

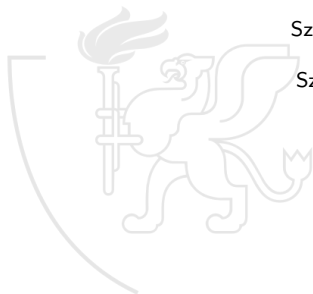
# Programozás Alapjai

Dr. Gergely Tamás  
Dr. Jász Judit

Szegedi Tudományegyetem  
Informatikai Intézet  
Szoftverfejlesztés Tanszék

2021

(v0901)



- 1 **Bemutakozás**
- Kurzus információk
  - A SZTE és az informatikai képzés

- 2 **Linux**
- Alapfogalmak
  - Linux parancsok
  - Linux shell
  - Felhasználók
  - Hálózat

- 3 **Gyors C áttekintés**
- Bevezető
  - Pénzváltás (1. verzió)
  - Pénzváltás (2. verzió)
  - Röppálya számítás
  - Röppálya szimuláció
  - Az év napja
  - Csúszoátlag adott elemszámmra
  - Csúszoátlag parancssorból
  - Basename standard inputról
  - Basename parancssorból
  - Tér legtávolabbi pontjai
  - A nappalis gyakorlat értékelése

- 4 **Alapok**
- Alapfogalmak
  - A programozás fázisai
  - Algoritmus vezérlése
  - A C nyelvű program
  - Szintaxis
  - A C nyelv elemi adattípusai
  - A C nyelv utasításai

- 5 **Vezérlési szerkezetek**
- Bevezetés
  - Szekvenciális vezérlés
  - Függvények
  - Szelekciós vezérlések
  - Ismétléses vezérlések 1.
  - Eljárásvezérlés
  - Ismétléses vezérlések 2.

6 **Folyamatábra és struktúradiagram**

- 7 **Adatszerkezetek**
- Az adatkezelés szintjei
  - Elemi adattípusok
  - Pointer adattípus
  - Tömb adattípus

- Sztringek
- Pointerek és tömbök C-ben
- Rekord adattípus
- Függvény pointer
- Halmaz adattípus
- Flexibilis tömbök
- Láncolt listák
- Típusokról C-ben

- 8 **IO**
- Alapok
  - Adatállományok

- 9 **C fordítás**
- A fordítás folyamata
  - A preprocessor
  - A C fordító
  - Assembler
  - Linker és modulok

- 10 **Gyakorlati kérdések**
- Memóriahasználat
  - Gyakori C hibák
  - where.c felboncolva

# Folyamatábra és struktúradiagram

- Az algoritmus működésének leírása tekintetében a struktúradiagram és a folyamatábra egyenértékű, vagyis
  - amelyik algoritmus szerkezeti ábrával leírható, annak működését folyamatábrával is tudjuk ábrázolni, illetve
  - amelyik algoritmus működését egy folyamatábra megadja, arra készíthető struktúradiagram.



# Folyamatábra és struktúradiagram

## Szerkezeti ábrához folyamatábra

- Mint azt korábban láttuk, minden vezérlési szerkezetnek meg tudtuk adni a folyamatábráját is.
- Ezek után viszont egy szerkezeti ábrát a hierarchia mentén át lehet írni folyamatábrává: amikor a szerkezeti ábrán egy  $P$  problémát valamilyen vezérlés szerint részproblémákra bontunk, akkor a folyamatábrán a  $P$ -hez tartozó egyetlen  $M$  művelet helyére a vezérlési szerkezetnek megfelelő folyamatábrát kötjük be, az  $M$  bemenő éleit a részletező folyamatábra *Start* utáni első, az  $M$  kimenő éleit pedig a *Stop* előtti utolsó elemhez hozzákötve.

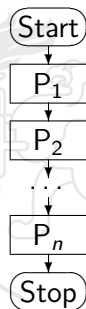


# Folyamatábra és struktúradiagram

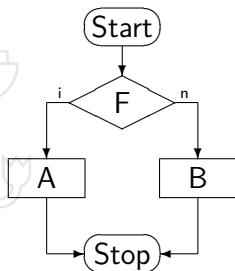
Szerkezeti ábrához folyamatábra

- Egy megjegyzés: mivel egy-egy eljárás szerkezeti ábrája megadható csupán a szekvenciális, egyszerű szelekciós és kezdőfeltételes ismétléses vezérlések használatával (ahogy azt korábban láttuk), elegendő lenne ehhez a háromhoz megadni a megfelelő folyamatábrát:

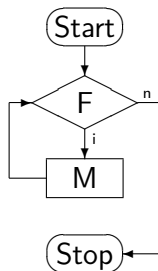
Szekvenciális



Szelekciós



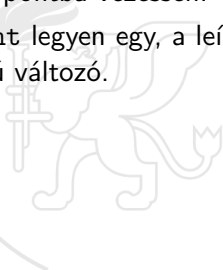
Ismétléses



# Folyamatábra és struktúradiagram

## Folyamatábrához szerkezeti ábra

- Megmutatjuk, hogy fordítva is igaz, tehát a folyamatábrával leírt algoritmusok megadhatók a megismert vezérlési módokat használva, vagyis készíthető hozzájuk szerkezeti ábra.
- Legyen  $G$  egy folyamatábra  $(M, F)$  felett, amely pontjainak száma  $n$ . Sorszámozzuk meg a gráf pontjait úgy, hogy a *Start* pont sorszáma  $0$ , a *Stop* pont sorszáma  $n$  legyen legyen, és a *Start* pontból kivezető él az  $1$ . pontba vezessen.
- A pont legyen egy, a leírt algoritmusban nem használt új, egész típusú változó.



# Folyamatábra és struktúradiagram

Folyamatábrához szerkezeti ábra

- Tekintsük az alábbi C programot:

```
{  
    int pont = 0;  
    while (pont != n) {  
        switch (pont) {  
            case 0 : pont = 1; break;  
            case 1 : U1; break;  
            ...  
            case n-1: Un-1; break;  
            case n : /* Stop */ break;  
        } /* switch */  
    } /* while */  
}
```

# Folyamatábra és struktúradiagram

Folyamatábrához szerkezeti ábra

- Az  $U_i$  utasítás:

- Ha az  $i$ . pontban az  $M_i$  művelet volt és a belőle kiinduló él a  $j$ . pontba vezetett:

```
{  $M_i$ ; pont =  $j$ ; }
```

- Ha az  $i$ . pontban az  $F_i$  feltétel volt és az igennel címkézett él a  $j$ ., a nemmel címkézett él pedig a  $k$ . pontba vezetett:

```
if ( $F_i$ ) { pont =  $j$ ; } else { pont =  $k$ ; }
```

- Az így megalkotott program a  $G$  folyamatábrával adott algoritmussal ekvivalens algoritmus kódolása lesz, és felírható rá egy szerkezeti ábra.