technische universität
dortmund

Fakultät für Informatik
Lehrstühle 11, 13, 14

Jun.-Prof. Dr. Maike Buchin
Łukasz Czajka, N. Kriege, D. Kurz, M. Ludewig

Sommersemester 2019
14. Mai

DAP2 Praktikum – Blatt 7

Abgabe: 20.05.–24.05.

Wichtig: Der Quellcode ist natürlich mit sinnvollen Kommentaren zu versehen. Überlegen Sie außerdem, in welchen Bereichen Invarianten gelten müssen, und überprüfen Sie diese ggf. an sinnvollen Stellen mit *Assertions* (siehe Hinweis auf Blatt 2).

Kurzaufgabe 7.1: Längste gemeinsame Teilfolge (8 Punkte)

Eine längste gemeinsame Teilfolge (Longest Common Subsequence) zweier Folgen a, b ist eine Folge c, für die gilt:

(a) c ist sowohl eine Teilfolge von a als auch eine Teilfolge von b und

(b) es gibt keine andere Teilfolge von a und b, welche länger als c ist.

In der Vorlesung wurde ein Algorithmus vorgestellt, der eine längste gemeinsame Teilfolge zweier Folgen unter Verwendung der Technik der dynamischen Programmierung bestimmt.

Implementieren Sie diesen Algorithmus und stellen Sie die Entwicklung der Laufzeit Ihrer Implementierung für zwei Zufallsfolgen der Länge n in Abhängigkeit von n grafisch dar.

Schreiben Sie ein Programm, welches wie folgt arbeitet:

1. Einlesen des Kommandozeilenparameters n.
2. Erzeugen von zwei Zufallsfolgen a und b der Länge n.
3. Berechnung und Ausgabe der Länge von längsten gemeinsamen Teilfolgen von a und b.
4. Berechnung und Ausgabe einer längsten gemeinsamen Teilfolge von a und b.
5. Messen der Zeit, welche für die Ermittlung der gemeinsamen Teilfolge benötigt wird.

Ihr Programm soll eine (beliebige) längste gemeinsame Teilfolge bestimmen und *ausgeben*. Die Laufzeitentwicklung soll graphisch dargestellt werden.

Verwenden Sie für die Erzeugung von Zufallsfolgen die folgende Funktion:

```
String randStr(int n, Random r) {
        String alphabet =
                "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";
        StringBuilder res = new StringBuilder(n);
        while (--n>=0) {
                res.append(alphabet.charAt(r.nextInt(alphabet.length())));
        }
        return res.toString();
}
```

Hinweise und Tipps

Messung von Laufzeiten

Um Laufzeiten in Programmen zu messen, kann man die Methode `System.currentTimeMillis()` benutzen. Diese gibt die Systemzeit in Millisekunden zurück, und hat den Rückgabetyp `long`. Ein Beispielcode zur Laufzeitmessung sieht dann wie folgt aus:

```
...
long tStart, tEnd, msecs;
...
// Beginn der Messung
tStart = System.currentTimeMillis();

// Hier wird der Code ausgeführt, dessen Laufzeit gemessen werden soll

// Ende der Messung
tEnd = System.currentTimeMillis();

// Die Vergangene Zeit ist die Differenz von tStart und tEnd
msecs = tEnd - tStart;
...
```

Beachten Sie, dass der Garbage-Collector einen Einfluss auf die Laufzeit haben kann. Bei einzelnen Messungen kann dies zu recht großen Laufzeitunterschieden führen. Außerdem kann es aufgrund verschiedener Effekte (wechselnde CPU-Auslastung der Poolrechner, wechselnde CPU-Taktungen bei Laptops und anderen Rechnern) zu Schwankungen in der Laufzeit kommen.

Daher ist es empfehlenswert, jede Messung mehrmals zu wiederholen und den Mittelwert der Laufzeiten zu bestimmen. Dem Garbage-Collector kann auch zwischen den Messungen mittels `System.gc()` empfohlen werden, den Speicher aufzuräumen. Vorher müssen natürlich die betreffenden Referenzen auf `null` gesetzt sein, damit der Speicher auch wirklich freigegeben werden kann.