Algoritmusok

- **Az algoritmus fogalma:**

  Az algoritmus egy bizonyos feladattípus megoldására szolgáló lépések (utasítások, előírások) véges sorozata, amely alapján a feladat véges lépésben megoldható.

- **Az algoritmusokkal szemben támasztott követelmények:**

  1. *Végesség*: a feladat megoldására szolgáló lépések számának és minden egyes lépésnek is végesnek kell lennie

  2. *Meghatározottság*: Az algoritmus minden lépésének pontosan definiáltnak, egyértelműnek, félreérthetetlennek kell lennie

  3. *Elvégezhetőség*: Az algoritmus minden lépésének elvégezhetőnek kell lennie

- **Néhány algoritmus leíró módszer:**

  Mondatszerű, folyamatábra, struktogram, struktúra diagram

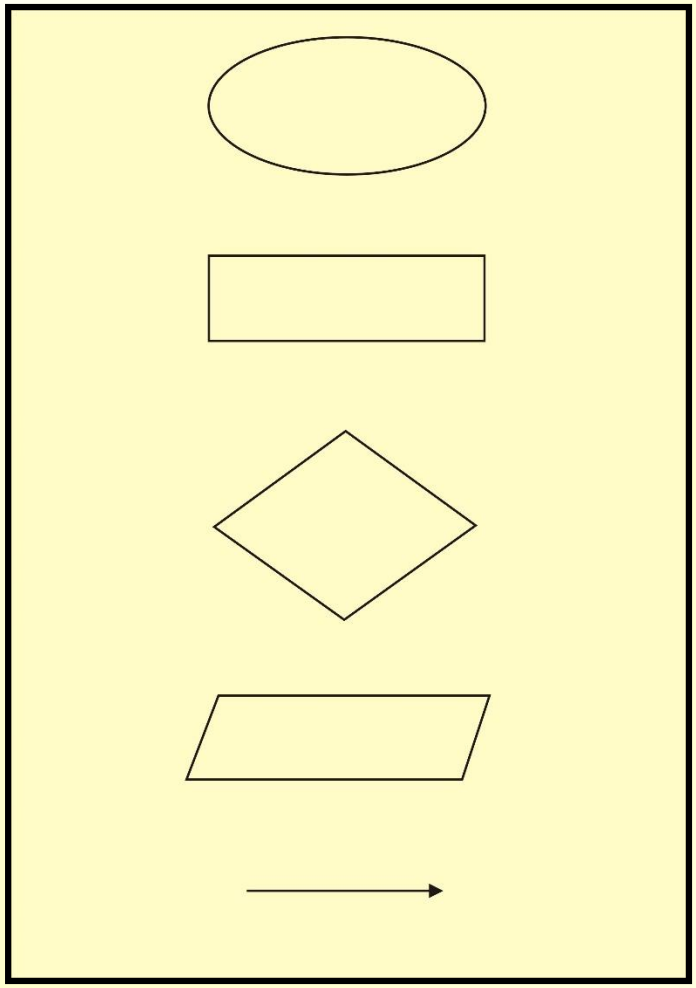
Mondatszerű leírás

- **Az algoritmusok megfogalmazásának első és legtermészetesebb módja a természetes emberi beszéd.**
- **Ennél a módszernél a lépések (utasítások, előírások) megfogalmazására betűket, szavakat, mondatokat használunk.**
- **Nagy hátránya, hogy nem egyértelmű. Ezért szükség van ennél pontosabb, formális leírásra.**

Például:

1. Elmentem moziba
2. Vettem pattogatott kukoricát
3. Ha maradt még pénzem vettem üdítőt
5. Megnéztem a filmet
6. Hazamentem a moziból

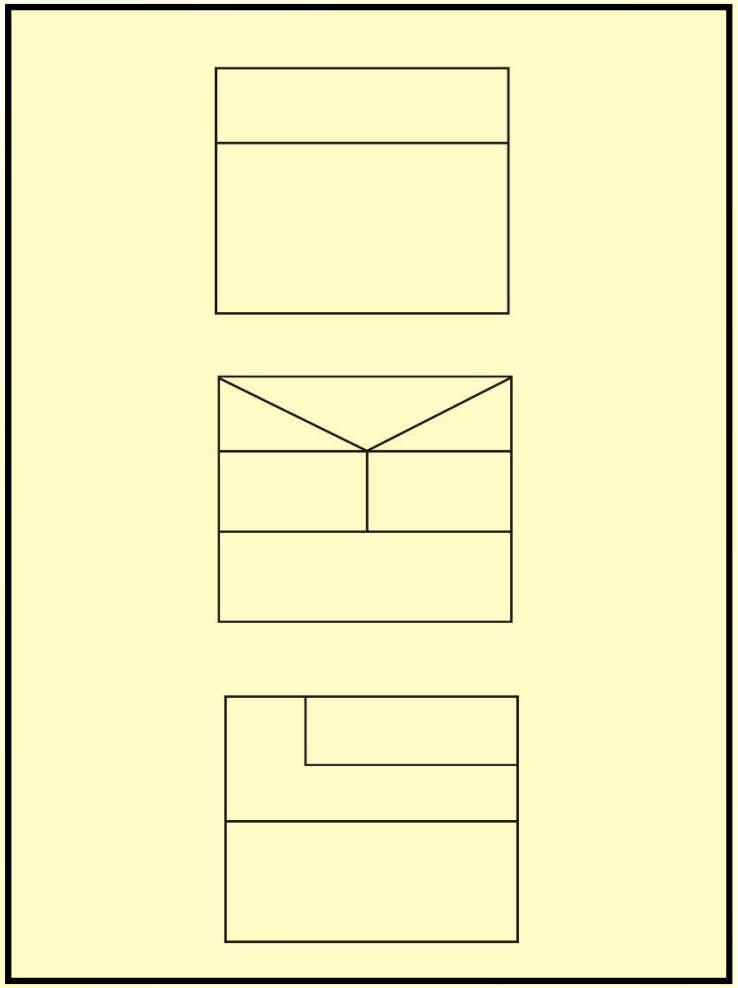
A folyamatábra jelei



A tevékenységeknek síkidomokat feleltetünk meg

- ***Ellipszis*: a folyamatábra indulási és befejezési pontja**
- ***Téglalap*: elemi tevékenységek**
- ***Rombusz*: elágazás, választás**
- ***Paralelogramma*: input és output tevékenységek**
- ***Nyilak*: jelzik a haladás irányát**

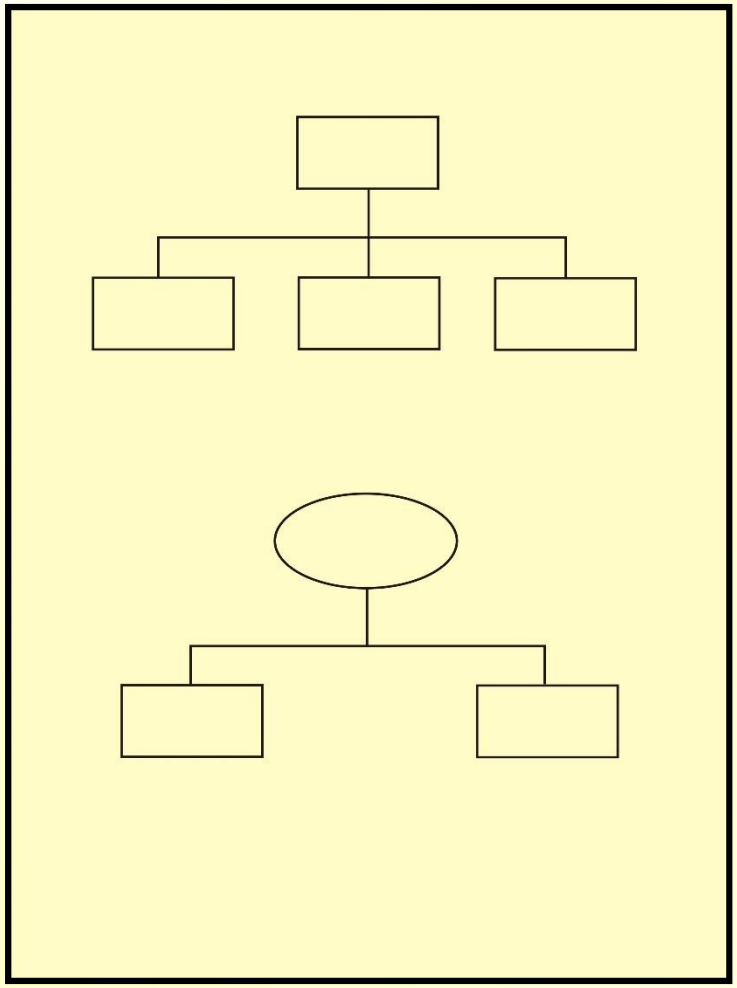
A struktogram jelei



Az egész algoritmusnak egy nagy téglalapot feleltetünk meg, amelyet tovább osztunk

- ***Téglala*p: elemi tevékenységek**
- ***Háromszögek és téglalapok*: elágazás, választás**
- ***L alakzat és téglalap*: iteráció, ismétlés, ciklus**

Struktúra diagram jelei



- **A struktúra diagram két fajta síkidomot feleltet meg a tevékenységeknek amelyek további síkidomokra bomlanak szét**
- ***Téglalap*: elemi tevékenységek**
- ***Ellipszis*: iteráció, ismétlés, ciklus**
- ***Vonalak*: a tevékenységeket a hozzájuk tartozó résztevékenységekkel kötik össze**

Az algoritmusok építőelemei

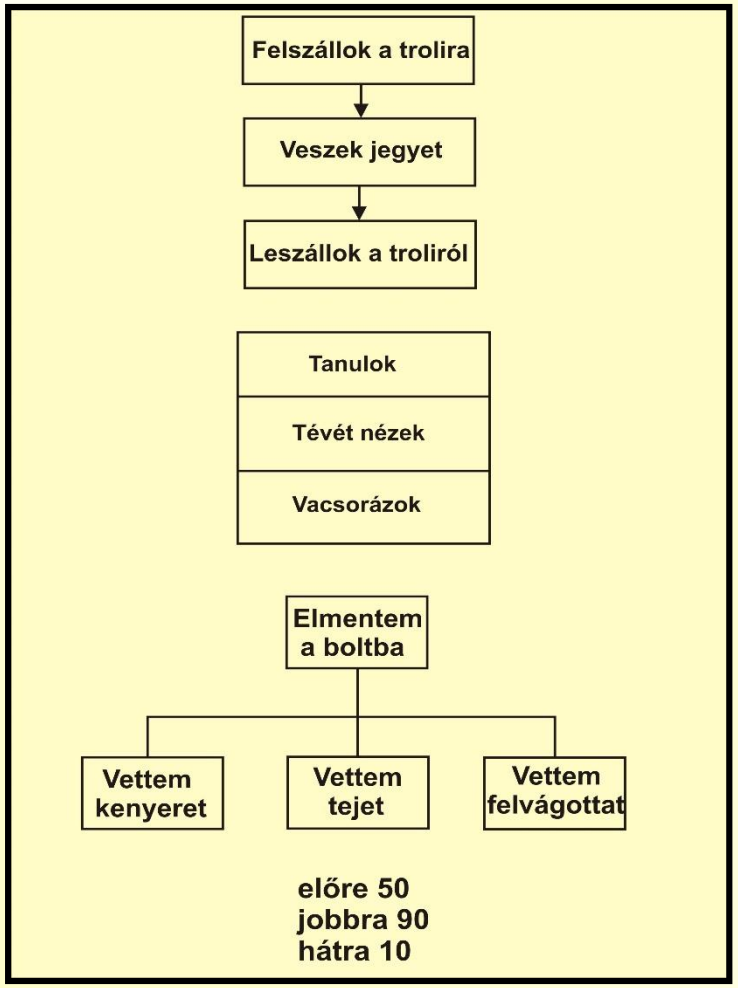
- **Minden algoritmus 3 alapvető szerkezeti elemből építhető fel:**

  1. *Szekvencia*: egymás után végrehajtandó tevékenységek sorozata

  2. *Szelekció (választás, elágazás)*: lépések, tevékenységek közötti választás

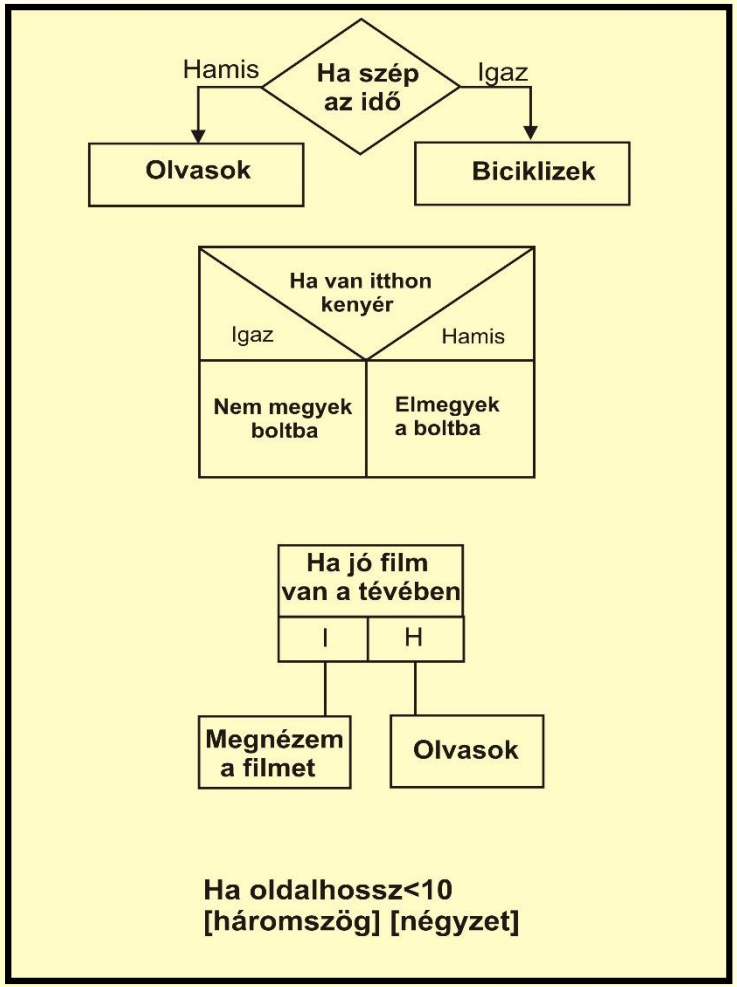
  3. *Iteráció (ismétlés, ciklus)*: valamely tevékenység sorozat ismételt végrehajtása

Szekvencia



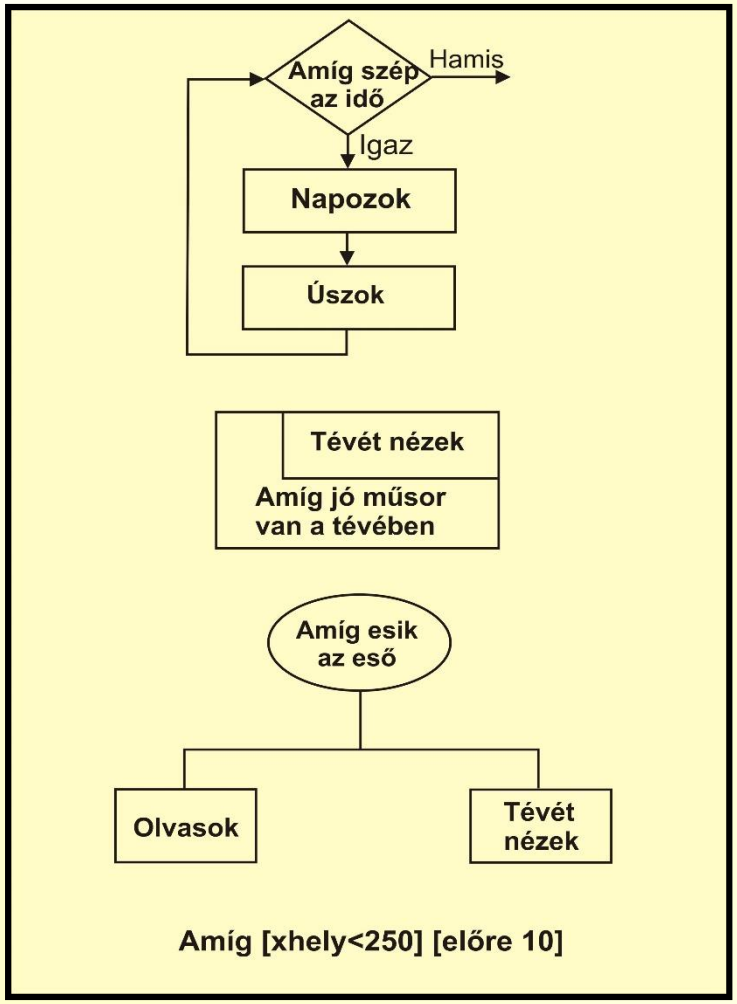
- **Folyamatábrával**
- **Struktogrammal**
- **Struktúra diagrammal**
- **Comenius Logoban**

Szelekció



- **Folyamatábrával**
- **Struktogrammal**
- **Struktúra diagrammal**
- **Comenius Logoban**

Iteráció



- **Folyamatábrával**
- **Struktogrammal**
- **Struktúra diagrammal**
- **Comenius Logoban**

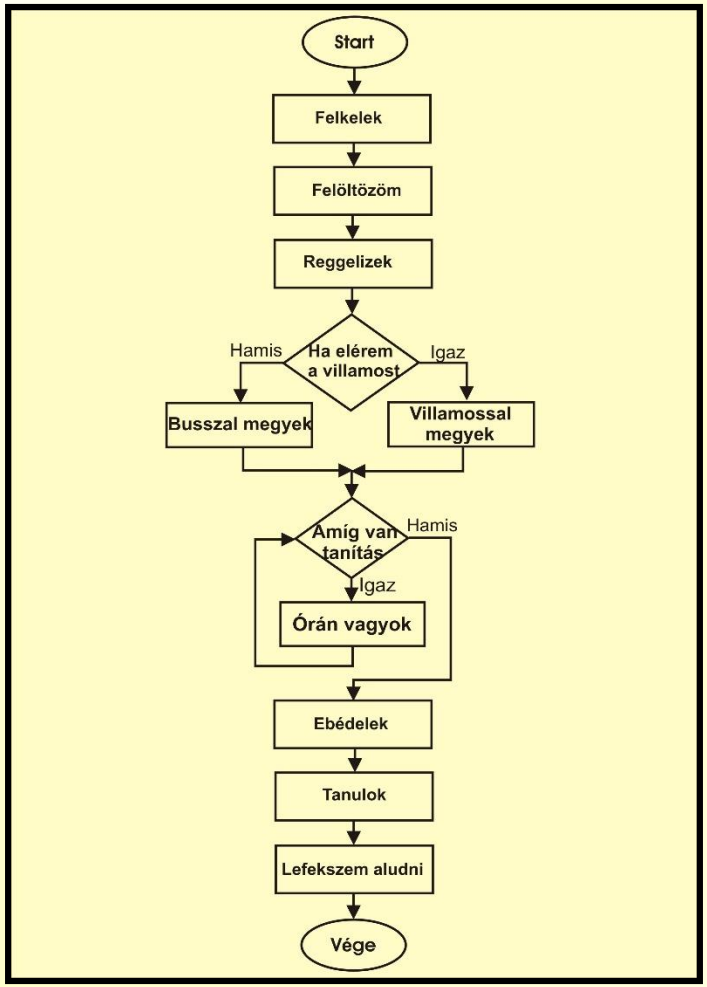
Egy napunk

1. Felkelek
2. Felöltözöm
3. Reggelizek
4. Ha elérem a villamost
   Villamossal megyek
   Ha nem érem el
   Busszal megyek
5. Amíg van tanítás
   Órán vagyok
6. Ebédelek
7. Tanulok
8. Lefekszem aludni

Egy napunk algoritmusa mondatszerűen leírva

- **Szekvencia**
- **Szelekció**
- **Iteráció**
- **Szekvencia**

Egy napunk



Egy napunk algoritmusa folyamatábrával leírva

- **Szekvencia**
- **Szelekció**
- **Iteráció**
- **Szekvencia**

Egy napunk

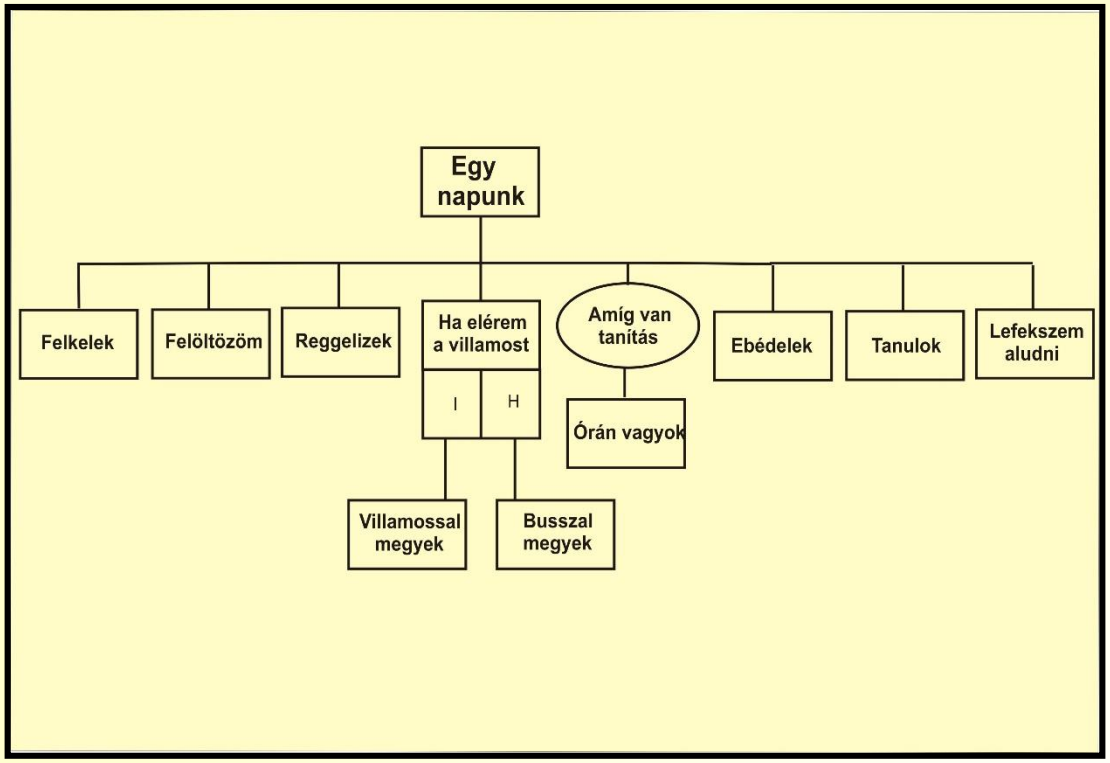
Egy napunk algoritmusa struktogrammal leírva

- **Szekvencia**
- **Szelekció**
- **Iteráció**
- **Szekvencia**

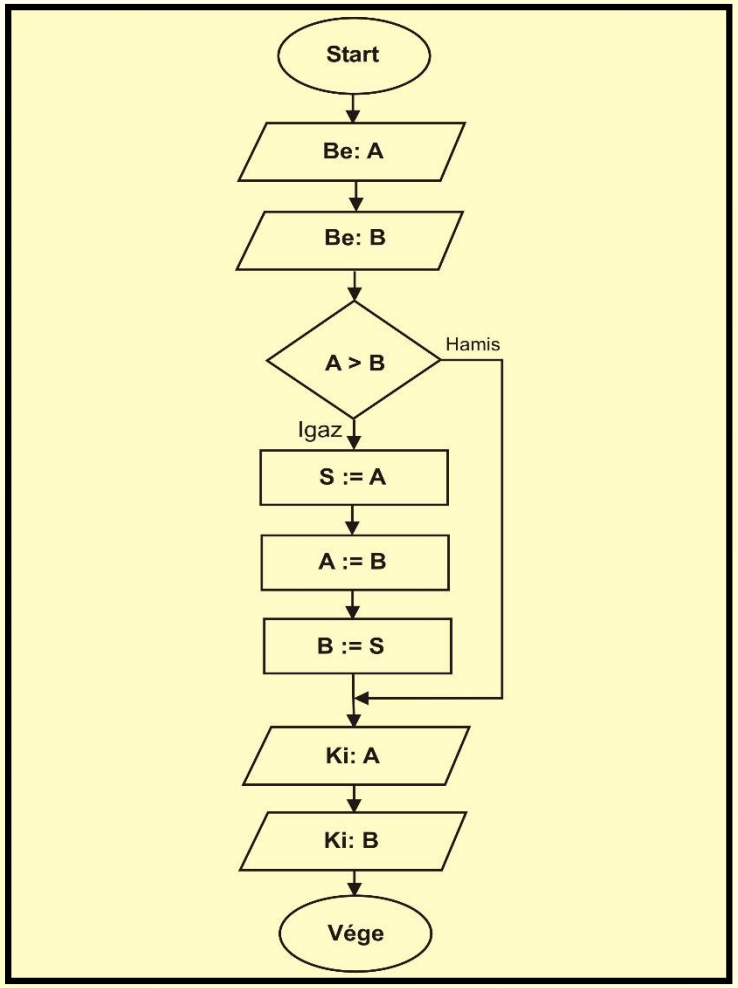
Egy napunk

Egy napunk algoritmusa struktúra diagrammal leírva

Szekvencia Szelekció Iteráció Szekvencia

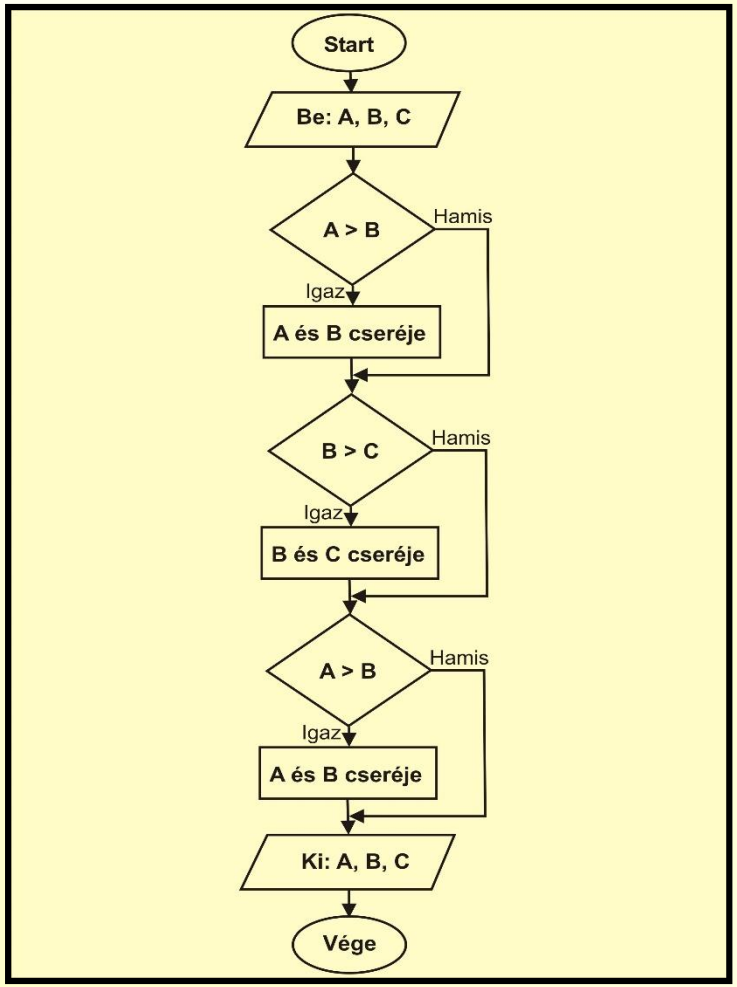


Két elem rendezése



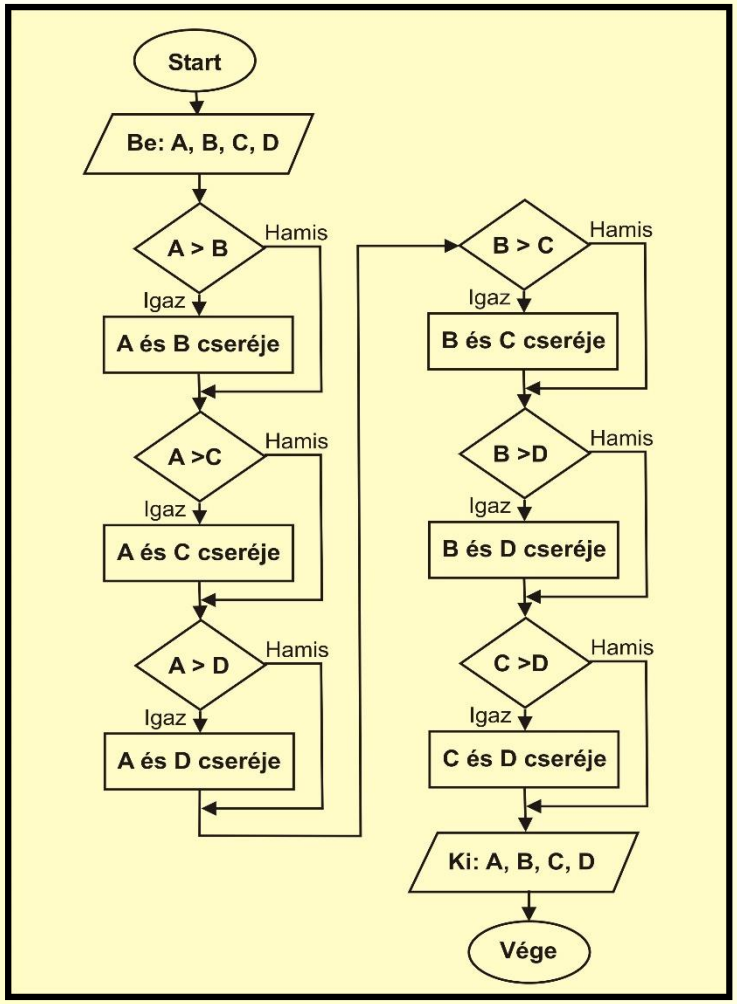
- **Két elemet rendezünk növekvő sorrendbe**
  - Bekérjük a két elemet
- Megvizsgáljuk, hogy A nagyobb-e mint B
- Ha nagyobb, akkor felcseréljük a két elemet egy segédváltozó segítségével
- Ha nem nagyobb, akkor nem kell csinálni semmit az elemek már növekvő sorrendben vannak
  - Kiírjuk a két elemet

Három elem rendezése



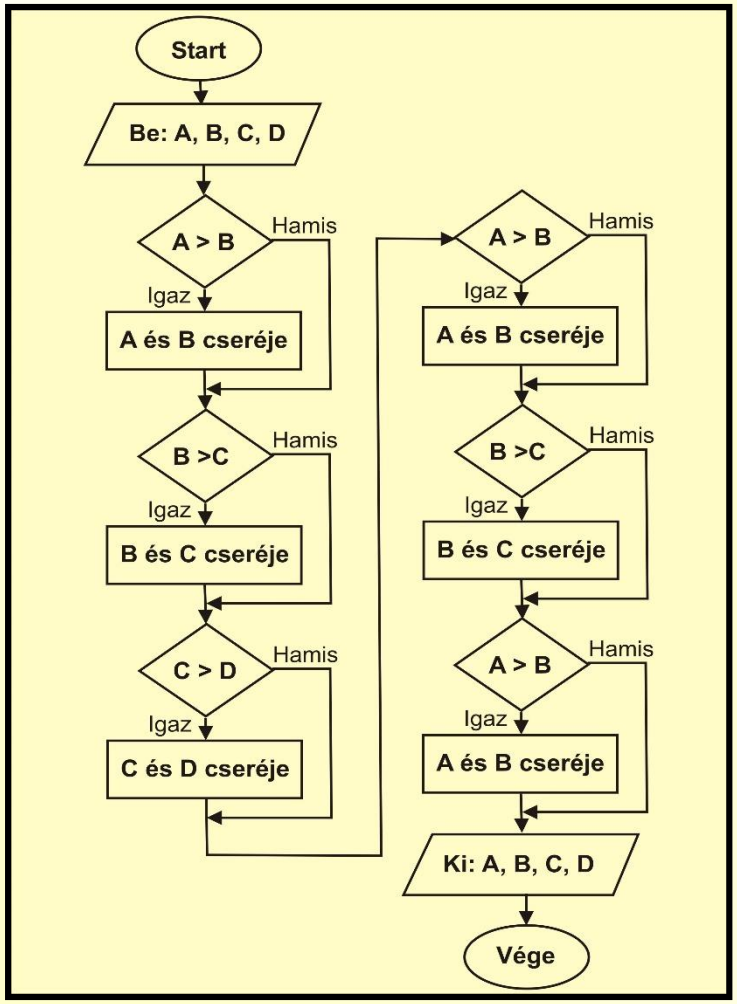
- **Három elemet rendezünk növekvő sorrendbe**
  - Bekérjük a három elemet
  - Ha A nagyobb mint B, akkor felcseréljük a két elemet
  - Ha B nagyobb mint C, akkor felcseréljük a két elemet
  - Ha A nagyobb mint B, akkor felcseréljük a két elemet
  - Kiírjuk a három elemet

Minimum elvű rendezés



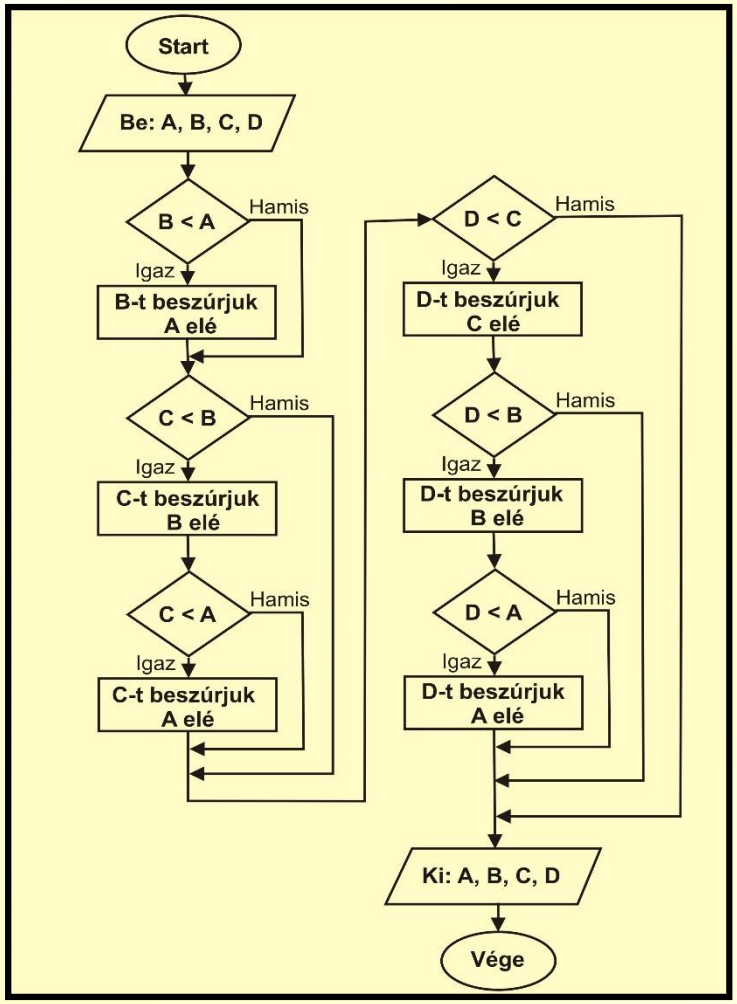
- **A minimum elvű rendezésnél a rendezetlen elemek közül a legkisebbet rakjuk be a már rendezett elemek után**
  - Első lépésben
    A elemet hasonlítom össze a mögötte álló elemekkel, ha bármelyik elem kisebb nála, felcserélem őket
- Ugyanígy kell eljárni az összes többi elem esetében kivéve az utolsó D elemet, mert mire hozzá érünk már rendezett lesz, vele nem kell tenni semmit

Buborék rendezés



- **Buborék rendezésnél a kisebb elemek a sor eleje felé, a nagyobbak pedig a sor vége felé haladnak**
  - Ennél a rendezésnél mindig a szomszédos elemeket hasonlítjuk össze, és ha kell felcseréljük őket
  - Az első futtásnál a sor összes elemén végigmegyünk és a legnagyobb elem a sor végére kerül
  - A második futásnál a második legnagyobb elem kerül a helyére
  - Így haladva tovább az algoritmus végén a sor rendezetté válik

Beszúró rendezés



- **A beszúró rendezésnél az elemeket egy már rendezett sorba szúrjuk be**
- Feltételezzük, hogy az elemünktől balra található elemek már rendezett sort alkotnak
- Megkeressük az elemünk helyét ebben a rendezett sorban
- Akkor találtuk meg a helyét a sorban, ha tőle balra nála kisebb elem helyezkedik el
- Az első elemet sohasem vizsgáljuk meg, mert az már magában rendezett, első lépésben mindig a második elemet hasonlítjuk össze az elsővel