



Algoritmizace

Marko Genyk-Berezovskyj, Daniel Průša

2010 – 2023

Přehled

- Selection, Insertion a Bubble sort
- Quicksort
- Stabilita řazení
- Šestá domácí úloha



Z minula

Operace	B-strom řádu k s n uzly
Find	$\Theta(k \cdot \log_k n)$
Insert	$\Theta(k \cdot \log_k n)$
Delete	$\Theta(k \cdot \log_k n)$



Jakou má Find asymptotickou složitost při použití binárního vyhledávání v uzlech?

$$(\log_2 k) \cdot (\log_k n) = (\log_2 k) \cdot \frac{\log_2 n}{\log_2 k} = \log_2 n$$

slido



**Join at slido.com
#403516**

① Start presenting to display the joining instructions on this slide.



Audience Q&A Session

① Start presenting to display the audience questions on this slide.

slido



**Které z uvedených
řadících algoritmů byste
z paměti naprogramovali?**

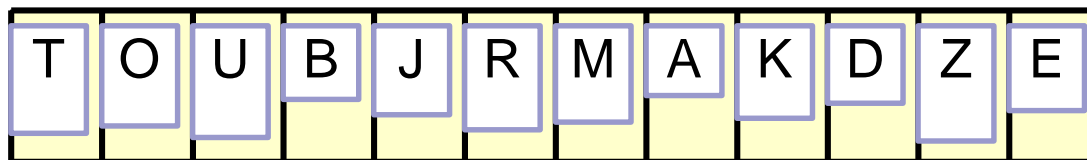
ⓘ Start presenting to display the poll results on this slide.

Selection sort (řazení výběrem)

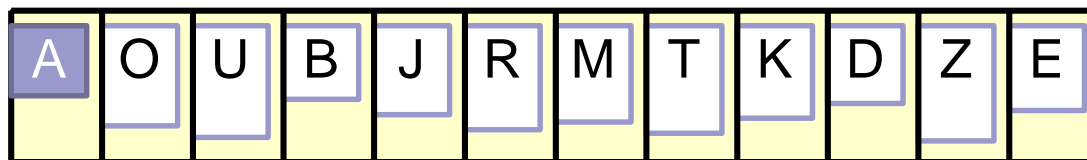
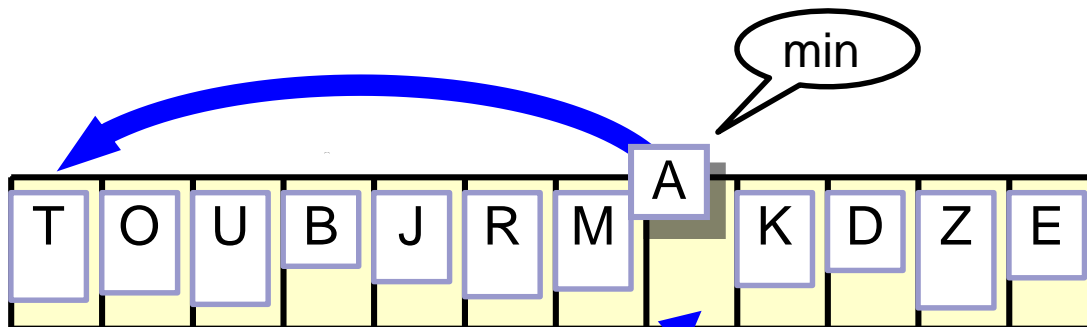
Neseřazeno

Seřazeno

Start

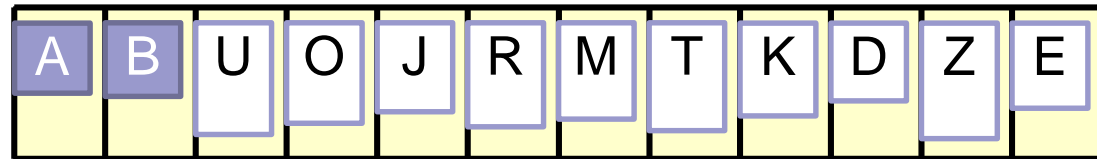
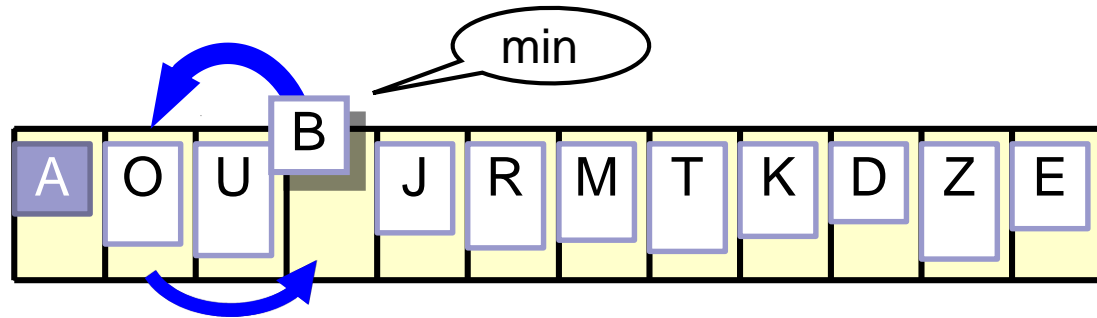


Krok1

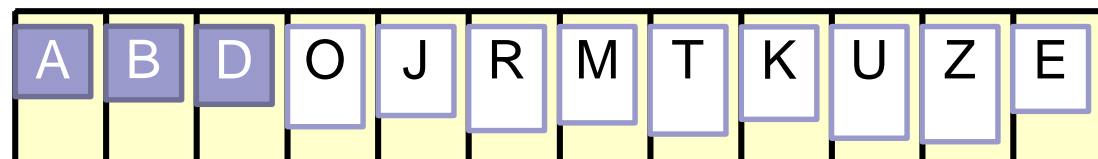
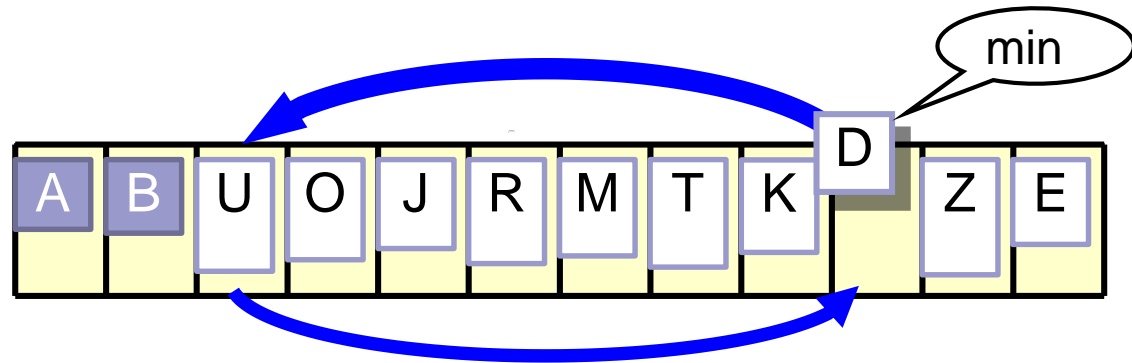


Selection sort

Krok 2



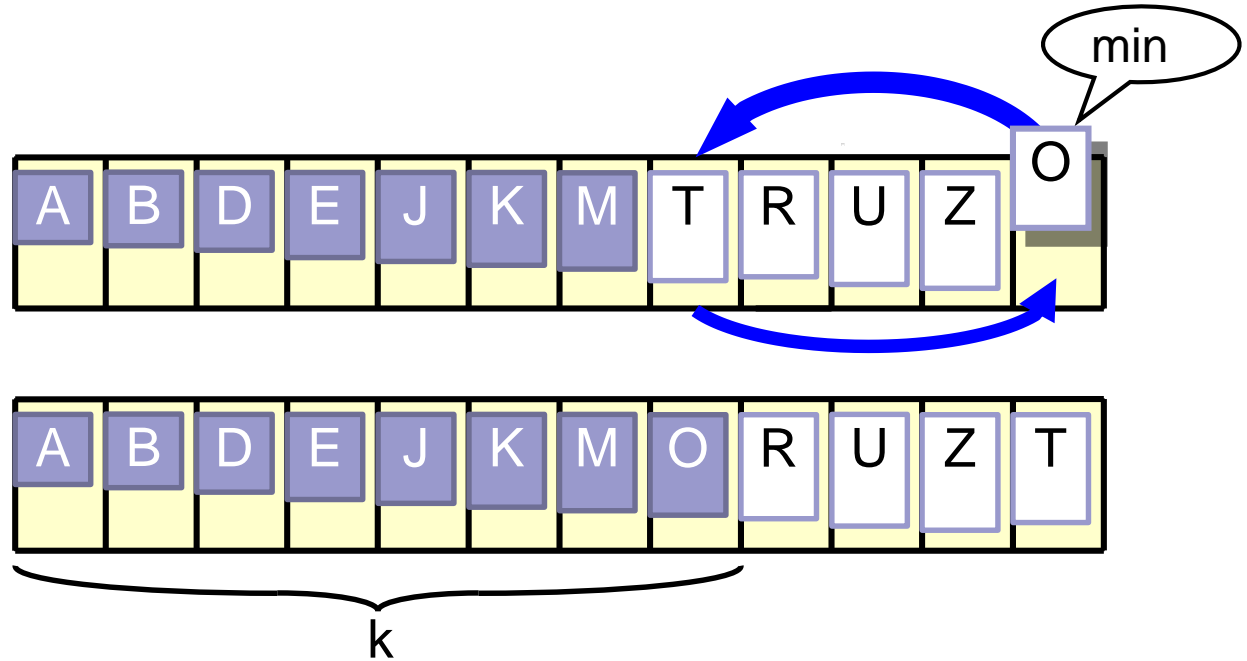
Krok 3



Selection sort

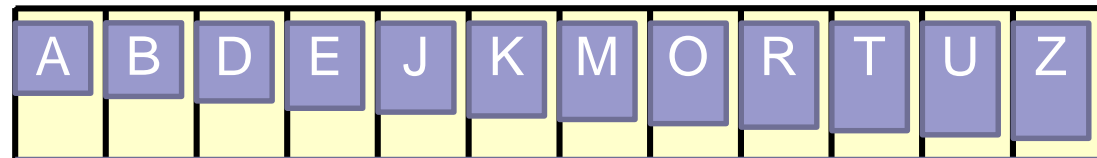
...

Krok k



...

Seřazeno

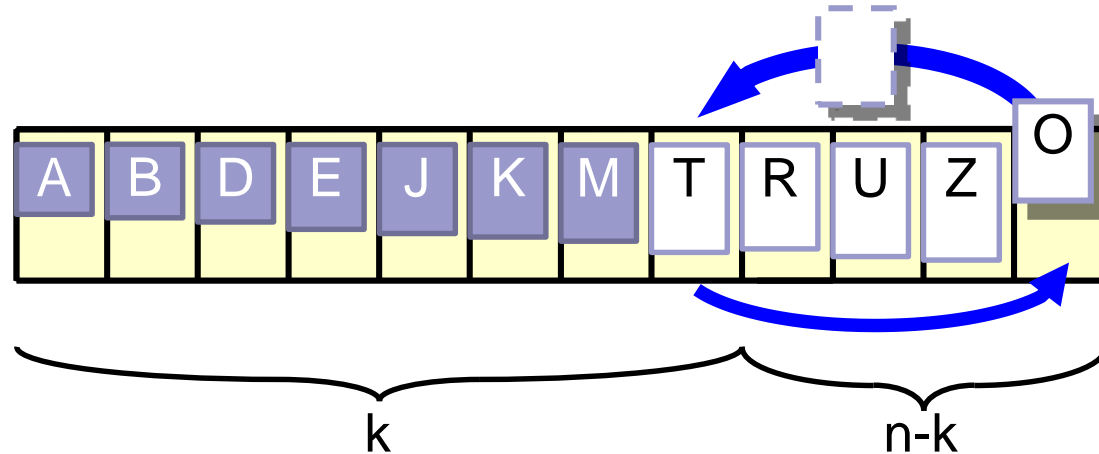


Selection sort

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

Selection sort

Krok k



přesuny ... 3, počet testů (porovnání) ... n-k

Celkem
přesunů

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

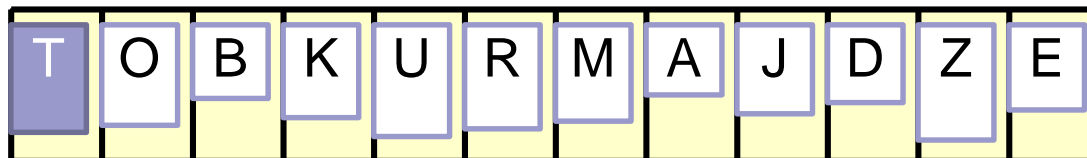
Celkem
testů

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

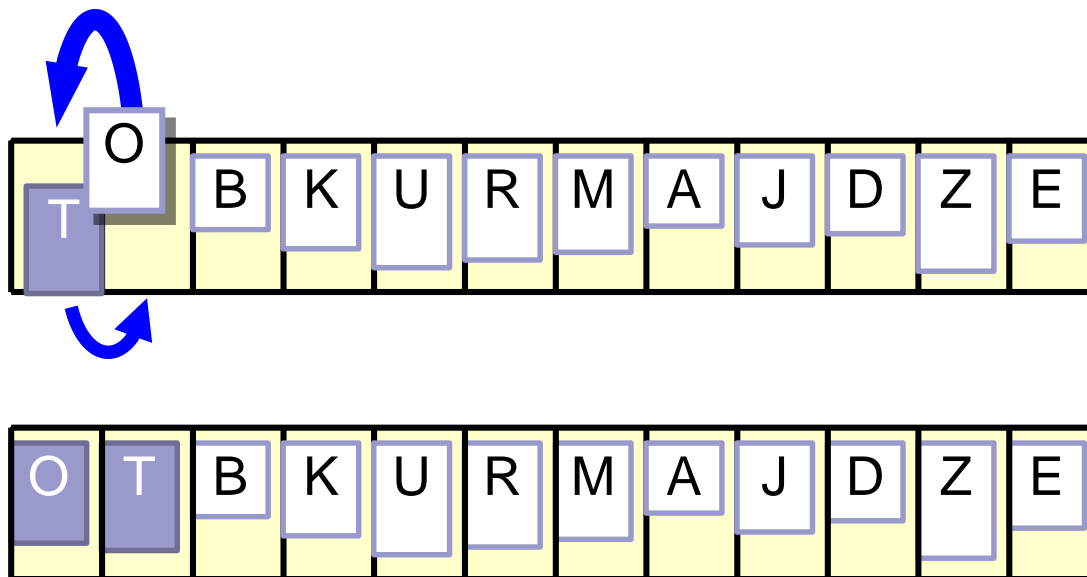
Asymptotická složitost Selection sortu je $\Theta(n^2)$.

Insertion sort (řazení vkládáním)

Start

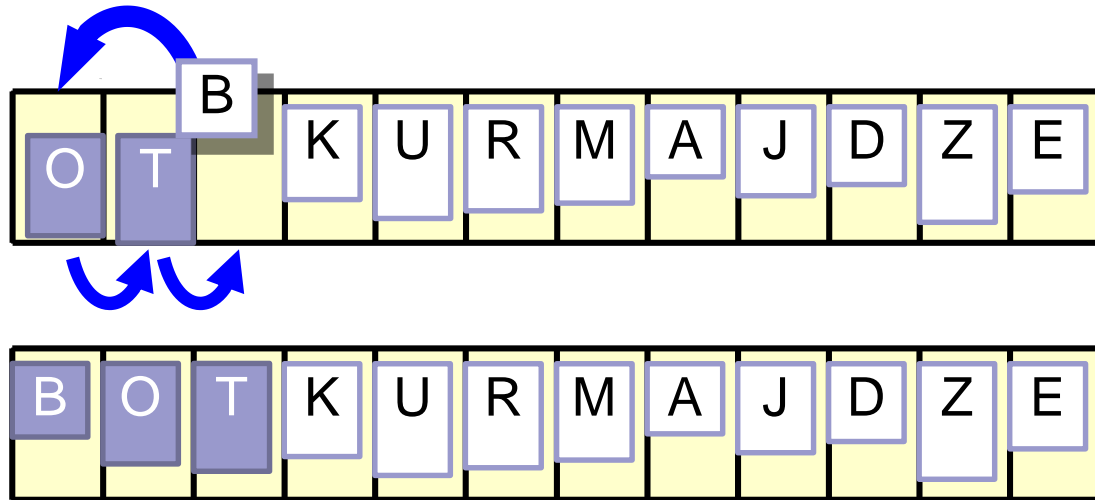


Krok1

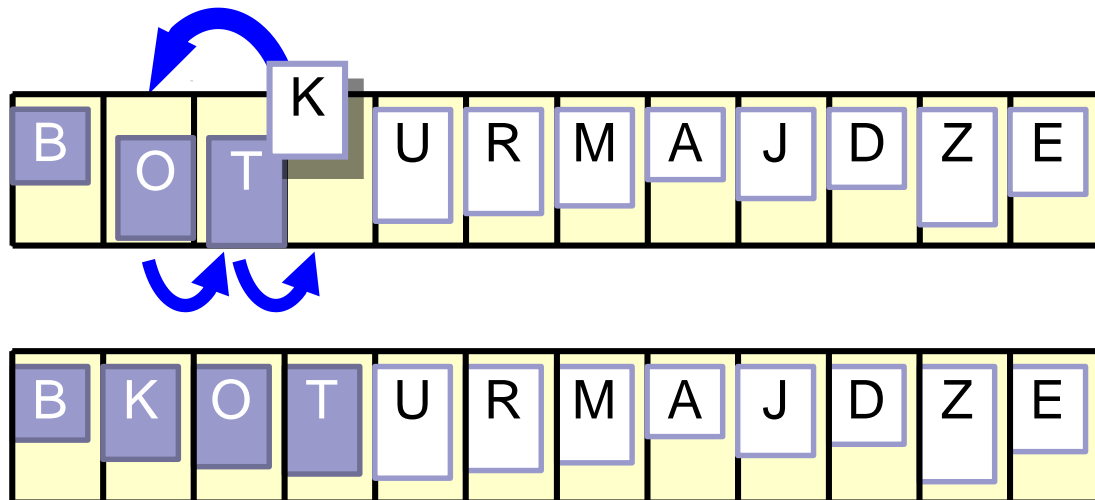


Insertion sort

Krok 2



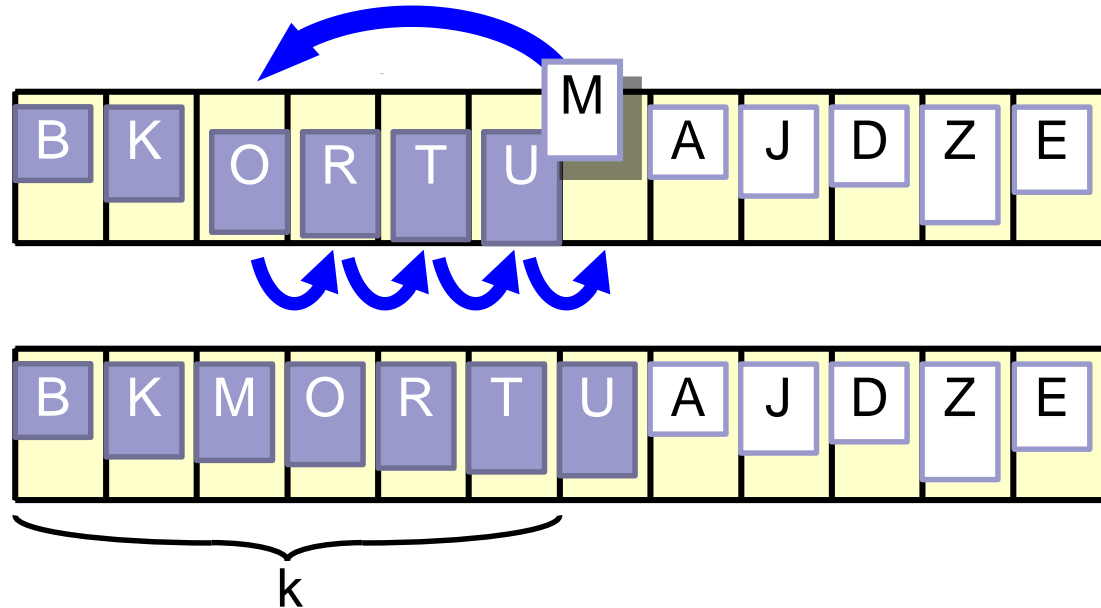
Krok 3



Insertion sort

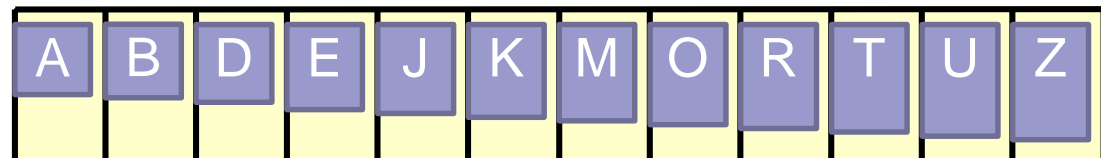
...

Krok k



...

Seřazeno

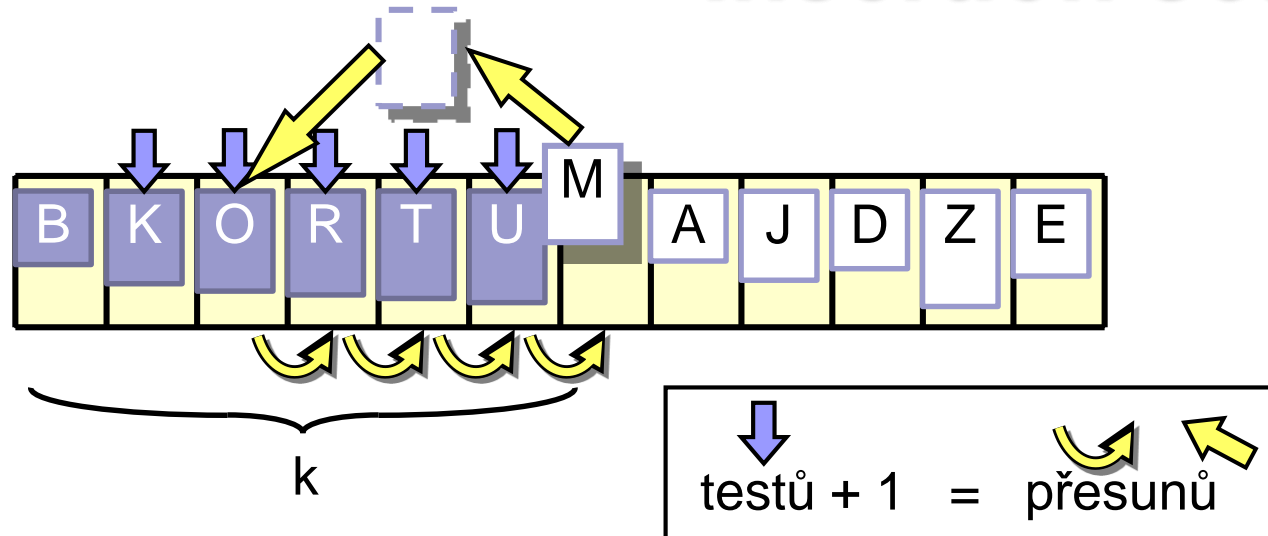


Insertion sort

```
for (i = 1; i < n; i++) {  
    // find & make  
    // place for a[i]  
  
    insVal = a[i];  
    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



testů ... 1 (nejlepší případ) až k (nejhorší případ), v průměru $(k+1)/2$
přesunů ... 2 (nejlepší případ) až k+1 (nejhorší případ), v průměru $(k+3)/2$

Celkem testů

nejlepší případ: $n - 1 \in \Theta(n)$

nejhorší případ: $(n^2 - n)/2 \in \Theta(n^2)$

Celkem přesunů

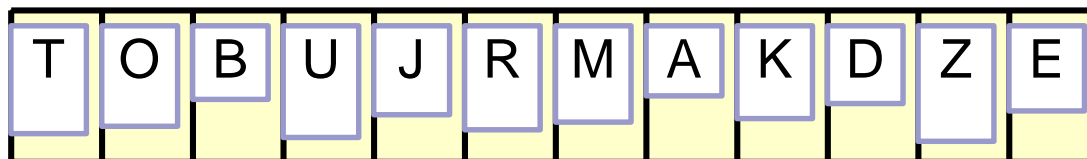
$2n - 2 \in \Theta(n)$

$(n^2 + n - 2)/2 \in \Theta(n^2)$

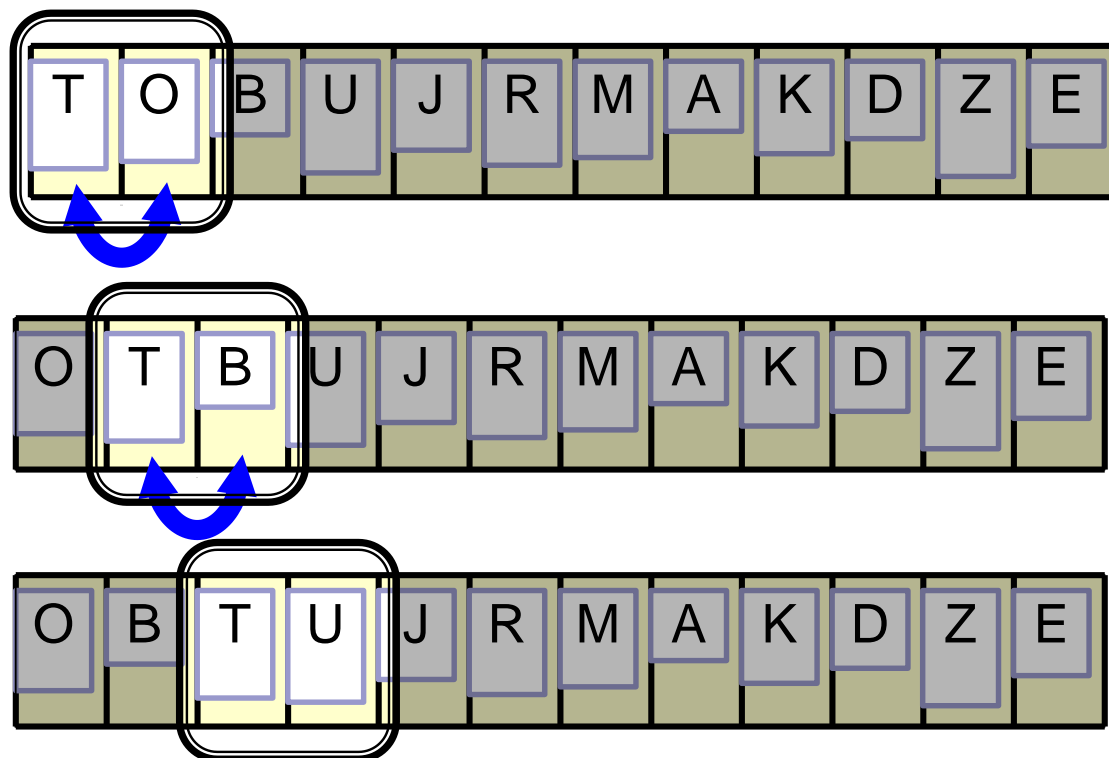
Asymptotická složitost Insertion Sortu je $O(n^2)$.

Bubble sort (řazení záměnou)

Start

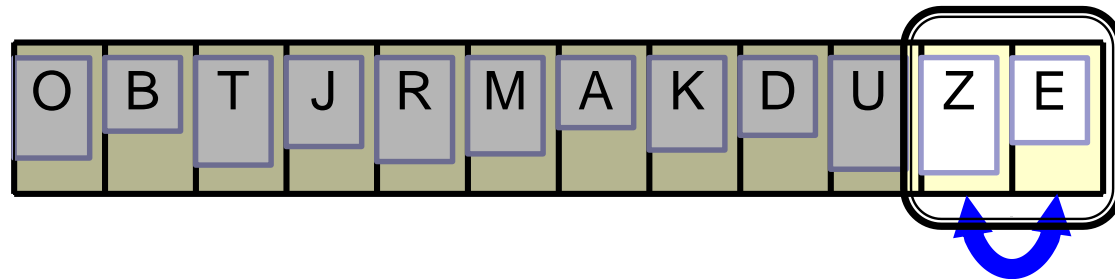
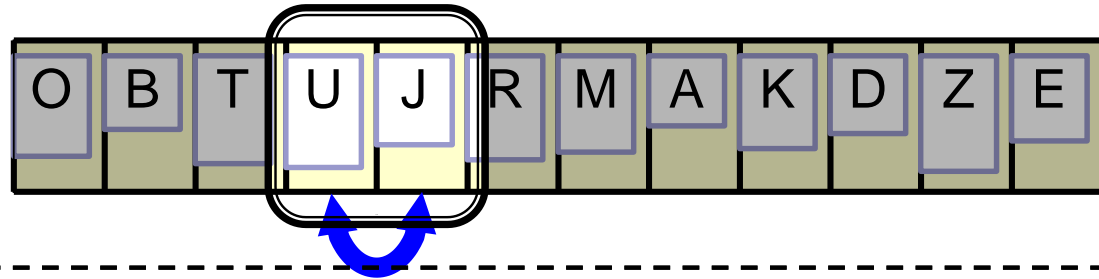


Fáze 1

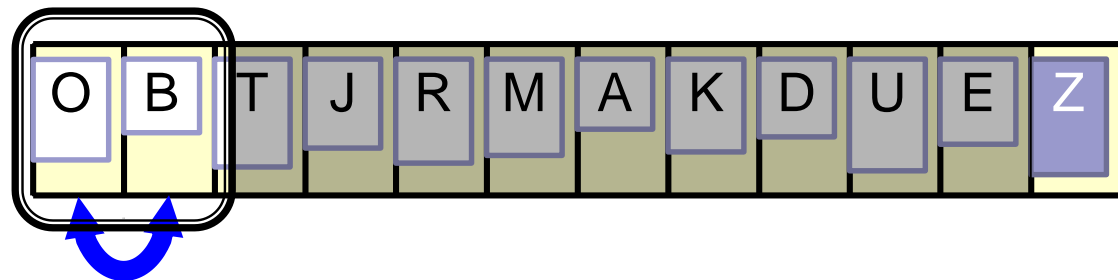


Bubble sort

Fáze 1

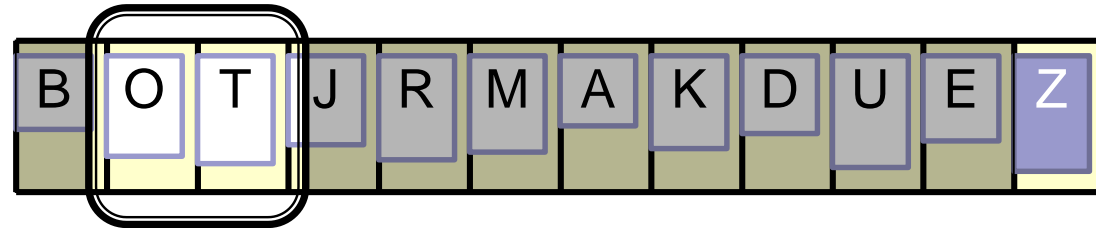


Fáze 2

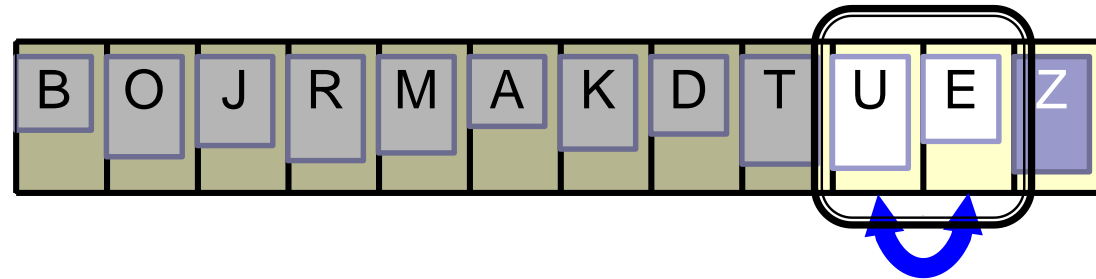


Bubble sort

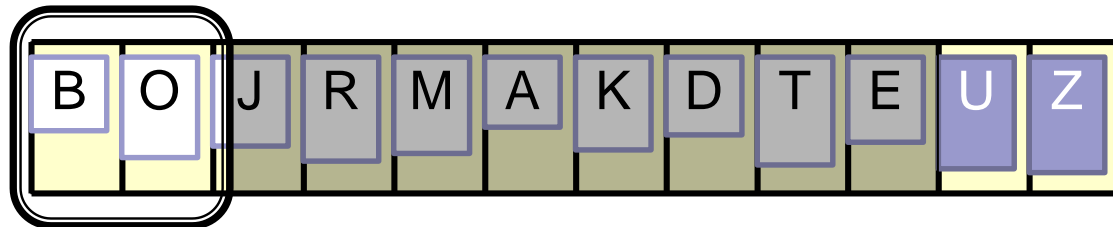
Fáze 2



...



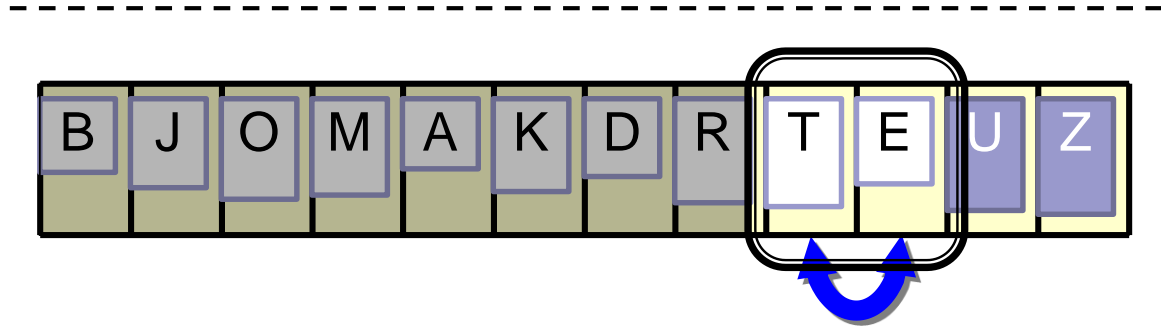
Fáze 3



Bubble sort

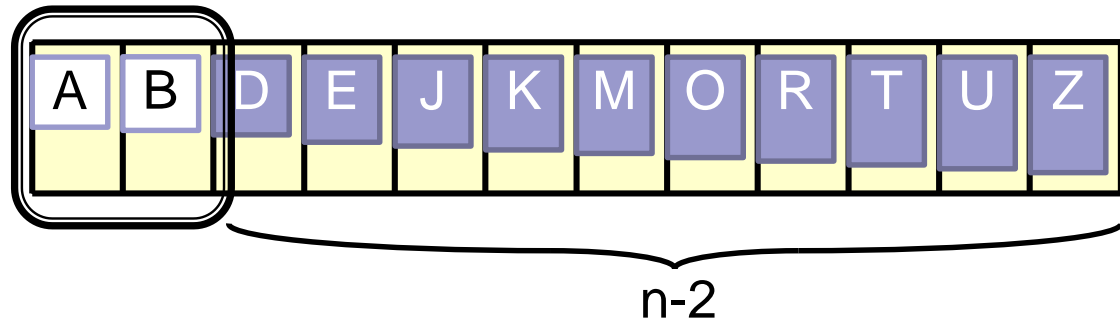
Fáze 3

...

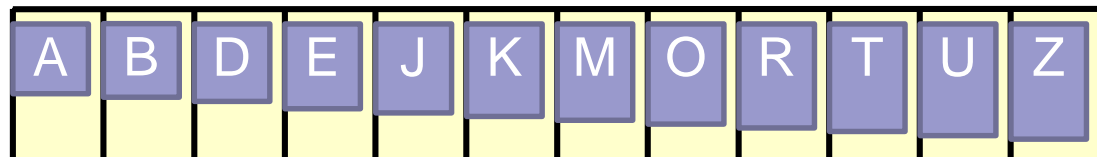


...

Fáze n-1



Seřazeno



Bubble sort

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

Celkem
testů

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

Celkem
přesunů

$$0 \in \Theta(1)$$

nejlepší případ

$$\frac{1}{4}(n^2 - n) \in \Theta(n^2)$$

průměrný případ

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

nejhorší případ

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

Kontrolní otázka

Na low-level hardware chceme vzestupně seřadit 25 náhodně vygenerovaných čísel uložených v paměti. Všechny základní operace v registrech (porovnání, inkrementace, čtení čísla z paměti) se vykonají za 1 μ s, s výjimkou zápisu čísla do paměti, které se vykoná za 500 μ s.

Který z níže uvedených algoritmů je pro tuto úlohu nejvýhodnější?

- A. Selection sort
- B. Insertion sort
- C. Bubble sort
- D. vyjde to zhruba nastejno



**Na low-level hardware
chceme vzestupně seřadit
25 náhodně**

**vygenerovaných čísel
uložených v paměti.**

**Všechny základní operace
v registrech se vykonají za**

1 μ s. s výjimkou zápisu

Porovnání algoritmů

Selection sort

- Lineární počet přesunů (zázpisů do paměti).

Insertion sort

- Velmi rychlý pro téměř uspořádané vstupy.

Bubble sort

- Nejkratší kód.

Obecně se tyto algoritmy snadno implementují a lze je použít pro „malá“ data.

Jejich časová složitost je však v průměrném případě kvadratická.

Insertion sort vs. Bubble sort

<https://youtu.be/TZRWRjq2CAg>



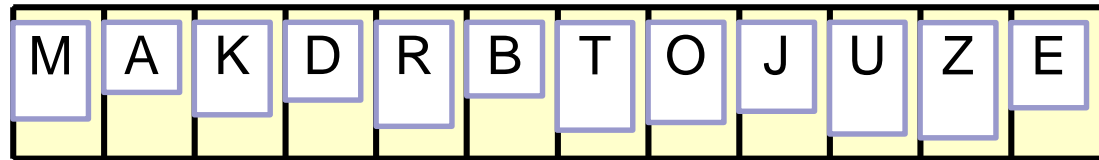
Audience Q&A Session

① Start presenting to display the audience questions on this slide.

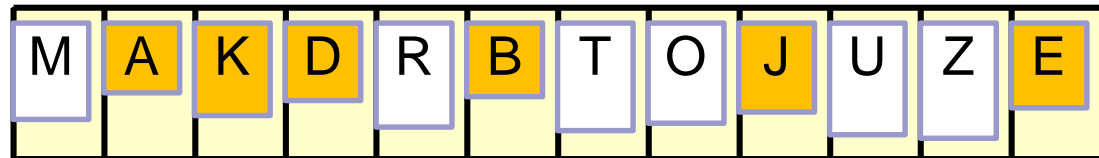
Quicksort

- Hoare (1962), technika rozděl a panuj

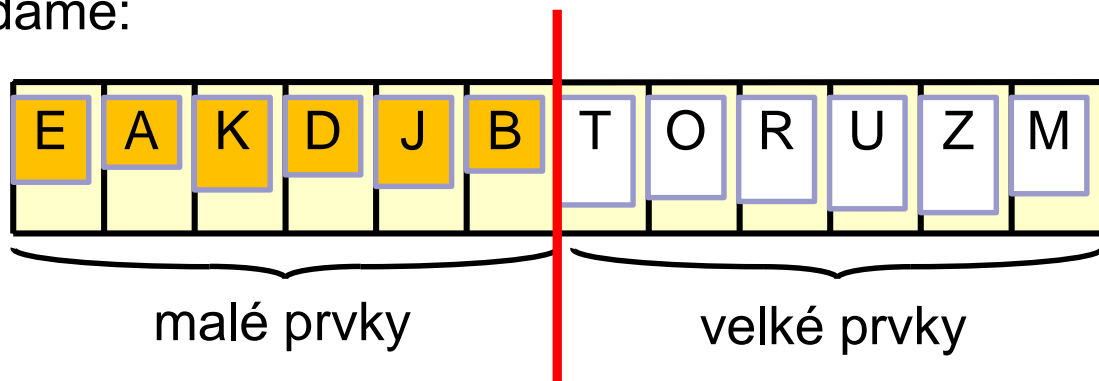
Start



Vstupní pole rozdělíme na „malé“ a „velké“ prvky



a přeuspořádáme:

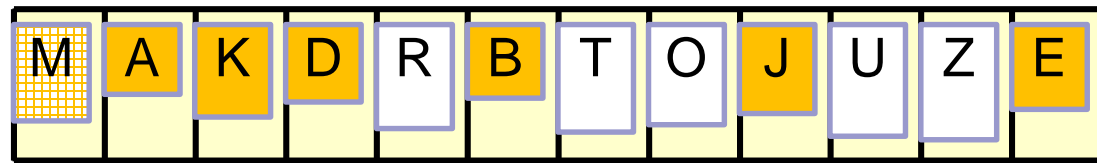


Úlohu tak rozdělíme na 2 části, které řešíme rekurzivně.

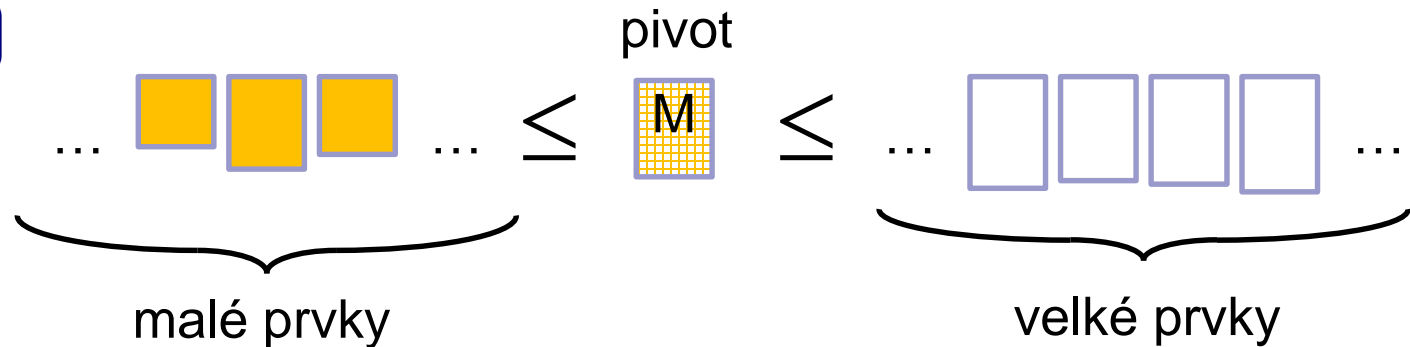
Quicksort

- Podle čeho určíme malé a velké hodnoty?
 - ⇒ Podle libovolně zvoleného prvku (tzv. pivot).

Nechť je pivot například první prvek v poli.



Cíl dělení

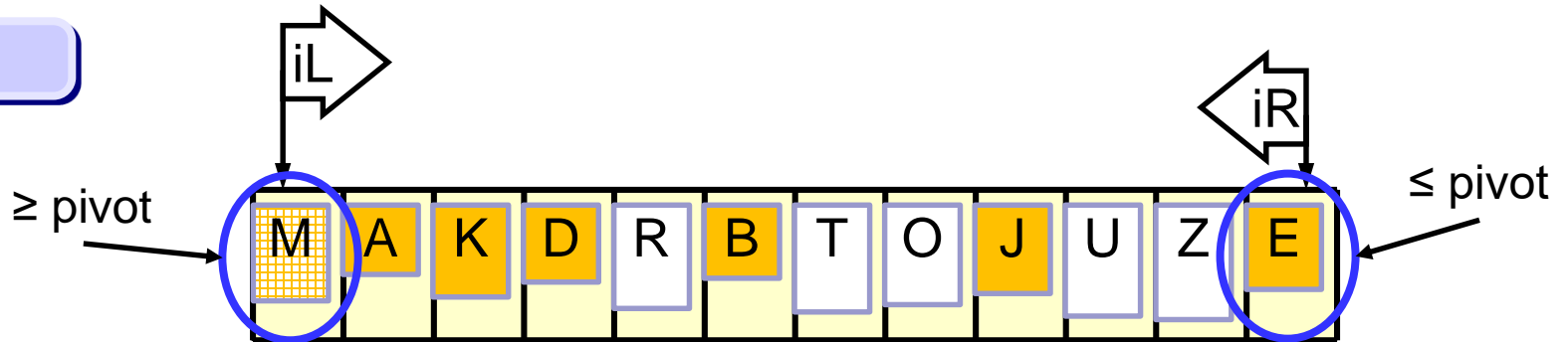


- Další možné strategie pro volbu pivotu:
 - Náhodný prvek
 - Medián ze tří prvků

Quicksort

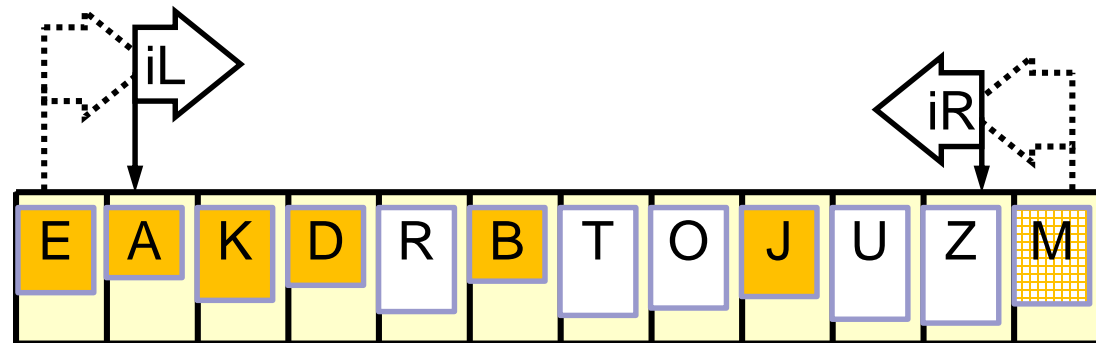
- Jak prvky efektivně přeuspořádáme podle pivotu?

Init



Krok 1

prohodíme

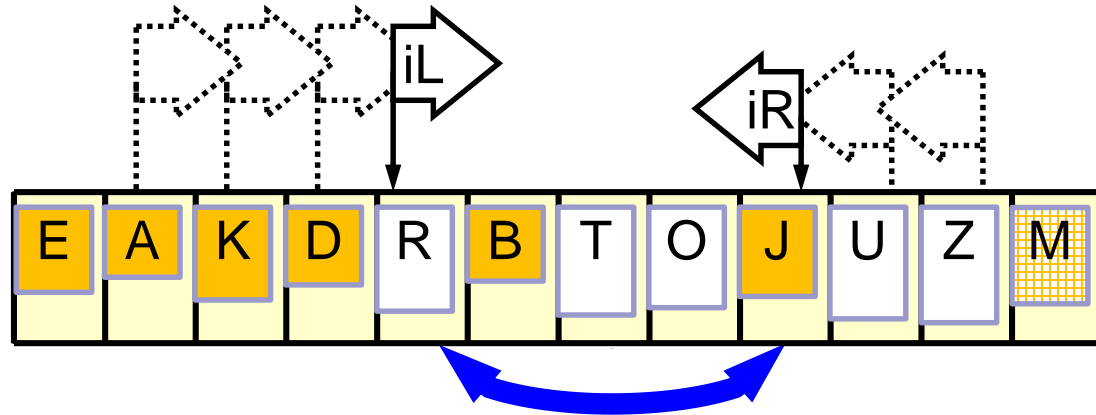


Po prohození prvků ukazatele posuneme směrem k sobě.

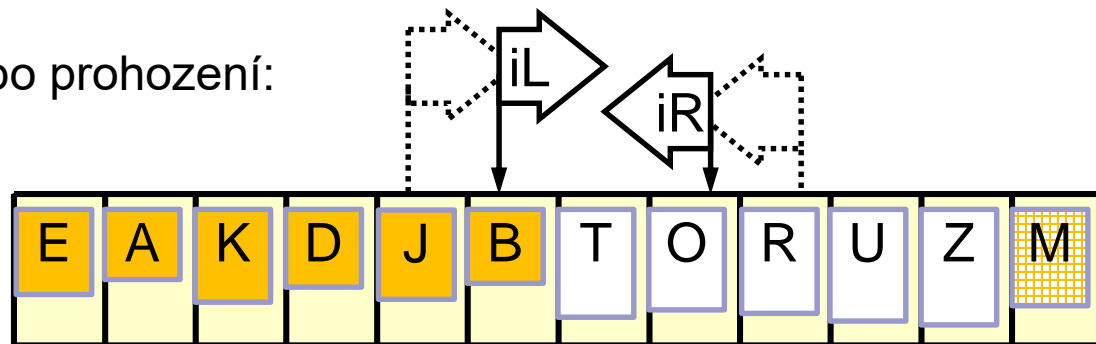
Quicksort

Krok 2

Hledáme další dvojici prvků k prohození.

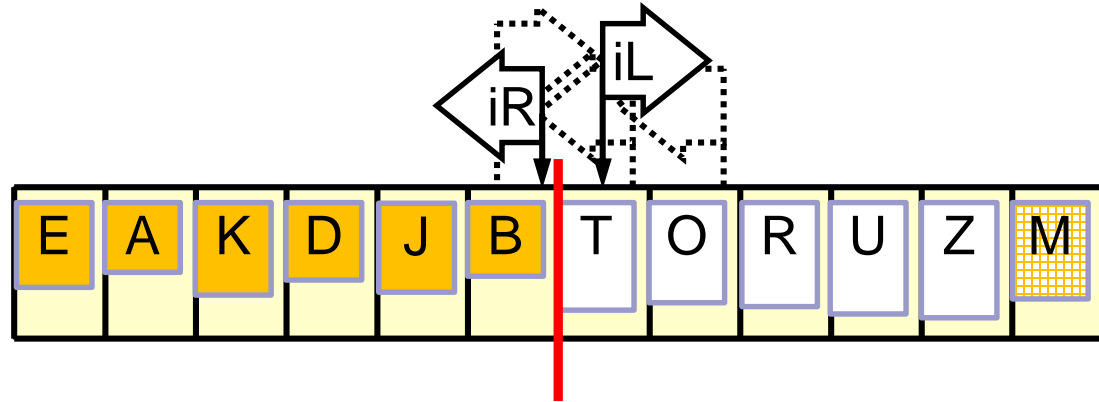


Posun ukazatelů po prohození:



Quicksort

Krok 3

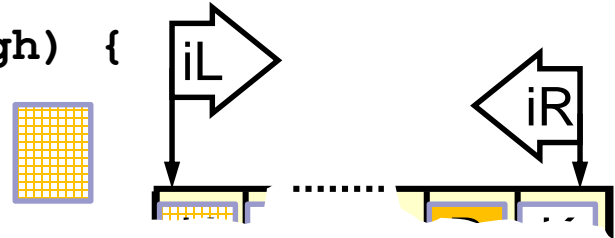


$\leftarrow iR < iL \rightarrow \Rightarrow \text{Stop}$

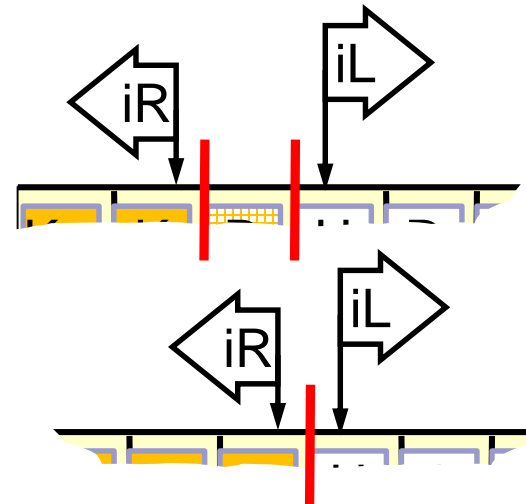
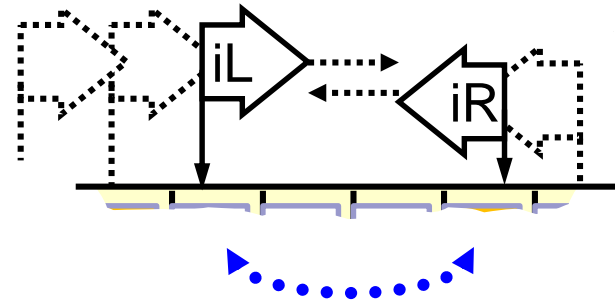
Přeuspořádáno v lineárním čase.

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);  
}
```



Quicksort

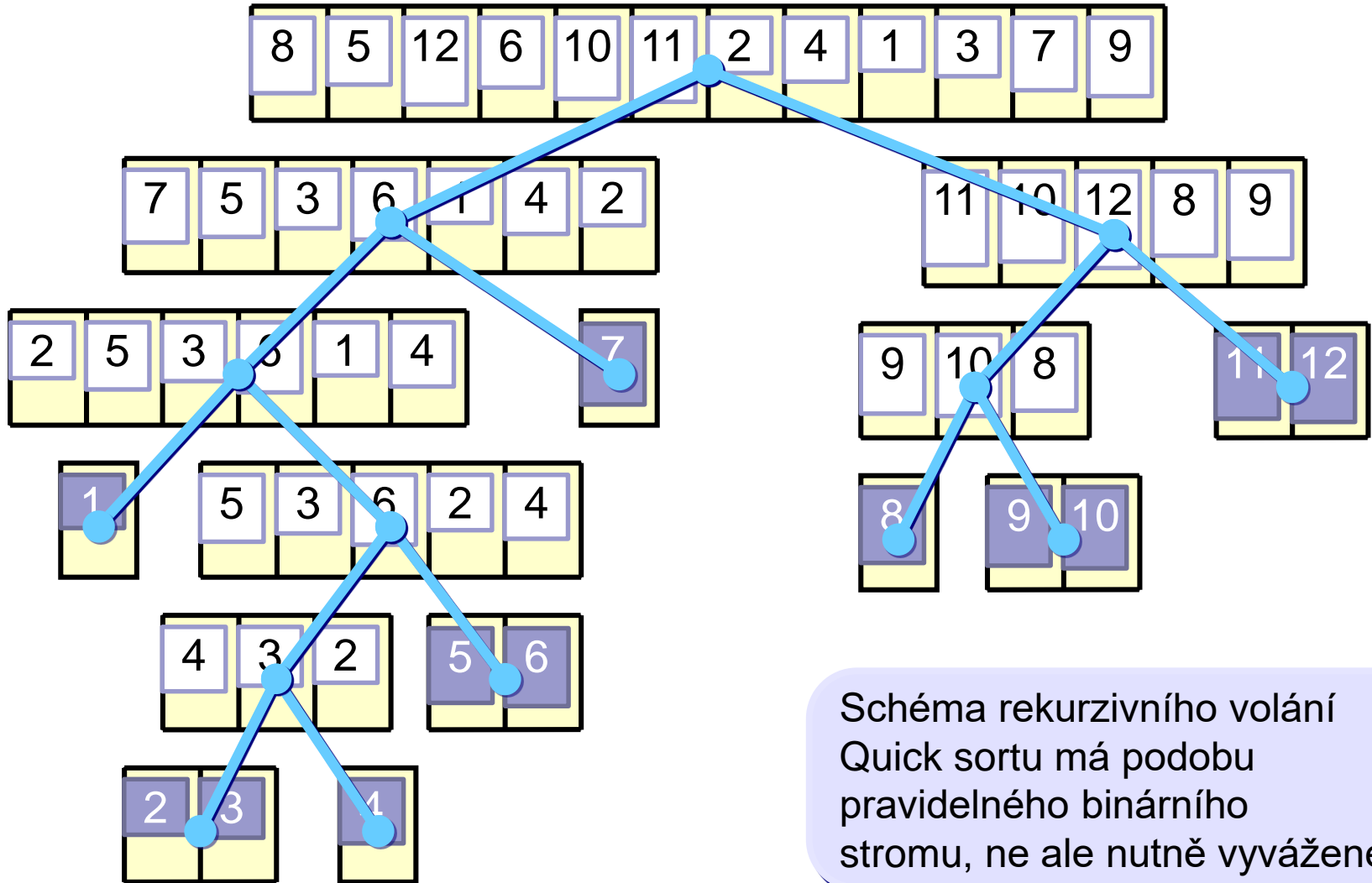


Schéma rekurzivního volání Quick sortu má podobu pravidelného binárního stromu, ne ale nutně vyváženého.

Quicksort

- Jaká je asymptotická složitost?

Celkem přesunů
a testů

$\Theta(n \log n)$

nejlepší případ

$\Theta(n \log n)$

průměrný případ

$\Theta(n^2)$

nejhorší případ

Např. pokud je
vstupní posloupnost
již uspořádaná.

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

... ale! :

Očekávaná složitost Quicksortu je $\Theta(n \log n)$.

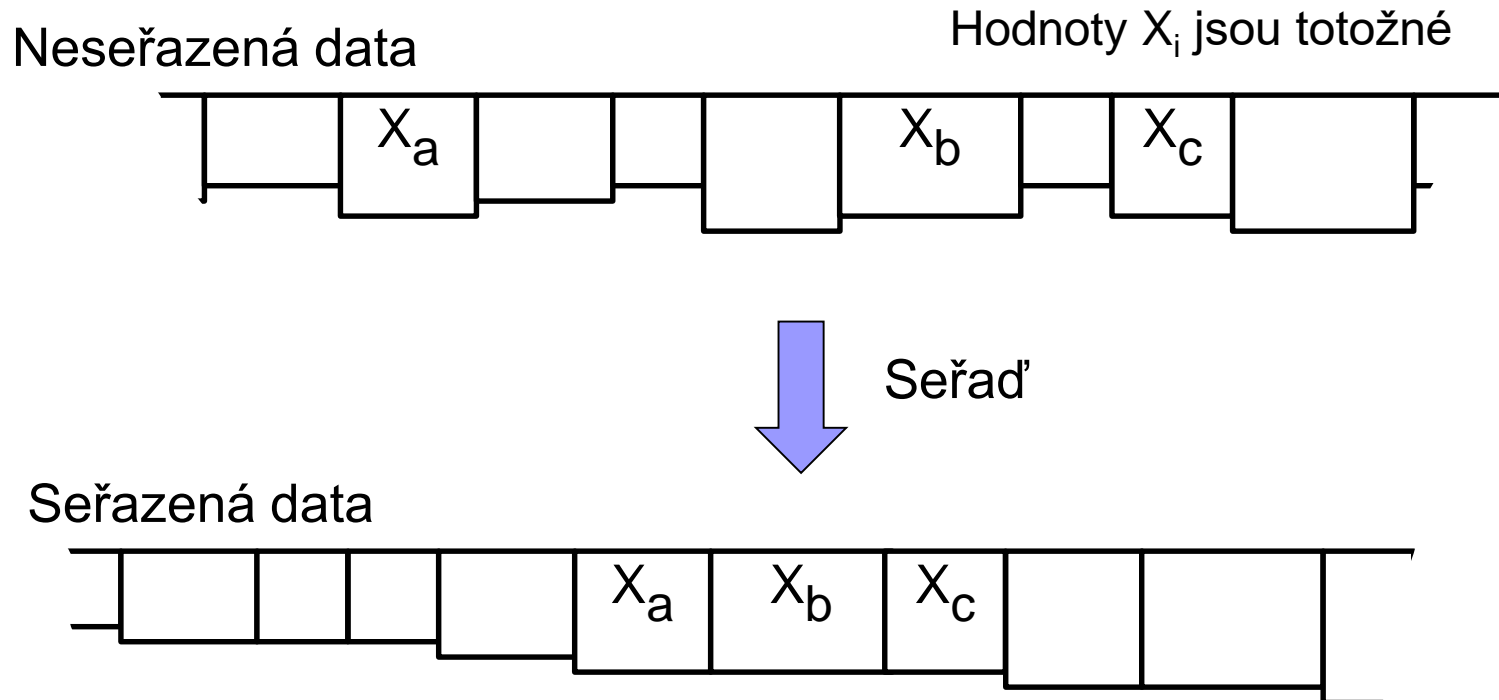


Audience Q&A Session

① Start presenting to display the audience questions on this slide.

Stabilita řazení

- Řazení je **stabilní**, pokud nemění vzájemné pořadí prvků se stejnou hodnotou.



Stabilita řazení – příklad využití

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

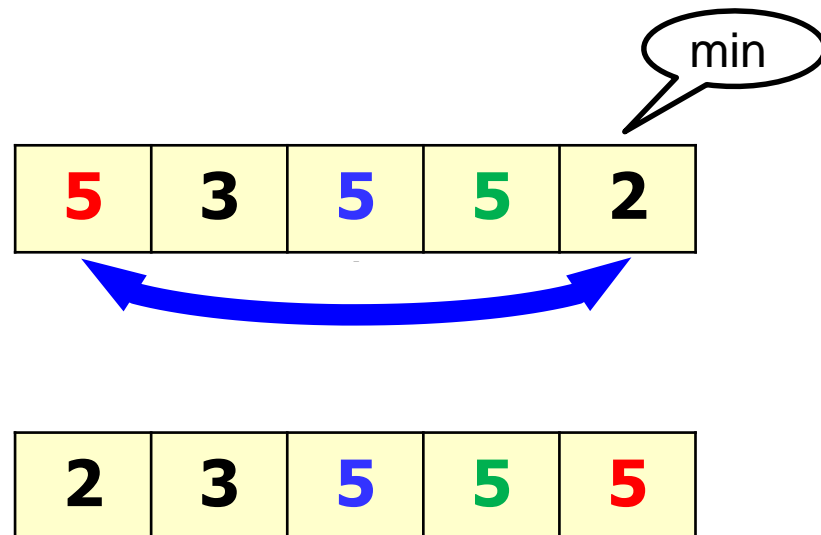
Stabilita řazení

- Insertion sort ✓
- Bubble sort ✓
- Selection sort ✗
- Quicksort ✗

Prohazují se jen sousední prvky, stabilitu lze zajistit.

Při prohazování vzdálených prvků se nekontroluje, co je mezi nimi.

Příklad (Selection sort):



Šestá domácí úloha

Líný AVL strom

Delete(3)

```
  2+
1+  3+
    4+
```

Insert(5)

```
  2+
1+  3-
    4+
```

Insert(3)

```
  2+
1+  4+
    3- 5+
```

Delete(5), Delete(2)

```
  2+
1+  4+
    3+ 5+
```

konsolidace

```
  2-
1+  4+
    3+ 5-
```

Existuje cesta z kořene do listu s $1 + \lfloor \text{depth}(T)/2 \rfloor$ uzly označenými jako smazané
=> označené uzly odebereme v pořadí postorder.

Remove(5)

```
  2-
1+  4+
    3+ 5-
```

Remove(2)

```
  2-          3+
1+  4+      1+  4+
    3+          3+
```



Audience Q&A Session

① Start presenting to display the audience questions on this slide.