

LU01.A01 - Imperativer Bubblesort



Implementieren Sie den Bubble-Sort-Algorithmus, um eine gegebene Liste von Ganzzahlen zu sortieren.

Erklärung von Bubble Sort:

Bubble Sort ist ein Vergleichssortieralgorithmus, der durch wiederholtes Vertauschen benachbarter Elemente, die in der falschen Reihenfolge sind, funktioniert. Die Liste wird mehrmals durchlaufen, wobei jedes benachbarte Paar verglichen und bei Bedarf vertauscht wird, bis keine Vertauschungen mehr erforderlich sind.

Anforderungen:

Schreiben Sie eine Funktion namens `bubble_sort`, die eine Liste von Ganzzahlen als Parameter annimmt. Implementieren Sie den Bubble-Sort-Algorithmus, um die Liste in aufsteigender Reihenfolge zu sortieren. Geben Sie die sortierte Liste aus.

Beispielinput

```
zahlen = [64, 34, 25, 12, 22, 11, 90]
```

Beispieloutput

```
Sortierte Liste: [11, 12, 22, 25, 34, 64, 90]
```

Ausführliche Erklärung des Bubble Sort

Der Bubble-Sort-Algorithmus ist ein einfacher Sortieralgorithmus, der mehrmals über die Liste läuft, benachbarte Elemente vergleicht und sie vertauscht, wenn sie in der falschen Reihenfolge sind. Die Erklärung ist wie folgt:

Funktionsweise:

1. **Durchlauf der Liste:** Der Algorithmus beginnt am Anfang der Liste und vergleicht die ersten beiden benachbarten Elemente miteinander (Element i und Element $i+1$).
2. **Vergleich und Tausch:** Wenn das Element i größer ist als das Element $i+1$, werden die Elemente vertauscht, sodass das größere Element an

einer höheren Position in der Liste steht.

3. **Fortsetzung des Durchlaufs:** Der Algorithmus fährt fort, indem er zum nächsten Paar benachbarter Elemente übergeht und den gleichen Vergleichs- und Tauschprozess wiederholt.
4. **Wiederholung des Durchlaufs:** Der gesamte Durchlauf wird wiederholt, bis kein Tausch mehr erforderlich ist. Bei jedem Durchlauf „schwimmt“ das größte Element der verbleibenden unsortierten Elemente nach oben (daher der Name „Bubble Sort“), da es durch aufeinanderfolgende Tauschaktionen an die richtige Position verschoben wird.
5. **Abschluss:** Der Algorithmus ist abgeschlossen, wenn ein Durchlauf ohne Tauschaktionen erfolgt, was bedeutet, dass die Liste sortiert ist.

Beispiel:

Angenommen, wir haben die Liste [5, 2, 9, 1, 5, 6].

Erster Durchlauf: Vergleiche und vertausche die benachbarten Elemente:

- (5, 2): Vertauscht zu (2, 5)
- (5, 9): Kein Tausch
- (9, 1): Vertauscht zu (1, 9)
- (9, 5): Vertauscht zu (5, 9)
- (9, 6): Vertauscht zu (6, 9)

Ergebnis nach dem ersten Durchlauf: [2, 5, 1, 5, 6, 9]

Zweiter Durchlauf: Wiederhole den Vorgang:

- (2, 5): Kein Tausch
- (5, 1): Vertauscht zu (1, 5)
- (5, 5): Kein Tausch
- (5, 6): Kein Tausch

Ergebnis nach dem zweiten Durchlauf: [2, 1, 5, 5, 6, 9]

Weitere Durchläufe: Fortsetzen, bis die Liste sortiert ist: Ergebnis: [1, 2, 5, 5, 6, 9]

Eigenschaften:



- **Stabil:** Gleiche Elemente behalten ihre relative Reihenfolge bei.
- **In-Place:** Benötigt keinen zusätzlichen Speicherplatz.
- **Adaptiv:** Die Effizienz verbessert sich, wenn die Liste teilweise sortiert ist.

Trotz seiner Einfachheit ist Bubble Sort in der Praxis oft ineffizient im Vergleich zu anderen Sortieralgorithmen wie Quick Sort oder Merge Sort, besonders wenn es um große Datenmengen geht.



© Kevin Maurizi

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

<https://wiki.bzz.ch/modul/m323/learningunits/lu01/aufgaben/imperativerbubblesort>

Last update: **2024/03/28 14:07**

