

186.866 Algorithmen und Datenstrukturen VU

Übungsblatt 1

für die Übung am Montag den 19. bzw. Dienstag den 20. März 2018.

Geben Sie bis **spätestens Freitag, 16.03.2018, 13:00 Uhr** über TUWEL an, welche Beispiele Sie bearbeitet und gelöst haben. Gehen Sie dabei folgendermaßen vor:

- TUWEL (<https://tuwel.tuwien.ac.at>)
Kurs *186.866 Algorithmen und Datenstrukturen (VU 5.5)* im Abschnitt *Übungsblätter*
- Bearbeitete Beispiele ankreuzen **und** abgeben
 - Link *Ankreuzen Übungsblatt 1*
Button *Abgabe bearbeiten*
Bearbeitete Beispiele anhängen und *Änderungen speichern*.
 - Link *Hochladen Lösungen Übungsblatt 1*
Button *Abgabe hinzufügen*
PDF-Datei mit Lösungen hochladen und *Änderungen sichern*.
 - Wenn Sie das Programmierbeispiel gelöst haben:
Link *Hochladen Source-Code Übungsblatt 1*
Button *Abgabe hinzufügen*
Java-Datei (*E1.java*) hochladen und *Änderungen sichern*.

Bitte beachten Sie:

- Sie können **vor** der Deadline beliebig oft ihre Auswahl an Beispielen und das zugehörige Lösungs-PDF verändern, aber **nach** der Deadline gibt es **keine** Veränderung ihrer angekreuzten Beispiele **und** der abgegebenen Dateien!
- Sie können Ihre Lösungen entweder direkt in einem Textverarbeitungsprogramm erstellen, oder aber auch gut leserliche Scans bzw. Fotos von handschriftlichen Ausarbeitungen einreichen.
- Bitte geben Sie Ihren Namen, Matrikelnummer und E-Mail-Adresse in den Ausarbeitungen an.
- Wenn Sie das Programmierbeispiel kreuzen, muss sowohl die theoretische Ausarbeitung im PDF, als auch der Source-Code im Tuwel abgegeben werden.
- Beachten Sie die Richtlinien für das An- und Aberkennen von Beispielen in den Vorbesprechungsfolien. Neben der Überprüfung in der Übungseinheit werden danach stichprobenartig weitere Abgaben überprüft.

Aufgabe 1 Vier Schülerinnen (A, B, C, D) gehen auf Schüleraustausch. Jede von ihnen soll einer Gastfamilien (W, X, Y, Z) zugeordnet werden. Sowohl Schülerinnen, als auch Gastfamilien gaben je eine Reihenfolge an, mit wem sie gerne zusammen wohnen möchten. Finden Sie ein Stable Matching mithilfe des Gale-Shapley-Algorithmus aus den Vorlesungsfolien. Lässt der Algorithmus offen, welche Schülerin/Familie als nächstes betrachtet werden soll, dann gehen Sie in alphabetischer Reihenfolge vor. Stellen Sie die einzelnen Berechnungsschritte in Form der aktuell fixierten Paare dar (alle Zwischenschritte sollen angegeben werden).

	1.	2.	3.	4.		1.	2.	3.	4.
W	B	A	D	C	A	Z	Y	X	W
X	B	C	D	A	B	X	Y	W	Z
Y	A	B	D	C	C	Z	X	Y	W
Z	D	C	B	A	D	Z	W	Y	X

Aufgabe 2 Gegeben ist die folgende Funktionen:

$$f(n) = 2^n(100n^2 + 20n + 7)$$

$$g_1(n) = \begin{cases} 100n^2 + 2^n \cdot n^2 + 7n + \left(\frac{3}{2}\right)^n & \text{falls } (n < 10^3) \text{ oder } (n \geq 10^5) \\ 5^n + 2^{\frac{3n}{2}} + 100n & \text{sonst} \end{cases}$$

$$g_2(n) = \frac{2^{2^n}}{10000}$$

$$g_3(n) = \begin{cases} 60 & \text{falls } n \text{ eine Primzahl ist} \\ n^{n^n} & \text{sonst} \end{cases}$$

Kreuzen Sie in der folgenden Tabelle die zutreffenden Felder an und begründen Sie Ihre Antworten:

$f(n)$ ist in	$\Theta(\cdot)$	$O(\cdot)$	$\Omega(\cdot)$	keines
$g_1(n)$				
$g_2(n)$				
$g_3(n)$				

Hinweis: Setzen Sie statt dem Punkt die entsprechende Funktion (g_1 , g_2 bzw. g_3) ein. Beispielsweise ist die Zelle links oben als „ $f(n)$ ist in $\Theta(g_1(n))$ “ zu lesen.

Aufgabe 3 Bestimmen Sie die Laufzeit für den folgenden Pseudocode in Abhängigkeit von n in Θ -Notation.

```
 $i \leftarrow n$   
 $j \leftarrow n$   
 $k \leftarrow 0$   
while  $i > 0$   
     $j \leftarrow j \cdot n$   
     $i \leftarrow i - 1$   
while  $j > 0$   
    if  $j = \lfloor \frac{j}{2} \rfloor \cdot 2$  then  
         $k \leftarrow k + 1$   
     $j \leftarrow \lfloor \frac{j}{2} \rfloor$ 
```

Aufgabe 4 Bestimmen Sie die Laufzeit für den folgenden Pseudocode in Abhängigkeit von n in Θ -Notation.

```
 $max \leftarrow 0$   
 $ind \leftarrow 0$   
for  $i \leftarrow 1, \dots, n$   
     $A[i] \leftarrow n$   
if  $n = 1$  then  
    while  $A[1] > 0$   
         $A[1] \leftarrow A[1] - 1$   
else  
    while  $A[n] > 0$   
         $max \leftarrow 0$   
        for  $j \leftarrow 1, \dots, n$   
            if  $A[j] > max$  then  
                 $max \leftarrow A[j]$   
                 $ind \leftarrow j$   
            if  $A[1] > A[2]$  then  
                for  $k \leftarrow 1, \dots, j$   
                     $max \leftarrow max + 1$   
         $A[ind] \leftarrow A[ind] - 1$ 
```

Aufgabe 5 Beweisen oder widerlegen Sie folgende Aussage: Falls $f_1(n) = O(g_1(n))$ und $f_2(n) = O(g_2(n))$ dann gilt $f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$.

Anmerkung: Wir nehmen an, dass alle Funktionen die nicht-negativen ganzen Zahlen als Definitions- und Wertebereich haben.

Aufgabe 6 (Programmieraufgabe) Ihre Aufgabe ist es den Insertionsort-Algorithmus aus der Vorlesung in Java zu implementieren und dabei mit dem in diesem Semester verwendenden Framework vertraut zu werden.

Sie bekommen das benötigte Framework („AlgoDat_E1.zip“) im TUWEL zu Verfügung gestellt. Implementieren Sie darin in der Datei „exercise/E1.java“ die vorgegebene Methode *sort*. Das Framework verwendet dann Ihren Algorithmus um die mitgelieferten Testdaten zu sortieren und stoppt dabei die Zeit. Lassen Sie sich danach mit den enthaltenen Hilfsmitteln einen Plot der Ausführungszeiten in Relation zur Eingabegröße anzeigen und beantworten Sie folgende Fragen:

1. Die Testdaten enthalten aufsteigende, absteigende und zufällige Zahlenfolgen. Welchen Einfluss hat dies auf die jeweilige Ausführungszeit?
2. Begründen Sie aufgrund des Algorithmus warum sich diese Laufzeitunterschiede ergeben.
3. Schauen Sie sich hier speziell folgende zwei Plots genauer an:
 - Alle Instanz-Kategorien eingeschaltet
 - „Zufaellig“ und alle langsameren Instanz-Kategorien abgeschaltet so dass auch die Unterschiede in den schnelleren Kategorien erkennbar werden.

Sie können dazu die einzelnen Instanz-Kategorien durch Klicken auf den Namen ein- und ausschalten.

4. Welche Laufzeit (in O -Notation) hat der Algorithmus in den drei Fälle (auf-, absteigend und zufällig). Stimmt dies mit den theoretischen Überlegungen aus der Vorlesung überein?
5. In welchen Situationen ist Insertion-Sort daher in der Praxis gut geeignet?

Hinweis: Bevor Sie beginnen, lesen Sie sich unbedingt das Tutorial zu den Programmieraufgaben durch („Tutorial.pdf“). Dieses finden Sie im ZIP-Archiv des Programmier-Frameworks im TUWEL. Diese bietet eine kurze Einführung in das verwendete Framework und erklärt Ihnen detailliert was Sie zu tun haben.