**Team**: 9, Tim Hagemann, Tim Hartig

**Aufgabenaufteilung**:

1. Insertion-Sort und Messungen
2. Selection-Sort und Zahlengenerator

**Quellenangaben**: Keine.

**Begründung für Codeübernahme**: Es wurde kein Code Übernommen.

**Bearbeitungszeitraum**:  
12.11.14, 09:30 – 11:15  
16.11.14, 15:15 – 16:30

**Aktueller Stand**: Skizze ist fertiggestellt und Verständnis ist vorhanden. Außerdem wurden Teile zum Testen des Verständnissen und der Realisierbarkeit implementiert.

Zahlengenerator

## sortNum: length x mode 🡪 ∅

**length** ist eine natürliche Zahl.  
**mode** ε {random, ascending, descending}

**sortNum** generiert Zahlenfolgen der Länge **length** und speichert sie in der Datei „zahlen.dat“.  
**mode** gibt an, welche Art von Zahlenfolge generiert werden soll.

Random  
Es wird eine zufällige Zahlenfolge generiert.

Ascending  
Es wird eine aufsteigend sortierte Zahlenfolge generiert. (Bspw. 1, 3, 5,…)  
Diese stellt den sog. „Best-Case“ als Eingabe für eine Sortierung dar.

Descending  
Es wird eine absteigend sortierte Zahlenfolge generiert (Bspw. 6, 4, 2,…)  
Diese stellt den sog. „Worst-Case“ als Eingabe für eine Sortierung dar.

Sortieren I

Vorab: Bei einem Sortierdurchgang werden Leistungsdaten bzgl. des Algorithmus‘ ermittelt und in einem Tupel **messdaten** zusammengefasst. Enthaltene Daten sind beschrieben im Abschnitt „Messung“.

## insertionSort: array x pos x pos 🡪 array x messdaten

(array , von, bis)

Sortiert ein Array von der ersten gegebenen Position, bis zur zweiten gegebenen Position mit dem „Insertion Sort“-Algorithmus und gibt das sortierte (Teil-)Array zurück. Die während der Sortierung ermittelten Messdaten werden zusammen mit dem sortierten Array zurückgegeben.  
Der Algorithmus lautet wie folgt:

*Von Start bis Ende, tue  
 Nehme das Element an aktueller Position  
 Gehe von aktueller Position bis zum Start  
 Ist Element nicht größer als Element auf aktueller Position ?  
 Tausche das nebenstehende Element mit dem aktuellem Element  
 Verlasse Schleife  
 Ende gehe  
Ende tue*

## selectionSort: array x pos x pos 🡪 array x messdaten

(array , von, bis)

Sortiert ein Array von der ersten gegebenen Position, bis zur zweiten gegebenen Position mit dem „Selection Sort“-Algorithmus und gibt das sortierte (Teil-)Array zurück. Die während der Sortierung ermittelten Messdaten werden zusammen mit dem sortierten Array zurückgegeben.

Am Anfang des Algorithmus wird die gegebene Liste als unsortiert betrachtet.  
Angenommen, sie ist tatsächlich unsortiert, ergeben sich nach dem ersten Durchgang des Algorithmus zwei Bereiche innerhalb der Liste:  
Sortiert (grün) und unsortiert (weiß). Der Sortierte Bereich ordnet sich auf der linken Seite an.  
  
C:\Users\Timmay\Google Drive\HAW\Git_Projects\AD\aufgabe_2\selsort_example\1.png Ausgangsliste (unsortierter Bereich == ganze Liste)C:\Users\Timmay\Google Drive\HAW\Git_Projects\AD\aufgabe_2\selsort_example\2.png Kleinstes Element suchen 🡪 2C:\Users\Timmay\Google Drive\HAW\Git_Projects\AD\aufgabe_2\selsort_example\3.png 2 und Startelem. des unsortierten Bereichs tauschenC:\Users\Timmay\Google Drive\HAW\Git_Projects\AD\aufgabe_2\selsort_example\4.png Durchgang wiederholen (Min 🡪 9)C:\Users\Timmay\Google Drive\HAW\Git_Projects\AD\aufgabe_2\selsort_example\5.png Tausch…

Der Algorithmus für „Selection Sort“ lautet wie folgt:

*Suche kleinstes Element zwischen einschließlich Start und Ende des unsortierten Bereichs  
Tausche das ermittelte Elemente, mit dem Element am Start des unsortierten Bereichs  
Sind Start und Ende des unsortierten Bereichs an gleicher Stelle?  
 Sortierung ist fertig, beende Algorithmus!  
Setze den Start des unsortierten Bereichs rechts neben das Ende des sortierten Bereichs  
Beginne von vorne*

Messung

Bei der Messung wird jeder Algorithmus 100-mal ausgeführt. Davon sind 80 Durchgänge mit Zufallsfolgen und je zehn Durchgänge mit aufsteigend und absteigend sortierten Zahlenfolgen.  
Dabei werden folgende Daten ermittelt:

* Dauer des Sortierdurchgangs
* Anzahl von Vergleichen
* Anzahl von Tauschvorgängen

Aus allen Messungen werden jeweils die Mittelwerte gebildet. Zum Schluss werden diese Mittelwerte, zusammen mit dem Namen des getesteten Algorithmus‘ und der Größe der sortierten Zahlenfolge, in eine Datei namens „messung.log“ geschrieben.