universität wien

Algorithmen und Datenstrukturen 1

Theoretische Prüfung

Haupttermin

28.06.2022

Name:	
Matrikelnummer:	

Die Angaben sind beidseitig bedruckt!

13	11	32	15	18	23	39	34
+	+	+	+	+	+	+	+
z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8

Aufgabe 1 [2]

Fügen Sie in obiger Tabelle in den leeren Kästchen, vor denen das Pluszeichen steht, die Ziffern Ihrer Matrikelnummer ein. Führen Sie die Additionen durch und ermitteln Sie die Zahlen z_1 bis z_8.

Aufgabe 2 [18]

Gegeben sind folgende Funktionen:

```
void g(int i, int n) {
  if (i>0) {
    for (int j=n+10; j>0; j-=5)
      g(i-2, n);
  }
}

void f(int n) {
  if (!n) return;
  f(n/(z4%10+2));
  g(z6%5+1, n);
  for (int i=0; i<z7%10; i=i+2)
    f(n/(z4%10+2));
  g(z6%5+1, n);
}
```

Berechnen Sie die Laufzeit der Funktion `f` in Θ-Notation abhängig von `n`. Setzen Sie dazu für z_4, z_6 und z_7 die in Aufgabe 1 ermittelten Werte ein.

(Hinweis: Erstellen Sie Rekurrenzgleichungen für die Laufzeiten von g bzw. f und lösen Sie diese mittels fortgesetztem Einsetzen bzw. Master Theorem.)

Aufgabe 3 [20]

Die Werte z_1 bis z_8. (aus Aufgabe 1) seien in dieser Reihenfolge von links nach rechts in einem Array gespeichert. Sortieren Sie die Werte aufsteigend mit

a) [10] Quicksort
b) [4] Selectionsort
c) [6] Heapsort

Geben Sie alle notwendigen Schritte so genau an, dass die Arbeitsweise des Algorithmus klar ersichtlich wird.

Aufgabe 4 [20]

a) [10] Fügen Sie die Werte z_2 bis z_8 aus Aufgabe 1 (in dieser Reihenfolge) in eine zu Beginn leere Hashtabelle der Länge 7 ein.
Verwenden Sie als Hashfunktion $h(k) = k\%7$ und double hashing zur Kollisionsbehandlung.
Die zweite Hashfunktion ist $g(k) = k\%5 + 1$.
Skizzieren Sie den Zustand der Hashtabelle nach jedem Einfügeschritt.

b) [2] Löschen Sie den Wert z_7 aus der Tabelle und skizzieren Sie den Zustand der Hashtabelle.

c) [4] Geben Sie den Kollisionspfad (besuchte Indexpositionen) bei einer Suche nach dem Wert z_2 an.

d) [4] Geben Sie den Kollisionspfad (besuchte Indexpositionen) bei einer Suche nach dem Wert 49 an.

Aufgabe 5 [20]

a) [8] Fügen Sie die Werte z_1 bis z_8 aus Aufgabe 1 (in dieser Reihenfolge) in einen zu Beginn leeren binären Suchbaum ein. Skizzieren Sie den Zustand des Baums nach jedem Einfügeschritt.
Anmerkung: der Baum kann Werte mehrfach enthalten.

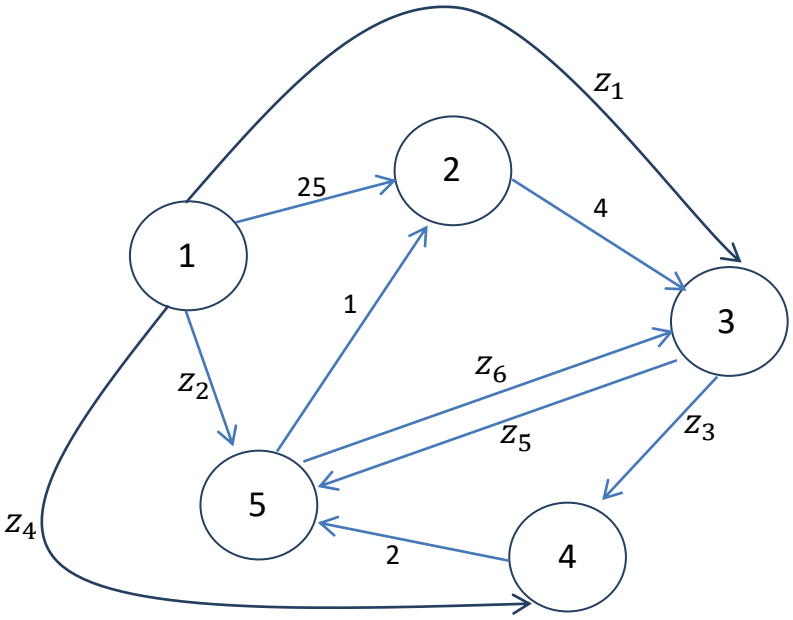
b) [3] Geben Sie in C++ ähnlicher Notation die Definition einer Datenstruktur für einen binären Suchbaum an.

c) [5] Geben Sie in C++ ähnlicher Notation eine Definition einer effizienten Funktion oder Methode an, welche überprüft ob ein Wert (Parameter s) im Baum vorhanden ist. Es soll true im Erfolgsfall und ansonsten false zurückgegeben werden.

d) [4] Bestimmen Sie die Laufzeitkomplexität Ihrer Funktion abhängig von der Anzahl (n) der im Suchbaum gespeicherten Werte in Big-O-Notation. Begründen Sie Ihr Ergebnis kurz.

Aufgabe 6 [20]

Gegeben ist der folgende gerichtete Graph
(die Werte z_1 bis z_6. sind aus Aufgabe 1 zu übernehmen):



a) [3] Skizzieren Sie die Adjazenzliste des Graphen.
b) [10] Bestimmen Sie mit dem Algorithmus von Dijkstra die jeweils kürzesten Wege vom Knoten 1 zu allen anderen Knoten des Graphen.
c) [3] Ist der oben dargestellte Graph topologisch sortierbar? Falls ja, geben Sie eine topologische Sortierung an, andernfalls begründen Sie, warum eine solche nicht gefunden werden kann.

d) [4] Welche der folgenden Voraussetzungen ist hinreichend, damit der Dijkstra-Algorithmus das korrekte Resultat liefert? Zutreffendes bitte ankreuzen.

(1) Alle Kantengewichte des Eingabegraphen sind nicht-negativ.

(2) Der Eingabegraph enthält keinen negativen Kreis.

(3) Der Eingabegraph enthält einen negativen Kreis.

(4) Der Eingabegraph ist ein DAG.