

Algorithmen und Datenstrukturen 1

ALGO1 · SoSe-2021 · tcs.uni-frankfurt.de/algo1



★-Aufgabe: Balance

Revision [b511c78](#) (2021-05-07)

Schreibe ein Programm in C/C++, Java, Python, oder einer anderen üblichen Programmiersprache (kein Pseudocode), welches das folgende Problem in linearer Zeit löst: Entscheide, ob ein gegebener String von zwei verschiedenen Arten von Klammern *balanciert* ist, das heißt, ob die zusammengehörigen öffnenden und schließenden Klammern richtig verschachtelt sind. Zum Beispiel ist "`([])([])`" balanciert, aber "`(("` und "`)"` ("`"` und "`)"` nicht.

Um präzise zu sein: (i) der leere String ist balanciert; (ii) wenn w balanciert ist, dann sind sowohl (w) als auch $[w]$ balanciert; und (iii) wenn w und x balanciert sind, dann ist auch wx balanciert.

Eingabe. Die Eingabe besteht aus mehreren Zeilen. Jede Zeile ist eine Sequenz w , die nur Zeichen aus dem Alphabet $\{[,], (,)\}$ enthält.

Ausgabe. Für jede Zeile w , die balanciert ist, gib "1" aus, und sonst "0".

Beispiel.

Eingabedatei:	Ausgabedatei:
<code>([(())])[]</code>	1
<code>)()</code>	0
<code>[]</code>	0
<code>((</code>	0
<code>[(])</code>	0
<code>[][])</code>	0
<code>(</code>	0

Klassifizierte Testfälle. Teste dein Programm! Hier ist ein größeres Beispiel:

- <https://tcs.uni-frankfurt.de/teaching/summer21/algo1/balance-example.in>
- <https://tcs.uni-frankfurt.de/teaching/summer21/algo1/balance-example.out>

Stell sicher, dass dein Programm bei Eingabe `balance-example.in` *exakt* die Ausgabedatei `balance-example.out` erzeugt.

Hinweis. Erwähne dich an die Themen der Woche. Welche Datenstruktur eignet sich, um diese Aufgabe zu lösen? Nutze sie!

Hinweise zur Abgabe. Die Abgabe soll wie immer per PDF erfolgen und die grobe Idee, den diesmal echten Code, den Korrektheitsbeweis und die Laufzeitanalyse enthalten. Außerdem: Welche 20 Zeilen (jeweils 0 oder 1) werden bei Eingabe <https://tcs.uni-frankfurt.de/teaching/summer21/algo1/balance-secret20.in> erzeugt? Die Ausgabe ist in der PDF einzufügen. Weiterhin zu beachten:

- Der Code darf maximal 80 Zeilen lang sein. Möglichst kurz und elegant! (Eine 15-Zeilen Lösung in Python ist möglich. Kommentare zählen nicht dazu.)

- Falls die genutzte Datenstruktur in der Programmiersprache eingebaut ist, darf diese benutzt werden. Falls die Datenstruktur selbst implementiert wird, zählt diese Implementierung nicht zum Zeilenlimit dazu. Wichtig: Die Datenstruktur darf nur so benutzt werden, wie die in der Vorlesung beschriebene abstrakte Datenstruktur das erlaubt. Falls zum Beispiel eine Warteschlange benutzt wird, dürfen nur die entsprechenden Funktionen enqueue, dequeue und is_empty verwendet werden.