Algorithmen und Datenstrukturen
Aufgabenblatt 2

Abgabe: 05.05.2022 22:00
Abnahme: 06.05.2022

1. Komplexitätsbestimmung 5 Pkt.

Entwickeln Sie die Regel, nach der sich die Laufzeit-Komplexität im Kalkül von Schleifen bestimmen lässt. Sie können sich hier auf While-Schleifen beschränken. Orientieren Sie sich dabei am Beispiel aus der Vorlesung, in dem if-Ausdrücke betrachtet wurden.

Wenden Sie Ihre Regel an auf das folgende Programmfragment:

```
eingabe(n) //integer, O(1)
i = 0                    O(A) = 1
zahl = 0
while i <= n*n           O(C) = n^2
 if i gerade
  zahl += i              O(B) = 1
 else
  zahl += 2*i
 i++
```

<A> <C> <B>

Bsp:

= O(max(1, 1·n²))

liegt in O(n²)

Worst case Ausführung:
<A>
<C>
<B> (O(c) mal)

Alle drei Programmteile werden ausgeführt, die Häufigkeit der Ausführung von <B> hängt von O(c) ab.

Ann: <A> liegt in O(A), <B> liegt in O(B), <C> ∈ O(C)

While Schleife:
O(A) + O(B)·O(C)

⇒ Regel: O(max(A, O(B·C))

2. Aufwandsabschätzungen 5 Pkt.

Hier sollen Sie Regeln entwickeln zur Aufwandsabschätzung von rekursiven Funktionen im allgemeinen Fall. Dazu schätzen Sie sukzessive die Komplexität des in der Vorlesung vorgestellten rekursiven Algorithmus für die Berechnung der maximalen Teilsumme mit Hilfe der Strategie **Teile-und-herrsche** ab. Begründen Sie Ihre Antwort.

Beantworten Sie dazu zunächst folgende Fragen, die Ihnen helfen sollen, die Regeln zu finden:

Teilaufgabe 2.1: (½ Pkt)

Welche Komplexität haben die Funktionen `rechtesRandmax` und `linkesRandmax`?

Teilaufgabe 2.2: (½ Pkt)

Welche Komplexität hat die Funktion `maxTeilsummerek`, wenn man die Zeile mit den rekursiven Aufrufen nicht berücksichtigt?

Teilaufgabe 2.3: (½ Pkt)

Welche Komplexität hat die Funktion `maxTeilsummerek`, wenn man annimmt, dass sich die Anzahl der rekursiven Aufrufe mit $f(n)$ berechnen lässt.

Teilaufgabe 2.4: (½ Pkt)

Wie oft kann man eine Folge der Länge 2, der Länge 4, der Länge 8, der Länge 16, ... in zwei gleiche Hälften teilen, d. h. wie oft kann man diese Länge halbieren, bis man (in einem Ast) bei 1 angelangt ist?

Teilaufgabe 2.5: (½ Pkt)

Wie oft kann man eine Folge der Länge 27, der Länge 173 oder der Länge 291 in zwei möglichst gleichlange Teilfolgen aufteilen?

Teilaufgabe 2.6: (½ Pkt)

Wie oft kann man eine Folge der Länge n in zwei möglichst gleichlange Folgen aufteilen?

Teilaufgabe 2.7: (½ Pkt)

Wie oft wird $maxTeilsummerek$ bei einer Folgenlänge von n aufgerufen? bitte beachten: Zwei "gleichwertige" Aufrufe hintereinander bedeuten zwar doppelten Aufwand, der Faktor 2 wird aber nicht berücksichtigt bei der Worst-Case-Betrachtung, wie Sie wissen.

Teilaufgabe 2.8: (½ Pkt)

Welche Komplexität hat $maxTeilsummerek$? Begründen Sie Ihre Antwort mit Hilfe der Antworten auf die vorangehenden Fragen.

Teilaufgabe 2.9: (1 Pkt)

Wie berechnet man die Laufzeit im O-Kalkül für beliebige rekursive Funktionen? Begründen Sie Ihre Antwort mit Hilfe der Antworten auf die vorangehenden Fragen.

Ziel: Komplexitätsabschätzung von rekursiven Funktionen.