# L2\_2.2 Sortierung: Selection Sort

Der Selection Sort ist ein vergleichsbasierter Suchalgorithmus und hat eine intuitive Vorgehensweise:

1. Suche das kleineste Element und tausche das Element mit dem ersten Element.
2. Suche das zweitkleinste Element und tausche es mit dem zweiten Element.
3. usw.

**Beispiel:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | 5 | 4 | 9 | 1 | 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Bis Index 0 sortiert |  | 1 | 4 | 9 | 5 | 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Bis Index 1 sortiert |  | 1 | 3 | 9 | 5 | 4 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Bis Index 2 sortiert |  | 1 | 3 | 4 | 5 | 9 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Bis Index 3 sortiert |  | 1 | 3 | 4 | 5 | 9 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Bis Index 4 sortiert |  | 1 | 3 | 4 | 5 | 9 |

**Array sortiert!**

**🡺 letztes Arrayelement muss nicht mehr geprüft werden**

**Umsetzung**

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

**Erklärung zum Ablauf des Algorithmus:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| zahlen[0] | zahlen[1] | zahlen[2] | zahlen[3] | zahlen[4] |  | akt\_index | min\_index |
| 5 | 4 | 9 | 1 | 3 |  | 0 | 3 |

zahlen[akt\_index] != zahlen[min\_index] 🡪 Tausch   
 🡪 Sortierung bis zahlen[0] erfolgt 🡪 akt\_index = 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| zahlen[0] | zahlen[1] | zahlen[2] | zahlen[3] | zahlen[4] |  | akt\_index | min\_index |
| 1 | 4 | 9 | 5 | 3 |  | 1 | 4 |

zahlen[akt\_index] != zahlen[min\_index] 🡪 Tausch   
 🡪 Sortierung bis zahlen[1] erfolgt 🡪 akt\_index = 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| zahlen[0] | zahlen[1] | zahlen[2] | zahlen[3] | zahlen[4] |  | akt\_index | min\_index |
| 1 | 3 | 9 | 5 | 4 |  | 2 | 4 |

zahlen[akt\_index] != zahlen[min\_index] 🡪 Tausch   
 🡪 Sortierung bis zahlen[2] erfolgt 🡪 akt\_index = 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| zahlen[0] | zahlen[1] | zahlen[2] | zahlen[3] | zahlen[4] |  | akt\_index | min\_index |
| 1 | 3 | 4 | 5 | 9 |  | 3 | 3 |

zahlen[akt\_index] != zahlen[min\_index] 🡪 Tausch   
 🡪 Sortierung bis zahlen[3] erfolgt 🡪 akt\_index = 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| zahlen[0] | zahlen[1] | zahlen[2] | zahlen[3] | zahlen[4] |  | akt\_index | min\_index |
| 1 | 3 | 4 | 5 | 9 |  | 4 | 4 |

zahlen[akt\_index] != zahlen[min\_index] 🡪 Tausch   
 🡪 Sortierung bis zahlen[4] erfolgt 🡪 akt\_index = 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| zahlen[0] | zahlen[1] | zahlen[2] | zahlen[3] | zahlen[4] | **Array sortiert!** |
| **1** | **3** | **4** | **5** | **9** |

## **Ausgabe eines Arrays**

Für die Ausgabe aller Elemente eines Arrays kann die for-Schleife eingesetzt werden. Sie basiert auf einer Variablen für den Index und wird mit der Increment-Anweisung automatisch um die Schrittweite 1 erhöht, bis der letzte Index des Arrays erreicht ist.

laenge = len(*meinArray*)

for i in range(laenge):

print(meinArray[i])

Mit der print-Funktion kann aber auch der ganze Array ausgegeben werden.

print(*meinArray*)