

Tipuri de Date Structurate

1.1. Șir de caractere

1.1.1 Teste cu alegere multiplă și duală

1. Care dintre următoarele instrucțiuni sunt corecte sintactic? Variabilele care intervin sunt de tip *string* / *char **.

- | | |
|---|--|
| a) <code>pos('a',s):=2;</code> | a) <code>p=strchr('a',b);</code> |
| b) <code>c:=delete(s,2,2);</code> | b) <code>k = strchr(s,x)-s;</code> |
| c) <code>writeln(length('ana'));</code> | c) <code>cout<<strlen("ana");</code> |
| d) <code>insert('red',s,4);</code> | d) <code>k = strcat("ma","ma");</code> |

2. Se consideră variabila *s* de tip *string* / *char **. Care dintre următoarele secvențe afișează valoarea variabilei *s* din care lipsesc primul și ultimul caracter? Variabila *i* aparține unui tip întreg.

- | | |
|---|---|
| a) <code>delete(s,1,1);</code>
<code>delete(s,length(s),1);</code>
<code>writeln(s);</code> | a) <code>strcpy(s,s+1);</code>
<code>strcpy(s+strlen(s)-1,</code>
<code> s+strlen(s));</code>
<code>cout<<s<<endl;</code> |
| b) <code>for i:=2 to length(s)-1 do</code>
<code> write(s[i]);</code> | b) <code>for(i=1; i<strlen(s)-1; i++)</code>
<code> cout<<s[i];</code> |
| c) <code>n:=length(s);</code>
<code>delete(s,1,1);</code>
<code>delete(s,n,1);</code>
<code>writeln(s);</code> | c) <code>strcpy(s,strchr(s,s[strlen(s)-1])+1);</code>
<code>strcpy(s,s+1);</code>
<code>cout<<s;</code> |
| d) <code>write(copy(s,2,length(s)));</code> | d) <code>for(i=0; i<strlen(s)-2; i++)</code>
<code> cout<<s[i];</code> |

3. Ce se va afișa în urma executării următoarei secvențe de instrucțiuni?

<code>s:='Primavara';</code> <code>for i:=1 to 3 do</code> <code> delete(s,2,1);</code> <code>writeln(s);</code>	<code>a = "Primavara";</code> <code>for (int i=1;i<=3;i++)</code> <code> strcpy(a+1,a+2);</code> <code>cout<<a;</code>
--	---

- | | | | |
|------------|---------|---------|----------|
| a) Pavara; | b) ara; | c) rim; | d) Para. |
|------------|---------|---------|----------|

4. Ce se va afișa în urma executării următoarei secvențe de instrucțiuni?

```
x:='Mama'; y:='Macara';
if x>y then write(x)
else if x<y then write(y)
    else writeln('Incorect');
```

```
x = "Mama"; y = "Macara";
if (strcmp(x,y)>0) cout<<x;
else
if (strcmp(x,y)==0)
    cout<<"Incorect";
else cout<<y;
```

a) Macara

b) Mama

c) MamaIncorect d) Incorect

5. Care dintre următoarele expresii sunt corecte? Toate variabilele care intervin sunt de tip *string / char **.

```
a) a := b+'c';
b) a := pos(a,b);
c) a := a+b;
d) s := insert('red',s,2);
```

```
a) strcat(x,"abi");
b) x = strcmp(x,y);
c) strcat(strcpy(x,x+1),"abi");
d) x = strstr(x,y)-x;
```

6. Ce se va afișa la finalul executării următoarei secvențe de instrucțiuni? Toate variabilele care intervin sunt de tip *string / char **.

```
a := 'albacazapada';
x := upcase(a[1]) + a[1] +
    a[length(a)];
writeln(x);
```

```
x = "albacazapada";
x[0]=x[0]-32;
p=strchr(x, 'a');
cout<<x[0]<<p[0]<<x[strlen(x)-1];
```

a) aaa;

b) AaA;

c) Aaa;

d) AAA.

7. Considerând că variabila *x* este de tip *string* în varianta Pascal, ce valoare va avea variabila șir de caractere *s* după execuția următoarelor instrucțiuni? Variabila *i* aparține unui tip întreg.

```
s:='MacaraA';
x:='';
for i:=1 to 6 do
    x:=s[i]+x;
s:=x;
```

```
s = "MacaraA";
for (i=0;i<strlen(s)/2;i++){
    char x= s[i];
    s[i]= s[strlen(s)-i-1];
    s[strlen(s)-i-1] = x;}
```

a) MAAAAA;

b) Macara;

c) AAAAAA;

d) AracaM.

8. Considerând că variabila *x* este de tip *string / char **, care dintre următoarele variante verifică, în mod corect, dacă primul caracter al său este o minusculă?

```
a)
if (x[1]>='a')OR(x[1]<='z') then
    write('Corect');
b)
if (x[1]>=a)AND(x[1]<=z) then
    write('Corect');
```

```
a)
if (x[0]<'z')
    cout<<"Corect";
b)
if (x[0]<'z' || x[0]>'a')
    cout<<"Corect";
```

c)
 if upcase(x[1])=x[1] then
 write('Corect');

d)
 if (ord(x[1])>=97)AND
 (ord(x[1])<=122) then
 write('Corect');

c)
 if (!(x[0]>122&& x[0]<97))
 cout<<"Corect";

d)
 if (!(x[0]<'a')&&(x[0]<=122))
 cout<<"Corect";

9. Ce se va afișa pe ecran în urma rulării următorului program?

```
var
  a:array[1..3] of string[6];
begin
  a[1]:='vector';
  a[2]:='de';
  a[3]:='siruri';
  write(a[2], ' ');
  write(a[2][1], ' ');
  write(a[3][4], ' ');
  write(a[1][2]);
end.
```

```
#include<iostream.h>
#include<string.h>
char a[3][10];
void main() {
  strcpy(a[0], "vector");
  strcpy(a[1], "de");
  strcpy(a[2], "siruri");
  cout<<a[1]<<' ';
  cout<<a[1][0]<<' ';
  cout<<a[2][3]<<' ';
  cout<<a[0][1]; }
```

a) de dre

b) de e r d

c) e e r d

d) de d u e

10. Ce se va afișa pe ecran în urma rulării următorului program?

```
var a,b,c:string;
    i:byte;
begin
  a:='mama'; b:='cana';
  c:='casa';
  i:=pos('a',a);
  while i<>0 do begin
    delete(a,i,1);
    delete(c,i,1);
    i:=pos('a',a);
  end;
  insert(a,b,2);
  writeln(a, ' ', b, ' ', c)
end.
```

```
#include<iostream.h>
#include<string.h>
char *a,*b,*c,k[256]; int i;
void main(){
  a="mama"; b="cana"; c="casa";
  int i=strchr(a,'a')-a;
  while (i>0) {
    strcpy(a+i,a+i+1);
    strcpy(c+i,c+i+1);
    i=strchr(a,'a')-a; }
  strncpy(k,b,1);
  strcat(k,a);
  strcat(k,b+1);
  cout<<a<<' '<<k<<' '<<c<<endl;
}
```

a)

b)

c)

d)

mcanam cana casa mm cmmana cs

mm cn cs

mm cana cs

11. Considerăm următoarele declarații:

a) var a:array [1..9,1..9] of char;
 b) var a:char[9];
 c) var a:array [0..9] of word;
 d) var a:string[20];

a) char a[9][9];
 b) char a;
 c) unsigned int a[10];
 d) char* a;

Care dintre acestea reprezintă declarația corectă a unei variabile care poate avea valoarea 'dimineata' în Pascal, respectiv "dimineata" pentru C/C++?

12. Considerăm următoarele declarații:

```
var x,y,z:string[200];
```

```
char* z;  
char x[200],y[200];
```

Identificați expresiile corecte sintactic din lista următoare:

a) `x := '12' + 'zile';`
b) `z := y + x;`
c) `y := y + str(13, x);`
d) `x := dec(z);`
e) `x := copy('12Azile',4,2);`
f) `x := insert('dimineata',y,3);`

a) `z = z++;`
b) `strcat(x,y);`
c) `z = (x!=y && y!=z);`
d) `x = "12Azile";`
e) `z = strchr("exercitii",' ');`
f) `z = strcpy('A',"BBB");`

13. Considerăm următoarea secvență de program, în care *x* și *y* sunt variabile din tipul șir de caractere. Ce se va afișa în urma execuției lor ?

```
x:='dimineata';  
y:='min' + x[length(x)];  
writeln(pos(y,x));
```

```
x="dimineata"; y="min";  
strcat(y,x+strlen(x)-1);  
p=strstr(y,x);  
cout<<(p!=NULL) ? (p-y) : 0;
```

a) 0

b) 3

c) ta

d) dieta

14. Considerăm următoarele declarații:

```
var a:string[200]; i:byte;
```

```
char a[200]; unsigned int i;
```

Se știe că șirul de caractere *a* conține numai caractere distincte, excepție făcând ultimele două care sunt identice. Identificați care dintre secvențele următoare de instrucțiuni este echivalentă cu funcția *length()* / *strlen()* ?

a) `i:=1;`
`while a[i]<>a[i+1] do inc(i);`
`writeln(i+1);`
b) `i:=0;`
`while a[i]<>a[i+1] do inc(i);`
`writeln(i);`
c) `i:=1;`
`if a[i] = a[i+1] then write(i)`
d) `i:=1;`
`if a[i]<>a[i+1] then inc(i)`
`else write(i+1);`

a) `i=0;`
`while(a[i] != a[i+1]) i++;`
`cout<<i+2;`
b) `i=0;`
`while(a[i] != a[i+1]) i++;`
`cout<<i+1;`
c) `i=0;`
`if (a[i] == a[i+1]) cout<<i;`
d) `i=0;`
`if (a[i] != a[i+1]) i++ ;`
`else cout<<i ;`

15. Considerăm următoarele declarații:

```
var a,b: string[100] ;
```

```
char *a, *b;
```

Ce se va afișa în urma execuției următoarei secvențe de instrucțiuni?

```

a := 'mama'; b := 'Mamaie'
if a>b then write(a)
else
  if a=b then write('identice')
  else write(b);

```

```

a = "mama"; b="Mamaie";
if (strcmp(a,b)>0) cout<<a;
else
  if (strcmp(a,b)==0)
    cout<<"identice";
  else cout<<b;

```

a) *Mamaie*;

b) *mama*;

c) *identice*;

d) Secvența de instrucțiuni propusă nu execută nici o afișare.

16. Considerăm următoarele declarații:

```

var a:array[1..9] of string[20];
    n,i,j:byte; x:string[20];

```

```

char a[20][20]; unsigned int n,i,j;
char x[20];

```

și următoarea secvență de program:

```

readln(n);
for i:=1 to n do readln(a[i]);
for i:=1 to n-1 do
  for j:=i+1 to n do
    if a[j]<a[i] then begin
      x:=a[j]; a[j]:=a[i]; a[i]:=x;
    end;

```

```

cin>>n;
for(i=0;i<n;i++) cin>>a[i];
for (i=0;i<n-1;i++)
  for (j=i+1;j<n;j++)
    if (strcmp(a[j],a[i])<0) {
      strcpy(x,a[j]); strcpy(a[j],a[i]);
      strcpy(a[i],x); }

```

Ce prelucrare realizează această secvență asupra elementelor vectorului *a*?

a) Ordonează crescător elementele tabloului *a* după lungimea șirurilor de caractere;

b) Ordonează lexicografic crescător elementele tabloului *a*;

c) Înlocuiește elementele tabloului cu șirul de caractere maxim din punct de vedere lexicografic;

d) Înlocuiește elementele tabloului cu șirul de caractere de lungime maximă.

17. Considerăm programul următor:

```

var a,b: string;
    i,x,y:integer;
begin
  readln(a);readln(b);
  i:=1;
  x:=length(a); y:=length(b);
  while (y-x+1>=i)
    and(copy(b,i,x)<>a) do inc(i);
  if i > y-x+1 then i:=0;
  writeln(i);
end.

```

```

#include <iostream.h>
#include <string.h>
char a[256],b[256]; int i,x,y;
void main() {
  cin>>a; cin>>b;
  x=strlen(a); y=strlen(b);
  for(i=0; y-x>=i &&
    strncmp(b+i,x,a,x)!=0; i++);
  if (i>y-x) i=-1;
  cout<<i<<endl;
}

```

Identificați care dintre expresiile următoare sunt echivalente cu de programul de mai sus (la evaluarea expresiei se obține aceeași valoare cu cea afișată prin program).

- | | |
|-------------------|-------------------|
| a) concat(a,b) | a) strlen(a); |
| b) length(a + b); | b) strlen(b); |
| c) pos(a,b); | c) strstr(b,a)-b; |
| d) copy(a,b,i); | d) strcmp(a,b); |

18. Fie secvența de instrucțiuni următoare :

For i :=1 to length(a) do	for (i=0;i<strlen(a);i++)
if a[i] in ['A'..'Z'] then	if (a[i]>='A' && a[i]<='Z')
a[i] :=chr(ord(a[i])+32) ;	a[i] += 32;

Știind că *a* este un șir de caractere și *i* o variabilă de tip întreg, identificați prelucrarea realizată asupra caracterelor sale.

- a) transformarea caracterelor de tip minusculă în majuscula corespunzătoare;
- b) inserarea șirului de caractere 32 după fiecare caracter de tip majusculă.
- c) transformarea caracterelor de tip majusculă în minuscula corespunzătoare;
- d) ordonarea alfabetică a majusculilor în cadrul șirului;
- e) ștergerea caracterelor spațiu.

19. Fie următorul program:

var s:string[10];	#include<iostream.h>
i:integer; x:char;	#include<string.h>
begin	char s[10]; int i; char x;
s:='clasa';	void main()
for i:=1 to length(s)-1 do begin	{ strcpy(s,"clasa");
if (s[i]>s[i+1]) then begin	for (i=0;i<strlen(s)-1;i++){
x:=s[i];	if (s[i]>s[i+1]){
s[i]:=s[i+1];	x=s[i]; s[i]=s[i+1];
s[i+1]:=x;	s[i+1]=x;
end;	}
write(s,' ');	cout<<s<<" ";
end;	}
end.	}

Ce se va afișa pe ecran în urma execuției acestui program?

- a) clasa clasa calas calas
- b) clasa calsa calas
- c) clasa calsa calsa calas
- d) clasa aclsa aclas acals

20. Se consideră șirurile de caractere a și b . Identificați care dintre următoarele secvențe de instrucțiuni modifică valoarea șirului b prin ștergerea primei apariții a lui a în b . În situația în care a nu se regăsește în b , valoarea acestuia din urmă trebuie să rămână neschimbată. Variabila x este de tip *integer* pentru Pascal, respectiv de tip *int* pentru C/C++.

a)
delete(b, pos(a, b), length(a));

b)
x:=pos(a, b);
b:=copy(b, 1, x-1)+
copy(b, x+length(a), x)

c)
delete(a, pos(a, b), length(b));

d)
x:=pos(a, b);
if x<>0 then
b:=copy(b, 1, x-1)+
copy(b, x+length(a), x);

a)
x=strstr(b, a)-b;
if (x>=0)
strcpy(b+x, b+x+strlen(a));

b)
x=strstr(b, a)-b;
strcpy(b+x, b+x+strlen(a)-1);

c)
x=strstr(a, b)-a;
if (x>=0)
strcpy(a+x, a+x+strlen(b));

d)
x=strstr(a, b)-a;
strcpy(a+x, a+x+strlen(b));

21. Considerăm următorul program. Ce condiție trebuie îndeplinită pentru ca în urma rulării acestuia să se afișeze mesajul *Corect*?

```
var a,b:string;
begin
  readln(a);
  readln(b);
  while (a[1]=b[1]) and (a<>'' ) and
    (b<>'' ) do
    begin
      delete(a, 1, 1);
      delete(b, 1, 1);
    end;
  if (a='') and (b='') then
    write('Corect')
  else
    write('Incorect');
end.
```

```
#include<iostream.h>
#include<string.h>
char a[256],b[256];
void main()
{
  cin>>a;
  cin>>b;
  while(a[0]==b[0]&&a[0]!=0
    &&b[0]!=0) {
    strcpy(a, a+1);
    strcpy(b, b+1);
  }
  if (a[0]==0 && b[0]==0)
    cout<<"Corect";
  else cout<<"Incorect";
}
```

- a) afișează mesajul *Corect* doar dacă șirurile a și b au lungimi egale;
- b) afișează mesajul *Corect* doar dacă șirurile a și b au valori identice;
- c) afișează mesajul *Corect* doar dacă șirurile a și b au lungimi nule;
- d) afișează mesajul *Corect* doar dacă șirurile a și b au valori identice de lungime 1;

22. Se consideră următorul program:

```
var a,b:string;
    i:integer;
begin
    b:='';
    for i:=1 to 3 do begin
        readln(a);
        b:=b+copy(a,i,length(a)-i+1);
    end;
    writeln(b)
end.
```

```
#include<iostream.h>
#include<string.h>
char a[256],b[256]; int i;
void main()
{ strcpy(b,"");
  for(i=0; i<3; i++){
    cin>>a; strcat(b,a+i);
  }
  cout<<b;
}
```

Identificați ce se va afișa dacă de la tastatură se vor introduce, în ordine, șirurile de caractere: “copil”, “masina”, “bloc”.

- a) opilsinac b) cmb c) copilmasinbl d) copilasinaoc

23. Considerăm următoarul program:

```
var a,b:string;
begin
    readln(a);
    readln(b);
    while (pos(a[1],b)<>0) do
    begin
        delete(b,pos(a[1],b),1);
        delete(a,1,1);
    end;
    if (a='') and (b='') then
        write('Da')
    else
        write('nu');
end.
```

```
#include<iostream.h>
#include<string.h>
char a[256],b[256],*p; int i;
void main()
{ cin>>a; cin>>b;
  while (strchr(b,a[0])!=NULL &&
        a[0]!=0){
    p=strchr(b,a[0]);
    strcpy(p,p+1);strcpy(a,a+1);
  }
  if (a[0]==0 && b[0]==0)
    cout<< "da";
  else cout<<"nu";
}
```

La sfârșitul rulării acestuia se va afișa mesajul *Da* dacă și numai dacă:

- a) șirurile *a* și *b* au valori egale;
 b) șirurile *a* și *b* au lungimi identice;
 c) șirurile *a* și *b* sunt formate din exact aceleași caractere, eventual în altă ordine.
 d) fiecare caracter al șirului *a* apare și în șirul *b*;

24. Fie *a* un șir de caractere (*string* / *char* *) și *x* o variabilă întreagă. Care dintre următoarele instrucțiuni elimină toată secvența de caractere identice de la începutul șirului *a* (pentru a fi ștearsă, secvența va conține minimum 2 caractere):

```
a) x:=length(a);
   while ((a[1])=a[2]) and
   (length(a)>1) do delete(a,1,1);
   if length(a)<>x then
       delete(a,1,1);
```

```
a) x=strlen(a);
   while (*a==*(a+1)) {a=a++;}
   if (strlen(a)!=x)
       strcpy(a,a+1);
```


b) **while**((a[1])=a[2])**and**
 (length(a)>1) **do** delete(a,1,1);

c) **while**((a[1]) <> a[2])**and**
 (length(a)>1) **do** delete(a,1,1);

d) **while**((a[1]) <> a[2])**and**
 (length(a)>1) **do** delete(a,1,1);
 delete(a,1,1);

b) **while** (*a==*a+1) {
 a=a++;
 }
 strcpy(a,a+1);

c) **while** (*a!=*(a+1)) {a=a++;}
 strcpy(a,a+1);

d) **while** (*a==*(a+1)) {a=a++;}
 strcpy(a,a+1);

25. Fie a un șir de caractere (*string / char **) și i, j variabile întregi. Care dintre următoarele instrucțiuni permit afișarea mesajului "Da" dacă și numai dacă valoarea lui a este palindrom?

a) $i:=1$; $j:=\text{length}(a)$;
while ($a[i]<>a[j]$)**and**($i<j$) **do**
begin dec(i); inc(j); **end**;
if ($i>j$) **then** write('DA')
else write('NU');

b) $i:=1$; $j:=\text{length}(a)$;
while ($a[i]<>a[j]$)**and**($i<j$) **do**
begin inc(i); dec(j); **end**;
if ($i>j$) **then** write('DA')
else write('NU');

c) $i:=1$; $j:=\text{length}(a)$;
while ($a[i]=a[j]$)**and**($i<j$) **do**
begin inc(i); dec(j); **end**;
if ($i>j$) **then** write('DA')
else write('NU');

d)
 $i:=1$; $j:=\text{length}(a)$;
while ($a[i]=a[j]$) **and** ($i<j$) **do**
begin dec(i); inc(j); **end**;
if ($i>j$) **then** write('DA')
else write('NU');

a) $i=0$; $j=\text{strlen}(a)-1$;
while (($a[i]==a[j]$)&&($i<j$))
 { $i--$; $j++$; }
if ($i>j$) cout<<"DA";
else cout<<"NU";

b) $i=0$; $j=\text{strlen}(a)-1$;
while (($a[i]!=a[j]$)&&($i<j$))
 { $i++$; $j--$; }
if ($i<j$) cout<<"DA";
else cout<<"NU";

c) $i=0$; $j=\text{strlen}(a)-1$;
while (($a[i]==a[j]$)&&($i<j$))
 { $i++$; $j--$; }
if ($i>j$) cout<<"DA";
else cout<<"NU";

d) $i=0$; $j=\text{strlen}(a)-1$;
while (($a[i]!=a[j]$)&&($i<j$))
 { $i--$; $j++$; }
if ($i>j$) cout<<"DA";
else cout<<"NU";

26. Știind că variabila a este folosită pentru a memora, ca șir de caractere, numele unei discipline studiate în liceu (maximum 50 caractere), identificați o declarație corectă a sa:

a) **var** a=string;
 b) **var** a:string[39];
 c) **var** a:string[50];
 d) **var** a:array[1..50] of string;

a) **char** a;
 b) **char** a[39];
 c) **char** a[50];
 d) **char*** a[20];

27. Știind că variabila a este utilizată pentru a memora numele celor 7 zile ale săptămânii, cum trebuie ea declarată:

```

a) var a: string[7][7];
b) var a: array[1..7] of string;
c) var a: string[7];
d) var a: array[1..7] of char;

```

```

a) char a[7];
b) char a[7][17];
c) char a;
d) char** a[7];

```

28. Considerăm secvența de instrucțiuni următoare în care variabila *s* este un șir de caractere, *i* și *k* variabile întregi, *x* o variabilă din tipul *char* iar *ok* este o variabilă din tipul *boolean*(Pascal) - *int*(pentru C/C++):

```

i := 1; ok := true; k := 0;
while (i<=length(s))and ok do
begin
  if s[i]=x then begin
    k:=i; ok:=false;
  end;
  inc(i);
end;

```

```

i = 0;
ok = 1; k = 0;
while(i<strlen(s) && ok){
  if(s[i]==x)
  {
    k = i;
    ok = 0;
  }i++;}

```

Care dintre atribuirile următoare conduc la obținerea aceleiași valori pentru *k*, ca cea obținută în urma execuției secvenței prezentate ?

```

a) k := concat(s,x)
b) k := length(s)
c) k := pos(x,s)
d) k := pos(s,x)

```

```

a) k = strcat(s,x)
b) k = strlen(s)
c) k = strchr(s,x)-s
d) k = strchr(s,x)-x

```

1.1.2 Probleme rezolvate

1. Se consideră un text în care unicul separator este spațiul. Știind că între oricare două cuvinte pot exista mai mulți separatori, să se determine numărul de cuvinte din text.

Exemplu: Pentru textul 'Am venit repede.' se va afișa 3.

Soluție

Algoritmul presupune parcurgerea caracter cu caracter a textului și identificarea numărului de perechi de caractere alăturate care pot reprezintă finalul unui cuvânt (caracter diferit de spațiu urmat de un separator).

```

1  var s:string;
2      i,nr:integer;
3  begin
4      readln(s);
5      s:=s+' '; nr:=0;
6      for i:=1 to length(s)-1 do
7          if (s[i]<>' ')and
8              (s[i+1]=' ')then inc(nr);
9      writeln(nr);
10 end.
11

```

```

#include <stdio.h>
#include <string.h>
char s[256];int nr,i;
void main() {
  gets(s);
  strcat(s," "); nr=0;
  for (i=0;i+1<strlen(s);i++)
    if (s[i]!=' ' && s[i+1]==' ')
      nr++;
  printf("%d\n",nr);
}

```

2. Se citește de la tastatură un vers al unei poezii și o silabă. Să se realizeze un program care determină numărul de apariții al silabei citite în textul respectiv.

Exemplu : Pentru versul 'Un curcubeu multicolor' și silaba 'cu' se va afișa 2.

Soluție

Atât versul citit cât și silaba vor fi reținute în variabile de tip șir de caractere *vers* și *s*. Algoritmul propus se bazează pe căutarea repetată a subșirului *s*, folosindu-ne de funcția predefinită *pos()* / *strstr()*.

<pre> 1 var vers,s:string; 2 nr,p,l:integer; 3 begin 4 readln(vers); 5 readln(s); 6 nr:=0; 7 while pos(s,vers)<>0 do begin 8 p:=pos(s,vers); 9 l:=length(s); 10 delete(vers,p,l); 11 inc(nr); {nr:=nr+1} 12 end; 13 writeln(nr); 14 end.</pre>	<pre> #include <stdio.h> #include <string.h> char vers[256],s[256]; int nr,p,l; void main() { gets(vers); gets(s); nr=0; while (strstr(vers,s)!=NULL){ p=strstr(vers,s)-vers; l=strlen(s); strcpy(vers+p,vers+p+l); nr++; } printf("%d\n",nr); }</pre>
---	--

3. O propoziție se consideră fiind palindrom dacă ignorând diferențele dintre minuscule și majuscule și ignorând separatorii, va fi identică cu propoziția obținută prin citirea literelor de la dreapta spre stânga. De exemplu, propoziția 'Ele fac cafele' este palindrom. Să se realizeze un program care permite citirea propoziției de la tastatură și verifică dacă ea poate fi considerată palindrom. Cuvintele vor fi separate în cadrul propoziției prin spații (singurul separator prezent).

Soluție:

Algoritmul propus conține trei etape:

- ștergerea spațiilor dintre cuvinte;
- transformarea în majuscule a fiecărei litere;
- cuvântul astfel obținut se verifică dacă este palindrom.

<pre> 1 var s:string; ok:boolean; 2 i,p:integer; 3 begin 4 readln(s); 5 while pos(' ',s)<>0 do begin 6 p:=pos(' ',s); 7 delete(s,p,1); 8 end; 9 for i:=1 to length(s) do 10 s[i]:=upcase(s[i]); 11 ok:=true;</pre>	<pre> #include <string.h> #include <stdio.h> char s[256]; int i,p,ok; void main() { gets(s); while (strstr(s," ")!=NULL) { p=strstr(s," ")-s; strcpy(s+p,s+p+1);} for (i=0;i<strlen(s);i++) if (s[i]>='a'&&s[i]<='z') s[i]='A'-'a';</pre>
---	--

```

12  for i:=1 to length(s) div 2 do
13      if s[i]<>s[length(s)-i+1]
14      then
15          ok:=false;
15  writeln(ok);
16  end.
17

```

```

ok=1;
for(i=0;i<strlen(s)/2;i++)
    if (s[i]!=s[strlen(s)-i-1])
        ok=0;
printf("%d\n",ok);
}

```

4. Se citește de la tastatură un șir de caractere care reprezintă un număr în baza 16. Să se realizeze un program care permite conversia în baza 10 a numărului citit. În situația în care șirul conține caractere nepermise se va afișa mesajul "Imposibil".

Exemplu: Pentru numărul hexazecimal A1B se va afișa 2587.

Soluție

Știm că singurele cifre permise la scrierea unui număr într-o bază b ($2 \leq b \leq 10$) sunt $0..b-1$, corespunzătoare resturilor care se pot obține la împărțirea cu b . Dacă baza este mai mare ca 10, atunci fiecare din resturile obținute vor fi codificate în ordine cu literele A,B,C, ș.a.m.d. În baza 16 resturile mai mari ca 9 sunt codificate astfel: 'A'=10; 'B'=11; 'C'=12; 'D'=13; 'E'=14; 'F'=15.

De exemplu, numărul $2AC_{(16)}$ reprezintă numărul $2*16^2 + 10*16^1 + 12*16^0 = 684_{(10)}$; Pentru a evita calcularea puterii lui 16 din cadrul fiecărui termen putem rescrie expresia de mai sus și sub forma: $2*16^2 + 10*16^1 + 12*16^0 = ((0*16 + 2)*16 + 10)*16 + 12$.

În cazul în care cifrele sunt exprimate prin litere se va realiza o corespondență între codul ASCII al său și numărul pe care îl reprezintă în baza 16:

Caracter	Codul ASCII	Restul pe care îl indică în baza 16
'A'	65	10 = 65-55
'B'	66	11 = 66-55
'C'	67	12 = 67-55
'D'	68	13 = 68-55
'E'	69	14 = 69-55
'F'	70	15 = 70-55
		ord(Litera)-55

```

1  var s:string;
2      i,nr,c,er:integer;
3  begin
4      readln(s); nr:=0;
5  for i:=1 to length(s) do
6      begin
7          if s[i]<='9' then
8              val(s[i],c,er)
9          else
10             c:=ord(upcase(s[i]))-55;
11             nr:=nr*16 + c;
12         end;
13     writeln(nr);
14 end.

```

```

#include <string.h>
#include <stdio.h>
char s[256];
int i,nr,c;
void main() {
    gets(s); nr=0;
    for (i=0;i<strlen(s);i++) {
        if (s[i]<='9') c=s[i]-'0';
        else c=s[i]-55;
        nr=nr*16+c;
    }
    printf("%d\n",nr);
}

```

5. Se citește de la tastatură un vers al unei poezii. Să se realizeze un program care determină numărul de cuvinte din text. Cuvintele sunt separate între ele prin caracterele: spațiu (' '), virgula (,), punctul (.) sau punct și virgula (;).

Exemplu: Pentru propoziția 'Vremea trece...vremea vine,' se va afișa valoarea 4.

Soluție

În cadrul algoritmul pe care îl prezentăm vom număra câte cuvinte sunt în vers folosindu-ne de următoarea regulă:

O poziție în cadrul versului reprezintă începutul unui nou cuvânt dacă tipul caracterului de pe poziția respectivă este literă iar cel precedent este un separator. Pentru a identifica mai ușor tipul unui caracter vom folosi o variabilă șir de caractere a cărei valoare o va reprezenta șirul de separatori. ' , . ; ' Dacă un caracter al versului nu se regăsește printre caracterele acestuia atunci el este de tip literă. În C++ se poate folosi funcția *strtok()*.

<pre> 1 var s,sep:string; 2 x,y,nr,i:integer; 3 begin 4 readln(s); 5 nr:=0; 6 sep:=' ,. ;'; 7 s:='.' + s + '.'; 8 for i:=2 to length(s) do 9 begin 10 x:=pos(s[i],sep); 11 y:=pos(s[i-1],sep); 12 if (x=0)and(y<>0) then 13 nr:=nr+1; 14 end; 15 writeln(nr); 16 end.</pre>	<pre> #include <string.h> #include <stdio.h> char s[256],*p; int nr; void main() { gets(s); nr=0; p=strtok(s," ,. ;"); while (p!=NULL) { nr++; p=strtok(NULL," ,. ;"); } printf("%d\n",nr); }</pre>
--	---

6. Se consideră două cuvinte formate din literele mari și mici ale alfabetului englez. Verificați dacă ele sunt *anagrame*.

Două șiruri de caractere sunt anagrame, dacă unul dintre ele este format din caracterele celuilalt, eventual într-o altă ordine. *Exemplu:* 'are', 'era'.

Soluție:

Algoritmul propus caută succesiv prima literă a primului cuvânt citit (x), în cel de al doilea (y). În cazul în care este găsită ea va fi ștearsă din ambele cuvinte. Procedeu continuă până când fie litera nu este regăsită, fie când lungimea lui x este 0, caz în care cuvintele sunt anagrame.

O altă metodă ar consta în ordonarea caracterelor ambelor cuvinte și compararea acestora la final.

```

1 var x,y:string;
2 begin
3   readln(x);
4   readln(y);
5   if length(x)=length(y) then
6     begin
7       while pos(x[1],y)<>0 do
8         begin
9           delete(y,pos(x[1],y),1);
10          delete(x,1,1);
11        end;
12      if (x='') and (y='') then
13        write('Da')
14      else
15        write('Nu')
16      end
17    else write('Nu')
18  end.

```

```

#include <iostream.h>
#include <string.h>
char x[21],y[21],*p;
void main() {
  cin>>x>>y;
  if (strlen(x)==strlen(y)) {
    while (strchr(y,x[0])!=NULL
      && x[0]!=0){
      p=strchr(y,x[0]);
      strcpy(p,p+1);
      strcpy(x,x+1);
    }
    if (x[0]==0 && y[0]==0)
      cout<<"Da";
    else cout<<"Nu";
  }
  else cout<<"Nu";
}

```

7. De la tastatură se citește un text codificat după regula următoare: în fața fiecărui caracter este scris numărul de apariții consecutive ale acestuia. Realizați un program care decodifică textul. Numărul de apariții consecutive ale unui caracter este strict mai mic decât 10.

Exemplu: Pentru codificarea '1c1o1p3i' se va afișa textul 'copiii'

Soluție

În funcție de tipul fiecărui caracter (literă/cifră) se decide dacă trebuie realizată afișarea caracterului curent sau actualizarea cifrei care va reprezenta numărul de scrieri ale caracterului următor.

```

1 var s:string;
2   i,j,er,cifra:integer;
3 begin
4   readln(s);
5   for i:=1 to length(s) do
6     begin
7       if s[i] in ['0'..'9'] then
8         val(s[i],cifra,er)
9       else
10        for j:=1 to cifra do
11          write(s[i]);
12        end;
13      end.

```

```

#include <stdio.h>
#include <string.h>
char s[256];
int i,j,cifra;
void main() {
  gets(s);
  for (i=0;i<strlen(s);i++)
    if (s[i]>='0'&&s[i]<='9')
      cifra=s[i]-'0';
    else
      for (j=0;j<cifra;j++)
        printf("%c",s[i]);
}

```

8. Se citește de la tastatură un șir de caractere. Identificați în cadrul acestuia o secvență de lungime maximă care poate fi convertită către o variabilă de tip întreg.

Exemplu: Pentru șirul „25AB32042Xs23” se va afișa 32042.

Soluție

Algoritmul propus determină, după o singură parcurgere a caracterelor din șir, care este secvența de lungime maximă ce poate fi convertită către o dată de tip numeric. Se va reține în final, poziția de început (*poz*) a secvenței în cadrul șirului și lungimea acesteia (*max*).

<pre>1 var s,c:string; 2 i,lung,max,poz,er,x:integer; 3 begin 4 readln(s); lung:=0; 5 s:=s+' '; 6 for i:=1 to length(s) do 7 begin 8 val(s[i],x,er); 9 if (er=0) then inc(lung) 10 else begin 11 if max<lung then begin 12 max:=lung; 13 poz:=i-lung; 14 end; 15 lung:=0; 16 end; 17 end; 18 for i:=poz to poz+max-1 do 19 write(s[i]); 20 end.</pre>	<pre>#include <stdio.h> #include <string.h> char s[256]; int i,lung,max,poz; void main() { gets(s); strcat(s," "); lung=0; for (i=0;i<strlen(s);i++) if (s[i]>='0'&&s[i]<='9') lung++; else { if (max<lung) { max=lung; poz=i-lung; } } lung=0; } for (i=poz;i<poz+max;i++) printf("%c",s[i]); }</pre>
---	---

9. Se dorește ca operația *Find-Replace* să fie executată asupra unui text care nu conține mai mult de 250 de caractere. Această operație constă în înlocuirea tuturor aparițiilor unui subșir *s1* cu un alt subșir *s2*. În cazul de față cele două subșiruri se consideră a fi diferite și de lungimi egale. Creați un program care simulează această operație.

Exemplu: Considerând textul „care caracatita”, dacă subșirul „ca” se va înlocui cu „ta” atunci textul afișat va fi „tare taratatita”.

Soluție

Programul sursă realizat pentru varianta Pascal folosește apeluri la subprogramele predefinite pentru tipul șir de caractere. Sursa C/C++ construiește un nou șir de caractere *S* în care subșirul *s1* a fost înlocuit cu *s2*.

<pre>1 var 2 s,s1,s2:string; 3 c:integer; 4 5 begin 6 readln(s); 7 readln(s1); 8 readln(s2); 9 c:=pos(s1,s);</pre>	<pre>#include <stdio.h> #include <string.h> char S[256],s2[256]; int u,i,p; char s[56],s1[56]; void main() { gets(s); gets(s1); gets(s2);</pre>
--	---

```

10 while c<>0 do
11 begin
12     delete(s,c,length(s1));
13     insert(s2,s,c);
14     c:=pos(s1,s);
15 end;
16 writeln(s);
17 end.
18
19
20
21
22

```

```

do { char* pt=strstr(s,s1);
    if (!pt){
        for(i=u;i<strlen(s);i++)
            strncat(S,s+i,1); break;
    } p=pt-s;
    for(i=u;i<p;i++) S[i]=s[i];
    for(i=p;i<p+strlen(s1);i++)
        s[i]='!';
    for(i=p;i<p+strlen(s2);i++)
        strncat(S,s2+i-p,1);
    u=p+strlen(s1);} while (1);
puts(S);
}

```

10. Se consideră un șir de n cuvinte. Identificați mulțimea cu număr maxim de cuvinte care sunt anagrame între ele două câte două. Se va afișa cardinalul mulțimii și un element al acesteia, considerat reprezentantul ei.

Exemplu: pentru $n=6$ și cuvintele 'arc', 'rac', 'voi', 'car', 'armata', 'tamara' se va afișa mulțimea: arc 3.

Soluție

Algoritmul verifică pentru oricare două cuvinte dacă sunt anagrame. Se memorează elementul pentru care s-a determinat numărul maxim de anagrame.

```

1 var a:array[1..100]of
2 string[20];
3     n,i,nr,j,max:integer;
4     cuv,x,y:string;
5 begin
6     readln(n); max:=0;
7     for i:=1 to n do
8         readln(a[i]);
9     for i:=1 to n-1 do begin
10         nr:=1;
11         for j:=i+1 to n do begin
12             y:=a[j]; x:=a[i];
13             if length(x)=length(y) then
14                 begin
15                     while pos(x[1],y)<>0 do
16                         begin
17                             delete(y,pos(x[1],y),1);
18                             delete(x,1,1);
19                         end;
20                     if (x='')and(y='')then
21                         inc(nr)
22                     end;
23                 if max<nr then begin
24                     max:=nr; cuv:=a[i];
25                 end;
26             end;
27         end;
28     writeln(cuv,' ',max);
29 end.

```

```

#include <iostream.h>
#include <string.h>
char a[101][21],x[21],y[21];
char *p ,cuv[21];
int nr,n,i,j,k,max,poz;
void main() {
    cin>>n; max=0;
    for (i=0;i<n;i++) cin>>a[i];
    for (i=0;i<n-1;i++){
        for (nr=1, j=i+1;j<n;j++){
            strcpy(x,a[i]);
            strcpy(y,a[j]);
            if (strlen(x)==strlen(y)) {
                while(strchr(y,x[0])!=NULL
                    && x[0]!=0){
                    p=strchr(y,x[0]);
                    strcpy(p,p+1);
                    strcpy(x,x+1);
                }
                if (x[0]==0 && y[0]==0)
                    nr++;
            }
            if (max<nr){
                max=nr;strcpy(cuv,a[i]); }
        }
    }
    cout<<cuv<<" "<<max;
}

```


11. Se consideră un fișier text *in.txt* ce conține numere întregi dispuse pe mai multe linii. Orice caracter ce nu reprezintă un caracter numeric este considerat separator. Scrieți un program care creează un fișier *out.txt* ce conține pe fiecare linie media aritmetică a numerelor situate pe aceeași linie în fișierul *in.txt*. Media aritmetică va fi scrisă cu două zecimale. Pe fiecare linie a fișierului de intrare se află maximum 200 de caractere.

Exemplu: in.txt

2..3a 403bx
2..2 A,..5
1.92

out.txt

136.00
3.00
46.50

Soluție:

În problema de față, separatorii nu sunt reprezentați prin *spații albe*. Această situație impune folosirea la citire a unei variabile de tip șir de caractere.

Pentru determinarea mediei se va parcurge șirul citit caracter cu caracter, relizându-se conversia secvențelor de caractere numerice către date de tip numeric(întreg).

<pre> 1 var f,g:text; x,c:string; 2 i,v,s,n,er:integer; 3 begin 4 assign(f,'in.txt'); reset(f); 5 assign(g,'out.txt');rewrite(g); 6 while not eof(f) do begin 7 s:=0; n:=0; c:=''; 8 readln(f,x); 9 x:=x+'.'; 10 for i:=1 to length(x) do 11 if x[i] in ['0'..'9'] then 12 c:=c+x[i] 13 else 14 if c<>'' then begin 15 val(c,v,er); c:=''; 16 s:=s+v; 17 inc(n); 18 end; 19 writeln(g,s/n:0:2); 20 end; 21 close(f); close(g); 22 end.</pre>	<pre> #include <stdio.h> #include <string.h> FILE *f,*g; char x[256]; int i,n; float c,s; void main() { f=fopen("in.txt","r"); g=fopen("out.txt","w"); while (!feof(f)) { s=0; n=0; c=0; if (!fgets(x,256,f)) break; strcat(x,"."); for (i=0;i<strlen(x);i++) if (x[i]>='0'&&x[i]<='9') c=c*10+x[i]-'0'; else if (c) { s+=c; n++; c=0; } fprintf(g,"%f\n",s/n); } fclose(f); fclose(g); }</pre>
---	--

12. Se consideră un cuvânt din maximum 50 de caractere format numai din literele alfabetului englezesc. Se cere inserarea minusculei minime din punct de vedere lexicografic (aparținând cuvântului), între oricare două litere identice aflate pe poziții alăturate. La inserare nu se va face distincție între majuscule și minuscule. Dacă nu există minuscule în cuvânt atunci acesta nu va suferi nici o modificare.

Exemplu:

Pentru cuvântul "inNorat" minusculea minimă este 'a' și se va afișa „inaNorat”. Pentru șirul de caractere „boQlutton” minusculea minimă este 'b' și se va afișa „bobOlutbton”.

Soluție

Operația de inserare se efectuează simultan cu operația de parcurgere a șirului de caractere. La inserarea unui nou caracter între două litere identice, indicele curent se incrementează cu două poziții.

<pre>1 var 2 s:string; m:char; i:byte; 3 begin 4 readln(s); 5 m:=chr(127); 6 for i:=1 to length(s) do 7 if (s[i]<m) 8 and(s[i]<>upcase(s[i])) 9 then m:=s[i]; 10 i:=1; 11 if upcase(m)<>m then begin 12 while i<length(s) do 13 if upcase(s[i])= 14 upcase(s[i+1]) 15 then begin 16 insert(m,s,i+1); 17 inc(i,2); 18 end 19 else 20 inc(i); 21 end; 22 writeln(s); 23 end.</pre>	<pre>#include <string.h> #include <stdio.h> char s[256],m,c1,c2; int i; void main() { gets(s); m=127; for(i=0;i<strlen(s);i++){ c1=s[i]>='A'&&s[i]<='Z'? s[i]+'a'-'A':s[i]; if (m>c1) m=c1; } for (i=0;i+1<strlen(s);i++){ c1=s[i]>='A'&&s[i]<='Z'? s[i]+'a'-'Z':s[i]; c2=s[i+1]>='A'&&s[i+1]<='Z'? s[i+1]+'a'-'A':s[i+1]; if (c1==c2) { memmove(s+i+1,s+i, strlen(s)-i); s[++i]=m; } } puts(s); }</pre>
---	---

13. Fișierul text „P.txt” conține pe mai multe linii un algoritm reprezentat în pseudocod. Știm că singurele cuvinte cheie ce se regăsesc în fișier sunt: *intreg*, *daca*, *atunci*, *altfel*, *citeste*, *scrie*, *stop*. Să se determine numărul de variabile folosite în algoritm și identicatorii acestora. Toate caracterele care intervin în fișier au coduri ASCII asociate. Liniile din fișier se termină cu caracterul punct și virgulă(;).

Exemplu: Pentru secvența „daca a>b atunci xx=c altfel d=a+b stop;” se va afișa:

5
a b xx c d

Soluție

Tabloul unidimensional *v* va memora toate variabilele identificate. Fișierul va fi parcurs caracter cu caracter, formându-se de fiecare dată o secvență ce poate reprezenta un cuvânt cheie sau identicatorul unei variabile. Dacă acesta nu reprezintă un cuvânt cheie, atunci este salvat în vectorul *v*, numai în cazul în care nu a fost deja memorat.

<pre>1 type 2 sir=array[1..7] of string[10]; 3 const a : sir=('intreg', 4 'citeste','scrie', 'daca', 5 'atunci','altfel','stop'); 6 var f:text; i,n,m:integer;</pre>	<pre>#include <string.h> #include <stdio.h> const char a[8][11]={ "intreg","citeste","scrie", "daca","atunci","altfel", "stop"};</pre>
--	--

<pre> 7 s:string; x:char; ok:boolean; 8 v:array[1..50] of string[10]; 9 begin 10 assign(f,'p.txt'); reset(f); 11 m:=0; 12 while not eof(f) do begin 13 s:=''; 14 while not eoln(f) do begin 15 read(f,x); 16 if x in ['a'..'z'] then 17 s:=s+x 18 else if s<>'' then begin 19 ok:=true; 20 for i:=1 to 7 do 21 if a[i]=s then ok:=false; 22 for i:=1 to m do 23 if s=v[i] then ok:=false; 24 if ok then begin 25 inc(m); v[m]:=s; 26 end; s:=''; 27 end; end; readln(f); end; 28 writeln(m); 29 for i:=1 to m do 30 write(v[i], ' '); 31 end.</pre>	<pre> int i,j,m,ok; FILE *f; char s[256],x[256],v[51][11]; void main() { f=fopen("p.txt","r"); for (m=0;!feof(f);) { fgets(x,256,f); strcat(x," "); for(i=0;i<strlen(x);i++) if(x[i]>='a'&&x[i]<='z') strncat(s,x+i,1); else if (strlen(s)) { ok=1; for(j=0;j<7;j++) if(!strcmp(a[j],s)) ok=0; for(j=0;j<m;j++) if(!strcmp(v[j],s)) ok=0; if (ok) strcpy(v[m++],s); memset(s,0,sizeof(s)); } } printf("%d\n",m); for(i=0;i<m;i++) printf("%s ",v[i]); }</pre>
--	---

14. Se consideră un șir de n cuvinte. Se dorește afișarea lor pe verticală, respectând următoarele cerințe:

- pe fiecare coloană se află în ordine câte un cuvânt, de la primul până la ultimul;
- pe ultima linie se află primele litere ale fiecărui cuvânt;
- pe prima linie se află ultimele litere ale celor mai lungi cuvinte.

Exemplu: Pentru șirul de 4 cuvinte: 'eu', 'masa', 'noi', 'vine' se va afișa:

```

a   e
s i n
u a o i
e m n v
```

Soluție

Se folosește ca auxiliar un vector h în care este memorată lungimea fiecărui cuvânt. Numărul de linii pe care se va face afișarea este egală cu elementul maxim din h . Pentru fiecare cuvânt, în cadrul unei linii, se poate afișa fie caracterul spațiu, dacă lungimea acestuia este mai mică decât valoarea variabilei mx (care contorizează poziția literei ce urmează a fi afișată), fie litera de pe poziția mx .

<pre> 1 type 2 sir=array[1..99] of string; 3 var a:sir; 4 l,i,n,mx:integer;</pre>	<pre> #include <string.h> #include <stdio.h> int n,i,max,h[100]; char a[100][256];</pre>
---	--

```

5  h:array[1..99]of byte;
6  begin
7      readln(n);mx:=0;
8      for i:=1 to n do begin
9          readln(a[i]);
10         h[i]:=length(a[i]);
11         if h[i]>mx then mx:=h[i]
12     end;
13     for l:=1 to mx do begin
14         for i:=1 to n do
15             if h[i]=mx then begin
16                 write(a[i][h[i]]);
17                 dec(h[i])
18             end
19             else write(' ');
20         dec(mx);
21         writeln;
22     end;
23 end.

```

```

void main() {
    scanf("%d",&n); max=0;
    for(i=0;i<n;i++){
        scanf("%s",a[i]);
        h[i]=strlen(a[i])-1;
        if (max<h[i]) max=h[i];
    }
    for(;max>=0;max--){
        for (i=0;i<n;i++)
            if (h[i]==max){
                printf("%c",a[i][h[i]]);
                h[i]--;
            }
        else printf(" ");
        printf("\n");
    }
}

```

15. Se citește de la tastatură un șir de maximum 255 litere mici ale alfabetului englez. Să se realizeze un program care identifică cea mai lungă secvență de caractere care se repetă de minimum 2 ori în cadrul șirului.

Exemplu: Pentru șirul de caractere: "masectrsecsacrrr" se va afișa „sec”.

Soluție

Algoritmul parcurge fiecare poziție din șir și identifică lungimea maximă a secvenței care poate începe cu caracterul respectiv. Variabila *x* reține această secvență.

```

1  var l,max,i:integer;
2  s,r,x,y:string;
3  begin
4      readln(s); max:=0;
5      for i:=1 to length(s)-1 do
6          begin
7              l:=0; x:='';
8              repeat
9                  inc(l); y:=s;
10                 x:=x+s[i+l];
11                 delete(y,i,length(x));
12             until (pos(x,y)=0)or
13                 (l+i>length(s));
14             if length(x)-1>max then begin
15                 max:=length(x)-1;
16                 r:=copy(x,1,max);
17             end;
18         end;
19     writeln(r);
20 end.
21

```

```

#include <string.h>
#include <stdio.h>
int l,max,i; char
s[256],r[256],x[256],y[256];
void main() {
    gets(s); max=0;
    for(i=0;i+1<strlen(s);i++){
        l=0; memset(x,0,sizeof(x));
        do {
            strncat(x,s+i+1,1);l++;
            strcpy(y,s);
            strcpy(y+i,y+i+strlen(x));
        } while (i+1<strlen(s) &&
            strstr(y,x));
        if (max<strlen(x)-1){
            max=strlen(x)-1;
            strncpy(r,x,max);
        }
    }
    puts(r);
}

```

1.1.3 Probleme propuse

1. Se consideră un text de maximum 255 de caractere. Realizați un program care afișează:

- a) Numărul de apariții al unei litere în text. Litera va fi citită de la tastatură.
- b) Câte vocale apar în textul citit.
- c) Numărul de apariții al unei silabe în text. Silaba va fi citită de la tastatură.

Exemplu:

Pentru litera 'a', silaba „re” și textul:
„Ina are multe mere”

se va afișa:

- a) 2
- b) 8
- c) 2

2. Se consideră un cuvânt format din litere mici și mari ale alfabetului englez. Realizați un program care permite ștergerea tuturor aparițiilor primei litere în cadrul cuvântului respectiv.

Exemplu : Pentru cuvântul 'mamaie' se va afișa 'aaie'.

3. Se consideră un cuvânt format din literele mici și mari ale alfabetului englez. Să se scrie un program care afișează cuvintele obținute din cuvântul inițial prin eliminarea succesivă a primului și ultimului caracter al șirului.

Exemplu: Pentru cuvântul 'Deosebit' se va afișa:

eosebi
oseb
se

4. Se consideră o matrice de dimensiune $n \times m$ cu elemente de tip șir de caractere. Creați un program care afișează șirul de caractere de lungime maximă de pe fiecare linie a matricei.

5. Să se creeze un program care transformă literele mici ale unui cuvânt în litere mari și literele mari în litere mici.

Exemplu: Pentru cuvântul 'MiorItic' se va afișa: 'mIORiTic'.

6. Se consideră un șir de n cuvinte. Să se determine cuvântul de lungime maximă care se poate forma prin concatenarea a două dintre cuvintele citite.

Exemplu: Pentru $n=5$ și șirul de cuvinte: 'mama', 'arc', 'conduce', 'paine', 'vine' se va afișa: 'conducepaine' sau 'paineconduce'.

7. Se consideră un șir de n cuvinte. Să se determine cuvântul cel mai mic în ordine lexicografică obținut prin concatenarea a două dintre cuvintele citite.

Exemplu: Pentru $n=5$ și șirul: 'mama', 'arc', 'conduce', 'paine', 'vine' se va afișa: 'arconduce'.

8. Considerăm un text de maximum 255 de caractere. Propozițiile sunt delimitate prin caracterele punct(.) sau prin semnul exclamării(!). Realizați un program care afișează fiecare propoziție pe o singură linie, fiecare cuvânt începînd cu o majusculă.

Exemplu : Pentru textul 'Fie A un punct fix.Presupunem B punct mobil.' se va afișa:

Fie A Un Punct Fix.

Presupunem B Punct Mobil.

9. Se consideră o listă cu n ($n < 100$) prenume ale elevilor dintr-o clasă. Prenumele unei fete este recunoscut datorită faptului că fie are ultima literă 'a', fie este 'Carmen' sau 'Alice'. Să se creeze un program care afișează numărul fetelor din clasă și cel mai mare prenume în sens lexicografic ale cărui litere vor fi transformate în majuscule.

Exemplu: $n=5$ și lista 'Ana', 'Alice', 'Mihai', 'Maria', 'Ion', se va afișa: '3 MIHAI'.

10. Se consideră 2 cuvinte ce conțin numai litere mici. Considerăm că ultima silabă a unui cuvânt este subșirul care începe cu ultima lui vocală. Verificați dacă aceste cuvinte rimează (dacă au ultima silabă identică). Dacă un cuvânt nu conține vocale, atunci ultima silabă este întregul cuvânt.

Exemplu: Pentru cuvintele 'armat' și 'verificat' se va afișa mesajul 'Rimeaza'.

11. De la tastatură se citește un text codificat după regula următoare: în fața fiecărui caracter este scris un număr ce reprezintă numărul de apariții consecutive al acestuia. Realizați un program care decodifică textul. Numărul ce apare în fața unui caracter va fi mai mic sau cel mult egal cu 20.

Exemplu : Pentru '1G11o1L' se va afișa 'GooooooooooooL'.

12. Se consideră un text în care spațiul este unicul separator. Realizați un program care afișează numerele ce apar în text, despărțite prin câte un spațiu.

Exemplu: Pentru textul 'Ana are 7 mere si 223 de pere' se va afișa: '7 223'.

13. Se citește de la tastatură un șir de caractere. Realizați un program care determină cea mai lungă secvență de cifre alăturate din șir.

Exemplu : Pentru șirul de caractere 'a23Bw001234mcv34' se va afișa '001234'.

14. Se dorește ca operația *Find-Replace* să fie executată asupra unui text care nu conține mai mult de 255 de caractere. Această operație constă în înlocuirea tuturor aparițiilor unui subșir $s1$ cu un alt subșir $s2$. Realizați un program care simulează această operație. Șirurile $s1$ și $s2$ nu au neapărat aceeași lungime.

Exemplu: Pentru textul 'are mere si pere' $s1='re'$ și $s2='rare'$ se va afișa: 'arare merare si perare'.

15. Se consideră un șir de n cuvinte. Realizați un program care identifică toate anagramele primului cuvânt care se regăsesc în șir.

Exemplu : pentru $n=5$, cuvintele 'arc', 'rac', 'eu', 'tu', 'car' se va afișa: 'rac' și 'car'

16. Se consideră o propoziție care are maximum 255 de caractere. Orice caracter diferit de literă este considerat separator. Realizați un program care afișează fiecare cuvânt pe o linie a ieșirii standard.

Exemplu: „Am venit!...” se va afișa:

Am
venit

17. Se consideră un text de maximum 255 de caractere. Realizați un program care determină cea mai lungă secvență de litere alăturate care apar în ordine alfabetică.

Exemplu: 'AMC WCDRVAS' se va afișa 'CDRV'.

18. Se consideră un text de maximum 255 de caractere. Realizați un program care determină cea mai lungă secvență de litere care reprezintă un palindrom.

Exemplu: 'AM tCOjOCn wqaaqd' se va afișa 'COjOC'.

19. Se consideră un text de maximum 255 de caractere litere mici sau spații. Realizați un program care rescrie textul astfel încât cuvintele să apară ordonate alfabetic.

Exemplu: Pentru șirul de caractere 'ieri am venit devreme' se va afișa:
'am devreme ieri venit'.

20. Se consideră un text de maximum 255 de caractere. Realizați un program care inversează literele fiecărui cuvânt:

Exemplu: Pentru șirul de caractere 'Am venit!...spune el' se afișează:
'mA tinev !...enups le'.

21. Fie un șir cu maximum 100 de caractere alfanumerice. Să se afișeze toate subșirurile formate din două caractere alăturate care pot reprezenta un număr natural de două cifre, separate prin câte un singur spațiu.

Exemplu: Pentru șirul 'w234b5br6779' se va afișa: '23 34 67 77 79'.

22. De la tastatură se citește un șir de maximum 10 caractere. Realizați un program care verifică dacă acest șir poate reprezenta un număr real exprimat în forma zecimală cu 3 zecimale exacte.

Exemplu: Pentru șirul „31.0.234” se va afișa mesajul 'NU'.

23. Creați un program care afișează numărul vocalelor dintr-o dată de tip șir de caractere împreună cu șirul obținut după ștergerea lor.

Exemplu: Pentru șirul de caractere: "moale" se va afișa: '3 ml'.

24. Se consideră o propoziție care are maximum 250 de caractere. Ea cuprinde cuvinte formate din literele alfabetului englezesc. Cuvintele sunt despărțite prin unul sau mai mulți separatori. Orice caracter care nu este literă va fi considerat separator. Creați un program ce va afișa pe ecran fiecare cuvânt din propoziție pe câte o linie. Cuvintele vor avea prima literă majusculă.

Exemplu: Pentru propoziția:

'-Ai venit? intreba Alina' se va afișa pe ecran:

Ai

Venit

Intreba

Alina

25. Se citește de la intrarea standard un șir de maximum 200 de caractere reprezentând o propoziție. Cuvintele sunt secvențe de litere mici sau mari ale alfabetului englezesc. Realizați un program care afișează propoziția după ce cuvintele au fost ordonate lexicografic. După fiecare cuvânt vor fi afișați separatorii prezenți în propoziția inițială după cuvântul cu numărul de ordine respectiv.

Exemplu: Pentru textul „Ce spui.....?, a intrebat el.” se va afișa:

'Ce a.....?, el intrebat spui'.

26. Se consideră un șir de cel mult 100 de caractere, citit de pe prima linie a fișierului *in.txt*. Să se afișeze toate șirurile de caractere de lungime *maximă* $2*k$ obținute prin concatenarea prefixelor cu sufixele de aceeași lungime ale acestui șir. Valoarea lui k se citește de la tastatură.

Exemplu:

Pentru $k=4$

și fișierul *in.txt*

caracatita

se va afișa:

ca

cata

carita

caratita

27. Fișierul *in.txt* conține un text dispus pe mai multe linii. Orice caracter care nu reprezintă o literă mică a alfabetului englezesc se consideră separator. Numărul de caractere ale unei linii nu depășește 200. Realizați un program care afișează numărul de apariții ale unui cuvânt, preluat de la tastatură, în textul din fișier.

Exemplu: Pentru cuvântul „venit”

se va afișa: 3

și fișierul *in.txt*

Am venit acasa!

EA a,! venit!"

el a venit acum**?!

28. Conținutul fișierul *virus.txt* a fost deteriorat, iar valorile naturale care inițial erau separate în cadrul liniilor prin câte un singur spațiu, au acum ca separatori orice caracter ce nu reprezintă o cifră. Realizați un program care creează fișierul *sum.txt* în care pe fiecare linie se află suma numerelor aflate în fișierul *virus.txt*.

Exemplu: Pentru fișierul *virus.txt*

w23 , ,mmm230...20e

fsdj.4.sal..45,,sk56ddd90z

fișierul *sum.txt* va conține:

273

195

29. Se consideră o propoziție de maximum 50 de caractere, citită de la tastatură. Să se realizeze un program care afișează propoziția după eliminarea tuturor cuvintelor de lungime p .

Exemplu: Pentru $p=3$ și propoziția „Noi am gandit la fel ca voi...!” se va afișa:
„am gandit la ca ...!”

30. Fie un cuvânt de maximum 50 de caractere litere mici ale alfabetului englez. Să se afișeze toate cuvintele distincte obținute prin eliminarea unor secvențe de p litere aflate pe poziții consecutive în cuvântul inițial. Cuvintele afișate vor fi despărțite prin câte un spațiu.

Exemplu: Pentru cuvântul „mamaie” și $p=2$ se va afișa: 'maie mama mame'.

31. Se citesc de la tastatură două cuvinte formate doar din literele minuscule ale alfabetului englezesc. Se cere realizarea unui program care afișează:

- mulțimea literelor comune celor două cuvinte(intersecția);
- mulțimea literelor care se găsesc în unul din cele două cuvinte (reuniunea);
- mulțimea literelor care se găsesc în primul cuvânt citit și nu apar în al doilea.

Exemplu: Pentru cuvântul „mamaie” și cuvântul „macara” se va afișa:

{m,a}
{m,a,i,e,r,c}
{i,e}

32. Fișierul *in.txt* conține un text, fiecare propoziție fiind dispusă pe o singură linie. Afișați pe ecran propoziția de lungime maximă, obținută după eliminarea tuturor secvențelor de două caractere consecutive, ambele vocale.

Exemplu: Pentru propoziția:

Mama_Are_Mere
Maine_este_o_noua_zi

se va afișa:

Mne_este_o_na_zi

33. Se citește de la tastatură o propoziție de maximum 200 de caractere. Afișați pe ecran, pe mai multe linii, propoziția trunchiată astfel: pe ultima linie va fi afișat ultimul caracter al propoziției, pe penultima, următoarele ultime două caractere(antepenultimul și penultimul), ș.a.m.d. Pe prima linie se vor găsi primele n caractere ale propoziției, unde n este mai mic sau egal cu numărul de linii pe care s-a făcut afișarea.

Exemplu: Pentru propoziția:

abcdefghijkl

se va afișa:

ab
cdef
ghi
jk
l

34. Să considerăm următorul șir:

a, b, ba, bba, bbaba, bbabbaba, bbabbababbaba

Scrieți un program care să determine care este cel de-al n -lea termen al șirului. Valoarea lui n este strict mai mică decât 20.

Exemplu: Pentru $n=6$ se va afișa: 'bbabbaba'.

35. Se consideră un șir de n cuvinte reprezentând numele disciplinelor din orarul zilei următoare ale unui elev. Realizați un program care permite afișarea orarului pe ieșirea standard, astfel încât materiile să fie scrise pe coloane, în ordine lexicografică.

Exemplu: Pentru $n=4$ și disciplinele „muzica”, „desen”, „sport”, „biologie”, se va afișa sub forma:

```
b d m s
i e u p
o s z o
l e i r
o n c t
g   a
i
e
```

36. Se consideră un text ce conține maximum 100 de cuvinte, fiecare cuvânt fiind format din cel mult 50 de litere ale alfabetului englez. Orice alt caracter este considerat separator. Textul este dispus pe mai multe linii în fișierul *in.txt*. Să se afișeze pe o singură linie a ieșirii standard, cuvintele de lungime maximă din text, separate prin câte un spațiu.

Exemplu: *in.txt*

```
Am venit . . . acasa
maine , plec!
Tu vrei sa pleci****?
```

se va afișa:

venit acasa maine pleci

37. În fișierul *in.txt* se află dispuse pe mai multe linii informații despre ora de plecare a trenurilor și destinația lor finală. Momentul plecării din stație este exprimat prin 5 caractere, primele două reprezentând ora, iar ultimele două reprezentând minutele(*hh:mm*). În continuare, în cadrul liniei se află numele destinației. Afișați, pe câte o linie a ieșirii standard, lista destinațiilor în ordinea crescătoare a momentului plecării. Pe fiecare linie se va afișa numele destinației și ora plecării, separate printr-un spațiu. Ora de plecare va fi exprimată în minute.

Exemplu: *in.txt*

```
10:23Cluj
15:04Arad
05:00Iasi
```

se va afișa:

```
Iasi 300
Cluj 623
Arad 904
```

38. Considerăm validă o adresă de e-mail dacă este formată doar din litere mici ale alfabetului englez, din caracterul '.' și caracterul '@' care apare o singură dată. Caracterele alăturate acestuia nu pot fi puncte '.'. Se codifică o adresă de e-mail după următoarele reguli:

- caracterul '@' este înlocuit prin secvența 'at' numai în cazul în care nu este precedat sau urmat de caracterul punct.
- secvența de caractere 'cod' poate fi introdusă oriunde în adresă, dar nu mai mult de o dată.

Dându-se o adresă codată, afișați pe câte o linie adresele de e-mail din care ar fi putut proveni.

Exemplu:

Pentru adresa:

sco.at.matatam.icodm

se va afișa:

sco.at.m@atam.icodm

sco.at.m@atam.im

sco.at.mat@am.icodm

sco.at.mat@am.im

39. O formulă este restrânsă folosind parantezele rotunde astfel: șirul „ababab” va fi scris simplificat „(ab)3” iar șirul „axzyxzyw” va fi scris „a(xzy)2w”. Formula restrânsă se citește de la intrarea standard. Știind că nu sunt prezente perechi de paranteze incluse una în alta, afișați șirul inițial.

Exemplu: Pentru șirul s(ad)3f(ser)2, se va afișa ”sadayadfserser”.

40. Se citește, de la intrarea standard, un șir de maximum 200 de caractere. Realizați un program care determină cea mai lungă secvență de caractere ordonate lexicografic. În situația în care există mai multe astfel de secvențe se vor afișa toate, fiecare pe câte o linie a ieșirii standard.

Exemplu: Pentru textul „ABCADEMIA” se va afișa

ABC

AED

41. Se citește de la intrarea standard un șir de maximum 200 de caractere. Realizați un program care determină cea mai lungă secvență de caractere cu proprietate palindromică. În situația în care există mai multe astfel de secvențe se vor afișa toate, fiecare pe câte o linie a ieșirii standard.

Exemplu: Pentru textul „Acojoc tar eabbbar” se va afișa

cojoc

abba

42. Se consideră o expresie aritmetică ce conține numai operatorii {+,*}. Operanzii sunt reprezentați prin variabile ai căror identificatori sunt descriși printr-o singură literă, sau prin numere naturale. Realizați un program care verifică dacă expresia este corectă sintactic.

Exemplu: Pentru expresia 23*a+ba*+3 se va afișa „Incorect”.

Pentru expresia 23*a+b*3 se va afișa „Corect”.

43. Se consideră un șir de paranteze rotunde care intervin într-o expresie aritmetică. Realizați un program care verifică dacă acestea formează un șir de paranteze care se închid corect (după regulile aritmetice).

Exemplu:

Pentru șirul de paranteze „(() (()))” se afișează mesajul „Corect”.

Pentru șirul de paranteze „()) (() ()” se afișează mesajul „Incorect”.

44. Se consideră următoarea regulă de codificare a unui cuvânt: în fața oricărei vocale a cuvântului se inserează caracterul ‘p’. Realizați un program care realizează operația de codificare și de decodificare a unui cuvânt. De la tastatură se va introduce tipul operației dorite prin caracterele: C pentru codificare și D pentru decodificare.

Exemplu:

Pentru operația C și cuvântul „oaie” se va afișa: „popapipe”.

Pentru operația D și cuvântul „pparpintpe” se va afișa: „parinte”.

45. Se consideră un număr rațional n ($n \leq 30000$) care are partea fracționară formată din maximum 6 cifre. Să se scrie un program care efectuează conversia numărului n în baza 2. Datele se citesc de la tastatură și se afișează pe ecran.

Exemplu: Pentru $n=7.375$ se va afișa:

„111.011”.

46. În fișierul text *baze.in* sunt dispuse pe fiecare linie câte un număr. În scrierea lor intervin cifrele de la 0..9 și literele de la A la M (reprezentând, în ordine, valorile 10, 11, 12, ...). Baza în care sunt reprezentate acestea se consideră a fi baza minimă comună tuturor. Realizați un program care determină intervalul $[a,b]$, de lungime minimă, în care sunt incluse toate numerele din fișier, în urma conversiei în baza 10.

Exemplu:

Pentru fișierul *baze.in*

07B

1AA

C8A

se va afișa:

a=102

b=2142

47. Se consideră o secvență de n șiruri de caractere, fiecare reprezentând o parolă formată din maximum 20 de caractere (cifre 0..9 sau majuscule). Din fiecare parolă se elimină succesiv caracterele de la stânga spre dreapta până la cel mai mare caracter al său în sens lexicografic. De exemplu, parola „A250B089” devine „B089”. În urma acestei operații, unele dintre parole devin identice și sunt considerate invalide. Ele sunt înlocuite cu suma dintre poziția ei în șir și suma codurilor ASCII a caracterelor primei parole valide rămase în șir. Operația devine imposibilă dacă și noua parolă obținută este invalidă. Realizați un program care afișează șirul parolelor după aceste transformări sau mesajul „Imposibil”.

Exemplu: Pentru $n=3$ și șirurile

07BA023

AABBAB

A07BA023

se va afișa:

281

BBAB

283