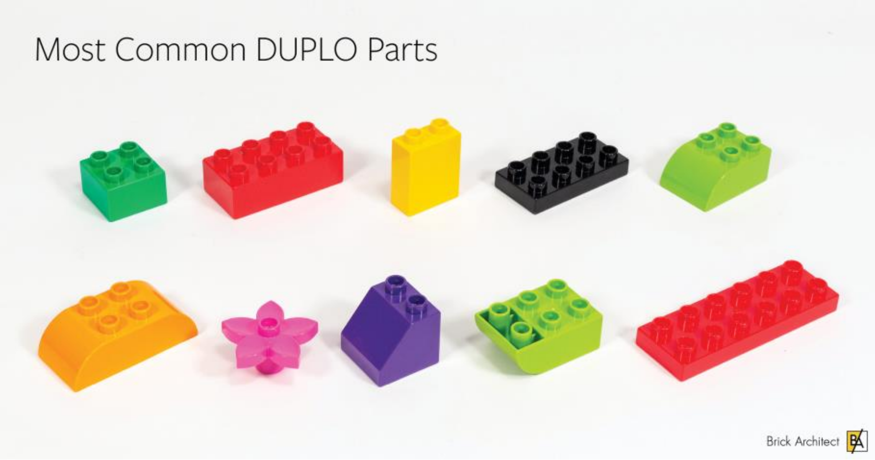
ELTE | IK

PROGRAMOZÁS

11. előadás

Horváth Győző, Horváth Gyula, Szlávi Péter

Programozási minták



1. Összegzés
2. Megszámolás
3. Maximumkiválasztás
   a. Minimumkiválasztás
4. Feltételes maximumkeresés
5. Keresés
6. Eldöntés
   a. Mind eldöntés
7. Kiválasztás
8. Másolás
9. Kiválogatás

Tanulságos feladatok

Függvény két visszatérési értékkel

Feladat:

Egy térképet egy négyzetrácsban tárolunk. Minden rácspontban egy egész szám mondja meg az ott lévő pont tengerszint feletti magasságát. Határozd meg a térképen a legészaknyugatibb pontot, ahol tó található!

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

1. változat

Feladat:

Egy egész számokat tartalmazó mátrixban melyik az a sor és oszlop, ahol sorfolytonosan először 0 szerepel?

Másképpen:

Egy egész számokat tartalmazó mátrixban melyik az a sor, amelyben van 0, és ebben hol fordul elő először?

Azaz:

Keresésben eldöntés és kiválasztás

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

1. változat

Feladat:

Egy egész számokat tartalmazó mátrixban melyik az a sor, amelyben van 0, és ebben hol fordul elő először?

Specifikáció:

```
Be: n∈N, m∈N, mátrix∈Z[1..n,1..m]
Ki: van∈L, sind∈N, oind∈N
Fv: vannulla:N->L, vannulla(i)=VAN(j=1..m, mátrix[i,j]=0)
Ef: -
Uf: (van,sind)=KERES(i=1..n, vannulla(i)) és
    van -> oind=KIVÁLASZT(j>=1, mátrix[sind,j]=0)
```

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

1. változat

Specifikáció:

```
Be: n∈N, m∈N, mátrix∈Z[1..n,1..m]
Ki: van∈L, sind∈N, oind∈N
Fv: vannulla:N->L, vannulla(i)=VAN(j=1..m, mátrix[i,j]=0)
Ef: -
Uf: (van,sind)=KERES(i=1..n, vannulla(i)) és
    van -> oind=KIVÁLASZT(j>=1, mátrix[sind,j]=0)
```

Keresés

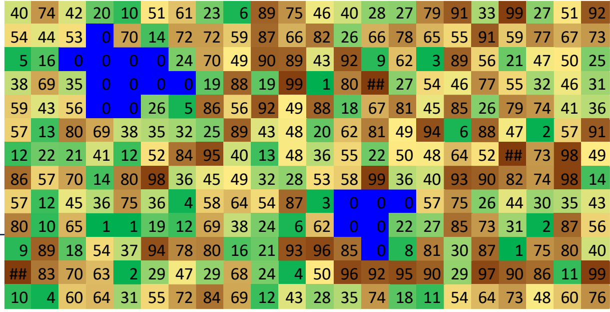
```
ind    ~ sind
e..u   ~ 1..n
T(i)   ~ vannulla(i)
```

Eldöntés (vannulla)

```
i      ~ j
e..u   ~ 1..m
T(i)   ~ mátrix[i,j]=0
```

Kiválasztás

```
ind    ~ oind
i      ~ j
e      ~ 1
T(i)   ~ mátrix[sind,j]=0
```



1. változat

Uf: (van,sind)=**KERES**(i=1..n, vannulla(i)) és
van -> oind=**KIVÁLASZT**(j>=1, mátrix[sind,j]=0)

Keresés

```
ind    ~ sind
e..u   ~ 1..n
T(i)   ~ vannulla(i)
```

Eldöntés (vannulla)

```
i      ~ j
e..u   ~ 1..m
T(i)   ~ mátrix[i,j]=0
```

Kiválasztás

```
ind    ~ oind
i      ~ j
e      ~ 1
T(i)   ~ mátrix[sind,j]=0
```

sind:=1

sind<=n és nem vannulla(sind)

  sind:=sind+1

van:=sind<=n

van

true

oind:=1

nem mátrix[sind,j]=0

  oind:=oind+1

false

-

vannulla(i:Egész):Logikai

Vált j:Egész, van:Logikai

j:=1

j<=m és nem mátrix[i,j]=0

  j:=j+1

van:=j<=m

vannulla:=van

2. változat

Feladat:

Egy egész számokat tartalmazó mátrixban melyik az a sor és oszlop, ahol sorfolytonosan először 0 szerepel?

Másképpen:

Egy egész számokat tartalmazó mátrixban melyik az a sor, amelyben van 0, és ebben hol fordul elő először?

Azaz:

Keresésben keresés és keresés

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

2. változat

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

Feladat:

Egy egész számokat tartalmazó mátrixban melyik az a sor, amelyben van 0, és ebben hol fordul elő először?

```
keresnulla(i)=(van,ind):((van,ind)=KERES(j=1..m, mátrix[i,j]>0))
```

Specifikáció:

```
Be: n∈N, m∈N, mátrix∈Z[1..n,1..m]
Ki: van∈L, sind∈N, oind∈N
Fv: keresnulla:N->L x N,
    keresnulla(i)=KERES(j=1..m, mátrix[i,j]=0)
Ef: -
Uf: (van,sind)=KERES(i=1..n, keresnulla(i).van) és
    van -> oind=keresnulla(sind).ind
```

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

2. változat

Specifikáció:

```
Be: n∈N, m∈N, mátrix∈Z[1..n,1..m]
Ki: van∈L, sind∈N, oind∈N
Fv: keresnulla:N->L x N,
    keresnulla(i)=KERES(j=1..m, mátrix[i,j]=0)
Ef: -
Uf: (van,sind)=KERES(i=1..n, keresnulla(i).van) és
    van -> oind=keresnulla(sind).ind
```

Keresés

```
ind  ~ sind
e..u ~ 1..n
T(i) ~ keresnulla(i).van
```

Keresés (keresnulla)

```
i     ~ j
e..u  ~ 1..m
T(i)  ~ mátrix[i,j]=0
```

2. változat

40	74	42	20	10	51	61	23	6	89	75	46	40	28	27	79	91	33	99	27	51	92
54	44	53	0	70	14	72	72	59	87	66	82	26	66	78	65	55	91	59	77	67	73
5	16	0	0	0	0	24	70	49	90	89	43	92	9	62	3	89	56	21	47	50	25
38	69	35	0	0	0	0	19	88	19	99	1	80	##	27	54	46	77	55	32	46	31
59	43	56	0	0	26	5	86	56	92	49	88	18	67	81	45	85	26	79	74	41	36
57	13	80	69	38	35	32	25	89	43	48	20	62	81	49	94	6	88	47	2	57	91
12	22	21	41	12	52	84	95	40	13	48	36	55	22	50	48	64	52	##	73	98	49
86	57	70	14	80	98	36	45	49	32	28	53	58	99	36	40	93	90	82	74	98	14
57	12	45	36	75	36	4	58	64	54	87	3	0	0	0	57	75	26	44	30	35	43
80	10	65	1	1	19	12	69	38	24	6	62	0	0	22	27	85	73	31	2	87	56
9	89	18	54	37	94	78	80	16	21	93	96	85	0	8	81	30	87	1	75	80	40
##	83	70	63	2	29	47	29	68	24	4	50	96	92	95	90	29	97	90	86	11	99
10	4	60	64	31	55	72	84	69	12	43	28	35	74	18	11	54	64	73	48	60	76

```
Uf: (van,sind)=KERES(i=1..n, keresnulla(i).van) és
    van -> oind=keresnulla(sind).ind
```

Keresés

```
ind   ~ sind
e..u  ~ 1..n
T(i)  ~ keresnulla(i).van
```

Keresés (keresnulla)

```
i     ~ j
e..u  ~ 1..m
T(i)  ~ mátrix[i,j]=0
```

sind:=1

sind<=n és nem keresnulla(sind).van

sind:=sind+1

van:=sind<=n

van

true | false

oind:=keresnulla(sind).ind | -

keresnulla(i:Egész):(Logikai,Egész)

Vált j, ind:Egész, van:Logikai

ind:=1

ind<=m és nem mátrix[i,ind]=0

ind:=ind+1

van:=ind<=m

vannulla:=(van,ind)

Maximumkiválasztás vs feltételes maximumkeresés

Feladat:

Egy gyártó bekéri egy üzlettől, hogy egy időszakon belül, amikor nyitva volt, hány darab fogyott egy adott termékből. Mennyi volt a legtöbb eladott darabszám?

5	3	0	0	2	1	7	8	4	3

Elemzés

5	3	0	0	2	1	7	8	4	3

Feladat:

Egy gyártó bekéri egy üzlettől, hogy egy időszakon belül, amikor nyitva volt, hány darab fogyott egy adott termékből. Mennyi volt a legtöbb eladott darabszám?

Maximumkiválasztás!

Specifikáció:

```
Be: n∈N, db∈N[1..n]
Ki: max∈N
Ef: -
Uf: (,max)=MAX(i=1..n, db[i])
```

Biztos???

Ha a sorozat lehet üres is, akkor nem biztos, hogy van maximális elem
→ feltételes maximumkeresés

1. változat

Feladat:

Egy gyártó bekéri egy üzlettől, hogy egy időszakon belül, amikor nyitva volt, hány darab fogyott egy adott termékből. Mennyi volt a legtöbb eladott darabszám?

Maximumkiválasztás!

Specifikáció:

```
Be: n∈N, db∈N[1..n]
Ki: van∈L, max∈N
Ef: -
Uf: van = n>0 és
    van -> (,max)=MAX(i=1..n, db[i])
```

2. változat

Feladat:

Egy gyártó bekéri egy üzlettől, hogy egy időszakon belül, amikor nyitva volt, hány darab fogyott egy adott termékből. Mennyi volt a legtöbb eladott darabszám?

Feltételes maximumkeresés

Specifikáció:

```
Be: n∈N, db∈N[1..n]
Ki: van∈L, max∈N
Ef: -
Uf: (van,,max)=MAX(i=1..n, db[i], igaz)
```

Vezérlőelv

Feladat:

Egy polgárőrség nyilvántartásában m őr van, és tudjuk, hogy n nap mindegyikén melyik őr volt szolgálatban. Melyik őr volt a legtöbbször szolgálatban?

1.változat: nap-vezérelt megoldás

Feladat:

Egy polgárőrség nyilvántartásában m őr van, és tudjuk, hogy n nap mindegyikén melyik őr volt szolgálatban. Melyik őr volt a legtöbbször szolgálatban?

Lépések:

1. Minden naphoz meghatározzuk, hogy az adott napi őr, hányszor volt még szolgálatban
2. Ezek közül azt a napot választjuk, ahol ez a szám a legnagyobb
3. Maximumkiválasztásban megszámolás

1	3
2	5
3	4
4	5
5	4
6	6
7	5
n=8	3

1.változat: nap-vezérelt megoldás

Feladat:

Egy polgárőrség nyilvántartásában m őr van, és tudjuk, hogy n nap mindegyikén melyik őr volt szolgálatban. Melyik őr volt a legtöbbször szolgálatban?

Specifikáció:

Be: ~~m∈N,~~ n∈N, őr∈N[1..n]

Ki: terhelt∈N

Sa: ind∈N

Fv: hány:N->N,

hány(nap)=**DARAB**(i=nap+1..n, őr[i]=őr[nap])

Ef: -

Uf: (ind,)=**MAX**(nap=1..n, hány(nap)) és

terhelt=őr[ind] ~~és m=m~~

1	3
2	5
3	4
4	5
5	4
6	6
7	5
n=8	3

2. változat: őr-vezérelt megoldás

Feladat:

Egy polgárőrség nyilvántartásában m őr van, és tudjuk, hogy n nap mindegyikén melyik őr volt szolgálatban. Melyik őr volt a legtöbbször szolgálatban?

nap	melyik őr		őrök=	1	2	3	4	5	6	7
1	5							x		
2	3					x				
3	4						x			
4	5							x		
5	4						x			
6	6								x	
7	3					x				
8	5							x		
			db=	0	0	2	2	3	1	0

2. változat: őr-vezérelt megoldás

Feladat:

Egy polgárőrség nyilvántartásában m őr van, és tudjuk, hogy n nap mindegyikén melyik őr volt szolgálatban. Melyik őr volt a legtöbbször szolgálatban?

Lépések:

1. Minden őrhöz meghatározzuk, hogy hányszor volt szolgálatban
2. Ezek közül azt az őrt választjuk, ahol ez a szám a legnagyobb
3. Maximumkiválasztásban megszámolás

őrök=	1	2	3	4	5	6	7
					x		
			x				
				x			
					x		
				x			
						x	
			x				
					x		
db=	0	0	2	2	3	1	0

2. változat: őr-vezérelt megoldás

őrök=	1	2	3	4	5	6	7
					x		
			x				
				x			
					x		
				x			
						x	
			x				
					x		
db=	0	0	2	2	3	1	0

Feladat:

Egy polgárőrség nyilvántartásában m őr van, és tudjuk, hogy n nap mindegyikén melyik őr volt szolgálatban. Melyik őr volt a legtöbbször szolgálatban?

Specifikáció:

```
Be: m∈N, n∈N, őr∈N[1..n]
Ki: terhelt∈N
Fv: hány:N->N,
    hány(melyikőr)=DARAB(i=1..n, őr[i]=melyikőr)
Ef: -
Uf: (terhelt,)=MAX(melyikőr=1..m, hány(melyikőr))
```

Adattranszformáció

Feladat:

Egy polgárőrség nyilvántartásában n őr van, és tudjuk minden őrről, hogy m napból mely naptól mely napig volt szolgálatban. Mikor volt a legvédettebb a helység?

Adattranszformáció

Feladat:

Egy polgárőrség nyilvántartásában n őr van, és tudjuk minden őrről, hogy m napból mely naptól mely napig volt szolgálatban. Mikor volt a legvédettebb a helység?

őr	tól	ig	nap=	1	2	3	4	5	6	7	8	9	10
1	1	3		x	x	x							
2	2	2			x								
3	5	8						x	x	x	x		
4	7	8								x	x		
n=5	8	9									x	x	
			db=	1	2	1	0	1	1	2	3	1	0

1. változat: nap-vezérelt megoldás

Feladat: Mikor volt a legvédettebb a helység?

Lépések:

Adattranszformáció másolással

1. Átalakítás: meghatározzuk minden naphoz, hogy akkor hány őr volt szolgálatban.
2. Melyik nap a legnagyobb a darabszám?
3. Másolásban megszámolás és maximumkiválasztás

őr	tól	ig	nap=	1	2	3	4	5	6	7	8	9	10
1	1	3		x	x	x							
2	2	2			x								
3	5	8						x	x	x	x		
4	7	8								x	x		
n=5	8	9									x	x	
			db=	1	2	1	0	1	1	2	3	1	0

1. változat: minden nap

őr	tól	ig
1	1	3
2	2	2
3	5	8
4	7	8
n=5	8	9

Feladat: Mikor volt a legvédettebb a helység?

nap=	1	2	3	4	5	6	7	8	9	10
	x	x	x							
		x								
					x	x	x	x		
							x	x		
								x	x	
db=	1	2	1	0	1	1	2	3	1	0

Specifikáció:

```
Be: m∈N, n∈N, őrök∈Intervallum[1..n],
    Intervallum=Tól x Ig, Tól=N, Ig=N
Ki: maxind∈N
Sa: db∈N[1..m]
Fv: hányőr:N->N,
    hányőr(nap)=DARAB(i=1..n, őrök[i].tól<=nap<=őrök[i].ig)
Ef: m>0 és ∀i∈[1..n]:(őrök[i].tól>=1 és őrök[i].ig<=m)
Uf: ∀nap∈[1..m]:(db[nap]=hányőr(nap)) és
    (maxind,)=MAX(nap=1..m, db[nap])
```

Nap-vezérelt megoldás

2. változat: másolás

őr	tól	ig
1	1	3
2	2	2
3	5	8
4	7	8
n=5	8	9

Feladat: Mikor volt a legvédettebb a helység?

Specifikáció:

nap=	1	2	3	4	5	6	7	8	9	10
	x	x	x							
		x								
					x	x	x	x		
							x	x		
								x	x	
db=	1	2	1	0	1	1	2	3	1	0

```
Be: m∈N, n∈N, őrök∈Intervallum[1..n],
    Intervallum=Tól x Ig, Tól=N, Ig=N
Ki: maxind∈N
Sa: db∈N[1..m]
Fv: hányőr:N->N,
    hányőr(nap)=DARAB(i=1..n, őrök[i].tól<=nap<=őrök[i].ig)
Ef: m>0 és ∀i∈[1..n]:(őrök[i].tól>=1 és őrök[i].ig<=m)
Uf: db=MÁSOL(nap=1..m, hányőr(nap)) és
    (maxind,)=MAX(nap=1..m, db[nap])
```

Érdemes megnézni, hogy a másolás összevonható-e a következő lépéssel!

3. változat: másolás összevonása

Feladat: Mikor volt a legvédettebb a helység?

őr	tól	ig
1	1	3
2	2	2
3	5	8
4	7	8
n=5	8	9

Specifikáció:

```
Be: m∈N, n∈N, őrök∈Intervallum[1..n],
    Intervallum=Tól x Ig, Tól=N, Ig=N
Ki: maxind∈N
Sa: db∈N[1..m]
Fv: hányőr:N->N,
    hányőr(nap)=DARAB(i=1..n, őrök[i].tól<=nap<=őrök[i].ig)
Ef: m>0 és ∀i∈[1..n]:(őrök[i].tól>=1 és őrök[i].ig<=m)
Uf: (maxind,)=MAX(nap=1..m, hányőr(nap))
```

nap=	1	2	3	4	5	6	7	8	9	10
	x	x	x							
		x								
					x	x	x	x		
							x	x		
								x	x	
db=	1	2	1	0	1	1	2	3	1	0

Adattranszformáció – mátrix, vezérlés

Feladat:

Egy busztársaság nyilvántartásában n buszjárat van, és mindegyikről tudjuk, hogy m város közül melyek között járnak buszok. Adj meg egy olyan várost, amelybe nem tudunk ezzel a busztársasággal utazni!

Adattranszformáció – mátrix, vezérlés

Feladat:

Egy busztársaság nyilvántartásában n buszjárat van, és mindegyikről tudjuk, hogy m város közül melyek között járnak buszok. Adj meg egy olyan várost, amelybe nem tudunk ezzel a busztársasággal utazni!

járatok	honnan	hova
1	3	6
2	5	7
3	2	3
4	2	5
5	1	6

városok=	1	2	3	4	5	6	7
1						x	
2			x		x		
3		x				x	
4							
5		x					x
6	x		x				
7					x		

Adattranszformáció – mátrix, vezérlés

Feladat:

Adj meg egy olyan várost, amelybe nem tudunk ezzel a busztársasággal utazni!

Lépések:

1. Átalakítás: meghatározzuk minden várospárhoz, hogy van-e a két város között járat.
2. Melyik oszlop üres?
3. Másolásban eldöntés és keresésben mind eldöntés

járatok	honnan	hova
1	3	6
2	5	7
3	2	3
4	2	5
5	1	6

városok=	1	2	3	4	5	6	7
1						x	
2			x		x		
3		x				x	
4							
5		x					x
6	x		x				
7					x		

1. változat: adattranszformáció

járatok	honnan	hova
1	3	6
2	5	7
3	2	3
4	2	5
5	1	6

Adj meg egy olyan várost, amelybe nem tudunk ezzel a busztársasággal utazni!

városok=	1	2	3	4	5	6	7
1						x	
2			x		x		
3		x				x	
4							
5		x					x
6	x		x				
7					x		

Specifikáció:

```
Be: m∈N, n∈N, járatok∈Járat[1..n],
    Járat=Honnan x Hova, Honnan=N, Hova=N
Ki: ind∈N
Sa: mátrix∈L[1..m,1..m]
Fv: vanjárat:N x N->L,
    vanjárat(i,j)=VAN(k=1..n,
      (járatok[k].honnan=i és járatok[k].hova=j) vagy
      (járatok[k].hova=i és járatok[k].honnan=j))
Fv: üres:N->L,
    üres(hova)=MIND(i=1..m, mátrix[i,hova]=hamis)
Ef: ∀i∈[1..n]:
       (1<=járatok[i].honnan<=m és 1<=járatok[i].hova<=m)
Uf: ∀i∈[1..m]:(∀j∈[1..m]:(mátrix[i,j]=vanjárat(i,j))) és
    (,ind)=KERES(hova=1..m, üres(hova))
```

Érdemes megnézni, hogy a másolás összevonható-e a következő lépéssel!

2. változat: város-vezérelt megoldás

járatok	honnan	hova
1	3	6
2	5	7
3	2	3
4	2	5
5	1	6

Adj meg egy olyan várost, amelybe nem tudunk ezzel a busztársasággal utazni!

városok=	1	2	3	4	5	6	7
1						x	
2			x		x		
3		x				x	
4							
5		x					x
6	x		x				
7					x		

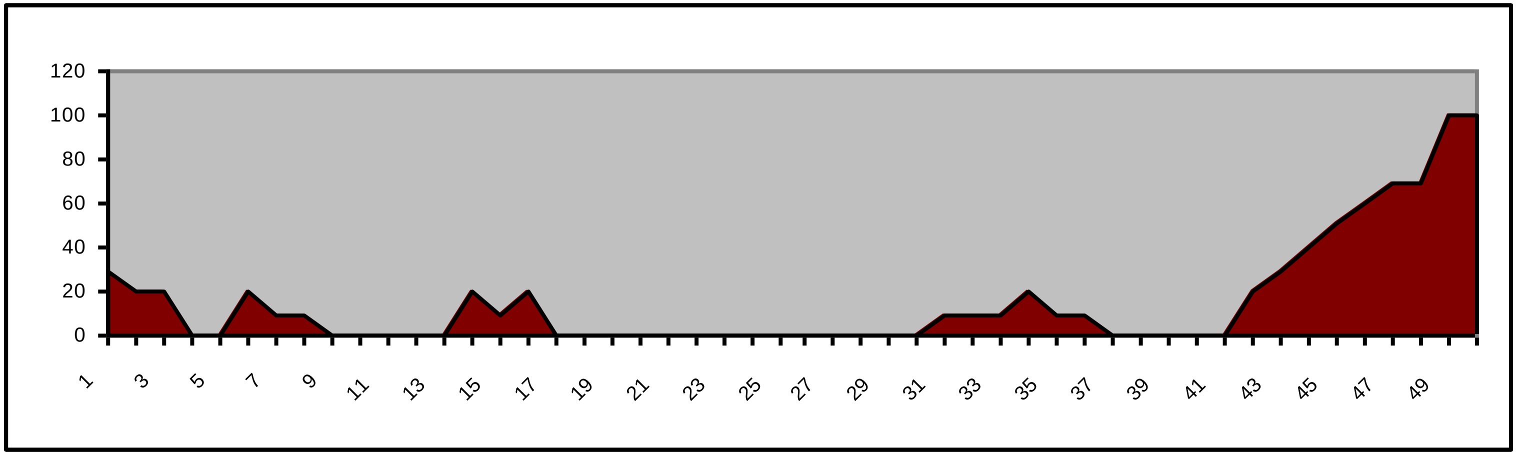
Specifikáció:

```
Be: m∈N, n∈N, járatok∈Járat[1..n],
    Járat=Honnan x Hova, Honnan=N, Hova=N
Ki: ind∈N
Fv: vanjárat:N->L,
    vanjárat(város)=VAN(k=1..n,
           járatok[k].honnan=város vagy járatok[k].hova=város)
Ef: ∀i∈[1..n]:
            (1<=járatok[i].honnan<=m és 1<=járatok[i].hova<=m)
Uf: (,ind)=KERES(város=1..m, nem vanjárat(város))
```

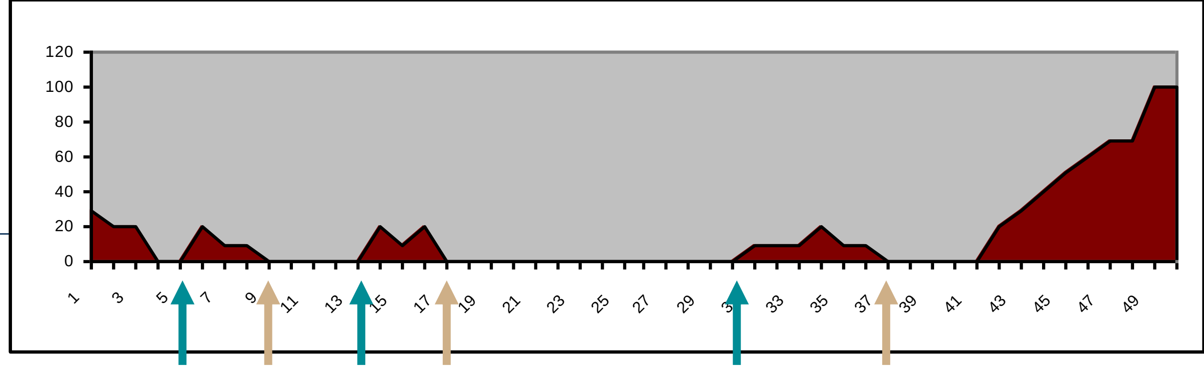
Intervallumos példák

Feladat:

Egy repülőgéppel Európából Amerikába repültünk. Az út során bizonyos kilométerenként mértük a felszín tengerszint feletti magasságát (≥0). 0 magasságot ott mértünk, ahol tenger van, >0-t pedig ott, ahol szárazföld. Adjuk meg a legszélesebb szigetet!



1. változat:

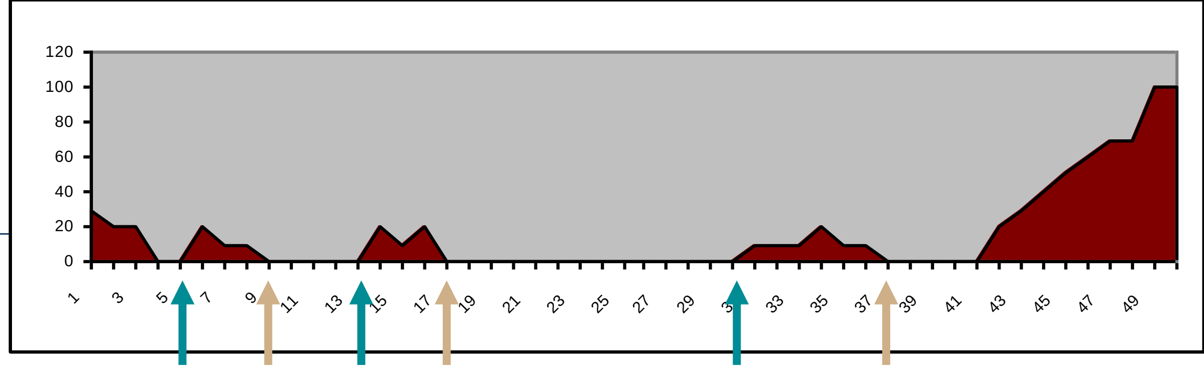


Specifikáció:

```
Be: n∈N, mag∈N[1..n]
Ki: van∈L, k∈N, v∈N
Sa: db∈N, kezdetek∈N[1..db], maxind∈N
Fv: szigetkezdet:N->L,
    szigetkezdet(i)={hamis, ha i=1;
                     mag[i-1]=0 és mag[i]>0 egyébként}
Fv: szigetvég:N->L,
    szigetvég(i)={hamis, ha i=n;
                  mag[i+1]=0 és mag[i]>0 egyébként}
Fv: keresvége:N->L x N,
    keresvége(i)=KERES(j=i..n, szigetvég(j))
Fv: táv:N->N, táv(i)=keresvége(i).ind - i
Ef: -
Uf: (db,kezdetek)=KIVÁLOGAT(i=1..n, szigetkezdet(i) és keresvége(i).van, i) és
    (van,maxind,)=MAX(i=1..db, táv(kezdetek[i]), igaz) és
    van -> k=kezdetek[maxind] és v=keresvége(k).ind
```

1. Válogassuk ki azokat a szigetkezdeteket, amelyeknek van vége is!
2. Másolással határozzuk meg hozzájuk a szigetvégeket! (összevonható)
3. Másolással határozzuk meg a távolságokat! (összevonható)
4. Határozzuk meg a legnagyobb távolságot, ha van!

2. változat:



Specifikáció:

```
Be: n∈N, mag∈N[1..n]
Ki: van∈L, k∈N, v∈N
Fv: szigetkezdet:N->L,
    szigetkezdet(i)={hamis, ha i=1;
                     mag[i-1]=0 és mag[i]>0 egyébként}
Fv: szigetvég:N->L,
    szigetvég(i)={hamis, ha i=n;
                  mag[i+1]=0 és mag[i]>0 egyébként}
Fv: keresvége:N->L x N,
    keresvége(i)=KERES(j=i..n, szigetvég(j))
Fv: táv:N->N, táv(i)=keresvége(i).ind - i
Ef: -
Uf: (van,k,)=MAX(i=1..n, táv(i), szigetkezdet(i) és keresvége(i).van) és
    van -> v=keresvége(k).ind
```

1. Válogassuk ki azokat a szigetkezdeteket, amelyeknek van vége is! (összevonható)
2. Másolással határozzuk meg hozzájuk a szigetvégeket! (összevonható)
3. Másolással határozzuk meg a távolságokat! (összevonható)
4. Határozzuk meg a legnagyobb távolságot, ha van!

Feltételes maximumkeresésben keresés

2. változat

```
Fv: keresvége:N->L x N,
    keresvége(i)=KERES(j=i..n, szigetvég(j))
Fv: táv:N->N, táv(i)=keresvége(i).ind - i
Ef: -
Uf: (van,k,)=MAX(i=1..n, táv(i), szigetkezdet(i) és keresvége(i).van) és
    van -> v=keresvége(k).ind
```

Feltételes maximumkeresés

maxind	~	k
e..u	~	1..n
f(i)	~	táv(i)
T(i)	~	szigetkezdet(i) és keresvége(i).van

Keresés (keresnulla)

i	~	j
e..u	~	i..n
T(i)	~	szigetvég(j)

2. változat

Feltételes maximumkeresés

maxind	~	k
e..u	~	1..n
f(i)	~	táv(i)
T(i)	~	szigetkezdet(i) és keresvége(i).van

Keresés (keresnulla)

i	~	j
e..u	~	i..n
T(i)	~	szigetvég(j)

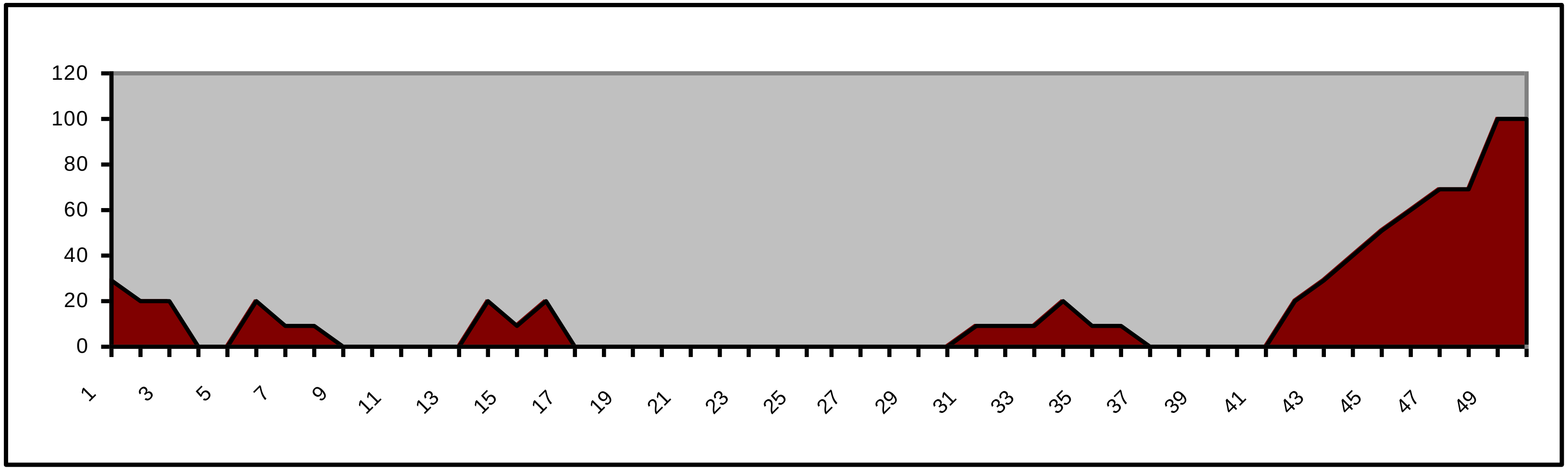
```
van:=hamis
i=1..n
    jóe:=szigetkezdet(i) és keresvége(i).van
    nem jóe | van és jóe                | nem van és jóe
    -       | t:=táv(i)                 | van:=igaz
            | t>maxért                  | maxért:=táv(i)
            | maxért:=t        | -      | k:=i
            | k:=i             |        |
```

keresvége(i:Egész):(Logikai,Egész)
Vált ind:Egész, van:Logikai

```
ind:=i
ind<=n és nem szigetvég(ind)
    ind:=ind+1
van:=ind<=n
keresvége:=(van,ind)
```

További változatok

- Mi és hogyan változik, ha
  - azt is szigetnek tekintem, ami esetleg a szélére esik? (ami eddig a kontinens volt)
  - a legrövidebb szigetet keresem?



Intervallum

Feladat:

Egy polgárőrség nyilvántartásában n őr van, és tudjuk minden őrről, hogy m napból mely naptól mely napig volt szolgálatban. Melyik volt az a leghosszabb időszak, amíg védve volt a helység?

őr	tól	ig	nap=	1	2	3	4	5	6	7	8	9	10
1	1	3		x	x	x							
2	2	2			x								
3	5	8						x	x	x	x		
4	7	8								x	x		
n=5	8	9									x	x	
			db=	1	2	1	0	1	1	2	3	1	0

1. változat

őr	tól	ig
1	1	3
2	2	2
3	5	8
4	7	8
n=5	8	9

Feladat: leghosszabb időszak, amíg védve

Specifikáció:

```
Be: m∈N, n∈N, őrök∈Intervallum[1..n],
    Intervallum=Tól x Ig, Tól=N, Ig=N
Ki: maxind∈N
Sa: db∈N[1..m]
Fv: hányőr:N->N,
    hányőr(nap)=DARAB(i=1..n, őrök[i].tól<=nap<=őrök[i].ig)
Ef: m>0 és ∀i∈[1..n]:(őrök[i].tól>=1 és őrök[i].ig<=m)
Uf: ∀nap∈[1..m]:(db[nap]=hányőr(nap)) és
    ...
```

nap=	1	2	3	4	5	6	7	8	9	10
	x	x	x							
		x								
					x	x	x	x		
							x	x		
								x	x	
db=	1	2	1	0	1	1	2	3	1	0

Kezdődhet a leghosszabb „sziget” meghatározása!

Rendezések

Rendezési feladat

Specifikáció:

```
Be: n∈N, x∈H[1..n],
    <=:H x H->L
Ki: y∈H[1..n]
Ef: -
Uf: RendezettE(y) és y∈Permutáció(x)
```

Rendezés(≤) és $\text{RendezettE}_{\leq}$(H)

Jelölések:

- $\text{RendezettE}_{\leq}$(X/H): X/H rendezett-e a ≤-ra?
- Y∈Permutáció(X): Y az X elemeinek egy permutációja-e?

Rendezési feladat

A rendezések egy részében olyan megvalósítást választunk, amiben a bemenetnek és a kimenetnek ugyanaz a sorozat felel meg, azaz helyben rendezünk.

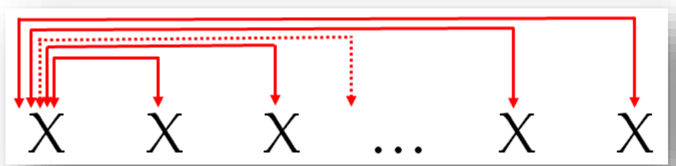
Specifikáció:

```
Be: n∈N, x∈H[1..n],
    <=:H x H->L
Ki: x'∈H[1..n]
Ef: -
Uf: RendezettE(x') és x'∈Permutáció(x)
```

Egyszerű cserés rendezés

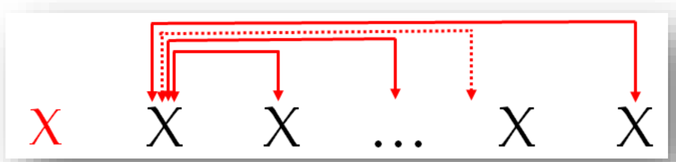
A lényeg:

- Hasonlítsuk az első elemet az összes mögötte levővel, s ha kell, cseréljük meg!



A minimum az „alsó” végére kerül.

- Ezután ugyanezt csináljuk a második elemre!
- …



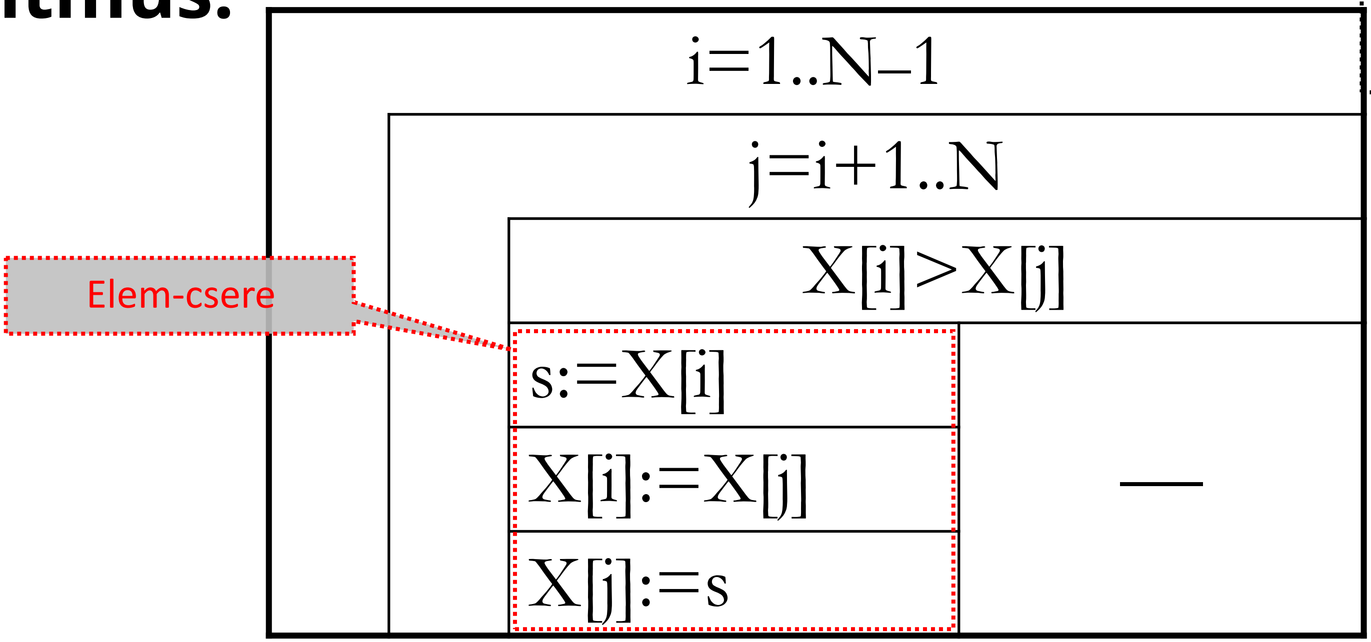
- Végül az utolsó két elemre!

A pirossal jelöltek már a helyükön vannak

Egyszerű cserés rendezés

Algoritmus:

Változó
i,j:**Egész**
s:TH

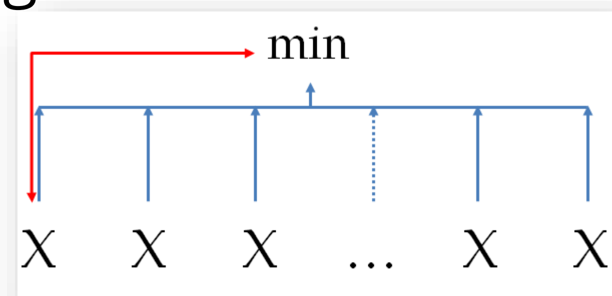


- Hasonlítások száma: 1+2+…+N–1= $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 0 .. $3 \cdot N \cdot \frac{N-1}{2}$

Minimumkiválasztásos rendezés

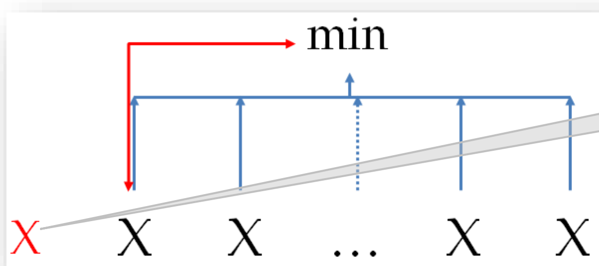
A lényeg:

- Határozzuk meg az 1..N elemek minimumát, s cseréljük meg az 1.-vel!



A minimum az „alsó" végére kerül.

- Ezután ugyanezt tegyük a 2..N elemre!



A pirossal jelöltek már a helyükön vannak

- ...
- Végül az utolsó két (N–1..N) elemre!

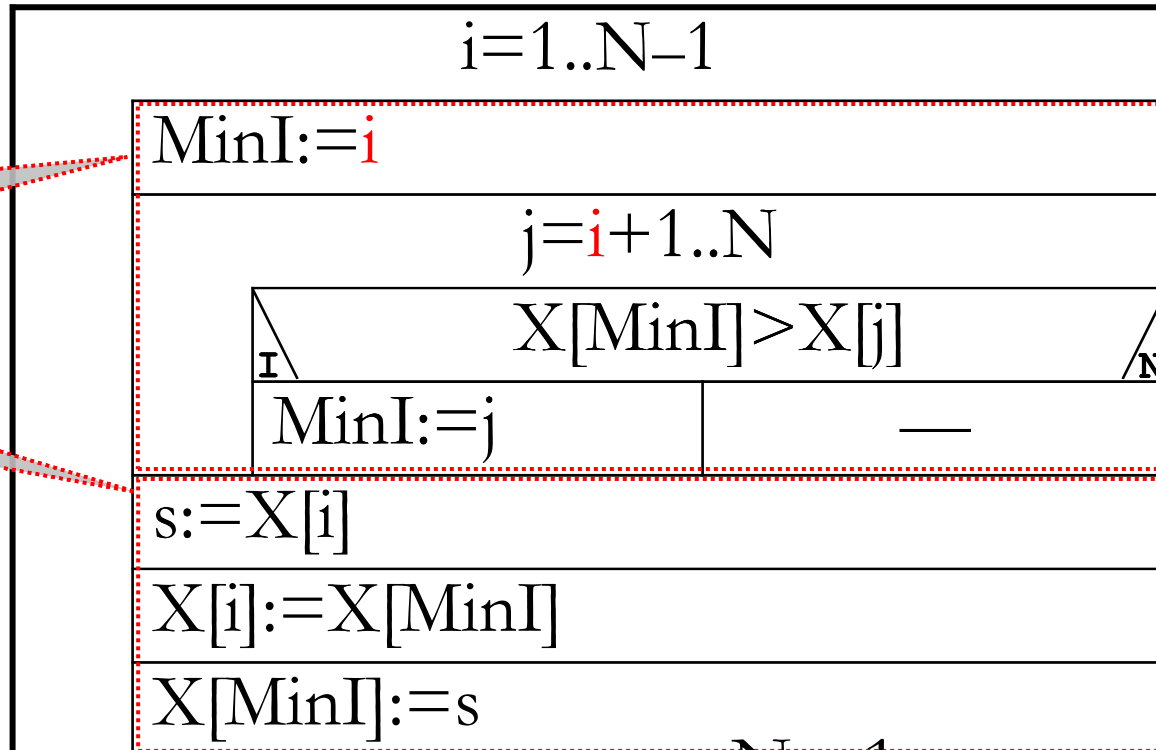
Minimumkiválasztásos rendezés

Változó
MinI,
i,j:**Egész**
s:TH

Algoritmus:

Minimum-kiválasztás az i.-től

Elem-csere

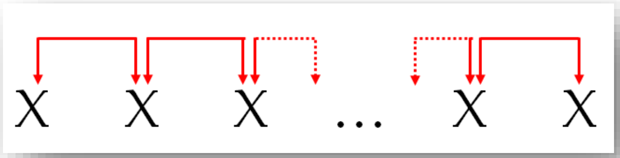


- Hasonlítások száma: 1+2+...+N–1= $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 3*(N–1)

Buborékos rendezés

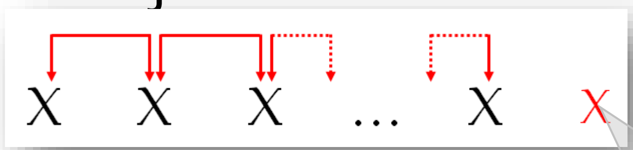
A lényeg:

- Hasonlítsunk minden elemet a mögötte levővel, s ha kell, cseréljük meg!



A maximum a „felső” végére kerül.

- Ezután ugyanezt csináljuk az utolsó elem nélkül!



- …

A többiek is tartanak a helyük felé.

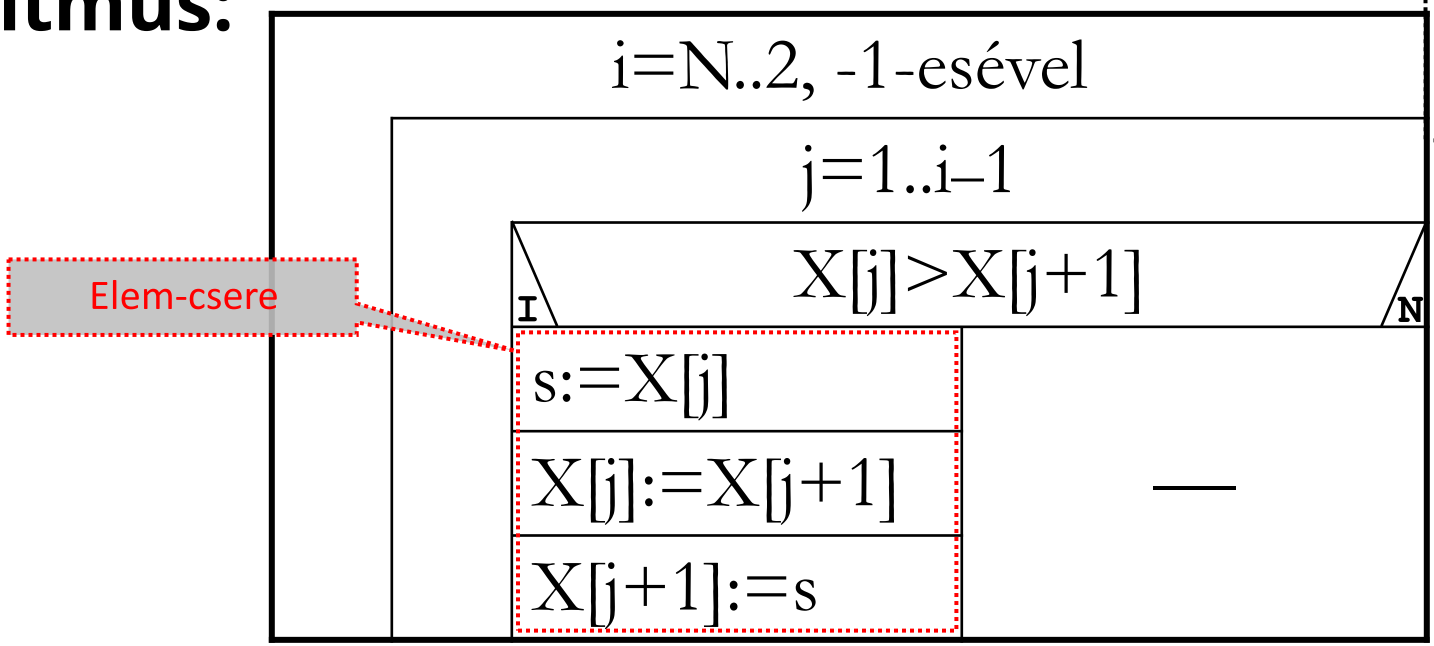
- Végül az első két elemre!

A pirossal jelöltek már a helyükön vannak

Buborékos rendezés

Algoritmus:

Változó
i,j:**Egész**
s:TH

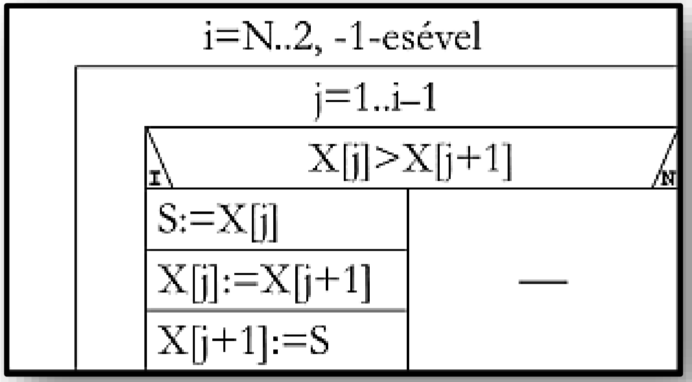


- Hasonlítások száma: 1+2+…+N–1= $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 0 .. $3 \cdot N \cdot \frac{N-1}{2}$

Javított buborékos rendezés

Megfigyelések:

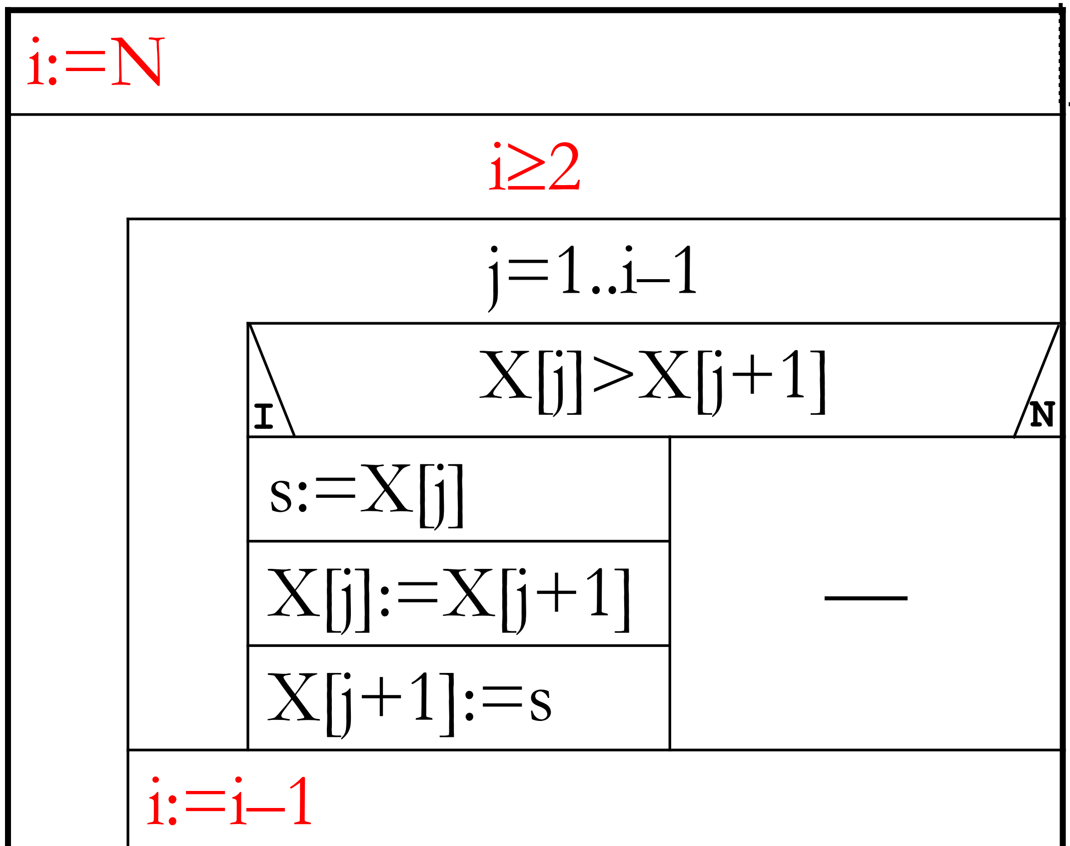
- Ha a belső ciklusban egyáltalán nincs csere, akkor be lehetne fejezni a rendezést.
- Ha a belső ciklusban a K. helyen van az utolsó csere, akkor a K+1. helytől már biztosan jó elemek vannak, a külső ciklus-változóval többet is léphetnénk.



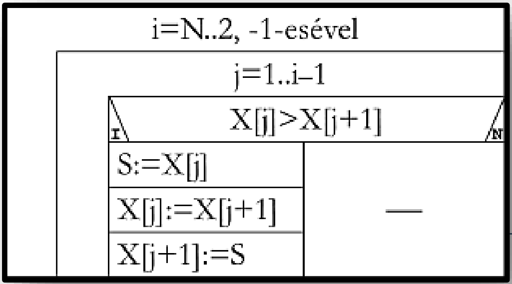
Javított buborékos rendezés

Algoritmus: (átalakítva feltételes ciklusúvá)

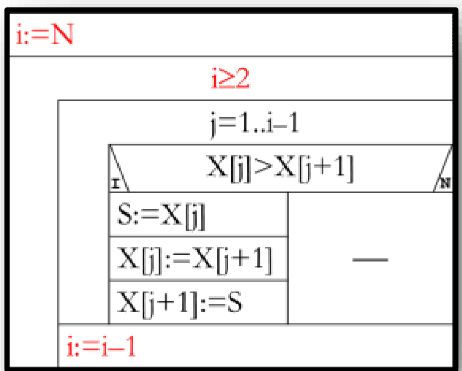
Változó
i,j:**Egész**
s:TH



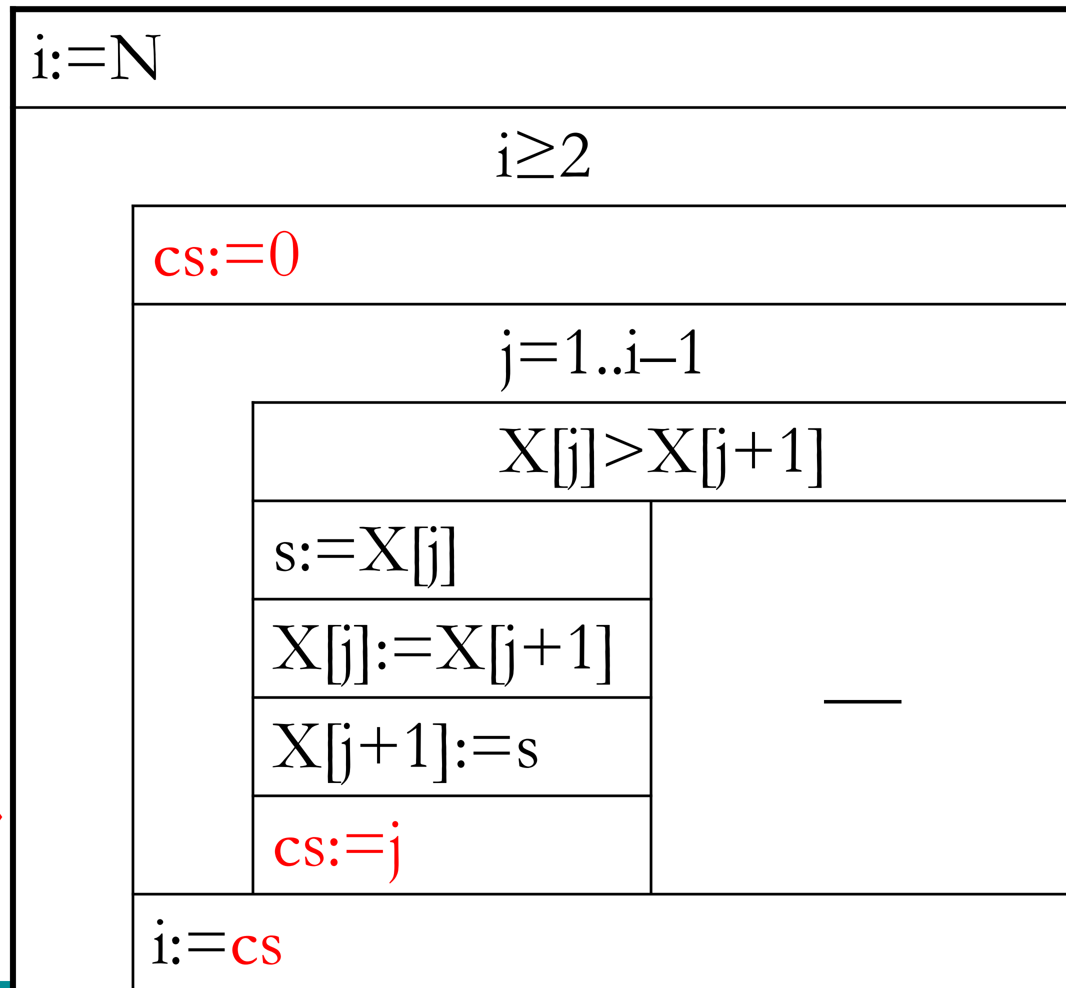
Javított buborékos rendezés



Átírás
'amíg'-os
ciklussá



Az utolsó
cserehely
feljegyzése



Változó
cs,
i,j:**Egész**
s:TH

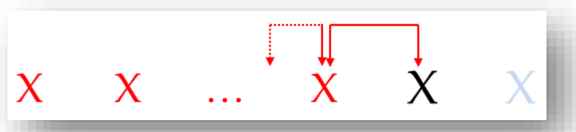
Beillesztéses rendezés

A lényeg:

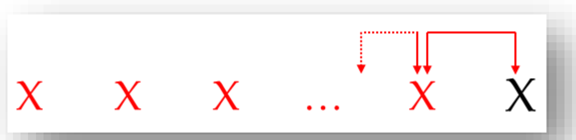
- *Egy* elem *rendezett*.

- A másodikat vagy mögé, vagy elé tesszük, így már *ketten* is *rendezettek*.

- …
- Az i-ediket a kezdő, i–1 *rendezett*ben addig hozzuk előre **cserékkel**, amíg a helyére nem kerül; így már *i darab rendezett* lesz.



- …
- Az utolsóval ugyanígy!



Beillesztéses rendezés

Változó
i,j:**Egész**
s:TH

Algoritmus:

Keresés tétel

Elem-csere

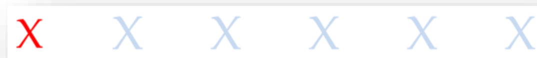
i=2..N		
	j:=i–1	
	j>0 és X[j]>X[j+1]	
		s:=X[j]
		X[j]:=X[j+1]
		X[j+1]:=s
		j:=j–1

- Hasonlítások száma: N–1 .. $N \cdot \frac{N-1}{2}$
- Mozgatások száma: 0 .. $3 \cdot N \cdot \frac{N-1}{2}$

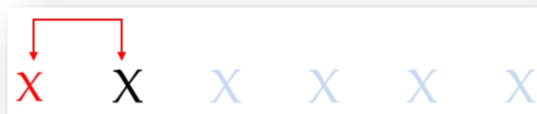
Javított beillesztéses rendezés

A lényeg:

- *Egy* elem *rendezett*.

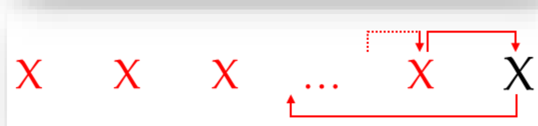


- A másodikat vagy mögé, vagy elé tesszük, így már *ketten* is *rendezettek*.



- …
- Az i-ediknél a nála nagyobbakat **tologassuk** hátra, majd illesszük be eléjük az i-ediket; így már *i darab rendezett* lesz.

- …
- Az utolsóval ugyanígy!



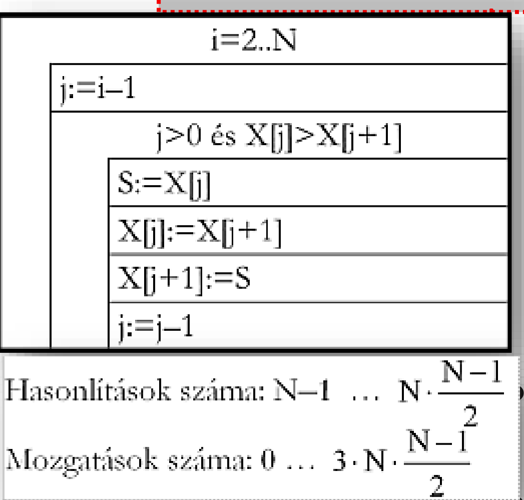
Javított beillesztéses rendezés

Algoritmus:

Változó
i,j:**Egész**
s:TH

```
i=2..N
    s:=X[i]
    j:=i–1
    j>0 és X[j]>s
        X[j+1]:=X[j]
        j:=j–1
    X[j+1]:=s
```

Elem-mozgatás, nem



- Hasonlítások száma: N–1 . . $N\cdot\frac{N-1}{2}$
- Mozgatások száma: 2*(N – 1) . . $(N+4)\cdot\frac{N-1}{2}$

Rendezésvizualizációk

- Inkább hatékonyság szemléltetésére, mint megértésére valók
- Vizualizációk
  - https://www.toptal.com/developers/sorting-algorithms
  - https://www.sortvisualizer.com/selectionsort/
  - https://sorting-algorithm-jet.vercel.app/
  - https://visualize-it.github.io/sorting_algos/simulation.html