26	32	22	16	14	27	19	10
+	+	+	+	+	+	+	+
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]

a. [10] Erstellen Sie in C++-ähnlichem Pseudocode eine rekursive Funktion f mit einem ganzzahligen Parameter n, auf die das Mastertheorem anwendbar ist und deren Laufzeit in $\Theta(n^2)$ liegt. Der Wert des Parameters c im Mastertheorem darf dabei nicht 2 sein.
b. [3] Zeigen Sie mit Hilfe des Mastertheorems, dass f die gewünschte Laufzeit hat.
c. [5] Erstellen Sie in C++-ähnlichem Pseudocode eine Funktion g mit einem ganzzahligen Parameter n, die f aufruft und deren Laufzeit in $\Theta(n^3 \log n)$ liegt.

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. [8] Quicksort
b. [4] Mergesort
c. [8] 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 Zahlen z_1 bis z_7 (aus Beispiel 1) in dieser Reihenfolge in eine ursprünglich leere Hashtabelle der Größe 7 ein. Verwenden Sie die Hashfunktion $h(x) = x\%7$ und zur Kollisionsbehandlung Double Hashing mit $g(x) = x\%5 + 2$ als zweite Hashfunktion. Geben Sie den Zustand der Hashtabelle nach jeder Einfügeoperation an.
b. [4] Geben Sie den Kollisionspfad an, der durchsucht wird, wenn versucht wird, in der nach Punkt a befüllten Hashtabelle zusätzlich z_8 (aus Beispiel 1) einzufügen.
c. [2] Wozu wird beim Double Hashing die Markierung „wiederfrei“ verwendet?
d. [2] Warum wäre für Double Hashing eine zweite Hashfunktion der Form $g(x) = x\%5$ nicht empfehlenswert?
e. [2] Nennen Sie außer Double Hashing zwei weitere statische Hashverfahren (Angabe der Namen ist ausreichend).

Aufgabe 5 [20]

a. [5] 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. [5] Entfernen Sie aus dem im Punkt a erhaltenen Baum nun immer wieder die Wurzel, bis der Baum leer ist. Skizzieren Sie den Zustand des Baums nach jedem einzelnen Löschvorgang.
c. [5] Geben Sie in C++ ähnlichem Pseudocode eine Definition einer Datenstruktur für einen binären Suchbaum an.
d. [5] Geben Sie in C++ ähnlichem Pseudocode eine Definition einer Funktion an, die das Maximum und das Minimum der im Baum gespeicherten Werte ermittelt und ausgibt. (Bei Bedarf können Sie auch weitere Hilfsfunktionen definieren, die von der Hauptfunktion aufgerufen werden.)

Aufgabe6 [20]

Gegeben ist die folgende Adjazenzmatrix mit Wegekosten für einen gerichteten Graphen
(die Werte $\boldsymbol{z_1}$ bis $\boldsymbol{z_8}$. sind aus Aufgabe 1 zu übernehmen):

$$\begin{pmatrix} 0 & z_6 & 0 & 58 & z_4 \\ 2 & 0 & z_3 & z_8 & 0 \\ z_2 & 0 & 4 & 0 & 7 \\ 0 & 1 & 0 & 0 & z_7 \\ 42 & 2 & z_5 & 0 & 0 \end{pmatrix}$$

a. [2] Skizzieren Sie den gerichteten Graphen.
b. [10] Bestimmen Sie mit dem Algorithmus von Dijkstra die jeweils kürzesten Wege vom Knoten 1 (erste Zeile, erste Spalte der Matrix) zu allen anderen Knoten des Graphen.
c. [8] Bestimmen Sie mit dem Algorithmus von Kruskal einen minimal spannenden Baum des Schattens des Graphen. (Sie erhalten den Schatten des Graphen, indem Sie die Richtungen der Kanten vernachlässigen. Werden dann zwei Knoten durch zwei Kanten verbunden, so werden diese Kanten zu einer zusammengefasst. Anders ausgedrückt: Zwei Knoten x und y im Schatten sind genau dann durch eine ungerichtete Kante verbunden, wenn im ursprünglich gerichteten Graphen zumindest eine der Kanten von x nach y oder von y nach x existiert. Als Gewicht der ungerichteten Kante wählen sie jeweils das Minimum aller durch sie repräsentierten gerichteten Kanten.)