# Задание

# Реализовать функцию чтения и фильтрации онтологии

def read\_ontology(ontology\_path, categories):

…

где

* ontology\_path – путь к файлу с онтологиями;
* categories – список категорий.

Онтология приставляет собой CSV файл с табуляцией в качестве разделителя и содержит 3 столбца:

* id термина;
* сам термин;
* id родительского термина (если таковой имеется).

Например:

0 растения

1 деревья 0

2 хвойные 1

3 лиственные 1

4 сосновые 2

5 ели 2

6 пихты 2

7 лиственницы 2

8 секвойи 2

9 сосна 4

10 сосна горная 4

11 кедр 4

12 клён 3

Особенности онтологии:

* онтология может содержать несколько корневых терминов (т.е. терминов без id родителя)
* столбец id содержит уникальные значения;
* термин может состоять из нескольких слов;
* порядок строк произвольный, и термин может иметь родителя, который расположен с строках ниже.

Список категорий представляет собой id терминов, которые нужно загрузить вмести со всеми их дочерними терминами.

Функция должна вернуть список словарей, которые содержат: id термина, сам термин и id категории. Например, если список категорий [2, 4], то для онтологии из примера выше функция должна вернуть:

[

{"id": 2, "term": "хвойные", "category": 2},

{"id": 3, "term": "лиственные", "category": 3},

{"id": 12, "term": "клен", "category": 3},

{"id": 4, "term": "сосновые", "category": 2},

{"id": 5, "term": "ели", "category": 2},

{"id": 6, "term": "пихты", "category": 2},

{"id": 7, "term": "лиственницы", "category": 2},

{"id": 8, "term": "секвойи", "category": 2},

{"id": 9, "term": "сосна", "category": 2},

{"id": 10, "term": "сосна горная", "category": 2},

{"id": 11, "term": "кедр", "category": 2},

]

Примечание к реализации: реализация не должна использовать рекурсивные функции, т.к. в случае длинных цепочек родительских и дочерних терминов возможно переполнение стека.

# 2 Реализовать функцию подсчета слов указанных категорий в заданных текстовых файлах

def calculate\_stat (text\_paths, categories\_info):

…

где

* text\_paths – список путей к текстовым файлам, которые нужно обработать;
* categories\_info – результаты работы функции read\_ontology.

Функция должна подсчитать общее количество терминов из каждой категории для каждого файла и вернуть следующий список с результатами:

[{

"filename": "canidae.txt",

"term\_cateogires": {

3: {

"found\_terms": ["лиственные", "клен", "клен"],

"total\_terms": 3

},

2: {

"found\_terms": ["сосна", "ель", "пихта", "сосна"],

"total\_terms": 4

}

}

}, {

...

}]

Примечание к реализации:

* исходные текстовые файлы могу содержать мусорную информацию (html ссылки, таблицы и т.д.)
* формы слов в тексте могут не совпадать с формой терминов (число, падеж), поэтому термины и слова текстов следует привести к нормальной форме перед обработкой. Это можно выполнить с помощью библиотеки pymorphy2 (<https://github.com/kmike/pymorphy2>).
* для эффективного поиска множества слов в тексте следует найти какую-либо python библиотеку или реализовать свой алгоритм.