Aufgabe 1:

Die Hochschulsportkurse haben eine neue digitale Schnittstelle. Wenn man sich die angebotenen Sportkurse für das aktuelle Semester anzeigen lässt, kann man einen Tag einstellen und bekommt dann eine Liste von Objekten `Sportkurs`, die an dem Tag stattfinden. Jedes dieser Objekte hat einen Startzeitpunkt s und einen Endzeitpunkt e.

Formal beschrieben bekommt man also für einen Tag eine Menge von Sportkursen $S = \{s_0, \ldots, s_n\}$. Jeder Kurs ist ein Intervall $s_i = (a_i, e_i)$, mit einem Startzeitpunkt a_i und einem Endzeitpunkt e_i.

Da wir die Anzeige etwas unübersichtlich finden, möchten wir die Intervalle nach ihrem Endzeitpunkt sortieren, also e_i.

Verwenden Sie den Quicksort-Algorithmus aus der Vorlesung um die Kurse im folgendem kleinen Beispiel zu sortieren. Geben Sie anschließend eine sortierte Liste der Sportkurse an.

Notieren Sie beim Sortieren der Übersichtlichkeit halber nur die Endzeiten. Rekonstruieren Sie nachträglich die Positionen der Kurse mit gleicher Endzeit (s_2 und s_5). Haben s_2 und s_5 ihre Reihenfolge getauscht?

Markieren Sie ihr Pivotelement und geben Sie die resultierenden Teilarrays in jeder Iteration an.

$$S = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9\}$$

$$s_0 = (13, 19), s_1 = (11, 12), s_2 = (16, 18), s_3 = (21, 23), s_4 = (17, 21),$$

$$s_5 = (17, 18), s_6 = (14, 16), s_7 = (18, 20), s_8 = (20, 22), s_9 = (11, 14)$$

Aufgabe 2:
Wir möchten an einem bestimmten Tag möglichst viele Hochschulsportkurse besuchen. Wir benutzen die neue Schnittstelle aus Aufgabe 1 und erhalten eine Liste an Sportkursen zu denen wir uns anmelden könnten. Wenn wir uns zu einem Kurs anmelden und es Überschneidungen mit anderen Kursen gibt, können wir uns zu diesen Kursen nicht mehr anmelden. Da wir uns Teleportieren können, stört es uns nicht, wenn wir zwei Kurse besuchen für die gilt $e_i = a_j$.

Implementieren Sie einen Greedy Algorithmus, der die Anzahl der besuchten Sportkurse maximiert. Für die Implementierung können Sie annehmen, dass alle Sportkurse zu einer vollen Stunde Anfangen und Aufhören. Sie können die a_i und e_i also als integer modellieren. Beantworten Sie bitte während der Implementation folgende Fragen:

- Welche Entscheidung sollte Greedy getroffen werden? Überlegen Sie ob es für ihre Greedy Entscheidung ein Gegenbeispiel gibt, bei dem die Greedy Entscheidung eine optimale Lösung blockiert.

  Beispiel: Man könnte versuchen die kürzesten Sportkurse zu wählen. Für eine Instanz der Form

  $$S = \{s_0, s_1, s_2\} \text{ mit } s_0 = (8, 15), s_1 = (16, 23), s_2 = (14, 17)$$

  findet dieser Ansatz aber nicht das Optimum.

- Gibt es ein preprocessing Schritt?

  Preprocessing bedeutet, dass die gegebene Datenmenge im Vorhinein bearbeitet wird.

- Was ist die Laufzeit ihres Algorithmus?