7. Strukturált típusok

1. Mintafeladat a különböző tömbtípusok konstanssal való feltöltésére és kiíratására! (*minta7_1*)

```
program minta7_1;
const
  fejlec:array[1..8] of char = 'Eredmény';
  adatok:array[1..4] of integer=(1,2,3,4);
  matrix:array[1..2,1..3] of real=((1,20,12),(3,4,7));
var
   i, j: integer;

begin
  writeln('fejléc tartalma');
  writeln(fejlec);
  writeln('adatok tömb tartalma');
  for i:=1 to 4 do
    writeln(adatok[i]);
  writeln('matrix tömb tartalma');
  for i:=1 to 2 do
  begin
    for j:=1 to 3 do
      write(matrix[i,j]:8:2);
    writeln;
  end;
  readln;
end.
```

A program futásának eredménye:

```
fejléc tartalma
Eredmény
adatok tömb tartalma
1
2
3
4
matrix tömb tartalma
    1.00   20.00   12.00
    3.00    4.00    7.00
```

2. Mintaprogram a különböző típusú tömbök egy elemének értékadására! (*minta7_2*)

```
program minta7_2;
const
    ar_db = 10;
var
   darab: array[1..10] of integer;
   ara  : array[1..ar_db] of real;
```

```
   logikai: array[0..5] of boolean;
   karakterek : array[1..3] of char;
begin
  darab[1]:= 12;
  ara[1]:= 12.56;
  logikai[1]:= true;
  karakterek[1]:='a';
  writeln(darab[1]:3,ara[1]:8:2,karakterek[1]:4);
  readln;
end.
```

A program futásának eredménye:

```
 12   12.56   a
```

3. Mintafeladat különböző típusú kétdimenziós tömb első elemének értékadására! (*minta7_3*)

```
program minta7_3;
var
   x  : array[1..3,1..3] of integer;
   y  : array[1..2,1..2] of real;
begin
  x[1,1]:= 122;
  y[1,1]:= 12.56;
  writeln(x[1,1]);
  writeln(y[1,1]);
  readln;
end.
```

A program futásának eredménye:

```
122
 1.25600000000000E+001
```

4. Mintafeladat a tömb típusának definálására, valamint ezzel a típussal deklarált tömbök értékadására! (*minta7_4*)

```
program minta7_4;
const
    ar_db = 10;
type
   tomb1 = array[1..10] of integer;
   tomb2 = array[1..ar_db] of real;
var
   darab: tomb1;
   ara  : tomb2;
   logikai: array[0..5] of boolean;
   karakterek : array[1..3] of char;

begin
  darab[1]:= 12;
  ara[1]:= 12.56;
```

```
  logikai[1]:= true;
  karakterek[1]:='a';
  writeln(darab[1]:3,ara[1]:8:2,karakterek[1]:4);
  readln;
end.
```

A program futásának eredménye:

```
 12   12.56   a
```

5. Mintafeladat a **const**, **type** használatára, valamint a konstansként deklarált tömbök feltöltésére! (*minta7_5*)

```
program minta7_5;
const
    ar_db = 4;
type
   tomb1 = array[1..4] of integer;
const
   darab : tomb1 =  (5,2,7,4);
   ara   : array[1..ar_db] of real = (1.2,2.3,4.1,5.3);
   a     : array[1..3,1..2] of integer = ((1,2),(3,4),(5,6));
var
   logikai: array[0..5] of boolean;
   karakterek : array[1..3] of char;
begin
  writeln(darab[1]:4,ara[1]:6:2);
  logikai[1]:= true;
  karakterek[1]:='a';
  writeln(a[1,1]:3,a[1,2]:3);
  writeln(a[2,1]:3,a[2,2]:3);
  writeln(a[3,1]:3,a[3,2]:3);
  readln;
end.
```

A program futásának eredménye:

```
   5  1.20
  1  2
  3  4
  5  6
```

6. Írjunk programot, amely beolvassa a tömb elemszámát és ellenőrzi, hogy a megengedett indexhatárba esik-e! Olvassuk be a tömb adatait, majd a páros adatok összegét és átlagát, a páratlan adatok szorzatát számítsuk ki! Az eredményeket jelenítsük meg! (*tomb01*)

7. Írjunk programot, amely beolvassa a tömb elemszámát és ellenőrzi, hogy a megengedett indexhatárba esik-e! Olvassuk be a tömb adatait. Keressük meg a tömb legkisebb és a legnagyobb elemét, majd számítsuk a legnagyobb és a legkisebb elem átlagát! (*tomb02*)

8. Írjunk programot, amely beolvassa a tömb elemszámát és ellenőrzi, hogy a megengedett indexhatárba esik-e! Olvassuk be a tömb adatait. Számláljuk meg a tömb páros és páratlan adatainak számát! (*tomb03*)

9. Írjunk programot, amely beolvassa a tömb elemszámát és ellenőrzi, hogy a megengedett indexhatárba esik-e! Olvassuk be a tömb adatait. Külön tömbbe tároljuk a 4-gyel és a 3-mal osztható adatokat! Írassuk ki az eredmény tömbök tartalmát! (*tomb04*)

10. Írjunk programot, amely beolvassa a tömb elemszámát és ellenőrzi, hogy a megengedett indexhatárba esik-e! Olvassuk be a tömb adatait! Számláljuk meg a tömb pozitív, negatív és zérus elemeinek számát! (*tomb05*)

11. Írjunk programot, amely beolvas egy szöveget és számláljuk meg, hogy magánhangzónként hány betűt tartalmaz! (*tomb06*)

12. Írjunk programot, amely beolvas egy szöveget! Bontsuk a szöveget szavakra! A szavak elválasztásánál vegyük figyelembe a vesszőt, a szóközt és a mondatzáró pontot! (*tomb07*)

13. Írjunk programot, amely karakterenként tölti fel a sztringet! A záró karakter a . (pont). A beolvasott mondatot szavanként írjuk vissza! (*tomb08*)

14. Írjunk programot, amely beolvassa a konstansként megadott elemszámú elem adatait, és az *index* tömbben tárolt index alapján számítsuk ki az adatokkal feltöltött tömb *index*-edik adatának összegét! (*tomb09*)

15. Írjunk programot, amely a konstansként megadott mátrixot feltölti a sor és az oszlop indexek összegével! Írassuk ki az így feltöltött mátrixot! (*tomb10*)

16. Írjunk programot, amely beolvassa a 10 elemű egész típusú tömb adatait és egy másik tömböt úgy tölti fel a beolvasott adatokkal, hogy a páros elemeket a tömb első indexétől, a páratlan elemeit az utolsó (10) indexétől tárolja! (*tomb11*)

17. Írjunk programot, amely beolvassa a megadott elemszámú tömb adatait! Vizsgáljuk meg, hogy az adatok számtani sorozatot alkotnak-e vagy nem! (*tomb12*)

18. Írjunk programot, amely beolvassa a megadott elemszámú tömb adatait! Számítsuk ki a páros adatok szorzatát és a páratlan adatok összegét! (*tomb13*)

19. Írjunk programot, amely beolvassa a megadott elemszámú tömb adatait! Keressük ki a tömb legnagyobb és legkisebb adatát úgy, hogy jegyezzük meg az adathoz tartozó indexet is! (*tomb14*)

20. Írjunk programot, amely beolvassa egy adott elemszámú tömb adatait! Keressük meg a tömb legnagyobb adatát és ezzel az adattal osszuk végig az összes elemét! A normált adatokat jelenítsük meg! (*tomb15*)

21. Írjunk programot, amely beolvassa egy 3x3-as mátrix adatait! Számítsuk ki az alsó háromszög mátrix elemeinek összegét, a főátlóbeli elemek szorzatát, valamint az alsó háromszög elemeinek szorzatát! (*tomb16*)

22. Írjunk programot, amely egy 3x3-as mátrix főátlójába 2, felső háromszög elemeibe 1-et, és az alsó háromszög elemeibe 3-at tesz! Számítsuk ki a háromszög sorainak összegét és szorzatát! (*tomb17*)

23. Írjunk programot, amely beolvas max. 80 karakter hosszú szöveget! Számláljuk meg a szövegben lévő magánhangzók és egyéb karakterek számát! (*tomb18*)

24. Írjunk programot, amely a *teglalap* rekordban definiált téglalap két oldalát olvassuk be! Számítsuk ki a téglalap területét és kerületét! (*rekord01*)

```
var
  teglalap : record
               a_oldal, b_oldal: real;
               ter, ker: real;
             end;
```

25. Írjunk programot, amely beolvassa *muvelet* típusú rekord két operandusát és a műveleti jelét! Számítsuk ki a megadott adatokkal a kijelölt műveletet! (*rekord02*)

```
type
     muvelet = record
               a, b, eredmeny: real;
               muvjel: char;
             end;
var
     m1: muvelet;
```

26. Írjunk programot, amely beolvassa a *tavolsag* típusú rekord koordinátapontjainak *x* és *y* koordinátáit! Számítsuk ki a két koordinátapont közötti távolságot! (*rekord03*)

```
type
     pont = record
               x, y: integer;
             end;
```

```
     tavolsag = record
             p1, p2 : pont;
             tav: real;
           end;
var  t: tavolsag;
```

27. Írjunk programot, amely beolvassa *teglatest_típus* típusú rekord három oldalát. Számítsuk ki a téglatest térfogatát és felszínét! (*rekord04*)

```
type
  teglatest_tipus = record
                     a_oldal, b_oldal, c_oldal: real;
                     terfogat, felszin: real;
                    end;
var
     tg: teglatest_tipus;
```

28. Írjunk programot, amely beolvassa a megadott elemszámú tömb *butor* típusú elemeinek adatait! Keressük meg a legdrágább és a legolcsóbb bútor azonosítóját és árát! Számláljuk meg az első osztályú és a másodosztályú bútorok számát! (*rekordt01*)

```
const butor_db = 25;
type
     butor = record
               azonosito: string;
               db: integer;
               ar: real;
               minoseg: integer;
             end;
     raktar = array[1..butor_db] of butor;
```

29. Írjunk programot, amely beolvassa a megadott elemszámú tömb *szemely* rekord elemeinek adatait! Keressük meg a legnagyobb fizetésű, a legkevesebb szabadságú, legidősebb, a legfiatalabb rekordját! Számítsuk ki a fizetések összegét és az átlagfizetést! (*rekordt02*)

```
const max_letszam = 25;
type
     szemely = record
               nev: string[20];
               kora: integer;
               lakhely: string[20];
               fizetes: real;
               szabadsag: integer;
             end;
     munkahely = array[1..max_letszam] of szemely;
```

30. Írjunk programot, amely beolvassa a tomb típuső struktúra adatait, melyben egy 5 elemű tömböt kell feltölteni adatokkal! Számítsuk ki az adatok összegét, átlagát és szorzatát! (*rekordt03*)

```
const db = 5;
type tomb = record
              x : array[1..db] of integer;
              osszeg, szorzat: integer;
              atlag: real;
            end;
```

31. Írjunk programot, amely beolvassa a megadott elemszámú tömb *tavolsag* rekord típusú elemeinek adatait, azaz két koordinátapontnak az x és y koordinátáit! Számítsuk ki a két koordinátapont távolságát és keressük meg a legkisebb és a legnagyobb távolságot! (*rekordt04*)

```
type
     pont = record
               x, y: integer;
             end;
     tavolsag = record
             p1, p2 : pont;
             tav: real;
           end;
     tavTomb = array[1..10] of tavolsag;
var
     t: tavTomb;
```

32. Írjunk programot, amely beolvassa a megadott elemszámú *muhely* típusú tömb *auto* típusú rekordjainak mezeit! Keressük meg a legnagyobb javítási költségű autót, a legrégebbi és a legfiatalabb kocsi gyártási évét! Számítsuk ki az össz javítási és az össz vizsga költséget! (*rekordt05*)

```
const max_autoszam = 12;
type
     auto  = record
               tipus: string[15];
               gyartasiev: integer;
               rendszam: string[7];
               javitas_ara: real;
               vizsgakoltseg: real;
             end;
     muhely = array[1..max_autoszam] of auto;
```

33. Írjunk programot, amely beolvassa a megadott elemszámú *raktar* típusú tömb *aru* típusú rekord elemeit! Számítsuk ki az össz értéket, átlag árat és az össz darabszámot! (*rekordt06*)

```
type
     aru = record
               azonosito: string[20];
               cikkszam: string[7];
               ar: real;
               db: integer;
             end;
     raktar = array[1..40] of aru;
```

34. Írjunk programot, amely beolvassa a megadott elemszámú *adat_tomb* típusú tömb *adat* típusú rekordjainak *x, y* elemeit! Számítsuk ki a két adatának számtani és mértani közepét, a számtani közepek átlagát és a mértani közepek összegét! (*rekordt07*)

```
type
     adat = record
               x, y : real;
               szk, mk: real;
            end;
     adat_tomb = array[1..20] of adat;
```

35. Írjunk programot, amely beolvassa a megadott elemszámú *tankor* típusú tömb *diak* típusu rekord elemeit! Keressük meg a legjobb átlagot és a legkisebb kreditpontot! (*rekordt08*)

```
type
     diak = record
               nev: string;
               kreditpont: integer;
               atlag: real;
            end;
     tankor = array[1..20] of diak;
```

36. Írjunk programot, amely halmazokon végez különféle halmazműveleteket és az eredményt megjeleníti! (*halmaz1*)

```
type
   ABC = (A,B,C,D,E,F,G,H);
   halmaz = set of ABC;
```

37. Írjunk programot, amely egy tíz elemű egész típusú tömb minden elemét elhelyezi a csere irányától függően eggyel nagyobb, vagy eggyel kisebb indexű helyre, (a tizedik elemet az első, ill. az első elemet a tizedik helyére), majd írassuk ki a keletkezett tömböt (*csere*)

38. Írjunk programot, amelyben hozzunk létre és írassunk vissza egy 5*7-es kétdimenziós tömböt úgy, hogy a tömb elemei a két index összegét vegyék fel! (*ketdim*)

39. Írjunk programot, amelyben beolvasunk egy max. 40 karakter hosszúságú karaktersorozatot, és jelenítsük meg a képernyőn, a sor közepére illesztve. Egy képernyősorban 80 karakter helyezhető el! (*kozepre*)

40. Írjunk programot, amely beolvas egy max 80 karakter hosszúságú sztringet! Vizsgáljuk meg, hogy a sztringben hol fordulnak elő numerikus karakterek és a pozíciójukat is jelenítsük meg! (*numerikus*)

41. Írjunk programot, amely egy n*n-es kétdimenziós tömböt listáz sorfolytonosan, oszlopfolytonosan, csak a főátlóban lévő elemeket! (*tomblist*)