

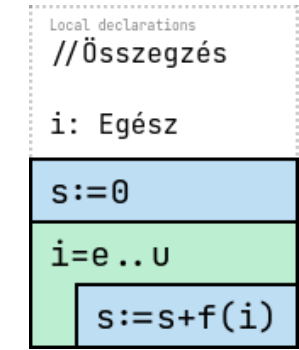
# Programozás "PUSKA"

Type

| Specifikáció                               | Algoritmus Stílus                    | C# Kód Snippet                                | Magyarázat   |
|--|--------------------------------------|---|--|
| $a \in \mathbb{N}$                         | a: Természetes                       | int a;  | a egy darab természetes szám.  |
| $b \in \mathbb{Z}$                         | b: Egész                             | int b;  | b egy darab egész szám.  |
| $c \in \mathbb{R}$                         | c: Valós                             | double c;                                     | c egy valós szám (tizedestört).  |
| $d \in \mathbb{L}$                         | d: Logikai                           | bool d;                                       | d egy logikai érték (igaz/hamis).  |
| $e \in \mathbb{S}$                         | e: Szöveg                            | string e;                                     | e egy szöveg (karakterlánc).   |
| $f \in \mathbb{C}$                         | f: Karakter                          | char f;                                       | f egy darab karakter.  |
| $iv1 \in [1..3]$                           | iv1: [1..3]                          | int iv1;                                      | iv1 egy egész szám (az [1..3] korlátot kódban kell ellenőrizni).         |
| $x1 \in \mathbb{N}[1..3]$                  | x1: Tömb(1..3, Természetes)          | int[] x1 = new int[3];                        | x1 egy 3 elemű tömb (fix méretű).  |
| $n \in \mathbb{N}, x \in \mathbb{Z}[1..n]$ | n: Természetesx: Tömb(1..n, Egész)   | int n;int[] x;x = new int[n];                 | n egy szám, x pedig egy n elemű tömb (futásidőben dől el).               |
| $x3 \in \mathbb{N}[1..]$                   | x3: Tömb(1.., Természetes)           | List<int> x3;x3 = new List<int>();            | x3 egy dinamikus méretű tömb (lista), aminek nem tudjuk előre a méretét. |
| $m1 \in \mathbb{Z}[1..8,1..8]$             | m1: Tömb(1..8, 1..8, Egész)          | int[,] m1;m1 = new int[8, 8];                 | m1 egy 8×8-as mátrix (fix méretű).                                       |
| $m2 \in \mathbb{Z}[1..n,1..n]$             | m2: Tömb(1..n, 1..n, Egész)          | int[,] m2;m2 = new int[n, n];                 | m2 egy n x n-es mátrix (futásidőben dől el).                             |
| $r1 \in \text{név:S x jegy:N}$             | r1: Rekord(név: Szöveg, jegy: Term.) | EgyediRekord r1;                              | r1 egy rekord/struktúra típusú változó.                                  |
| Hallgató=(...)                             | Típus Hallgató = Rekord(...)         | struct Hallgato { ... }                       | Hallgató néven egy struktúra típust definiálunk.                         |
| $h2 \in \text{Hallgató}[1..]$              | h2: Tömb(1.., Hallgató)              | List<Hallgato> h2;h2 = new List<Hallgato> (); | h2 egy dinamikus lista, ami Hallgato struktúrákat tárol.                 |

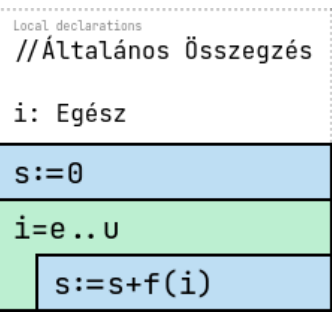
Összegzés

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
Ki:  $s \in \mathbb{H}$   
Ef: -  
Uf:  $s = \text{SZUMMA}(i = e..u, f(i))$



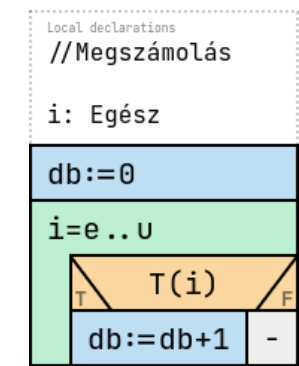
Általános Összegzés

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
Ki:  $s \in \mathbb{H}$   
Ef: -  
Uf:  $s = \text{SZUMMA}(i = e..u, f(i), 0, +)$



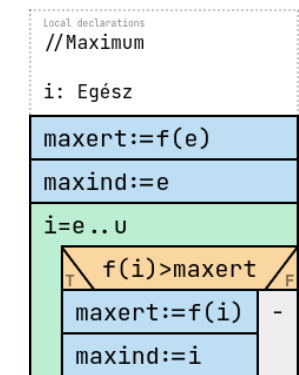
Megszámolás

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
Ki:  $db \in \mathbb{N}$   
Ef: -  
Uf:  $db = \text{SZUMMA}(i = e..u, 1, T(i))$   
Rövidítve:  
Uf:  $db = \text{DARAB}(i = e..u, T(i))$



Maximum

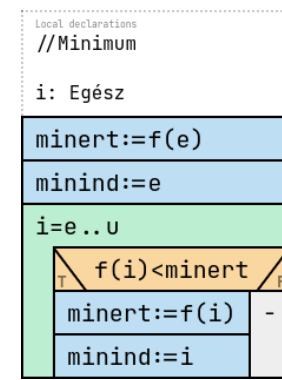
Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
Ki:  $\text{maxind} \in \mathbb{Z}, \text{maxért} \in \mathbb{H}$   
Ef:  $e \leq u$   
Uf:  $\text{maxind} \in [e..u]$  és  
 $\forall i \in [e..u]: (f(\text{maxind}) \geq f(i))$  és  
 $\text{maxért} = f(\text{maxind})$   
Rövidítve:  
Uf:  $(\text{maxind}, \text{maxért}) = \text{MAX}(i = e..u, f(i))$





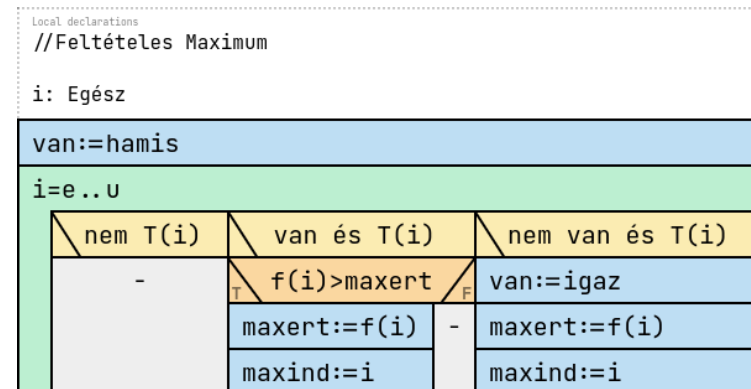
## Minimum

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $\text{minind} \in \mathbb{Z}, \text{minért} \in \mathbb{H}$   
 Ef:  $e \leq u$   
 Uf:  $\text{minind} \in [e..u]$  és  
 $\forall i \in [e..u]: (f(\text{minind}) \leq f(i))$  és  
 $\text{minért} = f(\text{minind})$   
 Uf:  $(\text{minind}, \text{minért}) = \text{MIN}(i=e..u, f(i))$



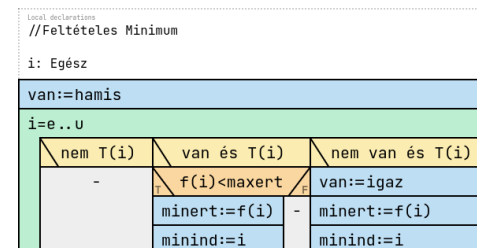
## Feltételes Maximum

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $\text{van} \in \mathbb{L}, \text{maxind} \in \mathbb{Z}, \text{maxért} \in \mathbb{H}$   
 Ef: -  
 Uf:  $\text{van} = \exists i \in [e..u]: (T(i))$  és  
 $\text{van} \rightarrow (\text{maxind} \in [e..u] \text{ és } \text{maxért} = f(\text{maxind}) \text{ és } T(\text{maxind}) \text{ és } \forall i \in [e..u]: (T(i) \rightarrow \text{maxért} \geq f(i)))$   
 Uf:  $(\text{van}, \text{maxind}, \text{maxért}) = \text{FELTMAX}(i=e..u, f(i), T(i))$



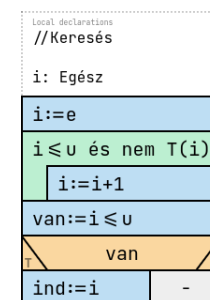
## Feltételes Minimum

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $\text{van} \in \mathbb{L}, \text{minind} \in \mathbb{Z}, \text{minert} \in \mathbb{H}$   
 Ef: -  
 Uf:  $\text{van} = \exists i \in [e..u]: (T(i))$  és  
 $\text{van} \rightarrow (\text{minind} \in [e..u] \text{ és } \text{maxért} = f(\text{minind}) \text{ és } T(\text{minind}) \text{ és } \forall i \in [e..u]: (T(i) \rightarrow \text{minert} \geq f(i)))$   
 Uf:  $(\text{van}, \text{minind}, \text{minert}) = \text{FELTMIN}(i=e..u, f(i), T(i))$



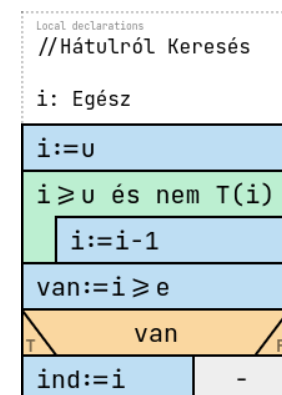
## Keresés

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $\text{van} \in \mathbb{L}, \text{ind} \in \mathbb{Z}$   
 Ef: -  
 Uf:  $\text{van} = \exists i \in [e..u]: (T(i))$  és  
 $\text{van} \rightarrow (\text{ind} \in [e..u] \text{ és } T(\text{ind}) \text{ és } \forall i \in [e..\text{ind}-1]: (\text{nem } T(i)))$   
 Rövidítve:  
 Uf:  $(\text{van}, \text{ind}) = \text{KERES}(i=e..u, T(i))$



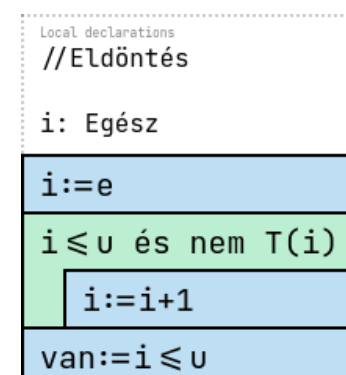
## Hátulról keresés

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $\text{van} \in \mathbb{L}, \text{ind} \in \mathbb{Z}$   
 Ef: -  
 Uf:  $(\text{van}, -\text{ind}) = \text{KERES}(i=u..-e, T(-i))$



## Eldöntés

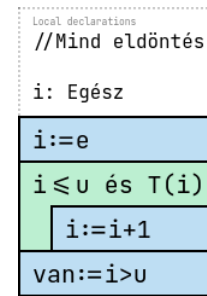
Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $\text{van} \in \mathbb{L}$   
 Ef: -  
 Uf:  $\text{van} = \exists i \in [e..u]: (T(i))$   
 Rövidítve:  
 Uf:  $\text{van} = \text{VAN}(i=e..u, T(i))$





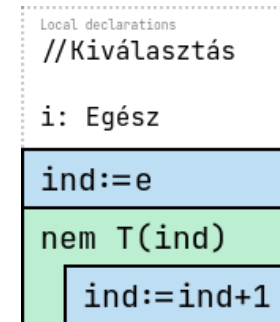
## Mind/optimista eldöntés

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $van \in \mathbb{L}$   
 Ef: -  
 Uf:  $van = \forall i \in [e..u]: (T(i))$   
 Rövidítve:  
 Uf:  $van = VAN(i=e..u, T(i))$



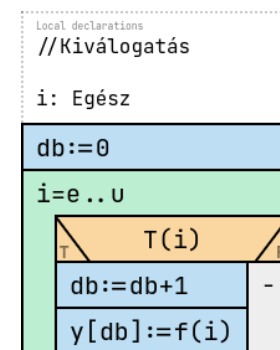
## Kiválasztás

Be:  $e \in \mathbb{Z}$   
 Ki:  $ind \in \mathbb{Z}$   
 Ef:  $\exists i \in [e..\infty]: (T(i))$   
 Uf:  $ind > e$  és  $T(ind)$  és  
 $\forall i \in [e..ind-1]: (nem T(i))$   
 Rövidítve:  
 Uf:  $ind = KIVÁLASZT(i > e, T(i))$



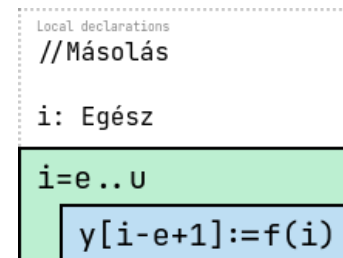
## Kiválogatás

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $db \in \mathbb{N}, y \in H[1..db]$   
 Ef: -  
 Uf:  $db = DARAB(i=e..u, T(i))$  és  
 $\forall i \in [1..db]: (\exists j \in [e..u]: T(j) \text{ és } y[i] = f(j))$   
 és  $y \subseteq (f(e), f(e+1), \dots, f(u))$   
 Rövidítve:  
 Uf:  $(db, y) = KIVÁLOGAT(i=e..u, T(i), f(i))$



## Másolás

Be:  $e \in \mathbb{Z}, u \in \mathbb{Z}$   
 Ki:  $y \in H[1..u-e+1]$   
 Ef: -  
 Uf:  $\forall i \in [e..u]: (y[i-e+1] = f(i))$   
 Rövidítve:  
 Uf:  $y = MÁSOL(i=e..u, f(i))$



```

using System;
using System.Collections.Generic; // A Dynamic List-hez kell

namespace Puska
{
    // (Hallgató=(név:S x jegy:N))
    public struct Hallgato
    {
        public string nev;
        public int jegy;
    }

    class Program
    {
        static void Main(string[] args)
        {
            #region Deklaráció

            // 1. Elemi típusok (pl.  $a \in \mathbb{N}$ ,  $e \in \mathbb{S}$ )
            int a;
            string e;

            // 2. Tömb (fix méretű, pl.  $x \in \mathbb{Z}[1..n]$ )
            int n; // A tömb mérete
            int[] x;

            // 3. Lista (dinamikus méretű, pl.  $x3 \in \mathbb{N}[1..]$ )
            List<int> x3;

            // 4. Struktúra (egyedi adat, pl.  $r1 \in \text{név:S x jegy:N}$ )
            Hallgato r1; // Létrehozunk egy 'r1' változót a 'Hallgato' tervrajz alapján

            // 5. Struktúrák listája (pl.  $h2 \in \text{Hallgató}[1..]$ )
            List<Hallgato> h2;

            #endregion

            #region Beolvasás (és Inicializálás)

            // Értékadás
            a = 10;
            e = "Hello";

            // Beolvasás int ként
            n = int.Parse(Console.ReadLine());

            // Tömb létrehozása 'n' alapján
            x = new int[n];

            // Listák létrehozása (üresen)
            x3 = new List<int>();
            h2 = new List<Hallgato>();

            // Struktúra mezőinek feltöltése
            r1.nev = "Minta Béla";
            r1.jegy = 5;

            #endregion

            #region Feldgozás
            // Adunk a listákhoz 1-1 elemet
            x3.Add(100);

            // Létrehozunk egy új hallgatót és hozzáadjuk a listához
            Hallgato ujHallgato;
            ujHallgato.nev = "Nagy Anna";
            ujHallgato.jegy = 4;
            h2.Add(ujHallgato);

            #endregion

            #region Kiírás
            Console.WriteLine($"''a' (int) értéke: {a}");
            Console.WriteLine($"''e' (string) értéke: {e}");
            Console.WriteLine($"''r1' (struct) neve: {r1.nev}");
            Console.WriteLine($"''h2' (lista) első elemének neve: {h2[0].nev}");
            #endregion
        }
    }
}

```