



Algoritmusok és adatstruktúrák 1. előadás

Pusztai Pál
pusztai@sze.hu

Tartalom

- Egy mintafeladat – megoldási lépések
- Egyszerű adattípusok
 - Egész, karakter, logikai és valós adattípusok
 - Konstansok, műveletek
- Változók és kifejezések
 - Prioritás, kiértékelés
- Függvények
 - Matematikai és konverziós függvények
- Az értékadó, a beolvasó és a kiíró utasítás
- Az adatszerkezeti táblázat
- Konstansok és típusok
 - Deklarálás és használat



Bevezetés

- Ha egy feladat megoldására **számítógépes programot** készítünk, akkor általában az alábbi lépéseket, tevékenységeket kell elvégeznünk:
 1. A feladat megfogalmazása, pontosítás, általánosítás.
 2. Matematikai (vagy egyéb) modell kiválasztása, megadása (ha szükséges ill. lehetséges).
 3. Az adatszerkezet definiálása, az input-output specifikálása.
 4. A megoldást megvalósító algoritmus megtervezése, elkészítése.
 5. Programírás, kódolás (az adatszerkezet és az algoritmus alapján).
 6. Tesztelés, hibakeresés.
 7. Dokumentálás (felhasználóknak, fejlesztőknek).

Mintafeladat

- **Feladat:** Alakítsunk át egy pozitív egész számot egy adott (2-16) számrendszerbe!
 1. **Általánosítás:** a megoldást nem két konkrét adatra, hanem megadható, input adatokra készítjük el, így „tetszőlegesen” konvertálhatunk.



Mintafeladat

2. Matematikai modell: ez maga az algoritmus, miszerint:

- Először a megadott számot, utána a keletkező hányadosokat mindaddig osszuk el az adott számrendszer alapszámával, amíg a hányados nulla nem lesz!
- Minden osztásnál jegyezzük fel az osztás maradékát!
- Ezekből, mint számjegyekből, a feljegyzésük fordított sorrendjében számot képezve, a megoldást kapjuk.

40	0
20	0
10	0
5	1
2	0
1	1
0	

40-et 2-s számrendszerbe

40	8
2	2
0	

40-et 16-s számrendszerbe

Mintafeladat

3. Adatszerkezet: az adatszerkezeti táblázattal adjuk meg.

- Tekintettel a hexadecimális számrendszer betűvel jelölt számjegyeire, ill. a nagy egész számok (4 bájt) esetén a kettes számrendszerbeli, esetlegesen 32 számjegyes eredményre, az eredmény típusa csak sztring lehet.

Funkció	Azonosító	Típus	Jelleg
Az átalakítandó szám	A	Egész	I
A számrendszer alapszáma	B	Egész	I
Az eredmény „szám”	ER	Sztring	M, O

Mintafeladat - pszeudokód

4. Algoritmus:

- A modell alapján elkészíthető az algoritmus.
- A lehetséges számjegyeket egy sztringkonstansban deklaráljuk:

Konstans

SZAMJEGYEK "0123456789ABCDEF"

KONVERTAL(A,B)

ER \leftarrow ""

repeat

ER \leftarrow SZAMJEGYEK[A MOD B+1]+ER

A \leftarrow A DIV B

until A=0

return ER

Mintafeladat - Python

5. Programírás:

- A megoldások *Python* és *Pascal* nyelvű programjai letölthetők a www.sze.hu/~pusztai honlapról.
- Ezt a mintafeladatot *C*, *Basic* és *Assembly* nyelven is programmá írtuk.

```
def konvertal(a, b):  
    szamjegyek = "0123456789ABCDEF"  
    er, kilep = "", False  
    while not kilep:  
        er = szamjegyek[a % b] + er  
        a = a // b  
        kilep = a == 0  
    return er
```



Mintafeladat - Pascal

```
function Konvertal(a,b:integer):string;  
var er:string;  
begin  
    er:='';  
    repeat  
        er:=SzamJegyek[a mod b+1]+er;  
        a:=a div b;  
    until a=0;  
    Konvertal:=er;  
end;
```



Mintafeladat - C

```
void Konvertal(int a, int b, char *er)
{ int i,j; char cs;
  /* Osztogetas */
  i=0;
  do {
    er[i++]=SzamJegyek[a%b];
    a=a/b;
  } while (a>0);
  /* Végjel */
  er[i]='\0';
  /* Megfordítás */
  for (j=0,i--; j<i; j++,i--) {
    cs=er[j]; er[j]=er[i]; er[i]=cs;
  }
}
```



Mintafeladat - Visual Basic

```
Function Konvertal(a As Integer, b As Integer) As String
    Dim er As String
    er = ""
    Do
        er = Mid(SzamJegyek, a Mod b + 1, 1) + er
        a = a \ b
    Loop Until a = 0
    Konvertal = er
End Function
```



Mintafeladat - Assembly

```

; Input:  AX = a konvertálandó pozitív egész szám
;         BX = a cél számrendszer alapszáma
;         DI = az eredményterület offset címe
; Output: DS:DI = az eredmény sztring kezdőcíme
;         CX   = az eredmény sztring hossza
; Elromló regiszterek: AX,DX,DI,SI
Konv      PROC
            mov     cx,0
Oszt:     mov     dx,0                ;Az előző maradék törlése
            div     bx                ;Osztás
            inc     cx
            push    dx                ;Maradékot a verembe
            cmp     ax,0
            jne     Oszt
            mov     dx,cx             ;CX megjegyzése
Cikl:     pop     si                 ;Az eredmény előállítása
            mov     al,SzJ[si]        ;AL-be a megfelelő karaktert
            mov     [di],al           ;A célterületre írjuk
            inc     di
            loop    Cikl
            mov     cx,dx             ;CX beállítása
            ret                     ;Visszatérés a hívóhoz
Konv      ENDP
    
```



Egész típus

Egész számok használatát megengedő adattípus.

- Konstans:

Pl. -326

- Műveletek:

- Előjel (+, -)

- Multiplikatív:

- Szorzás (*)

- Egész osztás hányadosa (//, DIV)

- Egész osztás maradéka (% , MOD)

- Additív:

- Összeadás (+), Kivonás (-)

- Hasonlítások:

- Egyenlő (=), Nem egyenlő (<>), Kisebb (<), Nagyobb (>),

- Kisebb vagy egyenlő (<=), Nagyobb vagy egyenlő (>=)

Pl. $5//3 \rightarrow 1$ $5 \text{ DIV } 3 \rightarrow 1$ $5 \text{ MOD } 3 = 5 \% 3 \rightarrow \text{igaz}$



Karakter típus

Karakterek használatát megengedő adattípus.

- Konstans:

Pl. 'A', #27

- Műveletek:

- Összefűzés (+)

- Hasonlítások:

- Mint az egészeknél, csak itt nem a számok értéke, hanem a karakterek ASCII kódja szerint történik a hasonlítás

Pl. 'A' < 'B' → igaz 'T' + 'ó' → "Tó"



Logikai típus

- Konstans:
 - igaz, hamis
- Műveletek:
 - Tagadás (NOT), És (AND), Vagy (OR)
 - Hasonlítások:
 - Mint az egészeknél

Pl. $\text{hamis} < \text{igaz} \rightarrow \text{igaz}$

A	B	NOT A	A AND B	A OR B
igaz	igaz	hamis	igaz	igaz
igaz	hamis	hamis	hamis	igaz
hamis	igaz	igaz	hamis	igaz
hamis	hamis	igaz	hamis	hamis

Valós típus

Valós számok használatát megengedő adattípus.

- Konstans:

Pl. 3.14

- Műveletek:

- Mint az egészeknél, kivéve az osztás (/) műveletét. Az egész osztás műveletek (//, DIV, %, MOD) valós számokra nem értelmezettek.

Pl. $3/2 \rightarrow 1.5$

- Megjegyzés:

- A programfejlesztő rendszerek általában többféle egész ill. valós típus használatát engedik meg.
- Adattárolás/számábrázolás
 - Egész számok: fixpontos (1, 2, ill. 4 bájt, előjelesen, ill. anélkül)
 - A 4 bájtos előjeles egészek értékkészlete: -2,147,483,648.. 2,147,483,647
 - Valós számok: lebegőpontos (4, ill. 8 bájt, előjelesen)
 - 4 bájt esetén a nagyságrend: kb. 1E38, pontosság: 7-8 decimális számjegy
 - 8 bájt esetén a nagyságrend: kb. 1E308, pontosság: 15-16 decimális számjegy



Változók, kifejezések

- Változó:
 - Olyan azonosítóval ellátott memóriaterület, ahol a változó típusának megfelelő adatot tárolhatunk.
 - A **változó értékén** az éppen benne tárolt adatot értjük.
- Kifejezés:
 - Olyan számítási műveletsor, amellyel megmondjuk, hogy milyen adatokkal, milyen műveleteket, milyen sorrendben kívánunk elvégezni.
 - A kifejezés kiértékelése után egy új érték – a **kifejezés értéke** – keletkezik.
- A kifejezésben szerepelhetnek:
 - Konstansok, változók, függvényhívások
 - Műveletek
 - Zárójelek

Pl. $(-B + \text{SQRT}(B*B - 4*A*C)) / (2*A)$



Prioritás, kiértékelés

- A műveletek prioritása csökkenő erőssorrendben:
 - Egyoperandusú: Előjel (+, −), Tagadás (NOT)
 - Multiplikatív (*, /, DIV, //, MOD, %, AND)
 - Additív (+, −, OR)
 - Hasonlítások (=, <>, <, >, <=, >=, IN)
- A kifejezések kiértékelésének szabályai:
 - A zárójelbe tett kifejezések és függvényhívások operandus szintre emelkednek.
 - A magasabb prioritású műveletek végrehajtása megelőzi az alacsonyabb prioritású műveletek végrehajtását.
 - Az azonos prioritású műveleteknél: balról-jobbra szabály.
 - A logikai kifejezéseknél: rövid kiértékelés.
 - A numerikus műveletek eredménye
 - egész, ha az operandusok egészek és a művelet nem az osztás (/),
 - valós, ha valamelyik operandus valós vagy a művelet az osztás (/).



Függvények

■ Matematikai függvények:

- ABS(X) X abszolút értéke.
- EXP(X) Az exponenciális függvény (e^x) értéke az X helyen.
- LOG(X) A természetes alapú logaritmus függvény értéke az X helyen.
- SIN(X) X szinusza (X radiánban adott).
- COS(X) X koszinusza (X radiánban adott).
- SQR(X) X négyzete.
- SQRT(X) X négyzetgyöke.
- RANDOM(X) Egy véletlen egész szám 0-tól X-1-ig (X egész).

■ Konverziós függvények:

- ASC(X) Az X karakter ASCII kódja (pl. ASC('A') \rightarrow 65).
- CHR(X) Az X ASCII kódú karakter (pl. CHR(65) \rightarrow 'A').
- INT(X) Az X valós szám egész része (pl. INT(3.14) \rightarrow 3).



Az értékadó utasítás

- Jelölés:

változó \leftarrow kifejezés

vagy

változó \leftarrow ... \leftarrow változó \leftarrow kifejezés

- Végrehajtás:

- Kiértékelés:

- A kifejezés értékének kiszámítása.

- Tárolás:

- Ha az érték tárolható, akkor tárolódik a változó(k)ban, különben hiba lép fel.

Az értékadó utasítás

- Egy érték akkor tárolható egy adott változóban, ha
 - a típusuk megegyezik,
 - valós típusú változóhoz egész értéket rendelünk,
 - egész típusú változóhoz valós értéket rendelünk,
 - sztring típusú változóhoz karaktert rendelünk.
- Megjegyzés:
 - Az esetleges túlcsordulástól, konverzióktól eltekintünk, feltesszük, hogy a konverziók automatikusan végrehajtnak.
 - Valós értékek egész változókhoz rendelését megengedjük, ekkor a szám egész része konvertálódik, a törtrész figyelmen kívül marad.

Pl. $I \leftarrow I + 1$
 $I \leftarrow J \leftarrow K \leftarrow 0$



A beolvasó utasítás

- Jelölés:

Be: változólista

A változólista változók vesszővel elválasztott sorozata.

Pl. Be: A, B, C

- Végrehajtás:

- A felhasználó által, a billentyűzettel megadott adatok tárolódnak a felsorolt változókbán.

- Megjegyzés:

- A bekérést segítő tájékoztató üzenetek kiírását elhagyjuk (\Rightarrow programmá írás).
- Feltesszük, hogy a felhasználó jó adatot ad meg.
- Csak az egész, valós, karakter és sztring típusú adatok beolvasása megengedett.



A kiíró utasítás

- Jelölés:

Ki: kifejezéslista

A kifejezéslista kifejezések vesszővel elválasztott sorozata.

Pl. Ki: "A kör sugara:", R, "területe:", $R*R*3.14$

- Végrehajtás:

- A kifejezések értéke kiértékelődik, majd sorban kiíródik a képernyőre.

- Megjegyzés:

- A kiírás esetleges pozícionálását, tagolását elhagyjuk (\Rightarrow programmá írás).
- Csak az egész, valós, karakter és sztring típusú adatok kiírása megengedett.



Adatszerkezeti táblázat

- Egy feladat megoldásához szükséges adatokat és a tárolásukra használt változókat egy táblázatban adjuk meg.
 - **Funkció:**
 - A változóba milyen adat kerül, mire használjuk a változót.
 - **Azonosító:**
 - A változó azonosítója (neve), ezzel hivatkozunk az adatra.
 - **Típus:**
 - A változó, és ezen keresztül az adat típusa, tárolási módja.
 - **Jelleg:**
 - A változóban kiindulási (input), végeredmény (output), ill. részeredmény (munka) adatot tárolunk-e.
- **Megjegyzés:**
 - A változók azonosítóinak egyedieknek kell lenniük.
 - Egy változó, a benne tárolt adattól függően többféle jelleggel is rendelkezhet.

Adatszerkezeti táblázat

Pl. Egy rendező algoritmushoz az alábbi adatszerkezeti táblázat definiálható:

Funkció	Azonosító	Típus	Jelleg
A rendezendő elemek	A	Egydimenziós, tetszőleges elemtípusú tömb	I, M, O
A rendezendő elemek száma	N	Egész	I
Két elem cseréjéhez	CS	Az A tömb elemeivel megegyező típusú	M
Segédváltozók	I, J, K	Egész	M

Konstansok, típusok

■ Formalizmus:

- Konstans
 - Azonosító Adat
- Típus
 - Azonosító Típusleírás

■ Példa:

- Konstans
 - SORMAX 10
 - OSZLMAX 20
- Típus
 - ELEM Egész
 - MATRIX Kétdimenziós ELEM tömb[SORMAX, OSZLMAX]