char *sir, char ch);
```

Funcția **strset** înlocuiește toate caracterele din șirul *sir* cu caracterul *ch*. Returnează un pointer la *sir*.

d) Funcția strnset

```
char *strnset(char *sir, char ch, unsigned n);
```

Funcția **strnset** înlocuiește primele *n* caractere din șirul *sir* cu caracterul *ch*. Returnează un pointer la *sir*.

e) Funcția atof

```
float *atof(char *sir);
```

Funcția **atof** convertește un șir către tipul float. Dacă această conversie eșuează (se întâlnește un caracter nenumeric), valoarea returnată este 0. Această funcție (ca și cele similare) necesită includerea librăriei **stdlib.h**.

f) Funcția atoi

```
int *atoi(char *sir);
```

Funcția **atoi** convertește un șir către tipul int. Dacă această conversie eșuează (se întâlnește un caracter nenumeric), valoarea returnată este 0.

g) Funcția atol

```
long atol(char *sir);
```

Funcția **atol** convertește un șir către tipul long. Dacă această conversie eșuează (se întâlnește un caracter nenumeric), valoarea returnată este 0.

h) Funcția itoa

```
char itoa(int val, char *sir, int baza);
```

Funcția **itoa** convertește o valoare de tip int în șir, care este memorat în variabila *sir*. Variabila *baza* reține baza de numerație către care să se facă conversia. În cazul bazei 10, șirul reține și eventualul semn -.

i) Funcția ltoa

```
char ltoa(long val, char *sir, int baza);
```

Funcția **ltoa** convertește o valoare de tip long în șir, care este memorat în variabila *sir*.

j) Funcția ultoa

```
char ultoa(unsigned long val, char *sir, int baza);
```

Funcția **ltoa** convertește o valoare tip unsigned long în șir, care este memorat în variabila *sir*.

Aplicația

Aplicația realizată de mine constă într-un program în C++ cu probleme cu șiruri de caractere care au fost date la simulările și la examenele de bacalaureat în anii 2020-2024.

Programul generează un meniu principal folosind instrucțiunea *switch* iar utilizatorul poate selecta un anumit an. Pentru anul selectat se generează un nou meniu care afișează problemele disponibile. Atunci când o problemă este selectată, se afișează cerința, un exemplu iar utilizatorul poate introduce date ce vor fi prelucrate conform algoritmului necesar rezolvării problemei respective. După ce se afișează rezultatul, consola este eliberată și se revine la meniu.

Pentru a organiza codul programului am decis să creez o librărie proprie numită "problemebac.h" care conține funcții cu algoritmii necesari rezolvării problemelor.

main.cpp

```
//
//  main.cpp
//  Atestat informatica - Siruri de caractere
//
//  Realizat de Fernando-Emanuel Donea
//

#include <iostream>
#include "problemebac.h"
using namespace std;
int main()
{
   int opt;
    do
    {
        meniu_principal();
        cin>>opt;
        switch (opt) {
            case 1:{
                probleme2024();
                }break;
            case 2:{
                probleme2023();
                }break;
            case 3:{
                probleme2022();
                }break;
            case 4:{
                probleme2021();
                }break;
            case 5:{
                probleme2020();
            }break;

        }
    }while(opt!=6);
    return 0;
}
```

problemebac.h

```
//
//  problemebac.h
//  Atestat informatica - Siruri de caractere
//
//  Realizat de Fernando-Emanuel Donea
//

#ifndef PROBLEMEBAC_H_INCLUDED
#define PROBLEMEBAC_H_INCLUDED


void stergere();
void meniu_principal();


void probleme2024();
void menu2024();
void sim_martie_2024();
void model_sub_2023();


void menu2023();
void sim_martie_2023();
void probleme2023();


void menu2022();
void bac_august_2022();
void sim_martie_2022();
void probleme2022();


void menu2021();
void bac_iulie_2021();
void probleme2021();


void menu2020();
void bac_iulie_2020();
void bac_august2020();
void probleme2020();



#endif // PROBLEMEBAC_H_INCLUDED
```

problemebac.cpp

```
//
//  problemebac.cpp
//  Atestat informatica - Siruri de caractere
//
//  Realizat de Fernando-Emanuel Donea
//

#include "problemebac.h"
#include <iostream>
#include <cstring>
#include <windows.h>
using namespace std;

void stergere()//functie pentru eliberarea continutului consolei
{
    cin.get(); system("CLS");
}

void meniu_principal()
{
    system("CLS");
    cout<<"Probleme cu siruri de caractere date la examenul de
bacalaureat"<<endl<<endl;
    cout<<"1. Bac 2024"<<endl;
    cout<<"2. Bac 2023"<<endl;
    cout<<"3. Bac 2022"<<endl;
    cout<<"4. Bac 2021"<<endl;
    cout<<"5. Bac 2020"<<endl;
    cout<<"6. Iesire"<<endl;
    cout<<"Optiunea dvs:";
}


void menu2024()//functie de afisare a meniului cu probleme din anul
2024
{
    stergere();
    cout<<"Probleme din anul 2024"<<endl<<endl;
    cout<<"1. Simulare - 6 martie 2024"<<endl;
    cout<<"2. Modele de subiecte - noiembrie 2023"<<endl;
    cout<<"3. Iesire"<<endl;
    cout<<"Optiunea dvs:";
}
void sim_martie_2024()
{
    stergere();
    cout<<"Un sablon este un text in care cuvintele sunt separate prin
cate un spatiu si sunt formate fie numai din litere mici si mari ale
alfabetului englez, fie numai din caractere *, in ultimul caz numindu-
se cuvinte generice. Lungimea unui cuvant este egala cu numarul de
caractere care il compun.Un computer genereaza o fraza pe baza unui
astfel de sablon, prin inlocuirea fiecarui cuvant generic cu unul
dintre cuvintele de aceeasi lungime, preluat dintr-o lista data.
Scrieti un program C/C++ care citeste de la tastatura un numar natural,
n (n,àà[1,100]), si o lista de n cuvinte, urmata de un sablon de tipul
```

```
precizat. Fiecare cuvant din lista este format din maximum 10 litere
mici si mari ale alfabetului englez si la citire este introdus singur
pe linie. sablonul contine maximum 100 de caractere. Programul obtine
in memorie si apoi afiseaza pe ecran una dintre frazele care pot fi
generate pe baza sablonului si a listei citite sau mesajul imposibil
daca nu se poate genera o astfel de fraza."<<endl<<endl;
    cout<<"Exemplu: daca n=6, iar lista de cuvinte este: rece placuta
acasa calduta innorata soare"<<endl<<"pentru sablonul 'Era o vreme
******* si ***** din belsug *****' se genereaza fraza:"<<endl<<"Era o
vreme placuta si soare din belsug soare"<<endl<<endl;
    unsigned n;
    char s[101],a[101][11];
    cout<<"n=";cin>>n;
    for(int i=1;i<=n;i++)
    {
        cin>>a[i];
    }
    cin.get();
    cout<<"Introduceti sablonul:";
    cin.getline(s,101);

    char *x,t[101],aux[101];
    unsigned long lx,ly;
    bool ok=true;
    x=strtok(s," ");
    t[0]=NULL;
    while(x!=NULL && ok==true)
    {
        strcpy(aux,x);
        if(x[0]=='*')//cuvant generic
        {
            lx=strlen(x);
            for(int i=1;i<=n;i++)
            {
                ly=strlen(a[i]);
                if(lx==ly)
                {
                    strcpy(aux,a[i]);
                }
            }
        }
        if(aux[0]=='*')ok=false;
        strcat(t,aux);
        strcat(t," ");
        x=strtok(NULL," ");
    }
    if(ok==true)
    {
        t[strlen(t)]=NULL;
        strcpy(s,t);
        cout<<t;
    }
    else cout<<"imposibil";
}
void model_sub_2024()
{
    stergere();
```

```
    cout<<"Un text are cel mult 100 de caractere, iar cuvintele sale
sunt formate numai din litere mici ale alfabetului englez, sunt
distincte si sunt separate prin cate un spatiu. Scrieti un program
C/C++ care citeste de la tastatura un numar natural n (n,àà[1,102]),
apoi un text de tipul precizat mai sus, si afiseaza pe ecran cuvinte
ale acestuia, pe doua linii separate, astfel incat prima linie sa
contina multimea cuvintelor care au mai putin de n litere, iar a doua
linie sa contina multimea cuvintelor care au mai mult de n litere.
Cuvintele de pe fiecare linie sunt afisate intr-o ordine oarecare, iar
daca una dintre cele doua multimi este vida, se afiseaza pe ecran doar
mesajul nu exista."<<endl<<endl;
    cout<<"Exemplu: pentru n=3 si textul 'era o apa rece si cu gust
bun' se poate afisa pe ecran textul:"<<endl<<"o si cu"<<endl<<"rece
gust"<<endl<<endl;
    unsigned long n,lx;
    char s[101];
    cout<<"n=";cin>>n;
    cin.get();
    cout<<"Introduceti sirul:";cin.getline(s,101);

    char *x,a[101],b[101];

    a[0]=NULL;b[0]=NULL;
    x=strtok(s," ");
    while(x!=NULL)
    {
        lx=strlen(x);
        if(lx<n)
        {
            strcat(a,x);
            strcat(a," ");
        }
        else if(lx>n)
        {
            strcat(b,x);
            strcat(b," ");
        }
        x=strtok(NULL," ");
    }
    if(strlen(a)==0 || strlen(b)==0)cout<<"nu exista";//multime vida
    else
    {
        cout<<a<<endl<<b<<endl;
    }

}
void probleme2024()//meniul cu probleme din anul 2024
{
    int opt;
    do
    {
        menu2024();
        cin>>opt;
        switch (opt) {
            case 1:{
                sim_martie_2024();
            }break;
            case 2: {
```

```
                model_sub_2024();
            }break;
        }
    }while(opt!=3);
}


void menu2023()
{
    stergere();
    cout<<"Probleme din anul 2023"<<endl<<endl;
    cout<<"1. Simulare - 29 martie 2023"<<endl;
    cout<<"2. Iesire"<<endl;
    cout<<"Optiunea dvs:";
}
void sim_martie_2023()
{
    stergere();
    cout<<"Intr-un text de cel mult 100 de caractere cuvintele sunt
separate prin cate un spatiu si sunt formate din litere mari ale
alfabetului englez, iar daca sunt scrise prescurtat sunt urmate de
caracterul . (punct). Textul reprezinta denumirea stiintifica a unei
pasari si doar cuvintele din multimea {FAMILIA, GENUL, SPECIA},
specifice sistemului de clasificare a organismelor, sunt mereu
prescurtate, prin eliminarea ultimelor lor litere."<<endl;
    cout<<"Scrieti un program C/C++ care citeste de la tastatura un
text de tipul precizat si construieste in memorie, apoi afiseaza pe
ecran denumirea stiintifica, in care pentru cuvintele specifice
sistemului de clasificare a organismelor se pastreaza doar primele trei
litere, scrise cu litere mici, si urmate de punct, ca in
exemplu."<<endl<<endl;
    cout<<"Exemplu: pentru textul 'FAMIL. PHASIANIDAE GEN. MELEAGRIS
SP. GALLOPAVO' sau pentru textul 'FAM. PHASIANIDAE G. MELEAGRIS SPECI.
GALLOPAVO' se obtine "<<endl<<"'fam. PHASIANIDAE gen. MELEAGRIS spe.
GALLOPAVO'"<<endl<<endl;
    char s[101];
    cout<<"Introduceti sirul:";cin.getline(s,101);
    unsigned long n;
    char t[101],*x;
    t[0]=NULL;
    x=strtok(s," ");
    while(x!=NULL)
    {
        n=strlen(x)-1;
        if(x[n]=='.')
        {
            if(x[0]=='F')strcat(t,"fam. ");
            else if(x[0]=='S')strcat(t,"spe. ");
            else if(x[0]=='G')strcat(t,"gen. ");
        }
        else
        {
            strcat(t,x);
            strcat(t," ");
        }
        x=strtok(NULL," ");
    }
    t[strlen(t)-1]=NULL;
```

```
    strcpy(s,t);
    cout<<s;
}
void probleme2023()
{
    int opt;
    do
    {
        menu2023();
        cin>>opt;
        switch (opt) {
            case 1:{
                sim_martie_2023();
            }break;
        }
    }while(opt!=2);
}

void menu2022()
{
    stergere();
    cout<<"Probleme din anul 2022"<<endl<<endl;
    cout<<"1. Bacalaureat sesiunea august - septembrie - 18 august
2022"<<endl;
    cout<<"2. Simulare - 30 martie 2022"<<endl;
    cout<<"3. Iesire"<<endl;
    cout<<"Optiunea dvs:";
}
void bac_august_2022()
{
    stergere();
    cout<<"Se considera o vocala oarecare a alfabetului englez, notata
cu v, si o consoana oarecare a alfabetului englez, notata cu c. Litera
v se numeste vocala prietena a lui c daca in sirul literelor
alfabetului englez, ordonat lexicografic, v il precede pe c, iar intre
v si c nu exista nicio vocala. Se considera vocale literele a, e, i, o,
u. Exemplu: e este vocala prietena pentru consoanele f, g si h, dar nu
este vocala prietena pentru consoanele d si j."<<endl;
    cout<<"Un elev vrea sa transmita unui prieten o parola, codificata.
Parola este formata dintr-un singur cuvant de cel mult 50 de caractere,
litere mici ale alfabetului englez, cel putin una fiind consoana.
Codificarea se face prin inlocuirea fiecarei consoane cu vocala sa
prietena, ca in exemplu."<<endl;
    cout<<"Scrieti un program C/C++ care citeste de la tastatura un
cuvant, reprezentand o parola de tipul precizat si determina, in
memorie, forma codificata a acesteia. Programul afiseaza pe ecran
parola codificata obtinuta."<<endl<<endl;
    cout<<"Exemplu: pentru parola 'rame' se afiseaza 'oaie', iar pentru
parola 'sport' se afiseaz√£ 'ooooo'"<<endl<<endl;

    char s[51];
    cout<<"Introduceti parola:";cin>>s;
    unsigned long n;
    n=strlen(s)-1;
    for(int i=0;i<=n;i++)
    {
        if('u'<=s[i])
            s[i]='u';
```

```
        else if('o'<=s[i])
            s[i]='o';
        else if('i'<=s[i])
            s[i]='i';
        else if('e'<=s[i])
            s[i]='e';
        else s[i]='a';
    }
    cout<<s;
    cin.get();
}
void sim_martie_2022()
{
    stergere();
    cout<<"Un text, de cel mult 250 de caractere, reprezinta o lista cu
date de identificare ale invitatilor la o petrecere; fiecare invitat
are un prenume si un nume, care apar in lista in aceasta ordine, urmate
de simbolul ; (punct si virgula), ca in exemplu. Numele si prenumele
sunt formate din cate un singur cuvant, compus din litere mari ale
alfabetului englez, si sunt separate printr-un spatiu."<<endl;
    cout<<"Scrieti un program C/C++ care citeste de la tastatura un
text de tipul precizat mai sus apoi, de pe randul urmator, un cuvant,
x, si afiseaza pe ecran, separate prin cate un spatiu, numele tuturor
invitatilor care au prenumele x, ca in exemplu, sau mesajul NU daca nu
exista astfel de invitati."<<endl<<endl;
    cout<<"Exemplu: daca lista este DAN MARIS; DANILA PREPELEAC; DAN
POPA; EDANA DAN; si cuvantul x este DAN se afiseaza pe ecran MARIS
POPA"<<endl<<endl;
    char s[251],x[251];
    cout<<"Introduceti lista invitatilor:";cin.getline(s, 251);
    cout<<"Prenumele x=";cin>>x;

    char *p,aux[251];
    bool ok=false,gasit=false;
    p=strtok(s," ");
    while(p!=NULL)
    {
        if(strcmp(p, x)==0)
        {
            ok=true;
            gasit=true;
        }
        else if(gasit==true)
        {
            strcpy(aux,p);
            aux[strlen(aux)-1]=NULL;
            cout<<aux<<" ";
            gasit=false;
        }
        p=strtok(NULL, " ");
    }
    cin.get();
}
void probleme2022()
{
    int opt;
    do
    {
```

```
        menu2022();
        cin>>opt;
        switch (opt) {
            case 1:
                bac_august_2022();
                break;
            case 2:
                sim_martie_2022();
            default:
                break;
        }
    }while(opt!=3);

}


void menu2021()
{
    stergere();
    cout<<"Probleme din anul 2021"<<endl<<endl;
    cout<<"1. Bacalaureat sesiunea iunie-iulie 2021 - 30 iunie
2021"<<endl;
    cout<<"2. Iesire"<<endl;
    cout<<"Optiunea dvs:";
}
void bac_iulie_2021()
{
    stergere();
    cout<<"Scrieti un program C/C++ care citeste de la tastatura doua
numere naturale n si k, apoi n cuvinte, separate prin Enter. Fiecare
cuvant este format din cel mult 10 caractere, numai litere mici ale
alfabetului englez, iar numerele citite sunt din intervalul
[1,20]."<<endl;
    cout<<"Programul afiseaza pe ecran, pe linii separate, primele k
cuvinte dintre cele citite pentru care ultima litera este o vocala, sau
doar mesajul nu exista daca nu exista k astfel decuvinte. Se considera
vocale literele a, e, i, o, u."<<endl<<endl;
    cout<<"Exemplu: n=5, k=2 si cuvintele 'norii cumulus pluteau pe
cer' se va afisa pe ecran pe doua linii distincte 'norii
pluteau'"<<endl<<endl;
    unsigned n,k;
    cout<<"n=";cin>>n;
    cout<<"k=";cin>>k;

    char a[21][11],x[11];
    unsigned long lx,nr=0;
    for(int i=1;i<=n;i++)
    {
        cin>>x;
        lx=strlen(x)-1;
        if(strchr("aeiou",x[lx])!=NULL)
        {
            strcpy(a[++nr],x);
        }
    }
    if(nr<k)cout<<"NU EXISTA";
    else for(int i=1;i<=k;i++)
        cout<<a[i]<<endl;
```

```
    cin.get();
}
void probleme2021()
{
    int opt;
    do
    {
        menu2021();
        cin>>opt;
        switch (opt) {
            case 1:
                bac_iulie_2021();
                break;
            default:
                break;
        }
    }while(opt!=2);
}


void menu2020()
{
    stergere();
    cout<<"Probleme din anul 2020"<<endl<<endl;
    cout<<"1. Bacalaureat sesiunea iunie - iulie 2020"<<endl;
    cout<<"2. Bacalaureat sesiunea august - septembrie 2020"<<endl;
    cout<<"3. Iesire"<<endl;
    cout<<"Optiunea dvs:";

}
void bac_iulie_2020()
{
    stergere();
    cout<<"Numim rotire spre stanga a unui cuvant format din cel putin
trei litere operatia prin care prima sa litera se muta la final, iar
toate celelalte litere se muta cu o pozitie spre stanga.";
    cout<<"Exemplu: in urma rotirii spre stanga a cuvantului ilumina se
obtine cuvantul luminai."<<endl;
    cout<<"Un text are cel mult 100 de caractere, iar cuvintele sale
sunt formate din litere mici ale alfabetului englez si sunt separate
prin cate un spatiu. Scrieti un program C/C++ care citeste de la
tastatura un text de tipul mentionat mai sus si il transforma in
memorie prin rotirea spre stanga a fiecarui cuvant al sau format din
cel putin trei litere, ca in exemplu. Programul afiseaza pe ecran
textul obtinut sau mesajul nu exista, daca in text nu exista niciun
cuvant de cel putin trei litere."<<endl<<endl;
    cout<<"Exemplu: pentru textul 'un palc mic de scolarite ilumina
sala' se afiseaza pe ecran 'un alcp icm de colarites luminai
alas'"<<endl<<endl;
    char s[101];
    cout<<"Introduceri sirul:";cin.getline(s,101);
    bool ok=false;
    char *x,t[101],aux;
    t[0]=NULL;
    x=strtok(s," ");
    while(x!=NULL)
    {
        unsigned long lx;
```

```
        lx=strlen(x);
        if(lx>=3)
        {
            ok=true;
            aux=x[0];
            for(int i=0;i<lx-1;i++)
                x[i]=x[i+1];
            x[lx-1]=aux;
        }
        strcat(t,x);
        strcat(t," ");
        x=strtok(NULL," ");
    }
    if(ok==false)cout<<"nu exista";
    else
    {
        t[strlen(t)-1]=NULL;
        strcpy(s,t);
        cout<<s;
    }
}
void bac_august2020()
{
    stergere();
    cout<<"Doua cuvinte distincte se numesc in oglinda daca fiecare
dintre ele se obtine prin citirea literelor celuilalt de la dreapta la
stanga."<<endl;
    cout<<"Exemplu: animate si etamina sunt in oglinda, iar pentru
cuvantul reper nu exista un cuvant cu care sa fie in oglinda."<<endl;
    cout<<"Se considera un text cu cel mult 100 de caractere, in care
cuvintele sunt formate din litere mici ale alfabetului englez si sunt
separate prin cate un spatiu. Scrieti un program C/C++ care citeste de
la tastatura un text de tipul mentionat mai sus si il transforma in
memorie, inlocuind fiecare cuvant cu numar impar de litere cu acel
cuvant cu care el este in oglinda, daca acesta exista, ca in exemplu.
Programul afiseaza pe ecran textul obtinut sau mesajul nu exista, daca
in text nu s-a inlocuit niciun cuvant."<<endl<<endl;
    cout<<"Exemplu: pentru textul 'era o selectie reper de desene
animate prezenta' se obtine textul 'are o selectie reper de desene
etamina prezenta'"<<endl;
    cout<<"pentru textul 'un reper pentru desene' se afiseaza pe ecran
mesajul nu exista"<<endl<<endl;
    char s[101];
    cout<<"Introduceti sirul:";cin.getline(s,101);
    bool ok=false;
    unsigned long lx;
    char *x,t[101],y[101];
    x=strtok(s," ");
    while(x!=NULL)
    {
        lx=strlen(x);
        if(lx%2==1)
        {
            strcpy(y,x);
            for(int i=0;i<lx/2;i++)
            {
                char aux=y[i];
                y[i]=y[lx-1-i];
```

```
                y[lx-1-i]=aux;
            }
            strcat(t,y);
            strcat(t," ");
            if(strcmp(x, y)!=0)ok=true;
        }
        else
        {
            strcat(t,x);
            strcat(t," ");
        }
        x=strtok(NULL," ");
    }
    t[strlen(t)-1]=NULL;
    strcpy(s,t);
    if(ok==true)cout<<s;
    else cout<<"nu exista";
}
void probleme2020()
{
    int opt;
    do
    {
        menu2020();
        cin>>opt;
        switch (opt) {
            case 1:{
                bac_iulie_2020();
            }break;
            case 2:{
                bac_august2020();
            }break;
            default:
                break;
        }

    }while(opt!=3);
}
```