1. Java - řazení

# Řazení datových struktur

* Řazení datových struktur znamená uspořádání prvků v datové struktuře v určitém pořadí
* To se obvykle provádí s cílem usnadnit vyhledávání, vkládání nebo mazání prvků v datové struktuře
* Existuje mnoho různých metod řazení datových struktur, záleží na tom, jaká datová struktura je použita

# Souvislost s porovnáváním

* Porovnávání se provádí pomocí rozhraní **Comparator** nebo **Comparable**
* Comparator
  + umožňuje vytvořit samostatnou třídu pro porovnávání prvků
* Comparable
  + je implementováno přímo v třídě, která se řadí
* V obou případech se používá metoda **compare()** k porovnání dvou prvků
  + Vrací hodnotu **menší než 0**, pokud první prvek je menší než druhý
  + Vrací hodnotu **větší než 0**, pokud první prvek je větší než druhý
  + Vrací hodnotu **0**, pokud jsou prvky stejné
* Obsah obrázku text

  Popis byl vytvořen automatickyPři použití algoritmů řazení, jako je například **Arrays.sort()** nebo **Collections.sort()**, se použije daný **Comparator** nebo **Comparable** k porovnání prvků a jejich řazení podle daného kritéria

# Operace při řazení nad datovými strukturami

* Porovnání prvků
  + Tato operace je nezbytná pro určení, který prvek je větší nebo menší než druhý, což je nezbytné pro uspořádání prvků v pořadí
* Výměna prvků
  + Pokud jsou dva prvky ve špatném pořadí, musí se vyměnit, aby se správně seřadily
* Vkládání prvku
  + Při použití metod řazení jako Insertion Sort se vkládají prvky na správná místa v již seřazené posloupnosti
* Mazání prvku
  + Pokud je prvek duplicitní a chceme ho odstranit, musí se z datové struktury odstranit
* Rozdělení datové struktury
  + V metodách řazení jako Merge Sort nebo Quick Sort se datová struktura rozdělí na menší části, aby se prvkům mohlo lépe řídit a řadit
* Sloučení datových struktur
  + V metodách řazení jako Merge Sort se seřazené menší části spojí do jedné velké seřazené datové struktury

# Významné řadící algoritmy

* Bubble sort
  + Algoritmus opakovaně prochází seznam, přičemž porovnává každé dva sousedící prvky, a pokud nejsou ve správném pořadí, prohodí je
  + Pro praktické účely je neefektivní, využívá se hlavně pro výukové účely či v nenáročných aplikacích
* Selection sort
  + Myšlenka spočívá v nalezení minima, které se přesune na začátek pole (nebo můžeme hledat i maximum, a to dávat na konec)
  + V prvním kroku tedy nalezneme nejmenší prvek v poli a ten poté přesuneme na začátek, v druhém kroku již nebudeme při hledání minima brát v potaz dříve nalezené minimum, po dostatečném počtu kroků dostaneme pole seřazené, algoritmus má nepříliš výhodnou časovou složitost a není stabilní, je však velice jednoduchý na pochopení i implementaci
* Insertion sort
  + Řazení vkládáním, je jednoduchý řadicí algoritmus založený na porovnávání
  + Algoritmus pracuje tak, že prochází prvky postupně a každý další nesetříděný prvek zařadí na správné místo do již setříděné posloupnosti
* Merge sort
  + Algoritmus, založený na tzv. principu rozděl a panuj, to znamená, že pokud nějaký problém neumíme vyřešit v celku, rozložíme si ho na více menších a jednodušších problémů, ten samý postup aplikujeme i na tyto problémy
* Quick sort
  + Jeden z nejrychlejších běžných algoritmů řazení založených na porovnávání prvků, paměťově nenáročný, funguje dobře na malých i velkých polích

# Prakticky

* Poznání řadícího algoritmu dle kódu, vysvětlení algoritmu, prohození prvků v poli

Obsah obrázku text

Popis byl vytvořen automatickyObsah obrázku text

Popis byl vytvořen automaticky

Obsah obrázku text

Popis byl vytvořen automaticky