# SIRURI DE CARACTERE

## 1.Implementarea sirului de caractere

Sirul de caractere este o succesiune de caractere cuprinsa intre doua ghilimele si poate sa contina orice caracter: litere mari si mici, caractere speciale(„#”,”&”, etc.) si delimitatori(virgula, punctul, etc).

**Exemplu**: â€žBorland C++â€ , â€žProgramareâ€ž, â€ L&Mâ€ , etc.

In C++, un sir de caractere poate fi definit ca un vector de caractere, in doua moduri:

*char nume_sir [dim_max];*

*char *nume_sir;*

– unde **nume_sir** este identificatorul variabilei de tip sir sau vector de caractere, iar **dim_max** ne da numarul maxim de caractere ce pot fi memorate in sirul respectiv.In primul caz se declara un vector cu elemente de tip caracter, iar in a doua varianta se declara un pointer catre tipul caracter.

**Exemplu**:

*char s[30]; // variabila s se declara ca un vector ce poate memora siruri de maxim 30 de caractere*

*char *s; // variabila s se declara ca un pointer catre tipul char*

Intr-o variabila de tipul sir de caractere putem memora un sir in doua moduri:prin atribuire sau prin citire. Numarul de caractere ale sirului astfel memorat se numeste lungimea sau dimensiunea efectiva a sirului, sau, pe scurt lungimea sirului.In urma declararii unei variabile de tipul sir de caractere, compilatorul C++ aloca memorie pentru variabila respectiva sub forma unui vector de caractere.

Caracterele sirului vor ocupa pozitii consecutive in vector, atentie, incepand cu pozitia 0.Putem accesa orice caracter al sirului, scriind numele variabilei-vector in care memoram sirul urmat de pozitia caracterului intre doua paranteze patrate. Astfel,caracterele sirului s declarat anterior vor fi *s[0], s[1], s[2], etc.*

**Exemplu:**

```
?
1    #include<iostream>
2    int main()
3    {
4        char a[8]="test";
5        char b[5]="informatica";
6        char c[15]="foarte usor !";
         c[3]='$';
```

```
7         a="klm";
8         c=a;
9     }
10
```

**Explicatii**:

O variabila de tipul sir de caractere poate fi initializata la declarare cu o constanta sir de caractere(un sir de caractere cuprins intre ghilimele). Este cazul variabilei a din codul de mai sus, care primeste ca valoare sirul â€žtestâ€• .

*char a[8]=”test”;*

In urma initializarii, caracterele sirului efectiv memorat in variabila a vor fi: a[0]=’t’,a[1]=’e’, a[2]=’s’,si a[3]=’t’.

Trebuie insa remarcat un aspect foarte important. La sfarsitul sirului, compilatoeul C++ memoreaza un caracter special: asa numitul caracter NULL avand valoarea 0 binar.

Acesta are rolul de a marca sfarsitul constantei sir de caractere. Prin urmare, o declaratie de genul char a[4]=”test”; ar fi fost eronata. De ce? Dimensiunea maxima a variabilei a, precizata la declarare trebuie sa fie cel putin 5, pentru a putea „incapea” atat cele patru caractere efective ale sirului „test” cat si caracterul NULL.Este si cazul instructiunii de declarare si initializare char b[5]=”informatica”; din exemplul dat, unde dimensiunea maxima a vectorului de caractere b este 5, iar sirul „informatica” impreuna cu caracterul de sfarsit NULL necesita 12 pozitii.In cazul in care dimensiunea maxima a variabilei este mai mare decat lungimea efectiva a sirului plus unu, pozitiile ramase libere la sfarsitul vectorului de caractere aferent variabilei vor fi complete automat cu caracterul NULL.

Instructiunea char c[15]=”foarte usor !”; este la randul ei corecta, de acelasi tip cu precedenta, aratand in plus faptul ca sirul cu care se initializeaza o variabila de tip sir de caractere poate contine orice fel de caractere, inclusiv spatii si caractere speciale. Un caracter al unui sir deja definit poate fi modificat prin atribuire. Astfel, instructiunea c[3]=’$’; este corecta: caracterul cu indicele 3(al patrulea in sir!), devine â€š$’. In urma atribuirii, sirul c va fi „foarte usor !”, in loc de cel initial. In schimb, atribuirile a=”klm”; si c=a; sunt eronate: a si c au fost declarati vectori de caractere, iar unui vector nu i se poate atribui direct o valoare decat la declarare, ca initializare.

**Exemplu:**

```
?
1   char *s="INFORMATICA", *t,*u="TEST";
2   t=s; t+=2;
3   cout<<endl<<t;
```

```
4    cout<<endl<<u[t-s];
```

**Explicatii:**

Se declara trei siruri de caractere s, t si u, toate fiind definite ca pointeri catre char; sirul s este initializat cu „INFORMATICA”, iar u cu „TEST”. In felul acesta,elementele vectorului de caractere s vor fi: s[0]=’I’, s[1]=’N’, etc.

Instructiunea t=s; este o atribuire de pointeri, in urma caruia pointerul t va adresa tot sirul s.

Instructiunea t+=2, adica t=t+2, face adunarea dintre un intreg si un pointer. Mai exact: t adreseaza sirul s, deci contine adresa caracterului s[0]; t+2 va contine adresa caracterului s[0+2], adica adresa lui s[2]; rezultatul este atribuit tot lui t. Instructiunea cout<<endl<<t; va afisa sirul adresat de catre t. Cum t s-a fixat pe caracterul s[2], rezulta ca se vor afisa toate caracterele lui s incepand cu s[2] (al treilea) pana la intalnirea caracterului NULL care marcheaza sfarsitul sirului. Astfel se va afisa „FORMATICA”.

Instructiunea cout<<endl<<u[t-s]; este corecta. Se afiseaza caracterul de pe pozitia t-s al sirului /vectorului de caractere u. Intr-adevar, t-s este un intreg,reprezentand diferenta a doi pointeri: pointerul t adreseaza sirul incepand cu s[2],pointerul s trimite catre sirul care incepe cu s[0], deci t-s este 2-0, adica 2. In consecinta, se tipareste caracterul u[2]=’S’.

# 2.Citirea si scrierea sirurilor de caractere, lungimea unui sir

Citirea de la tastatura a unei variabile de tip sir de caractere se poate face:
– cu ajutorul instructiunii cin

*cin<<variabila_sir;*

Instructiunea cin preia caractere in variabila citita pana la intalnirea primului caracter ‘spatiu’, iar functia gets opreste citirea numai la tastarea lui <ENTER<. Cu alte cuvinte, daca de exemplu vrem sa citim intr-o variabila de tip sir un text alcatuit din mai multe cuvinte (o propozitie), va trebui sa folosim functia cin.get sau cin.getline, deoarece instructiunea cin nu retine decat primul cuvant.

Totodata pentru citirea sirurilor de caractere care contin cuvinte separate prin spatii sau alte caractere albe se utilizeaza si functia in forma cu parametri:

*cin.get(nume_sir, nr, ch);*

unde:

- nume_sir reprezinta identificatorul sirului de caractere
- nr reprezinta numarul de caractere care se vor citi impreuna cu caracterul terminal NULL

- ch desemneaza caracterul la intalnirea caruia se opreste citirea sirului (valoarea lui implicita este ‘\n’), prin urmare in cazul in care nu dorim ca citirea sa se termine cu un alt caracter diferit de ‘\n’, acest parametru poate sa lipseasca.

Functia cin.get citeste caractere pana cand este indeplinita una din conditiile:fie s-au citit nr-1 caractere, fie a intalnit caracterul specificat de parametrul al treilea. Exista inca o implementare a functiei cin.get fara parametri, apelul fiind:

**cin.get();**

De exemplu, in cazul citirii a doua siruri de caractere, in buffer-ul de citire se depune dupa citirea primului sir, caracterul ‘\n’, datorita faptului ca s-a tastat <enter<. Citirea celui de-al doilea sir se opreste pentru ca se preia ‘\n’ din buffer. Apeland functia cin.get() aceasta va prelua caracterul ‘\n’, astfel eliminand acest incovenient, iar al doilea sir se va citi in mod normal.

Afisarea unei variabiale de tip sir de caractere se poate face cu instructiunea cout, functia printf sau functia puts, de forma:

*cout<<variabila_sir;*

Pentru a afla lungimea efectiva a unui sir de caractere avem la dispozitie functia predefinita strlen, careia ii dam ca parametru sirul respectiv. Astfel, valoarea returnata de catre functia **strlen(s)** reprezinta lungimea sirului memorat in variabila s.

Pentru a putea utiliza functia strlen intr-un program, trebuie sa includem biblioteca *<string.h>*

**Exemplu**:

```
char s[]=”C++”;
cout<<strlen(s); // se va afisa 3
```

**Parcurgerea pe caractere a unui si**r.

*Metoda 1*

```
int n=strlen(s);
for(int i=0;i<n;i++)
<prelucreaza =ir s[i]<;
```

*Metoda 2*

```
for(int i=0;i<strlen(n)-1;i++)
<prelucreaza sir s[i]>
```

# Prelucrarea a doua siruri de caractere

# 1. Functiile strcpy si strncpy-atribuirea intre siruri

- Daca s1 si s2 sunt doua variabile de tip sir de caractere, o atribuire de genul s2=s1 va genera eroare.
- De ce? Dupa cum stiti sirurile de caractere sunt de fapt vectori cu elemente de tip char, iar intre doi vectori nu se poate face o atribuire directa. In cazul vectorilor de numere nu avem alta solutie decat sa copiem elementele unul cate unul dinvectorul sursa in vectorul destinatie.
- Pentru siruri de caractere insa, limbajul C++ ne pune ladispozitie doua functii care simuleaza atribuirea.

***strcpy(<sir1>,<sir2>);***
***strncpy(<sir1>,<sir2>,<nr>);***

Functia **strcpy** copiaza sirul sursa <sir2> in sirul destinatie <sir1> si intoarce adresa lui <sir1>.Primul parametru <sir1> trebuie sa fie obligatoriu un identificator de variabila de tip sir de caractere,deoarece in el se va memora sirul rezultat dupa copiere. In schimb, parametrul <sir2>, adica sirul ce trebuie copiat, poate fi dat fie printr-o variabila, fie direct ca si constanta sir de caractere.Functia **strncpy** este similara, cu deosebirea ca se copiaza in <sir1> doar primele <nr> caractere din <sir2>.

**Exemplu**:

In secventa urmatoare, variabilele s, t si u sunt de tipul sir de caractere.

```
(1) strcpy(s,"EXEMPLU");
(2) strcpy(t,s);
(3) strncpy(u,s,2);
(4) cout<<s<<endl; cout<<t<<endl; cout<<u<<endl;
```

**Explicatii:**

(1) In variabila s se copiaza sirul „EXEMPLU"
(2) In variabila t se copiaza sirul „EXEMPLU"(echivalent cu atribuirea t=s care nu se poate face direct)
(3) In variabila u se vor copia primele doua caractere ale lui s, rezultand u="EX".
(4) Se vor afisa sirurile obtinute in urma folosirii celor doua functii

# 2. Functiile strcat si strncat – concatenarea a doua siruri

Concatenarea a doua siruri de caractere s si t inseamna „lipirea" celui de-al doilea sir la sfarsitul primului. Aceasta operatie se realizeaza cu ajutorul functiilor predefinite strcat si strncat.

Atentie insa, concatenarea nu este comutativa!

***strcat(<sir1>,<sir2>);***
***strncat(<sir1>,<sir2>,<nr>);***

Functia strcat concateneaza sirul <sir2> la sfarsitul sirului <sir1>. Acest sir rezultat in urma concatenarii va fi returnat de catre functie prin intermediul parametrului <sir1>. Cu alte cuvinte,compilatorul va „lipi" sirurile date prin parametri <sir1> si <sir2>, si va memora sirul obtinut in parametrul <sir1>, sir care este apoi vazut in tot programul. Primul parametru, sirul destinatie, <sir1> trebuie sa fie obligatoriu un identificator de variabila de tip sir de caractere, asa incat compilatorul sa aiba unde depozita sirul rezultat in urma concatenarii. In schimb, parametrul sirul sursa, <sir2>, adica sirul ce trebuie concatenat(adaugat), poate fi dat fie printr-o variabila, fie direct ca si constanta sir de caractere.
Functia strncat este similara, cu deosebirea ca se concateneaza la sfarsitul lui <sir1> doar primele <nr> caractere ale sirului <sir2>. Sirul rezultat in urma concatenarii, va fi returnat de catre functie prin intermediul parametrului <sir1>.

**Exemplu:**

Ce afiseaza secventa urmatoare stiind ca s-au citit de la tastatura sirurile s1="MINI" si s2="PROGRAM" ?

*(1) strcat(s1,s2); puts(s1);*
*(2) strcat(s1,"ELE NOASTRE"); puts(s1);*
*(3) strncat(s1,s2,4); puts(s1);*
*(4) strcat(s2,s1); puts(s2);*

**Explicatii:**

*(1) concateneaza sirul s2="PROGRAM" la sfarsitul sirului s1="MINI", se va afisa sirul „MINIPROGRAM"*
*(2) concateneaza sirul „ELE NOASTRE" la sfarsitul sirului s1="MINIPROGRAM", se va afisa „MINIPROGRAMELE NOASTRE"*
*(3) concateneaza primele patru caractere ale sirului „PROGRAM" la sfarsitul sirului „MINI", tiparindu-se sirul s1="MINIPROG"*
*(4) similar cu (1) in care se ilustreaza concatenarea inversa, a sirului s1 la sfarsitul sirului s2, in care evident va rezulta s2="PROGRAMMINI".*

**Observatie!**

Atunci cand se declara variabila de tip vector de caractere ce reprezinta sirul destinatie,trebuie sa aveti grija ca dimensiunea maxima a acesteia sa fie suficient de mare astfel incat variabila respectiva sa poata ingloba sirul rezultat dupa concatenare. Pentru exemplul anterior, daca dimensiunea maxima a variabilei s1 ar fi tot 10 caractere cat are s2,

acest fapt ar genera eroare. Chiar daca sirul „MINI” cu care a fost initializat s1 incape in 10 caractere, nu acelasi lucru se poate spune despre sirurile care vor fi memorate in s1 in urma celor doua concatenari, respectiv „MINIPROGRAM” si „MINIPROGRAMELE NOASTRE”.

# 3. Functiile strcmp, stricmp, strncmp, strnicmp, -compararea sirurilor

Compararea a doua siruri de caractere s si t vizeaza ordinea lor alfabetica(lexicografica) si se face caracter cu caracter(se compara intre ele caracterele aflate pe aceeasi pozitie in cele doua siruri,adica s[0] cu t[0], s[1] cu t[1], etc.). Limbajul C++ foloseste standardul ASCII, conform caruia fiecare caracter se caracterizeaza printr-un asa numit cod ASCII, un intreg cuprins intre 0 si 255. Cand se compara doua caractere, calculatorul compara de fapt codurile ASCII. Daca c1 si c2 sunt doua variabile de tipul char, atunci:
c1<c2 <-> codul caracterului c1 este mai mic decat codul caracterului c2.

***strcmp(<sir1>,<sir2>);***

Functia strcmp compara din punct de vedere lexicografic sirurile <sir1> si <sir2>.
Functia intoarce:

*– o valoare mai mica decat 0, daca sir1<sir2;*
*– 0, daca sir1=sir2;*
*– o valoare mai mare decat 0 daca sir1>sir2.*

***strncmp(<sir1>,<sir2>,<nr_caract>);***

Functia strcmp compara din punct de vedere lexicografic primele <nr_caract> ale sirurilor <sir1> si <sir2>. Valoarea returnata este similara cu cea a functiei strcmp.

**Exemplu:**

Analizati urmatoarea secventa de program in care presupunem ca s-au citit sirurile:
*s=”PROGRAM”, t=”PROGRAMARE” si x=”PROGRAME” .*

```
(1) if(strcmp(t,x)<0)
cout<<,,\n sirul t este mai mic decat sirul x”;
else
cout<<,,\n sirul t este mai mare decat sirul x”;
(2) if(strcmp(t,s)>0)
cout<<,,\n sirul t este mai mare decat sirul s”;
```

*else*

*cout<<„\n sirul t este mai mic decat sirul s”;*

Ce se va afisa pentru (3) strcmp(t,s) si (4) strncmp(s,t,7)?

**Explicatii:**

**(1)** primele sapte caractere in cele doua siruri sunt identice, rezultatul fiind dat de al optulea caracter (t[7]<x[7]), adica ‘A'<‘E’ in sens alfabetic). Prin urmare, sirul t este mai mic lexicografic decat sirul x,deci functia strcmp(t,x) returneaza o valoare mai mica decat 0, deci se va afisa mesajul: „sirul t este mai mic decat sirul x”.

**(2)** primele sapte caractere in cele doua siruri sunt identice, dar sirul t contine trei caractere in plus. In consecinta, t>s, deci apelul returneaza o valoare pozitiva afisandu-se astfel mesajul: „sirul t este mai mare decat sirul s”.

**(3)** functia va returna o valoare pozitiva.

**(4)** Functia va returna valoarea 0, deoarece primele sapte caractere ale celor doua siruri sunt identice.

**Observatie!**

Cele doua functii strcmp si strncmp fac deosebirea intre literele mari si literele mici atunci cand se compara caractere de tip litera. Limbajul C++ dispune de alte doua functii, absolut similare, stricmp si strnicmp, care nu fac deosebirea intre litere mari si mici.

***stricmp(<sir1>,<sir2>);***

***strnicmp(<sir1>,<sir2>,<nr_caract>);***

# 4. Functia strstr – cautarea unui subsir intr-un sir de caractere

Se defineste subsirul ca fiind o portiune dintr-un sir identificata prin pozitia din care incepe si prin lungime. Prin aceasta operatie se furnizeaza prima pozitie din
care incepe intr-un sir un subsir.Aceasta operatie se realizeaza cu ajutorul functiei predefinite strstr:

***strstr(<sir>,<sb>);***

unde <sir> este sirul in care se cauta, iar <sb> este subsirul care se cauta. Daca gaseste subsirul, functia furnizeaza ca rezultat un pointer catre prima aparitie a subsirului, in caz contrar furnizeaza valoarea NULL.Cu alte cuvinte aceasta functie are rolul de a verifica daca <sb> apare ca subsir in cadrul lui <sir>. In caz afirmativ returneaza adresa primei aparitii a lui <sb> in <sir>, iar in caz contrar intoarce NULL. Altfel spus, din sirul <sir> va fi retinut un subsir incepand de la pozitia primei aparitii a lui <sb> pana la sfarsitul lui <sir>.

**Exemplu:**

Fie sirurile s=”BACALAUREAT” si t=”LAU” si secventa:
u=strstr(s,t);puts(u);
**Explicatii:**
Sirul t=”LAU” apare in cadrul sirului s=”BACALAUREAT” incepand cu pozitia patru.
Sirul returnat in urma apelului u=strstr(s,t), atribuit lui u, este subsirul „LAUREAT”(incepand cu pozitia patru a sirului s).

# 5. Alte functii utile in prelucrarea sirurilor de caractere mai pot fi:

***strspn(<sir1>,<sir2>);***

– furnizeaza ca rezultat numarul de caractere consecutive din sirul <sir1>, incepand cu primul caractere, care se gasesc printre caracterele din sirul <sir2>.

***strcspn(<sir1>,<sir2>);***

– furnizeaza ca rezultat numarul de caractere consecutive din sirul <sir1>, incepand cu primul caractere, care nu se gasesc printre caracterele din sirul <sir2>.

***strpbrk(<sir1>,<sir2>);***

– furnizeaza ca rezultat un pointer catre primul caracter din sirul <sir1> care se gaseste si in sirul <sir2>. Daca nici un caracter din sirul <sir1> nu se gaseste printre caracterele sirului <sir2>, functia furnizeaza ca rezultat adresa nula.

***strtok(<sir1>,<sir2>);***

– sirul <sir2> este un sir de caractere care poate fi folosit ca separatori, iar sirul <sir1> este format din mai multe entitati separate prin unul dintre separatorii din sirul <sir2>. Functia inlocuieste separatorii prin caracterul NULL si furnizeaza ca rezultat un pointer catre primul caracter al primei entitati. Pentru a gasi urmatoarea entitate din sirul <sir1>, apelarea functiei se va face cu strtok(NULL,<sir2);.

# Aplicatii

1. Sa se afiseze lungimea unui sir de caractere citit de la tastatura.(Rezolvati problema prin cele trei metode: indici, pointeri si functie).
2. Realizati operatia de copiere a continutului unui sir sursa intr-un sir destinatie prin cele trei posibilitati, cu utilizarea de indici, pointeri si respectivi functii.Tratati si situatia (trei

variante de program) prin care copiati intr-un sir de caractere primele n caractere dintr-un al doilea sir de caractere (sirul din care se copiaza si n se citesc de la tastatura).
3. Scrieti secventele care sa ateste concatenarea a doua siruri prin folosirea celor trei forme( indici, pointeri si functii). Tratati si situatia (trei variante de program) prin care concatenati la un sir de caractere primele n caractere din cel de al doilea sir de caractere (cele doua siruri de caractere si n se citesc de la tastatura).
4. Scrieti secventele care sa arate compararea a doua siruri de caractere cu ajutorul indicilor, a pointerilor si a functiilor invatate.
5. Scrieti secventa prin care atribuiti unei variabile de tip sir de caractere o constanta de tip sir de caractere folosind functia adecvata.
6. Urmariti instructiunile urmatorului program. Ce ar trebui sa afiseze? Executati programul. Ce constatati? Explicati de ce au fost afisate aceste valori pentru rezultate.

```
1
2   #include<iostream>
3   #include<string.h>
4   using namespace std;
5   int main()
6   {
7   char sir1[4]="alfa", sir2[4];
8   cout<<sir1<<" "<<&sir1<<" "<<&sir2<<endl;
9   strcpy(sir2,"alfabet"); cout<<sir1<<" "<<sir2;
10  }
```

7. Precizati rezultatul furnizat de urmatoarele functii:
strcmp(„abc”,”abc”), strcmp(„ab”,”a”),strcmp(„ab”,”abc”), strcmp(„ab”,”Ab”), strcmp(„ab”,”A”), strncmp(„abc”,”abcd”,3),stricmp(„Abcd”,”Abcd”), stricmp(„abcd”,”Abcd”), stricmp(„ab”,”Abc”), strnicmp(„abcd”,”Abcd”,2),strnicmp(„abc”,”Abc”,4), stricmp(„a”,”A”), strcmp(„ab”,”Ab”), stricmp(„ab”,”Ab”).
8. Sa se ordoneze alfabetic o multime de n cuvinte citite de la tastatura. Refaceti apoi programul conceput folosind sortarea prin metoda bulelor.
9. Analizati si explicati secventa urmatoare, unde s,t,u sunt variabile de tip sir de caractere:
strcpy(s,”INFORMATICA”); strcpy(t,s); strncpy(u,s,2); puts(s); puts(t); puts(u);
10. Ce afiseaza programul de mai jos?

```
1   #include<iostream>
2   #include<string.h>
3   using namespace std;
4   int main()
5   {
6   char *s[10]={"10","00","10","10","01","11"};
```

```
6     char *t="10"; int i=0,j=i-1;
7     while(s[i])
8     {
9     if(!strcmp(s[i],t)) j=i;
10    i++;
11    }
12    cout<<j;
13    }
14
```

a) -1 b) 0 c) 1 d) 3 e) 4

11. Fie sirurile declarate astfel:

char sir1[]=”aranjare”, sir2[]=”aranjament”;

Ce va afisa urmatoarea secventa de program?

if(strcmp(sir1,sir2)>0)

cout<<sir1;

else

cout<<sir2;

12. Spuneti care este efectul programului de mai jos?

int main()

{ char sir1[]=”calculator”, sir2[]=”performant”;

strcat(sir1,””); strcat(sir1,sir2); cout<<sir1;

}

13. Spuneti care este efectul programului de mai jos?

int main()

{ char sir1[]=”calculator”, sir2[]=”performant”;

strcpy(sir1+3,sir2+6); cout<<sir1;

}

14. Spuneti care este efectul programului de mai jos?

int main()

{ char sir1[20]=”calculator”, sir2[20]=”performant”;

strncat(sir1,sir2,6); cout<<sir1;

}

15. Fie s un sir de caractere cu lungimea maxima 10. In urma executarii secventei urmatoare s-au

afisat caracterele: *a*b***c*d*e. Care este continutul sirului s?

gets(s); int i=0;

while(s[i]!=â€™\0â€™)

cout<<„*”<<s[i++];

16. Fie sirurile de caractere x,s si t. Care dintre instructiunile de mai jos determina interschimbarea
continuturilor sirurilor s si t?
a) x=s; s=t; t=x; b) strcpy(t,s); strcpy(s,t);
c) strcpy(t,x); strcpy(x,s); strcpy(s,t); d) strcpy(x,s); strcpy(s,t); strcpy(t,x);
17. Daca sirurile de caractere a=â€™abcdeâ€™, b=â€™abccdeâ€™, c=â€™aacdeâ€™, d=â€™abcdâ€™ sunt sortate lexicografic, care
este ordinea corecta a acestora?
a) c,a,d,b b) c,d,b,a c) d,c,b,a d) c,b,d,a
18. Care dintre instructiunile urmatoare determina stergerea ultimelor n caractere ale sirului s (se
presupune ca n este mai mic decat lungimea sirului)?
a) strcpy(s,s-n); b) strcpy(s,s-n+1); c) strcpy(s+strlen(s)-n,””); d) *(s+n)=0;
19. Care dintre instructiunile urmatoare determina stergerea primelor n caractere ale sirului s (se
presupune ca n este mai mic decat lungimea sirului)?
a) strcpy(*s,s-n); b) strcpy(s,s+n); c) strcpy(s+strlen(s)-1,””); d) *(s+n)=0;
20. Care dintre expresiile urmatoare reprezinta suma lungimilor celor doua siruri, stiind ca s si t sunt
siruri de caractere?
a) strlen(strcat(s,t)); b) strlen(s)+strlen(t);
c) strlen(strcat(t,s)); d) toate variantele de mai sus
21. Ce se va afisa in urma executarii programului?

?

```
#include<iostream.h>
#include<string.h>
int main() {
char a[3]="1", b[3]="2", c[3]="3";
strcat(a,c); strcat(c,b); strcat(b,a); cout<<b<<endl;
}
```

22. Care dintre instructiunile urmatoare determina inserarea la pozitia p, in sirul s, a sirului t?
a. char x[30]; strcpy(x,””); strncpy(x,s,p-1); strcpy(s,s+p-1); strcat(x,t); strcat(x,s); strcpy(s,x);
b. strcpy(s,s+p+strlen(t)-1); strcat(s+p,t);
c. strncat(s+p,t,strlen(t));

d. for(int i=0;i<strlen(t);i++) s[p+i-1]=t[i];

23. Care dintre instructiunile urmatoare determina stergerea tuturor caracterelor care ocupa in sirul s
pozitiile incepand de la pozitia p1, pana la pozitia p2, inclusiv?

a) strcpy(s+p1, s+p2+1); b) strcpy(s[p1], s[p2]);

c) strcpy(s+p1, s+p2); d) strcpy(&s[p1], &s[p2+1]);

24. Se citeste un text de la tastatura. Cuvintele se considera separate prin spatiu, virgula sau punct. Numarati cate cuvinte contine textul.

25. Se citeste un text de la tastatura, format dintr-o singura propozitie. Se considera ca separarea cuvintelor se face prin cel putin un spatiu. Afisati numarul de cuvinte din text si apoi fiecare cuvant pe cate un rand. Eliminati din text spatiile suplimentare si afisati textul.

26. Se citeste de la tastatura un caracter c si apoi se introduce un text, in care separarea cuvintelor se face prin cel putin un spatiu. Sa se numere cuvintele care contin caracterul c. Sa se afiseze cuvintele in care apare acest caracter.

27. Se introduce un text de la tastatura. Sa se afiseze numarul literelor distincte din text si de cate ori apar ele in text. Se va tine cont de diferenta dintre literele mari si literele mici.