Organising information: trees

Silvio Peroni
silvio.peroni@unibo.it 0000-0003-0530-4305 @essepuntato

Computational Thinking and Programming (A.Y. 2017/2018)
Second Cycle Degree in Digital Humanities and Digital Knowledge
Alma Mater Studiorum - Università di Bologna

Communication 1

Today there will be also the official assessment of the course run by the University

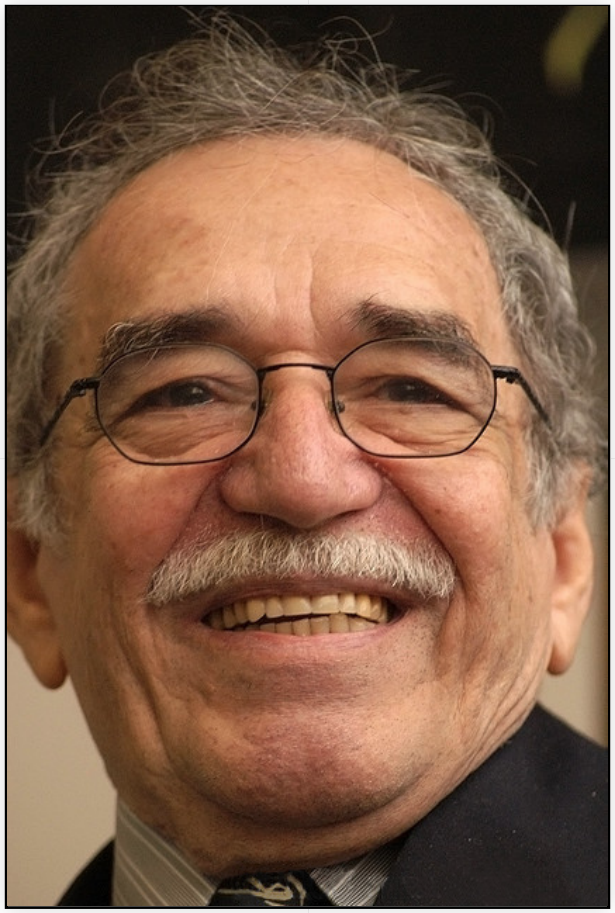
Communication 2

I will integrate the official assessment questionnaire with another brief one that will be presented to you after the final written examination

Any question about the previous lecture?

Historic hero: Gabriel Garcia Marquez

He was a novelist, and won the Nobel Prize for Literature in 1982



He is mainly known for his novels: *One Hundred Years of Solitude*, *Love in the Time of Cholera*

In the *One Hundred Years of Solitude* he narrates the story of seven different generations of people of the Buendia family

At the beginning of the book there is the Buendia family *tree*, for helping the reader to follow the story

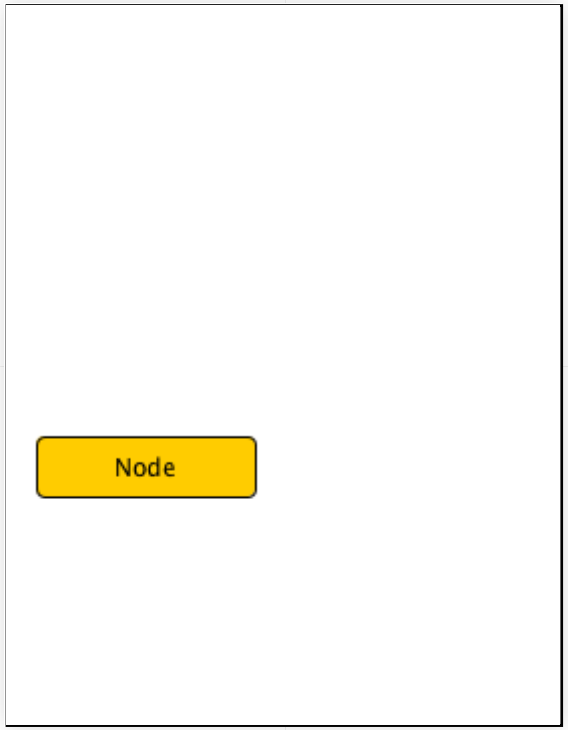
Other uses of trees

Recursive algorithm: we have used a tree for showing the execution of the recursive calls to the Fibonacci algorithm implemented by means of the divide and conquer approach

Document markup: languages such as TEI and HTML are examplars of languages which allows one to construct hierarchies of markup elements for structurally and semantically annotating a text

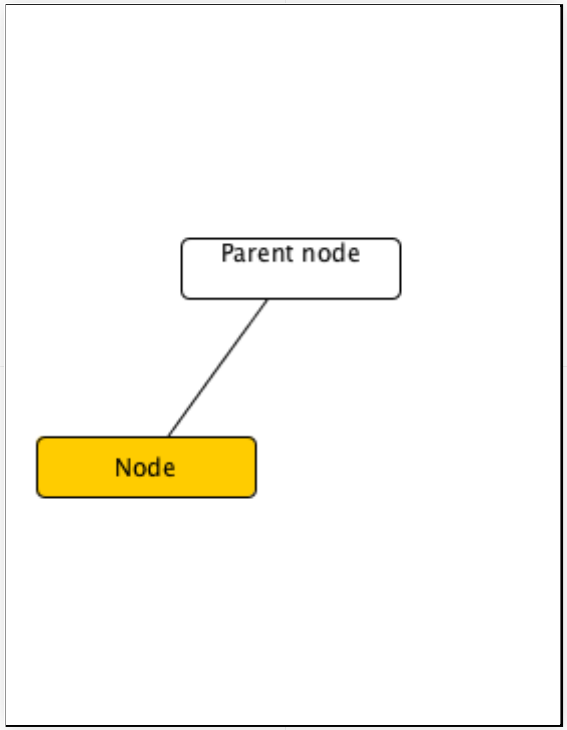
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



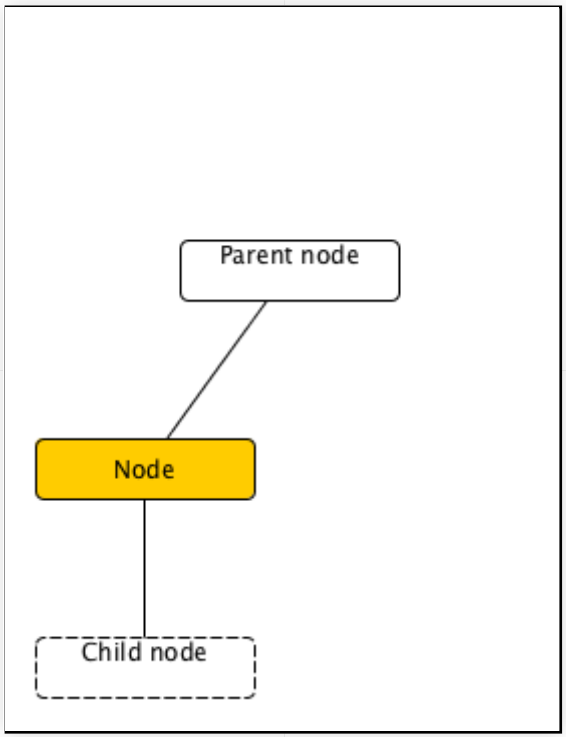
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



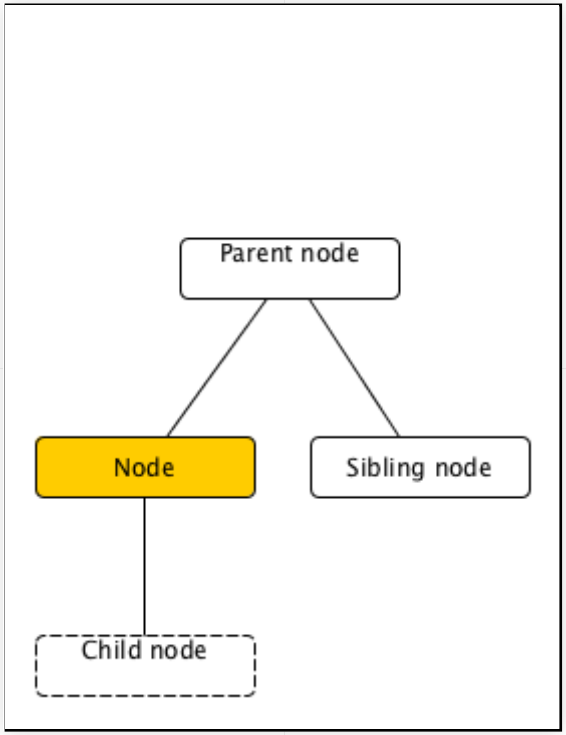
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



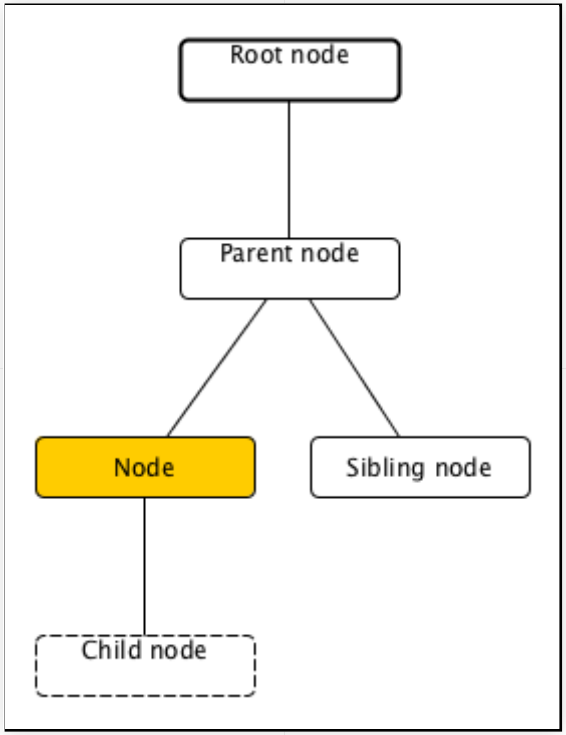
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



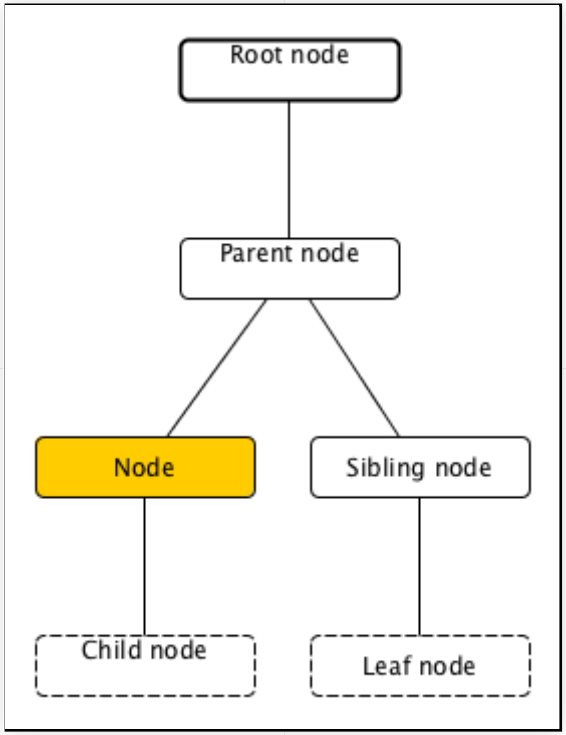
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



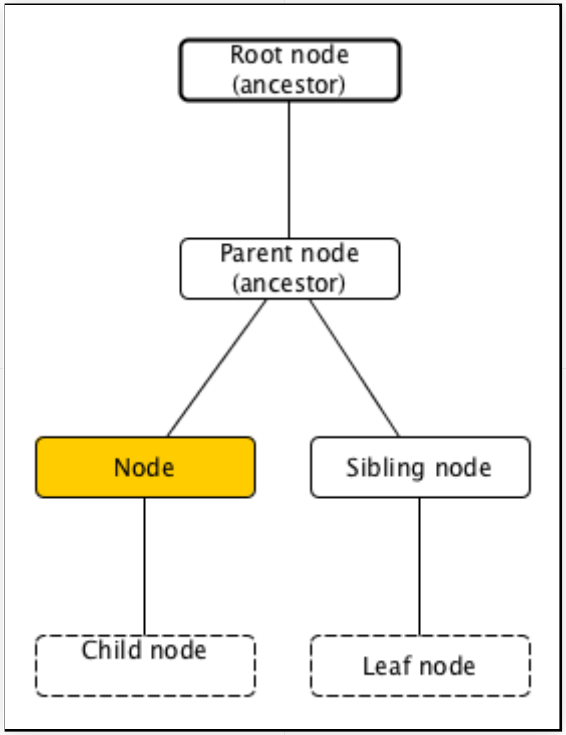
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



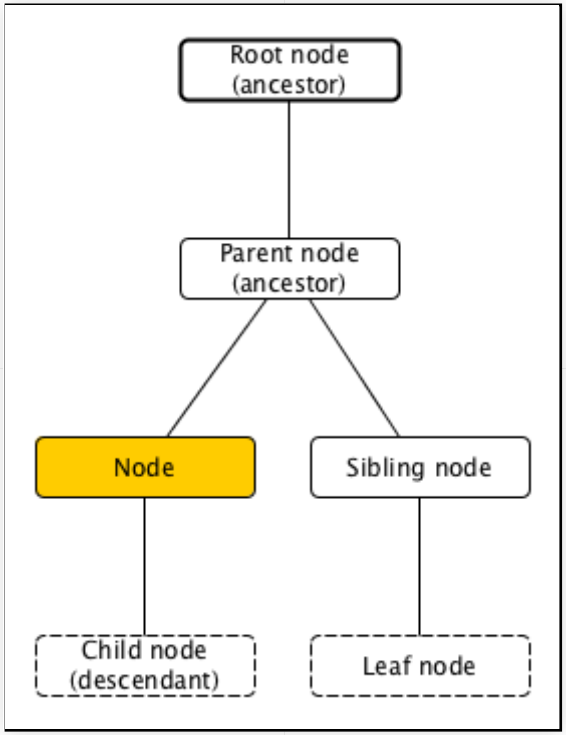
Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



Tree: definition

A tree is a data structure that simulates a hierarchical tree, composed by a set of nodes related to each other by a particular hierarchical parent-child relation



Tree and Python

There is no built-in implementation of the tree data structure in Python

We use the external *anytree* package

Constructor for nodes:
`def Node(name, parent=None)`

Each node must specify a name (as a string) and a parent – if the parent is not specified then it will assume `None` as value and it is implicitly defined as the root node of a tree

Example

```
from anytree import Node

root = Node("html")

head = Node("head", root)
title = Node("title", head)

body = Node("body", root)
paragraph_1 = Node("p", body)
paragraph_2 = Node("p", body)
```

The siblings of a certain parent are actually ordered among them – the order is defined by the order of insertion as a child of that particular parent

For instance, in the example, `paragraph_1` precedes `paragraph_2`

Hooks

- `<node>.children` returns a tuple listing all the children of a node
- `<node>.parent` returns the parent of a node
- `<node>.descendants` returns a tuple listing all the descendants of a node (including its children)
- `<node>.ancestors` returns a tuple listing all the ancestors of a node (including its parent)
- `<node>.siblings` returns a tuple listing all the siblings of a node
- `<node>.root` returns the root node of the tree where a node is contained

END

Organising information: trees

Silvio Peroni

silvio.peroni@unibo.it 0000-0003-0530-4305 @essepuntato

Computational Thinking and Programming (A.Y. 2017/2018)

Second Cycle Degree in Digital Humanities and Digital Knowledge

Alma Mater Studiorum - Università di Bologna