Aufgabe 2 Skizze

Team: <03>, <Jan Dennis Bartels, Patrick Steinhauer>

Aufgabenaufteilung:

Bisher wurde alles von uns beiden erstellt

Quellenangaben: ----------

Begründung für Codeübernahme: ----------

Bearbeitungszeitraum: Für die Skizze bisher ca. 7 Stunden

Aktueller Stand: Zahlen generator ist Fertig

Skizze:

Funktion zahlengenerator:

- Die Funktion soll eine vorgegebene Anzahl an positiven ganzen zufälligen Zahlen erzeugen
- Zusätzlich soll es möglich sein "worst case" Situationen zu erzeugen also vorsortiere Zahlenreihen (auf bzw absteigend)
- Die erzeugten Zahlen sollen in einer Datei gespeichert werden.
- Die worst case Szenarien lassen sich durch eine Rekursive schleife erzeugen.
- - DIe Funktion ruft sich einfach immer wieder auf und verringert immer den max wert und schreibt diesen bzw in die Datei bzw speichert ihn zwischen.
- Verwenden werden wir jeweils eine endrukurision.

Algorithmus Selectionsort:

- Der Algorithmus selectionSort, soll rekursiv implementiert werden.
- Der Funktionsaufruf des Algorithmus soll so aussehen :
  - selectionSort(Array, Von, Bis)
- SelectionSort geht folgendermaßen vor:
  - Das erste Element wird sich gemerkt und mit allen anderen verglichen
  - Wenn ein folgendes Element kleiner ist wird sich dieses gemerkt.
  - Dies wird so lange gemacht, bis kein Element kleiner ist.
  - Dann wird das erste gemerkte Element mit dem zweiten gemerkten Element getauscht.
  - Dies wird wiederholt bis alles sortiert ist.
  - Selectionsort geht immer alle zu sortierenden Objekte durch, sucht das kleinste und geht bis zum Ende weiter.
- Pseudocode des Algorithmus:

```
selectionSort(ZuSortierendeObjekte, Von, Bis) {

        Von = 0;

        Bis = laenge(ZuSortierendeObjekte);

                Solange Von <= Bis – 1 erhöhe Von + 1

                        Min = sucheMinimum(Von, Bis)

                        Vertausche(Von, min, ZuSortierendeObjekte);

}

sucheMinimum(Von, Bis) {

        minimum = Von

                Solange vonNeu = Von + 1,  vonNeu <= Bis erhöhe vonNeu + 1

                        Wenn Von < vonNeu

                                Minimum = vonNeu

}

Vertausche(Von, Min, ZuSortierendeObjekte) {

        Von merken

        ZuSortierendeObjekte[Von] mit dem Objekt an der Stelle Min ersetzen

        Gemerktes Von an der Stelle ZuSortierendeObjekte[Min] einfügen

}
```

Algorithmus Insertionsort:

- Der Algorithmus insertionSort soll ebenfalls rekursiv implementiert werden.
- Der Funktionsaufruf soll dabei so aussehen: insertionSort(Array, Von, Bis)
- Bei einem Insertionsort geht man davon aus, dass das erste Element schon sortiert ist am Anfang.

Pseudocode:

```
insertionSort(ZuSortierendeObjekte, Von, Bis) {
    Von = 2
    Bis = laenge(ZuSortierendeObjekte)
    Solange Von <= Bis erhöhe Von + 1
    {
    WertZumEinsortieren = zuSortierendeObjekte[Von]
        VonNeu = Von
    }
    Solange VonNeu > 1  UND ZuSoriterendeObjekte[j-1] > WertZumEinsortieren
    {
        ZuSortierendeObjekte[VonNeu] = ZuSortierendeObjekte[VonNeu -1]
        VonNeu = VonNeu – 1
        ZuSortierendeObjekte[VonNeu] = WertZumEinsortieren
    }
```