

---

# A C programozási nyelv I. Bevezetés

Ficsor Lajos

Miskolci Egyetem

Általános Informatikai Tanszék

# A C nyelv története

- Dennis M. Ritchie AT&T Lab., 1972
  - rendszerprogramozás, de **magas szintű nyelven**
- 1973: a UNIX átírása C-re
- 1978: K&R C (Kvázi szabvány)
- Széleskörű használat után:
  - 1983: szabvány előkészítése
  - 1989: ANSI C (X3.159-1989)

A továbbiakban "C nyelv" alatt mindig "ANSI C" értendő

# A C nyelv jellemzői

- általános célú
- viszonylag alacsony szintű
  - alapobjektumok: karakterek, számok, címek
  - nincsenek összetett objektumokat kezelő utasítások
- általában kicsi és hatékony célprogram
- törekvés a hordozhatóságra
- számos implementáció
- széleskörű használat
- gazdag függvénykönyvtár választék

# A C program szokásos szerkezete 1.

direktívák

globális deklarációk és definíciók

main( )

{

lokális deklarációk

utasítások

}

függvény definíciók

# A C program szokásos szerkezete 2.

- A program a main függvény első végrehajtható utasításával indul.
- A minimális (működő) C program:

```
main( )  
  
    {  
  
    }
```

# Szintaktikai egységek 1.

## *Azonosítók*

- Betűk, számjegyek és \_ (aláhúzás)
- Nem kezdődhet számjeggyel
- Kis-nagybetű érzékeny!

## *Kulcsszavak*

- Fenntartott szavak, kisbetűsek.

# Szintaktikai egységek 2.

## *Állandók*

- egész: 123, 0123, 0x123
- karakter: 'a', '\n', '\137'
- lebegőpontos: 1. 1.3, .6, 1.2e12

## *Karakterlánc (string)*

- **"karaktersorozat"**
- Tárolás: a karaktereket tartalmazó byte-sorozat után még egy 0 byte!

# Szintaktikai egységek 3.

## *Operátorok*

- Egy- két- és három operandusú.
- Széles választék

## *Egyéb elválasztók*

- utasításvég: ; (pontosvessző)
- blokkhatárok: { }
- "üres helyek" (space, tab, újsor, megjegyzés)

## *Megjegyzés*

- **/\* Karaktersorozat \*/**
- Nem skatulyázhatók egymásba!





# Változók

## *Négy jellemző:*

- név (ez azonosító kell legyen)
- típus (memóriaméret)
- cím (a memóriában)
- aktuális érték (memóriatartalom)

# Egész (integrális) típusok 1.

Típusnév	Hossz
<code>char</code>	1 byte
<code>int</code>	gépfüggő, a "természetes" hossz (szóhossz)
<code>Short (int)</code>	(legalább) 16 bit
<code>long (int)</code>	(legalább) 32 bit

## Egész (integrális) típusok 2.

### Megjegyzések:

- A nyelv csak azt írja elő, hogy  
 $\text{short} \leq \text{int}$  és  $\text{int} \leq \text{long}$
- Mindegyik elé alkalmazható a
  - **signed** (előjeles) - a **char** kivételével ez a default
  - **unsigned** (előjel nélküli)
- A **char** előjelessége gépfüggő!

# Lebegőpontos típusok

**Típusnév**

**float**

**double**

**long double**

**Hossz**

gépfüggő (ált. 4 byte)

gépfüggő (ált. 8 byte)

gépfüggő (ált. 16 byte)

# Deklaráció (bevezetés)

*Változó:*

típusnév azonosító[, azonosító]

*Tömb:*

típusnév azonosító[elemek száma]

**Az indexelés 0-tól indul!**

# Kifejezések

Operátorok és operandusok váltakozó sorozata

## Operandusok:

- konstans
- változó
- függvényhívás
- kerek zárójelek közé zárt kifejezés

# Operátorok 1.

Csoport	Jel	Asszociativitás
elsődleges	<code>()</code> , <code>[]</code> , <code>-&gt;</code> , <code>.</code>	b-j
egyoperandusú	<code>cast</code> , <code>sizeof</code> , <code>&amp;</code> , <code>*</code> , <code>++</code> , <code>--</code> , <code>~</code> !	j-b
multiplikatív	<code>*</code> , <code>/</code> , <code>%</code>	b-j
additív	<code>+</code> , <code>-</code>	b-j
eltolás	<code>&lt;&lt;</code> , <code>&gt;&gt;</code>	b-j
relációs	<code>&lt;</code> , <code>&lt;=</code> , <code>&gt;</code> , <code>&gt;=</code>	b-j

# Operátorok 2.

Csoport	Jel	Asszociativitás
egyenlőség	==, !=	b-j
AND	&	b-j
XOR	^	b-j
OR		b-j
logikai AND	&&	b-j
logikai OR		b-j



# Operátorok 3.

Csoport	Jel	Asszociativitás
értékadás	$=, +=, -=, /=,$ $\% =, >> =, << =,$ $\& =, \wedge =$	j-b
kif. lista	, (vessző)	b-j

# Operátorok (Megjegyzések) 1.

A kiértékelés sorrendjét a precedencia és az asszociativitás határozza meg.

## Specialitások:

- minden kifejezés pontosvesszővel lezárva utasítássá válik!
- Egészosztás
- cím-operátorok
- ++ és --

# Operátorok (Megjegyzések) 2.

## Specialitások (folytatás):

- "Boolean" típus nincs!
- nincs értékadó utasítás, csak értékadó kifejezés!
- összetett értékadás
- típuskonverziók rendje
- explicit (kényszerített) típuskonverzió (cast):  
**(típusnév) kife**

# Szimbolikus konstans 1.

- Valójában makródefiníciók (részletesebben később)
- Formája:  
**#define azonosító karaktersor**
- A forrásban az azonosító minden magyarázaton kívüli előfordulása kicserélődik karaktersor -ra

## Szimbolikus konstans 2.

- Pl.:

```
#define IGAZ 1
```

```
#define HAMIS 0
```

```
#define BEGIN {
```

```
#define END {
```

- Szokásjog: **nagybetűs név!!!!**
- Vannak előredefiniált konstansok (pl. **EOF**)

# Standard header file-ok (bevezetés) 1.

Alapszabály: C programban csak **deklarált** azonosítók használhatók!

A standard függvények deklarációját standard header file-ok tartalmazzák. A

**#include**

un. direktíva a forrásszövegbe beszúrja a megadott file-t.

# Standard header file-ok (bevezetés) 2.

Pl.:

```
#include <stdio.h>
```

a standard input-output függvények és  
konstansok deklarációinak beillesztésére

```
#include <math.h>
```

matematikai függvények deklarációi

```
#include "fuggvenyeim.h"
```

saját függvények deklarációi

# Egyszerű input-output 1.

A standard inputról és a standard outputra

**Karakter beolvasása:**

- `int getchar(void)` függvény
- Visszatérési érték: a beolvasott karakter kódja

**Karakter kiírása**

- `int putchar (char c)` függvény



# Egyszerű input-output 2.

## Formattált kiírás

```
printf ("formátum", arg1, arg2,  
...)
```

Kiírja az első paraméterének megfelelő formátumban a további paramétereinek az értékét.

# Egyszerű input-output 3.

A formátum-specifikáció tartalma:

- közönséges karakterek
  - változatlanul kinyomtatódnak
- konverzió-specifikáció

A soron következő argumentum nyomtatási formáját határozza meg. Pl.:

# Egyszerű input-output 4.

**%nd**

Egész érték kiírása n karakter széles helyre, balra igazítva

**%d**

Egész érték kiírása a szükséges szélességben

**%s**

Karakterlánc kiírás végig (a \0-t tartalmazó byte-ig)

# Egyszerű input-output 5.

**%ns**

Karakterlanc első n karakterének kiírása, ha kell,  
balra igazítva

**%n.mf**

Lebegőpontos szám fixpontos kiírása n  
szélességben, m tizedesjeggyel

**%n.me**

Lebegőpontos szám kiírása lebegőpontos formában,  
n szélességben, a karakterisztikában m  
tizedesjegyet használva

# Egyszerű input-output 6.

## Formattált beolvasás

**`scanf ( "form. " , címlista )`**

ahol a **"form"** az alábbiakból állhat:

- **%d**          decimális
- **%c**          karakter
- **%s**          string
- **%o**          oktális
- **%x**          hexadecimális
- **%p**          pointer (cím)

## Egyszerű input-output 7.

A címlista lehet pl. változók címei vesszővel elválasztva.

Az egyes beolvasandó értékeket legalább egy "üres" karakter választja el.

```
int a;
```

```
scanf( "%d" , &a ) ;
```

Mindkét függvénynél az első paraméter tartalma határozza meg a további paraméterek számát és típusát -> **HIBA OK!**