41. Bundeswettbewerb Informatik

Dokumentation zur Junioraufgabe 2: „Container“

Teamname: SRZinfo4/1
Team-Id: 00495
Bearbeiter: Jakob Paridon

Dresden, den 21. November 2022

Schülerrechenzentrum der TU-Dresden

Inhaltsverzeichnis

1 Lösungsidee

Alle Container werden in Form eines Baums notiert. So wird garantiert, dass höheres Gewicht weitergetragen wird, d. h., dass $a > b \ \& \ b > c \implies a > c$ gilt.
Befinden sich alle Container in einem Baum mit einer Wurzel, so ist diese Wurzel der schwerste Container. Die Wurzel ist entweder selbst schwerer als alle anderen Container oder schwerer als einige Container, welche wiederum schwerer sind als andere, welche wiederum schwerer sind als andere usw., sodass die Wurzel das höchste Gewicht haben muss.
Erhält man einen Baum mit mehreren Wurzeln, kann der schwerste Container nicht festgestellt werden, da nicht bestimmt werden kann, welche der Wurzeln die schwerste ist: die verschiedenen Wurzeln stehen nur durch die Container, welche leichter als sie selbst sind in Beziehung zueinander, wodurch jedoch der schwerste Container nicht ermittelt werden kann.

2 Umsetzung

Das Programm wird mit `python junior2.py [Name der Datei mit den zu untersuchenden Containern]` bzw. `junior2.exe [Name der Datei mit den zu untersuchenden Containern]` gestartet.
Zuerst wird festgestellt, wie viele Container gewogen wurden. Dazu wird die Nummer des Containers mit der höchsten Zahl gespeichert. Dies bedeutet, dass alle Container aufeinanderfolgende Nummern haben müssen. Ist dies der Fall, bezeichnet der Container mit der höchsten Zahl auch die Menge der gewogenen Container.
Nun wird ein Feld erstellt, welches für jeden Container einen Knoten enthält (siehe 4.1). Jeder Knoten ist dabei ein Objekt, welches die Nummer des Containers sowie all jene Container, die leichter als der untersuchte sind, speichert (siehe 4.2).
Nun werden all diese Knoten in Verbindung zueinander gesetzt. Dabei wird bei jedem gewogenen Paar der leichtere Container an die "Kinder"des schwereren (also jene Container, die leichter als der untersuchte sind), angehängt (siehe 4.3).
Als nächstes wird untersucht, ob ein Container schwerer als alle anderen ist. Hierfür werden rekursiv alle Container, welche diesen selbst, seine Kinder, Kindeskinder, Kindeskindeskinder usw. darstellen, in eine Liste übertragen (siehe 4.4). Ist die Länge der Liste gleich der Menge der gewogenen Container, so befinden sich alle gewogenen Container in ihr. Dies bedeutet, dass jeder gewogenen Container leichter als der untersuchte ist, dieser also den schwersten darstellt.
Ist keiner der Container schwerer als alle anderen, so kann der schwerste Container nicht festgestellt werden.

3 Beispiele

Die Testung erfolgte unter Artix Linux mit Python 3.10.7.

3.1 container0.txt

```
Der schwerste Container ist nicht feststellbar
```

3.2 container1.txt

```
Der schwerste Container ist 4
```

3.3 container2.txt

```
Der schwerste Container ist nicht feststellbar
```

3.4 container3.txt

```
Der schwerste Container ist nicht feststellbar
```

3.5 container4.txt

```
Der schwerste Container ist 5
```

4 Quellcode

4.1 Übertragung aller Container in ein Feld

```
containers = []
for i in range(1, count+1):
    containers.append(Node(i))
```

4.2 Klasse der Container-Knoten

```
class Node:
    def __init__(self, number):
        self.number = number
        self.children = []
```

4.3 Auswertung der Wiegvorgänge

```
for pair in pairs:
    containers[int(pair.split()[0])-1].children.append(containers[int(pair.split()[1])-1])
```

4.4 Übertragung aller Container, die leichter als der untersuchte sind, in eine Liste

```
def getAll(tree):
    if tree:
        if tree.number not in foundChildren:
            foundChildren.append(tree.number)
            for child in tree.children:
                getAll(child)
```