	22	**17**	**38**	**27**	**29**	**32**	**39**
	+	**+**	**+**	**+**	**+**	**+**	**+**
21							
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_2 bis z_8. (z_1 ist bereits mit dem fixen Wert 21 belegt.)

Aufgabe 2 [18]

a) [13] Erstellen Sie in C++-ähnlichem Pseudocode **eine** rekursive Funktion f mit einem ganzzahligen Parameter n, auf die das Mastertheorem anwendbar ist und deren Laufzeitkomplexität **gleichzeitig** in $O(n^n)$, $\Theta(n^4 \log n)$, $\Omega(\log^2 n)$ und liegt. Der Parameter b im Mastertheorem muss dabei den Wert $z_4 \,\%\, 2 + 3$ annehmen.

b) [5] Zeigen Sie mit Hilfe des Mastertheorems, dass f die gewünschte Laufzeitkomplexität hat.

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 (Verwenden Sie als Pivotelement immer den ersten - ganz linken - Wert).
b. [6] Heapsort
c. [4] Selection Sort

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

Aufgabe 4 [20]

a) [5] Fügen Sie Ihren Vornamen und Ihren Zunamen der folgenden Liste von Zeichenketten hinzu: ''balg'',''bald'',''balduin'',''an'',''anders''. Fügen Sie dann alle Zeichenketten in einen Trie ein und skizzieren Sie den Zustand des Tries, nachdem alle Zeichenketten eingefügt wurden.

b) [5] Geben Sie in C++ - ähnlichem Pseudocode eine Definition für die Datenstruktur eines de la Briandais Tries an. (Die benötigten Instanzvariablen und ein Konstruktor für die Erstellung eines leeren Tries reichen aus.)

c) [10] Entwerfen Sie für Ihre Datenstruktur eine passende Methode, die die größte Zeichenkette, die im Trie gespeichert ist (also jene , die in einer alphabetisch geordneten Liste aller im Trie gespeicherten Zeichenketten ganz am Schluss stehen würde), ausgibt.

Aufgabe 5 [20]

a) [10] Fügen Sie die Zahlen $\boldsymbol{z_1}$ bis $\boldsymbol{z_7}$ (aus Beispiel 1) in dieser Reihenfolge in eine ursprünglich leere Hashtabelle der Größe 7 ein. Verwenden Sie die Hashfunktion $\boldsymbol{h(x) = x\%7}$ und zur Kollisionsbehandlung double Hashing mit $\boldsymbol{g(x) = x\%3 + 1}$ als zweiter 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 a) befüllten Hashtabelle zusätzlich $\boldsymbol{z_8}$ (aus Beispiel 1) einzufügen.

c) [2] Wozu wird beim double Hashing die Markierung „wiederfrei“ verwendet?

d) [2] Warum ist es empfehlenswert, für double Hashing eine Tabellengröße zu verwenden, die eine Primzahl ist?

e) [2] Geben Sie die Namen von jeweils zwei statischen und zwei dynamischen Hashverfahren an.

Aufgabe 6 [20]

Gegeben ist die folgende Adjazenzmatrix, die die Kosten der Verbindungen zwischen den Knoten eines gerichteten Graphen beschreibt:

$$\begin{pmatrix} 0 & z_6 & 0 & 34 & z_4 \\ 2 & 0 & z_3 & z_8 & 0 \\ z_2 & 0 & 4 & 0 & 7 \\ 0 & 1 & 0 & 0 & z_7 \\ 28 & 2 & z_5 & 0 & 0 \end{pmatrix}$$

a) [2] Skizzieren Sie den Graphen, der durch diese Adjazenzmatrix beschrieben wird.

b) [10] Bestimmen Sie mit dem Algorithmus von Dijkstra die kürzesten Wege vom Knoten 1 zu allen anderen Knoten des Graphen (Dabei entspricht Knoten 1 dem Knoten der ersten Zeile/Spalte in der Adjazenzmatrix).

c) [8] Entfernen Sie eine möglichst kleine Anzahl von Kanten, sodass der Graph topologisch sortierbar wird (führen Sie genau an, welche Kanten entfernt werden müssen) und führen Sie eine topologische Sortierung durch (es reicht, die Abfolge der Knoten anzugeben, Sie müssen nicht den Graphen für das Resultat zeichnen).