Algorithmen und Datenstrukturen

Wintersemester 2023/24

Prof. Dr. Georg Schied

Aufgabenblatt 2

verlängert bis So. 29. Oktober 2023, 23:59 Uhr

Abgabetermin: ~~Do. 26. Oktober 2023, 23:59 Uhr~~

Zum Bestehen müssen 10 von 20 Punkten erreicht werden.

Aufgabe 2.1

Implementieren Sie eine Methode

```
public static TreeSet<String> nestedBrackets(int n)
```

die alle Strings aus *n* Paaren runder Klammern erzeugt (für $n \geq 1$), die korrekt geschachtelt sind. Klammergruppen dürfen ineinander geschachtelt sein oder nebeneinander stehen. Beispielsweise gibt es für *n*=3 folgende fünf Möglichkeiten:

```
((()))
(()())
(())()
()(())
()()()
```

Verwenden Sie einen rekursiven Ansatz. Sie finden in Moodle dazu eine Programmvorlage und JUnit-Tests.
Tipp: Folgende kontextfreie Grammatik erzeugt alle korrekt geschachtelten Klammerfolgen.

$K \rightarrow$ `()` | `(` K `)` | $K\,K$

Aufgabe 2.2 - Scheinaufgabe (10 P)

Programmieren Sie eine Methode

```
public static TreeSet<String> palindroms(int length,
                                         char[]chSet)
```

die alle Palindrome der Länge `length` berechnet, die mit den Zeichen aus der Zeichenmenge `chSet` gebildet werden können. Beispielsweise gibt es für die Zeichenmenge { `a`, `b` } bei Länge 2 die Palindrome

```
aa, bb
```

bei Länge 3 die Palindrome

```
aaa, aba, bab, bbb
```

und bei Länge 4 die Palindrome:

```
aaaa, abba, baab, bbbb
```

Verwenden Sie dazu einen rekursiven Lösungsansatz (es dürfen aber auch Schleifen mit verwendet werden). In Moodle finden Sie eine Programmvorlage und JUnit-Tests.

Tipp: Ist *p* ein Palindrom der Länge *n* und c ein Zeichen, dann ist c*p*c ein Palindrom der Länge *n*+2.

Aufgabe 2.3 - Scheinaufgabe (5 P)

a) Ein Array mit den Werten

[7, 4, 3, 9, 2, 5]

ist gegeben. Zeigen Sie, wie *Insertionsort* das Array sortiert. Geben Sie den Inhalt des Arrays nach jeder Iteration der äußeren Schleife an und kennzeichnen Sie den Teil des Arrays, der bereits sortiert ist.

b) Betrachten Sie die innere While-Scheife von Insertionsort (Algorithmus 4.4). Welche der folgenden Zusicherungen sind Schleifeninvarianten für diese Schleife (jeweils mit kurzer informeller Begründung).

(1) `insertPos` ≤ `j`

(2) 0 < `insertPos`

Aufgabe 2.4 - Scheinaufgabe (5 P)

Die folgende Methode soll 2^n für $n \geq 0$ berechnen.

```
public static int power2(int n) {
  // Vorbedingung: n ≥ 0
  int res = 1;
  int k = 0;
  while (k < n) {
       res = res + res;
       k++;
  }
  return res;
  // Nachbedingung: res = 2^n
}
```

Welche der folgenden Zusicherungen sind Invarianten für die while-Schleife? (mit kurzer Begründung).

(1) k ≥ 0

(2) k < n

(3) k ≤ n

(4) res = 2^{k+1}

(5) res = 2^k

Aufgabe 2.5

Folgende Methode *quadrat*(*n*) soll für Argumente $n \geq 0$ den Wert n^2 berechnet.

```
public static int quadrat(int n) {
  int q = 0;
  int i = 1;
  while (i <= n) {
       q = q + 2 * i - 1 ;
       i++;
  }
  return q;
}
```

a) Welche der folgenden Zusicherungen sind eine Invariante für die While-Schleife? Tipp: Es gilt $k^2 = (k\text{-}1)^2 + 2k - 1$.

(1) $q = i^2$

(2) $q = (i\text{-}1)^2$

(3) $q = (i\text{-}1)^2 \wedge i \leq n+1$

b) Zeigen Sie mit Hilfe einer geeigneten Schleifeninvariante, dass am Ende $q = n^2$ gilt, sofern $n \geq 0$.

Aufgabe 2.6

a) Zeigen Sie mit Hilfe einer geeigneten Schleifeninvariante, dass bei Methode `power2` aus der vorigen Aufgabe die Nachbedingung $res = 2^n$ gilt (vorausgesetzt $n \geq 0$).

b) Geben Sie eine geeignete Terminierungsfunktion an, mit der gezeigt werden kann, dass die Schleife für alle $n \geq 0$ terminiert..