Aufgabe 6

Gegeben sei ein einfacher Sortieralgorithmus, der ein gegebenes Feld A dadurch sortiert, dass er das *Minimum* m von A *findet*, dann das Minimum von A ohne das Element m usw.

(a) Geben Sie den Algorithmus in Java an. Implementieren Sie den Algorithmus *in situ*, d. h. so, dass er außer dem Eingabefeld nur konstanten Extraspeicher benötigt. Es steht eine Testklasse zur Verfügung.

```java
public class SortierungDurchAuswaehlen {
  static void vertausche(int[] zahlen, int index1, int index2) {
    int tmp = zahlen[index1];
    zahlen[index1] = zahlen[index2];
    zahlen[index2] = tmp;
  }

  static void sortiereDurchAuswählen(int[] zahlen) {
    // Am Anfang ist die Markierung das erste Element im Zahlen-Array.
    int markierung = 0;
    while (markierung < zahlen.length) {
      // Bestimme das kleinste Element.
      // 'min' ist der Index des kleinsten Elements.
      // Am Anfang auf das letzte Element setzen.
      int min = zahlen.length - 1;
      // Wir müssen nicht bis letzten Index gehen, da wir 'min' auf
      ↪ das letzte Element
      // setzen.
      for (int i = markierung; i < zahlen.length - 1; i++) {
        if (zahlen[i] < zahlen[min]) {
          min = i;
        }
      }

      // Tausche zahlen[markierung] mit gefundenem Element.
      vertausche(zahlen, markierung, min);
      // Die Markierung um eins nach hinten verlegen.
      markierung++;
    }
  }

  public static void main(String[] args) {
    int[] zahlen = { 5, 2, 7, 1, 6, 3, 4 };
    sortiereDurchAuswählen(zahlen);
    for (int i = 0; i < zahlen.length; i++) {
      System.out.print(zahlen[i] + " ");
    }
  }
}
```

(b) Analysieren Sie die Laufzeit Ihres Algorithmus.

Beim ersten Durchlauf des *Selectionsort*-Algorithmus muss $n-1$ mal das Minimum durch Vergleich ermittel werden, beim zweiten Mal $n-2$. Mit Hilfe der *Gaußschen Summenformel* kann die Komplexität gerechnet werden:

$$(n-1)+(n-2)+\cdots+3+2+1 = \frac{(n-1)\cdot n}{2} = \frac{n^2}{2} - \frac{n}{2} \approx \frac{n^2}{2} \approx n^2$$

Da es bei der Berechnung des Komplexität um die Berechnung der asymptotischen oberen Grenze geht, können Konstanten und die Addition, Subtraktion, Multiplikation und Division mit Konstanten z. b. $\frac{n^2}{2}$ vernachlässigt werden.

Der *Selectionsort*-Algorithmus hat deshalb die Komplexität $\mathcal{O}(n^2)$, er ist von der Ordnung $\mathcal{O}(n^2)$.

(c) Geben Sie eine Datenstruktur an, mit der Sie Ihren Algorithmus beschleunigen können.

Der *Selectionsort*-Algorithmus kann mit einer Min- (in diesem Fall) bzw. einer Max-Heap beschleunigt werden. Mit Hilfe dieser Datenstruktur kann sehr schnell das Minimum gefunden werden. So kann auf die viele Vergleiche verzichtet werden. Die Komplexität ist dann $\mathcal{O}(n \log n)$.