Using "linked" structures: looping down tree

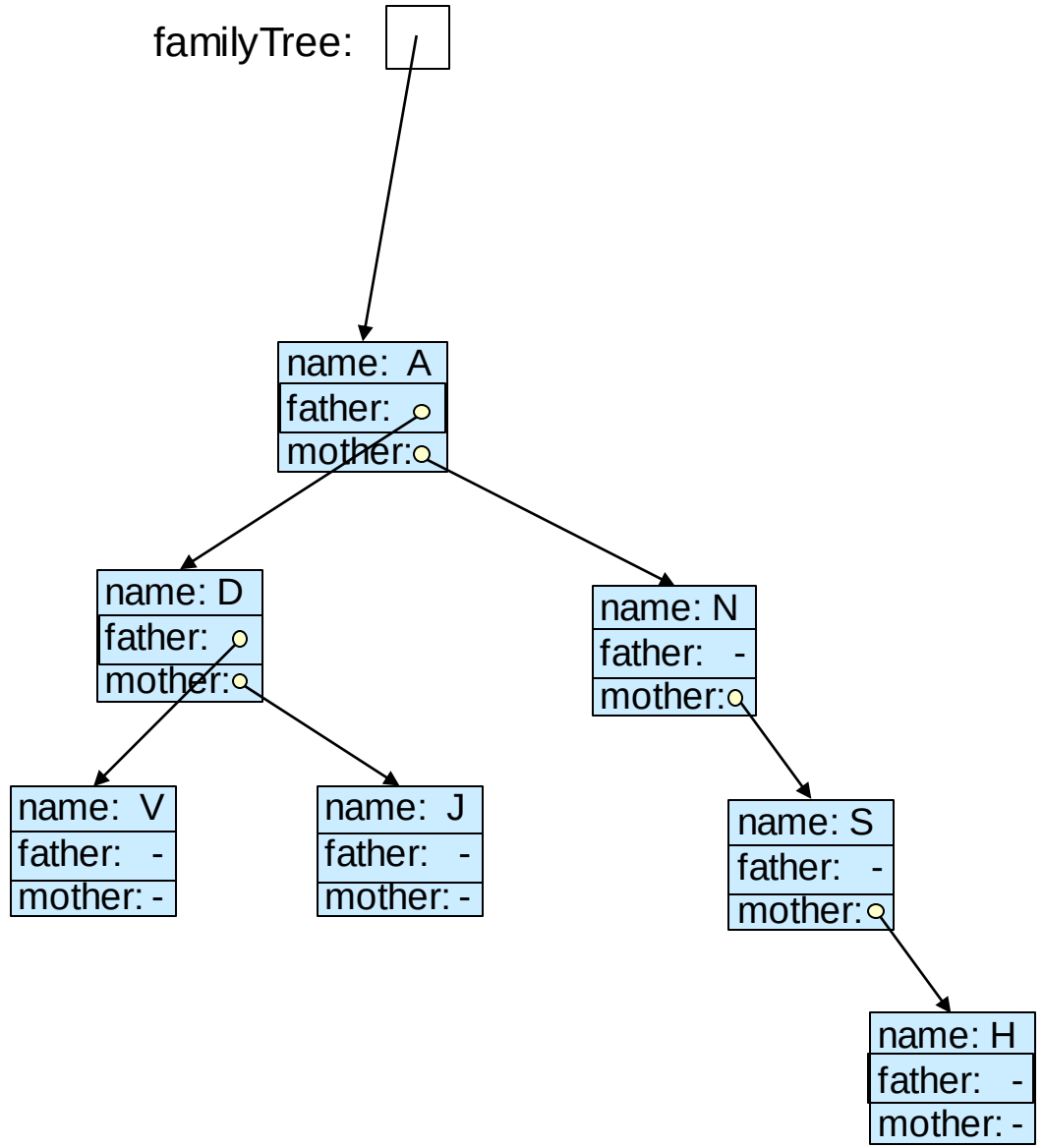
Stepping along a path from root.

eg: Print out maternal line:

```
Person p = familyTree;
while (p != null){
    UI.println(p);
    p = p.getMother();
}
```

p:

runs off the end



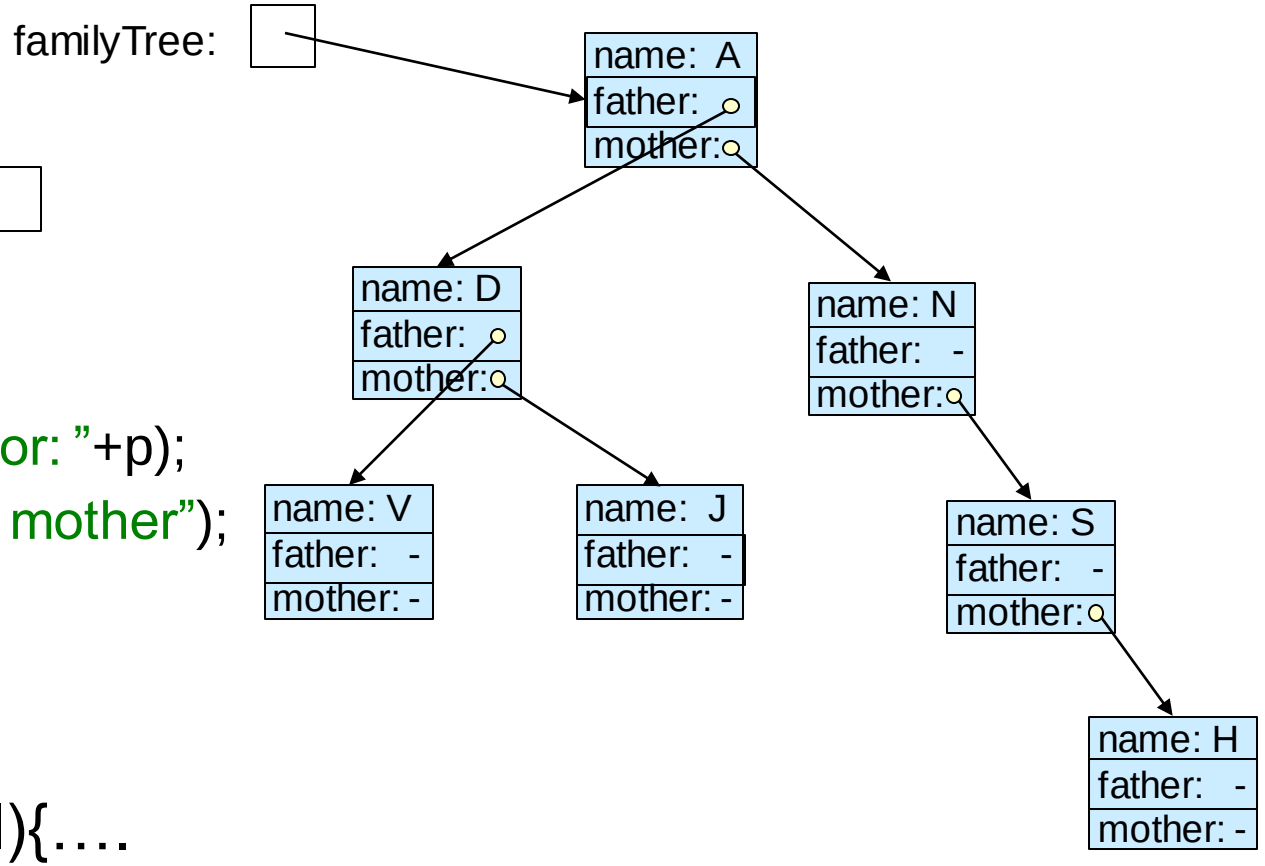
Using "linked" structures: looping down tree

Finding a leaf node:

eg: Add next maternal ancestor:

```
Person p = familyTree;
while (p.getMother() != null){
    p = p.getMother();
}
UI.println("Oldest known maternal ancestor: "+p);
String name = UI.askString("Name of her mother");
int dob = UI.askInt("year of birth");
p.setMother(new Person(name, dob));
```

Why?



Running off the end: **while** (p != null){….

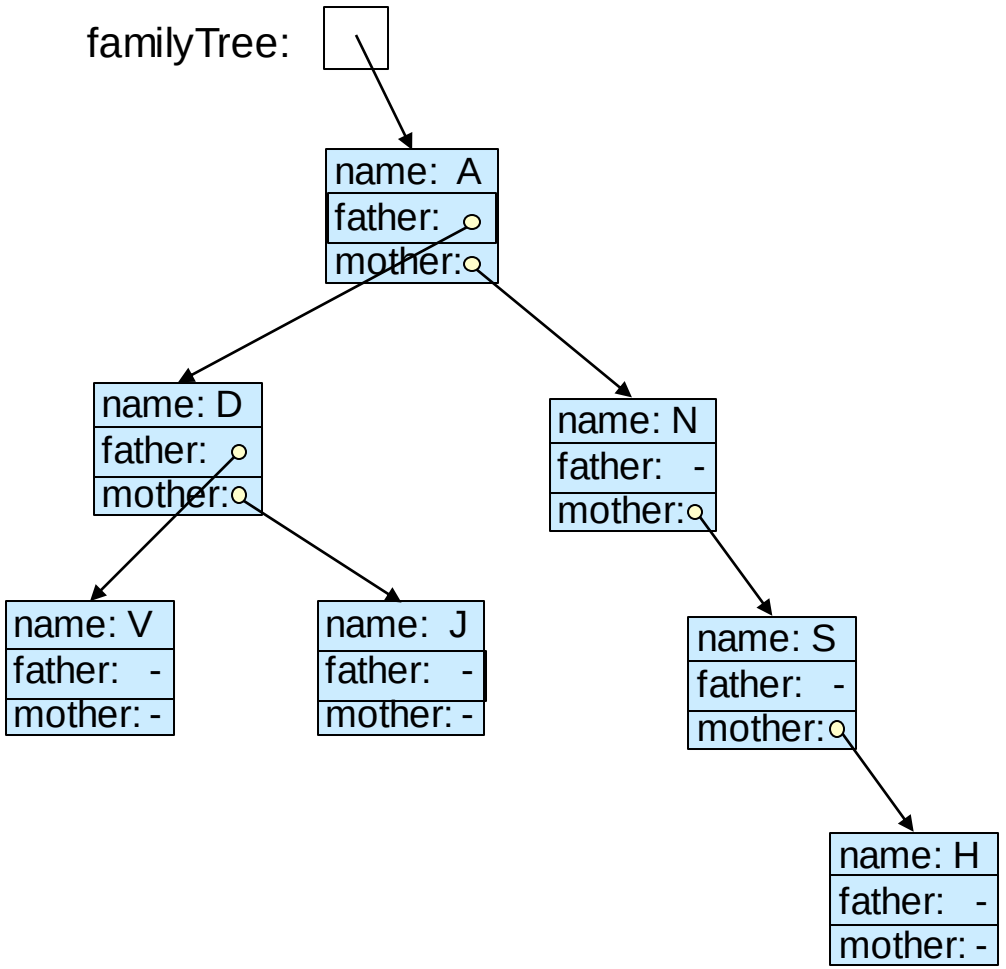
Stopping at the end: **while** (p.getMother() != null){….

Using “linked” tree structures:

Add next maternal ancestor:

```
public Person oldestMatAnc (Person p){
    Person tmp = p;
    while (tmp.getMother()!=null){
        tmp = tmp.getMother();
    }
    return tmp;
}
```

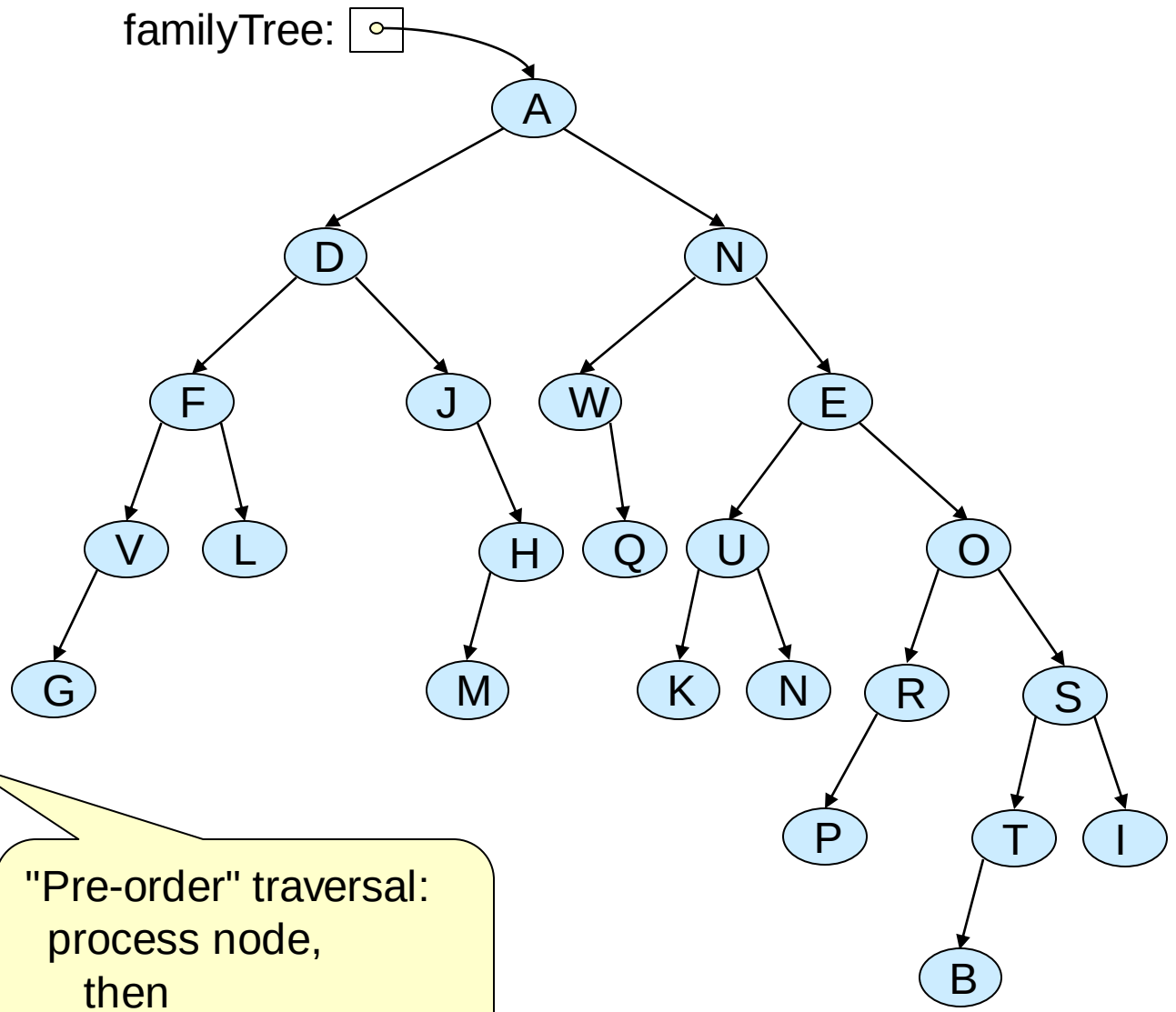
```
    :
Person p = oldestMatAnc(familyTree);
UI.println(“Oldest known maternal ancestor: ”+p);
String name = UI.askString(“Name of her mother”);
int dob = UI.askInt(“year of birth”);
p.setMother(new Person(name, dob));
```



Traversing a tree

- Traversing => processing every node
- Traversing is harder with a loop; much easier to use recursion.

```
public void printAll (Person p){
    if (p != null){
        UI.println(p);
        printAll(p.getFather());
        printAll(p.getMother());
    }
}

printAll(familyTree);
```



"Depth First" traversal:
process whole subtree
before next child

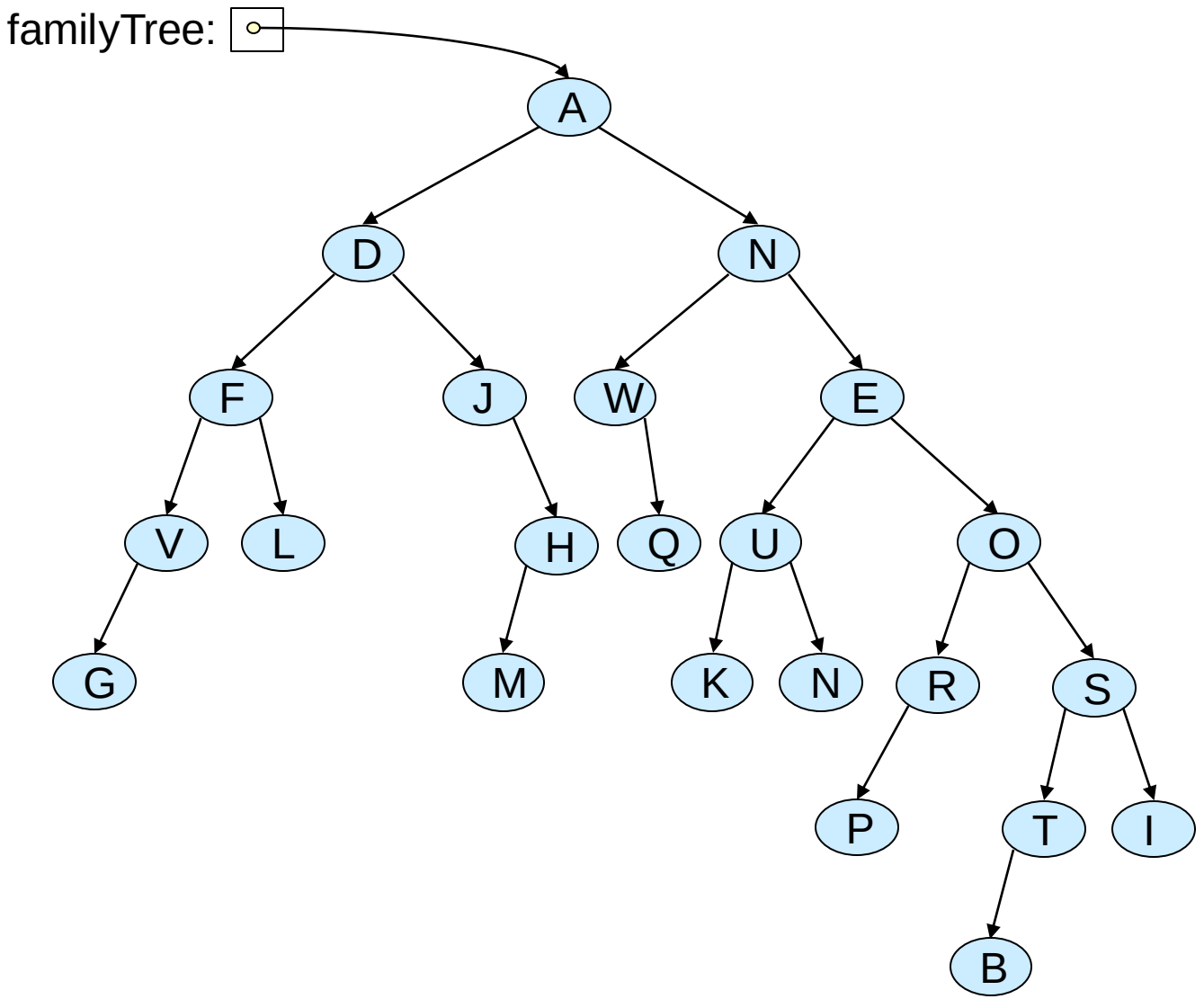
"Pre-order" traversal:
process node,
then
process subtrees.

More traversals: depth-first, post-order

- "Depth-first, Post-order" traversal:

process subtrees
then
process node:

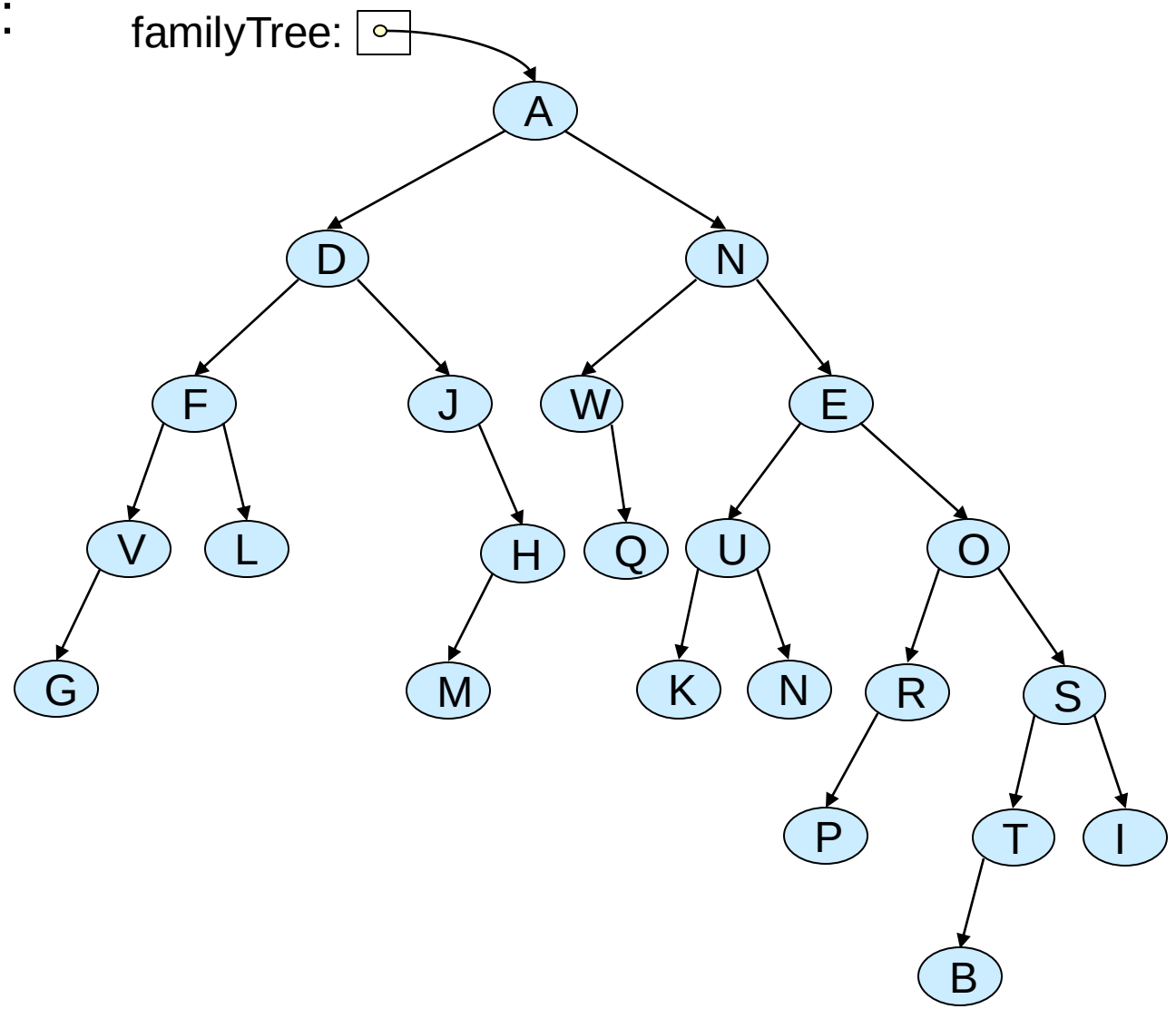
```
public void printAll (Person p){
    if (p!=null){
        printAll(p.getFather());
        printAll(p.getMother());
        UI.println(p);
    }
}

printAll(me);
```



Traversing with an extra parameter

- Traversing the tree, printing generation:

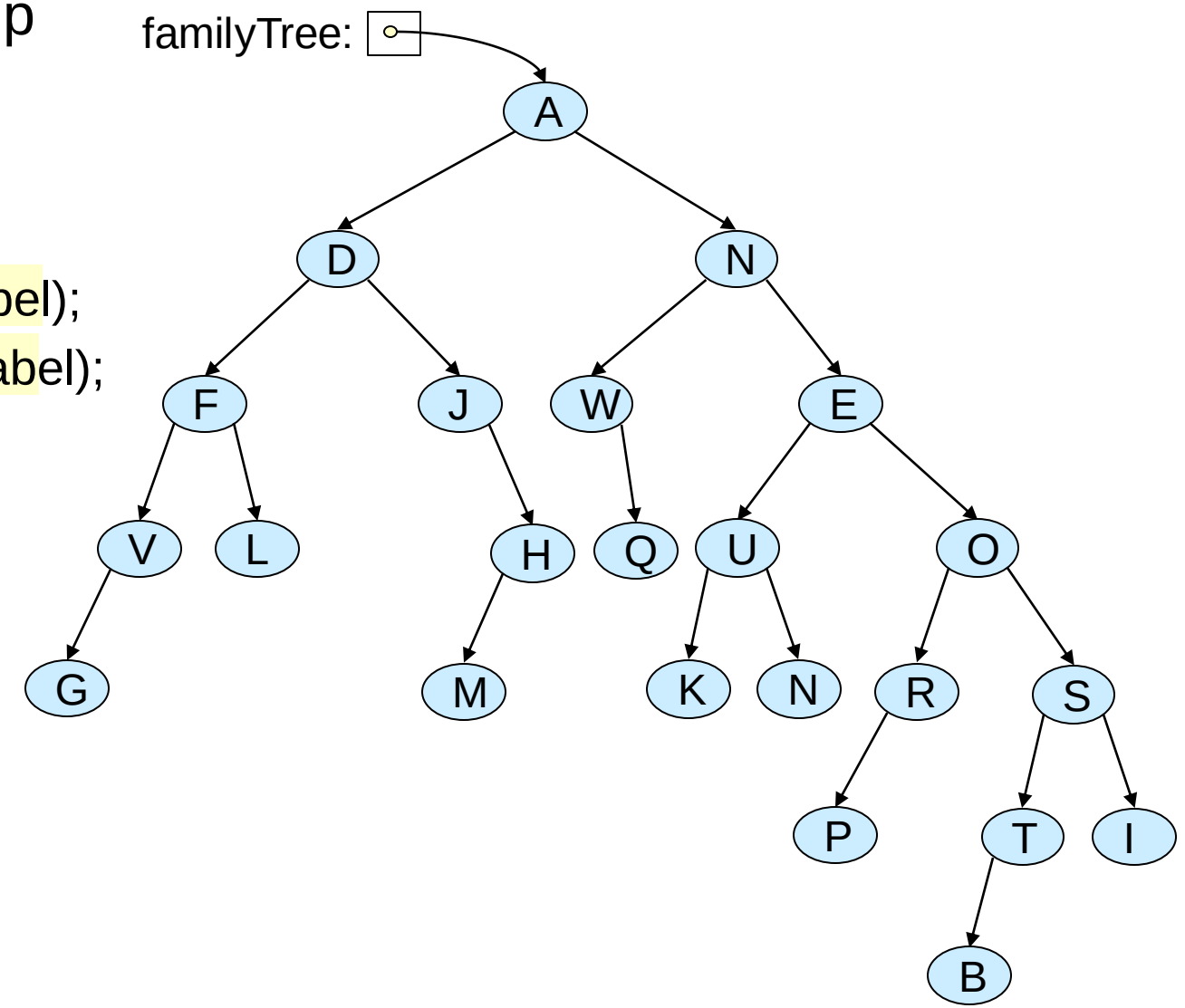
```
public void printAll (Person p, int gen){
    if (p!=null){
        UI.println(gen + ": " + p);
        printAll(p.getMother(), gen+1);
        printAll(p.getFather(), gen+1);
    }
}
printAll(familyTree, 1);
```



Traversing with an extra parameter

- Traversing the tree: printing relationship

```
public void printAll (Person p, String label){
    if (p!=null){
        UI.println(label + ": " + p);
        printAll(p.getFather(), "father of " + label);
        printAll(p.getMother(), "mother of "+ label);
    }
}
printAll(familyTree, "me");
```



Another traversal: depth-first, in-order

- Depth-first, in-order traversal

```
public void printAll (Person p){
    if (p!=null){
        printAll(p.getFather());
        UI.print(" < " + p + " > ");
        printAll(p.getMother());
    }
}


printAll(familyTree);
```