Quicksort Arbeitsblatt

Was ist Quicksort?

Quicksort ist ein schnelles, rekursives Sortieralgorithmus, das auf dem Prinzip der Teilen und Herrschen basiert. Es wurde von Tony Hoare entwickelt und ist einer der am häufigsten verwendeten Sortieralgorithmen.

Quicksort funktioniert indem es ein Element aus der Liste auswählt, das als Pivot-Element bezeichnet wird. Es teilt dann die Liste in zwei Teile auf: einen Teil mit Elementen, die kleiner als das Pivotelement sind, und einen Teil mit Elementen, die größer als das Pivotelement sind. Dieser Prozess wird für jeden Teil der Liste wiederholt, bis jedes Element in der richtigen Reihenfolge ist.

Wie funktioniert Quicksort?

1. Wähle das erste Element in der Liste als Pivot-Element.
2. Teile die Liste in zwei Teile auf: einen Teil mit Elementen, die kleiner als das Pivotelement sind, und einen Teil mit Elementen, die größer als das Pivotelement sind.
3. Sortiere die beiden Teile der Liste rekursiv mit Quicksort.
4. Füge die beiden sortierten Teile der Liste zusammen und gib das Ergebnis zurück.

Beispiel

Angenommen, wir haben die folgende unsortierte Liste: [8, 5, 2, 9, 7, 6, 3]

1. Wähle das erste Element der Liste, 8, als Pivot-Element.
2. Teile die Liste in zwei Teile auf: [5, 2, 3] und [9, 7, 6].
3. Sortiere beide Teile rekursiv mit Quicksort: [2, 3, 5] und [6, 7, 9].
4. Füge die beiden sortierten Teile zusammen: [2, 3, 5, 6, 7, 8, 9].

Das Ergebnis ist eine sortierte Liste.

3	5	4	1	2
3	2	1	4	5
1	2	3	4	5

Vor- und Nachteile von Quicksort

Vorteile

- Schnelle Sortiergeschwindigkeit: Im Durchschnitt ist Quicksort schneller als andere Sortieralgorithmen wie Bubblesort oder Insertion Sort.
- Platzeffizient: Quicksort benötigt keinen zusätzlichen Speicherplatz zum Sortieren, da es in-place sortiert.

Nachteile

- Unvorhersehbare Laufzeit: Im schlechtesten Fall kann Quicksort langsamer sein als andere Sortieralgorithmen.

Pseudocode

```
QUICKSORT(A, lo, hi)
  if lo < hi
    p = PARTITION(A, lo, hi)
    QUICKSORT(A, lo, p - 1)
    QUICKSORT(A, p + 1, hi)

PARTITION(A, lo, hi)
  pivot = A[hi]
  i = lo
  for j = lo to hi - 1
    if A[j] < pivot
      swap A[i] and A[j]
      i = i + 1
  swap A[i] and A[hi]
  return i
```

Dieser Algorithmus nimmt ein Array A und sortiert es von lo bis hi. Der PARTITION-Schritt teilt das Array in zwei Teile auf, indem es ein Pivotelement auswählt und alle Elemente, die kleiner sind als das Pivotelement, zu einem Teil des Arrays verschiebt und alle anderen Elemente zum anderen Teil. Der QUICKSORT-Schritt sortiert dann rekursiv jeden dieser beiden Teile.

Hinweis: Dies ist nur eine mögliche Implementierung von Quicksort. Es gibt andere Variationen, die leicht von dieser abweichen können.