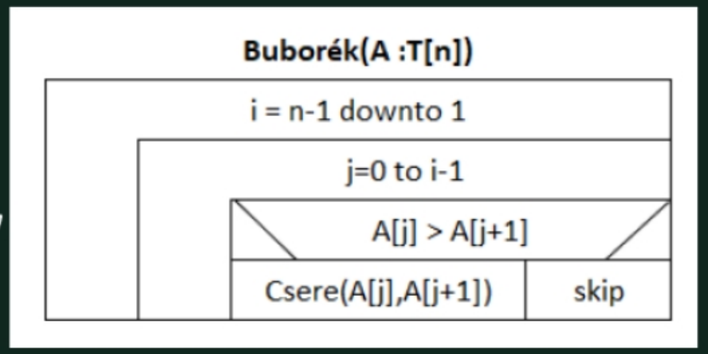


Futásidő =

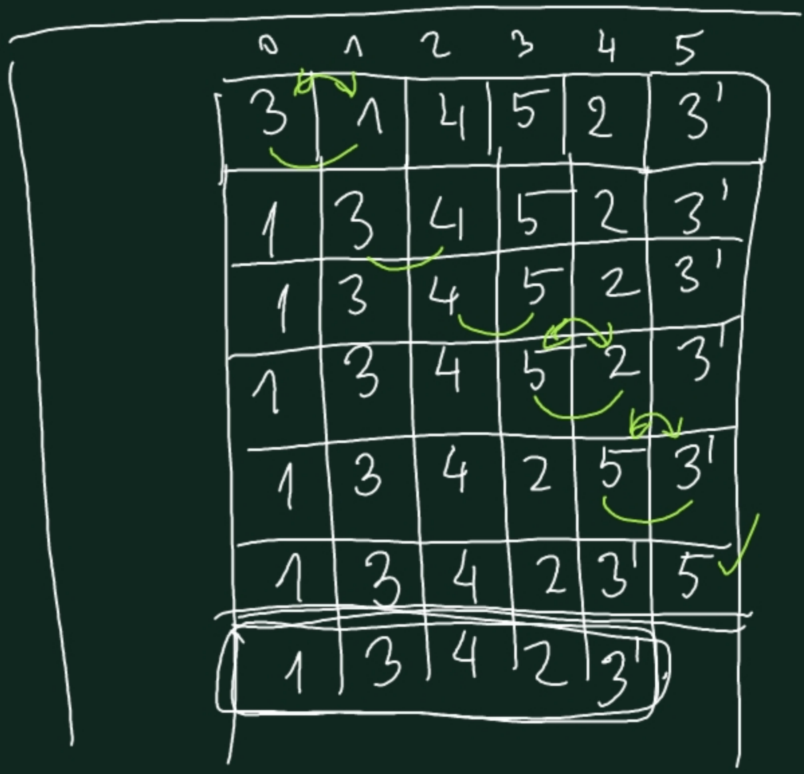
= ciklusiterációk száma + alprogram-hívások száma

$$n-1 + \underbrace{n-1 + n-2 + n-3 + \ldots + 1}_{} =$$

$$= \frac{n(n+1)}{2} - 1 \in \Theta(n^2)$$

$$mT(n) = MT \in \Theta(n^2)$$

minimális, maximális

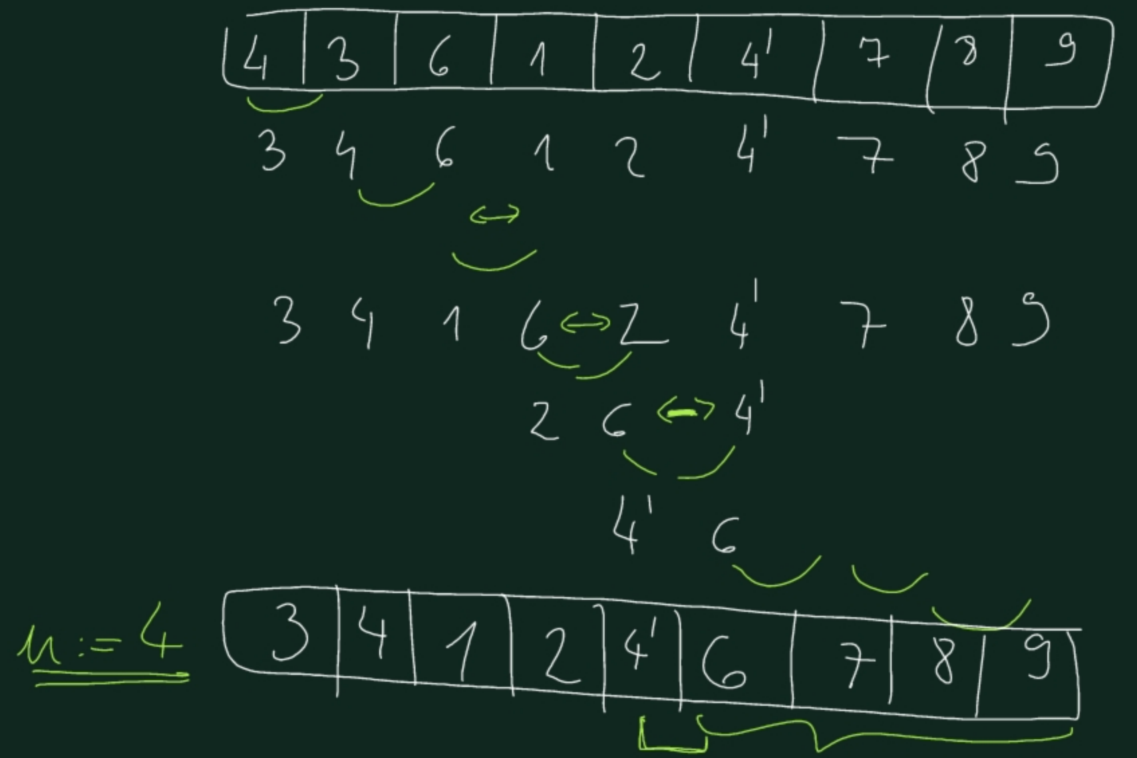


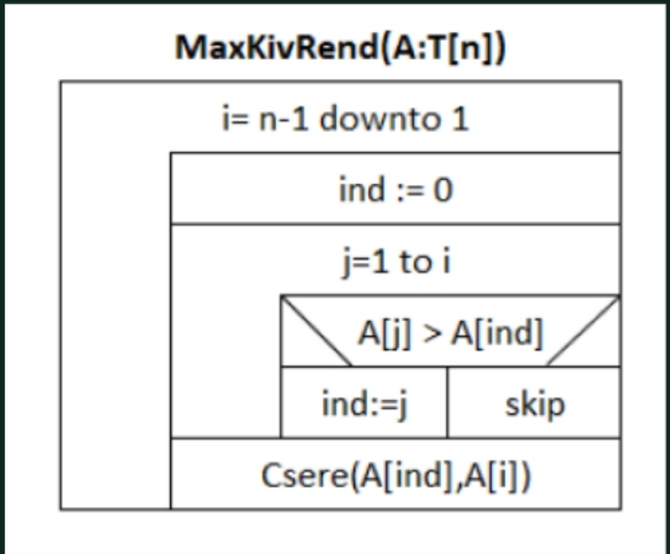
JavítottBuborék(A :T[n])

```
i:=n-1
i ≥ 1
    u:=-1
    j = 0 to i-1
        A[j] > A[j+1]
            Csere(A[j],A[j+1])
            u:=j
        skip
    i:=u
```

$mT(n) \in \Theta(n)$

$MT(n) \in \Theta(n^2)$





$$mT(n) = MT(n) \in \Theta(n^2)$$

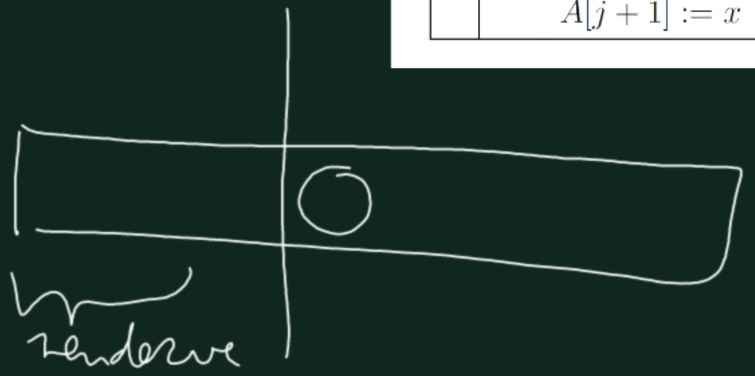
ciklusiterációk száma

minden inputra megegyezik

(cserék száma is)

```
naiveInsertionSort(A : T[n])
    i := 1 to n − 1
        j := i
        j > 0 ∧ A[j − 1] > A[j]
            swap(A[j − 1], A[j])
            j := j − 1
```

```
insertionSort(A : T[n])
    i := 1 to n − 1
        A[i − 1] > A[i]
        x := A[i]                          | SKIP
        A[i] := A[i − 1]                   |
        j := i − 2                         |
        j ≥ 0 ∧ A[j] > x                   |
            A[j + 1] := A[j]               |
            j := j − 1                     |
        A[j + 1] := x                      |
```



Feladat: szemléltessük a beszúró rendezést az alábbi tömbön, adjuk meg pontosan hány összehasonlítást és mozgatást végez az algoritmus!

Piros keret jelzi a kulcs-összehasonlításokat, kék a kulcs-mozgatásokat.

Jelöljük, mely kulcsokkal történt összehasonlítás!

```
insertionSort(A : T[n])
    i := 1 to n − 1
        A[i − 1] > A[i]
        x := A[i]                          | SKIP
        A[i] := A[i − 1]                   |
        j := i − 2                         |
        j ≥ 0 ∧ A[j] > x                   |
            A[j + 1] := A[j]               |
            j := j − 1                     |
        A[j + 1] := x                      |
```

		0	1	2	3	4	5	6	7	öh	mozg.
	A	4	3	1	3'	5	2	6	4'		

		0	1	2	3	5	6	7	8		
1. menet	A	4	3	1	3'	5	2	6	4'	1	3
2. menet	A	3	4	1	3'	5	2	6	4'	2	4
3. menet	A	1	3	4	3'	5	2	6	4'	2	3
4. menet	A	1	3	3'	4	5	2	6	4'	1	0
5. menet	A	1	3	3'	4	5	2	6	4'	5	6
6. menet	A	1	2	3	3'	4	5	6	4'	1	0
7. menet	A	1	2	3	3'	4	5	6	4'	3	4
Összesen:	A	1	2	3	3'	4	4'	5	6	15	20

$$mT(n) \in \Theta(n)$$

$$MT(n) \in \Theta(n^2)$$

mergeSort($A : \mathcal{T}[n]$)

$B : \mathcal{T}[n]$; $B[0..n) := A[0..n)$
// Sort $B[0..n)$ into $A[0..n)$ non-decreasingly:
ms($B, A, 0, n$)

ms($B, A : \mathcal{T}[]$; $u, v : \mathbb{N}$)

// Initially $B[u..v) = A[u..v)$.	
// Sort $B[u..v)$ into $A[u..v)$ non-decreasingly:	
$v - u > 1$	
$m := \lfloor \frac{u+v}{2} \rfloor$	SKIP
ms(A, B, u, m) // Sort $A[u..m)$ into $B[u..m)$ non-decreasingly.	
ms(A, B, m, v) // Sort $A[m..v)$, into $B[m..v)$ non-decreasingly.	
merge(B, A, u, m, v) // merge $B[u..m)$ and $B[m..v)$ into $A[u..v)$	

merge($B, A : \mathcal{T}[]\ ;\ u, m, v : \mathbb{N}$)

// sorted merge of $B[u..m)$ and $B[m..v)$ into $A[u..v)$

$k := u$ // in loop, copy into $A[k]$

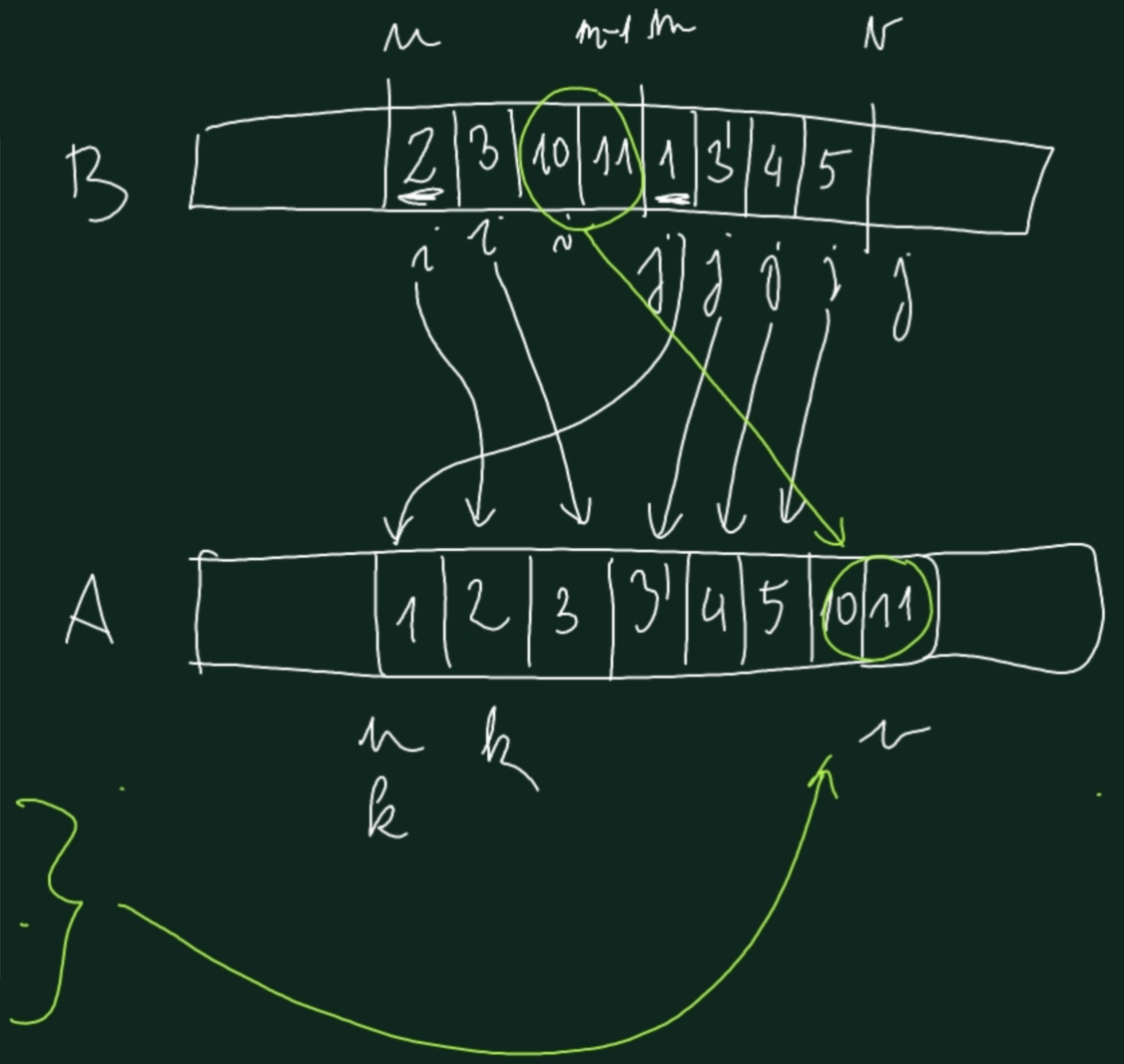
$i := u\ ;\ j := m$ // from $B[i]$ or $B[j]$

$i < m \wedge j < v$

$B[i] \leq B[j]$	
$A[k] := B[i]$	$A[k] := B[j]$
$i := i + 1$	$j := j + 1$
$k := k + 1$	

$i < m$

$A[k..v) := B[i..m)$	$A[k..v) := B[j..v)$

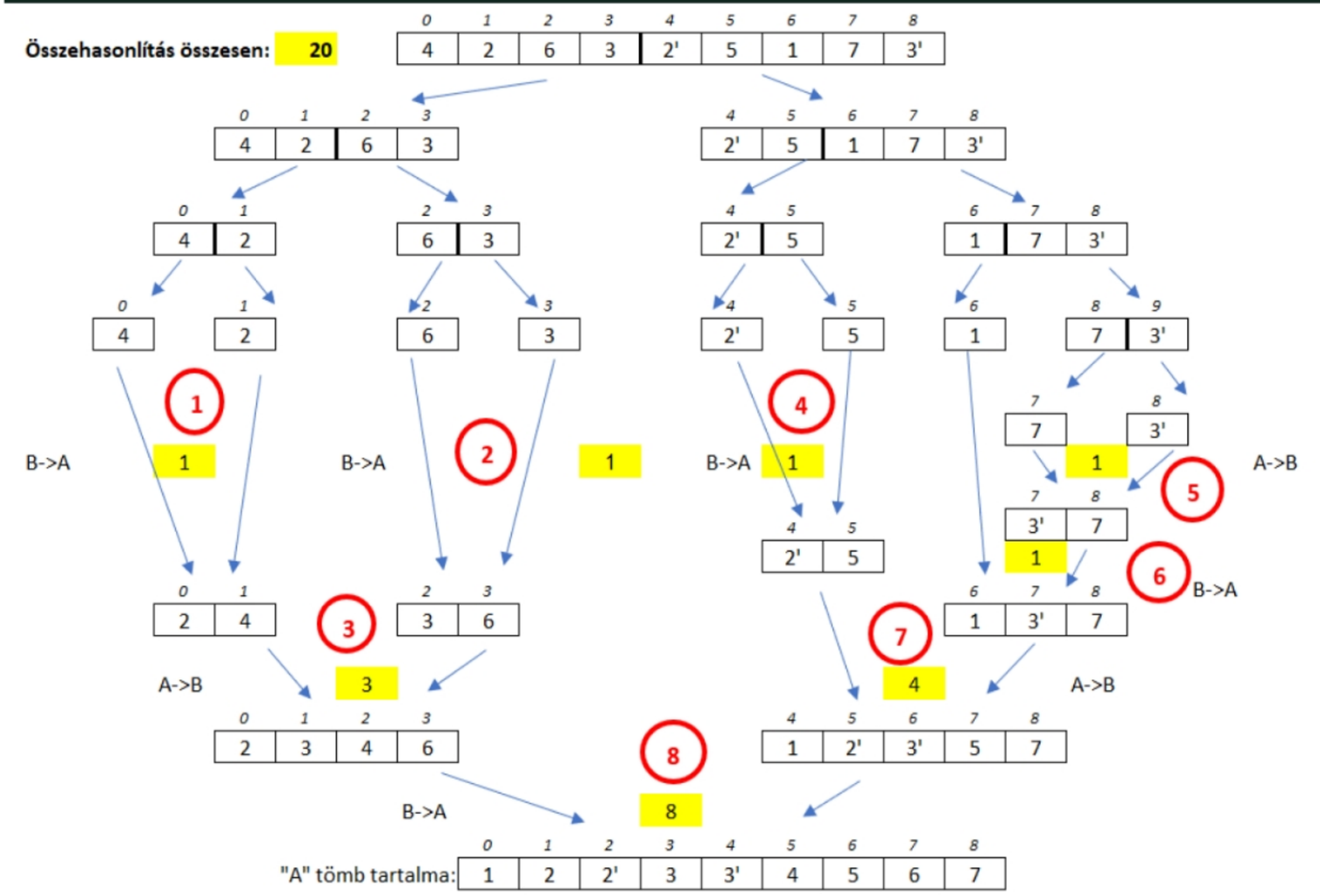


$\text{ms}(B, A : \mathcal{T}[]\ ;\ u, v : \mathbb{N})$

// Initially $B[u..v) = A[u..v)$.
// Sort $B[u..v)$ into $A[u..v)$ non-decreasingly:
$v - u > 1$
$m := \lfloor \frac{u+v}{2} \rfloor$
$\text{ms}(A, B, u, m)$ // Sort $A[u..m)$ into $B[u..m)$ non-decreasingly.
$\text{ms}(A, B, m, v)$ // Sort $A[m..v)$, into $B[m..v)$ non-decreasingly.
$\text{merge}(B, A, u, m, v)$ // merge $B[u..m)$ and $B[m..v)$ into $A[u..v)$

SKIP

$$mT(n) = MT(n) = \Theta(n \log n)$$



Egy rendezés stabil, ha az azonos értékű elemek sorrendje ugyanaz marad.

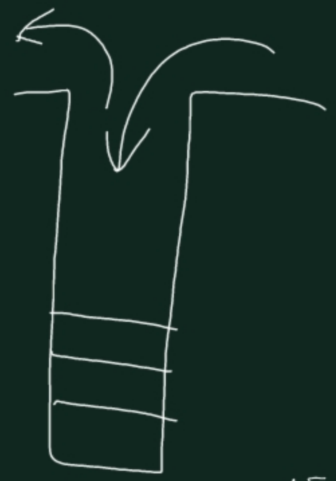
Buborék, Beszúró, Merge ✓

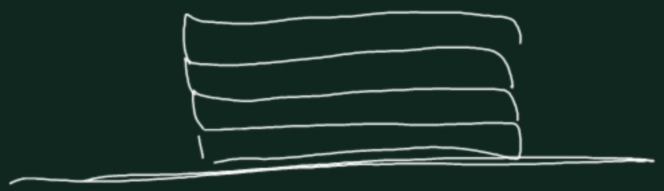
(Max Kiv ← nem stabil)

1 5 2 3 5'

4 3 1 1'

Verem Stack





LIFO

last in first out

v : Stack(n) ← max méret

v.push(x)

v.top ← kiolvassa a legfölső elemet

v.pop ← kiolvassa és kiveszi

v.isEmpty : B

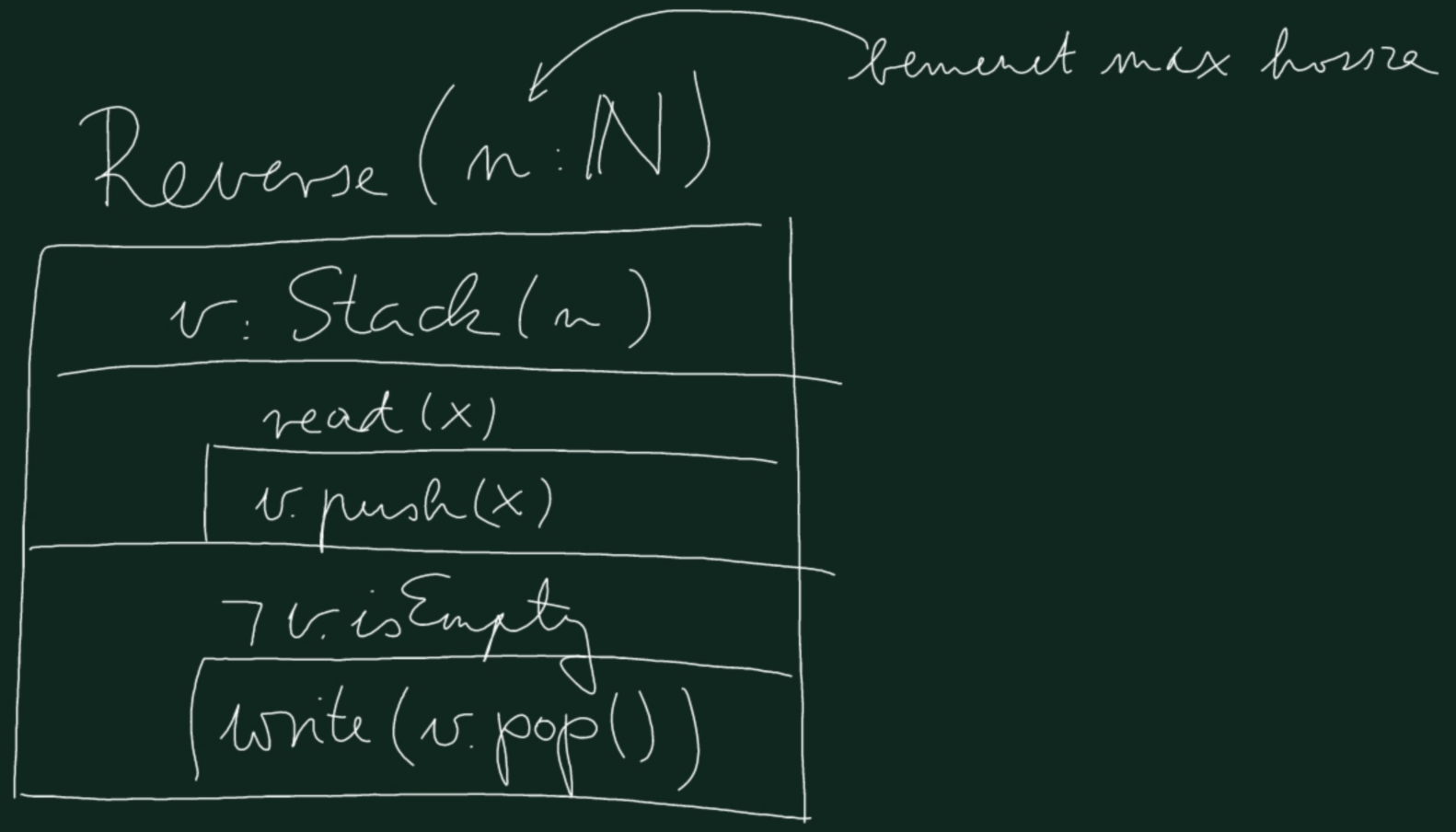
v.isFull : B

Bemeneten karakterenként érkező legfeljebb n hosszú szöveget írjunk ki a kimenetre tükrözve.

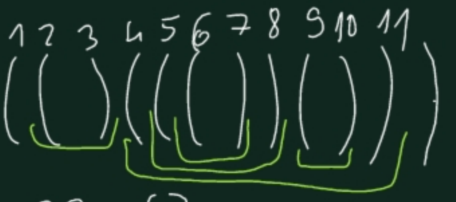
write(x)

read(x)

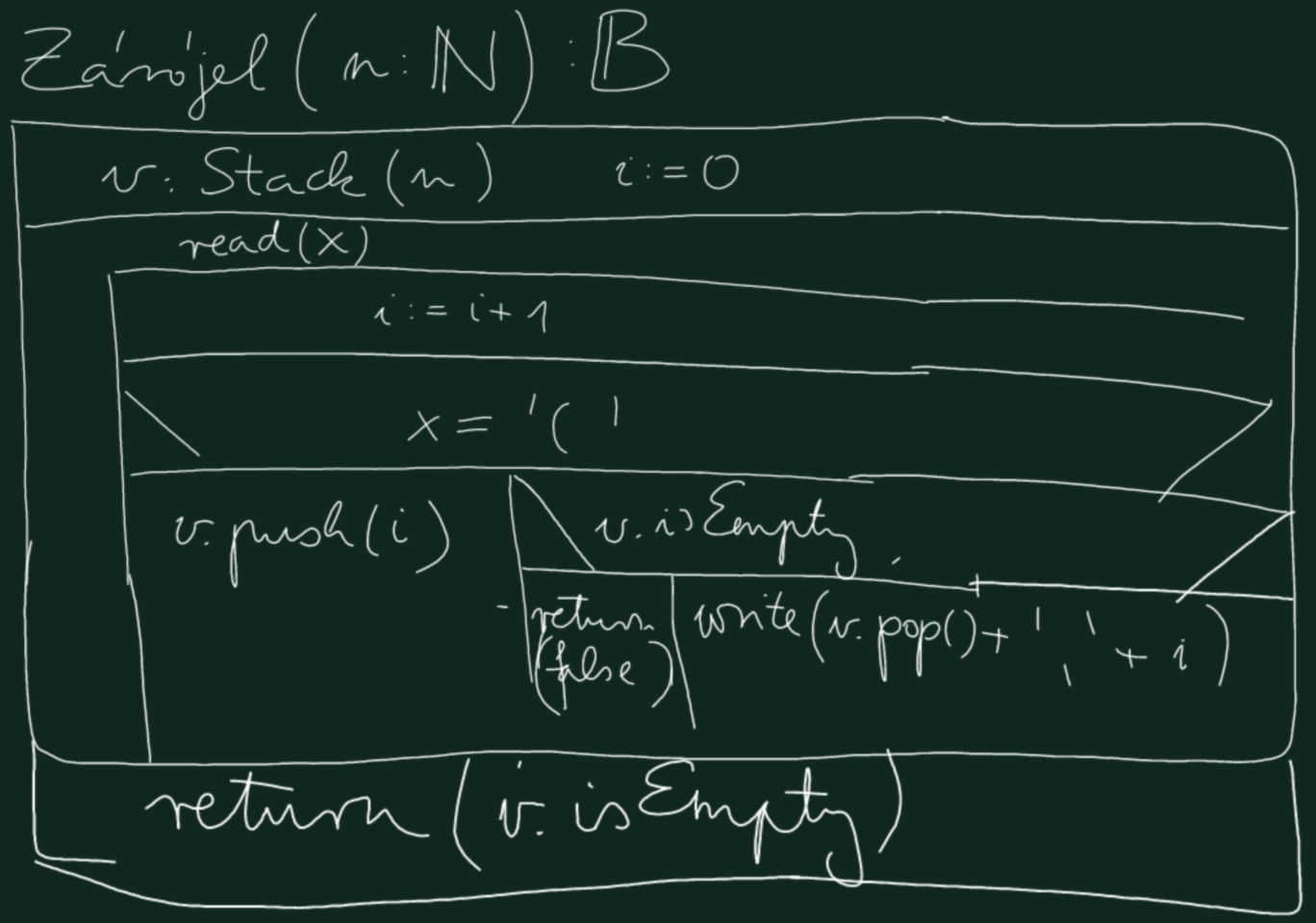
- x-be beolvasson a köv. karaktert (egységet/elemet)
- ha nem olvas be semmit, akkor hamis értéket ad (különben igazat)


Reverse(n : ℕ)
v : Stack(n)
read(x)
v.push(x)
¬ v.isEmpty
write(v.pop())
bement max hossza

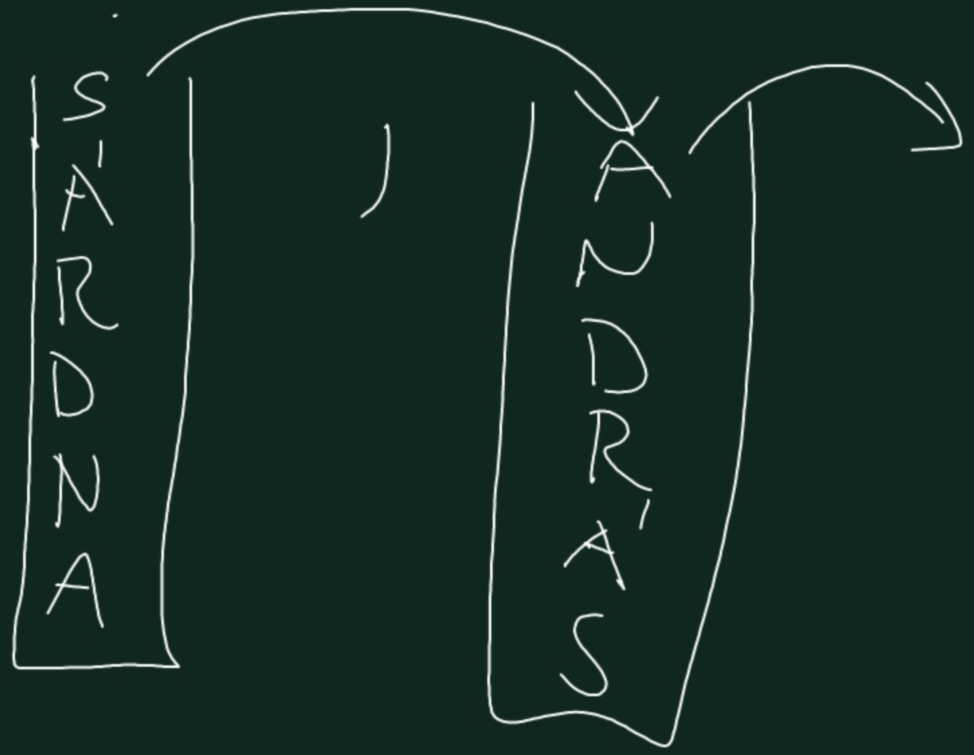
Adott egy zárójelekből álló, legfeljebb n hosszú karaktersorozat a bemeneten. Olvassuk be, és döntsük el róla, hogy helyes zárójelezést határoz-e meg! írassuk ki az összetartozó zárójelpárok indexeit!



2 3
6 7
5 8
9 10
4 11
1 12



A bemenetről karakterenként beolvasunk egy (legfeljebb n karakterből álló) névsort, ahol a nevek egymástól vesszővel vannak elválasztva. írjuk ki a neveket fordított sorrendben! Például: András,Béla,Csaba → Csaba,Béla,András



Stack
$-$ $A : \mathcal{T}[]$ // $\mathcal{T}$ is some known type ; $A.length$ is the physical
$-$ constant $m0 : \mathbb{N}_+ := 16$ // size of the stack, its default is m0.
$-$ $n : \mathbb{N}$ // $n \in 0..A.length$ is the actual size of the stack
$+$ Stack($m : \mathbb{N}_+ := m0$) $\{A :=$ **new** $\mathcal{T}[m]$; $n := 0\}$ // create empty stack
$+ \sim$ Stack() { **delete** A }
$+$ push($x : \mathcal{T}$) // push x onto the top of the stack
$+$ pop() : $\mathcal{T}$ // remove and return the top element of the stack
$+$ top() : $\mathcal{T}$ // return the top element of the stack
$+$ isEmpty() : $\mathbb{B}$ {**return** $n = 0$}
$+$ setEmpty() $\{n := 0\}$ // reinitialize the stack