Mr. T. hat wiedermal die Infostunde überzogen, jetzt stehen Sie nach Essen in einer ...

Abstrakte Speicherkonzepte

b) die Schlange / Queue

Eine Queue (Warteschlange) ist eine Datenstruktur, die nach FIFO-Prinzip (**First In, First Out**) arbeitet. Das erste Element, das einer Queue hinzugefügt wird, wird auch zuerst wieder entfernt. Operationen[1] auf Schlangen sind wie folgt spezifiziert:

- **`enQueue(Element)`**: ein Element (hinten) an die Schlange anhängen.
- **`deQueue()`**: das erste Element der Schlange entfernen.
- **`isEmpty()`**: Prüffunktion, ob die Schlange leer ist.

Offenbar lassen sich Schlangen einfach auf Python-Listen nachbilden, wenn man den **Zugriff** auf beliebige Elemente innerhalb der Liste **einschränkt,** aber

ACHTUNG: **Wir kennen keine Listen und haben NIEMALS von der Existenz der Module Collections oder Queue gehört. Wir verfolgen einen eigenen Ansatz!**

Aber welchen?
Oo-Sicht: Elemente sind Objekte mit Attributen, Werte kennzeichnen Zustand.

---

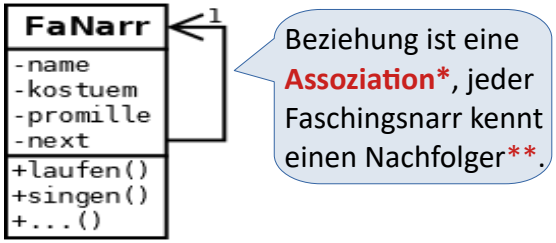
1 Manchmal wird wie beim Stack das Hinzufügen von Daten als push(Element), das Entfernen auch als pop() bezeichnet. Dann muss aber klar sein, dass es sich um eine Queue handelt!

Die Polonaise ist eine Schlange tanzender Faschingsnarren, bei der jeder seine Hand auf der Schulter des Vordermanns hat, also auf ihn verweist.

=> Ergänze zu den Attributen eines Objektes ein **Verweisattribut**

(es hat sich eingebürgert, nach hinten zu verweisen)

```
class FaNarr:
    def __init__(self,n,k,p,f:FaNarr):
        self.__name = n
        self.__kostuem = k
        self.__promille = p
        self.__next = f
```



Beachte, dass die Referenzvariable `self.next` Teil des Objektes ist (also auch im Heap liegt).

* Denkbar wäre auch **Aggregation**, zu Beziehungstypen vertiefend siehe HIER (Selbststudium!).

** *? Wirklich jeder ? ? Immer ?*

Hilfreich: Betrachte die Struktur rekursiv und setze den passenden Trivialfall.

A1: Analysieren Sie die Implementation in **H:\Ausgeteilt\Queueklasse.py**
Zeichnen Sie die Zeiger nach jedem Kommando der deQueue Methode.

Anwendung:

Es ist Montagmorgen 7:20:00 Uhr, im Lehrerzimmer herrscht Stress für den Kopierer mit PC-Anbindung. Der bekommt Druck-/Kopieraufträge, die alle noch zum Unterrichtsbeginn gebraucht werden. Die Situation ist in **`mondaymorning.csv`** dargestellt.

A2: Entwickeln Sie ein Programm, dass unter Ausnutzung der Queueklasse die Druckerwarteschlange des Gerätes simuliert. Auf der Konsole ist auszugeben, welcher Druckauftrag mit welchem Zeitstempel gerade angefangen / fertiggestellt wurde. Beachten Sie folgende Festlegungen:

- Der Druck einer Seite Simplex dauert 2 sek., eine Duplexseite dauert 5 sek.
- Seiten eines Dokuments werden ohne Wartezeit fortlaufend gedruckt, zwischen verschiedenen Dokumenten braucht der Drucker 3 sek. zum Einlesen und Puffern[2] der Druckdaten.
- Das Scannen einer Kopie kann nicht parallel zum Druck erfolgen, es unterbricht laufende Drucke für 20 sek.
- Nach 160 Druckseiten ist der Papiervorrat aufgebraucht. Nachfüllen dauert 10 sek.

Tipps: Erstellen Sie eine Klasse Druckauftrag und geben Sie deren Objekte als data in die Queue.
Simulation importiert Druckauftrag und Queue.
Zeitstempelei (geht auch ohne time-Modul) in Druckauftrag oder Simulation. Entscheiden Sie!
BUT DON'T fummel an der Warteschlange rum! Benutzen Sie deren Methoden!

2 Der Puffer ist übrigens auch ein FIFO-Speicher, der hardwareseitig implementiert ist und hier nicht berücksichtigt werden muss.