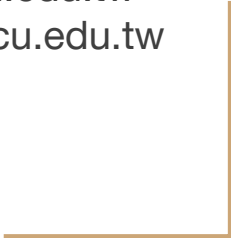




# HW6

陳卉縈 112356043@nccu.edu.tw  
王瀚 111306078@g.nccu.edu.tw  
劉亭妤 113356048@g.nccu.edu.tw



# HW6

Compute the score of a *website*!

- Construct a tree and its nodes according to a given website
  - An element (referred by a node) represents one web page and has three fields: (name, url, score)
- Given a keyword and its weight, compute the score of each node
  - $\text{Score} = \text{number of appearance} * \text{weight}$
  - The score of a node = the score of the content of its url + the scores of its children
  - This can be done by a postorder traversal of a tree
- Output the hierarchy of the website (with names and scores) using parentheses
  - This can be done by an euler tour

# Required Class for HW6

1. Main
2. Keyword : Store keywords
3. WordCounter : Count keyword occurrences
4. WebPage : Calculate webpage score
5. WebNode : Define the relationship between the parent and child nodes and store the accumulation score calculated by the WebPage
6. WebTree : Define tree

# Class: Main

- Main Webpage

Webpage ("URL", "Given website name")

```
WebPage rootPage = new WebPage("http://soslab.nccu.edu.tw/Welcome.html", "Soslab");
```

- Sub Webpage (Attached to the main webpage)

tree.root.addChild : Root node add child node

tree.root.children.get(1).addChild : Take child node whose index is 1 and add child node

```
tree.root.addChild(new WebNode(new WebPage("http://soslab.nccu.edu.tw/Publications.html", "Publication")));  
tree.root.addChild(new WebNode(new WebPage("http://soslab.nccu.edu.tw/Projects.html", "Projects")));  
tree.root.children.get(1).addChild(new WebNode(new WebPage("https://vlab.cs.ucsb.edu/stranger/", "Stranger")));  
tree.root.addChild(new WebNode(new WebPage("http://soslab.nccu.edu.tw/Members.html", "Member")));  
tree.root.addChild(new WebNode(new WebPage("http://soslab.nccu.edu.tw/Courses.html", "Course")));
```

# Class: Keyword

- Store the keyword

ex: [Yu,1.2]、[Fang, 1.8]

Keyword is a data structure defined as:

```
{  
    String name;  
    double weight;  
}
```

# Class: WordCounter

- Calculate the number of times the keyword appears in the webpage of each node
- Referring to HW3

# Class: WebPage

- Calculate each node's own webpage score

WebPage is a data structure defined as:

```
{  
    String url;  
    String name;  
  
    WordCounter counter; //from WordCounter  
    double score; // calculated from a given Keyword set  
}
```

# How to calculate WebPage's score

- Given  $w$ , which is a instance of WebPage
- Given  $K$ , which is a set of Keywords
- $w.score = \sum_{k \in K} k.weight \times \text{appearance times of } k.name \text{ in } w's \text{ HTML}$



# Class: WebNode

- Store the accumulation score (own score + child score) and the relationship between parent and child nodes

WebNode is a tree node structure defined as:

```
{  
    WebNode parent;  
  
    List<WebNode> children;  
  
    WebPage webPage; // node's own webpage score  
    double nodeScore; // own nodeScore + all children's nodeScore  
}
```

# Class: WebTree

- Building a tree
- Use postorder traversal to visit the tree (start from the bottom children), and use WebPage to calculate the own score, and the accumulation score will be stored in WebNode. The score of each child node will accumulate to the parent node until the root node, then use preorder(eular) to print the output.

WebTree is a data structure defined as:

```
{  
    WebNode root;  
}
```

# Input Example

Input: A set of keywords, the size of this set is N

N. [keyword.name1 keyword.weight1] [name2 weight2] [...]

Ex: 2 Yu 1.2 Fang 1.8

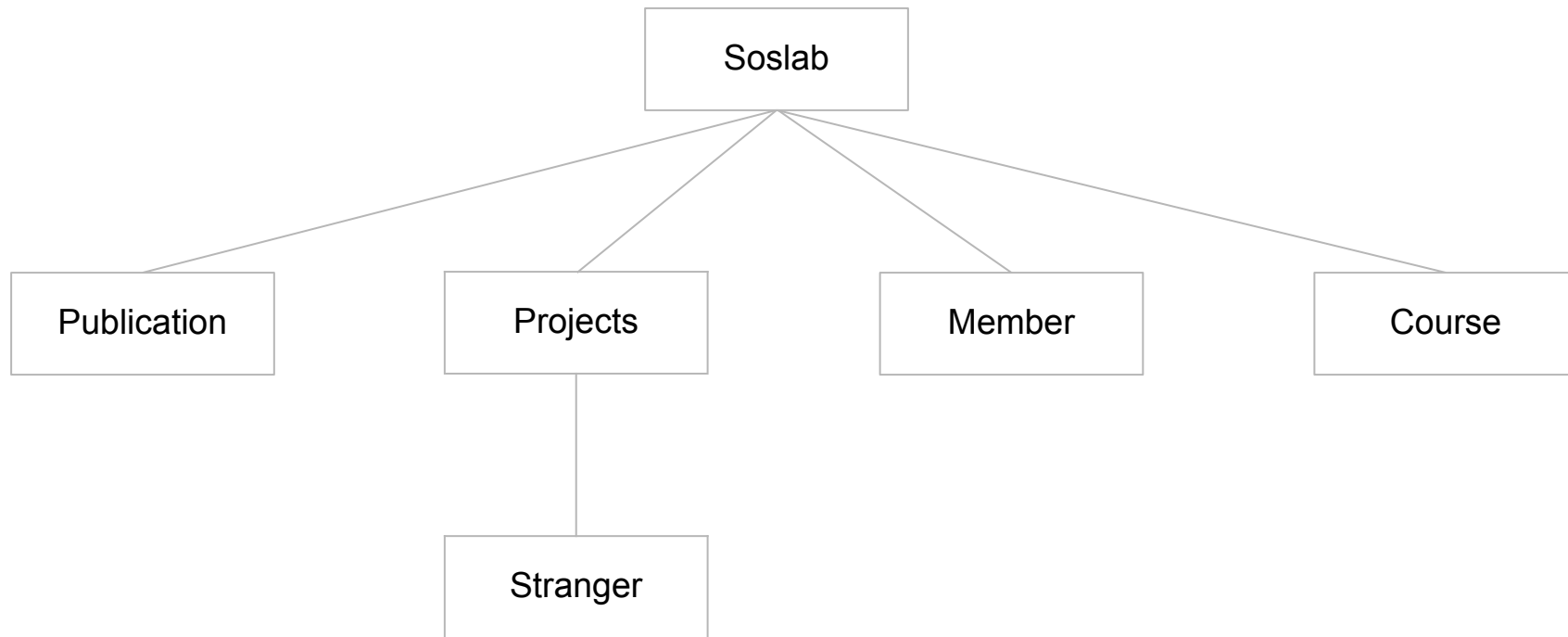
- N=1, only one keyword

Ex: 1 Fang 3.3

- N=3, three keywords

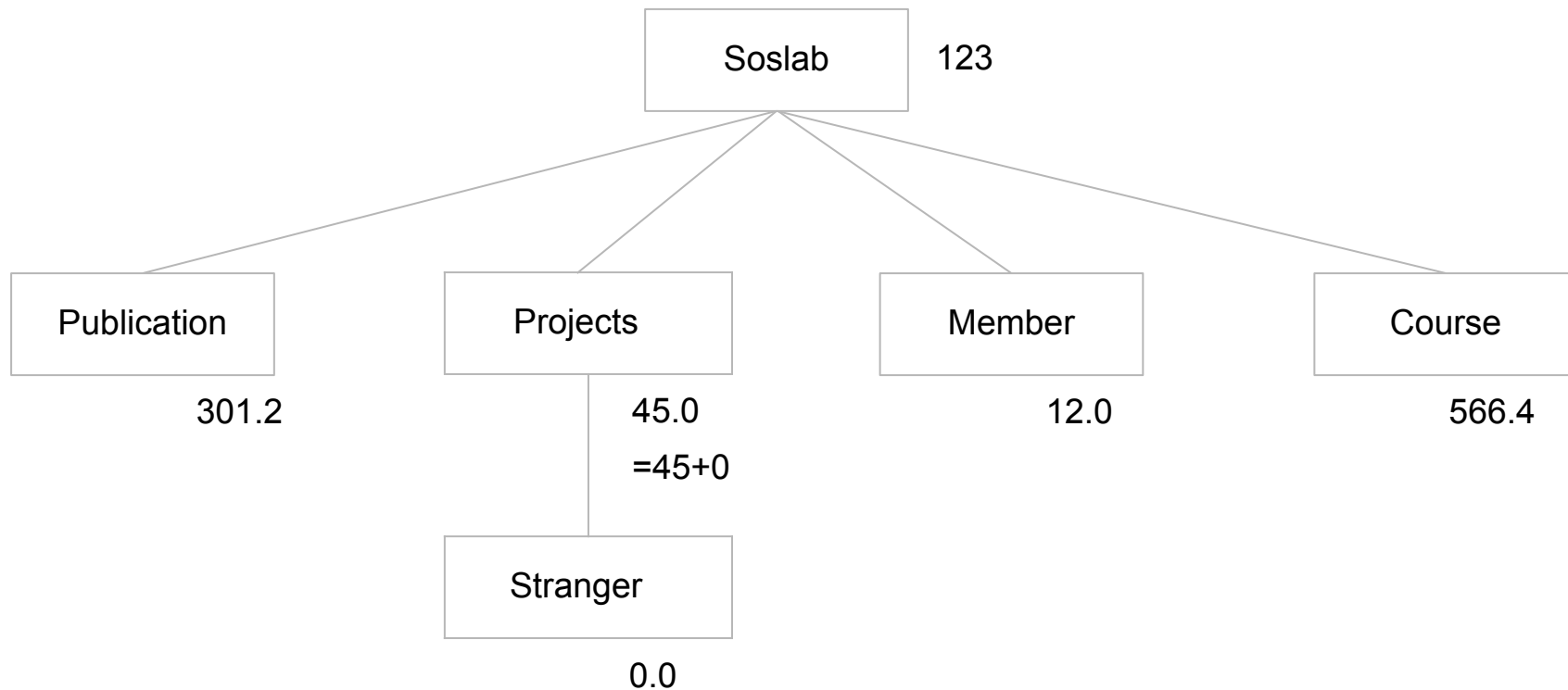
Ex: 3 Fang 3.3 Yu 4.4 soslab 2.0

# Tree - Main



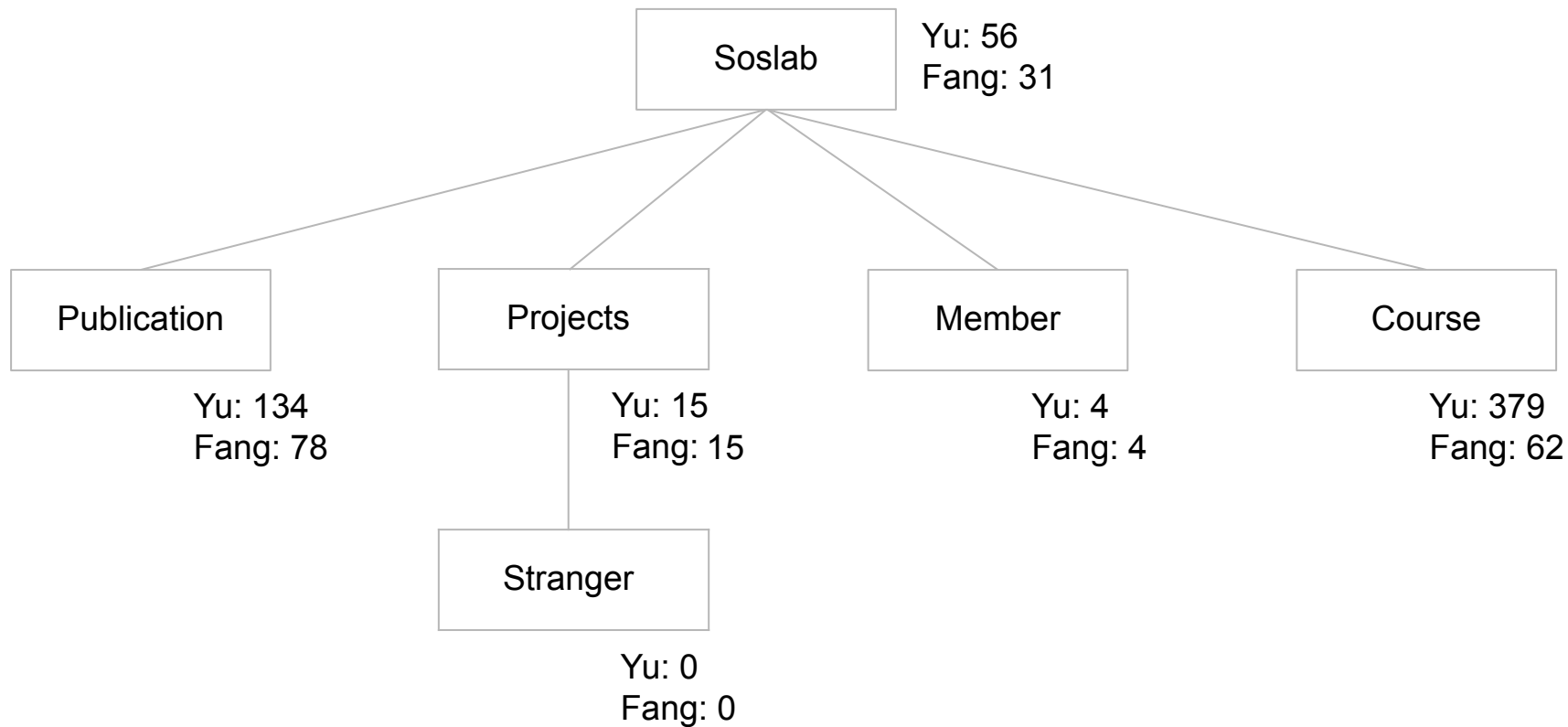
# Tree

Soslab total score: 1047.6  
~ = 123 + 301.2 + 45 + 12 + 566.4



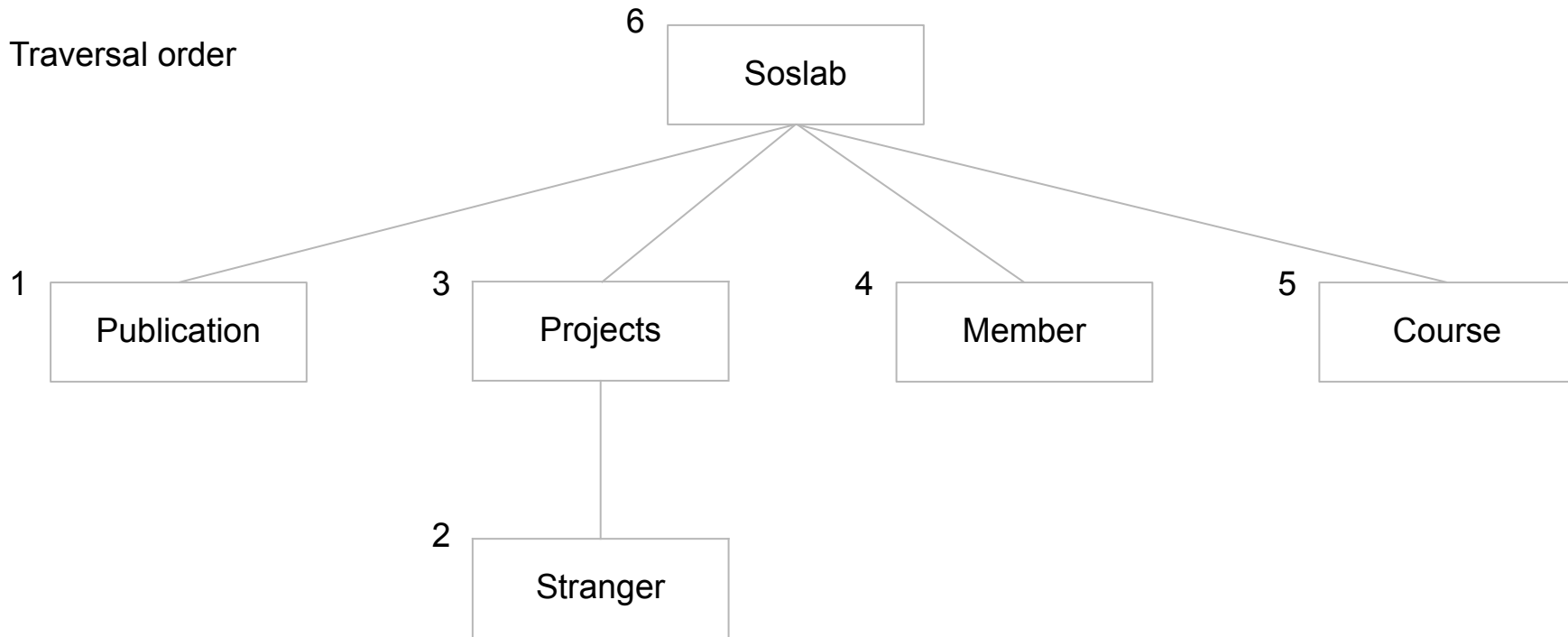
WebTree call WebNode to count nodeScore

# Tree



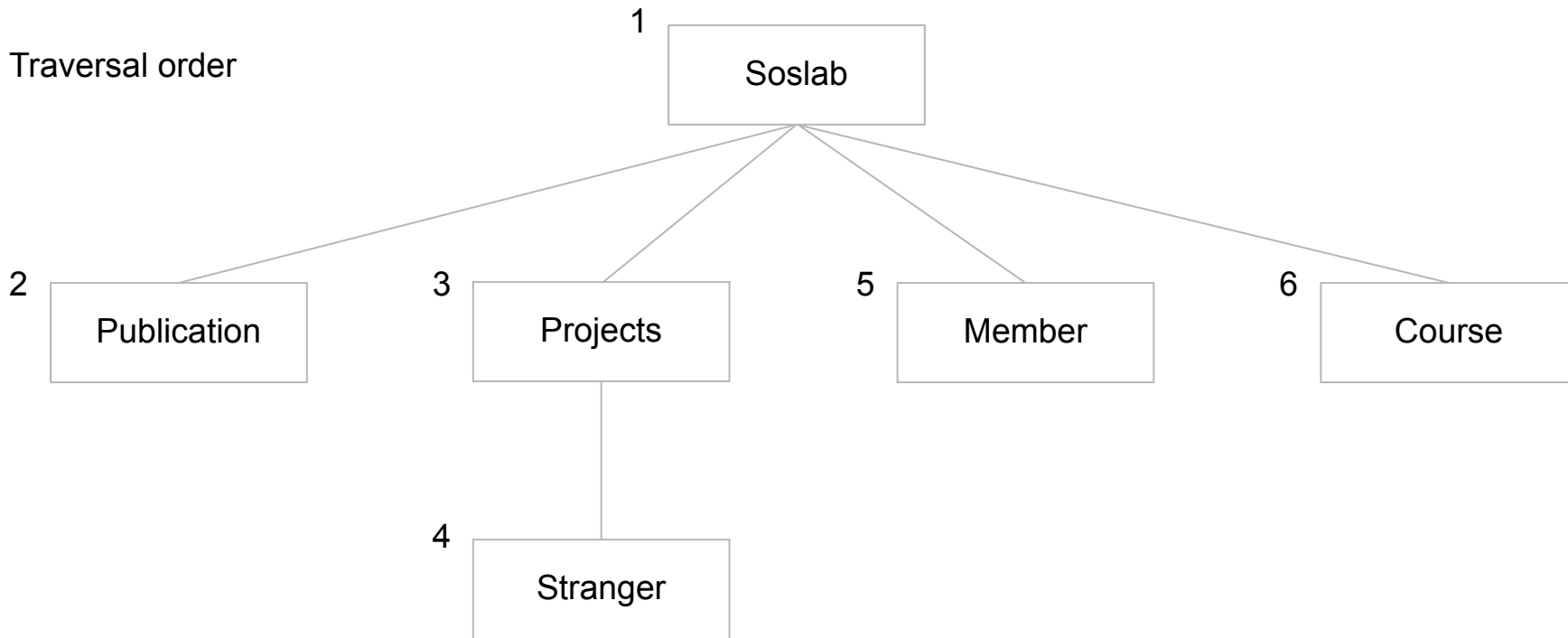
# Tree - setPostOrderScore

Traversal order



# Tree - eularPrintTree

Traversal order





# Output Example

Output: Given a set of keywords, , you shall output something like

```
(Soslab,1047.6
  (Publication,301.2)
  (Projects,45.0
    (Stranger,0.0)
  )
  (Member,12.0)
  (Course,566.4)
)
```

```
(Soslab,1047.6
  ←tab→ (Publication,301.2)
  (Projects,45.0
    ←tab→ (Stranger,0.0)
  )
  (Member,12.0)
  (Course,566.4)
)
```

newline, no extra space in the end

- Soslab 1047.6 indicates that the sum of the score in the content of the given URL (<http://soslab.nccu.edu.tw/Welcome.html>) and its sublinks