



ELTE | IK

# PROGRAMOZÁS

## Alapok

Horváth Győző



# Bevezetés



# Bevezetés

---

- Tárgy ismertetése
  - Módszeres feladatmegoldás számítógép segítségével
  - Eszközök bonyolultabb feladatok helyes megoldásához
    - Programozási minták
  - Programozási alapok
- Követelmények
  - Folyamatos számonkérésű tárgy
  - Minden információ a tárgy Canvas oldalán található.

# A programkészítés folyamata

---

1. **Specifikálás** (miből?, mit?) → *specifikáció*
2. **Tervezés** (mivel?, hogyan?) → *adat- + algoritmus-leírás*
3. **Kódolás** (a gép hogyan?) → *kód* (reprezentáció + implementáció)
4. **Tesztelés** (hibás-e?) → *hibalista* (diagnózis)
5. **Hibakeresés** (hol a hiba?) → *hibahely, -ok*
6. **Hibajavítás** (hogyan jó?) → *helyes program*
7. **Minőségvizsgálat, hatékonyság** (jobbítható-e?, hogyan?) → *jó program*
8. **Dokumentálás** (hogyan működik, használható?) → *használható program*
9. **Használat, karbantartás** (még mindig jó?) → *időtálló program*

# A specifikáció



# Példa

---

## Feladat:

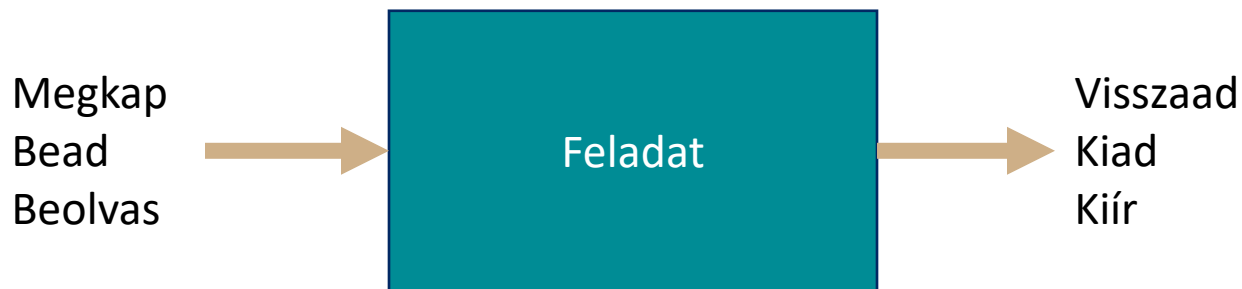
*Egy ötgyerekes nagycsalád nyaralni indul a Balatonra. Az autópályán a 11 éves, a számok iránt mindig nagy érdeklődést mutató Matyi nem látja a kilométerórát, de szeretné megtudni, mekkora sebességgel mennek. Így elkezdi megszámolni hány kilométer táblát hagytak el, és közben az időt is méri. Milyen eredményt kap?*

# Példa

---

## Feladat:

*Az út, idő ismeretében határozd meg a sebességet!*



# Példa adatok

---

## Feladat:

120km; 1óra → **Feladat** → 120km/óra

*Az út, idő ismeretében határozd meg a sebességet!*

## Példa:

120km; 1óra → 120km/óra

12km; 0,1óra → 120km/óra

10,5km; 0,1óra → 105km/óra



# Adatok köre

## Példa:

$$s=120; t=1 \rightarrow v=120$$

$$s=10,5; t=0,1 \rightarrow v=105$$

Megkap:  $s, t$  azonosítójú valós számok  
Visszaad:  $v$  azonosítójú valós szám

- „Megkapni”
  - két számot
  - pontosabban: két valós számot
  - még pontosabban:  $s, t$  azonosítójú valós számokat
- „Visszaadni”
  - egy számot
  - pontosabban: valós számot
  - még pontosabban:  $v$  azonosítójú valós számot

# Megszorítások

Mindent elfogadok?

$s=-1; t=0 \rightarrow ?$

Megkap:  $s, t$  azonosítójú valós számok  
Visszaad:  $v$  azonosítójú valós szám  
Megszorítás:  $s$  nemnegatív,  $t$  pozitív

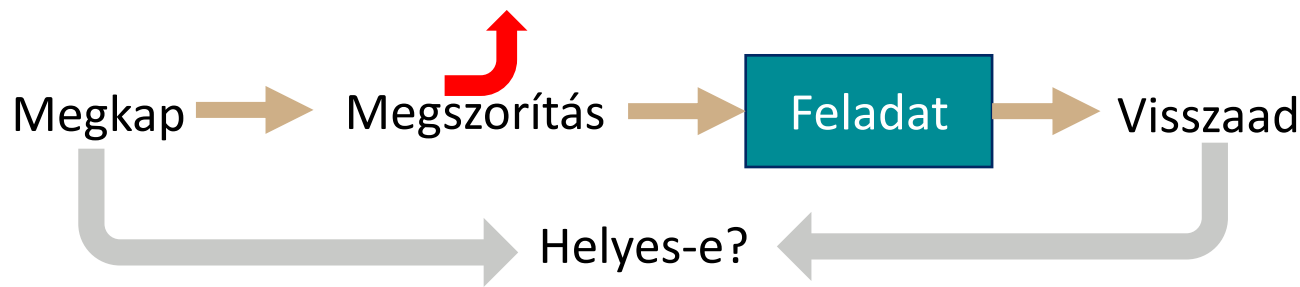
„Megkapni”:

$s$  **nemnegatív** és  $t$  **pozitív** valós számokat.



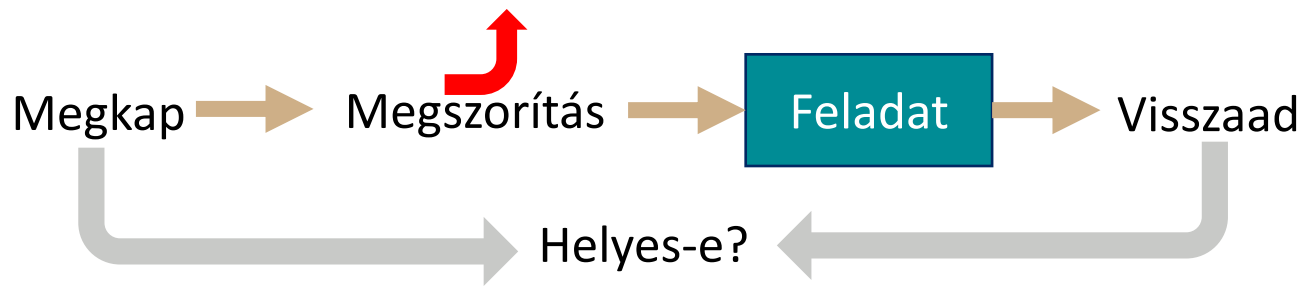
# Helyes-e a feladat?

- Honnan tudjuk, hogy Matyi helyesen számolt?
- Honnan tudjuk, hogy a program **helyesen oldotta meg** a feladatot?
- Definiálni kell az **összefüggést** a kapott és a visszaadott adatok között!
  - $v=s/t$



# Nem formálisan összegezve

- Megkapunk:  $s, t$  valós számokat
- Visszaadunk:  $v$  valós számot
- Megszorítás:  $s$  nemnegatív,  $t$  pozitív
- Megoldás:  $v = s / t$



# Példa: út-idő-sebesség

## Feladat:

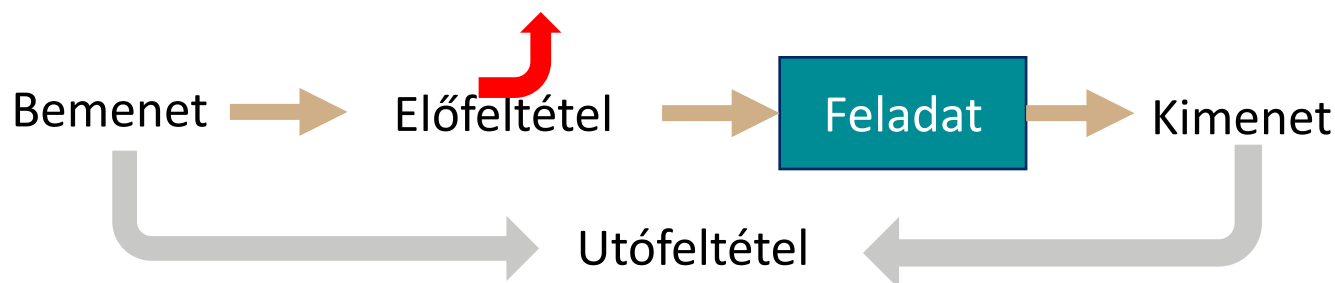
*Az út, idő ismeretében határozd meg a sebességet!*

Bemenet:  $s \in \mathbb{R}$ ,  $t \in \mathbb{R}$

Kimenet:  $v \in \mathbb{R}$

Előfeltétel:  $s \geq 0$  és  $t > 0$

Utófeltétel:  $v = s / t$



# Példa: út-idő-sebesség

## Feladat:

*Az út, idő ismeretében határozd meg a sebességet!*

## Specifikáció:

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$

Ki:  $v \in \mathbb{R}$

Ef:  $s \geq 0$  és  $t > 0$

Uf:  $v = s / t$

$\mathbb{R}$ =Valós számok **halmaza**

# A specifikáció fogalma

---

## Célja:

a feladat formális megragadása  
szerződés a megbízó és a fejlesztő között

## Kérdések:

- Mitől függ a megoldás? – *bemenet*
- Mi a megoldás? – *kimenet*
- Mit jelent: „megoldásnak lenni”? – *utófeltétel*
- Mindig/Mikor *van* megoldás? – *előfeltétel*

# A specifikáció fogalma

---

## Összetevői:

1. Bemenő adatok (azonosító, értékhalmoz [mértékegység])
2. Ismeretek a bemenetről (előfeltétel)
3. Eredmények (azonosító, értékhalmoz)
4. Az eredményt meghatározó logikai állítás (utófeltétel),  
amely a helyesen összetartozó adatokra igaz értéket ad
5. A használt fogalmak definíciói
6. A megoldással szembeni követelmények
7. Korlátozó tényezők



# A specifikáció fogalma

---

## **Tulajdonságai:**

1. „Egyértelmű”, pontos, teljes
2. Tömör ( $\leftarrow$ formalizált)
3. Érthető, szemléletes (fogalmak)

A három szempont sokszor ellentmond egymásnak.

## **Specifikációs eszközök:**

1. Szöveges leírás
2. Matematikai megadás

# Jelölések

Megnevezés	Jelölés, halmaz	Specifikációs eszköz	Példa
Egész szám	$\mathbb{Z}$	Z	...;-2; -1; 0; 1; 2; ...
Természetes szám	$\mathbb{N}$	N	0; 1; 2; 3; ...
Valós szám	$\mathbb{R}$	R	10,234; $\pi$
Logikai érték	$\mathbb{L}$	L	igaz, hamis
Szöveg	$\mathbb{S}$	S	„alma”
Karakter	$\mathbb{C}$ vagy $\mathbb{K}$	C vagy K	„a”

# Példa: út-idő-sebesség

---

## Feladat:

*Az út, idő ismeretében határozd meg a sebességet!*

*Helyes-e az alábbi utófeltétel?*

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$

Ki:  $v \in \mathbb{R}$

Ef:  $s \geq 0$  és  $t > 0$

Uf:  $s = v * t$

# Specifikáció mint függvény

Sebesség:  $\mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$   
Sebesség:  $\mathbb{R}^2 \rightarrow \mathbb{L}$   
Sebesség( $s, t$ ):= $v$  és  $v=s/t$   
Sebesség( $s, t$ ):= $s/t$



- **Bemenet:**  $s \in \mathbb{R}, t \in \mathbb{R}$   
a függvény értelmezési tartománya:  $\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$  (amelynek egyes komponenseire lehet hivatkozni a specifikációban  $s$ -sel,  $t$ -vel)
- **Kimenet:**  $v \in \mathbb{R}$   
a függvény értékkészlete:  $\mathbb{R}$  (amelyre hivatkozhatunk a specifikációban  $v$ -vel)
- **Előfeltétel:**  $s \geq 0$  és  $t > 0$   
a függvény értelmezési tartományának  $(\mathbb{R}^2)$  szűkítése  $(\mathbb{R}_{0,+}^2)$
- **Utófeltétel:**  $v = s/t$   
mi igaz a végeredményre: a „kiszámítási szabály”

# Az algoritmus



# Az algoritmus

---

**Hogyan** oldjuk meg a feladatot?

A megoldás *elemi lépésekre* bontása

Sebesség:

- $s / t \rightarrow v$
- $v \leftarrow s / t$
- $v := s / t$

# Az algoritmus fogalma

---

**Elemi** tevékenységek:

értékadás, beolvasás, kiírás.

Az algoritmusok **összeállítási módjai**:

- Szekvencia (egymás utáni végrehajtás)
- Elágazás (választás 2 vagy több tevékenységből)
- Ciklus (ismétlés adott darabszámszor vagy adott feltételtől függően)
- Alprogram (egy összetett tevékenység, egyedi néven – absztrakció)

# Algoritmusleíró nyelvek - pszeudokód

## Szekvencia:

```
utasítás1  
utasítás2  
utasítás3
```

## Elágazások:

```
Ha feltétel akkor  
    utasítások igaz esetén  
különben  
    utasítások hamis esetén  
Elágazás vége
```

kétirányú

```
Elágazás  
    feltétel1 esetén utasítások1  
    feltétel2 esetén utasítások1  
    ...  
    különben utasítások  
Elágazás vége
```

többirányú

## Ciklusok:

```
Ciklus amíg feltétel  
    utasítások  
Ciklus vége
```

előltesztelő

```
Ciklus  
    utasítások  
amíg feltétel  
Ciklus vége
```

hátteltesztelő

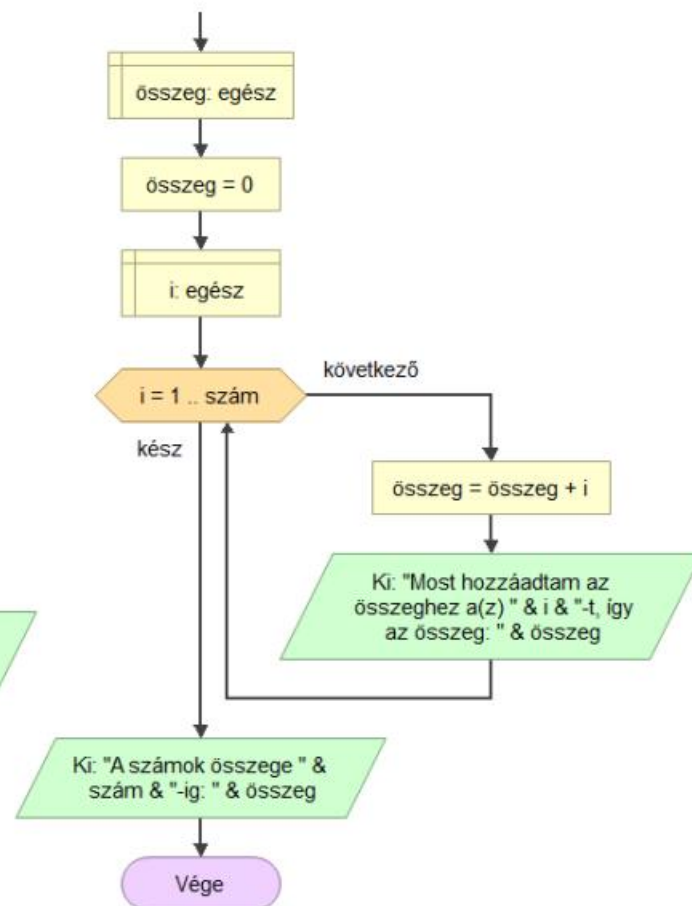
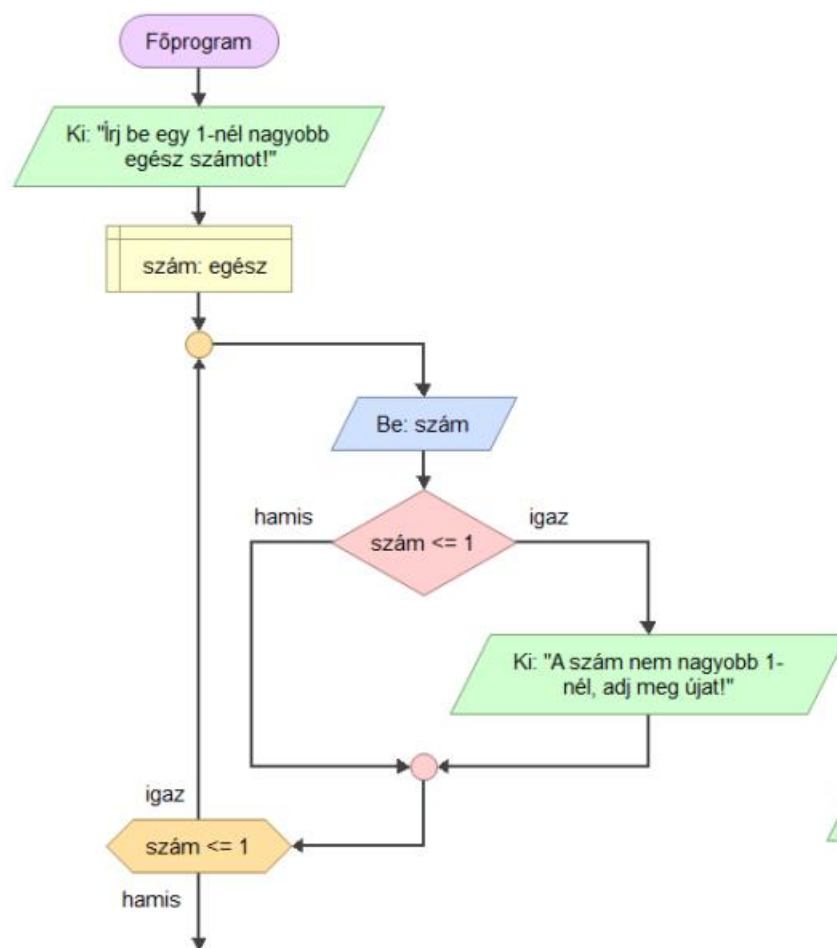
```
Ciklus i=e-től u-ig  
    utasítások  
Ciklus vége
```

számlálós



# Algoritmusleíró nyelvek - folyamatábra

## Szekvencia, elágazás, ciklus



# Algoritmusleíró nyelvek - Struktogram

**Szekvencia:**

utasítás1
utasítás2
utasítás3

**Elágazások:**

feltétel	
<i>true</i>	<i>false</i>
utasítások1	utasítások2
...	...

kétirányú

Feltétel1	Feltétel2	...	egyébként
Utasítások1	Utasítások2	...	Utasítások

többsíriányú

# Algoritmusleíró nyelvek - Struktogram

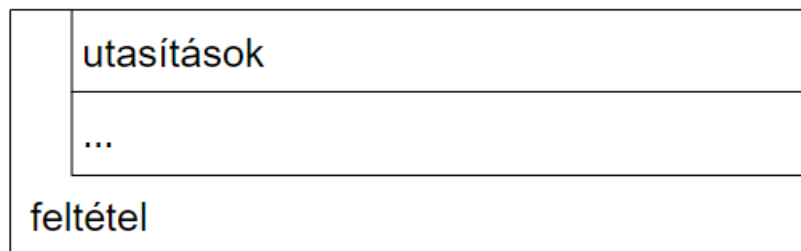
## Ciklusok:



számlálós



előletesztelős



hátulatesztelős

# Algoritmusleíró nyelvek - Struktogram

---

- Struktogramszerkesztés
  - <https://progalap.elte.hu/stuki/>

# Specifikáció és megvalósítás

---

## Specifikáció és megvalósítás:

A feladat specifikációja valós világbeli objektumokhoz rendel valamilyen valós világbeli eredményt. Emiatt valós világbeli dolgokkal (pontosabban azok absztrakciójával, pl. valós számok halmaza) foglalkozik.

A feladat számítógépes megoldása emiatt több részből áll

- a valós világbeli objektumokat leíró adatokat be kell juttatni a számítógépbe, annak memóriájában tárolni kell – ezek lesznek a megoldásbeli változók, amelyek típusa a számítógépes világ által elfogadott/megvalósított típusokból állhat (azaz pl. a számítógépes valós számok halmaza a matematika valós számhalmazának egy véges része lehet) – a specifikációban szereplő neveket (egyelőre) azonos nevű memóriabeli változókkal azonosítjuk;

# Specifikáció és megvalósítás

---

- a memóriában megjelenő változókból valamilyen függvénnyel kiszámítjuk az eredményt, amit szintén a memóriában tároljuk – ezek neve (egyelőre) szintén megegyezik a specifikációban szereplő elnevezésekkel;
- végül az eredményt tartalmazó változók értékeit valahogyan kijuttatjuk a külvilágba.

Megjegyzés: lehetnek olyan változók is (látni fogjuk), amelyek a specifikációban nem jelennek meg.

Ebből alakul ki a klasszikus programok három fő lépése (= 3 algoritmus szekvenciája):

- az adatok beolvasása;
- az eredmény kiszámítása;
- az eredmény kiírása.

# Specifikáció és algoritmus

---

- Specifikáció
  - Adathoz adatot rendel,  $\text{adat} \rightarrow \text{adat}$
  - Például:  $10,5; 0,1 \rightarrow 105$
  - Az adatokra címkéken keresztül hivatkozunk
  - Például:  $s \in R, t \in R \rightarrow v \in R$

# Specifikáció és algoritmus

---

- Algoritmus

- A specifikációbeli címkékhez ugyanolyan nevű **változókat** hozunk létre
  - Például:  $s \in R \rightarrow s$ : Valós
- Az algoritmus végrehajtása előtt a bemeneti változók felveszik a bemeneti adatok értékeit (**beolvasás**)
- Az algoritmus a megoldás során módosíthatja a változók értékét (**feldolgozás**)
- Az algoritmus végrehajtása után a kimeneti változók a kimeneti adatok értékét kell tartalmazzák (**kiírás**)



# Példa: sebesség

## Algoritmus:

A programunk 4 fő részből áll: az adatok **deklarálása**, **beolvasása**, az eredmény **kiszámítása**, az eredmény **kiírása**:

### Specifikáció:

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$   
Ki:  $v \in \mathbb{R}$   
Ef:  $s \geq 0$  és  $t > 0$   
Uf:  $v = s / t$

Változó  
 $s, t: \text{Valós},$   
 $v: \text{Valós}$

Valós: Valós számok típusa

Be: $s, t$	$[s \geq 0 \text{ és } t > 0)$
$v := s/t$	
Ki: $v$	

A **deklarációt**, az „elemi” utasításokat egy-egy „dobozba” írjuk.

# Példa: sebesség

## Algoritmus:

A be- és kimenetet nem algoritmizáljuk!

A specifikációból egyértelműen származtatott deklarációkat nem tüntetjük fel.

### Specifikáció:

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$

Ki:  $v \in \mathbb{R}$

Ef:  $s \geq 0$  és  $t > 0$

Uf:  $v = s / t$

~~Változó  
 $s, t: \text{Valós},$   
 $v: \text{Valós}$~~

$v := s / t$

~~Be:  $s, t [s \geq 0 \text{ és } t > 0)$~~

$v := s / t$

~~Ki:  $v$~~

# Jelölések

Specifikáció halmaz	Algoritmus típus
Z	Egész
N	Egész
R	Valós
L	Logikai
S	Szöveg
C vagy K	Karakter

Például:

$a \in \mathbb{Z}$



Változó  $a$ : **Egész**

A kód



# Kód keret

---

ňăņêşřăĉê şêĉêşşêĝ  
îņțêsňăĺ ĉĺăşş Rsôĝsăņ  
șțățîĉ wôîđ Năîņ șțsîņĝ ắsĝş  
Dêĺĺăşăĉiồ  
  
Bêộĺwắşắş  
  
Gêĺđộĺĝộcắş  
  
Kîisắş

# Kód megoldás

Dél-ázsiai

duclé s t

duclé w

Bé-lwász

Cönölé wst t s

duclé TsýRásse Cönölé RéadLîne üt s

Cönölé wst t t

duclé TsýRásse Cönölé RéadLîne üt t

Géldölöcász

w s t

Kisász

Cönölé wstLîne A sêcsésg . w

# Kód: deklaráció

Specifikáció	Algoritmus	Kód
$\mathbb{Z}$	Egész	sbyte, short, <b>int</b> , long
$\mathbb{N}$	Természetes	byte, ushort, <b>uint</b> , ulong
$\mathbb{R}$	Valós	float, <b>double</b>
$\mathbb{L}$	Logikai	bool
$\mathbb{S}$	Szöveg	string
$\mathbb{C}$	Karakter	char

Változó **s:Valós**



**độ ư ấ ợ**

# Kód: beolvasás, kiírás

- Beolvasás

Cộng số lẻ  $\ddot{w}$  s i t j e     $\mathfrak{s}$

độ u c l e    \mathfrak{T} s y R a s s e    Cộng số lẻ    R e a d L i n e       o u t     $\mathfrak{s}$

- Kiírás

Cộng số lẻ  $\ddot{w}$  s i t j e L i n e    A     $\mathfrak{s e c s s e g}$     .        $\hat{w}$

- Feldolgozás

- Értékadás:    =    operátor

$\hat{w}$      $\mathfrak{s}$      $\mathfrak{t}$



# Összefoglalva – specifikáció

## Feladat:

*Egy ötgyerekes nagycsalád nyaralni indul a Balatonra. Az autópályán a 11 éves, a számok iránt mindig nagy érdeklődést mutató Matyi nem látja a kilométerórát, de szeretné megtudni, mekkora sebességgel mennek. Így elkezdi megszámolni hány kilométer táblát hagytak el, és közben az időt is méri. Milyen eredményt kap?*



## Feladat:

*Az út, idő ismeretében határozd meg a sebességet!*



## Specifikáció:

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$

Ki:  $v \in \mathbb{R}$

Ef:  $s \geq 0$  és  $t > 0$

Uf:  $v = s / t$



## Példa:

$s=120; t=1 \rightarrow v=120$

$s=10,5; t=0,1 \rightarrow v=105$

# Összefoglalva – algoritmus

## Specifikáció:

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$   
Ki:  $v \in \mathbb{R}$   
Ef:  $s \geq 0$  és  $t > 0$   
Uf:  $v = s / t$

Változó  
 $s, t: \text{Valós},$   
 $v: \text{Valós}$

Be:	$s, t$	$[s \geq 0 \text{ és } t > 0)$
$v := s/t$		
Ki:	$v$	

$v := s/t$

# Összefoglalva – kód

## Specifikáció:

Be:  $s \in \mathbb{R}, t \in \mathbb{R}$   
Ki:  $v \in \mathbb{R}$   
Ef:  $s \geq 0$  és  $t > 0$   
Uf:  $v = s / t$

Változó  
 $s, t$ : Valós,  
 $v$ : Valós



```
// Deklaráció
double s, t;
double v;
// Beolvasás
Console.Write("s = ");
double.TryParse(Console.ReadLine(), out s);

Console.Write("t = ");
double.TryParse(Console.ReadLine(), out t);

// Feldolgozás
v = s / t;

// Kiírás
Console.WriteLine("A sebesség: {0}", v);
```

**Be:**  $s, t$  [ $s \geq 0$  és  $t > 0$ ]

$v := s/t$

**Ki:**  $v$

# Példa



# Feladat

---

ÁFA kalkulátor: a bruttó összeg és az ÁFA kulcs ismeretében határozd meg a nettó értéket!

Lépések:

1. Példa adatok
2. Specifikáció
3. Algoritmus
4. Kód

Adatokkal kapcsolatos fogalmak



# Adatokkal kapcsolatos fogalmak

---

- **Konstans**

Az az adat, amely a műveletvégzés során nem változtat<sup>hat</sup>ja meg értékét, mindvégig ugyanabban az „állapotban” marad.

- **Változó**

Az ilyen adatféleségnek lényegéhez tartozik a „változékony-ság”, más szóval: vonatkoz<sup>hat</sup>nak rá olyan műveletek is, amelyek új értékkel látják el.

Tudományosan fogalmazva: végrehajtás során megváltozhat az állapothalmaza.

# Adatokkal kapcsolatos fogalmak

---

- **Változók fajtái céljuk szerint**

- bemeneti változó: bemenetkor kap értéket
- eredmény: kiszámítás tartozik hozzá
- részeredmény: kiszámítás tartozik hozzá, belőle további kiszámítások indulnak
- ... *(lesznek még továbbiak)*



# Adatokkal kapcsolatos fogalmak

---

- **Értékadás**

Az az utasítás, amely révén a pillanatnyi állapotból egy meghatározott állapotba kerül a változó. (Nyilvánvaló, hogy konstans adatra nem vonatkozhat értékadás, az egy, kezdő-értéket meghatározón kívül.)

- **Típus**

Olyan „megállapodás” (absztrakt kategória), amely adatok egy lehetséges körét jelöli ki az által, hogy rögzíti azok állapothalmazát és az elvégezhető műveletek készletét.

# Az adatjellemzők összefoglalása

---

## Azonosító

Az a jelsorozat, amellyel hivatkozhatunk a tartalmára, amely által módosíthatjuk tartalmát.

## Kezdőérték

A születéskor hozzárendelt érték.

Konstansoknál nyilvánvaló, hogy deklarációban kapja; változóknál akár deklarációban, akár futáskor szerez értéket magának.

# A típus

---

Összetettség (strukturáltság) szempontjából beszélhetünk

- strukturálatlan (vagy skalár, elemi) típusról, ha (az adott szinten) szerkezetet nem tulajdonítunk neki; vagy
- strukturált (más szóval: összetett) típusról, ha (elemibb) összetevőkre bontjuk.

# Elemi típusok

## Egész típus

- Értékhalma:  $-2^{31}..+2^{31}-1$   
(Min'Egész..Max'Egész)
- Műveletek:  $+$ ,  $-$ ,  $*$ , Div (egészosztás),  
Mod (osztási maradék),  $-$  (unáris mínusz),  
 $^$  (pozitív egészkitevős hatványozás)
- Relációk:  $=$ ,  $<$ ,  $\leq$ ,  $\neq$ ,  $\geq$ ,  $>$
- Ábrázolás: kettes komplement kódú
- Változatai: méret és előjel szerint sokfélék

Példaként: 4-bájtos  
ábrázolást  
feltételezve.

A beolvasáson, a  
kiíráson és  
értékadáson túliakkal  
foglalkozunk csak.

Pl. 3-bites 2-es komplement kódú egész számok:

$+0=0|00_2$ ,  $+1=0|01_2$ ,  $+2=0|10_2$ ,  $+3=0|11_2$ ,

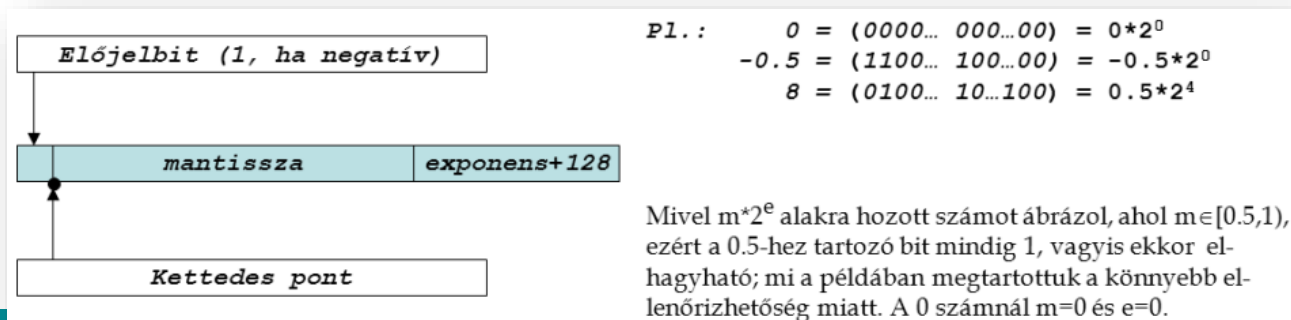
$-1=1|11_2$ ,  $-2=1|10_2$ ,  $-3=1|01_2$ ,  $-4=1|00_2$ ,

Vegye észre a „szabályszerűségeket”!

# Elemi típusok

## Valós típus

- Értékhalmoz: ?????..????  
(Min'Valós..Max'Valós nem definiáltak, vagy reprezentáció-függőek)
- Műveletek: +, -, \*, /, ^, - (unáris mínusz)
- Relációk: =, <, ≤, ≠, ≥, >
- Ábrázolás: lebegőpontos ábrázolás (pontosabb lenne, ha e típust racionálisnak neveznénk, mert csak racionális számot képes ábrázolni)



# Elemi típusok

---

## Logikai típus

- Értékhalmoz: Hamis..Igaz  
(Min'Logikai..Max'Logikai: Hamis, illetve Igaz)
- Műveletek: nem, és, vagy (a szokásos logikai műveletek)
- Relációk: =, <, ≤, ≠, ≥, >
- Ábrázolás: 0B = Hamis, -1B = Igaz  
(esetleg: 1B = Igaz)... ahol xB = x érték „bináris egészként” ábrázolva
- Megjegyzés: a rendezésnek nem nagy a gyakorlati jelentősége.

# Elemi típusok

---

## Karakter típus

- Értékhalma: 0..255 - kódú jelek – ASCII  
(Min'Karakter..Max'Karakter: a 0, illetve a 255 kódú karakter)
- Műveletek: karakter-specifikus nincs  
(esetleg a Kód:Karakter→Egész függvény, és inverze a Karakter:Egész→Karakter függvény, amelyek a belső ábrázolással hozza kapcsolatba)
- Relációk: =, <, ≤, ≠, ≥, >  
(a belső ábrázolásuk alapján → nem ABC-sorrend!)

# Összefoglalás





# Adatok, típusok, változók

Specifikáció	Algoritmus	Kód
$\mathbb{Z}$	Egész	sbyte, short, <b>int</b> , long
$\mathbb{N}$	Természetes	byte, ushort, <b>uint</b> , ulong
$\mathbb{R}$	Valós	float, <b>double</b>
$\mathbb{L}$	Logikai	bool
$\mathbb{S}$	Szöveg	string
$\mathbb{C}$	Karakter	char

Specifikáció	Algoritmus	Kód
Be: $a \in \mathbb{R}$	a: Valós	<b>double</b> $\hat{a}$

# Megfelelések

Algoritmus	Kód
:=	=
=	==
és	&&
vagy	
nem	!
Be: a	Cộng số lẻ Wsítjê ấ độ ấ ấ TsýRắssê Cộng số lẻ RắắđLínê ộtj ấ
Ki: a	Cộng số lẻ WsítjêLínê ấ Cộng số lẻ WsítjêLínê ấ . ấ

# Ellenőrző kérdések



# Kérdések

---

- Milyen lépésekből áll a programkészítés folyamata?
- Mi a specifikáció? Milyen részei vannak? Mi a célja?
- Mi a szerepe a specifikáció egyes részeinek?
- Mi az algoritmus, milyen elemi tevékenységeket tartalmaz?
- Milyen összeállítási módjai vannak az algoritmusnak?
- Hogyan néznek ki a különböző vezérlési szerkezetek struktogrammal írva?
- Hogyan lesz a specifikációból megvalósítás? Hogyan függ össze a specifikáció és az algoritmus?
- Az út, idő ismeretében határozd meg a sebességet! Írd le a feladat specifikációját!
- Számítsuk ki az oldalhosszak ismeretében egy téglalap területét! Írd le a feladat specifikációját!