

# ALG 07

**Selection sort (Select sort)**

**Insertion sort (Insert sort)**

**Bubble sort deprecated**

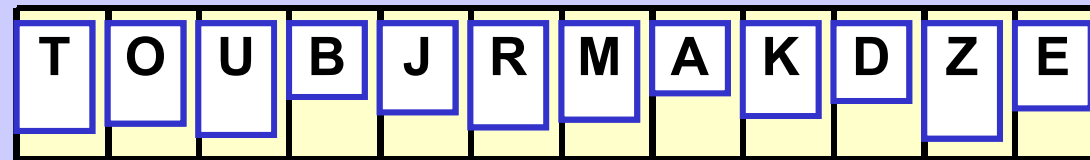
**Quicksort**

**Stabilita řazení**

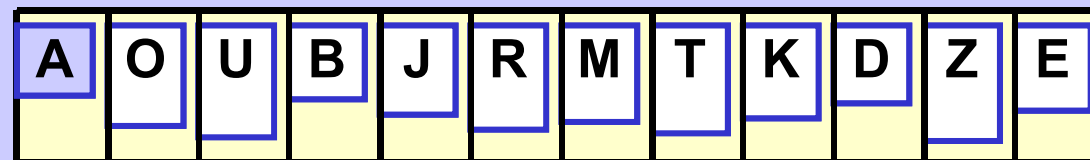
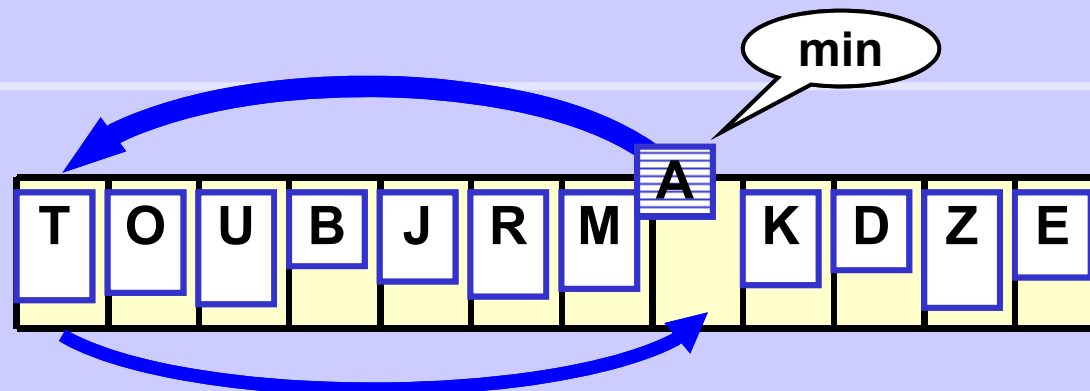
# Selection sort

Neseřazeno Seřazeno 

Start

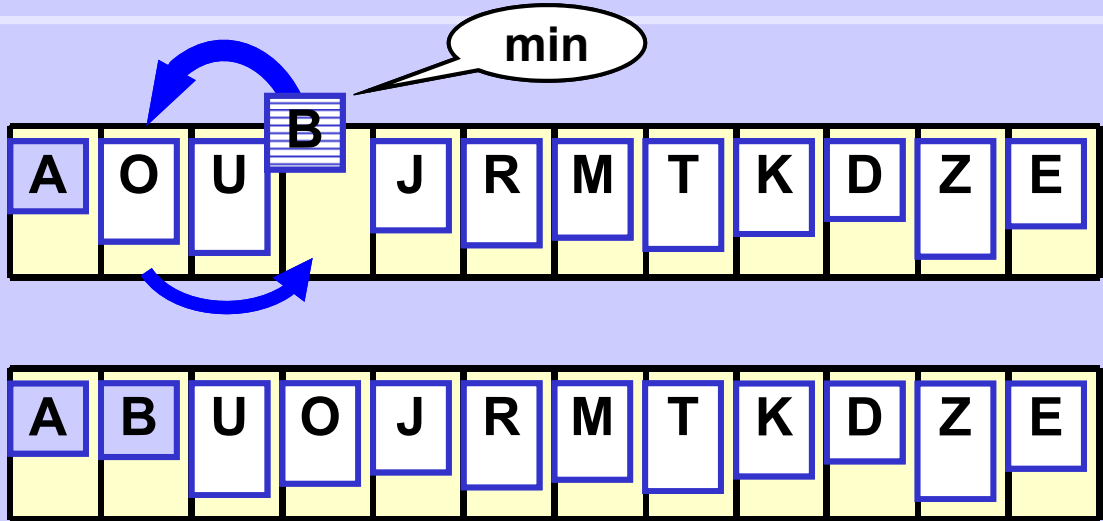


Krok1

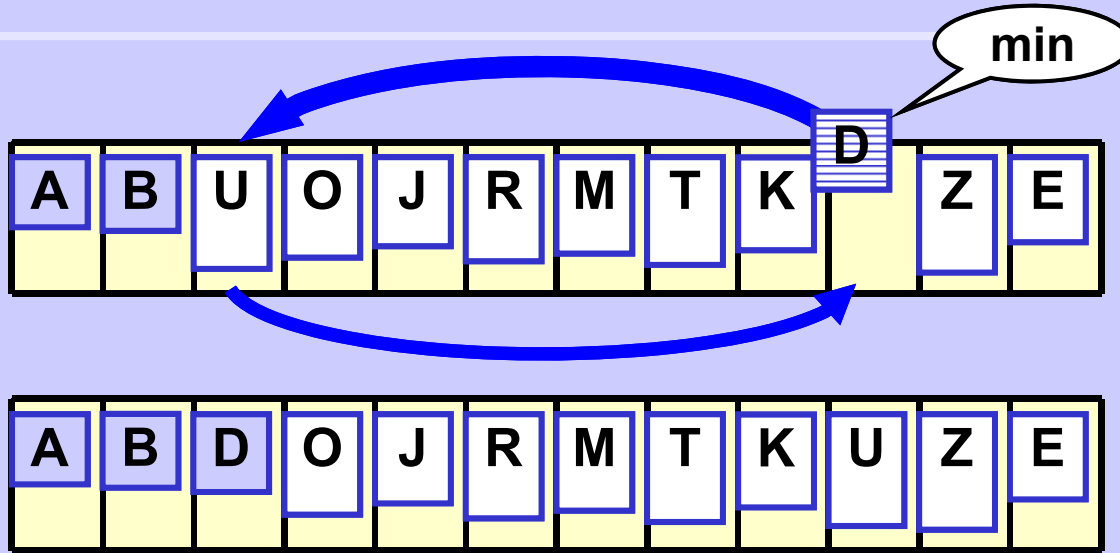


# Selection sort

Krok 2



Krok 3

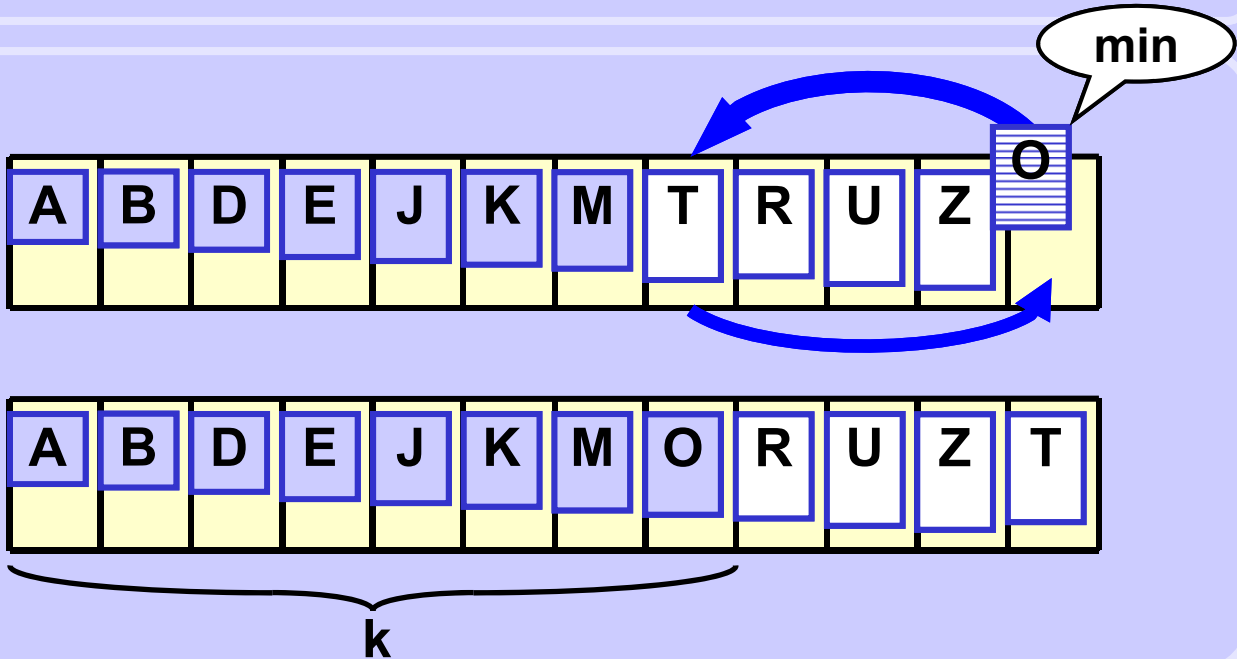


# Selection sort

...

...

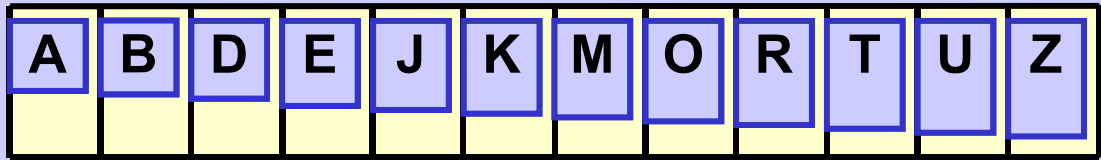
Krok k



...

...

Seřazeno

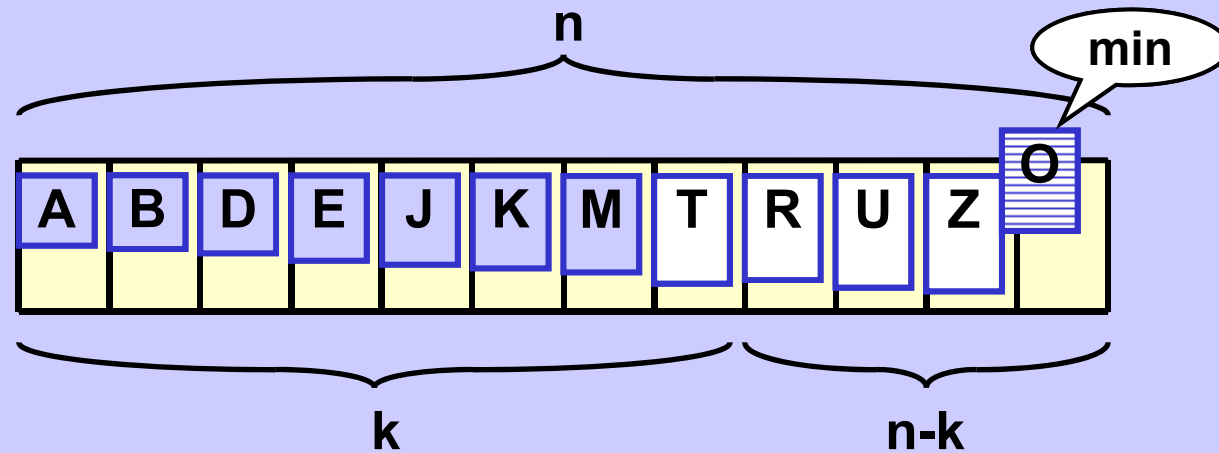


## Selection sort

```
for( int i = 0; i < n-1; i++ ){  
  
    // select min  
    jmin = i;  
    for( int j = i+1; j < n; j++ )  
        if( a[j] < a[jmin] )  
            jmin = j;  
  
    // put min to its place  
    min = a[jmin];  
    a[jmin] = a[i];  
    a[i] = min;  
}
```

## Selection sort

Krok k



Výběr minima



.....

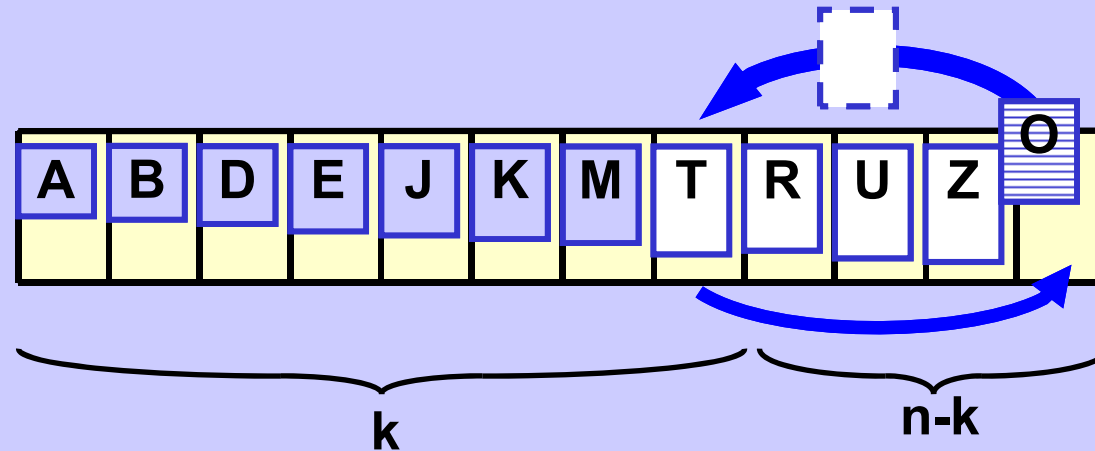
$(n-k)$  testů

Celkem  
testů

$$\sum_{k=1}^{n-1} (n-k) = \sum_{k=1}^{n-1} n - \sum_{k=1}^{n-1} k = n(n-1) - \frac{n(n-1)}{2} = \frac{1}{2}(n^2 - n)$$

## Selection sort

Krok k



přesuny .....

3

Celkem  
přesunů

$$\sum_{k=1}^{n-1} 3 = 3(n-1)$$

## Selection sort

### Shrnutí

**Celkem  
testů**

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

**Celkem  
přesunů**

$$3(n-1) = \Theta(n)$$

**Celkem  
operací**

$$\frac{1}{2}(n^2 - n) + 3(n-1) = \Theta(n^2)$$

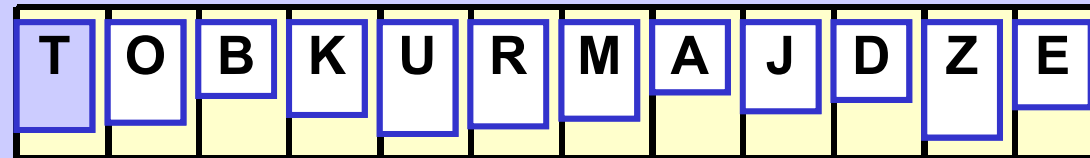
---

**Asymptotická složitost Selection Sortu je  $\Theta(n^2)$**

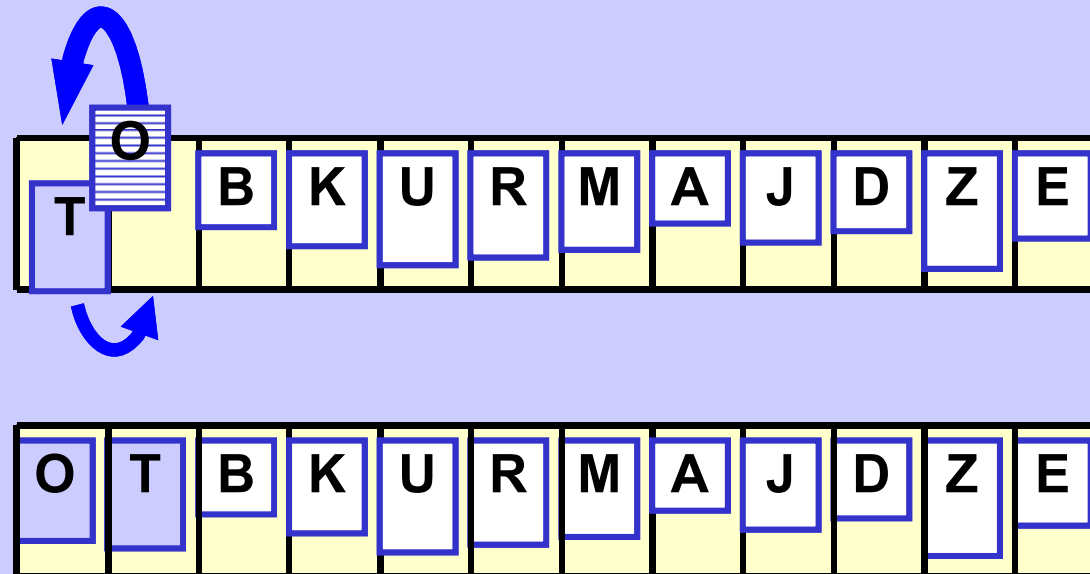


## Insertion sort

Start

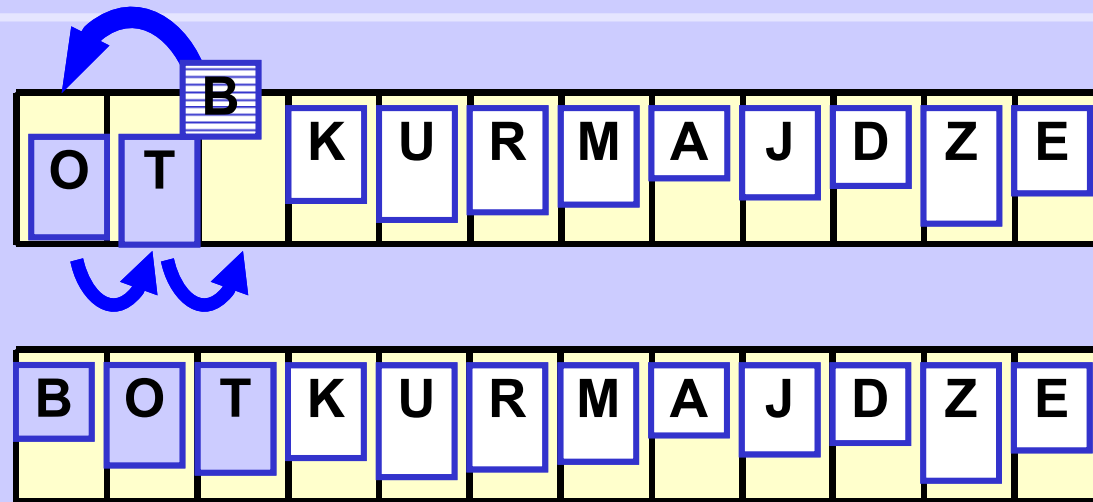


Krok1

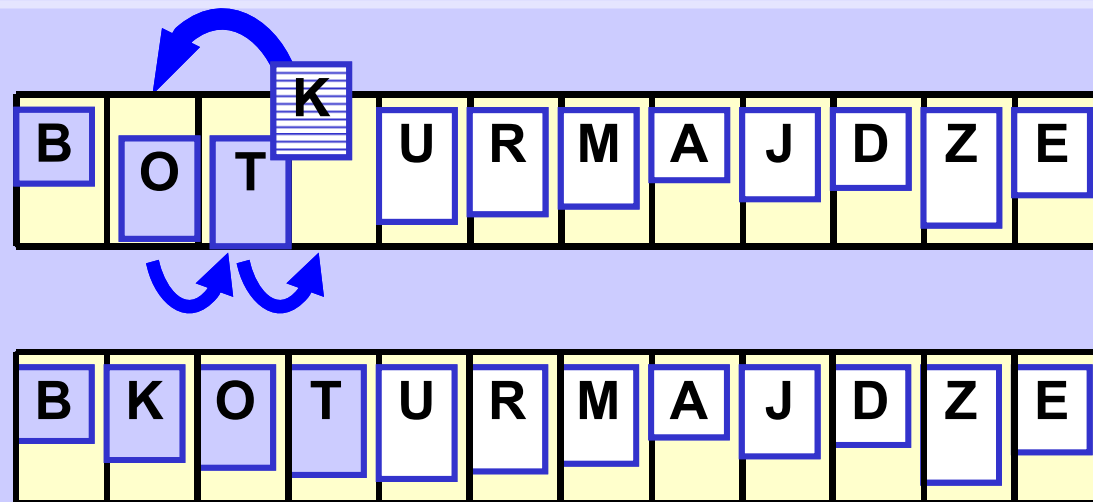


## Insertion sort

Krok 2



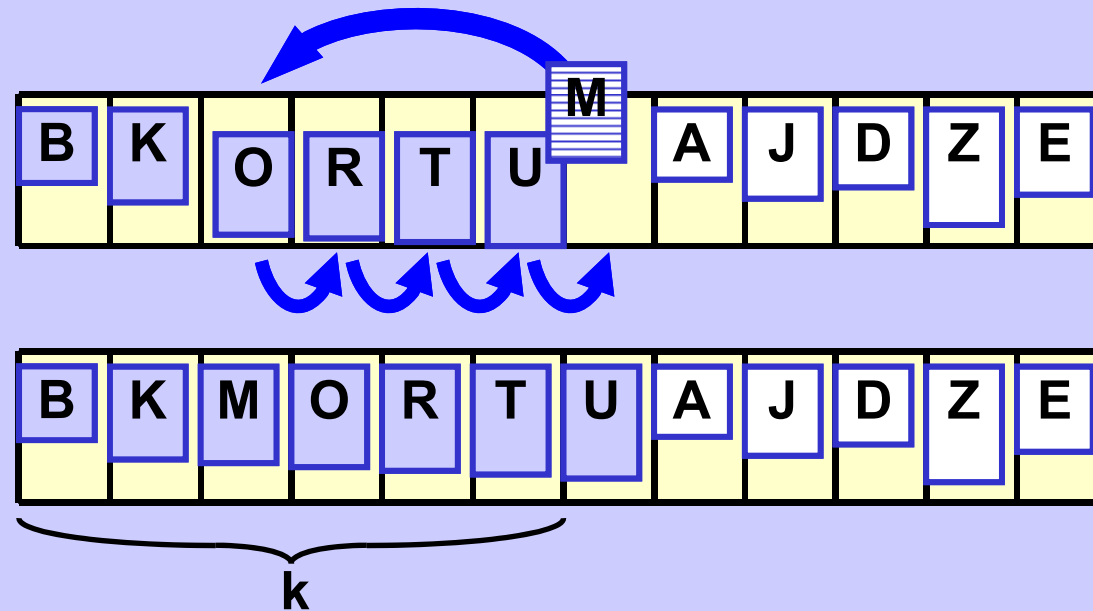
Krok 3



## Insertion sort

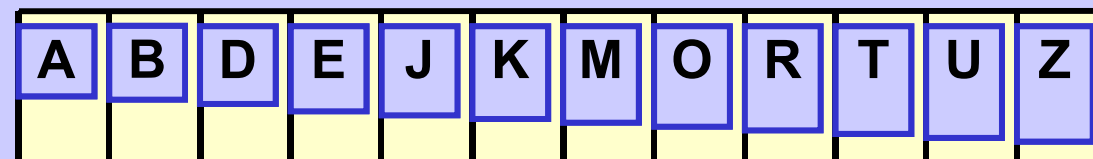
...

...

**Krok k**

...

...

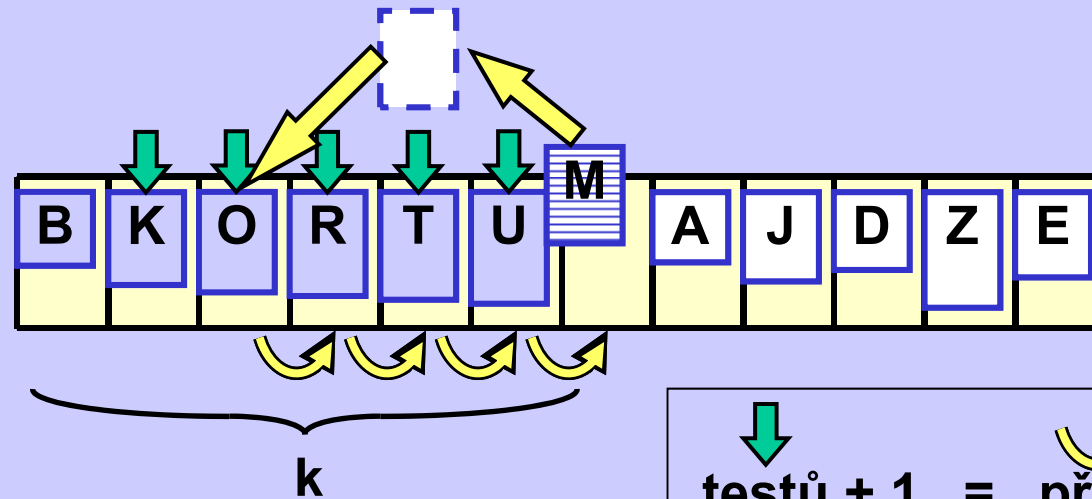
**Seřazeno**




## Insertion sort

```
for( int i = 1; i < n; i++ ){  
  
    // find & make place for a[i]  
    insVal = a[i];  
    int j = i-1;  
    while( (j >= 0) && (a[j] > insVal) ){  
        a[j+1] = a[j];  
        j--;  
    }  
  
    // insert a[i]  
    a[j+1] = insVal;  
}
```

# Insertion sort

Krok k




 $\text{testů} + 1 =$ 


 $\text{přesunů}$

testů .....	1 k $(k+1)/2$	nejlepší případ nejhorší případ průměrný případ
přesunů .....	2 k+1 $(k+3)/2$	nejlepší případ nejhorší případ průměrný případ

## Insertion sort

### Shrnutí

Celkem  
testů

$n - 1$	$= \Theta(n)$	nejlepší případ
$(n^2 - n)/2$	$= \Theta(n^2)$	nejhorší případ
$(n^2 + n - 2)/4$	$= \Theta(n^2)$	průměrný případ

Celkem  
přesunů

$2n - 2$	$= \Theta(n)$	nejlepší případ
$(n^2 + n - 2)/2$	$= \Theta(n^2)$	nejhorší případ
$(n^2 + 5n - 6)/4$	$= \Theta(n^2)$	průměrný případ

Asymptotická složitost Insertion Sortu je  $O(n^2)$  (!!)

# Bubble sort

Start

T	O	B	U	J	R	M	A	K	D	Z	E
---	---	---	---	---	---	---	---	---	---	---	---

Fáze 1

T	O	B	U	J	R	M	A	K	D	Z	E
---	---	---	---	---	---	---	---	---	---	---	---

Blue arrows indicate a swap between 'T' and 'O'.

O	T	B	U	J	R	M	A	K	D	Z	E
---	---	---	---	---	---	---	---	---	---	---	---

Blue arrows indicate a swap between 'T' and 'B'.

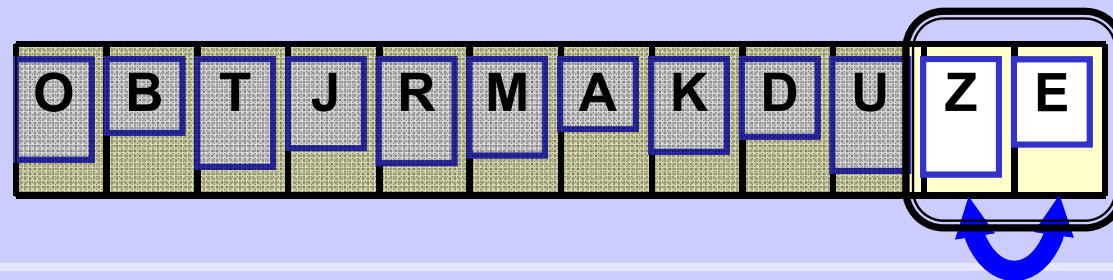
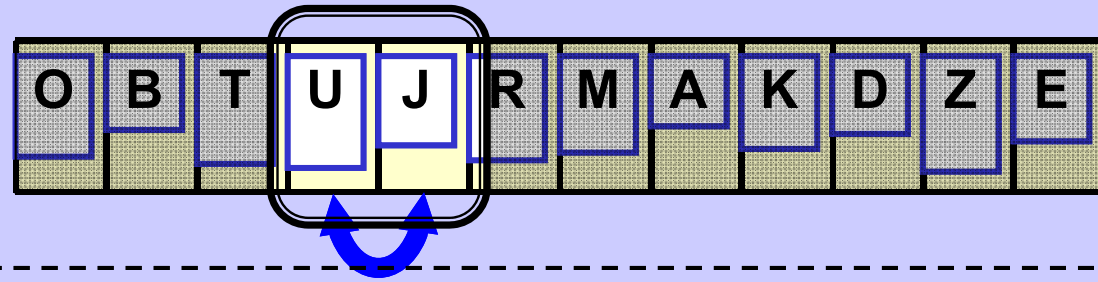
O	B	T	U	J	R	M	A	K	D	Z	E
---	---	---	---	---	---	---	---	---	---	---	---

Blue arrows indicate a swap between 'T' and 'U'.

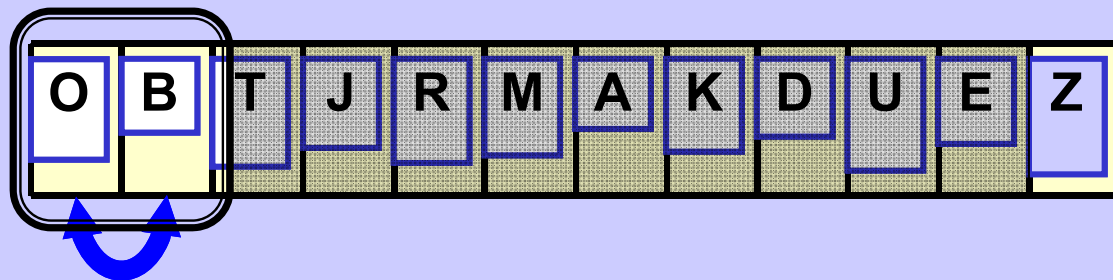


# Bubble sort

Fáze 1



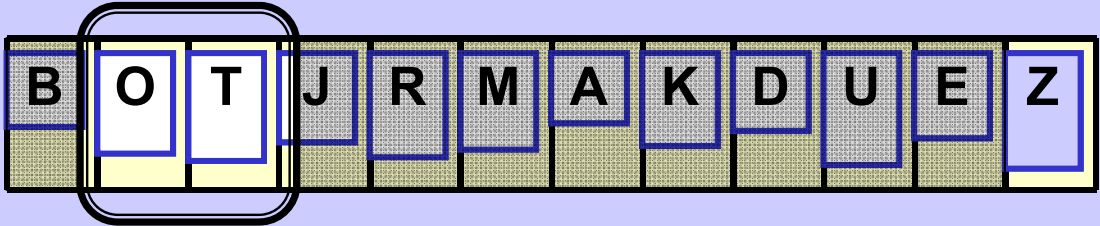
Fáze 2



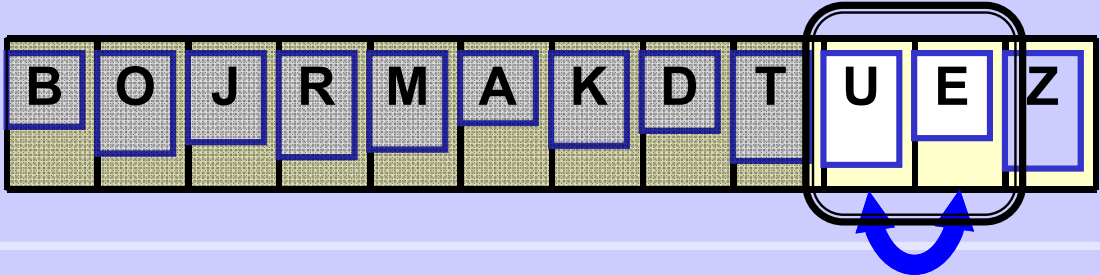


# Bubble sort

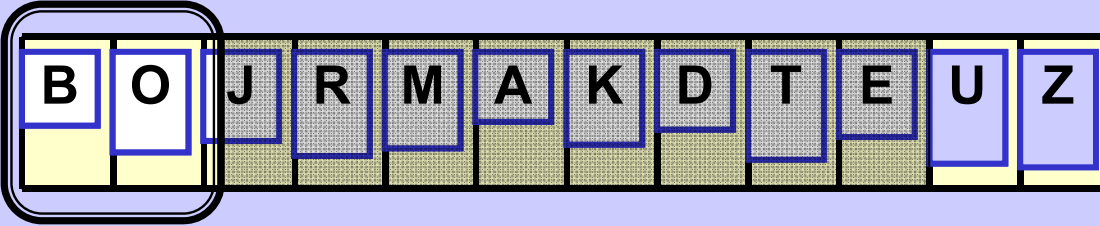
Fáze 2



...etc...



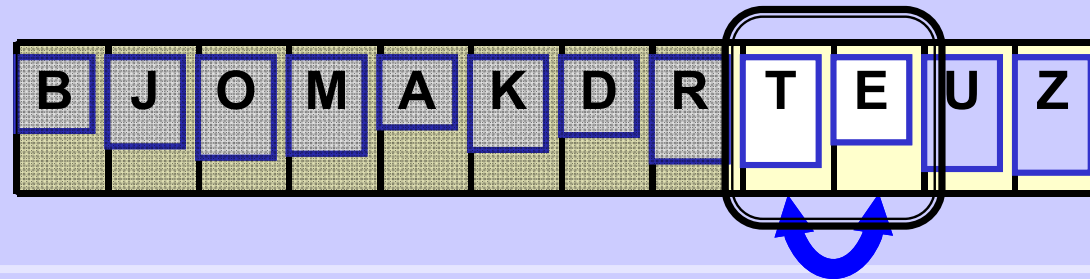
Fáze 3



# Bubble sort

Fáze 3

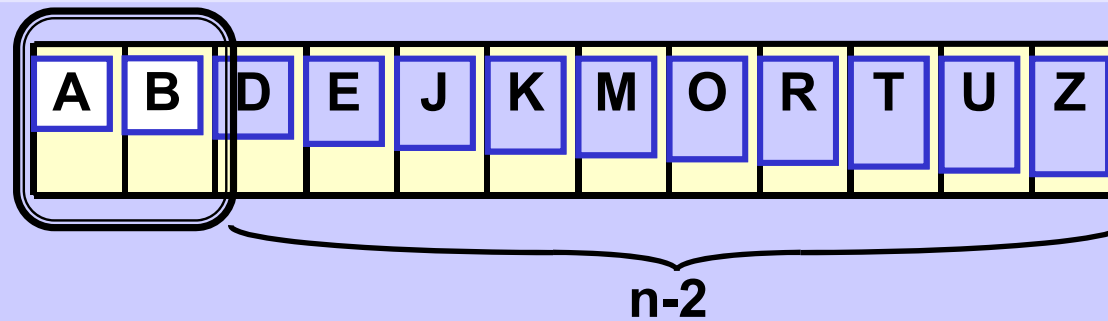
... atd ...



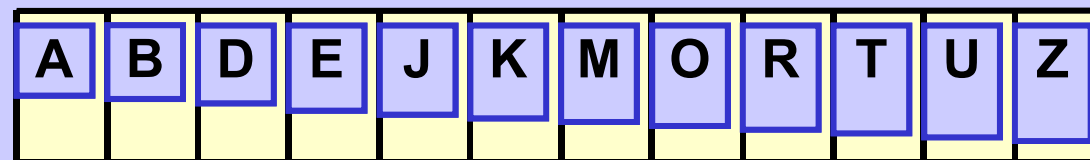
...

...

Fáze n-1



Seřazeno



## Insertion sort

```

for( int lastPos = n-1; lastPos > 0; lastPos-- )
    for( int j = 0; j < lastPos; j++ )
        if( a[j] > a[j+1] ) swap( a, j, j+1 );

```

### Shrnutí

Celkem  
testů

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

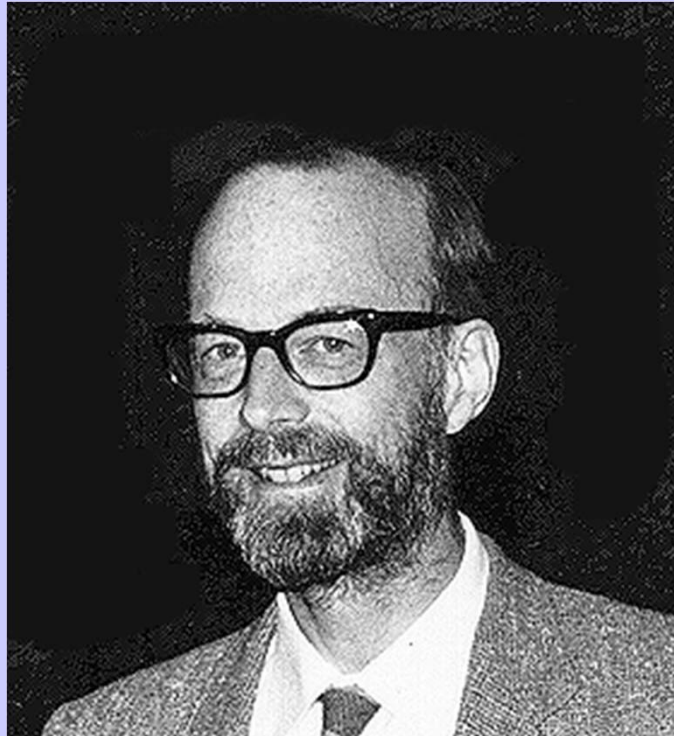
Celkem  
přesunů

$$0 = \Theta(1) \quad \text{nejlepší případ}$$

$$\frac{1}{2}(n^2 - n) = \Theta(n^2) \quad \text{nejhorší případ}$$

Asymptotická složitost Bubble Sortu je  $\Theta(n^2)$

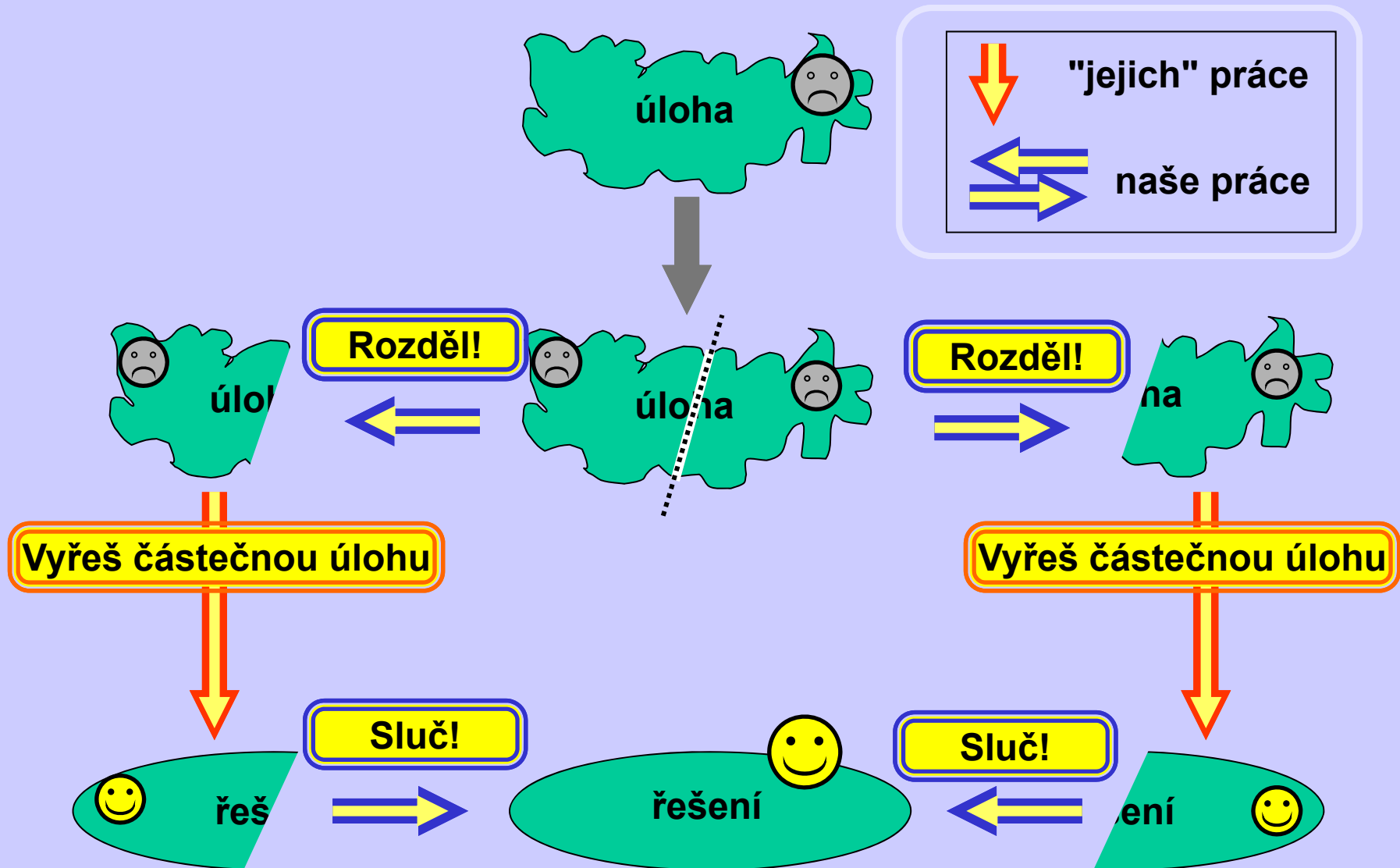
## Quicksort



**Sir Charles Antony Richard Hoare**

**C. A. R. Hoare: Quicksort. Computer Journal, Vol. 5, 1, 10-15 (1962)**

# Rozděl a panuj! Divide and conquer! Divide et impera!



# Quicksort

Myšlenka

Start



Malá



Velká



Divide & Conquer!

M A K D R B T O J U Z E

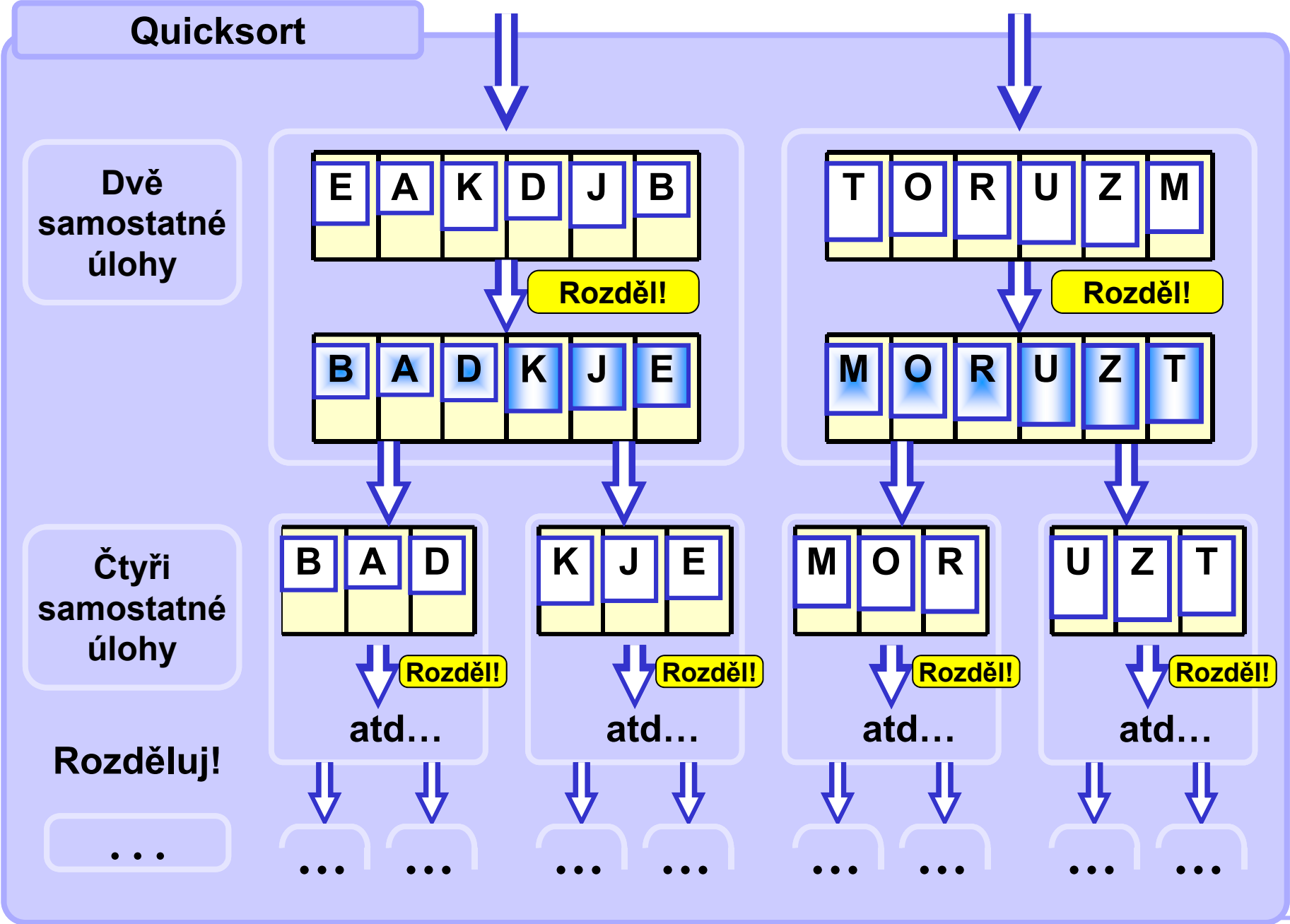
M A K D R B T O J U Z E

Rozděl!

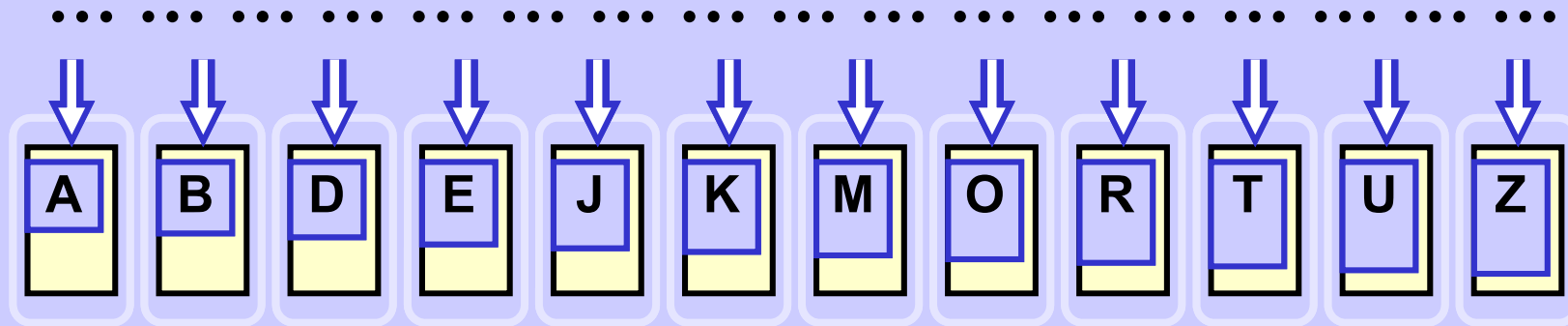
E A K D J B T O R U Z M

Malá

Velká



## Quicksort



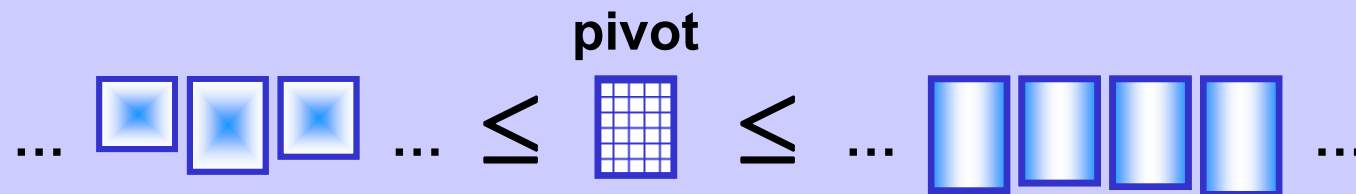
**Opanováno!**



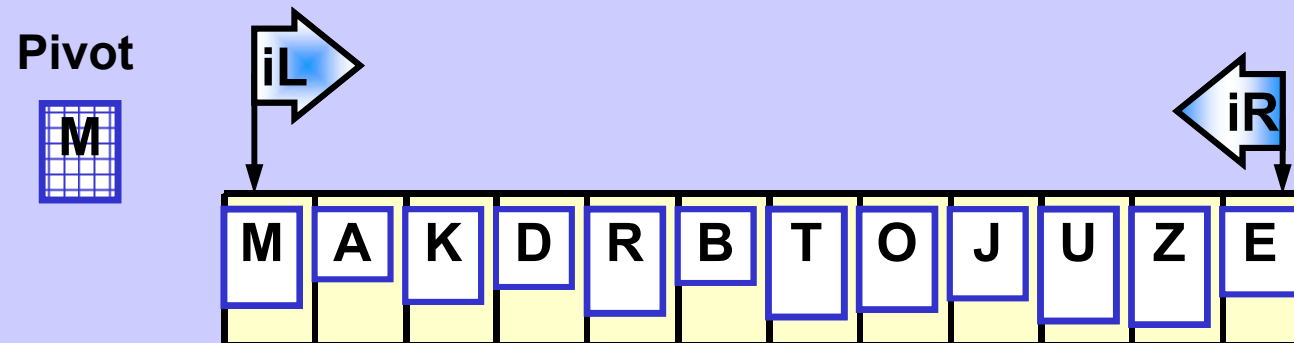
# Quicksort

## Dělení

Pivot



Init



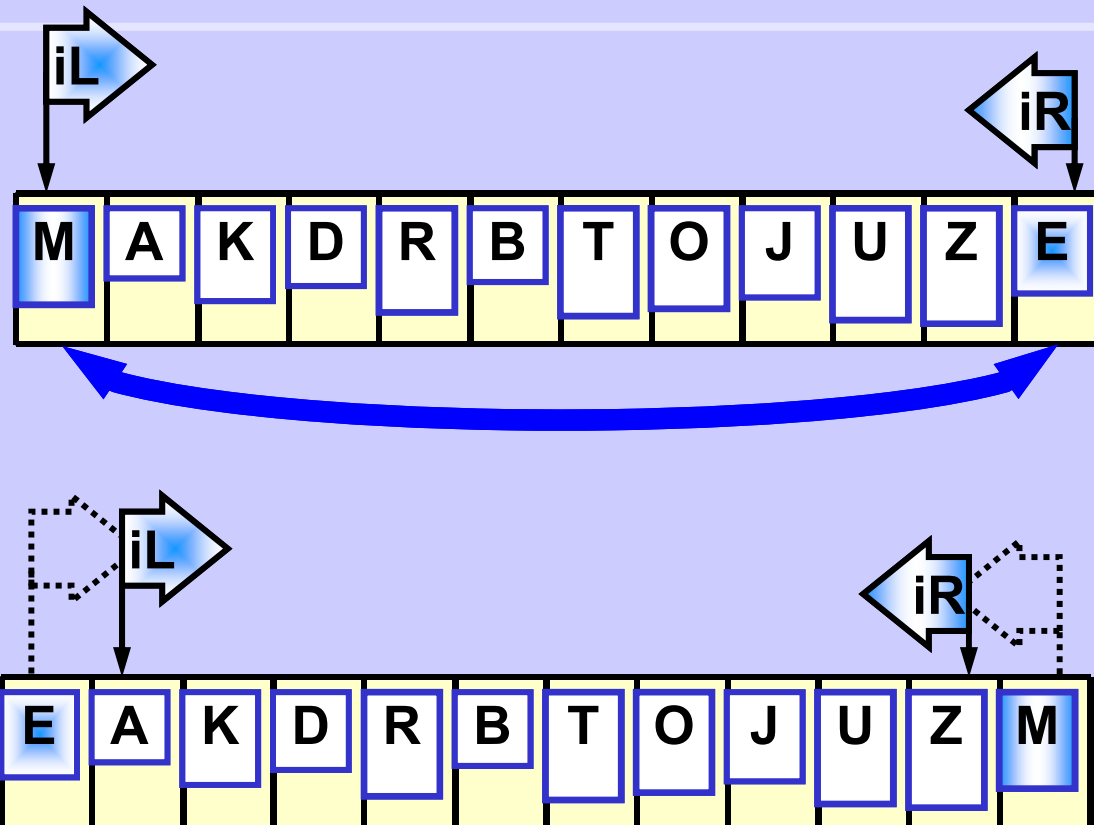
# Quicksort

## Dělení

Krok 1

Pivot

M



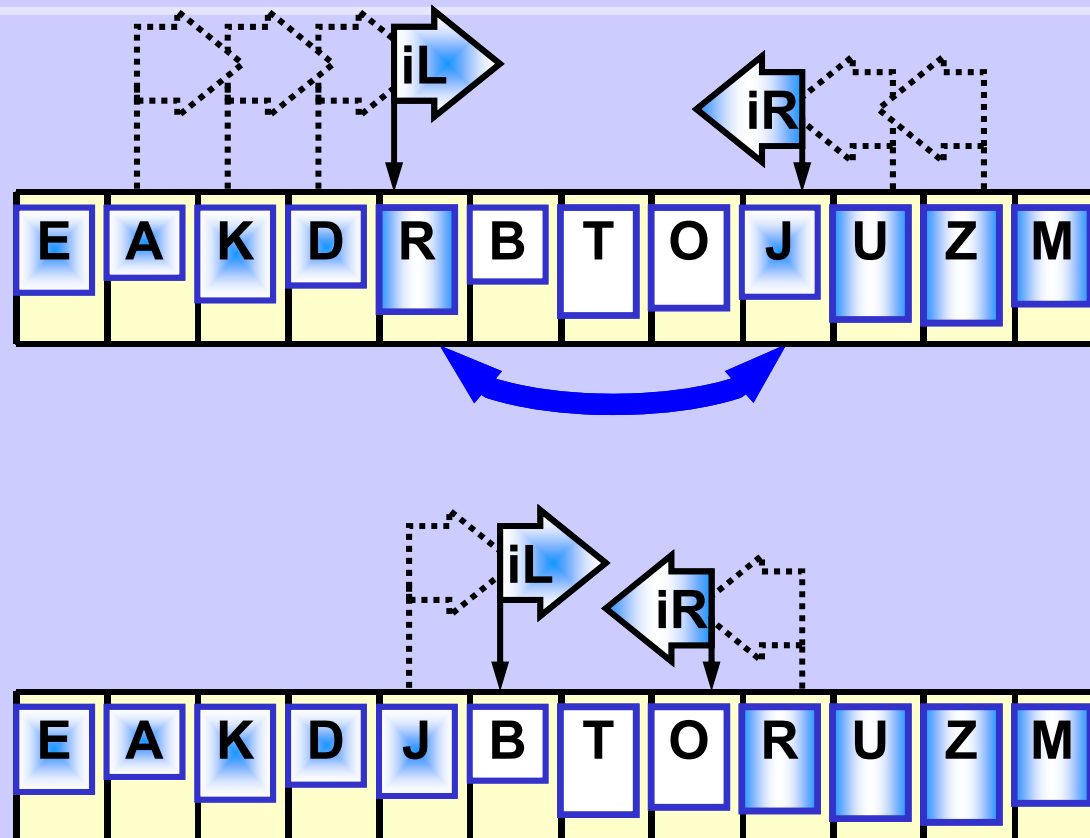
## Quicksort

## Dělení

Krok 2

Pivot

M

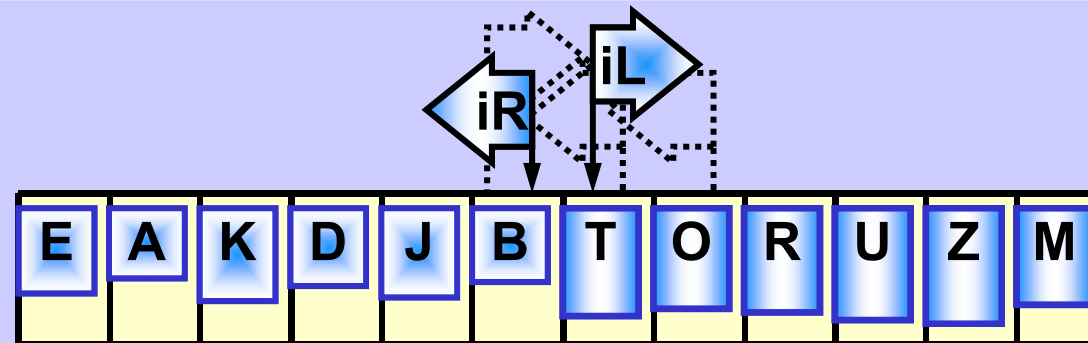


# Quicksort

## Dělení

Krok 3

Pivot

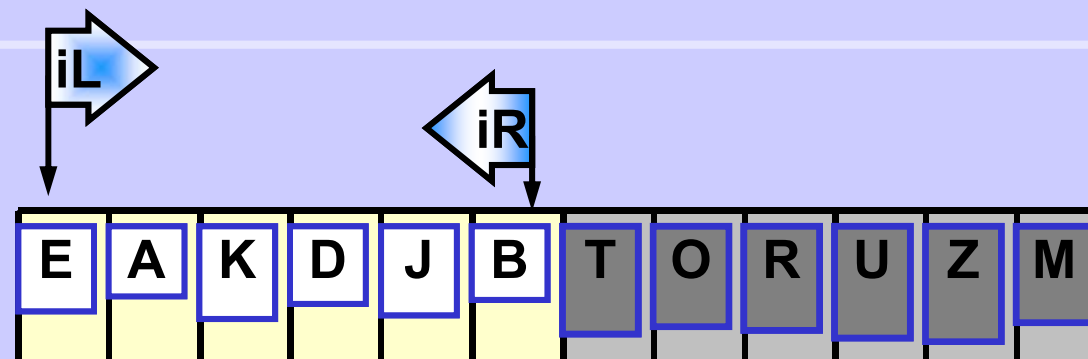


$iR < iL$  Stop

**Rozdě!**

Init

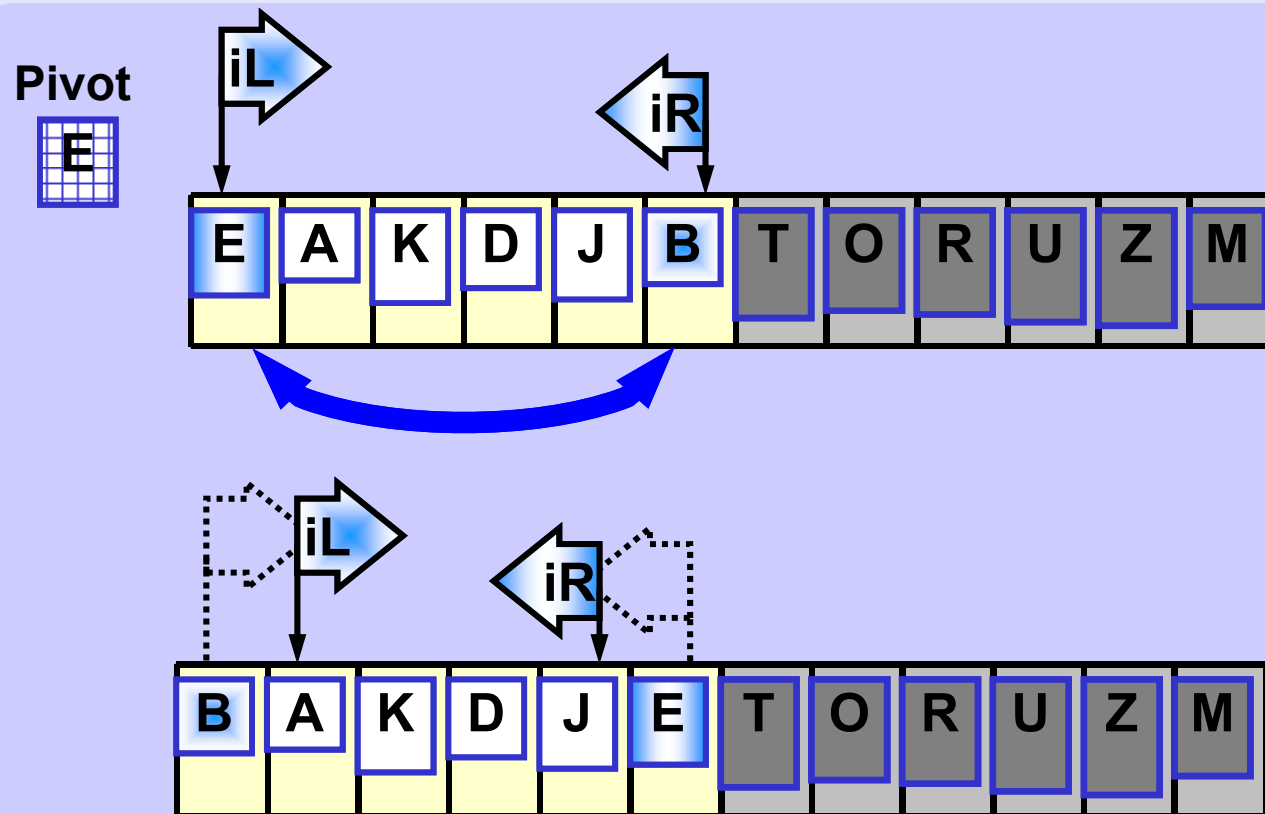
Pivot



# Quicksort

## Dělení

Krok 1

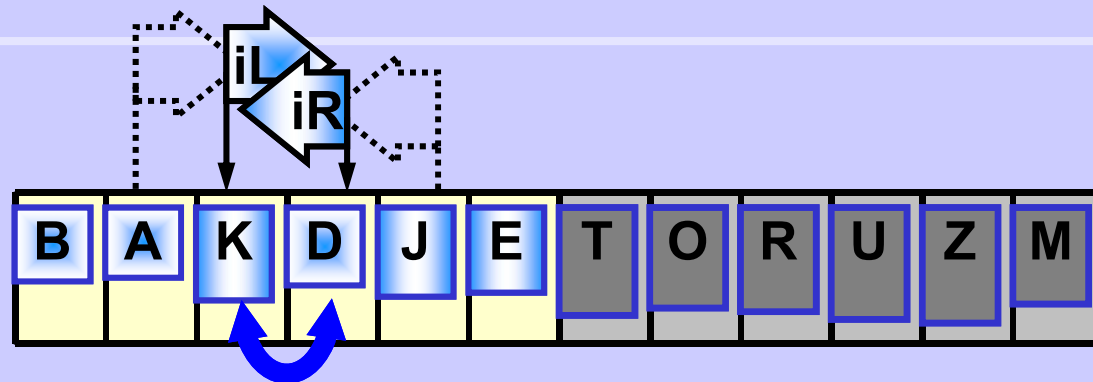


# Quicksort

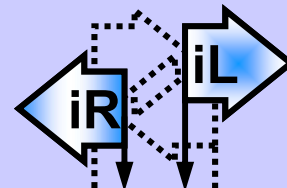
## Dělení

Pivot

E

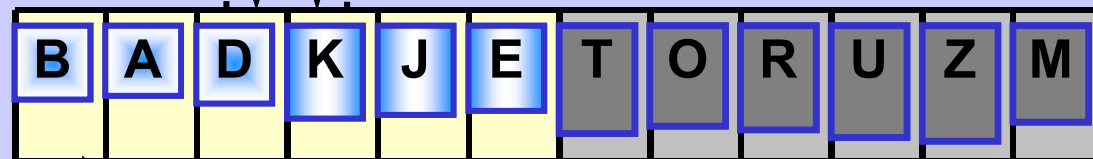


Krok 2



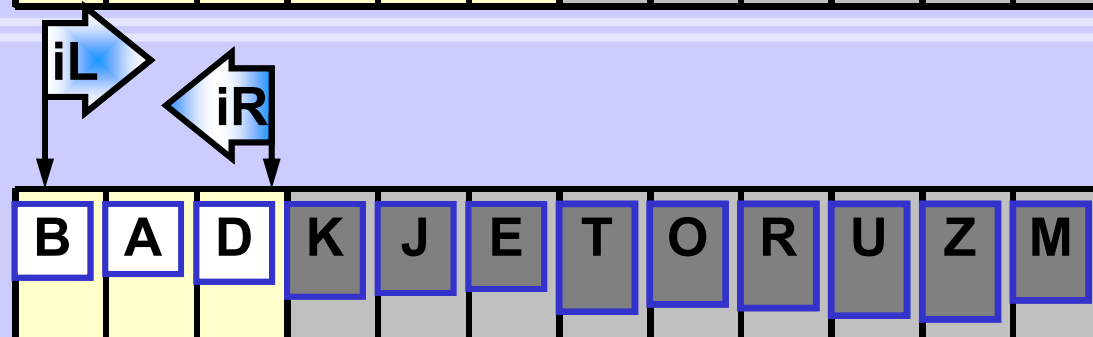
$iR < iL$  Stop

Rozdě!



Pivot

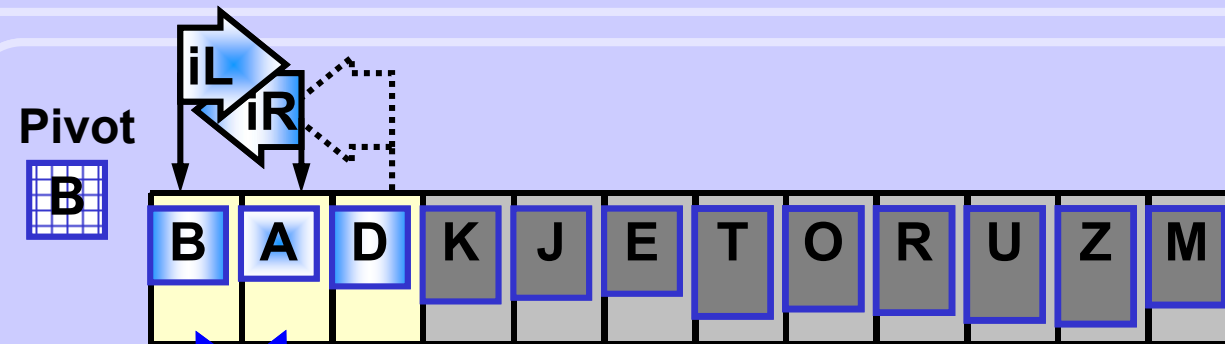
B



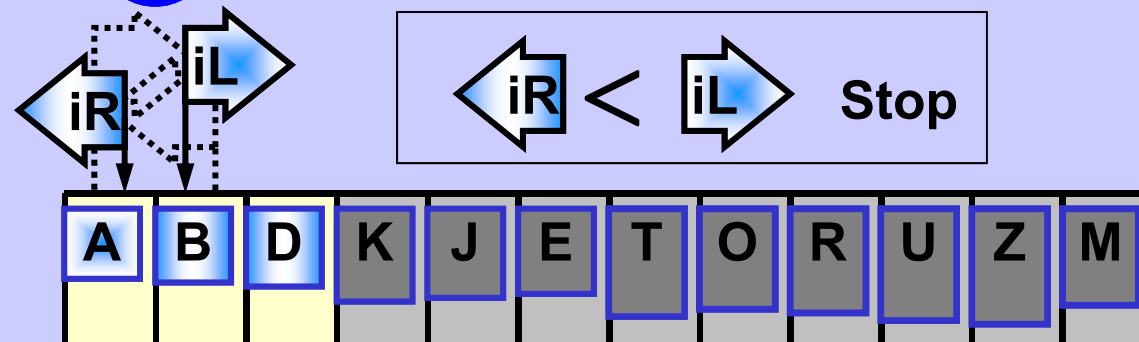
Init

# Quicksort

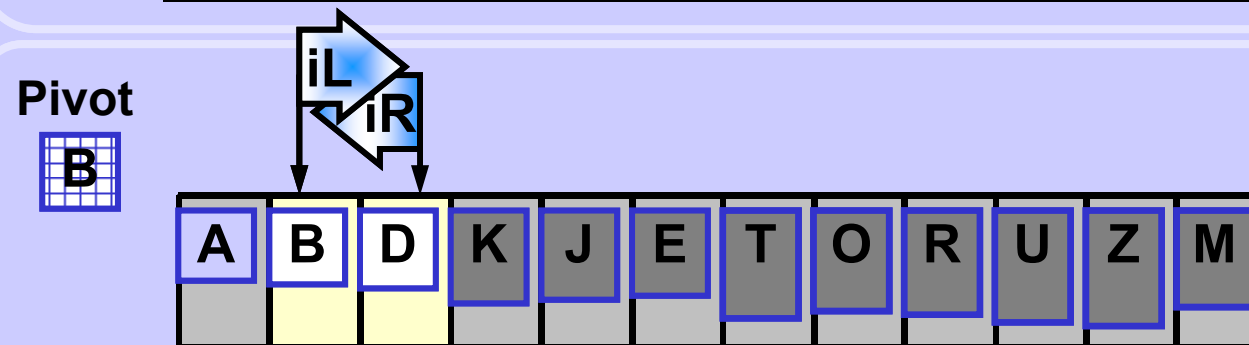
## Dělení



Krok 1



Rozdě!



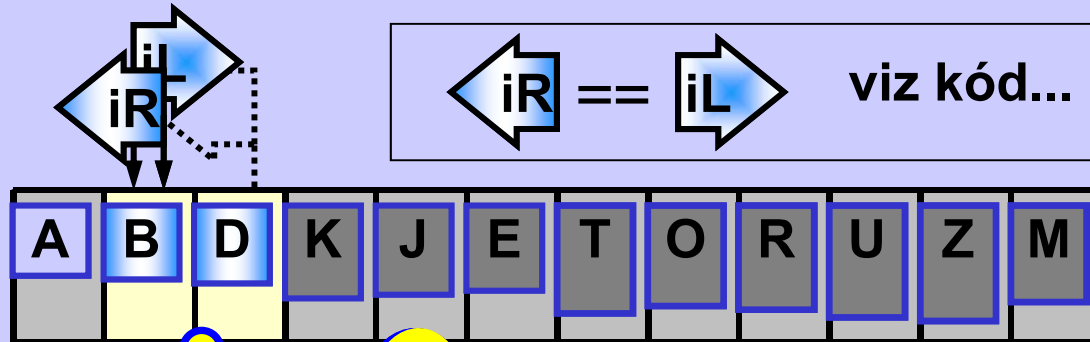
Init

# Quicksort

## Dělení

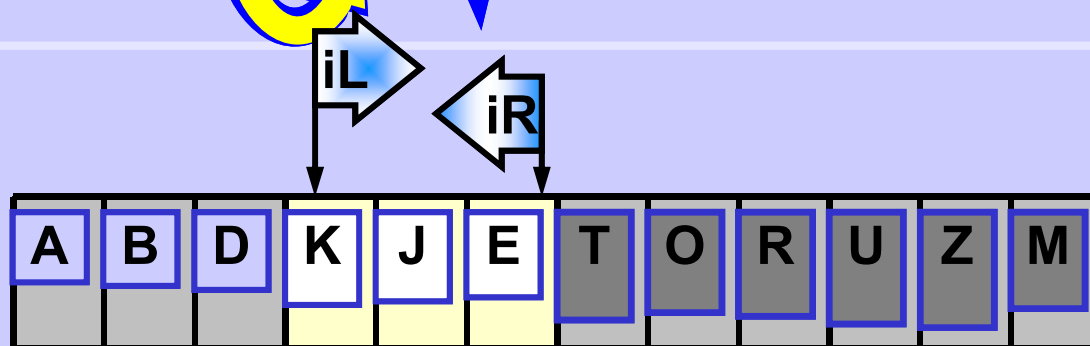
Krok 1

Pivot



Další oddíl

Pivot



Init

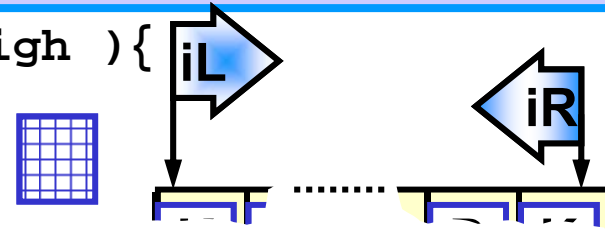
atd...

atd...

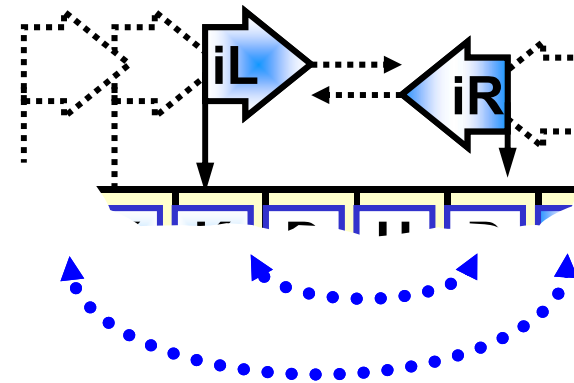


## Quicksort

```
void qSort( Item a[], int low, int high ){
    int iL = low, iR = high;
    Item pivot = a[low];
```



```
do {
    while( a[iL] < pivot ) iL++;
    while( a[iR] > pivot ) iR--;
    if( iL < iR ) {
        swap(a, iL, iR);
        iL++; iR--;
    }
    else
        if( iL == iR ) {iL++; iR--;}
} while( iL <= iR );
```



```
if( low < iR ) qSort( a, low, iR );
if( iL < high ) qSort( a, iL, high );
}
```

**Rozděli!**

## Quicksort

Levý index se nastaví na začátek zpracovávaného úseku pole, pravý na jeho konec, zvolí se pivot.

Cyklus (rozdělení na „malé“ a „velké“) :

Levý index se pohybuje doprava  
a zastaví se na prvku větším nebo rovném pivotovi.

Pravý index se pohybuje doleva  
a zastaví se na prvku menším nebo rovném pivotovi.

Pokud je levý index ještě před pravým,  
příslušné prvky se prohodí,  
a oba indexy se posunou o 1 ve svém směru.

Jinak pokud se indexy rovnají,  
jen se oba posunou o 1 ve svém směru.

Cyklus se opakuje, dokud se indexy neprekříží,  
tj. pravý se dostane před levého.

Následuje rekurzivní volání (zpracování „malých“ a „velkých“ zvlášť)  
na úsek od začátku do pravého(!) indexu včetně  
a na úsek od levého(!) indexu včetně až do konce,  
má-li příslušný úsek délku větší než 1.

## Quicksort

### Asymptotická složitost

Celkem  
přesunů a testů

$$\Theta(n \cdot \log_2(n))$$

nejlepší případ

$$\Theta(n \cdot \log_2(n))$$

průměrný případ

$$\Theta(n^2)$$

nejhorší případ

Asymptotická složitost Quick Sortu je  $O(n^2)$ , ...

... ale! :

“Očekávaná” složitost Quick Sortu je  $\Theta(n \cdot \log_2(n))$  (!!)

## Quicksort



## Porovnání efektivity



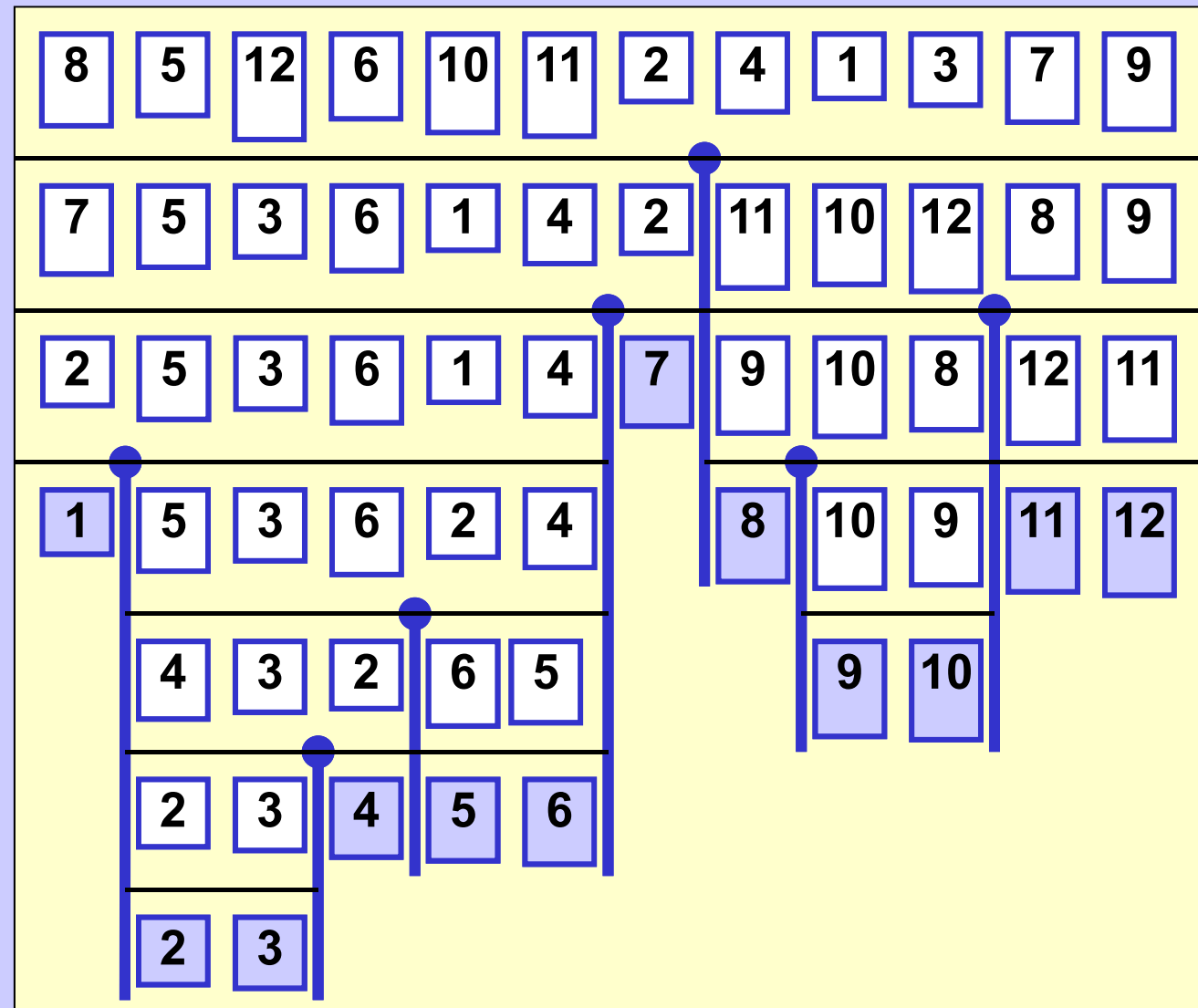
N	$N^2$	$N \times \log_2(N)$	$\frac{N^2}{N \times \log_2(N)}$	zpoma- lení (1~1sec)
1	1	0		
10	100	33.2	3.0	3 sec
100	10 000	6 64.4	15.1	15 sec
1 000	1 000 000	9 965.8	100.3	1.5 min
10 000	100 000 000	132 877.1	752.6	13 min
100 000	10 000 000 000	1 660 964.0	6 020.6	1.5 hod
1 000 000	1 000 000 000 000	19 931 568.5	50 171.7	14 hod
10 000 000	100 000 000 000 000	232 534 966.6	430 042.9	5 dnů

tab. 1

# Quicksort

Ukázka  
průběhu

pivot =  
= první  
v úseku



## Quicksort



Schéma rekurzivního volání  
Quick sortu  
má podobu pravidelného  
binárního kořenového stromu.

## Stabilita řazení

Stabilní řazení nemění pořadí prvků se stejnou hodnotou.

Neseřazená data

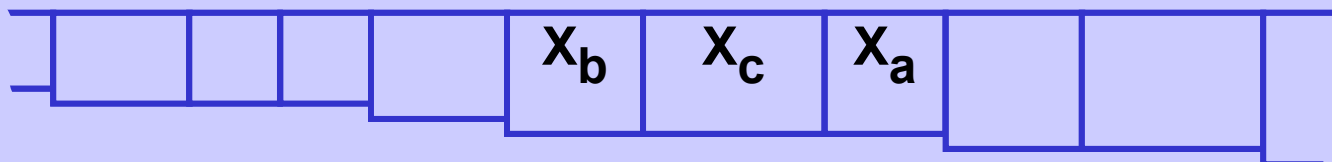


Hodnoty  $X_i$  jsou totožné



Seřad'

Seřazená data



## Stabilita řazení

B<sub>1</sub> D<sub>1</sub> C<sub>1</sub> A<sub>1</sub> C<sub>2</sub> B<sub>2</sub> A<sub>2</sub> D<sub>2</sub> B<sub>3</sub> A<sub>3</sub> D<sub>3</sub> C<sub>3</sub>

Insert Bubble -- Stabilní implementace



A<sub>1</sub> A<sub>2</sub> A<sub>3</sub> B<sub>1</sub> B<sub>2</sub> B<sub>3</sub> C<sub>1</sub> C<sub>2</sub> C<sub>3</sub> D<sub>1</sub> D<sub>2</sub> D<sub>3</sub>

B<sub>1</sub> D<sub>1</sub> C<sub>1</sub> A<sub>1</sub> C<sub>2</sub> B<sub>2</sub> A<sub>2</sub> D<sub>2</sub> B<sub>3</sub> A<sub>3</sub> D<sub>3</sub> C<sub>3</sub>

Insert Bubble -- Nestabilní implementace



QuickSort Vždy nestabilní!!  
Select Sort

A<sub>2</sub> A<sub>1</sub> A<sub>3</sub> B<sub>2</sub> B<sub>3</sub> B<sub>1</sub> C<sub>3</sub> C<sub>1</sub> C<sub>2</sub> D<sub>3</sub> D<sub>2</sub> D<sub>1</sub>



## Stabilní řazení

Záznam: 

Jméno	Příjmení
-------	----------

Vstup: Seznam seřazen  
pouze podle jména.

Andrew	Cook
Andrew	Amundsen
Andrew	Brown
Barbara	Cook
Barbara	Brown
Barbara	Amundsen
Charles	Amundsen
Charles	Cook
Charles	Brown

stabilní řazení  
Seřad' záznamy  
pouze podle"  

Příjmení
----------

Výstup: Seznam seřazen  
podle jména i příjmení.

Andrew	Amundsen
Barbara	Amundsen
Charles	Amundsen
Andrew	Brown
Barbara	Brown
Charles	Brown
Andrew	Cook
Barbara	Cook
Charles	Cook

Pořadí záznamů se stejným příjmením se nezměnilo

## Nestabilní řazení

Záznam: 

Jméno	Příjmení
-------	----------

Vstup: Seznam seřazen pouze podle jména.

Andrew	Cook
Andrew	Amundsen
Andrew	Brown
Barbara	Cook
Barbara	Brown
Barbara	Amundsen
Charles	Amundsen
Charles	Cook
Charles	Brown

QuickSort



Seřad' záznamy pouze podle:

Příjmení
----------

Výstup: Původní pořadí jmen je ztraceno.

seřazeno

Barbara	Amundsen
Andrew	Amundsen
Charles	Amundsen
Barbara	Brown
Charles	Brown
Andrew	Brown
Charles	Cook
Andrew	Cook
Barbara	Cook

Pořadí záznamů se stejným příjmením se změnilo.