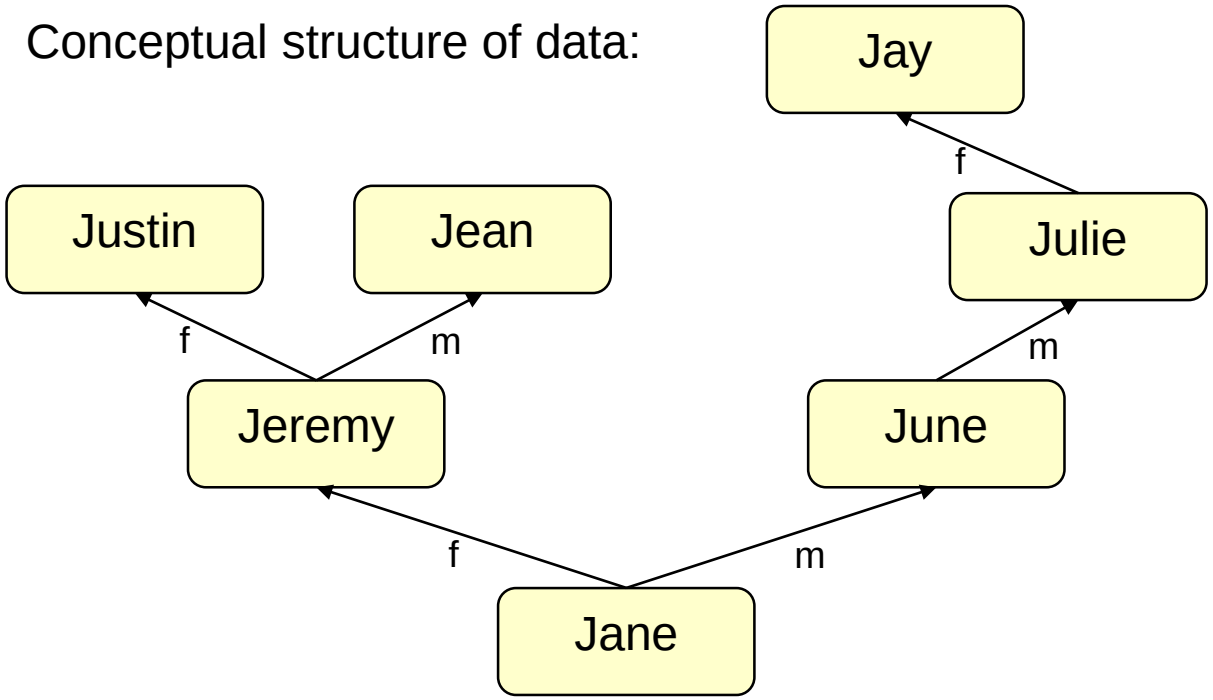
Trees

- Maps, Sets, Bags: collections with no structure
- Lists, Queues, Stacks, Deques: collections with linear structure (in order)

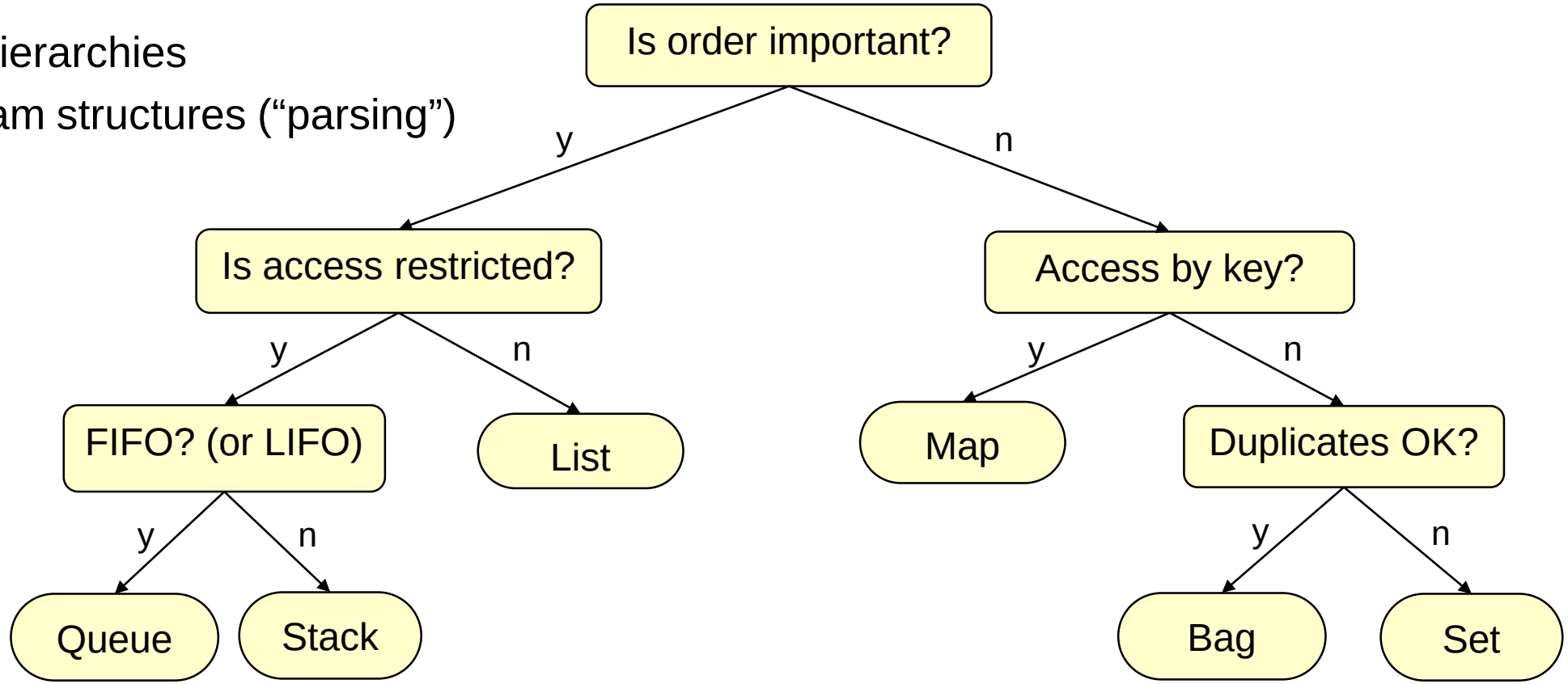
- Not all collections fit into those two structures.
- eg, genealogy data

Conceptual structure of data:



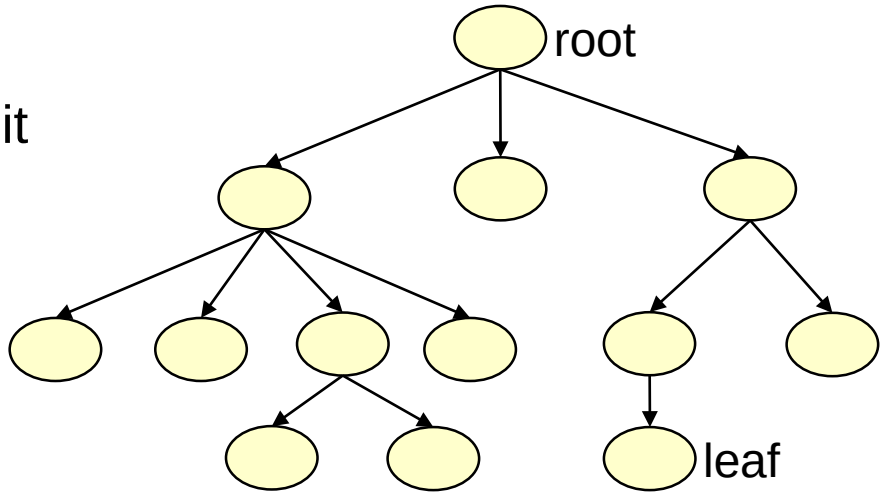
Tree Structured Data

- Examples:
  - genealogy
  - organisational hierarchies
  - language/program structures (“parsing”)
  - decision trees



Tree Notation:

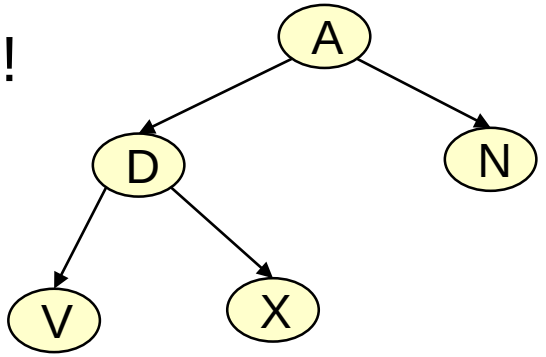
- Tree made of nodes with links
- Nodes linked to child nodes
  - might have a limit on number of children, or no limit
  - each node has one parent
- Root node is the base of the tree
  - root node has no parent
  - we typically draw it at the top!!
- Leaf nodes are nodes with no children
  - we typically draw them at the bottom!



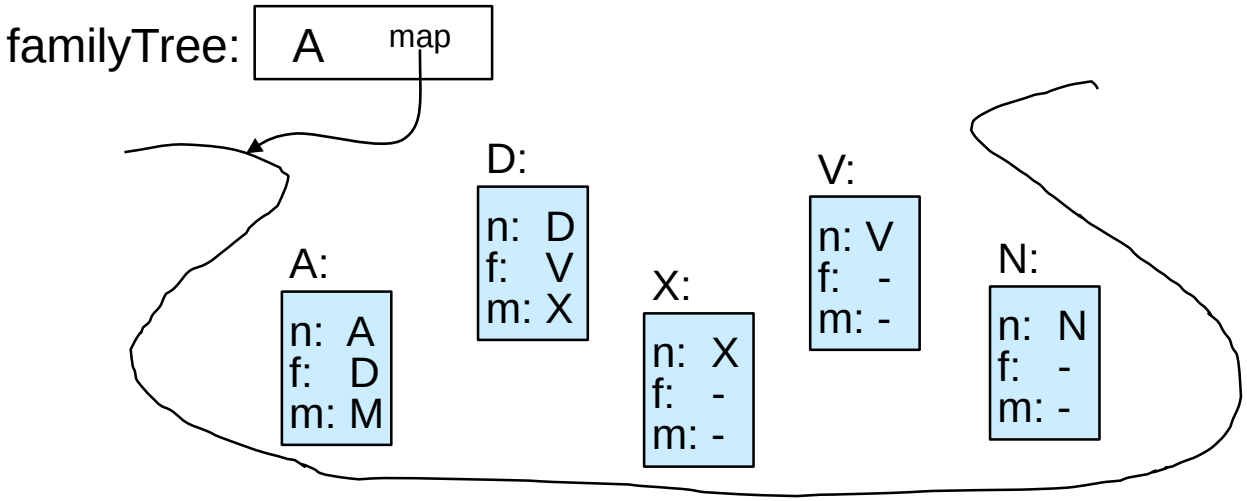
- What Data Structures support tree structured data?

Data Structures for Tree Structured data

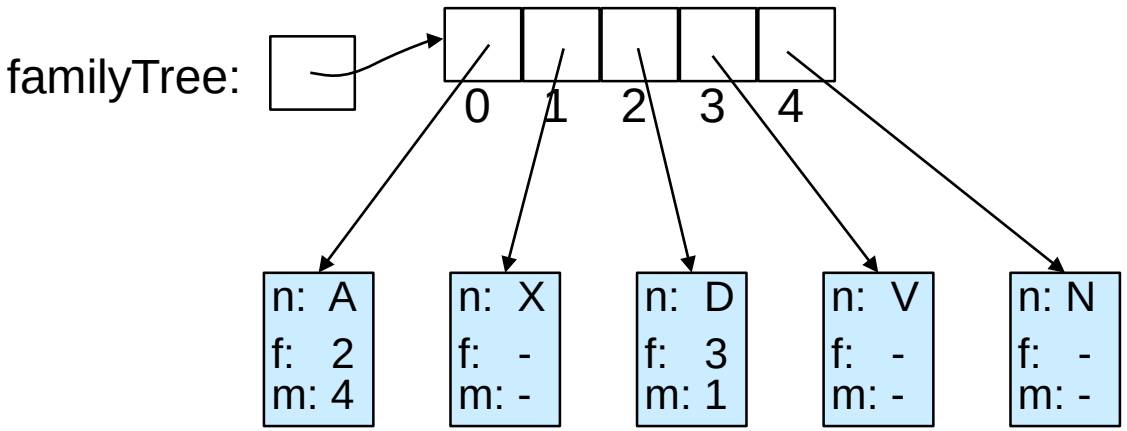
- Nothing new - you already have all the bits!



- Map:
  - key = name of item,
  - item contains data plus names of child nodes
  - need name of root node.



- List:
  - item contains data plus the index of child nodes
  - root at index 0

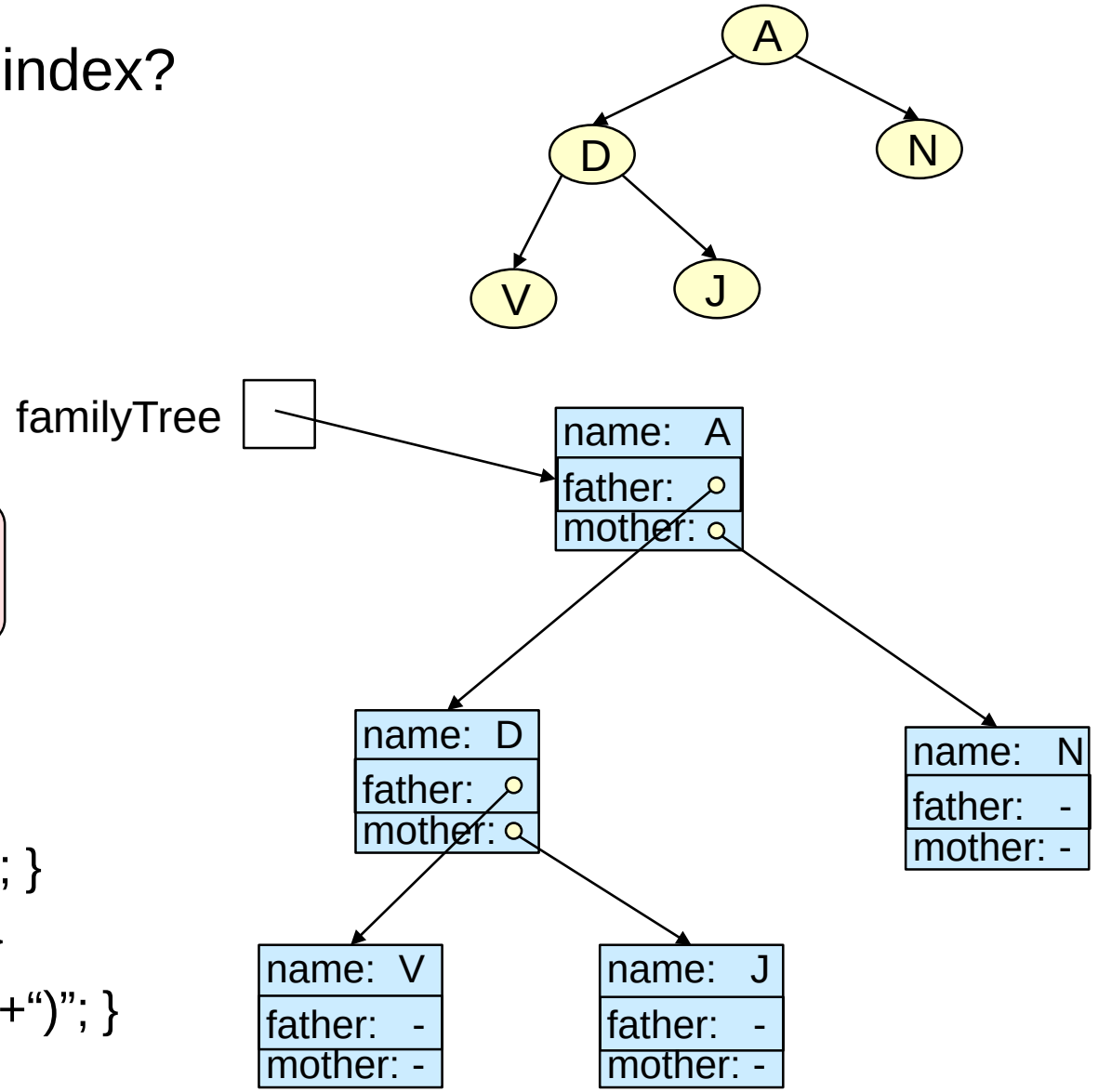


Data Structures for Tree Structured data

- But why do we have to go via a key or an index?
- "Linked Structure"

```
public class Person {
    private String name;
    private int dob;
    private Person father;
    private Person mother;
        :
    public Person getMother(){ return mother; }
    public Person getFather(){ return father; }
    public void setMother(Person p){ mother = p; }
    public void setFather(Person p){ father = p; }
    public void toString(){ return name +"("+dob+")"; }
}
```

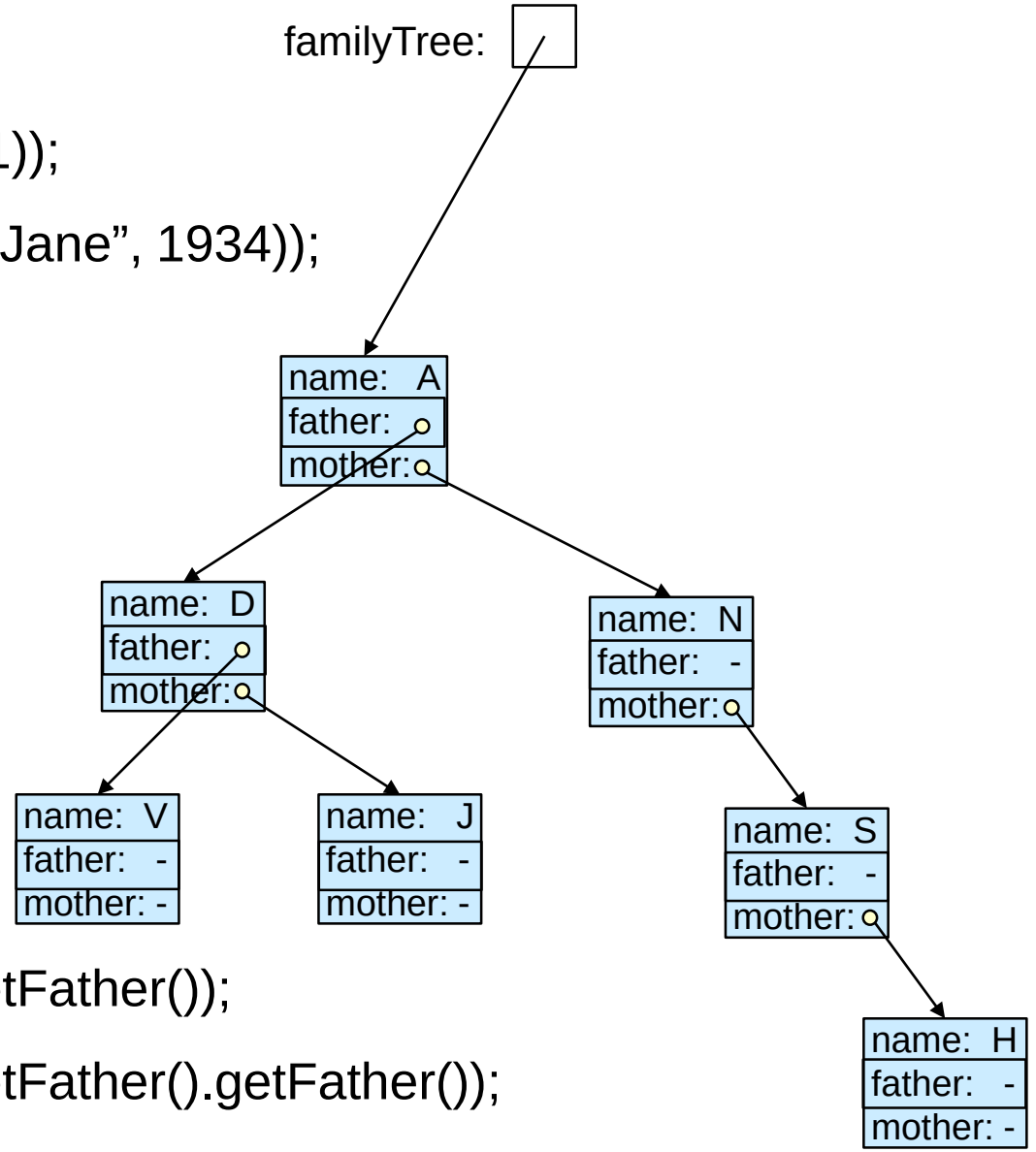
Recursive Data Structure!



Using “linked” tree structures

```
familyTree  = new Person(“Alice”, 2000);
familyTree.setFather(new Person(“David”, 1971));
familyTree.getFather().setMother(new Person(“Jane”, 1934));
```

```
UI.println(familyTree.getMother());
UI.println(familyTree.getFather().getMother());
```




```
UI.println(familyTree.getFather().getMother().getFather());
UI.println(familyTree.getFather().getMother().getFather().getFather());
```