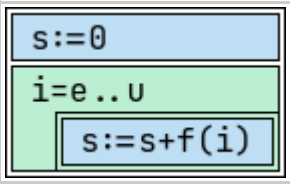
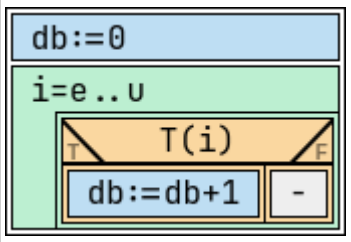
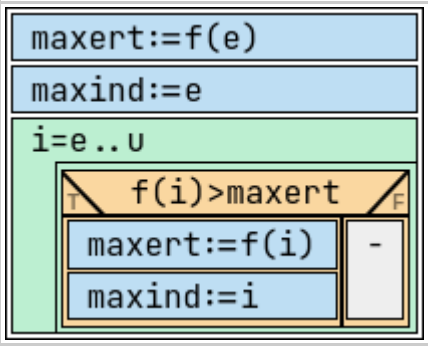
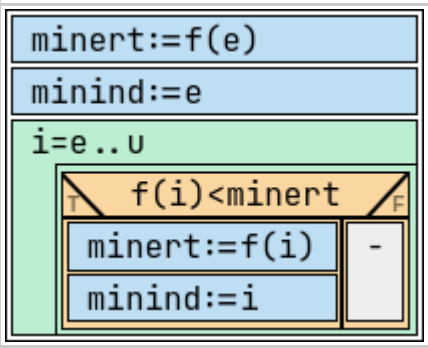
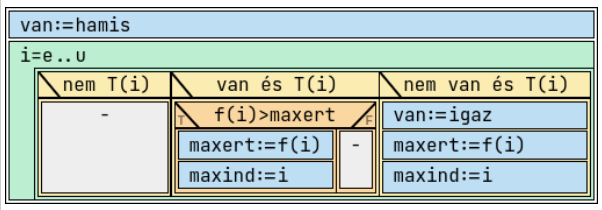
Minta neve	Specifikáció	Algoritmus	C# kód

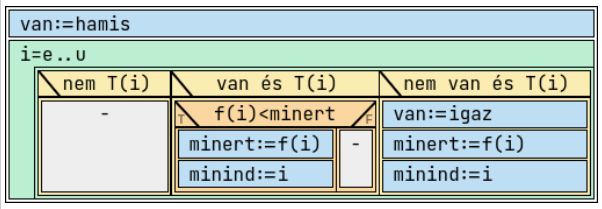
Összegzés	`osszeg=SZUM(i=e..u, f(i))`		public int Osszegzes(int[] T)
			{
			int osszeg = 0;
			for (int i = 0; i < T.Length; i++)
			{
			osszeg = osszeg + T[i];
			}
			return osszeg;
			}

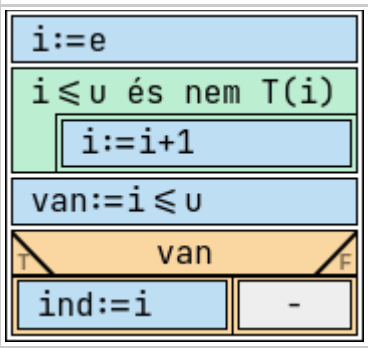
Megszámolás	`db=MEGSZAMOL(i=e..u, T(i)) <=> db=SZUM(i=e..n, 1, T(i))`		public int Megszamolas(int[] T)
			{
			int db = 0;
			for (int i = 0; i < T.Length; i++)
			{
			// A "T[i] % 2 == 0" a feltétel (pl. páros)
			if (T[i] % 2 == 0)
			{
			db++;
			}
			}
			return db;
			}

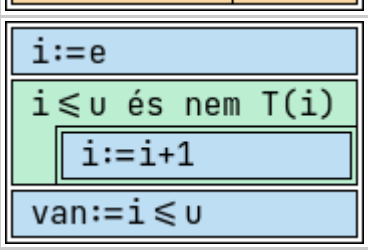
Maximumkiválasztás	`(maxind, maxert)=MAX(i=e..u, f(i))`		public (int maxInd, int maxErtek) Maximumkivalasztas(int[] T)
			{
			int maxInd = 0;
			int maxErtek = T[0];
			
			for (int i = 1; i < T.Length; i++)
			{
			if (T[i] > maxErtek)
			{
			maxErtek = T[i];
			maxInd = i;
			}
			}
			return (maxInd, maxErtek);
			}

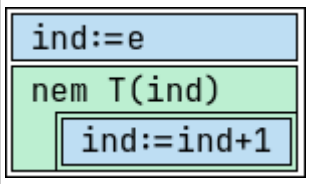
Minimumkiválasztás	`(minind, minert)=MIN(i=e..u, f(i))`		public (int minInd, int minErtek) Minimumkivalasztas(int[] T)
			{
			int minInd = 0;
			int minErtek = T[0];
			
			for (int i = 1; i < T.Length; i++)
			{
			if (T[i] < minErtek)
			{
			minErtek = T[i];
			minInd = i;
			}
			}
			return (minInd, minErtek);
			}

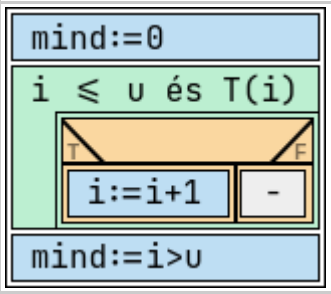
Feltételes maximumkeresés	`(van,maxind,maxert)=FELTMAX(i=e..u, f(i), T(i))`		public (bool van, int maxInd, int maxErtek) FeltetelesMaximum(int[] T)
			{
			bool van = false;
			int maxInd = -1;
			int maxErtek = 0; // Kezdőérték nem számít, ha nincs találat
			
			for (int i = 0; i < T.Length; i++)
			{
			// Feltétel: T[i] % 2 == 0 (Páros)
			if (T[i] % 2 == 0)
			{
			if (!van || T[i] > maxErtek)
			{
			maxErtek = T[i];
			maxInd = i;
			van = true;
			}
			}
			}
			return (van, maxInd, maxErtek);
			}

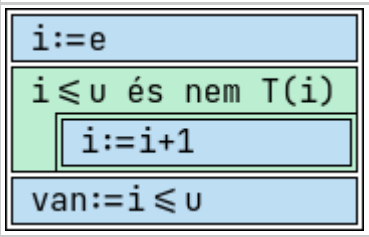
Feltételes minimumkeresés	`(van,minind,minert)=FELTMIN(i=e..u, f(i), T(i))`		public (bool van, int minInd, int minErtek) FeltetelesMinimum(int[] T)
			{
			bool van = false;
			int minInd = -1;
			int minErtek = 0;
			
			for (int i = 0; i < T.Length; i++)
			{
			// Feltétel (pl. páros)
			if (T[i] % 2 == 0)
			{
			if (!van || T[i] < minErtek)
			{
			minErtek = T[i];
			minInd = i;
			van = true;
			}
			}
			}
			return (van, minInd, minErtek);
			}

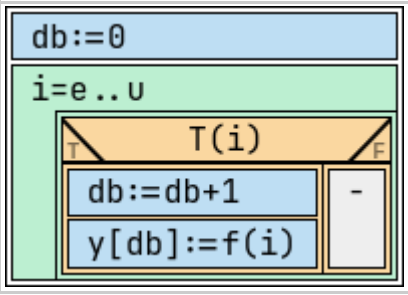
Keresés	`(van, ind)=KERES(i=i..u, T(i))`		public (bool van, int ind) Kereses(int[] T, int keresettSzam)
			{
			int i = 0;
			while (i < T.Length && T[i] != keresettSzam)
			{
			i++;
			}
			
			bool van = (i < T.Length);
			return (van, van ? i : -1);
			}

Eldöntés	`van=VAN(i=e..u, T(i))`		public bool Eldontes(int[] T)
			{
			int i = 0;
			// Feltétel: pl. nagyobb mint 10
			while (i < T.Length && !(T[i] > 10))
			{
			i++;
			}
			return i < T.Length; // Igaz, ha találtunk ilyet
			}

Kiválasztás	`ind=KIVÁLASZT(i>=e, T(i))`		public bool MindEldontes(int[] T)
			{
			int i = 0;
			// Feltétel: pl. pozitív szám (T[i] > 0)
			while (i < T.Length && T[i] > 0)
			{
			i++;
			}
			return i == T.Length; // Igaz, ha végigértünk a cikluson
			}

Mind eldöntés	`mind=MIND(i=e..u, T(i))`		public int Kivalasztas(int[] T, int keresettSzam)
			{
			int i = 0;
			// Itt nem kell "i < N", mert BIZTOS, hogy benne van
			while (T[i] != keresettSzam)
			{
			i++;
			}
			return i;
			}

Másolás	`y=MÁSOL(i=e..u, f(i))`		public int[] Masolas(int[] T)
			{
			int[] Y = new int[T.Length];
			for (int i = 0; i < T.Length; i++)
			{
			Y[i] = T[i] * 2; // f(x) transzformáció
			}
			return Y;
			}

Kiválogatás	`(db,y)=KIVÁLOGAT(i=e..u, T(i), f(i))`		public List<int> Kivalogatas(int[] T)
			{
			List<int> Y = new List<int>();
			for (int i = 0; i < T.Length; i++)
			{
			if (T[i] > 5) // Feltétel
			{
			Y.Add(T[i]);
			}
			}
			return Y; // Vagy Y.ToArray(), ha tömb kell
			}

Specifikáci	Algoritmus	C#
a∈N	a: Természetes	int a;
b∈Z	b: Egész	int b;
c∈R	c: Valós	double c;
d∈L	d: Logikai	bool d;
e∈S	e: Szöveg	string e;
f∈C	f: Karakter	char f;
iv1∈[1..3]	iv1: [1..3]	int iv1;
x1∈N[1..3]	x1: Tömb(1..3, Természetes)	int[] x1 = new int[3];
n∈N, x∈Z[1..n]	n: Természetesx: Tömb(1..n, Egész)	int n;int[] x;x = new int[n];
x3∈N[1..]	x3: Tömb(1.., Természetes)	List<int> x3;x3 = new List<int>();
m1∈Z[1..8,1..8]	m1: Tömb(1..8, 1..8, Egész)	int[,] m1;m1 = new int[8, 8];
m2∈Z[1..n,1..n]	m2: Tömb(1..n, 1..n, Egész)	int[,] m2;m2 = new int[n, n];
r1∈név:S x jegy:N	r1: Rekord(név: Szöveg, jegy: Term.)	EgyediRekord r1;
Hallgató=(...)	Típus Hallgató = Rekord(...)	struct Hallgato { ... }
h2∈Hallgató[1..]	h2: Tömb(1.., Hallgató)	List<Hallgato> h2;h2 = new List<Hallgato>();