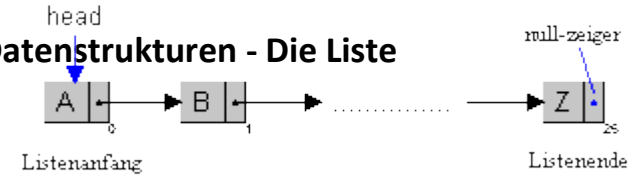
Dynamische (lineare) Datenstrukturen - Die Liste

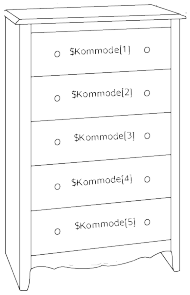


Rückblick

Datenstrukturen sind – genauso wie Algorithmen – ein fundamentaler Bestandteil der Informatik! Für unterschiedliche Anwendungszwecke sind stets auch verschiedene Datenstrukturen die jeweils geeigneten. Zu aller erst seid ihr im Laufe Eurer informatischen Laufbahn mit einer sehr grundlegenden Datenstruktur konfrontiert worden – den ***Variablen***: in diesen werden Werte gespeichert, die während der Ausführung von Algorithmen anfallen. Sie haben stets einen bestimmten Typ sowie einen speziellen Namen, über den sie angesprochen werden können (z.B. soll eine Variable „i“ die Anzahl der Durchläufe in einer Schleife zählen und wird wie folgt deklariert: var i: integer;).

Was aber ist, wenn riesige Datenmengen anfallen oder die zu verarbeitenden Werte keine simplen Zahlen sind, sondern komplizierte Dinge der realen Welt beschreiben?

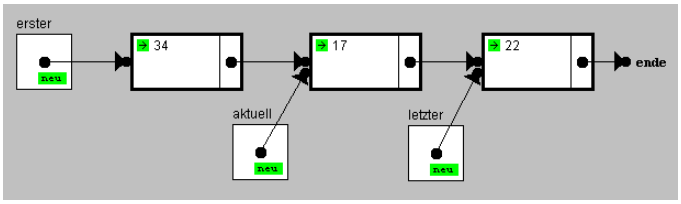
Für jeden Zweck gibt es wie gesagt eine adäquate Datenstruktur. Die Wahl einer geeigneten Datenstruktur ist im Allgemeinen eine wesentliche Entscheidung bei der Verarbeitung von Daten durch Programme. Für dieselben Daten benötigen manche Datenstrukturen mehr oder weniger Platz als andere, und für die gleichen Operationen auf den Daten führen manche Datenstrukturen zu effizienteren Algorithmen als andere.



Ihr habt z.B. bei der Untersuchung von Sortieralgorithmen schon komplexere Datenstrukturen kennen gelernt, und zwar so genannte ***„Arrays“*** – bei diesen wird eine ***feste*** Anzahl von Daten eines gemeinsamen Typs zu einem neuen Typ zusammengefasst und über einen ***Index*** kann auf die einzelnen Komponenten zugegriffen werden. Arrays können aber auch Nachteile haben - vergleiche das zugehörige Arbeitsblatt.

Die Datenstruktur „Liste“

graphische Veranschaulichung einer Liste



Nun werden wir im Verlauf der nächsten Wochen eine komplett neue, ebenfalls komplexere Datenstruktur kennenlernen – eine so genannte (informatische) **“Liste“** – inkl. ihrer Spezialfälle „Stack“ und „Queue“ – die gegenüber einem Array gewisse Vorteile bietet!

Sie zeichnet sich vor allem durch ihre **Dynamik** aus und wird insbesondere dann benötigt, wenn Objekte während der Laufzeit eines Programms erzeugt und einem vorhandenen Datensatz hinzugefügt oder mit ihm verkettet werden sollen. Ein solches Szenario entsteht beispielsweise, wenn während der Programmausführung vom Benutzer Daten eingegeben oder gelöscht werden sollen, die in ihrer Anzahl dynamisch wachsen oder schrumpfen können.

Vergleich

Beim Array beschränkt sich die Anzahl der Komponenten auf eine fest vorgegebene Zahl (und alle Komponenten müssen von gleicher Art sein). Bei dynamischen Strukturen liegt die Beschränkung der Größe nur im vorhandenen Speicherplatz.

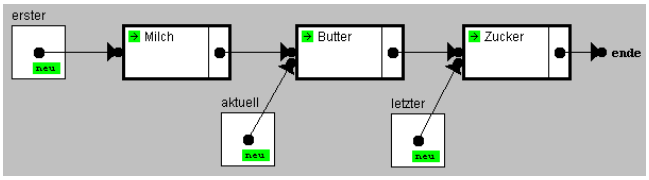
Charakteristika einer dynamischen Liste:

- **die Anzahl der Listenelemente ist nicht beschränkt oder vorher festgelegt**
- **die Listenelemente können beliebig an jeder Stelle der Liste eingefügt oder gelöscht werden.**

Damit aber alles formal festgelegt ist und jeder weiß, wovon jemand anders genau spricht, wenn er in der Informatik das Wort „Liste“ in den Mund nimmt, sind **Definitionen** erforderlich!

Definition (Listenelement)

Einen Eintrag in einer Liste bezeichnet man als **Listenelement**. Ein solches Listenelement besteht aus einem **Objekt/ Inhalt** (z.B. einem Text) und einem **Zeiger** auf ein anderes Listenelement oder – am Ende der Liste – auf **NULL**.



Definition (Liste)

Eine Liste besteht aus einer beliebigen Anzahl an **Listenelementen** und drei Zeigern, die hier (erstmal) wie folgt (selbsterklärend) genannt werden: „**erster**“, „**aktuell**“ und „**letzter**“.

Vergleich

Ein wichtiger Unterschied (und Nachteil) im Vergleich mit Arrays ist, dass kein referentieller Zugriff auf jedes einzelne Listenelement möglich ist. Man kann lediglich den Zeigern folgen, um ein spezielles Element anzusprechen.

Operationen auf Listen – Suchen, Einfügen und Löschen von Elementen (u.a.)

Die verschiedenen, auf Listen möglichen Operationen werden wir uns im Laufe der nächsten Zeit Schritt für Schritt erarbeiten! Wichtig ist allerdings bereits an dieser Stelle zu erwähnen, dass bei jeder durchgeführten Operation darauf zu achten ist, dass die **Struktur der Liste nicht zerstört** wird: *d.h. der Anfangs-Zeiger muss immer auf das erste Element zeigen, der Aktuell-Zeiger auf das aktuell betrachtete Element, der End-Zeiger auf das letzte Element und die Verkettung darf zu keiner Zeit auseinanderfallen*!

Vergleich - Listen im Alltag

Definition:

Eine Liste ist eine thematische Sammlung bestimmter Objekte und ihrer Anordnung nach einem Prinzip, beispielsweise alphabetisch oder chronologisch.

Beispiele findet man dabei zu Hauf – Einkaufslisten, Schülerlisten in einem Kursbuch, das Telefonbuch in Eurem Handy oder der Posteingang in Eurem E-Mail-Account (usw.).

Bei genauerer Betrachtung fallen hier allerdings ebenfalls verschieden Listentypen auf, die im Folgenden beispielhaft verglichen werden:

	E-Mail-Posteingang	**Schülerliste im Klassenbuch**
Anzahl	unbegrenzt	begrenzt
Hinzufügen	Sortierung bleibt stets erhalten	Sortierung ggf. nicht mehr gegeben, wenn neuer Schüler hinzu kommt
Löschen	Listenplatz wird frei geräumt	es entsteht eine Lücke
ALLGEMEIN	***DYNAMISCH***	***STATISCH***

Fazit: Der E-Mail-Posteingang hat eine dynamische Gestalt im Gegensatz zur statischen der Schülerliste und biete viele Vorteile, die hauptsächlich der *Flexibilität* zuzuordnen sind.