Programozás
1. előadás

Tartalom

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?) → *évelő (időtálló) program*

A specifikáció fogalma

„Interface” a megbízó és a fejlesztő között

Célja:

a feladat formális megragadása.

Összetevői:

1. Bemenő adatok (azonosító, értékhalmaz [mértékegység])
2. Ismeretek a bemenetről (előfeltétel)
3. Eredmények (azonosító, értékhalmaz)
4. Az eredményt meghatározó állítás (utófeltétel)
5. A használt fogalmak definíciói

6. A megoldással szembeni követelmények
7. Korlátozó tényezők

A specifikáció fogalma

Tulajdonságai:

1. „Egyértelmű”, pontos, teljes
2. Rövid, tömör; formalizált
3. Szemléletes, érthető (fogalmak)

Specifikációs eszközök:

1. Szöveges leírás
2. Matematikai megadás

Az algoritmus fogalma

Az algoritmusok **összeállítási mód**jai:

- 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)

Példa: háromszög
(specifikáció)

Feladat:

3 szám lehet-e egy derékszögű háromszög 3 oldala?

Specifikáció:

- **Bemenet:** $x,y,z \in \mathbb{R}$
- **Kimenet:** lehet$\in \mathbb{L}$
- **Előfeltétel:** $x>0$ **és** $y>0$ **és** $z>0$
- **Utófeltétel:** lehet$=(x^2+y^2=z^2)$

$\mathbb{R}$=Valós számok **halmaza**

$\mathbb{L}$=Logikai értékek **halmaza**

Megjegyzés: a 3 szám sorrendjét ezek szerint implicite rögzítettük – z az átfogó hossza!

Példa: háromszög
(specifikáció)

Feladat:

3 szám lehet-e egy derékszögű háromszög 3 oldala?

Specifikáció$_2$:

- **Bemenet:** $x,y,z \in \mathbb{R}$
- **Kimenet:** $lehet \in \mathbb{L}$
- **Előfeltétel:** $0 < x \leq y \leq z$
- **Utófeltétel:** $lehet = (x^2+y^2=z^2)$

Megjegyzés: a 3 szám sorrendjét ezek szerint explicite rögzítettük – z az átfogó hossza!

Példa: háromszög
(specifikáció)

Specifikáció = függvény:

Független változók

- **Bemenet**: $x,y,z \in \mathbb{R}$
  a függvény értelmezési tartománya: $\mathbb{R}\times\mathbb{R}\times\mathbb{R}=\mathbb{R}^3$ (amelynek egyes komponenseire lehet hivatkozni a specifikációban x-szel, y-nal, z-vel)

Függő változó

- **Kimenet**: lehet$\in\mathbb{L}$
  a függvény értékkészlete: $\mathbb{L}$ (amelyre hivatkozhatunk a specifikációban lehet-tel)
- **Előfeltétel**: $x>0$ **és** $y>0$ **és** $z>0$
  a függvény értelmezési tartományának ($\mathbb{R}^3$) szűkítése ($\mathbb{R}_+^3$)
- **Utófeltétel**: lehet$=(x^2+y^2=z^2)$
  mi igaz a végeredményre: a „kiszámítási szabály”

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:

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

Példa: háromszög
(algoritmus)

Valós: Valós számok **típus**a
Logikai: Logikai értékek **típus**a

Algoritmus:

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

Specifikáció:
- **Bemenet**: x,y,z∈R
- **Kimenet**: lehet∈L
- **Előfeltétel**: x>0 és y>0 és z>0
- **Utófeltétel**: lehet=($x^2+y^2=z^2$)

Változó
x,y,z:**Valós**
lehet:**Logikai**

Be: x,y,z [x>0 és y>0 és z>0]

lehet:=($x^2+y^2=z^2$)

Ki: lehet

A deklarációt, az „elemi” utasításokat egy-egy „dobozba” írjuk. Később a be- és kimenetet nem algoritmizáljuk!

Példa: háromszög
(algoritmus)

Egy másik **algoritmus** a lényegi részre:

Segéd változók deklarálása „széljegyzetként”

Változó
xx,yy,zz:**Valós**

$xx:=x^2$
$yy:=y^2$
$zz:=z^2$
$lehet:=(xx+yy=zz)$

Bevezethetők/-endők segéd (belső, saját) változók.

Példa: másodfokú egyenlet
(specifikáció)

Feladat:

Adjuk meg a másodfokú egyenlet egy megoldását!

Az egyenlet: $ax^2+bx+c=0$.

Kérdések:

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

Példa: másodfokú egyenlet
(specifikáció)

Feladat:

*Adjuk meg a másodfokú egyenlet egy **megoldását**! Az egyenlet:* $ax^2+bx+c=0$

Kérdések:

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

Specifikáció$_1$:

- Bemenet: $a,b,c \in \mathbb{R}$
- Kimenet: $x \in \mathbb{R}$
- Előfeltétel: –
- Utófeltétel$_1$: $ax^2+bx+c=0$

Megjegyzés: az uf. nem ad algoritmizálható információt.
Nem baj, sőt tipikus, de … próbálkozzunk még!

Megoldóképlet:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4*a*c}}{2*a}$$

Példa: másodfokú egyenlet
(specifikáció)

Specifikáció$_2$:

- Bemenet: $a,b,c \in \mathbb{R}$
- Kimenet: $x \in \mathbb{R}$
- Előfeltétel: $a \neq 0$
- Utófeltétel$_2$: $x = \frac{-b + \sqrt{b^2 - 4 * a * c}}{2 * a}$

Nyitott kérdések:

- *Mindig/Mikor van megoldás?*
- *Egy megoldás van?*

Példa: másodfokú egyenlet
(specifikáció)

Kimenet bővítés:

- Kimenet: $x \in \mathbb{R}$, $van \in \mathbb{L}$
- Utófeltétel: $van = (b^2 - 4*a*c \geq 0)$ és

A „feladat-függvény” értékkészlete: $\mathbb{R} \times \mathbb{L}$

$$van \rightarrow x = \frac{-b + \sqrt{b^2 - 4*a*c}}{2*a}$$

Nyitott kérdés:

- *Egy megoldás van? – hf.*

Példa: másodfokú egyenlet (algoritmus)

Algoritmus:

Specifikáció$_2$:

- Bemenet: a,b,c∈ℝ
- Előfeltétel: a≠0
- Kimenet: x∈ℝ, van∈𝕃
- Utófeltétel: van=(b²–4*a*c≥0) és

$$van \rightarrow x = \frac{-b + \sqrt{b^2 - 4*a*c}}{2*a}$$

Változó
a,b,c,x:**Valós**
van:**Logikai**
d:**Valós**

Be: a,b,c [a≠0]

d:=b²–4*a*c

van:=d≥0

van?

I

N

Igaz-ág

$$x := \frac{-b+\sqrt{d}}{2*a}$$

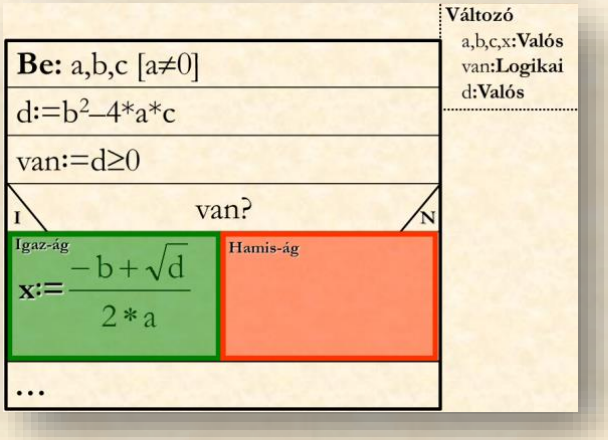
Hamis-ág

A **feltételes** utasítás „3-dobozos” struktúra.

Példa: másodfokú egyenlet (algoritmus)

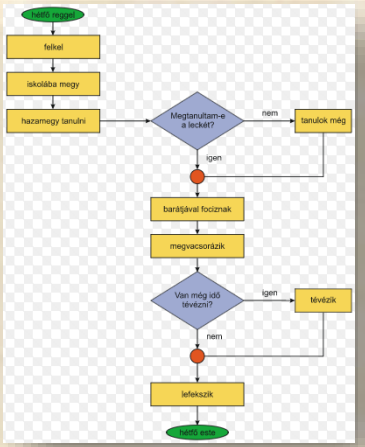
Algoritmus másképpen:

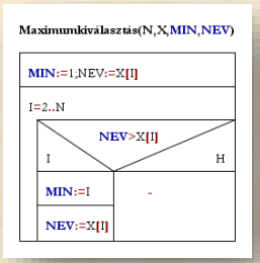
```
Program MásodfokúEgyenlet:
  Változó
     a,b,c,x:Valós
     van:Logikai
     d:Valós
  Be:a,b,c [a≠0]
  d:=b²-4*a*c
  van:=d≥0
  Ha van akkor
  …
Program vége.
```



Algoritmusleíró nyelvek

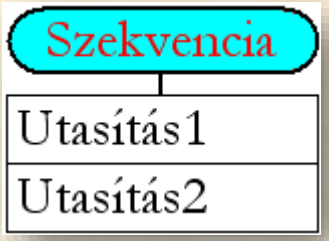
- Szöveges leírás
  - Mondatokkal leírás
  - Mondatszerű elemekkel – **`pszeudokód`**
- **Rajzos leírás**
  - Folyamatábra
  - **Struktogram**





Struktogram
(és pszeudokód)

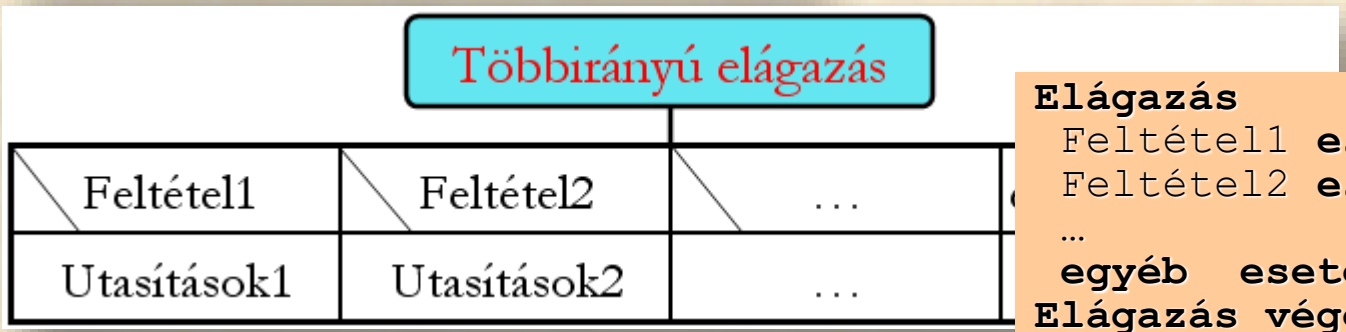
- **Szekvencia:**




```
Utasítás1
Utasítás2
```

- **Elágazások:**

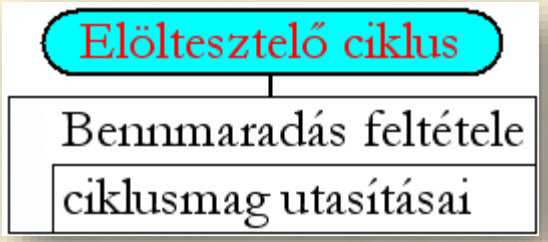

```
Ha Feltétel akkor
   Igaz-ág utasításai
különben
   Hamis-ág utasításai
Elágazás vége
```

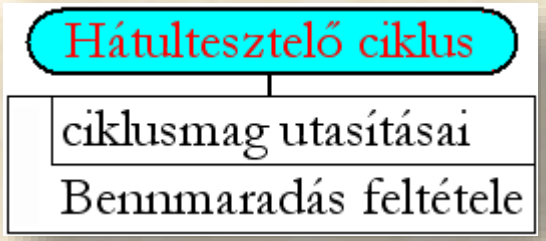



```
Elágazás
  Feltétel1 esetén Utasítások1
  Feltétel2 esetén Utasítások2
  …                …
  egyéb esetekben Utasítások
Elágazás vége
```

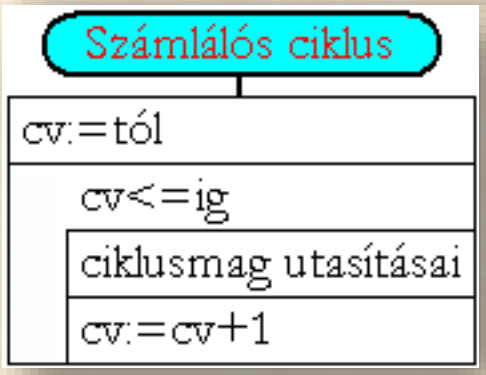
Struktogram
(és pszeudokód)

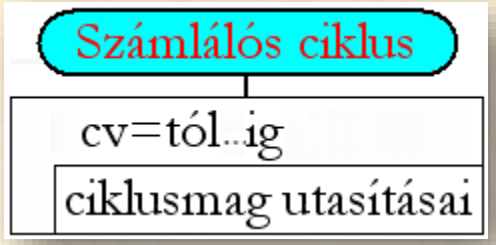
- **Ciklusok:**






```
Ciklus amíg Feltétel
  ciklusmag utasításai
Ciklus vége
```






```
Ciklus
  ciklusmag utasításai
amíg Feltétel
Ciklus vége
```

```
Ciklus cv=tól-tól ig-ig
  ciklusmag utasításai
Ciklus vége
```

- Struktogramszerkesztés:
  - Táblázatkezelővel/szövegszerkesztővel
  - Célprogramokkal

Adatokkal kapcsolatos fogalmak

- **Konstans**

  Az az adat, amely a műveletvégzés során nem változtathatja meg értékét, mindvégig ugyanabban az „állapotban” marad.

- **Változó**

  Az ilyen adatféleségnek lényegéhez tartozik a „változékonyság”, más szóval: vonatkozhatnak rá olyan műveletek is, amelyek új érték-kel látják el.
  Tudományosabban fogalmazva: nem egyelemű 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

- Értékhalmaz: $-2^{31}..+2^{31}-1$
  (Min'Egész..Max'Egész)
- Műveletek: +, – , *, Div (egészosztás) ,
  Mod (osztási maradék), – (unáris mínusz),
  ^ (pozitív egészkitevős hatványozás)
- Relációk: =, <, ≤, ≠, ≥, >
- Ábrázolás: kettes komplemens kódú
- Változatai: méret és előjel szerint sokfélék

Példaként: 4-bájtos ábrázolást feltételezve.

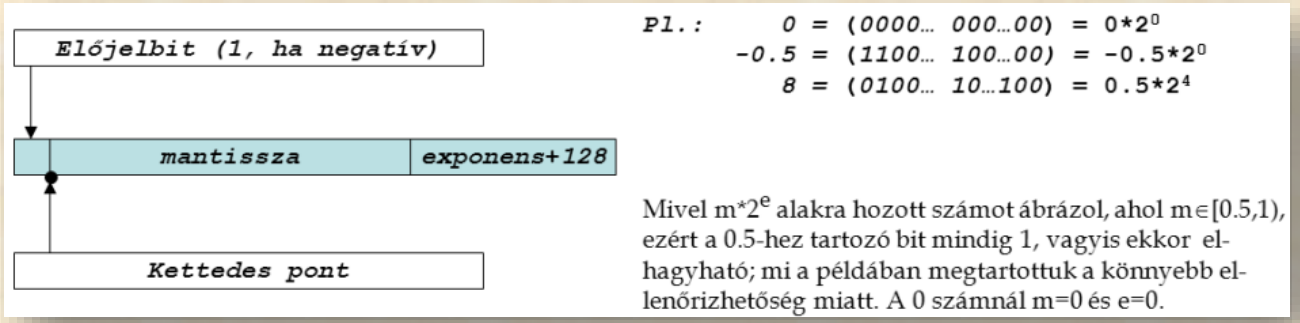
A beolvasáson, a kiíráson és értékadáson túliakkal foglalkozunk csak.

Pl. 3-bites 2-es komplemens 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

- Értékhalmaz: ????..????
  (Min'Valós..Max'Valós nem definiáltak, vagy implementáció 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

- Értékhalmaz: 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

- Értékhalmaz: 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!)

Feladatok elágazásra: vércsoport – 1

Feladat:

Egy ember vércsoportját (Rh negatív vagy pozitív) egy génpár határozza meg. Mindkét gén lehet „+” vagy „–” típusú. A „++” és a „+−” típusúak az „Rh pozitívok”, a „– –” típusúak pedig az „Rh negatívok”.

Írj programot, amely megadja egy ember vércsoportját a génpárja ismeretében!

Egy ember vércsoportját (Rh negatív vagy pozitív) egy génpár határozza meg. Mindkét gén lehet „+” vagy „–” típusú. A „++” és a „+–” típusúak az „Rh pozitívok”, a „––” típusúak pedig „Rh negatívok”.

elágazásra: vércsoport – 1

Specifikáció:

- Bemenet: x,y∈𝕂
- Kimenet: v∈**𝕊**
- Előfeltétel: x,y∈{"+", "–"}
- Utófeltétel: (x="+" vagy y="+") és v="Rh+" vagy (x="–" és y="–") és v="Rh–"

𝕂=Karakterek halmaza

𝕊=Karakter-sorozatok (szövegek) halmaza

Algoritmus:

Innentől kezdve elhagyjuk a programparaméterek **deklarálás**át, a **beolvasás**t, és **kiírás**t

x="+" vagy y="+"	
I	N
v:="Rh+"	v:="Rh–"

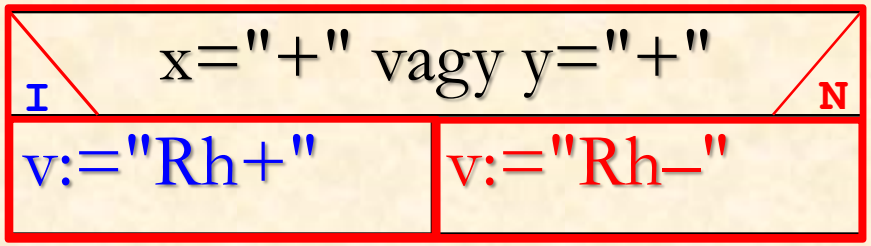
Feladatok elágazásra: vércsoport – 1

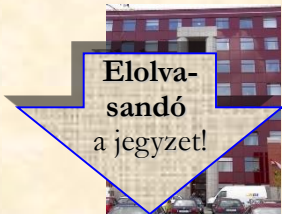
Specifikáció:

Egy ember vércsoportját (Rh negatív vagy pozitív) egy génpár határozza meg. Mindkét gén lehet „+” vagy „–” típusú. A „++” és a „+–” típusúak az „Rh pozitívok”, a „––” típusúak pedig „Rh negatívok”.

- Bemenet: x,y∈𝕂
- Kimenet: v∈**S**
- Előfeltétel: x,y∈{"+", "–"}
- Utófeltétel: (x="+" vagy y="+") → v="Rh+" és
  (x="–" és y="–") → v="Rh–"

Algoritmus:





Feladatok elágazásra: vércsoport – 2

Feladat:

Egy ember vércsoportját (A, B, AB vagy 0) egy gén-pár határozza meg. Mindkét gén lehet ***a, b*** *vagy* ***0*** *típusú.*

A vércsoport meghatározása: A={aa,a0,0a}; B={bb,b0,0b}; AB={ab,ba}; 0={00}.

Írj programot, amely megadja egy ember vércsoportját a génpárja ismeretében!

Feladatok elágazásra: vércsoport – 2

*Egy ember vércsoportját (A, B, AB vagy 0) egy gén-pár határozza meg. Mindkét gén lehet **a**, **b** vagy **0** típusú.*

A vércsoport meghatározása: A={aa,a0,0a}; B={bb,b0,0b}; AB={ab,ba}; 0={00}.

Specifikáció:

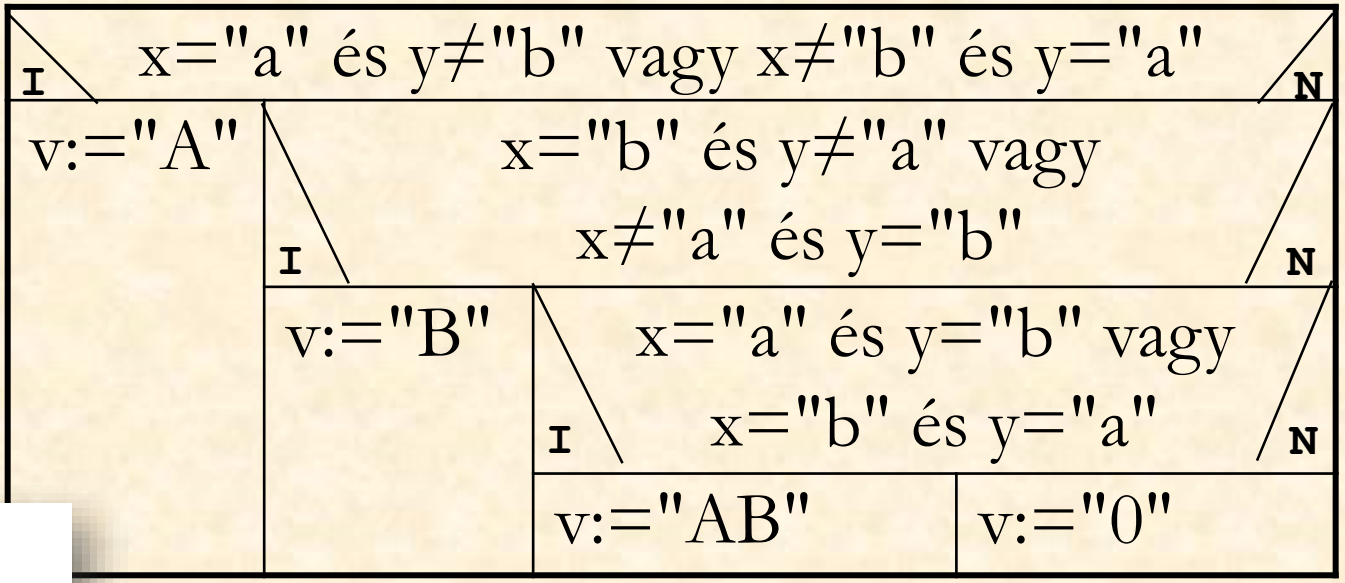
- Bemenet: x,y∈K
- Kimenet: v∈**S**
- Előfeltétel: x,y∈{"a", "b", "0"}
- Utófeltétel:

(x="a" és y≠"b" vagy x≠"b" és y="a")
és v="A" **vagy**
(x="b" és y≠"a" vagy x≠"a" és y="b")
és v="B" **vagy**
(x="a" és y="b" vagy x="b" és y="a")
és v="AB" **vagy**
x="0" és y="0" és v="0"

Feladatok elágazásra: vércsoport – 2

Algoritmus$_1$:

Kétirányú elágazások egymásba ágyazásával.



Specifikáció:

- Bemenet: x,y∈K
- Kimenet: v∈**S**
- Előfeltétel: x,y∈{"a", "b", "0"}
- Utófeltétel:

(x="a" és y≠"b" vagy x≠"b" és y="a")
és v="A" **vagy**
(x="b" és y≠"a" vagy x≠"a" és y="b")
és v="B" **vagy**
(x="a" és y="b" vagy x="b" és y="a")
és v="AB" **vagy**
x="0" és y="0" és v="0"

Feladatok elágazásra: vércsoport – 2

Algoritmus$_2$:

Sokirányú elágazással.

x="a" és y≠"b" vagy x≠"b" és y="a"	x="b" és y≠"a" vagy x≠"a" és y="b"	x="a" és y="b" vagy x="b" és y="a"	x="0" és y="0"
v:="A"	v:="B"	v:="AB"	v:="0"

Specifikáció:

- Bemenet: x,y∈K
- Kimenet: v∈**S**
- Előfeltétel: x,y∈{"a", "b", "0"}
- Utófeltétel:
  (x="a" és y≠"b" vagy x≠"b" és y="a") és v="A" **vagy**
  (x="b" és y≠"a" vagy x≠"a" és y="b") és v="B" **vagy**
  (x="a" és y="b" vagy x="b" és y="a") és v="AB" **vagy**
  x="0" és y="0" és v="0"

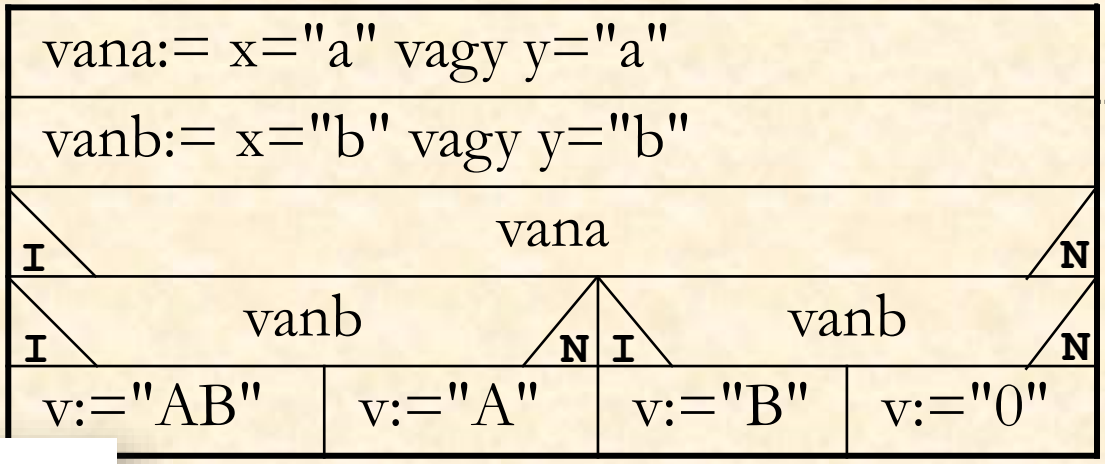
Feladatok elágazásra: vércsoport –

Lokális változók deklarálása

Algoritmus$_3$:

Változó
vana,
vanb:**Logikai**

Segédváltozók bevezetésével.



Specifikáció:

- Bemenet: x,y∈K
- Kimenet: v∈**S**
- Előfeltétel: x,y∈{"a", "b", "0"}
- Utófeltétel:
  (x="a" és y≠"b" vagy x≠"b" és y="a") és v="A" **vagy**
  (x="b" és y≠"a" vagy x≠"a" és y="b") és v="B" **vagy**
  (x="a" és y="b" vagy x="b" és y="a") és v="AB" **vagy**
  x="0" és y="0" és v="0"

Rekordok/Struktúrák

Feladat:

Adjuk meg, hogy egy P síkbeli pont melyik síknegyedbe esik!

Megoldás felé – specifikáció:

A síkbeli pontokat ($\mathbb{R}\times\mathbb{R}$, $\times$:direktszorzás művelet) x- és y-koordinátájukkal adjuk meg.

Ezt a specifikációban így írjuk le:

$P \in Pont$, $Pont = \mathbf{X}\times\mathbf{Y}$, $\mathbf{X},\mathbf{Y}=\mathbb{R}$

A nevükkel hivatkozhatunk rájuk a specifikációban:

$P = (P.\mathbf{x}, P.\mathbf{y})$.

Tehát: $P.\mathbf{x}\in\mathbf{X}$, $P.\mathbf{y}\in\mathbf{Y}$

Rekordok/Struktúrák

Feladat:

Adjuk meg, hogy egy P síkbeli pont melyik síknegyedbe esik!

Specifikáció:

Új halmaz definíciója

- Bemenet: $P \in Pont$, $Pont = \mathbf{X} \times \mathbf{Y}$, $\mathbf{X}, \mathbf{Y} = \mathbb{R}$
- Kimenet: $SN \in \mathbb{N}$
- Előfeltétel: –
- Utófeltétel: $P.x \geq 0$ és $P.y \geq 0 \rightarrow SN = 1$ és
  $P.x < 0$ és $P.y \geq 0 \rightarrow SN = 2$ és
  $P.x < 0$ és $P.y < 0 \rightarrow SN = 3$ és
  $P.x \geq 0$ és $P.y < 0 \rightarrow SN = 4$

Rekordok/Struktúrák

Specifikáció → algoritmus$_{\text{adatleírás}}$:

- P∈Pont, Pont=**X**×**Y**, **X,Y=**$\mathbb{R}$

Típus TPont=**Rekord**(**x,y:Valós**)

Változó P:TPont

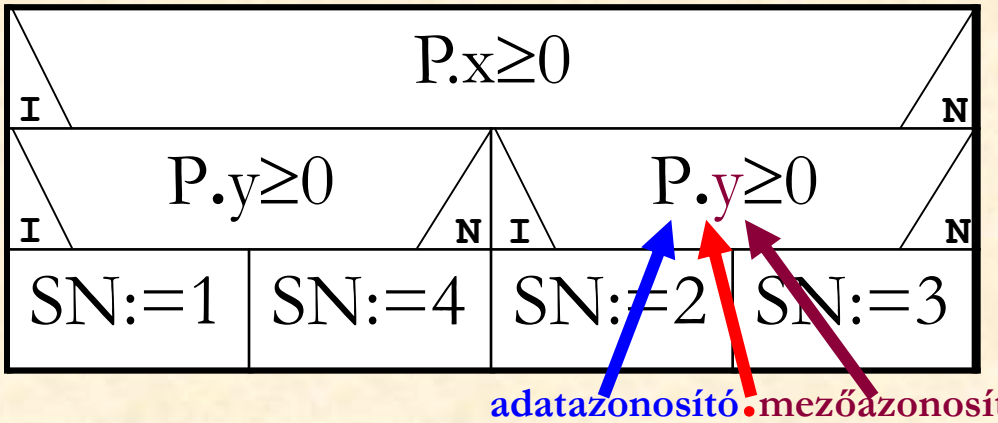
Tehát a TPont egy új összetett **adattípus.**

- A rekordok a tömbökhöz hasonlóan összetett adatszerkezetek, itt azonban az elemeknek nem indexük (sorszámuk) van, hanem nevük (P.**x**, P.**y**).

Rekordok/Struktúrák

Specifikáció → algoritmus$_{\text{tevékenység}}$:

- Utófeltétel: P.x≥0 és P.y≥0 → SN=1 és
  P.x<0 és P.y≥0 → SN=2 és
  P.x<0 és P.y<0 → SN=3 és
  P.x≥0 és P.y<0 → SN=4



Rekordok/Struktúrák

Specifikáció → algoritmus → kód:

- **Típus**

  TPont=**Rekord**(x,y:**Valós**)

Specifikáció:
- Bemenet: P∈Pont, Pont=**X**×**Y**, **X**,**Y**=ℝ

C++ típusdefiníció:

```
typedef struct {double x,y;} TPont;
```

- **Változó**

  P:TPont

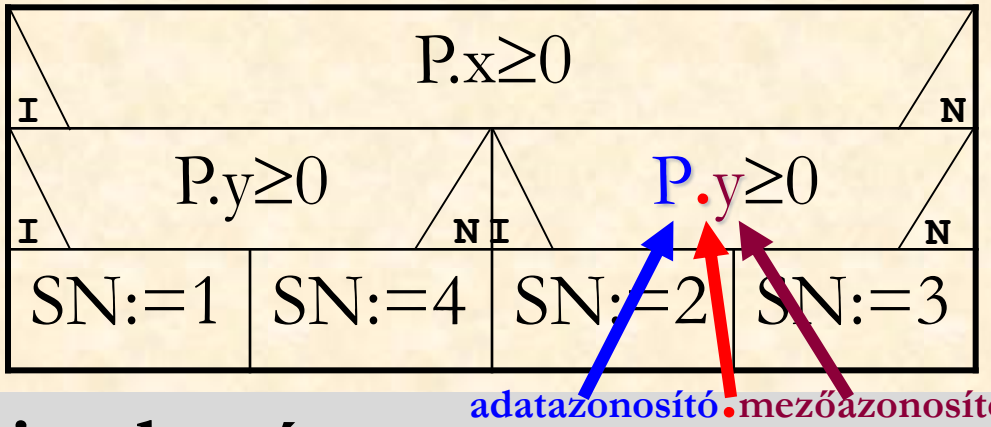
C++ típusdeklaráció:

```
TPont P;
```

Rekordok/Struktúrák

Algoritmus → kód:

- Utófeltétel: P.x≥0 és P.y≥0 → SN=1 és
  P.x<0 és P.y≥0 → SN=2 és
  P.x<0 és P.y<0 → SN=3 és
  P.x≥0 és P.y<0 → SN=4



C++ hivatkozás:

```
if (P.x>=0){
   if (P.y>=0)  SN=1;
          else  SN=4;
}else{
   if (P.y>=0)  SN=2;
          else  SN=3;
}
```

adatazonosító.mezőazonosító

Kódolás
(C++)

Kód:

kétirányú elágazás

```
if (Felt)
{
   Ut_I
}
else
{
   Ut_H
}
```

elhagyható

sokirányú (általános)

```
if (Felt_1) {
   Ut_1
}
else if (…) {
   …
}
else if (Felt_N) {
   Ut_N
}
else {
   Ut
}
```

elhagyható

Kódolási stílus-változatok (ANSI/K&R)

Kódolás (C++)

Kód:

kétirányú elágazás

```
if (Felt) {
   Ut_I
}
else {
   Ut_H
}
```

elhagyható

sokirányú (speciális)

```
switch (kif)
{
   case érték_1: Ut_1; break;
   case …      : …  ; break;
   case érték_N: Ut_N; break;
   default     : Ut_H;
}
```

elhagyható

Kódolási stílus-változatok (K&R/ANSI)

Áttekintés

- A problémamegoldás lépései – a programkészítés folyamata
- A specifikáció
- Az algoritmus
- Algoritmikus nyelvek – struktogram
- Adatokkal kapcsolatos fogalmak
- Adattípusok, elemi adattípusok
- Elemi feladatok
- Rekordok
- Kódolás