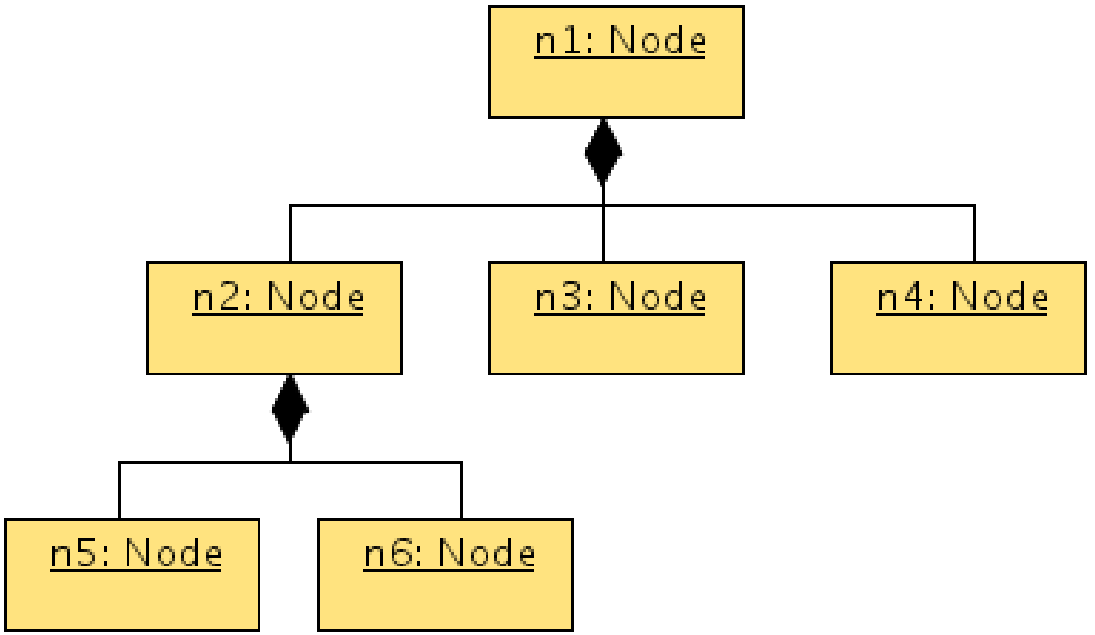
Övning 2: Iterator



Problemet

- Node-klassen innehåller en (ordnad) mängd barn-noder, som i sin tur kan innehålla barn-noder osv.
- Vi vill iterera igenom trädstrukturen som skapas av noderna
  - Behövs en ny subklass till java.util.Iterator

Filerna:

- Node.java – enkel trädstruktur
  - Node.createShallowNodeIterator() ger en iterator för barnen (men inte barnbarnen osv)
  - Node.createDFSNodeIterator() returnerar en NodeIterator som ger alla barnen (transitivt) depth-first
- NodeDFSIterator.java - oimplementerad fil
- NodeTest.java – enkelt testprogram som använder sig av NodeDFSIterator

JUnit

- Bibliotek för unit tests i Java
- Använder sig av s.k. annoteringar i test-filen
- Stöder skrivandet av test
  - Before/After-metoder
  - TestSuite
  - mm

Exempel på JUnit-fil

```
import java.util.Iterator;

import org.junit.*;
import static org.junit.Assert.*;

public class NodeTest {

	@Test
	public void test1() {
		Node n1 = new Node();
		Iterator<Node> i = n1.createDFSNodeIterator();
		assertEquals(i.hasNext(), true);
		assertEquals(i.hasNext(), true);
		assertEquals(i.next(), n1);
		assertEquals(i.hasNext(), false);
		assertEquals(i.hasNext(), false);
		System.out.println("test1 OK");
	}
}
```

Uppgift

- Implementera NodeDFSIterator
  - P.g.a minnesanvändning får man inte göra detta genom att samla ihop alla element i t.ex. en privat räcka, och sedan ge ut dem en och en

- För att kompilera:
  - javac -cp junit-4.7.jar *.java
- För att exekvera:
  - java -cp junit-4.7.jar:. org.junit.runner.JUnitCore NodeTest