What is a collection?

- From mathematics we have a **set** of entities
- What are the **properties** of a set?
  - N elements (entities)
  - Each element is **unique** (no duplicates)
  - **Unordered**
- What are the **operations** on a set?
  - **Cardinality** the number of elements in the set (size)
  - **Is_empty** cardinality == 0
  - **Add** an element
  - **Remove** an element
  - **Find** an element (member)
  - **Union** of 2 sets
  - **Intersection** of 2 sets
  - **Difference** of 2 sets

ADTs & Collections

ADT	relationships	implementation
set	**none** (non-ordered) unique elements	**<u>sequence</u>**
sequence	**successor;** ordered –position; ?sorted?	array / structure & pointer
tree GT➔BT➔ BST ➔AVL	**hierarchical** (non-)ordered, (LC RC), ?sorted?	array / **structure & pointer**
graph **G = (V, E)**	**general** (non-ordered)	array / structure & pointer adjacency list / matrix **<u>(seq)</u>**

Operations on collections
add, find, remove, size, is_empty, sort, search, navigate

ADTs: recursive definitions **(¤ = empty)**

- **Sequence** ::= Head Tail | ¤
- Head ::= Element
- Tail ::= **Sequence**

- **BT** ::= LC Node RC | ¤
- Node ::= Element
- LC, RC ::= **BT**

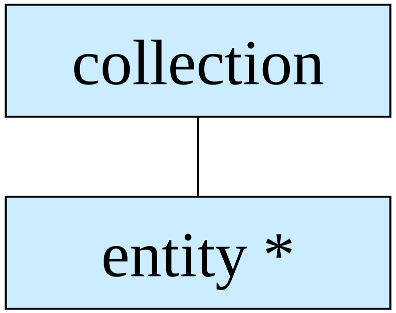
Tree & Graph

Tree
GT Non-ordered → children set Ordered → children sequence
GT → BT (+ order; max 2 children)
BT → BST (+ sorting)
BST → AVL (+ balancing)
B-Tree (completely balanced – bushy)

Graph G= (V, E)
V = set of vertices (nodes)
E = set of elements
Edge connects 2 vertices
directed $V_1 \rightarrow V_2$
undirected $V_1 \leftrightarrow V_2$

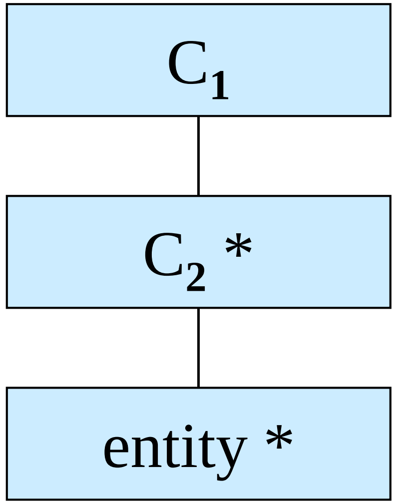
Levels of Abstraction

- **Collection of Entities**



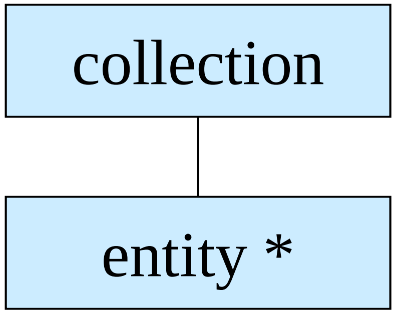
- **Operations**
  - on the collection, C
  - on the entity, E
  - on both C and E

- **An entity may be a collection!**



Collections: operations

- **Collection of Entities**

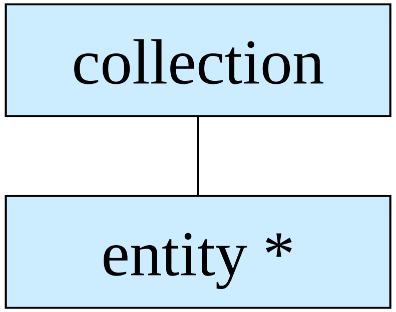


- **Operations**
  - on the collection, C
  - on the entity, E
  - on both C and E

- **Operations**
  - create_C : ¤ → C
  - destroy_C : C → ¤
  - display_C : C → C
  - is_empty : C → Bool
  - cardinality : C → int
  - create_E : ¤ → E
  - destroy_E : E → ¤
  - display_E : E → E
  - add : C x E → C'
  - remove : C x E → C'
  - find : C x E → Boolean

Collections: operations

- **Collection of Entities**



- **Operations**
  - on the collection, C
  - on the entity, E
  - on both C and E

- NB operations on C may use operations on E
  - display_C :
    for all E in C display_E;

- C may be displayed in a pre-defined order
  e.g. Trees: Pre-/in-/post-order

- => OO composition

Levels of Abstraction

Collection Implementation