Data Structures and Algorithms (2)

Instructor: Ming Zhang

Textbook Authors: Ming Zhang, Tengjiao Wang and Haiyan Zhao

Higher Education Press, 2008.6 (the "Eleventh Five-Year" national planning textbook)

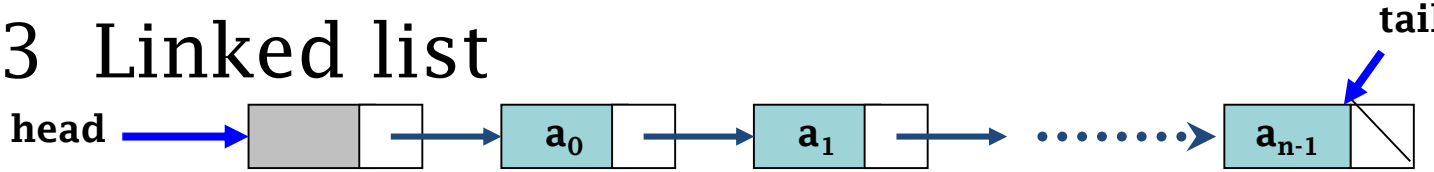
https://courses.edx.org/courses/PekingX/04830050x/2T2014/

Linear List

Chapter II Linear Lists

- 2.1 Linear list $\{a_0, a_1, \ldots, a_{n-1}\}$
- 2.2 Sequential list

- 2.3 Linked list



- 2.4 Comparison of sequential list and linked list

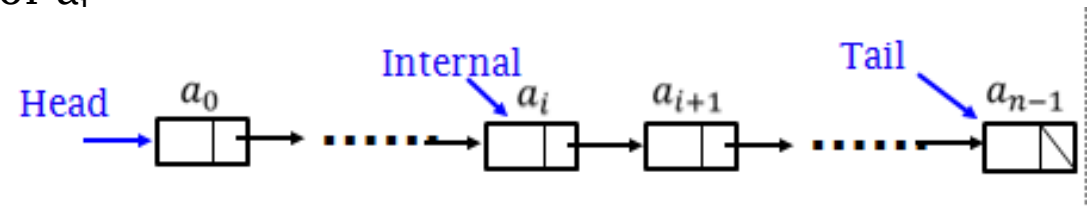
The Concepts of Linear List

- List for short, is a finite sequence of zero or more elements, usually represented as k_0 , k_1, … , k_{n-1} ($n \geq 1$)
  - Entries: elements of linear list (can contain multiple data items, records)
  - Index: i is called the "Index" of entry ki
  - Length of the list: the number of elements contained in the list n
  - Empty list: a linear list with the length of zero (n = 0)
- Features of Linear list:
  - Flexible operations
  - Dynamically changed length

Linear structure

- Tuple $B = (K, R)$ $K = \{a_0, a_1, \ldots, a_{n-1}\}$ $R = \{r\}$
  - There is one and only one starting point that has no previous node and has only one successive node.
  - There is one and only one ending point that has only one previous node and has no successive node.
  - The other nodes are called internal nodes that have only one previous node and also have only one successive node.

<a_i,a_{i+1}> a_i is previous node of a_{i+1} , and a_{i+1} is the successive node of a_i



Linear structure

- Features

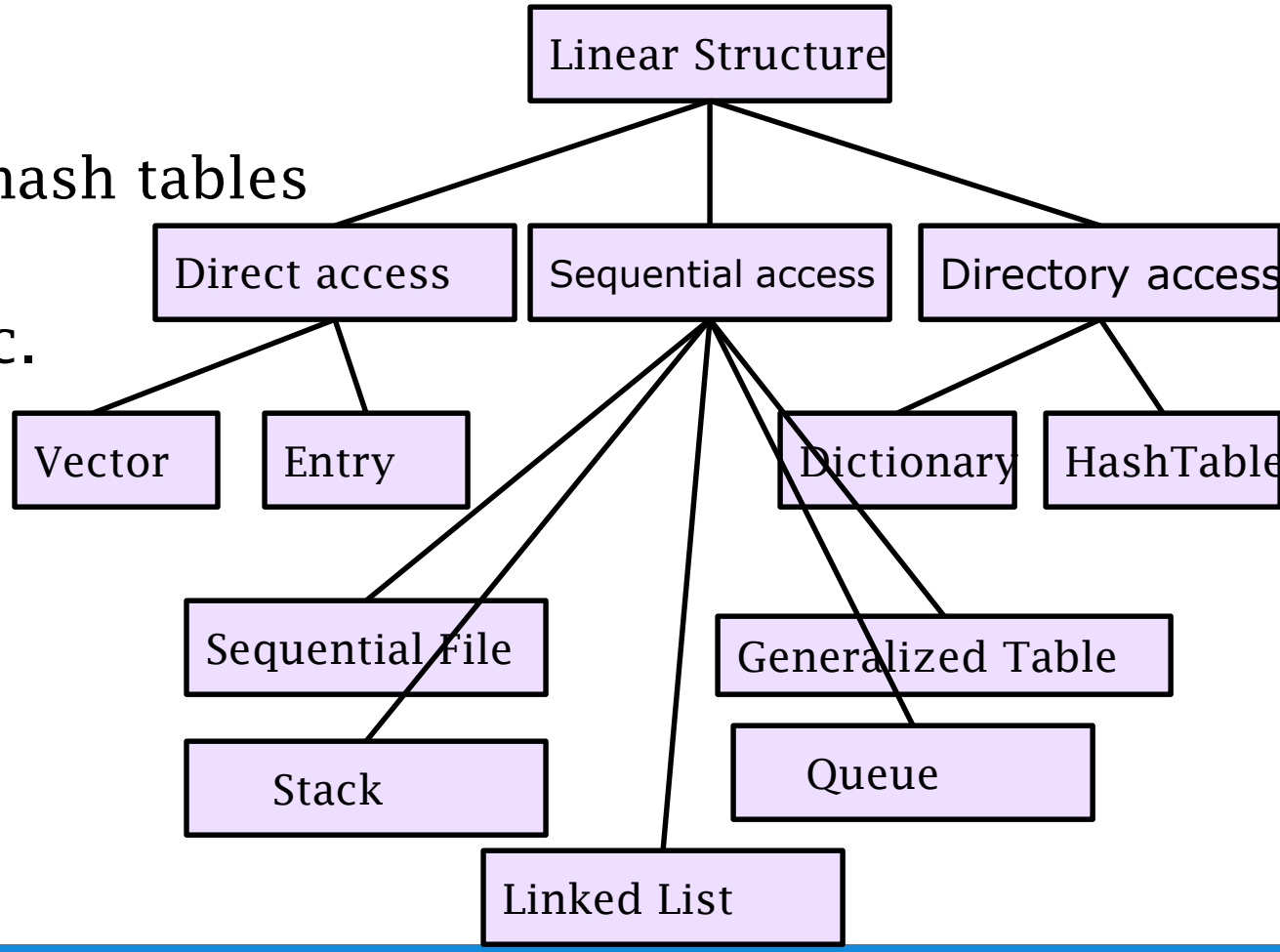
✓ Uniformity: Although the data elements of different linear lists may be diverse, but the data elements of the same linear list normally have the same data type and length

✓ Orderliness: each data element has its own position in the list and their relative positions are linear

Classification

Linear structure

- According to the complexity
  - Simple: Linear lists, stacks, queues, hash tables
  - Advanced: generalized lists, multidimensional arrays, files etc.
- Divided by access ways
  - Direct access type
  - Sequential access type
  - Contents Index type (directory access)



Linear structure

- **Classified by operation (see later)**

-Linear List

- All entries are nodes of the same type of linear lists
- No need to limit the form of operation
- Divided into: the sequential list, linked list depending on the difference of storage

-Stack (LIFO, Last In First Out)

Insert and delete operations are restricted to the same end of the list

-Queue (FIFO, First In First Out)

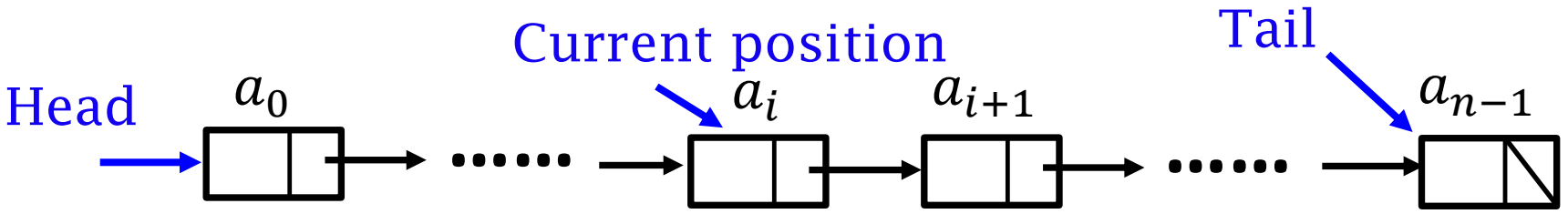
Insert at one end of the list, while delete at the other end

2.1 Linear List

- Three aspects
  - Logical structure of the linear list
  - Storage structure
  - Operation of linear list

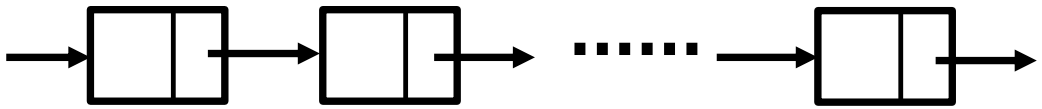
Logical structure of the linear list

- The main properties
  - Length
  - Head
  - Tail
  - Current position



Classification (By storage)

- Linear List
  - All entries are nodes of the same type of linear lists
  - No need to limit the form of operation
  - Divided into: the sequential list, linked list depending on the difference of storage

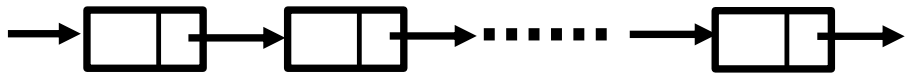


Storage Structures

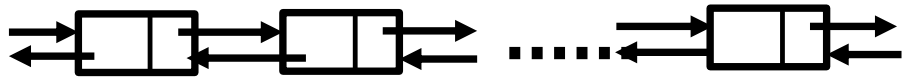
- Sequential list

  - Store according to index values from small to large in an adjacent continuous region
  - Compact structure, and the storage density is 1

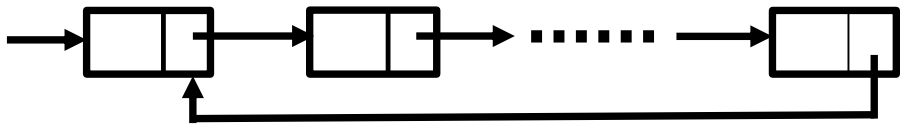
- Linked list
  - Single list



  - Double linked list



  - Circular list

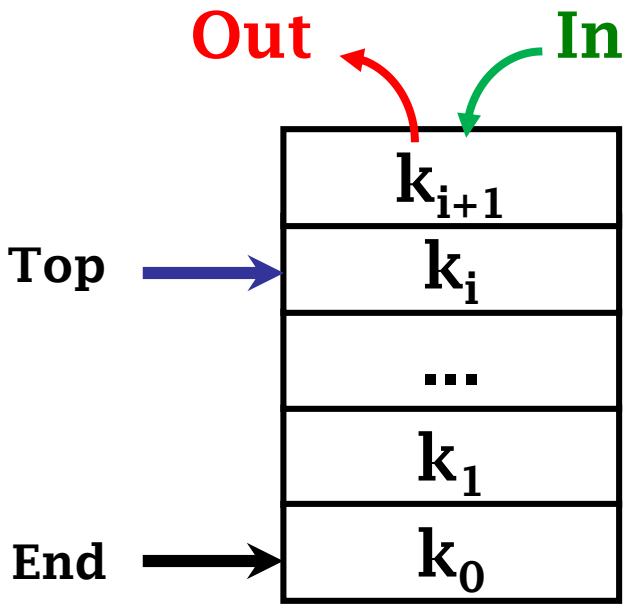


Classification (By operation)

- Linear List
  - No need to limit the form of operation
- Stack
  - At the same end
- Queue
  - At both ends

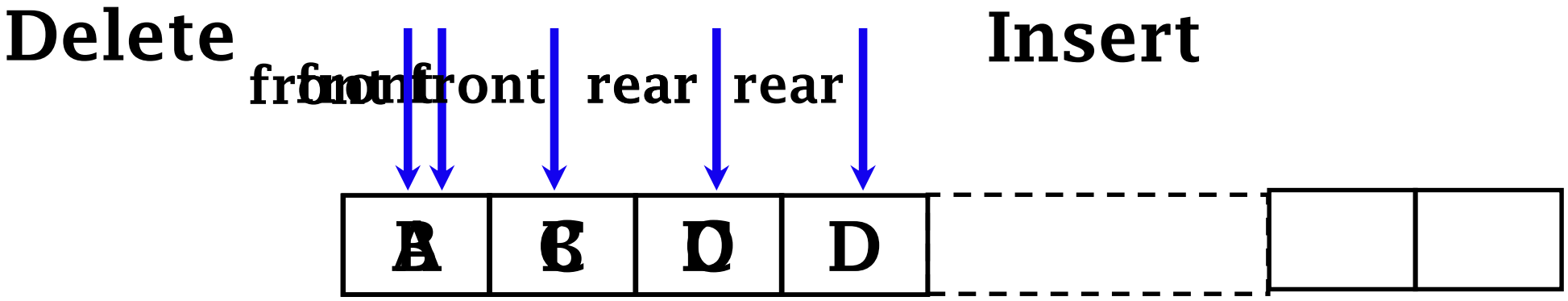
Classification (By operation)

- Stack (LIFO, Last In First Out)
  - Insert and delete operations are restricted to the same end of the list



Classification (By operation)

- Queue (FIFO, First In First Out)
  - Insert at one end of the list while delete at the other end
- Rear(true pointer)



Operation on linear Lists

- Construct a linear list
- Destruct the linear list
- Insert a new element
- Delete a specific element
- Modify a specific element
- Sort
- Search
- …

Class Template of Linear lists

```
template <class T> class List {
    void clear();          // clear the linear list
    bool isEmpty();        // When it is empty, returns true
    bool append(const T value);
                           // insert the value  at the end , length adds by 1
    bool insert(const int p, const T value);
                           // insert the value at position P , length adds  by 1
    bool delete(const int p);
                           // delete the value at position p , length decreases by 1
    bool getPos(int& p, const T value);
                           // find the value and returns its position
    bool getValue(const int p, T& value);
                           // return the element's value at  position P
                           //and assign it to the variable of value
    bool setValue(const int p, const T value);
                           // set value for position P
};
```

Thinking

- What kind of classification are there for the linear list?

- In all kinds of names of linear lists , which are related to storage structures? Which are related to operations?

Data Structures and Algorithms

Thanks

the National Elaborate Course (Only available for IPs in China)
http://www.jpk.pku.edu.cn/pkujpk/course/sjjg/

Ming Zhang, Tengjiao Wang and Haiyan Zhao
Higher Education Press, 2008.6 (awarded as the "Eleventh Five-Year" national planning textbook)