



ELTE | IK

PROGRAMOZÁS

2. előadás

Horváth Győző, Horváth Gyula, Szlávi Péter



Ismétlés



Feladatmegoldás lépései

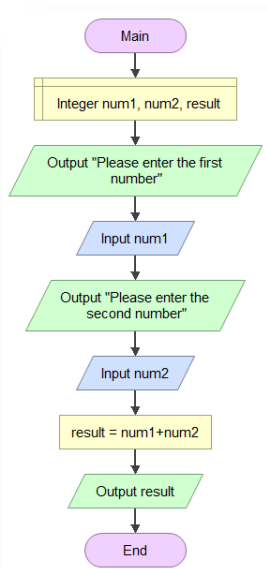
- Specifikáció
 - mi a feladat?
 - adatok, megszorítások, összefüggések
- Algoritmus
 - hogyan oldjuk meg a feladatot?
 - milyen lépésekre bontjuk?
 - szekvencia (utasítások egymás után)
 - elágazás (utasítások feltételes végrehajtása)
- Kód
 - megvalósítás a gép számára érthető módon
 - adatok deklarációja, beolvasás, feldolgozás, kiírás

Elágazás



Szekvencia

- Hétköznapi algoritmus
 - pl. recept
- Algoritmusleíró nyelvek



húst megtisztítani

1cm vastag szeletekre vágni

klopolás

...

Elkészítés

A húst a zsiradéktól megtisztítjuk, 1 cm vastag szeletekre vágjuk. Enyhén mindkét

1. oldalán kiklopfoljuk, kb. fél centi vastag szeleteket kapunk. Egy csipet sóval mindkét oldalát meghintjük.

2. Előbb lisztbe, majd kicsi sóval elkevert, felvert tojásba, végül zsemlemorzsába forgatjuk a húsokat.

Közepesen forró olajban, ami ellepi a hússzeleteket, szép aranybarnára kisütjük, először

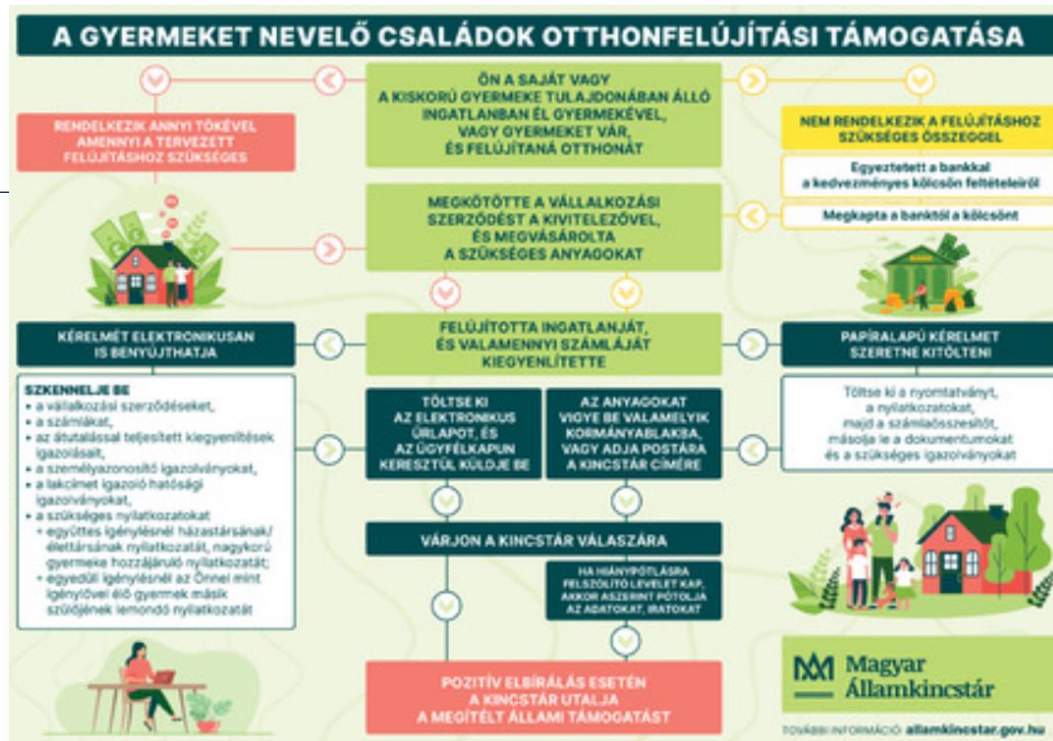
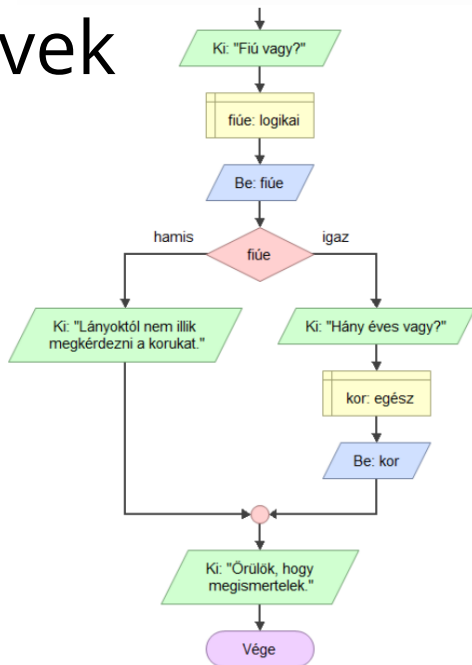
3. az egyik, majd a másik oldalát. (Én két részletben sütöttem ki: adagonként 30 perc alatt)

4. Háztartási papírtörlővel béelt tálba szedjük, hogy a felesleges olajat felitassuk róla.



Elágazás

- Hétköznapi algoritmus
 - pl: ügyintézés
- Algoritmusleíró nyelvek



rendelkezik saját tőkével?	
true	false
:)	bankkal egyeztetni
	hitelfelvétel

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!

Példa: $x=+$, $y=-$ \rightarrow $v=\text{Rh}+$

Feladatok elágazásra: vércsoport

Példa: $x="+" , y="-" \rightarrow v="Rh+"$

Specifikáció:

Be: $x \in C, y \in C$

Ki: $v \in S$

Ef: $(x="+" \text{ vagy } x="-") \text{ és } (y="+" \text{ vagy } y="-")$

Uf: $((x="+" \text{ vagy } y="+") \text{ és } v="Rh+") \text{ vagy } ((x="-" \text{ és } y="-") \text{ és } v="Rh-")$

C =Karakterek halmaza

S =Karakter-sorozatok
(szövegek) halmaza

Ef: $x, y \in \{ "+", "-" \}$

Algoritmus:

$\text{nem}(x="+" \text{ vagy } y="+")$

Elhagyjuk a
változók
deklarálását, a
beolvasást és a
kiírást

$x="+" \text{ vagy } y="+"$	
I	N
$v:="Rh+"$	$v:="Rh-"$

Feladatok elágazásra: vércsoport

1		
a	b	a->b
igaz	igaz	igaz
igaz	hamis	hamis
hamis	igaz	igaz
hamis	hamis	igaz

Specifikáció:

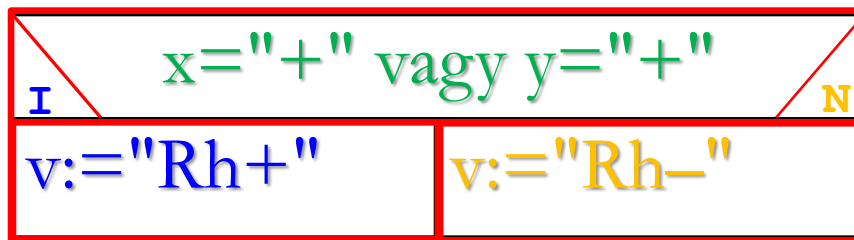
Be: $x \in C, y \in C$

Ki: $v \in S$

Ef: $(x = "+" \text{ vagy } x = "-") \text{ és } (y = "+" \text{ vagy } y = "-")$

Uf: $((x = "+" \text{ vagy } y = "+") \rightarrow v = "Rh+") \text{ és } (\text{nem}(x = "+" \text{ vagy } y = "+") \rightarrow v = "Rh-")$

Algoritmus:



Feladatok elágazásra: vércsoport – 2

Feladat:

*Egy ember vércsoportját (A, B, AB vagy 0) egy génpár 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!

Példa: $x="a", y="b" \rightarrow v="AB"$

Feladatok elágazásra: vércsoport – 2

Példa: $x="a", y="b" \rightarrow v="AB"$

Specifikáció:

Be: $x \in C, y \in C$

Ki: $v \in S$

Ef: $(x="a" \text{ vagy } x="b" \text{ vagy } x="0") \text{ és } (y="a" \text{ vagy } y="b" \text{ vagy } y="0")$

Uf: $((x="a" \text{ és } y \neq "b" \text{ vagy } x \neq "b" \text{ és } y="a") \rightarrow v="A") \text{ és } ((x="b" \text{ és } y \neq "a" \text{ vagy } x \neq "a" \text{ és } y="b") \rightarrow v="B") \text{ és } ((x="a" \text{ és } y="b" \text{ vagy } x="b" \text{ és } y="a") \rightarrow v="AB") \text{ és } ((x="0" \text{ és } y="0") \rightarrow v="0")$

Egy ember vércsoportját (A, B, AB vagy 0) egy génpár 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

Algoritmus₂:

Sokirányú
elágazással.

$x = "a"$ és $y \neq "b"$ vagy $x \neq "b"$ és $y = "a"$	$x = "b"$ és $y \neq "a"$ vagy $x \neq "a"$ 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:

Be: $x \in C$, $y \in C$

Ki: $v \in S$

Ef: $(x = "a" \text{ vagy } x = "b" \text{ vagy } x = "0")$ és
 $(y = "a" \text{ vagy } y = "b" \text{ vagy } y = "0")$

Uf: $(x = "a" \text{ és } y \neq "b" \text{ vagy } x \neq "b" \text{ és } y = "a" \rightarrow v = "A")$ és
 $(x = "b" \text{ és } y \neq "a" \text{ vagy } x \neq "a" \text{ és } y = "b" \rightarrow v = "B")$ és
 $(x = "a" \text{ és } y = "b" \text{ vagy } x = "b" \text{ és } y = "a" \rightarrow v = "AB")$ és
 $(x = "0" \text{ és } y = "0" \rightarrow v = "0")$

Feladatok elágazásra: vércsoport – 2

Algoritmus₁:

Kétirányú
elágazások
egymásba
ágyazásával.

Specifikáció:

Be: $x \in C$, $y \in C$

Ki: $v \in S$

Ef: $(x = "a" \text{ vagy } x = "b" \text{ vagy } x = "0") \text{ és } (y = "a" \text{ vagy } y = "b" \text{ vagy } y = "0")$

Uf: $(x = "a" \text{ és } y \neq "b" \text{ vagy } x \neq "b" \text{ és } y = "a" \rightarrow v = "A") \text{ és } (x = "b" \text{ és } y \neq "a" \text{ vagy } x \neq "a" \text{ és } y = "b" \rightarrow v = "B") \text{ és } (x = "a" \text{ és } y = "b" \text{ vagy } x = "b" \text{ és } y = "a" \rightarrow v = "AB") \text{ és } (x = "0" \text{ és } y = "0" \rightarrow v = "0")$

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

Feladatok elágazásra: vércsoport –

Lokális változók
deklarálása

Algoritmus₃:

Segédváltozók
bevezetésével.

vana:= x="a" vagy y="a"			
vanb:= x="b" vagy y="b"			
I	vana		N
I	vanb	N	I
v:="AB"		v:="A"	I
		N	v:="B"
			N
			v:="0"

Változó
vana,
vanb:**Logikai**

Specifikáció:

Be: x∈C, y∈C

Ki: v∈S

Ef: (x="a" vagy x="b" vagy x="0") és
(y="a" vagy y="b" vagy y="0")

Uf: (x="a" és y≠"b" vagy x≠"b" és y="a" -> v="A") és
(x="b" és y≠"a" vagy x≠"a" és y="b" -> v="B") és
(x="a" és y="b" vagy x="b" és y="a" -> v="AB") és
(x="0" és y="0" -> v="0")

Elágazás

Kód:

kétirányú

```
if (felt) {  
    utasítás1  
}  
else {  
    utasítás2  
}
```

elágazás

sokirányú
(általános)

```
if (felt1) {  
    utasítás1  
}  
else if (...) {  
    ...  
}  
else if (feltn) {  
    utasításn  
}  
else {  
    utasítás  
}
```

Elágazás

Kód:

sokirányú
elágazás (speciális)

```
switch (kif)
{
    case érték1: utasítás1; break;
    case ... : ... ; break;
    case értékN: utasításN; break;
    default elhagyható; break;
}
```


Kód

```
// 1. deklarálás
```

```
char x, y;
```

```
string v = "";
```

```
// 2. beolvasás
```

```
Console.Write("x = ");
```

```
char.TryParse(Console.ReadLine(), out x);
```

```
Console.Write("y = ");
```

```
char.TryParse(Console.ReadLine(), out y);
```

```
// 3. feldolgozás
```

```
if ((x == 'a' && y != 'b') || (x != 'b' && y == 'a')) {
```

```
    v = "A";
```

```
}
```

```
else if ((x == 'b' && y != 'a') || (x != 'a' && y == 'b')) {
```

```
    v = "B";
```

```
}
```

```
else if ((x == 'a' && y == 'b') || (x == 'b' && y == 'a')) {
```

```
    v = "AB";
```

```
}
```

```
else if (x == '0' && y == '0') {
```

```
    v = "0";
```

```
}
```

```
// 4. kiírás
```

```
Console.WriteLine("v = {0}", v);
```

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"

Ciklus

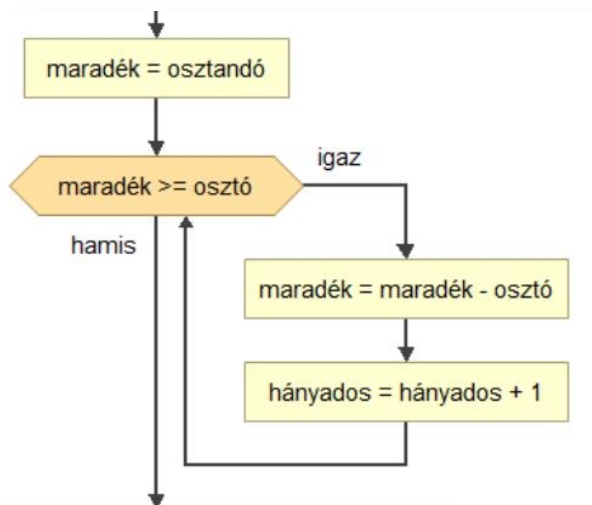


Ciklus

- Ismételt végrehajtás
- Hétköznapi algoritmusok
 - Addig jár a korsó a kútra, **amíg** el nem törik
 - Recept
- Algoritmusleíró nyelvek

Egyszerűen csak várjuk meg, **amíg** felforr a víz és tegyük bele a tojásokat, így biztos nem okoz majd nagy nehézséget a hámozás.

ÁPRILY LAJOS: ÁMULNI MÉG... (részlet)
„Ámulni még, **ameddig** lehet, **amíg** a szíved jó ütemre dobban, megőrizni a táguló szemet, mellyel csodálkoztál gyermekkorodban..



nem törött?

menj a kútra korsó

Ciklusok

Feladat:

Add meg egy természetes szám (>1) **1-től különböző legkisebb osztóját!**

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $o \in \mathbb{N}$

Ef: $n > 1$

Uf: $1 < o \leq n$ és $o \mid n$ és $\forall i \in [2..o-1]: (i \nmid n)$

Példa: $n=15 \rightarrow o=3$

A megoldás reprezentálása:

Specifikáció:

Be: $n \in \mathbb{N}$
Ki: $o \in \mathbb{N}$
Ef: $n > 1$
Uf: $1 < o \leq n$ és $o \mid n$ és
 $\forall i \in [2..o-1]: (i \nmid n)$

Változó

n : **Egész**

o : **Egész**

Programváltozók
deklarálása

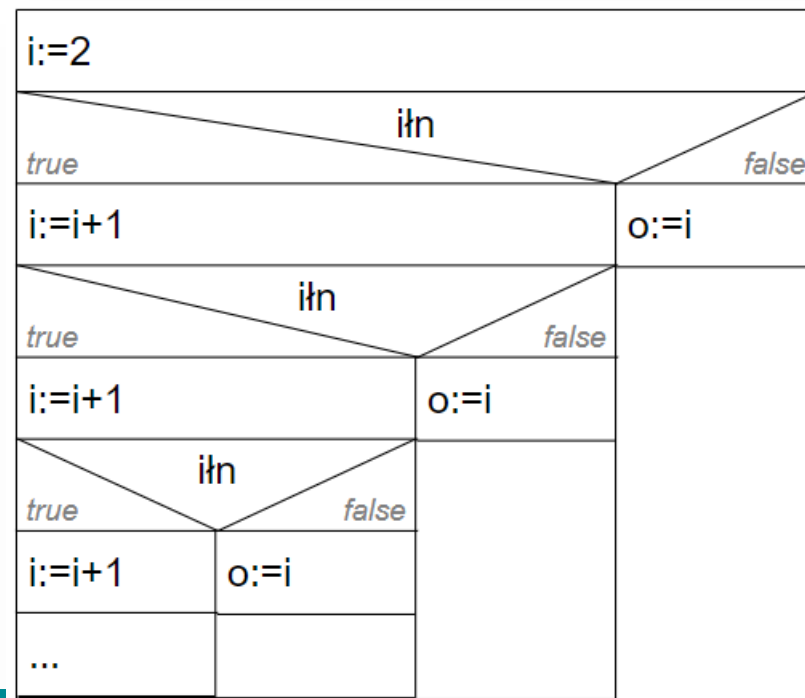
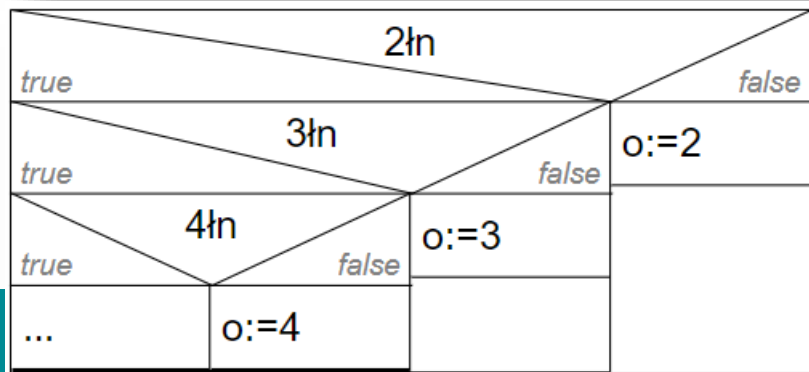
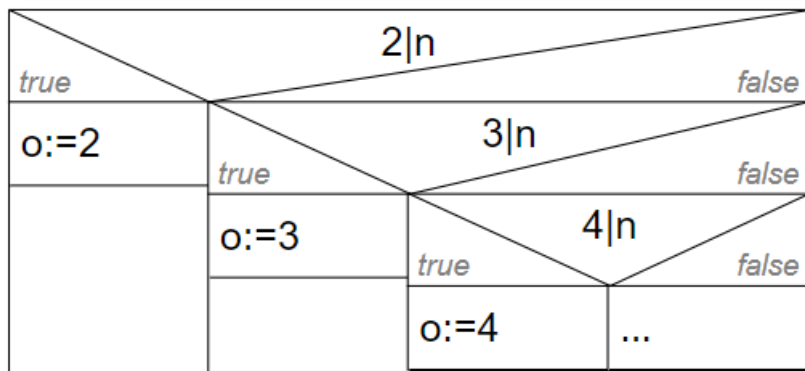
Reprezentációs „szabály” a specifikáció \rightarrow reprezentáció
áttéréskor:

$n \rightarrow$ **Egész**

Ciklusok

A megoldás ötlete:

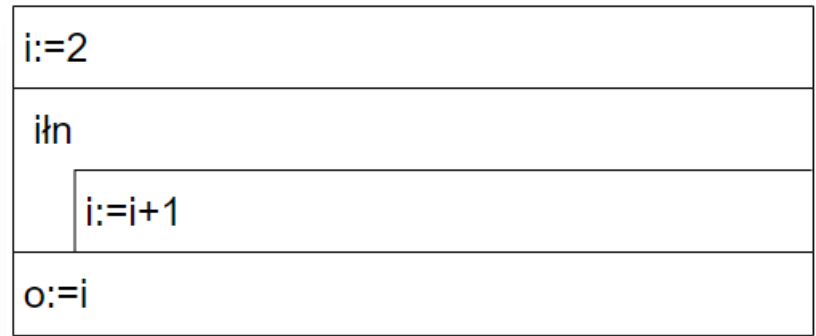
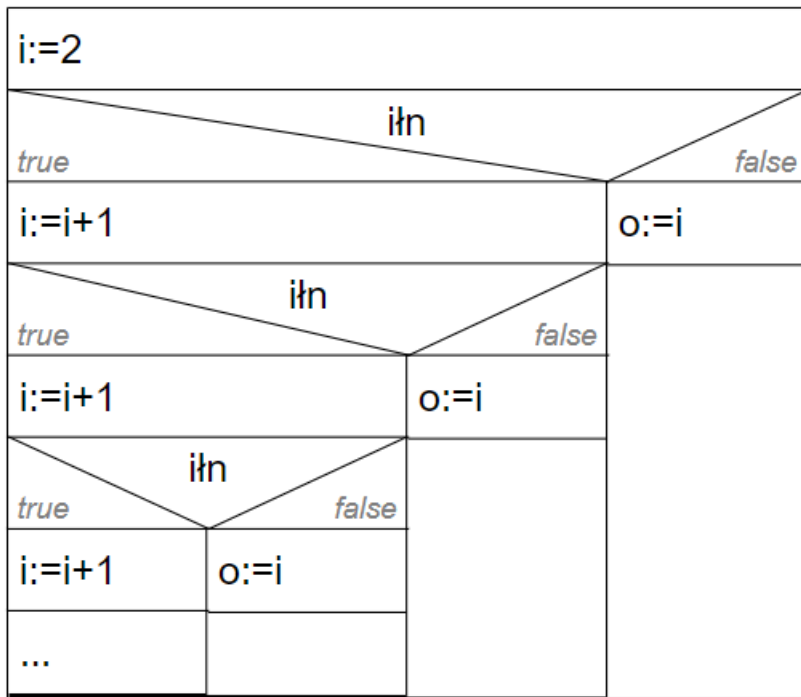
Próbáljuk ki a 2-t; ha nem jó, akkor a 3-at, ha az sem, akkor a 4-et, ...; legkésőbb az n jó lesz!



Ciklusok

A megoldás ötlete:

Próbáljuk ki a 2-t; ha nem jó, akkor a 3-at, ha az sem, akkor a 4-et, ...; legkésőbb az n jó lesz!



Ciklusok

A megoldás ötlete:

Próbáljuk ki a 2-t; ha nem jó, akkor a 3-at,
ha az sem, akkor a 4-et, ...; legkésőbb az n jó lesz!

Az ezt kifejező lényegi algoritmus:

Az i változó szerepe: végigmenni egy halmaz elemein.

Specifikáció:

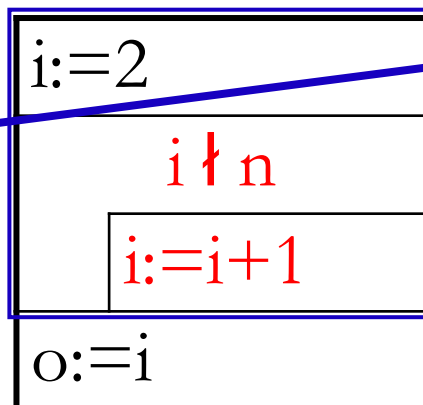
Be: $n \in \mathbb{N}$

Ki: $o \in \mathbb{N}$

Ef: $n > 1$

Uf: $1 < o \leq n$ és $o | n$ és

$\forall i \in [2..o-1]: (i \nmid n)$



Változó
 i :Egész

Lokális változó
deklarálása

Ciklusok

Példa: $n=15 \rightarrow lko=3; lno=5$

Feladat:

Határozzuk meg egy természetes szám ($n>1$) **1-től különböző legkisebb** és **önmagától különböző legnagyobb** osztóját!

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $lko \in \mathbb{N}, lno \in \mathbb{N}$

Ef: $n > 1$

Uf: $1 < lko \leq n$ és $1 \leq lno < n$ és

$lko \mid n$ és $\forall i \in [2..lko-1]: (i \nmid n)$ és

$lno \mid n$ és $\forall i \in [lno+1..n-1]: (i \nmid n)$

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $o \in \mathbb{N}$

Ef: $n > 1$

Uf: $1 < o \leq n$ és $o \mid n$ és
 $\forall i \in [2..o-1]: (i \nmid n)$

Ciklusok

Megjegyzés:

A specifikációból az algoritmus megkapható, de az lno az utófeltételben az lko ismeretében másképp is megfogalmazható: $lko * lno = n$!

Az erre építő algoritmus:

Specifikáció:

Be: $n \in \mathbb{N}$
Ki: $lko \in \mathbb{N}, lno \in \mathbb{N}$
Ef: $n > 1$
Uf: $1 < lko \leq n$ és
 $lko \mid n$ és
 $\forall i \in [2..lko-1]: (i \nmid n)$ és
 $lko * lno = n$

$i := 2$
$i \nmid n$
$i := i + 1$
$lko := i$
$lno := n \text{ div } lko$

Változó
 $i: \text{Egész}$

Ciklusok

Feladat:

Határozzuk meg egy természetes szám ($n > 1$) 1-től és önmagától különböző legkisebb osztóját (ha van)!

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $o \in \mathbb{N}$, $van \in \mathbb{L}$

Ef: $n > 1$

Uf: $van = \exists i \in [2..n-1] : (i | n)$ és
 $van \rightarrow (2 \leq o < n \text{ és } o | n \text{ és } \forall i \in [2..o-1] : (i \nmid n))$

Példa:

$n=15 \rightarrow van=igaz; o=3$

$n=17 \rightarrow van=hamis; o=???$

Ciklusok

Algoritmus:

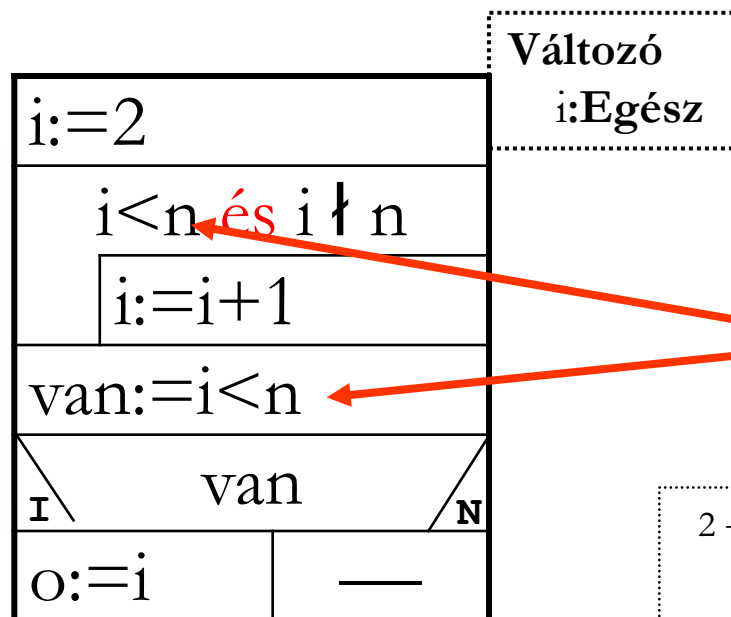
Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $o \in \mathbb{N}$, $van \in \mathbb{L}$

Ef: $n > 1$

Uf: $van = \exists i \in [2..n-1] : (i \mid n)$ és
 $van \rightarrow (2 \leq o < n \text{ és } o \mid n \text{ és } \forall i \in [2..o-1] : (i \nmid n))$



$$i \leq \sqrt{N}$$

$$2 \rightarrow i \leq N \text{ Div } i \leftarrow N \text{ Div } 2$$

azaz

$$i * i \leq N$$

azaz

$$i \leq \sqrt{N}$$

Megjegyzés:

Ha i osztója n -nek, akkor $(n \text{ div } i)$ is osztója, azaz elég az osztókat a szám gyökéig keresni!

Ciklusok

Feladat:

Határozzuk meg egy természetes szám ($N > 1$) osztói összegét!

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $s \in \mathbb{N}$

Ef: $n > 1$

Uf: $s = \text{SZUM}(i=1..n, i, i | n)$

$$s = \sum_{\substack{i=1 \\ i|n}}^n i$$

A feltételes szumma értelmezéséhez egy példa:

$N=15 \rightarrow \Sigma =$

$i=1 : (1 | 15) \rightarrow 1$

$i=2 : (2 \nmid 15) \rightarrow 1$

$i=3 : (3 | 15) \rightarrow 1+3$

$i=4 : (4 \nmid 15) \rightarrow 1+3$

...

$i=15 : (15 | 15) \rightarrow 1+3+\dots+15$

Ciklusok

Algoritmus:

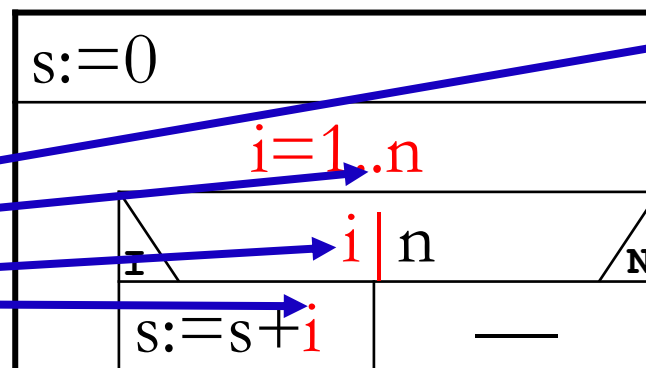
Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $s \in \mathbb{N}$

Ef: $n > 1$

Uf: $s = \text{SZUM}(i=1..n, i, i|n)$



Változó
 i :Egész

Az s változót nem egy képlettel számoljuk, hanem gyűjtjük benne az eredményt.

Kérdés:

Lehetne itt is \sqrt{n} -ig menni?

Az $s := s + i + (n \text{ div } i)$ értékadással?

Ciklusok

Feladat:

Határozzuk meg egy természetes szám ($n > 1$) páratlan osztói összegét!

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $s \in \mathbb{N}$

Ef: $n > 1$

Uf: $s = \text{SZUM}(i=1..n, i,$
 $i | n \text{ és páratlan}(i))$

$$s = \sum_{\substack{i=1 \\ i|n \text{ és páratlan}(i)}}^n i$$

páratlan(i)=???

Ciklusok

Algoritmus₁:

Specifikáció:

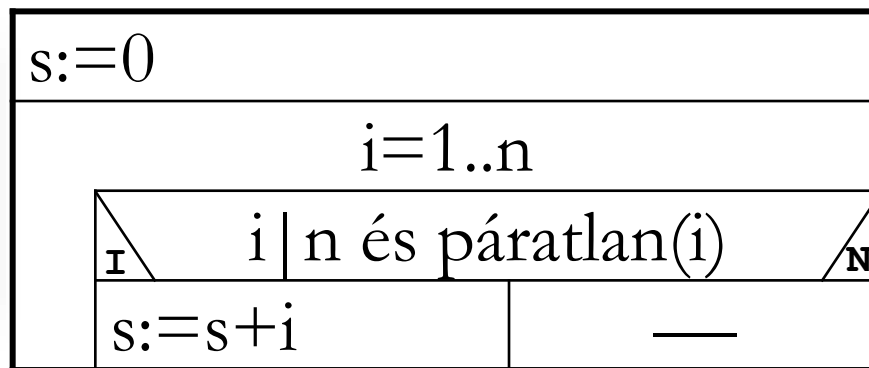
Be: $n \in \mathbb{N}$

Ki: $s \in \mathbb{N}$

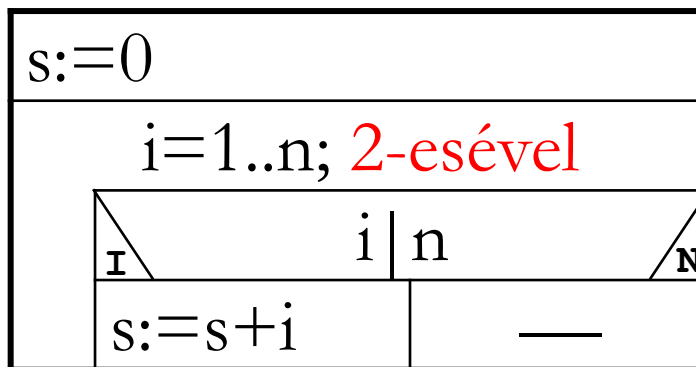
Ef: $n > 1$

Uf: $s = \text{SZUM}(i=1..n, i, i|n \text{ és páratlan}(i))$

Algoritmus₂:



Változó
 i :Egész



Változó
 i :Egész

Ciklusok

Tanulságok:

- Ha az utófeltételben \exists , \forall , vagy Σ jel van, akkor a megoldás mindig **ciklus**!
- Ha az utófeltételben \exists vagy \forall jel van, akkor a megoldás sokszor **feltételes ciklus**!
- Ha az utófeltételben Σ jel van, akkor a megoldás sokszor **számlálós ciklus**! (Π is...)
- **Feltételes Σ esetén a ciklusban elágazás lesz.**

Ciklusok

Feltételes ciklus:

Tipikus előfordulás: a beolvasás ellenőrzésénél

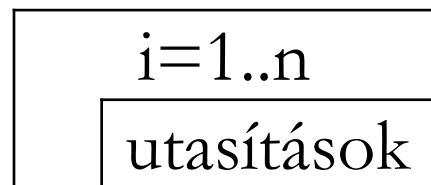


```
while (feltétel){  
    utasítások  
}
```

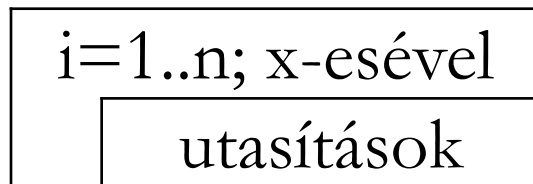


```
do{  
    utasítások  
} while (feltétel);
```

Számlálós ciklus:



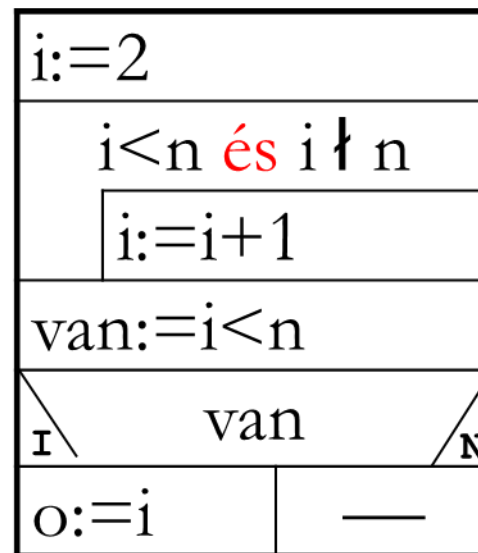
```
for (int i=1;i<=n;++i){  
    utasítások  
}
```



```
for (int i=1;i<=n;i+=x){  
    utasítások  
}
```

Kód

```
// 1. deklarálás
int n;
int o = 0;
bool van;
// 2. beolvasás
Console.Write("n = ");
int.TryParse(Console.ReadLine(), out n);
// 3. feldolgozás
int i = 2;
while (i < n && n % i != 0) {
    i = i + 1;
}
van = i < n;
if (van) {
    o = i;
}
// 4. kiírás
if (van) {
    Console.WriteLine("o = {0}", o);
}
else {
    Console.WriteLine("Nincs osztó");
}
```



Változó
i:Egész

Kód – ellenőrzött beolvasás

```
bool jo;  
do {  
    Console.Write("n = ");  
    jo = int.TryParse(Console.ReadLine(), out n);  
    jo = jo && (n > 1);  
    if (!jo) {  
        Console.WriteLine("Nem jó!");  
    }  
} while (!jo);
```

Specifikáció:

Be: $n \in \mathbb{N}$

Ki: $o \in \mathbb{N}$, $van \in \mathbb{L}$

Ef: $n > 1$

Uf: $van = \exists i \in [2..n-1] : (i | n)$ és
 $van \rightarrow (2 \leq o < n \text{ és } o | n \text{ és } \forall i \in [2..o-1] : (i \nmid n))$

Rekord



Rekord

Példák:

$x=3,3; y=2,1 \rightarrow sn=1$

$x=3,3; y=-2 \rightarrow sn=4$

...

Feladat:

Adjuk meg, hogy egy síkbeli pont melyik síknegyedbe esik!

Specifikáció₁ és algoritmus₁:

Be: $x \in \mathbb{R}, y \in \mathbb{R}$

Ki: $sn \in \mathbb{N}$

Ef: -

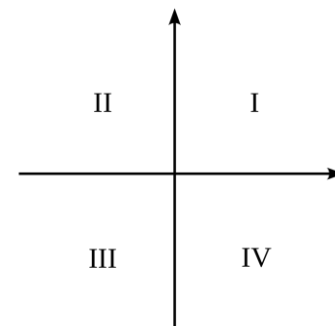
Uf: $((x \geq 0 \text{ és } y \geq 0) \rightarrow sn=1)$ és

$((x < 0 \text{ és } y \geq 0) \rightarrow sn=2)$ és

$((x < 0 \text{ és } y < 0) \rightarrow sn=3)$ és

$((x \geq 0 \text{ és } y < 0) \rightarrow sn=4)$

$\begin{array}{l} x \geq 0 \text{ és} \\ y \geq 0 \end{array}$	$\begin{array}{l} x < 0 \text{ és} \\ y \geq 0 \end{array}$	$\begin{array}{l} x < 0 \text{ és} \\ y < 0 \end{array}$	$\begin{array}{l} x \geq 0 \text{ és} \\ y < 0 \end{array}$
$sn:=1$	$sn:=2$	$sn:=3$	$sn:=4$



Be: $x \in \mathbb{R}, y \in \mathbb{R}$
Ki: $sn \in \mathbb{N}$

Rekord

// 1. deklarálás

```
double x, y;
```

```
int sn = 0;
```

// 2. beolvasás

```
Console.Write("x = ");
```

```
double.TryParse(Console.ReadLine(), out x);
```

```
Console.Write("y = ");
```

```
double.TryParse(Console.ReadLine(), out y);
```

// 3. feldolgozás

```
if (x >= 0 && y >= 0) { sn = 1; }
```

```
else if (x < 0 && y >= 0) { sn = 2; }
```

```
else if (x < 0 && y < 0) { sn = 3; }
```

```
else if (x >= 0 && y < 0) { sn = 4; }
```

// 4. kiírás

```
Console.WriteLine("Síknegyed = {0}", sn);
```

$\begin{array}{l} x \geq 0 \text{ és} \\ y \geq 0 \end{array}$	$\begin{array}{l} x < 0 \text{ és} \\ y \geq 0 \end{array}$	$\begin{array}{l} x < 0 \text{ és} \\ y < 0 \end{array}$	$\begin{array}{l} x \geq 0 \text{ és} \\ y < 0 \end{array}$
sn:=1	sn:=2	sn:=3	sn:=4

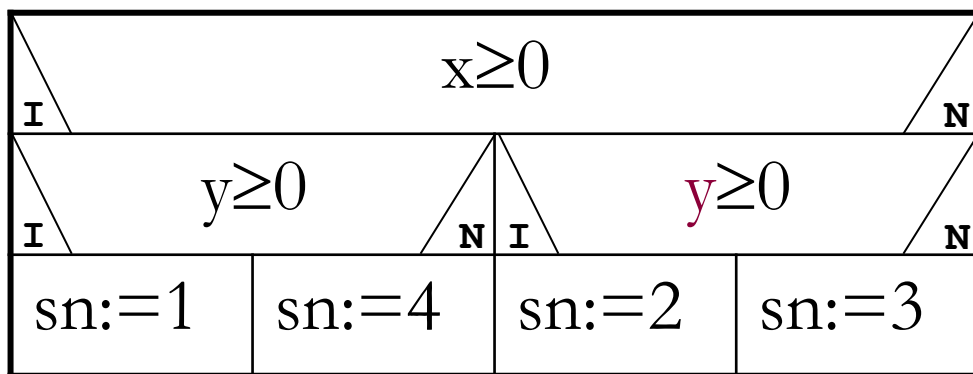
Rekord

Specifikáció₂ és algoritmus₂:

Uf: $(x \geq 0 \rightarrow (y \geq 0 \rightarrow sn = 1 \text{ és } y < 0 \rightarrow sn = 4))$

és

$(x < 0 \rightarrow (y \geq 0 \rightarrow sn = 2 \text{ és } y < 0 \rightarrow sn = 3))$



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


Rekord

- Jelenlegi megoldás:
 - nincs x és y között szemantikus kapcsolat
 - pedig nemcsak két szám, hanem **együtt** jelölnek egy koordinátpárt
- **Rekord:**
 - *különböző funkciójú adatok egybezárása*
 - szemantikus egység létrehozása
 - „funkció”: mit *jelent* az adat

Rekord

Példák:

$(x=3,3; y=2,1) \rightarrow sn=1$

$(x=3,3; y=-2) \rightarrow sn=4$

Specifikációbeli jelölés:

- egy adat

- $x \in R$

$p \in R \times R$

- adattöbbség

új halmaz definíciója

- $p \in \text{Pont}$, $\text{Pont} = X \times Y$, $X = R$, $Y = R$

- x : direkt szorzat

- hivatkozás a **nevükkel**

- $p.x$ ($p.x \in X$, azaz $p.x \in R$)

- $p.y$ ($p.y \in Y$, azaz $p.y \in R$)

Direkt szorzat példa:

$a \in [1..2]$, $b \in [4..5]$

a	b
1	4
1	5
2	4
2	5

A direkt szorzat felsorolja az összes lehetséges adatpárt.

Rekord

Példák:

$(x=3,3; y=2,1) \rightarrow sn=1$

$(x=3,3; y=-2) \rightarrow sn=4$

Feladat:

Adjuk meg, hogy egy síkbeli pont melyik síknegyedbe esik!

Specifikáció₃ és algoritmus₃:

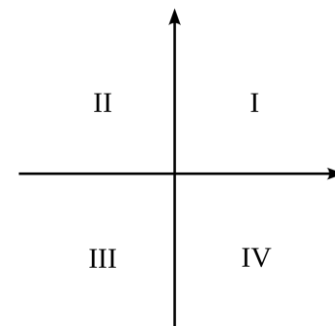
Be: $p \in \text{Pont}$,
Pont = X x Y,
X = R, Y = R

Ki: $sn \in \mathbb{N}$

Ef: -

Uf: $((p.x \geq 0 \text{ és } p.y \geq 0) \rightarrow sn=1)$ és
 $((p.x < 0 \text{ és } p.y \geq 0) \rightarrow sn=2)$ és
 $((p.x < 0 \text{ és } p.y < 0) \rightarrow sn=3)$ és
 $((p.x \geq 0 \text{ és } p.y < 0) \rightarrow sn=4)$

$p.x \geq 0 \text{ és } p.y \geq 0$	$p.x < 0 \text{ és } p.y \geq 0$	$p.x < 0 \text{ és } p.y < 0$	$p.x \geq 0 \text{ és } p.y < 0$
sn:=1	sn:=2	sn:=3	sn:=4



Rekord

Specifikáció → algoritmus_{adateleírás}

- $p \in \text{Pont}$, $\text{Pont} = X \text{ } \boxed{x} \text{ } Y, \boxed{X=R, Y=R}$

Típus $\text{TPont} = \text{Rekord}(x, y: \text{Valós})$

Változó $P: \text{TPont}$

Tehát a TPont egy új **adattípus**.

- A rekordok **összetett** adatszerkezetek, a részeikre „**nevük**” által meghatározott **szelektorok**kal hivatkozunk:
 $P = (P.x, P.y)$.

Rekord – típusdefiniálás kódban

Specifikáció → algoritmus → kód:

- Típus

TPont=Rekord(x,y:Egész)

Típusdefiniáció

C# típusdefiniáció:

```
struct TPont  
{  
    public int x,y;  
}
```

típusazonosító

hozzáférési jog

mezőtípus

mezőazonosítók

Rekord – típusdeklarálás kódban

Specifikáció → algoritmus → kód:

- Változó

P:TPont

Típusdeklaráció

C# típusdeklaráció:

TPont P; ← adattazonosító

típusazonosító

Kód

$\begin{array}{l} p.x \geq 0 \text{ és} \\ p.y \geq 0 \end{array}$	$\begin{array}{l} p.x < 0 \text{ és} \\ p.y \geq 0 \end{array}$	$\begin{array}{l} p.x < 0 \text{ és} \\ p.y < 0 \end{array}$	$\begin{array}{l} p.x \geq 0 \text{ és} \\ p.y < 0 \end{array}$
sn:=1	sn:=2	sn:=3	sn:=4

```
struct Pont {  
    public double x, y;  
}  
  
static void Main(string[] args) {  
    // 1. deklarálás  
    //double x, y;  
    Pont p;  
    int sn = 0;  
    // 2. beolvasás  
    Console.Write("x = ");  
    double.TryParse(Console.ReadLine(), out p.x);  
    Console.Write("y = ");  
    double.TryParse(Console.ReadLine(), out p.y);  
    // 3. feldolgozás  
    if (p.x >= 0 && p.y >= 0) { sn = 1; }  
    else if (p.x < 0 && p.y >= 0) { sn = 2; }  
    else if (p.x < 0 && p.y < 0) { sn = 3; }  
    else if (p.x >= 0 && p.y < 0) { sn = 4; }  
    // 4. kiírás  
    Console.WriteLine("Síknegyed = {0}", sn);  
}
```

Tömb



Feladat elágazásra – vagy más kell?

Feladat:

A japán naptár 60 éves ciklusokat tartalmaz, az éveket párosítják, s mindegyik párhoz valamilyen színt rendelnek (zöld, piros, sárga, fehér, fekete).

- o 1,2,11,12, ...,51,52: zöld évek
- o 3,4,13,14,...,53,54: piros évek
- o 5,6,15,16,...55,56: sárga évek
- o 7,8,17,18,...57,58: fehér évek
- o 9,10,19,20,...,59,60: fekete évek

Tudjuk, hogy 1984-ben indult az utolsó ciklus, amely 2043-ban fog véget érni.

Írj programot, amely megadja egy M évről ($1984 \leq M \leq 2043$), hogy milyen színű!

Feladat elágazásra – vagy más kel

Példa: $n=2023 \rightarrow s=\text{"fekete"}$

Specifikáció:

Be: $\text{év} \in \mathbb{N}$

Ki: $s \in \text{Szín}$,

$\text{Szín} = \{\text{"zöld"}, \text{"piros"}, \text{"sárga"}, \text{"fehér"}, \text{"fekete"}\}$

Ef: $1984 \leq \text{év} \leq 2043$

Uf: $y = ((\text{év} - 1984) \bmod 10) \div 2$ és

$y=0 \rightarrow s=\text{"zöld"}$ és

$y=1 \rightarrow s=\text{"piros"}$ és

...

- o 1,2,11,12, ...,51,52: zöld évek
- o 3,4,13,14,...,53,54: piros évek
- o 5,6,15,16,...55,56: sárga évek
- o 7,8,17,18,...57,58: fehér évek
- o 9,10,19,20,...,59,60: fekete évek

A Szín halmaz
definiálása.

év	év-1984	mod 10	div 2
1984	0	0	0
1985	1	1	0
1986	2	2	1
1987	3	3	1
1988	4	4	2
1989	5	5	2
1990	6	6	3
1991	7	7	3
1992	8	8	4
1993	9	9	4
1994	10	0	0
1995	11	1	0
1996	12	2	1
1997	13	3	1
1998	14	4	2
1999	15	5	2
2000	16	6	3
2001	17	7	3
2002	18	8	4
2003	19	9	4

Állapottér bővítés

y	szín
0	zöld
1	piros
2	sárga
3	fehér
4	fekete

Feladat elágazásra – vagy más kell?

Lokális változó
deklarálása

Algoritmus:

Változó
y:Egész

$y := ((\text{év} - 1984) \bmod 10) \text{ Div } 2$				
y=0	y=1	y=2	y=3	y=4
s:= "zöld"	s:= "piros"	s:= "sárga"	s:= "fehér"	s:= "fekete"

Kérdés:

Akkor is ezt tennénk, ha 5 helyett
90 ágat kellene írjunk?

A válasz előtt egy új adatszerkezet: a
tömb.

Specifikáció:

Be: $\text{év} \in \mathbb{N}$

Ki: $s \in \text{Szín}$,

$\text{Szín} = \{ \text{"zöld"}, \text{"piros"}, \text{"sárga"}, \text{"fehér"}, \text{"fekete"} \}$

Ef: $1984 \leq \text{év}$ és $\text{év} \leq 2043$

Uf: $((\text{év} - 1984) \bmod 10) \text{ div } 2 = 0 \rightarrow s = \text{"zöld"}$ és
 $((\text{év} - 1984) \bmod 10) \text{ div } 2 = 1 \rightarrow s = \text{"piros"}$ és
...

y	szín
0	zöld
1	piros
2	sárga
3	fehér
4	fekete

Tömb

Összerendelés

y	szín
0	zöld
1	piros
2	sárga
3	fehér
4	fekete

Excel

	A	
1	zöld	
2	piros	
3	sárga	
4	fehér	
5	fekete	
6		
7		
8		

hivatkozás:

A2 → piros

Tömb

	szín
0	zöld
1	piros
2	sárga
3	fehér
4	fekete

hivatkozás:

szín[2] → sárga

Sorozatok

Azonos funkció →
azonos halmazbeli

Specifikációbeli fogalmak:

- Sorozat: **azonos funkciójú elemek** egymásutánja, az elemei sorszámozhatók.
- Elem: a sorozat **i-edik elem**ére hivatkozás: $s[i]$.
- Index: $tól \dots ig$, tipikusan $1 \dots SorozatHossz$
- Például:
 - $HónapHosszak \in N[1..12]$ – a HónapHosszak 12 elemű, természetes számokból álló sorozat $\cong (HónapHosszak[1], \dots, HónapHosszak[12])$
 - $Emeletek \in S[-1..12]$ – az Emeletek 12 elemű, szövegeket tartalmazó sorozat $\cong (Emeletek[-1], Emeletek[0], \dots, Emeletek[10]) = ("Pince", "Földszint", \dots)$

Kérdés: az elemek lehetnek sorozatok, azaz van-e **sorozatok sorozata**?

Tömbök

Algoritmikus fogalmak:

- Tömb: véges hosszúságú sorozat algoritmikus párja, amelynek **i-edik tag**jával végezhetünk műveleteket (**adott a legkisebb és a legnagyobb index**, vagy az **elemszám**).
- Index: sokszor **1..n**, időnként **0..n-1**, ahol n az elemek számát jelöli. Általánosan **a..b** ($a \leq b$).
- Tömb-művelet: **értékadás**, pl. $t2 := t1$
(az értékazonosság operátort nem értelmezzük).
- Tömbelem-műveletek:
 - **elemérték-hivatkozás**: $x[i]$
 - **elemérték-módosítás**: $x[i] := 3$

Sorozat \rightarrow tömb

Példa

Specifikációban:

$n \in \mathbb{N}, x, y, z \in \mathbb{R}[1..n]$

– deklarációs példa (Be, Ki)

$z[1] = x[1] + y[1]$ és ...

– hivatkozási példa (Ef, Uf)

Algoritmusban:

$x, y, z : \text{Tömb}[1..n : \text{Valós}]$

– deklarációs példa

$z[1] := x[1] + y[1]$

– hivatkozási példa

Sorozat → tömb

SZÍNEK

0	zöld
1	piros
2	sárga
3	fehér
4	fekete

Példa – színes évek:

Az előbbi feladatpélda Szín halmaza a specifikációban egy szöveg konstansokból álló **sorozattal** ábrázolható:

```
SZÍNEKES[0..4]=  
("zöld","piros","sárga","fehér","fekete")
```

Az algoritmusban **tömb**bel reprezentálhatjuk:

```
Konstans SZÍNEK:Tömb[0..4:Szöveg]=  
("zöld","piros","sárga","fehér","fekete")
```


Elágazás helyett tömb

Specifikáció (végleges):

Be: év $\in \mathbb{N}$

Ki: $s \in S$,

$SZÍNEK \in S[0..4] =$

("zöld", "piros", "sárga", "fehér", "fekete")

Ef: $1984 \leq \text{év} \leq 2043$

Uf: $s = SZÍNEK[((\text{év} - 1984) \bmod 10) \div 2]$

A Szín halmaz
reprezentálása.

Specifikáció:

Be: év $\in \mathbb{N}$

Ki: $s \in Szín$,

$Szín = \{"zöld", "piros", "sárga", "fehér", "fekete"\}$

Ef: $1984 \leq \text{év} \leq 2043$

Uf: $((\text{év} - 1984) \bmod 10) \div 2 = 0 \rightarrow s = \text{"zöld"}$ és
 $((\text{év} - 1984) \bmod 10) \div 2 = 1 \rightarrow s = \text{"piros"}$ és
...

A Szín halmaz
reprezentációjához
alakított utófeltétel.

SZÍNEK

0	zöld
1	piros
2	sárga
3	fehér
4	fekete

Elágazás helyett tömb

Adatreprezentálás:

Változó

év:Egész

s:Szöveg

Konstans

SZÍNEK:Tömb[0..4:Szöveg]=
("zöld","piros","sárga","fehér","fekete")

Programparaméterek
deklarálása

Be: év ∈ ℕ
Ki: s ∈ S,
Színek ∈ S[0..4]=
("zöld","piros","sárga","fehér","fekete")

Algoritmus:

Uf: s = SZÍNEK[(((év-1984) mod 10) div 2)]

s := SZÍNEK[(((év-1984) mod 10) div 2)]

Tömb – algoritmus → kód

C#-ban a tömbök 0-tól indexelődnek.

1. ötlet: ne használjuk a 0. elemet!

Deklarációs példa:

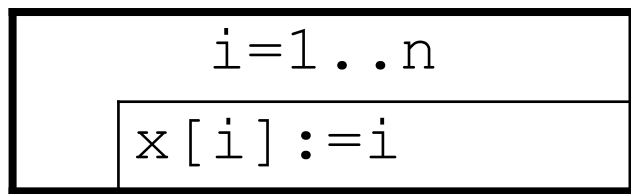
`x:Tömb[1..n:Valós]`

Algoritmus		Kód	
		0	?
1	a	1	a
2	b	2	b
3	c	3	c

C# kód:

```
float[] x=new float[n+1];
```

Algoritmus:



C# kód:

```
for (int i=1; i<=n; ++i) {  
    x[i]=i;  
}
```

Tömb – algoritmus → kód

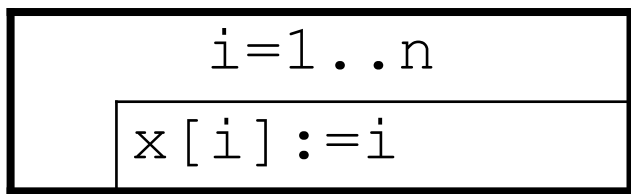
C#-ban a tömbök 0-tól indexelődnek.

2. ötlet: indexeltolás!

Deklarációs példa:

`x:Tömb[1..n:Valós]`

Algoritmus:



Algoritmus	Kód
1	a
2	b
3	c

0	a
1	b
2	c

C# kód:

```
float[] x=new float[n];
```

C# kód:

```
for (int i=1; i<=n; ++i) {  
    x[i-1]=i;  
}  
//----- vagy -----  
for (int i=1-1; i<=n-1; ++i) {  
    x[i]=i;  
}
```

Tömb – algoritmus → kód

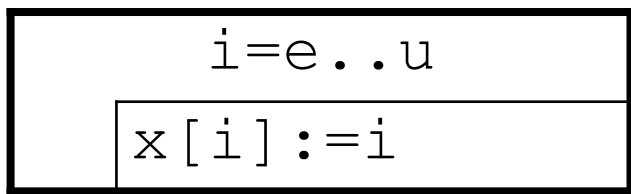
C#-ban a tömbök 0-tól indexelődnek.

3. ötlet: általános esetben!

Deklarációs példa:

`x:Tömb[e..u:Valós]`

Algoritmus:



Algoritmus		Kód	
e	a	0	a
e+1	b	1	b
u	c	2	c

C# kód:

```
float[] x=new float[u-e+1];
```

C# kód:

```
for (int i=e; i<=u; ++i) {  
    x[i-e]=i;  
}  
//----- vagy -----  
for (int i=e-e; i<=u-e; ++i) {  
    x[i]=i;  
}
```

Tömb – algoritmus → kód

C#-ban a tömbök 0-tól indexelődnek.

3. ötlet: példa

Deklarációs példa:

`x:Tömb[-1..10:Valós]`

Algoritmus:

$i = -1 \dots 10$
$x[i] := i + 5$

Algoritmus	Kód
-1	4
0	5
1	6

C# kód:

```
float[] x=new float[12];
```

C# kód:

```
for (int i=-1; i<=10; ++i) {  
    x[i+1]=i+5;  
}  
//----- vagy -----  
for (int i=0; i<=11; ++i) {  
    x[i]=i+(-1)+5;  
}
```

Konstans tömb

Algoritmus:

Konstans SZÍNEK:Tömb[0..4:Szöveg]=
("zöld", "piros", "sárga", "fehér", "fekete")

s:=SZÍNEK[((év-1984) mod 10) div 2]

Kód:

```
string[] SZINEK = new string[5]
    { "zöld", "piros", "sárga", "fehér", "fekete" };
// vagy
string[] SZINEK =
    { "zöld", "piros", "sárga", "fehér", "fekete" };

int ev = 2000;
string s = SZINEK[((ev - 1984) % 10) / 2];
```

Konstans tömb alkalmazása

Feladat:

Írj programot, amely egy 1 és 99 közötti számot betűkkel ír ki!

Specifikáció:

Be: $n \in \mathbb{N}$,

$EGYES \in S[0..9] = ("", "egy", \dots, "kilenc"),$
 $TIZES \in S[0..9] = ("", "tizen", \dots, "kilencven")$

Ki: $s \in S$

Ef: $1 \leq n \leq 99$

Uf: $n=10 \rightarrow s="tíz"$ és
 $n=20 \rightarrow s="húsz"$ és
 $(n \neq 10 \text{ és } n \neq 20) \rightarrow$

$s = TIZES(n \text{ div } 10) + EGYES(n \text{ mod } 10)$

Leglogikusabb helyre téve.
Az algoritmus szempontjából
„adottság”, azaz bemenet...

Konstans tömb alkalmazása

Algoritmus:

Változó n :Egész

Konstans $EGYES$:Tömb[0..9:Szöveg]=
("","egy",...,"kilenc")

$TIZES$:Tömb[0..9:Szöveg]=
("","tizen",...,"kilencven")

Változó s :Szöveg

Be: $n \in \mathbb{N}$,
 $EGYES[0..9] = ("","egy",\dots,"kilenc")$,
 $TIZES[0..9] = ("","tizen",\dots,"kilencven")$

Ki: $s \in S$

Uf: $n=10 \rightarrow s="tíz"$ és
 $n=20 \rightarrow s="húsz"$ és
 $(n \neq 10 \text{ és } n \neq 20) \rightarrow$
 $s = TIZES(n \text{ div } 10) + EGYES(n \text{ mod } 10)$

$n=10$	$n=20$	$n \neq 10 \text{ és } n \neq 20$
$s := "tíz"$	$s := "húsz"$	$s := TIZES[n \text{ div } 10] +$ $EGYES[n \text{ mod } 10]$

Konstans tömb alkalmazása

Feladat:

Írj programot, amely egy hónapnévhez a sorszámát rendeli!

Specifikáció:

Be: $h \in S$,

$HÓNÉV \in S[1..12] =$
("január", ..., "december")

Ki: $s \in N$

Ef: $h \in HÓNÉV$

Uf: $1 \leq s \leq 12$ és $HÓNÉV[s] = h$

Példa:

$h=9 \rightarrow s="szeptember"$

	HÓNÉV
1	január
2	február
3	március
4	április
5	május
6	június
7	július
8	augusztus
9	szeptember
10	október
11	november
12	december

Konstans tömb alkalmazása

Algoritmus:

Változó h :Szöveg, s :Egész

Konstans $HÓNÉV:Tömb[1..12: Szöveg]=$
("január", ..., "december")



Be: $h \in S$,
 $HÓNÉV[1..12]=$
("január", ..., "december")
Ki: $s \in \mathbb{N}$
Ef: $h \in HÓNÉV$
Uf: $1 \leq s \leq 12$ és $HÓNÉV[s]=h$

Kérdés: mi lenne, ha az előfeltétel nem teljesülne?
Futási hiba? Végtelen ciklus?

Konstans tömb – mit tárolunk?

Feladat:

Egy nap a nem szökőév hányadik napja?

Specifikáció₁:

Be: $h \in \mathbb{N}$, $n \in \mathbb{N}$,

$HÓ \in \mathbb{N}[1..12] = (31, 28, 31, \dots, 31)$

Ki: $s \in \mathbb{S}$

Ef: $1 \leq h \leq 12$ és $1 \leq n \leq HÓ[h]$

Uf: $s = \text{SZUM}(i=1..h-1, HÓ[i]) + n$

Példa:
 $h=9, n=18 \rightarrow s=261$

	HÓ
1	31
2	28
3	31
4	30
5	31
6	30
7	31
8	31
9	30
10	31
11	30
12	31

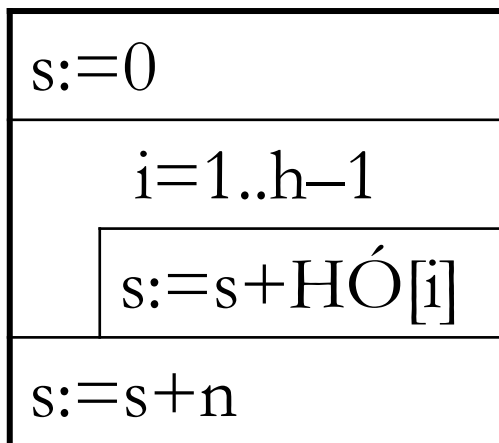
Konstans tömb – mit tárolunk?

Algoritmus:

Változó $h, n, s : \text{Egész}$

Konstans $\text{HÓ} : \text{Tömb}[1..12 : \text{Egész}] = (31, 28, 31, \dots, 31)$

Programparaméterek
deklarálása



Változó
 $i : \text{Egész}$

Be: $h \in \mathbb{N}, n \in \mathbb{N},$
 $\text{HÓ} \in [1..12] = (31, 28, 31, \dots, 31)$
Ki: $s \in \mathbb{S}$
Ef: $1 \leq h \leq 12$ és $1 \leq n \leq \text{HÓ}[h]$
Uf: $s = \text{SZUM}(i=1..h-1, \text{HÓ}[i]) + n$

Lokális változó
deklarálása

Megjegyzés: szökőév esetén $h \geq 3$ esetén s -et 1-gyel meg kellene növelni! (És az előfeltétel is módosul.)

Konstans tömb – mit tárolunk?

Egy másik megoldás:

Tároljuk minden hónapra, hogy az előző hónapokban összesen hány nap van!

Specifikáció₂:

Be: $h \in \mathbb{N}$, $n \in \mathbb{N}$,

$HÓ \in S[1..12] = (0, 31, 59, 90, \dots, 334)$

Uf: $s = HÓ[h] + n$

Példa:
 $h=9, n=18 \rightarrow s=261$

	HÓ
1	0
2	31
3	59
4	90
5	120
6	151
7	181
8	212
9	243
10	273
11	304
12	334

Kérdés: Ez jobb megoldás? Mi lesz az előfeltétellel?

Mátrix

- **Tömb:** azonos funkciójú elemek egyirányú sorozata
 - egy index egy elem kiválasztásához, pl. $x[i]$
- **Mátrix:** azonos funkciójú elemek kétirányú sorozata

	x
1	-4
2	2
3	5

- két index egy elem kiválasztásához, pl. $x[i, j]$
- specifikáció: $n \in \mathbb{N}$, $m \in \mathbb{N}$, $x \in \mathbb{Z}[1..n, 1..m]$
- algoritmus: $x: \text{Tömb}[1..n, 1..m: \text{Egész}]$
- kód: `int[,] x = new int[n, m];`

x	1	2	3
1	-4	3	2
2	2	10	11
3	5	4	-5

Összefoglalás



Összefoglalás

- Adat
 - egy-szerű: elemi
 - több különböző: rekord
 - több egyforma: tömb
- Vezérlési szerkezetek
 - Szekvencia: és
 - Elágazás: $->$
 - Ciklus: \forall, \exists, Σ
- Feladatmegoldás
 1. Példa
 2. Specifikáció (\leftarrow példa)
 1. Adatok (Be, Ki)
 2. Megszorítás ($Ef \rightarrow Be$)
 3. Összefüggés (Uf)
 3. Adat \rightarrow Változó
 4. Algoritmus ($\leftarrow Uf$)
 5. Kód ($\leftarrow Spec + Alg$)

Ellenőrző kérdések



Ellenőrző kérdések

- Mikor érdemes rekordtípust használni?
- Mikor használjunk feltételes és mikor használjunk számlálós ciklust?
- Mi a „filozófiai különbség” a számlálós és a feltételes ciklus között?
- Mi a különbség az előtesztelős és a hátultesztelős ciklus között?
- Hogyan néznek ki pszeudokódjai a feltételes ciklusoknak? Add meg az ezeknek megfelelő C# kóddarabokat is!
- Hogyan néznek ki a pszeudokódjai a számlálós ciklusnak és az elágazásnak? Add meg az ezek megfelelő C# kóddarabjait is!