Hinweis für alle Aufgaben
Nutzen sie eine einfache Konsolenanwendung für die Aufgaben, d.h. keine Verwendung einer grafischen Oberfläche.
Folgende Programmiersprachen sind zugelassen: C, C++ und Java
Entwicklungsumgebung: keine Vorgabe
Die Aufgaben sind von jedem Studierenden eigenständig zu lösen.

Abnahme
Die korrekte Funktion der erstellten Anwendungen wird durch Herrn Schirdewahn oder Herrn Staemmler geprüft und abgenommen. Erst dann kann entsprechend der Wertigkeit eine Anrechung erfolgen. Maximal können 20 Punkte = 20% der Prüfungsleistung erbracht werden.

Aufgabe 1.1 – Wertigkeit = 2
Nutzen Sie eine doppelt verkettete Liste zur Ablage von Stammdaten (Vorname, Nachname, Geburtsdatum, fiktive Versicherungsnummer). Implementieren Sie sie folgenden Methoden: print element, insert element, delete element, find element (gemäß einem Suchkriterium), count elemente, print list, write list to disk, read list from disk.

Aufgabe 1.2 – Wertigkeit = 1
Nutzen Sie Ihre Lösung von Aufgabe 1.1 und implementieren Sie die obigen Methoden mit einer Datenstruktur, die ein Feld von Records mit Stammdaten verwendet.

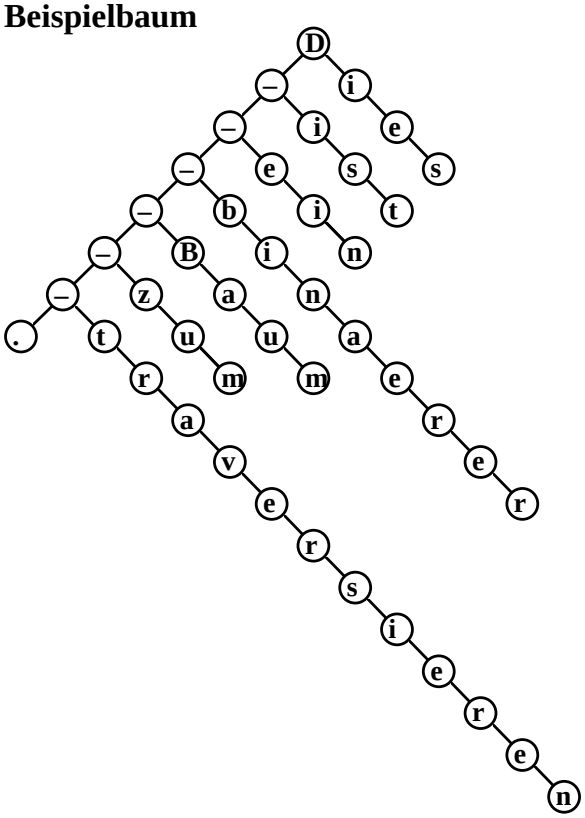
Aufgabe 2.1 – Wertigkeit = 2
Realisieren Sie einen "Spooler" für Tastatureingaben. Der Spooler (entspricht der Datenstruktur Schlange) soll eine Kapazität von 20 Bytes besitzen. Zunächst "sammelt" der Spooler die eingegebenen Zeichen (außer Steuerzeichen) auf, erst wenn der Spooler zur Hälfte gefüllt ist, erfolgt die Ausgabe. Als Steuerzeichen gelten z.B.: CTRL-S (0x13) hält die Ausgabe an, CTRL-Q (0x11) setzt die Ausgabe fort, '0' gibt alle noch im Spooler vorhandenen Zeichen aus und beendet das Programm. Stellen Sie sicher, dass der Spooler bei kontinuierlicher Eingabe nicht überlaufen kann bzw. geben Sie entsprechende Fehlermeldungen.
Implementieren Sie den Spooler unter Verwendung eines Feldes und/oder mit Hilfe einer verketteten Liste.

Aufgabe 2.2 – Wertigkeit = 1
Implementieren Sie den Euklidischen Algorithmus als rekursive Prozedur. Erweitern Sie die Ihre Implementation derart, das Sie zum einen die Anzahl der Rekursionen ermitteln und während des Ablaufs die Folge der Aufrufe protokollieren und damit in der Lage sind, den sogenannten Aufruf Stack (siehe auch z.B. MS Visual C++ → Ansicht → Aufruf Stack) auszugeben.

Aufgabe 3.1 – Wertigkeit = 1

Generieren Sie den rechts stehenden Beispielbaum durch eine verkettete Liste.
Realisieren Sie dazu Methoden die es erlauben, an einen vorhandenen Knoten einen neuen Knoten links bzw. rechts „anzuhängen“.
Vereinfachen Sie ggf. den Aufbau des Baums durch eine Methode, die es erlaubt nach rechts eine „Kette“ von Knoten entsprechend der Länge der übergebenen Zeichenkette anzuhängen.

Beispielbaum



Aufgabe 4.1 – Wertigkeit = 1
Implementieren Sie unter Verwendung des Beispielbaums aus der Aufgabe 4.1 die verschiedenen Methoden für das Traversieren eines Baumes. Verwenden Sie für die Methoden inorder, preorder und postorder entweder einen rekursiven Aufruf oder einen Stack, für die Methode level order bietet eine Schlange gute Möglichkeiten.
Zeigen Sie mittels der Ausgabe der in jedem Knoten abgelegten Zeichen die Reihenfolge des Traversierens auf.

Aufgabe 4.2 – Wertigkeit = 1
Ausgehend von den implementierten Methoden zum Traversieren ermitteln Sie die Kenngrößen des Beispielbaums aus Aufgabe 4.1, z.B. Höhe, Knotenanzahl, maximaler Pfad, Pfadlänge, Blätter, etc. .

Vorgabe: Nachstehend finden Sie eine "Hitliste" von 20 Vornamen (www.firstname.de): "Alexander, David, Felix, Maximilian, Leon, Lukas, Luca, Paul, Jonas, Tim, Anna, Emily, Julia, Maria, Laura, Lea, Lena, Leonie, Marie, Sophie".

Aufgabe 5.1 – Wertigkeit = 2
Implementieren Sie Hash Funktionen $h_i(x)$ zur Ablage dieser Vornamen in einen Feld mit m=32 Speicherorten. Realisieren Sie zudem ein Verfahren zur Generierung des Schlüssels aus dem Vornamen. Protokollieren Sie bei der Ablage der Vornamen die Anzahl der Kollisionen pro Vorname und die Summe der Kollisionen. Geben Sie das Feld nach vollständige Ablage der Vornamen mit der jeweiligen Statusinformation aus.

Aufgabe 5.2 – Wertigkeit = 1
Legen Sie diese Vornamen in einem binären Suchbaum ab. Verwenden Sie zwei Reihenfolgen bei der Ablage:
1. Verwenden Sie die in der Vorgabe dargestellte Reihenfolge zur Einspeicherung.
2. Gehen Sie dabei von der Reihenfolge aus, die mit aufsteigenden Feldindex aus Aufgabe 5.1 vorliegt.

Zeigen Sie durch eine „inorder" Traversierung die erhaltene Struktur der Ablage im binären Suchbaum für beide Fälle 1 und 2 auf. Bestimmen Sie die Höhe im binären Suchbaum für beide Fälle.

Vorgabe: Es steht für die folgenden Aufgaben der Datensatz "KHR95_red.txt" bereit, der eine Liste von Krankenhausadressen mit der Struktur:
<Name><TAB><Straße HausNr><TAB><PLZ><TAB><Ort><TAB><Bettenzahl><CR,LF>
enthält.

Aufgabe 6.1 – Wertigkeit = 1
Realisieren Sie eine Lese- und Schreibmethode für diesen Datensatz und verwenden Sie eine geeignete Datenstruktur zur Repräsentation im Speicher. Zur Überprüfung Ihrer Lese- und Schreibmethode erstellen Sie eine formatierte Ausgabe für einzelne Records bzw. des gesamten Datensatzes.

Aufgabe 6.2 – Wertigkeit = 2
Implementieren Sie zwei der vorgestellten Sortierverfahren und zeigen Sie an Hand des obigen Datensatzes die Funktion. Führen Sie eine Sortierung nach Postleitzahlen und nach Orten durch, berücksichtigen Sie dabei ggf. Sortierungen in weiteren Spalten (z.B. nach Ort und dann nach PLZ). Ermitteln Sie den Aufwand für jedes Ihrer Sortierverfahren durch einen im Programm verankerten Zähler.

Hinweis: Testen Sie Ihre implementierten Verfahren zunächst mit einen kleinen Datensatz, bevor Sie mit dem größeren Datensatz "KHR95_red.txt" arbeiten.

Aufgabe 7.1 – Wertigkeit = 1
Ausgehend von den unsortierten oben genannten Datensätzen (zumindest hinsichtlich der PLZ) ist der mittlere Aufwand für die sequentielle Suche durch 1000 Stichproben für PLZ aus dem Wertebereich 1000-99999 getrennt für die erfolgreiche und erfolglose Suche zu ermitteln.

Aufgabe 7.2 – Wertigkeit = 2
Sortieren Sie die obigen Datensätze entsprechend der PLZ in aufsteigender Reihenfolge*. Ermitteln Sie den mittleren Aufwand für die Verfahren “binäre Suche” und “Interpolationssuche” durch 1000 Stichproben von PLZ im Wertebereich von 1000-99999 und vergleichen Sie diese mit dem Ergebnis von Aufgabe 7.1. Bestimmen Sie den Aufwand für die erfolgreiche und erfolglose Suche getrennt.

*Falls Sie die Sortierung gemäß Aufgabe 7.2 nicht vorliegen haben, so können Sie auch mittels MS Excel sortieren und entsprechende Datensätze erzeugen.

Aufgabe 8.1 – Wertigkeit = 2
Ermitteln Sie die Häufigkeit für jedes Zeichen (ignorieren Sie Groß- und Kleinschreibung, indem Sie im Vorfeld alle kleinen Buchstaben in große Buchstaben wandeln), das in der obigen Datei vorkommt und geben Sie das Ergebnis in einer Liste (ggf. auch Histogramm) aus. Ermitteln Sie entsprechend der Häufigkeit die Huffmann Kodierung für diesen Text und führen sie die Umsetzung durch. Welche Ersparnis erhalten Sie? Wie sieht das Ergebnis aus, wenn Sie auch kleine Buchstaben zulassen und damit den Wertebereich, der zu kodieren ist, erweitern.