BibTeXPro, Um processador de BibTeX

Instituição: Universidade do Minho

Cadeira: Processamento de Linguagens e Compiladores (2021/2022)

Identificação: plc21TP1gr03

Equipa

Alef Keuffer (A91383)
Ivo Lima (A90214)
Catarina Quintas (A91650)

O formato $BibTeX$

$BibTeX$ é a ferramenta e formato de ficheiro usado para descrever e processar listas de referências, principalmente em conjunção com $LaTeX$.

Entrada $BibTeX$

Uma entrada $BibTeX$ consiste em um **tipo de entrada**, uma **chave de citação** e um número de **campos** que definem várias caraterísticas de uma entrada específica.[1].

Existem 17 tipos de entrada, sendo 14 categorias de referência e 3 de uso específico em $BibTeX$.

As entradas não são *case sensitive*[2].

A documentação diz que ficheiros $BibTeX$ podem conter 4 tipos de entrada: `@string`, `@preamble`, `@comment` e 14 categorias (e.g. `@article`, `@book`, etc)[3].

Note que:

`@string`
: define abreviações que podem se usadas depois em um campo.

`@preamble`
: como um texto especial deve ser formatado.

`@comment`
: para comentários que não devem ser levados em conta pelo $BibTeX$.

Estas três não seram processadas pelo nosso programa.

Especificação da Solução

A nossa solução deve satisfazer os seguintes requisitos:

(R1)
: Fazer a contagem das categorias presentes no documento, tais como: *phDThesis*, *Misc*, *InProceeding*, etc.

(R2)
: Produzir um documento em formamto *HTML* com (R2.1) o nome das categorias encontradas e (R2.2) respectivas contagens.

(R3)
: Filtrar, para cada entrada de cada categoria, a respetiva (R3.1) chave, (R3.2) autores e (R3.3) título. (R3.4) O resultado final deverá ser incluído no documento *HTML* gerado em (R2).

(R4)
: Criar um índice de autores, que mapeie cada autor nos respectivos registos, de modo a que posteriormente uma ferramenta de procura do Linux possa fazer a pesquisa.

(R5)
: Construir um Grafo que mostre, para um dado autor (definido à partida) todos os autores que publicam normalmente com o autor em causa.

(R6)
: Recorrendo à linguagem *Dot* do *GraphViz*, gerar um ficheiro com o grafo de (R5) de modo a que possa, posteriormente, usar uma das ferramentas que processam *Dot* para desenhar o dito grafo de associações de autores.

Execução do programa

Para realizar as modificações no ficheiro usamos `solve(author_name,INPUT_FILENAME=BIB_EXAMPLE_FILENAME)` , passando como argumento o nome do autor que queremos conforme (R5).

```
echo = False #!/usr/bin/python
```

Módulo não é `re`

Usamos o módulo `regex` que é um *superset* de `re` mais poderoso. Seu uso será justificado.

1. http://www.bibtex.org/Format/↩

2. https://tex.stackexchange.com/questions/163687/is-there-a-preferred-capitalization-style-for-reference-types-in-bibtex-biblatex↩

3. http://www.bibtex.org/Format/↩

```
import regex as re
```

Algumas constantes

Aqui incluimos *MathJax*, uma biblioteca em *javascript*, para renderizar fórmulas matemáticas nos navegadores.

```
MATHJAX = '''
            <script type="text/x-mathjax-config">
                MathJax.Hub.Config({
                    "extensions":["tex2jax.js"],
                    "jax":["input/TeX",
                           "output/HTML-CSS"],
                    "messageStyle":"none",
                    "tex2jax":{
                        "processEnvironments":false,
                        "processEscapes":true,
                        "inlineMath":[["$","$"]],
                        "displayMath":[]
                    },
                    "TeX":{
                        "extensions":["AMSmath.js",
                                      "AMSsymbols.js",
                                      "noErrors.js",
                                      "noUndefined.js"]
                    },
                    "HTML-CSS":{
                        "availableFonts":["TeX"]
                    }
                });
            </script>

            <script type="text/javascript" async
                src="https://cdnjs.cloudflare.com/ajax/libs/mathjax/2.7.5/MathJax.js">
            </script>
'''
```

O início do nosso *HTML* é

```
HTML_PROLOGUE = f'''<!DOCTYPE  html>
    <HTML lang="en">
        <HEAD>
            <meta charset="utf-8">
            <TITLE>Categories in BibTeX</TITLE>
            {MATHJAX}
        </HEAD>'''
```

Vamos ter que fechar a tag *HTML* que começamos em `HTML_PROLOGUE`:

```
HTML_EPILOGUE = '</HTML>'
```

O ficheiro dado como argumento de entrada é o ficheiro `exemplo-utf8.bib`.

```
ILENAME = "exemplo-utf8.bib"
```

Decidimos nomear o output `output.html`.

```
OUTPUT_FILENAME = 'output.html'
```

Função principal

Em `solve()` temos a váriavel `html_str_ls` que é uma lista de *strings* que serão concatenadas. A título de optimização da imutabilidade de *strings* (concatenação de *strings* em *python* cria uma cópia de cada), optamos por fazer sucessivos *appends* de custo $\mathcal{O}(1)$ e, por fim, concatena-las para formar o código *HTML*.

Em `bib_str` guardamos todo o texto contindo em `INPUT_FILENAME`.

```
def solve(author_name,INPUT_FILENAME=BIB_EXAMPLE_FILENAME):
    html_str_ls = [HTML_PROLOGUE]
    bib_str = get_bib_str(INPUT_FILENAME)

    entries = get_entries(bib_str)
    format_authors(entries)
    fix_repeated_authors(entries)

    html_str_ls.append(
        html_enclose('body',
            get_html_pub_type_counts(entries)
            + get_html_common_pub_author(author_name,entries)
            + get_html_pub_type_index(entries)
            + get_html_author_index(entries)))

    html_str_ls.append(HTML_EPILOGUE)
    with open(OUTPUT_FILENAME,'w') as file:
        file.write('\n'.join(html_str_ls))
```

`get_bib_str()` é utilizada para abrir e extrair o conteúdo do ficheiro.

```
def get_bib_str(filename):
    with open(filename,'r') as file:
        return file.read()
```

Recolhendo as entradas

Neste procedimento, criamos um dicionário onde cada chave é um par (categoria de publicação, nome do autor) e o valor é um dicionário em que cada chave é um campo da entrada $BibTeX$ e o valor é o valor do campo.

```
def get_entries(string):
    d = {}
```

Poderíamos coletar todos os campos de todas as entradas, mas neste trabalho só são relevantes alguns campos. Então, para economizar espaço no dicionário, definimos:

```
    RELEVANT_FIELDS = {'author','title'}
```

Como foi dito anteriormente, alguns tipos de entrada não são categorias de referência. Iremos ignorá-los.

```
    SPECIAL_TYPES = {'comment','string','preamble'}
```

É aqui que se justifica o uso do módulo `regex`. Note a expressão regular:

```
    between_cbrace_ex = r'(?:(?<rec>{(?<value>[^{}]+|(?P>rec))*+}))'
```

Nela usamos um padrão recursivo não suportado pelo módulo `re`. O padrão é usado para capturar o que está entre chavetas, que podem estar aninhadas arbitrariamente.

```
    field_match = re.compile(rf'(?<name>\w+)\s*=\s*({between_cbrace_ex}|"(?<value>[^"]+)"|(?<value>\d+))')
    entry_match = re.compile(rf'@(?<type>\w+)(?<value>{between_cbrace_ex})')
    key_match = re.compile(r'([^{},~#%\\]+),')

    for entry in entry_match.finditer(string):
        entry_type = entry['type'].lower()
        if entry_type not in SPECIAL_TYPES:
            key = key_match.search(entry['value'])[1]
            d[entry_type, key] = {
                field['name'].lower(): field['value']
                for field in field_match.finditer(entry['value'])
                if field['name'].lower() in RELEVANT_FIELDS
            }

    return d
```

Procedimentos usados ao longo do programa

Relacionados a expressões $LaTeX$

Chavetas

Assumimos que as chavetas estão balanceadas.

Chavetas dentro de títulos são usadas para prevenir que palavras sejam convertidas em letras minúsculas (se “sentence style” é usado ao invés de “title style”)[1].

Por exemplo, devemos usar chavetas para escrever

```
title = "The Life of {Albert} {Einstein}"
```

ou, de forma equivalente,

```
title = {The life of {Albert} {Einstein}}
```

se quisermos garantir que as letras ‘A’ e ‘E’ sejam sempre formatadas em letras maiúsculas mesmo que “sentence style” esteja sendo usado.

Suspeitamos que talvez pudessemos usar:

```
r'[^\\]\$.*[^\\]\$'
```

para identificar que não estamos dando *escape* no *dollar sign*. Mas já tinhamos esse procedimento feito e decidimos por não alterá-lo.

1. https://tex.stackexchange.com/questions/109064/is-there-a-difference-between-and-in-bibtex:↩

```
def unbrace(expression):
    string_ls = []
    is_between_dollar_sign = False
    is_previous_backslash = False
    for c in expression:
        if c == '$' and not is_previous_backslash:
            if is_between_dollar_sign:
                is_between_dollar_sign = False
            else:
                is_between_dollar_sign = True
        if c == '\\':
            is_previous_backslash = True
        else:
            is_previous_backslash = False
        if c in '{}' and not is_between_dollar_sign:
            continue
        string_ls.append(c)
    return ''.join(string_ls)
```

Relacionados a *HTML*

É muito comum em *HTML* o uso de *tags* para marcar uma expressão. Então, criamos um procedimento que facilita cercar uma expressão pela *tag* desejada.

```
def html_enclose(tag,string):
    return rf'<{tag.upper()}>{string}</{tag.upper()}>'
```

Nesse sentido, *spans* são muito úteis em *HTML*, logo criamos:

```
def html_create_span(expression):
    return html_enclose('span',expression)
```

Às vezes queremos adicionar um atributo numa *tag*, nesse caso podemos usar (há um pequeno problema com esse procedimento, mas para nosso uso não foi relevante):

```
def html_add_attr(attr,val,html_expression):
    return re.sub(r'<(\w+)([^>]*)\s*>(.*)</\1>',rf'<\1\2 {attr.upper()}="{val}">\3</\1>',html_expression)
```

Relacionados a *Regex*

Em certas ocasiões, desejamos fazer várias substituições seguidas numa *string*, logo criamos:

```
def mult_replace(string, replacement_list):
    for old, new in replacement_list:
        string = re.sub(old, new, string)
    return string
```

Tratamento do nome dos autores

Escolhemos remover acentuações e caráteres especiais que as representam em $LaTeX$ (e.g. `\~`) do nome dos autores.

É importante notar que ainda ocorrem alguns problemas de acentuações uma vez que não removemos acentuações do tipo `\~{}` .

```
def format_authors(data):
    for d in data.values():
        if "author" in d:
            author_lst = [ remove_consecutive_spaces(
                           str.strip(
                           invert_name(
                           unbrace(
                           remove_accents(name)))))
                           for name in re.split(r"\band\b", d["author"].replace("\n", " "))]
            d['author'] = [author for author in author_lst if author]
```

Procedimentos auxiliares

```
def remove_consecutive_spaces(name):
    return re.sub(r'\s+',' ',name)
```

Inversão das componentes do nome

Para inverter nomes do tipo: last_name, first_name. Por exemplo: “da Cruz, Daniela” → “Daniela da Cruz”

```
def invert_name(author_name):
    return re.sub(r"([^,]+),\s*([^,]+)", r"\2 \1", author_name)
```

Remoção de acentuações

```
def remove_accents(name):
    return remove_latex_accent(remove_normal_accent(name))
```

Para remoção de acentos $LaTeX$ consultamos a wiki.

Pensamos em possivelmente usar algo como:

```
re.sub(r'\\{?(acento)}?',rf'{fix_accent(\1)}',author_name)
```

para manter acentuações, mas devido a restrições de tempo não achamos prudente embarcar nesta ideia.

```
def remove_latex_accent(name):
    return re.sub(r'\\W','',name)
```

Aqui removemos palavras com acentuações normais, isto é, palavras como “á”, “é”, “í”, etc (não $LaTeX$).

```
def remove_normal_accent(name):
    import unicodedata
    return ''.join((c for c in unicodedata.normalize('NFD', name) if unicodedata.category(c) != 'Mn'))
```

Também usamos a já referenciada `unbrace()` .

Tratando autores repetidos na estrutura

No dicionário, teremos autores repetidos (pois podem estar escritos de formas diferentes, omitindo alguns sobrenomes ou até mesmo primeiros nomes).

Por exemplo, 'M. J. Varanda', 'M. Joao Varanda', 'Maria Joao Varanda' e 'Maria Joao V. Pereira' são diferentes maneiras de se referir a mesma pessoa.

```
def fix_repeated_authors(data):
    author_blocks = fix_block_func(
                        block_authors_with_two_common_names_v2(
                            get_author_list(data)))

    author_dict = {author_name:max(s,key=len)
                   for s in author_blocks
                   for author_name in s}

    for d in data.values():
        d['author'] = [author_dict[author]
                       for author in d['author']]
```

Para lidar com isso, realizamos três procedimentos:

1. Obtemos uma lista dos autores ordenada alfabeticamente.

2. Criamos conjuntos onde os seus elementos representam um mesmo autor. Ou seja, se temos três nomes: A, B e C e temos os conjuntos $\{A, B\}$ e $\{B, C\}$, então os nomes A, B, C referem-se ao mesmo autor.

3. Juntamos esses blocos para que cada bloco seja um único autor. Ou seja, determinamos uma "transitividade" nos blocos, usando o exemplo anterior, se temos $\{A, B\}$ e $\{B, C\}$, devemos obter $\{A, B, C\}$.

Procedimento 1

```
def get_author_list(data):
    return sorted(set([a for s in data.values() for a in s.get("author", [])]))
```

Procedimento 2

```
def block_authors_with_two_common_names_v2(authors):
    res = set()
    for author in authors:
        fs = set()
        for author2 in authors:
            a1 = set(re.findall(r'\w\w+',author))
            a2 = set(re.findall(r'\w\w+',author2))
```

Se os autores tem mais de duas componentes do nome em comum, então podemos considera-los o mesmo autor.

```
            if len(a1.intersection(a2)) > 1:
                fs.add(author2)
```

Para lidar com a situação descrita abaixo, assumimos que a primeira componente de um nome nunca será omitida.

Caso especial: Suponha que existe um autor que só tem uma componente do nome não (e.g. P. Henriques). Suponha que outro autor (e.g. P.R. Henriques) também só

tem uma componente do nome não abreviada e essa componente é igual a do primeiro autor. Então ainda temos que fortalecer a condição para que sejam considerados iguais. Para isso, usamos `is_a_first_last_match()` garantindo que pelo menos a abreviação da primeira componente dos nomes sejam iguais.

```
            elif len(a1) == 1 and len(a1.intersection(a2)) == 1 and is_a_first_last_match(author,author2):
                fs.add(author2)
        res.add(frozenset(fs))
    return res
```

Pra implementar `is_a_first_last_match()` usamos `get_crude_abbrev()` para obter uma string com o primeiro caráter de cada componente do nome de um autor (e.g. "Ricardo Henriques → RH").

```
def get_crude_abbrev(name):
    return ''.join(c for c in name if c.isupper())
```

Aqui iremos verificar se os primeiros caráteres da primeira e última componente do nome de um autor a_1 são iguais, respetivamente, aos primeiros e últimos caráteres da primeira e última componente do nome de um autor a_2.

```
def is_a_first_last_match(author1,author2):
    a1 = get_crude_abbrev(author1)
    a2 = get_crude_abbrev(author2)
    return a1[0] == a2[0] and a1[-1] == a2[-1]
```

Procedimento 3

Objetivo: Criar a "transitividade" nos blocos dos autores referenciada no ínicio.

```
def fix_block_func(data):
    res = set()
    for q in data:
        for s2 in data:
            if q.intersection(s2) != set():
                q = q.union(s2)
        res.add(frozenset(q))
    return res
```

Contagem dos tipos de publicações

Conforme (R1), devemos contar quantas publicações de cada tipo existem.

```
def get_pub_type_counts(data):
    pub_types_occur = [x[0] for x in data.keys()]
    pub_types = set(pub_types_occur)
    return [(pub_type, pub_types_occur.count(pub_type)) for pub_type in pub_types]
```

Conforme (R2) queremos incorporar a contagem de publicações de cada tipo no documento *HTML* que iremos produzir.

```
def get_html_pub_type_counts(data):
    string_ls = [html_enclose('h2',
                              'Number of Occurrences of Publication Types')]
    pub_counts = sorted(
        get_pub_type_counts(data),
        key=lambda x: x[1],reverse=True)
        mbda v: 's' if v > 1 else ''
    for pub_type, count in pub_counts:
```

```
        string_ls.append(
            html_enclose('p',
                         f'Type {pub_type} appears {count} time{time(count)}'
            )
        )

    return ''.join(string_ls)
```

Solução inicial

A estratégia para satisfazer (R1) consistiu em ler o arquivo linha a linha verificando se a categoria encontrada já aparecia no dicionário, se já existir, irá ser incrementado o número de ocorrências, senão será adicionado como primeira ocorrência, para que depois possa ser produzido um ficheiro *HTML* com todas as categorias e o devido número de ocorrências.

De seguida apresentamos a Expressão Regular utilizada para filtrar a informação pedida em (R1).

Uma vez que todas as categorias num $BibTeX$ têm como antecedente o caráter `@` e terminam numa `{`, tornou-se fácil criar um filtro que guarde toda a informação delimitada entre esses dois parâmetros.

Através dessa pequena realização chegamos à seguinte expressão: `r'^@(.*){'`

```
 import re

file = open("exemplo-utf8.bib", "r")
read = True
dic = {}
string_ls = ['<!DOCTYPE  HTML PUBLIC>\n<HTML>\n   <HEAD>\n      '\
             '<TITLE>Categories in BibTeX</TITLE>\n   </HEAD>\n   <BODY>']
while read:
    linhaFicheiro = file.readline()
    ncat = re.match(r'^@(.*){',linhaFicheiro)
    if ncat != None:
       cat_title = ncat.group(1).title()
       dic[cat_title] = dic.get(cat_title,0) + 1
    if not linhaFicheiro:
       read = False
       file.close()

time = lambda v: 's' if v > 1 else ''

for k, v in dic.items():
    string_ls.append(f'      <P>The category {k} appears {v} time{time(v)}.</P>')
    string_ls.append(f'   </BODY>\n</HTML>')
with open('output.html','w') as file:
     file.write('\n'.join(string_ls))
```

Infelizmente a estratégia adotada na primeira questão de unicamente guardar aquilo que interessava tornou-se impraticável pois não era escalável para a extração e manipulação necessária dos restantes parâmetros pedidos nas outras questões.

Criação do grafo

Conforme (R5) queremos construir um grafo que represente a co-autoria entre os autores.

```
def get_author_pub_graph(author,data):
    pub_partners = []
                in data.values():
            uthor' in entry and author in entry['author']:
            for partner in entry['author']:
```

```
            if partner != author:
                pub_partners.append(partner)
    return [(author_name,pub_partners.count(author_name))
            for author_name in set(pub_partners)]
```

Conforme (R6) iremos recorrer a linguagem *Dot* para renderizar o grafo.

```
def get_dot_graph(author,data):
    import textwrap
    g = sorted(get_author_pub_graph(author,data),key = lambda x: x[1])
    string_ls = ['graph{']
    string_ls2 = []
    for partner_author,no_joint_pub in g[-3:]:
        string_ls2.append(f'"{author}" -- "{partner_author}" [label="{no_joint_pub}"]')
    string_ls.append(textwrap.indent('\n'.join(string_ls2),'  '))
    string_ls.append('}')
    return '\n'.join(string_ls)
```

Como ideia nossa para ir além do que foi pedido, decidimos incorporar o grafo no documento *HTML* que iremos produzir.

Utilizamos `re.search()` porque o arquivo gerado contém um preâmbulo *XML* como *doctype*. Só queremos o *SVG*.

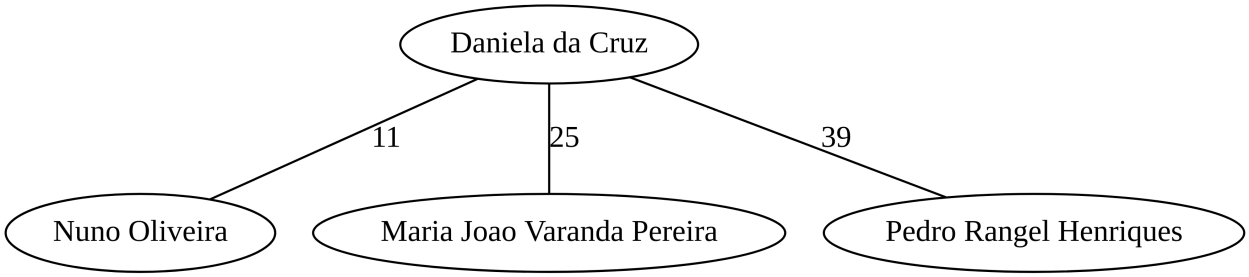
```
def get_html_dot_svg(author,data):
    import os
    DOT_INPUT_FILENAME = 'dot_input'
    with open(DOT_INPUT_FILENAME,'w') as file:
        file.write(get_dot_graph(author,data))
    os.system(f'dot -T svg -O {DOT_INPUT_FILENAME}')
    with open(DOT_INPUT_FILENAME + '.svg','r') as file:
        return re.search(r'<svg(?:.|\n)+</svg>',file.read()).group()
```

Finalmente,

```
def get_html_common_pub_author(author,data):
    string_ls = [html_enclose('h2','Author Graph')]
    string_ls.append(get_html_dot_svg(author,data))
    return ''.join(string_ls)
```

que é responsável por gerar:

Author Graph



na página *HTML*.

Filtrar (chave,autore,título)

Títulos

Pelo (R3.3), devemos incluir títulos. Para isso devemos aplicar um tratamento para uma formatação mais elegante (como manter expressões $LaTeX$).

Note que formatmos *small caps* como pode ser notado em entradas que contém "Camila". Também formatoms *sans serif* e algumas expressões matemáticas.

```
def fix_title(title):
    substitutions = [(r'\\textsc{((?:\\{|[^{])+)}',
                      lambda m: f'{html_to_small_caps(html_create_span(m.group(1)))}'),
                     (r'\\textsf{((?:\\{|[^{])+)}',
                      lambda m: f'{html_to_sans_serif(html_create_span(m.group(1)))}'),
                     (r'(\$(?:.|\\$)+\$)',
                      lambda m: f'{str_to_html_math(m.group(1))}')]


    replace = lambda x: mult_replace(x,substitutions)

    return   html_create_span(
             single_quote_latex(
             double_quote_latex(
             unbrace(
             replace(
             remove_latex_special_chars(
             ' '.join(s.strip() for s in title.split('\n'))))))))
```

Tratamento de caráteres $LaTeX$

```
def double_quote_latex(expression):
    return re.sub(r"``(.*[^\\])''",r'"\1"',expression)
```

```
def single_quote_latex (expression):
    return re.sub(r"[^`]`([^`].*[^'])'[^`]",r"'\1'",expression)
```

```
def remove_latex_special_chars(latex_expression):
    return re.sub(r'\\({|[^{])\b',r'\1',latex_expression)
```

Formatações especiais

Small caps

Usado para converter expressoes latex `\textsc{expression}`

```
def html_to_small_caps(html_expression):
    return html_add_attr('style','font-variant:small-caps',html_expression)
```

Para incluir no *HTML*:

```
def str_to_html_small_caps(expression):
    return html_to_small_caps(html_create_span(expression))
```

Sans serif

```
def html_to_sans_serif(html_expression):
    return html_add_attr('style','font-family:sans-serif',html_expression)
```

Matemática


```
ml_math(string):
    return html_add_attr('class','math inline',html_create_span(string))
```

Incorporar no documento *HTML*

Como fica renderizado:

Publication Type Index

article

Key = jj96
Title = NLlex -- a tool to generate lexical analysers for natural language
Autores = Jose Joao Dias de Almeida

Conforme (R3.4):

```
def get_html_pub_type_index(data):
    string_ls = [html_enclose('h2','Publication Type Index')]
    for entry_type in sorted(set(x[0] for x in data)):
        string_ls.append(html_enclose('h3',entry_type))
        for citation_key in [x[1] for x in data if x[0]==entry_type]:
            title = data[entry_type,citation_key].get('title','')
            authors = ', '.join((sorted(data[entry_type,citation_key].get('author',''))))
            string_ls.append(
                html_enclose('p',
                             f"Key = {citation_key}<br>Title = {fix_title(title)}<br>Autores = {authors}"))

    return '\n'.join(string_ls)
```

Índice de autores

Conforme (R4) iremos criar um índice de autores.

Geralmente, nos índices de autores, o apelido aparece primeiro seguido pelas iniciais dos outros nomes. Com esse propósito, criamos o procedimento que recebe um nome normalizado (e.g. Pedro Filipe H. Pereira), retornando-o “invertido” e abreviado (e.g. Pereira, P. F. H.).

```
def last_name_first(name):
    initials =  '. '.join(get_crude_abbrev(name))[:-2]
    last_name = name.split()[-1]
    return f'{last_name}, {initials}'
```

Utilizamos `last_name_first()` na construção de um dicionário que irá conter como chaves o nome do autor já formatado e como valor todas as chaves de citações de suas publicações.

```
def get_author_index_dict(data):
    index = {}
    for key, e in data.items():
        if 'author' in e:
            for author in e['author']:
                author_name = last_name_first(author)
                if author_name not in index:
                    index[author_name] = set()
                index[author_name].add(key[1])
    return index
```

Decidimos incorporar o índice no documento *HTML* que iremos produzir.

```
def get_html_author_index(data):
    index = sorted(get_author_index_dict(data).items())
    alphabet_order = sorted(set(c[0][0] for c in index))
    string_ls = [html_enclose('h2','Author Index')]
    i = 0
    string_ls.append(html_enclose('h3',alphabet_order[i]))
    for author,citation_keys in index:
        if author[0] ≠ alphabet_order[i]:
            i += 1
            string_ls.append(html_enclose('h3',alphabet_order[i]))
        citation_keys_str = ', '.join(citation_keys)
        string_ls.append(html_enclose('p',f'{author}, {citation_keys_str}'))
    return '\n'.join(string_ls)
```

Execução pela linha de comandos

Este bloco é utilizado para executar o programa pela linha de comandos.

```
if __name__ == '__main__':
    import sys
    import os.path
```

Para facilitar a realização de testes deixamos o valor *standard* do `filename` para o ficheiro exemplo.

```
    filename = sys.argv[1] if len(sys.argv) > 1 else BIB_EXAMPLE_FILENAME
    assert os.path.isfile(filename)
```

Novamente, para facilitar a realização de testes (executar sem passar o argumento com o nome do autor) escolhemos a autora Daniela da Cruz que está presente no ficheiro `exemplo-utf8.bib` com o intuito de gerar o grafo conforme (R5).

```
    if len(sys.argv) < 3:
        author_name = 'Daniela da Cruz'
    else:
        author_name = sys.argv[2]
    solve(author_name,filename)
```

Conclusão

Através deste projeto foi possível expandir as nossas competências intelectuais sobre o tópico de estudo: Expressões Regulares (ER) que nos possibilitou desenvolver um processador $BibTEX$ utilizando a linguagem *Pyhton* para extrair informações relavantes. Além disso, aumentamos a nossa familiaridade com *HTML* pois dispusemos as informações coletadas numa página *web*, onde inserimos um grafo gerado usando a linguagem *Dot*.

Consideramos que o produto final cumpre os requisitos.

Este projeto foi desafiante (pela complexidade do $LaTEX$) e enriquecedor (pela variedade de linguagens usadas) para cada um de nós, uma vez que tivemos a oportunidade de expandir, aprofundar e aperfeiçoar os nossos conhecimentos.

Apêndices

Source Code

```
 import regex as re

MATHJAX = '''
            <script type="text/x-mathjax-config">
                MathJax.Hub.Config({
                    "extensions":["tex2jax.js"],
                    "jax":["input/TeX",
                           "output/HTML-CSS"],
                    "messageStyle":"none",
                    "tex2jax":{
                        "processEnvironments":false,
                        "processEscapes":true,
                        "inlineMath":[["$","$"]],
                        "displayMath":[]
                    },
                    "TeX":{
                        "extensions":["AMSmath.js",
                                      "AMSsymbols.js",
                                      "noErrors.js",
                                      "noUndefined.js"]
                    },
                    "HTML-CSS":{
                        "availableFonts":["TeX"]
                    }
                });
            </script>

            <script type="text/javascript" async
                src="https://cdnjs.cloudflare.com/ajax/libs/mathjax/2.7.5/MathJax.js">
            </script>
'''

HTML_PROLOGUE = f'''<!DOCTYPE  html>
    <HTML lang="en">
        <HEAD>
            <meta charset="utf-8">
            <TITLE>Categories in BibTeX</TITLE>
            {MATHJAX}
        </HEAD>'''

HTML_EPILOGUE = '</HTML>'

BIB_EXAMPLE_FILENAME = "exemplo-utf8.bib"

OUTPUT_FILENAME = 'output.html'

def solve(author_name,INPUT_FILENAME=BIB_EXAMPLE_FILENAME):
    html_str_ls = [HTML_PROLOGUE]
    bib_str = get_bib_str(INPUT_FILENAME)

    entries = get_entries(bib_str)
    format_authors(entries)
    fix_repeated_authors(entries)

    html_str_ls.append(
        html_enclose('body',
            get_html_pub_type_counts(entries)
            + get_html_common_pub_author(author_name,entries)
            + get_html_pub_type_index(entries)
            + get_html_author_index(entries)))

    html_str_ls.append(HTML_EPILOGUE)
    with open(OUTPUT_FILENAME,'w') as file:
        file.write('\n'.join(html_str_ls))

def get_bib_str(filename):
    with open(filename,'r') as file:
        return file.read()

def get_entries(string):
    d = {}

    RELEVANT_FIELDS = {'author','title'}

    SPECIAL_TYPES = {'comment','string','preamble'}

    between_cbrace_ex = r'(?:(?<rec>{(?<value>[^{}]+|(?P>rec))*+}))'

    field_match = re.compile(rf'(?<name>\w+)\s*=\s*({between_cbrace_ex}|"(?<value>[^"]+)"|(?<value>\d+))')
    entry_match = re.compile(rf'@(?<type>\w+)(?<value>{between_cbrace_ex})')
    key_match = re.compile(r'([^{},~#%\\]+),')

    for entry in entry_match.finditer(string):
        entry_type = entry['type'].lower()
        if entry_type not in SPECIAL_TYPES:
            key = key_match.search(entry['value'])[1]
            d[entry_type, key] = {
                field['name'].lower(): field['value']
                for field in field_match.finditer(entry['value'])
                if field['name'].lower() in RELEVANT_FIELDS
            }


def unbrace(expression):
```

```
    string_ls = []
    is_between_dollar_sign = False
    is_previous_backslash = False
    for c in expression:
        if c == '$' and not is_previous_backslash:
            if is_between_dollar_sign:
                is_between_dollar_sign = False
            else:
                is_between_dollar_sign = True
        if c == '\\':
            is_previous_backslash = True
        else:
            is_previous_backslash = False
        if c in '{}' and not is_between_dollar_sign:
            continue
        string_ls.append(c)
    return ''.join(string_ls)

def html_enclose(tag,string):
    return rf'<{tag.upper()}>{string}</{tag.upper()}>'

def html_create_span(expression):
    return html_enclose('span',expression)

def html_add_attr(attr,val,html_expression):
    return re.sub(r'<(\w+)([^>]*)\s*>(.*)</\1>',rf'<\1\2 {attr.upper()}="{val}">\3</\1>',html_expression)

def mult_replace(string, replacement_list):
    for old, new in replacement_list:
        string = re.sub(old, new, string)
    return string

def format_authors(data):
    for d in data.values():
        if "author" in d:
            author_lst = [ remove_consecutive_spaces(
                           str.strip(
                           invert_name(
                           unbrace(
                           remove_accents(name)))))
                           for name in re.split(r"\band\b", d["author"].replace("\n", " "))]
            d['author'] = [author for author in author_lst if author]

def remove_consecutive_spaces(name):
    return re.sub(r'\s+',' ',name)

def invert_name(author_name):
    return re.sub(r"([^,]+),\s*([^,]+)", r"\2 \1", author_name)

def remove_accents(name):
    return remove_latex_accent(remove_normal_accent(name))

def remove_latex_accent(name):
    return re.sub(r'\\W','',name)

def remove_normal_accent(name):
    import unicodedata
    return ''.join((c for c in unicodedata.normalize('NFD', name) if unicodedata.category(c) != 'Mn'))

def fix_repeated_authors(data):
    author_blocks = fix_block_func(
                        block_authors_with_two_common_names_v2(
                            get_author_list(data)))

    author_dict = {author_name:max(s,key=len)
                   for s in author_blocks
                   for author_name in s}

    for d in data.values():
        d['author'] = [author_dict[author]
                       for author in d['author']]

def get_author_list(data):
    return sorted(set([a for s in data.values() for a in s.get("author", [])]))

def block_authors_with_two_common_names_v2(authors):
    res = set()
    for author in authors:
        fs = set()
        for author2 in authors:
            a1 = set(re.findall(r'\w\w+',author))
            a2 = set(re.findall(r'\w\w+',author2))

            if len(a1.intersection(a2)) > 1:
                fs.add(author2)

            elif len(a1) == 1 and len(a1.intersection(a2)) == 1 and is_a_first_last_match(author,author2):
                fs.add(author2)
        res.add(frozenset(fs))
    return res

def get_crude_abbrev(name):
    return ''.join(c for c in name if c.isupper())

def is_a_first_last_match(author1,author2):
    a1 = get_crude_abbrev(author1)
    a2 = get_crude_abbrev(author2)
    return a1[0] == a2[0] and a1[-1] == a2[-1]

    func(data):
    res = set()
    for g in data:
```

```
        for s2 in data:
            if q.intersection(s2) != set():
                q = q.union(s2)
        res.add(frozenset(q))
    return res

def get_pub_type_counts(data):
    pub_types_occur = [x[0] for x in data.keys()]
    pub_types = set(pub_types_occur)
    return [(pub_type, pub_types_occur.count(pub_type)) for pub_type in pub_types]

def get_html_pub_type_counts(data):
    string_ls = [html_enclose('h2',
                              'Number of Occurrences of Publication Types')]
    pub_counts = sorted(
        get_pub_type_counts(data),
        key=lambda x: x[1],reverse=True)

    time = lambda v: 's' if v > 1 else ''
    for pub_type, count in pub_counts:
        string_ls.append(
            html_enclose('p',
                         f'Type {pub_type} appears {count} time{time(count)}'
            )
        )

    return ''.join(string_ls)

def get_author_pub_graph(author,data):
    pub_partners = []
    for entry in data.values():
        if 'author' in entry and author in entry['author']:
            for partner in entry['author']:
                if partner != author:
                    pub_partners.append(partner)
    return [(author_name,pub_partners.count(author_name))
            for author_name in set(pub_partners)]

def get_dot_graph(author,data):
    import textwrap
    q = sorted(get_author_pub_graph(author,data),key = lambda x: x[1])
    string_ls = ['graph{']
    string_ls2 = []
    for partner_author,no_joint_pub in q[-3:]:
        string_ls2.append(f'"{author}" -- "{partner_author}" [label="{no_joint_pub}"]')
    string_ls.append(textwrap.indent('\n'.join(string_ls2),'  '))
    string_ls.append('}')
    return '\n'.join(string_ls)

def get_html_dot_svg(author,data):
    import os
    DOT_INPUT_FILENAME = 'dot_input'
    with open(DOT_INPUT_FILENAME,'w') as file:
        file.write(get_dot_graph(author,data))
    os.system(f'dot -T svg -O {DOT_INPUT_FILENAME}')
    with open(DOT_INPUT_FILENAME + '.svg','r') as file:
        return re.search(r'<svg(?:.|\n)+</svg>',file.read()).group()

def get_html_common_pub_author(author,data):
    string_ls = [html_enclose('h2','Author Graph')]
    string_ls.append(get_html_dot_svg(author,data))
    return ''.join(string_ls)

def fix_title(title):
    substitutions = [(r'\\textsc{((?:\\{|[^{])+)}',
                      lambda m: f'{html_to_small_caps(html_create_span(m.group(1)))}'),
                     (r'\\textsf{((?:\\{|[^{])+)}',
                      lambda m: f'{html_to_sans_serif(html_create_span(m.group(1)))}'),
                     (r'(\$(?:.|\\\$)+\$)',
                      lambda m: f'{str_to_html_math(m.group(1))}')]


    replace = lambda x: mult_replace(x,substitutions)

    return   html_create_span(
             single_quote_latex(
             double_quote_latex(
             unbrace(
             replace(
             remove_latex_special_chars(
             ' '.join(s.strip() for s in title.split('\n'))))))))

def double_quote_latex(expression):
    return re.sub(r"``(.*[^\\])''",r'"\1"',expression)

def single_quote_latex (expression):
    return re.sub(r"[^`]`([^`].*[^'])'[^`]",r"'\1'",expression)

def remove_latex_special_chars(latex_expression):
    return re.sub(r'\\({|[^{])\b',r'\1',latex_expression)

def html_to_small_caps(html_expression):
    return html_add_attr('style','font-variant:small-caps',html_expression)

def str_to_html_small_caps(expression):
    return html_to_small_caps(html_create_span(expression))

def html_to_sans_serif(html_expression):
    return html_add_attr('style','font-family:sans-serif',html_expression)

def str_to_html_math(string):
    return html_add_attr('class','math inline',html_create_span(string))
```

```
def get_html_pub_type_index(data):
    string_ls = [html_enclose('h2','Publication Type Index')]
    for entry_type in sorted(set(x[0] for x in data)):
        string_ls.append(html_enclose('h3',entry_type))
        for citation_key in [x[1] for x in data if x[0]==entry_type]:
            title = data[entry_type,citation_key].get('title','')
            authors = ', '.join((sorted(data[entry_type,citation_key].get('author',''))))
            string_ls.append(
                html_enclose('p',
                             f"Key = {citation_key}<br>Title = {fix_title(title)}<br>Autores = {authors}"))

    return '\n'.join(string_ls)

def last_name_first(name):
    initials =  '. '.join(get_crude_abbrev(name))[:-2]
    last_name = name.split()[-1]
    return f'{last_name}, {initials}'

def get_author_index_dict(data):
    index = {}
    for key, e in data.items():
        if 'author' in e:
            for author in e['author']:
                author_name = last_name_first(author)
                if author_name not in index:
                    index[author_name] = set()
                index[author_name].add(key[1])
    return index

def get_html_author_index(data):
    index = sorted(get_author_index_dict(data).items())
    alphabet_order = sorted(set(c[0][0] for c in index))
    string_ls = [html_enclose('h2','Author Index')]
    i = 0
    string_ls.append(html_enclose('h3',alphabet_order[i]))
    for author,citation_keys in index:
        if author[0] != alphabet_order[i]:
            i += 1
            string_ls.append(html_enclose('h3',alphabet_order[i]))
        citation_keys_str = ', '.join(citation_keys)
        string_ls.append(html_enclose('p',f'{author}, {citation_keys_str}'))
    return ''.join(string_ls)

if __name__ == '__main__':
    import sys
    import os.path

    filename = sys.argv[1] if len(sys.argv) > 1 else BIB_EXAMPLE_FILENAME
    assert os.path.isfile(filename)

    if len(sys.argv) < 3:
        author_name = 'Daniela da Cruz'
    else:
        author_name = sys.argv[2]
    solve(author_name,filename)
```

Dot Output

```
graph{
  "Daniela da Cruz" -- "Nuno Oliveira" [label="11"]
  "Daniela da Cruz" -- "Maria Joao Varanda Pereira" [label="25"]
  "Daniela da Cruz" -- "Pedro Rangel Henriques" [label="39"]
}
```

HTML Output

```
<!DOCTYPE  html>
    <HTML lang="en">
        <HEAD>
            <meta charset="utf-8">
            <TITLE>Categories in BibTeX</TITLE>

            <script type="text/x-mathjax-config">
                MathJax.Hub.Config({
                    "extensions":["tex2jax.js"],
                    "jax":["input/TeX",
                           "output/HTML-CSS"],
                    "messageStyle":"none",
                    "tex2jax":{
                        "processEnvironments":false,
                        "processEscapes":true,
                        "inlineMath":[["$","$"]],
                        "displayMath":[]
                    },
                    "TeX":{
                        "extensions":["AMSmath.js",
                                      "AMSsymbols.js",
                                      "noErrors.js",
                                      "noUndefined.js"]
                    },
```

```
                "HTML-CSS":{
                    "availableFonts":["TeX"]
                }
            });
        </script>

        <script type="text/javascript" async
            src="https://cdnjs.cloudflare.com/ajax/libs/mathjax/2.7.5/MathJax.js">
        </script>

    </HEAD>
<BODY><H2>Number of Occurrences of Publication Types</H2><P>Type inproceedings appears 112 times</P><P>Type article appear
 viewBox="0.00 0.00 577.59 131.00" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink">
<g id="graph0" class="graph" transform="scale(1 1) rotate(0) translate(4 127)">
<polygon fill="white" stroke="transparent" points="-4,4 -4,-127 573.59,-127 573.59,4 -4,4"/>
<!-- Daniela da Cruz -->
<g id="node1" class="node">
<title>Daniela da Cruz</title>
<ellipse fill="none" stroke="black" cx="249.74" cy="-105" rx="68.49" ry="18"/>
<text text-anchor="middle" x="249.74" y="-101.3" font-family="Times,serif" font-size="14.00">Daniela da Cruz</text>
</g>
<!-- Nuno Oliveira -->
<g id="node2" class="node">
<title>Nuno Oliveira</title>
<ellipse fill="none" stroke="black" cx="61.74" cy="-18" rx="61.99" ry="18"/>
<text text-anchor="middle" x="61.74" y="-14.3" font-family="Times,serif" font-size="14.00">Nuno Oliveira</text>
</g>
<!-- Daniela da Cruz&#45;&#45;Nuno Oliveira -->
<g id="edge1" class="edge">
<title>Daniela da Cruz&#45;&#45;Nuno Oliveira</title>
<path fill="none" stroke="black" d="M217,-89.19C182.39,-73.55 128.15,-49.02 93.8,-33.49"/>
<text text-anchor="middle" x="174.74" y="-57.8" font-family="Times,serif" font-size="14.00">11</text>
</g>
<!-- Maria Joao Varanda Pereira -->
<g id="node3" class="node">
<title>Maria Joao Varanda Pereira</title>
<ellipse fill="none" stroke="black" cx="249.74" cy="-18" rx="108.58" ry="18"/>
<text text-anchor="middle" x="249.74" y="-14.3" font-family="Times,serif" font-size="14.00">Maria Joao Varanda Pereira</te
</g>
<!-- Daniela da Cruz&#45;&#45;Maria Joao Varanda Pereira -->
<g id="edge2" class="edge">
<title>Daniela da Cruz&#45;&#45;Maria Joao Varanda Pereira</title>
<path fill="none" stroke="black" d="M249.74,-86.8C249.74,-72.05 249.74,-50.92 249.74,-36.18"/>
<text text-anchor="middle" x="256.74" y="-57.8" font-family="Times,serif" font-size="14.00">25</text>
</g>
<!-- Pedro Rangel Henriques -->
<g id="node4" class="node">
<title>Pedro Rangel Henriques</title>
<ellipse fill="none" stroke="black" cx="472.74" cy="-18" rx="96.68" ry="18"/>
<text text-anchor="middle" x="472.74" y="-14.3" font-family="Times,serif" font-size="14.00">Pedro Rangel Henriques</text>
</g>
<!-- Daniela da Cruz&#45;&#45;Pedro Rangel Henriques -->
<g id="edge3" class="edge">
<title>Daniela da Cruz&#45;&#45;Pedro Rangel Henriques</title>
<path fill="none" stroke="black" d="M286.82,-89.87C326.95,-74.57 390.74,-50.26 432.18,-34.46"/>
<text text-anchor="middle" x="381.74" y="-57.8" font-family="Times,serif" font-size="14.00">39</text>
</g>
</g>
</svg><H2>Publication Type Index</H2>
<H3>article</H3>
<P>Key = jj96<br>Title = <SPAN>NLlex -- a tool to generate lexical analysers for natural language</SPAN><br>Autores = Jose
<P>Key = Ramalho98<br>Title = <SPAN>Algebraic specification of documents</SPAN><br>Autores = Jose Carlos Ramalho, Jose Joa
<P>Key = RRAH99<br>Title = <SPAN>SGML documents: Where does quality go?</SPAN><br>Autores = Jorge Gustavo Rocha, Jose Carl
<P>Key = speaker:sepln2001<br>Title = <SPAN>Text to speech -- a rewriting system approach</SPAN><br>Autores = Alberto Manu
<P>Key = parguess2002<br>Title = <SPAN>Grabbing parallel corpora from the web</SPAN><br>Autores = Alberto Manuel Brandao S
<P>Key = sepln2003<br>Title = <SPAN>NATools -- A Statistical Word Aligner Workbench</SPAN><br>Autores = Alberto Manuel Bra
<P>Key = xmldt2<br>Title = <SPAN>- Down-Translating XML</SPAN><br>Autores = Alberto Manuel Brandao Simoes</P>
<P>Key = sepln2004<br>Title = <SPAN>Distributed Translation Memories implementation using WebServices</SPAN><br>Autores =
<P>Key = sepln06<br>Title = <SPAN>A Client-Server Architecture for building Parallel Corpora applications</SPAN><br>Autore
<P>Key = KMHVZ04<br>Title = <SPAN>Grammatical Approach to Problem Solving</SPAN><br>Autores = Maria Joao Varanda Pereira,
<P>Key = HVMLGW05<br>Title = <SPAN>Automatic Generation of Language-based Tools using LISA System</SPAN><br>Autores = Hui
<P>Key = RMHV06<br>Title = <SPAN>AspectLISA: an aspect-oriented compiler construction system based on attribute grammars</
<P>Key = RMHCV06<br>Title = <SPAN>Specifying Languages using aspect-oriented approach: AspectLISA</SPAN><br>Autores = Dami
<P>Key = GDH06<br>Title = <SPAN>AG-based interactive system to retrieve information from XML documents</SPAN><br>Autores =
<P>Key = BH98<br>Title = <SPAN>A Framework and Patterns for the   Specification of Reactive Systems</SPAN><br>Autores = Le
<P>Key = RAH98<br>Title = <SPAN>Algebraic Specification of Documents</SPAN><br>Autores = Jose Carlos Ramalho, Jose Joao Di
<P>Key = RARH98<br>Title = <SPAN>SGML Documents: Where does quality go?</SPAN><br>Autores = Jorge Gustavo Rocha, Jose Carl
<P>Key = GRH06<br>Title = <SPAN>Metamorphosis - A Topic Maps Based Environment to Handle Heterogeneous Information Resourc
<P>Key = JGRH04<br>Title = <SPAN>XCSL Tutorial</SPAN><br>Autores = Giovani Rubert Librelotto, Jose Carlos Ramalho, Marta J
<P>Key = JGRH03<br>Title = <SPAN>XCSL: XML Constraint Specification Language</SPAN><br>Autores = Giovani Rubert Librelotto
<P>Key = GRH04<br>Title = <SPAN>TM-Builder: An Ontology Builder based on XML Topic Maps</SPAN><br>Autores = Giovani Rubert
<P>Key = GRH05a<br>Title = <SPAN>Geração automática de interfaces Web para Sistemas de Informação: Metamorphosis</SPAN><br
<P>Key = RH98a<br>Title = <SPAN>Qualidade na Publicação Electrónica: como controlá-la?</SPAN><br>Autores = Jose Carlos Ram
<P>Key = MSH05<br>Title = <SPAN>Utilizando uma Base de Dados XML Nativa aplicada ao tratamento de erros num sistema de loc
<P>Key = ALHF02<br>Title = <SPAN>O Uso da Linguagem RS em Robótica</SPAN><br>Autores = Giovani Rubert Librelotto, Gustavo
<P>Key = CHV08ja<br>Title = <SPAN>Alma versus DDD</SPAN><br>Autores = Daniela da Cruz, Maria Joao Varanda Pereira, Pedro R
<P>Key = FPCH08jb<br>Title = <SPAN>Language in a Model-Based Engineering Environment for Control Systems -- An Approach fo
<P>Key = PMCH08j<br>Title = <SPAN>Program Comprehension for Domain-Specific Languages (invited paper)</SPAN><br>Autores =
<P>Key = CHV07<br>Title = <SPAN>Constructing program animations using a pattern-based approach</SPAN><br>Autores = Daniela
<P>Key = LARH09<br>Title = <SPAN>Topic Maps Constraint Languages: understanding and comparing</SPAN><br>Autores = Giovani
<P>Key = CBHP09<br>Title = <SPAN>Code Inspection Approaches for Program Visualization</SPAN><br>Autores = Daniela da Cruz,
<P>Key = OPHCC2010<br>Title = <SPAN>VisualLISA: A Visual Environment to Develop Attribute Grammars</SPAN><br>Autores = Bas
<P>Key = KOMPCCH2010<br>Title = <SPAN>Comparing General-Purpose and Domain-Specific Languages: An Empirical Study</SPAN><b
<H3>book</H3>
<P>Key = RH02<br>Title = <SPAN>XML \& XSL: da teoria à prática</SPAN><br>Autores = Jose Carlos Ramalho, Pedro Rangel Henri
<H3>incollection</H3>
<P>Key = avalon:ispell<br>Title = <SPAN>nas Morfolimpíadas</SPAN><br>Autores = Alberto Manuel Brandao Simoes, Jose Joao Di
<P>Key = avalon:avalinha<br>Title = <SPAN>Avaliação de alinhadores</SPAN><br>Autores = Alberto Manuel Brandao Simoes, Jose
:rena<br>Title = <SPAN>- Reconhecedor de Entidades</SPAN><br>Autores = Jose Joao Dias de Almeida</P>
<br>Title = <SPAN>Data Reduction to Improve Knowledge Extraction</SPAN><br>Autores = Carlos Ramos, Maria de
<P>Key = URH06<br>Title = <SPAN>Data Cleaning by Reusing Domain Knowledge</SPAN><br>Autores = Maria de Fatima Rodrigues, P
<H3>inproceedings</H3>
```

<P>Key = graminteractivas1990
Title = <SPAN>Mecanismos para Especificação e Prototipagem de Interfaces Utilizador-Siste
<P>Key = Almeida94b
Title = <SPAN>GPC -- a Tool for higher-order grammar specification</SPAN>
Autores = Jose Joao Di
<P>Key = Almeida95a
Title = <SPAN>YaLG -- extending DCG for natural language processing</SPAN>
Autores = Jose Joao D
<P>Key = Almeida94c
Title = <SPAN>Jspell -- um módulo para análise léxica genérica de linguagem natural</SPAN>
Autor
<P>Key = Ramalho95
Title = <SPAN>Algebraic Specification of Documents</SPAN>
Autores = Jose Carlos Ramalho, Jose Joa
<P>Key = Almeida96a
Title = <SPAN>Especificação e tratamento de Dicionários</SPAN>
Autores = Jose Joao Dias de Almei
<P>Key = Ulisses96
Title = <SPAN>Tratamento automático de termos compostos</SPAN>
Autores = Jose Joao Dias de Almeid
<P>Key = Almeida96b
Title = <SPAN>YaLG a tool for higher-order grammar specification</SPAN>
Autores = J.B. Barros, J
<P>Key = Ramalho96
Title = <SPAN>Document Semantics: two approaches</SPAN>
Autores = Jose Carlos Ramalho, Jose Joao
<P>Key = SGML97
Title = <SPAN>SGML Documents: where does quality go?</SPAN>
Autores = Jorge Gustavo Rocha, Jose Carl
<P>Key = Almeida98
Title = <SPAN>Programação de dicionários</SPAN>
Autores = Jose Joao Dias de Almeida</P>
<P>Key = Reis98
Title = <SPAN>Etiquetador morfo-sintáctico para o Português</SPAN>
Autores = Jose Joao Dias de Almei
<P>Key = ABNO97a
Title = <SPAN><SPAN STYLE="font-variant:small-caps">Camila</SPAN>: Formal Software Engineering Support
<P>Key = ABNO97b
Title = <SPAN><SPAN STYLE="font-variant:small-caps">Camila</SPAN>: Prototyping and Refinement of Const
<P>Key = AH97
Title = <SPAN>Dynamic Dictionary = cooperative information sources</SPAN>
Autores = Jose Joao Dias de
<P>Key = museums98
Title = <SPAN>Adapting Museum Structures for the Web: No Changes Needed!</SPAN>
Autores = J.L. Fa
<P>Key = ABBN98
Title = <SPAN>On The Development of <SPAN STYLE="font-variant:small-caps">Camila</SPAN></SPAN>
Autor
<P>Key = Gis99
Title = <SPAN>Systems Development</SPAN>
Autores = Ana Silva, Jorge Gustavo Rocha, Jose Joao Dias de
<P>Key = RPA99
Title = <SPAN>Maps</SPAN>
Autores = Jorge Gustavo Rocha, Jose Joao Dias de Almeida, Tiago Pedroso</P>
<P>Key = RSea99
Title = <SPAN>SIG</SPAN>
Autores = Ana Silva, Jorge Gustavo Rocha, Jose Joao Dias de Almeida, Mario
<P>Key = xmldt99
Title = <SPAN>a Perl Down-Translation module</SPAN>
Autores = Jose Carlos Ramalho, Jose Joao Dias d
<P>Key = Barbosa2000
Title = <SPAN>Polytypic Recursion Patterns</SPAN>
Autores = J.B. Barros, Jose Joao Dias de Alme
<P>Key = jj2001x
Title = <SPAN>Smallbook -- comando para produção de livros em pequena escala</SPAN>
Autores = Jose
<P>Key = mp2001
Title = <SPAN>-- Arquitectura</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jorge Gustavo Rocha, J
<P>Key = alfarrabio2001
Title = <SPAN>Alfarrábio: Adding value to an Heterogeneous Site Collection</SPAN>
Autores =
<P>Key = freq2002
Title = <SPAN>Cálculo de frequências de palavras para entradas de dicionários através do uso conjunto
<P>Key = jspell2002
Title = <SPAN>Jspell.pm -- um módulo de análise morfológica para uso em processamento de linguagem
<P>Key = dag2002
Title = <SPAN>Directory Attribute Grammars</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jose Joa
<P>Key = elpub2002
Title = <SPAN>Library::* -- a toolkit for digital libraries</SPAN>
Autores = Alberto Manuel Brand
<P>Key = APL2k2.Parquess
Title = <SPAN>Extracção de corpora paralelo a partir da web: construção e disponibilização</SP
<P>Key = APL2k2.Synthesis
Title = <SPAN>Geração de voz com sotaque</SPAN>
Autores = Alberto Manuel Brandao Simoes, J
<P>Key = xata:xmldt
Title = <SPAN>Engenharia reversa de HTML usando tecnologia XML</SPAN>
Autores = Alberto Manuel B
<P>Key = xata:museudapessoa
Title = <SPAN>essoa</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jose Joao Dias de Al
<P>Key = elpub2003
Title = <SPAN>Music publishing</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jose Joao Dias de
<P>Key = cp3a:terminum2003
Title = <SPAN>Projecto TerminUM</SPAN>
Autores = Alberto Manuel Brandao Simoes, Bruno Mar
<P>Key = cp3a:kvec2003
Title = <SPAN>Lingua-Biterm: um módulo Perl para extracção de terminologia bilingue</SPAN>
Au
<P>Key = cp3a:natools2003
Title = <SPAN>Alinhamento de corpora paralelos</SPAN>
Autores = Alberto Manuel Brandao Sim
<P>Key = xata04:tx
Title = <SPAN>baseada em tipos dinâmicos</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jose Joa
<P>Key = xata04:mtd
Title = <SPAN>Memórias de Tradução Distribuídas</SPAN>
Autores = Alberto Manuel Brandao Simoes,
<P>Key = linguateca
Title = <SPAN>Linguateca: um centro de recursos distribuído para o processamento computacional da l
<P>Key = xata05:fs
Title = <SPAN>Representação em XML da Floresta Sintáctica</SPAN>
Autores = Alberto Manuel Brandac
<P>Key = xata05:tdt
Title = <SPAN>Inferência de tipos em documentos XML</SPAN>
Autores = Alberto Manuel Brandao Simc
<P>Key = xata06:navegante
Title = <SPAN>Navegante: um proxy de ordem superior para navegação intusiva</SPAN>
Autores
<P>Key = xata06:xmlauto
Title = <SPAN>XML</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jose Joao Dias de Almeida<
<P>Key = eamt06
Title = <SPAN>Combinatory Examples Extraction for Machine Translation</SPAN>
Autores = Alberto Manue
<P>Key = lrec06
Title = <SPAN>--- Recycling Thesauri into a Multilingual Ontology</SPAN>
Autores = Alberto Manuel Br
<P>Key = elpub06-t2o
Title = <SPAN>Publishing multilingual ontologies: a quick way of obtaining feedback</SPAN>
Autc
<P>Key = elpub06-blind
Title = <SPAN>Transcoding for Web Accessibility for the Blind: Semantics from Structure</SPAN><b
<P>Key = xata07:xmltmx
Title = <SPAN>--- Processamento de Memórias de Tradução de Grandes Dimensões</SPAN>
Autores =
<P>Key = MP07
Title = <SPAN>Dependency Specification Language</SPAN>
Autores = Alberto Manuel Brandao Simoes, Jose J
<P>Key = epia-bio-2007
Title = <SPAN>An Ontology-Based Approach To Systems Biology Literature Retrieval and Processing<
<P>Key = epia-music-2007
Title = <SPAN>Using Text Mining Techniques for Classical Music Scores Analysis</SPAN>
Autor
<P>Key = HKMVZ03
Title = <SPAN>Grammatical Approach to Problem Solving</SPAN>
Autores = Maria Joao Varanda Pereira,
<P>Key = VH01
Title = <SPAN>Visualization / Animation of Programs based on Abstract Representations and Formal Mappings
<P>Key = VH02
Title = <SPAN>Automatic Generation of Language-based Tools</SPAN>
Autores = Maria Joao Varanda Pereira
<P>Key = VH03
Title = <SPAN>Visualization / Animation of Programs in Alma: obtaining different results</SPAN>
Autore
<P>Key = RMHVC06
Title = <SPAN>Specifying Languages using Aspect-oriented Approach: AspectLISA</SPAN>
Autores = Dami
<P>Key = BHVU07d
Title = <SPAN>PICS una Herramienta para la Comprensión e Inspección de Programas</SPAN>
Autores = M
<P>Key = BHVU07c
Title = <SPAN>Program Inspection to Incerconnect Behavioral and Operational View for Program Compreher
<P>Key = BHVU07b
Title = <SPAN>Comprensi'on de Programas por Inspecci'on Visual y Animaci'on</SPAN>
Autores = Maria
<P>Key = BHVU07a
Title = <SPAN>Static and Dynamic Strategies to Understand C Programs by Code Annotation</SPAN>
Autc
<P>Key = CHLB07a
Title = <SPAN>O Sitio de Pico, Software Educativo para Crianças con Paralisia Cerebral</SPAN>
Autor
<P>Key = BHVU06a
Title = <SPAN>Herramientas para la compresión de programas</SPAN>
Autores = Maria Joao Varanda Pere
<P>Key = BHVU06b
Title = <SPAN>Comprensión de Algoritmos de Ruteo</SPAN>
Autores = Maria Joao Varanda Pereira, Mario
<P>Key = BHVUM06
Title = <SPAN>A Language Processing Tool for Program Comprehension</SPAN>
Autores = G. Montejano, M
<P>Key = BHVU08
Title = <SPAN>Simplificando la Comprensión de Programas a través de la Interconnexión de Dominios</SPAN
<P>Key = BHV06
Title = <SPAN>A System for Evaluate and Understand Routing Algorithms</SPAN>
Autores = Maria Joao Var
<P>Key = BCVHU08
Title = <SPAN>Evaluation Criteria of Software Visualization Systems used for Program Comprehension</SP
<P>Key = BUHV08
Title = <SPAN>Inspección de Código para relacionar los Dominios del Problema y Programa para la Compren
<P>Key = OVH05
Title = <SPAN>Compreensão de Aplicações Web: O Processo e as Ferramentas</SPAN>
Autores = Eva Oliveir
<P>Key = OHV06
Title = <SPAN>Proposta de um Sistema para Compreensão de Aplicações Web</SPAN>
Autores = Eva Oliveira
<P>Key = GH07b
Title = <SPAN>Analyzing the structure of scientific articles to improve information retrieval</SPAN>
<P>Key = GH07a
Title = <SPAN>Using data together with metadata to improve XML information access</SPAN>
Autores = Al
<P>Key = GFH08
Title = <SPAN>Using data together with metadata to improve XML information access</SPAN>
Autores = Al
<P>Key = FGH08
Title = <SPAN>Information access from XML using semantics and context: application to the Portuguese Emi
<P>Key = FH08
Title = <SPAN>Using OWL to specify and build different views over the Emigration Museum resources</SPAN><
<P>Key = LPRH07
Title = <SPAN>Navegando na Rede Semântica dos Topic Maps com o Ulisses</SPAN>
Autores = Giovani Rube
<P>Key = LPRH07-TM
Title = <SPAN>Topic Maps Constraint Specification Languages: comparing AsTMa!, OSL, and XTche</SPAN>
<P>Key = LRHGT08
Title = <SPAN>A Framework to specify, extract and manage Topic Maps driven by ontologie</SPAN>
Autc
<P>Key = LGFSSAH08
Title = <SPAN>Uma Ontologia aplicada a um Ambiente Pervasivo Hospitalar</SPAN>
Autores = Fabio L.
<P>Key = LMMVRH08
Title = <SPAN>Generating a Semantic Network for PubMed</SPAN>
Autores = Giovani Rubert Librelotto,
<P>Key = CPH07f
Title = <SPAN>Pattern-based Program Visualization</SPAN>
Autores = Daniela da Cruz, Maria Joao Varar
<P>Key = CH07g
Title = <SPAN>Slicing wxHaskell modules to derive the User Interface Abstract Model (short paper and pos
<P>Key = CH07h
Title = <SPAN>Laboratory Site (poster)</SPAN>
Autores = Daniela da Cruz, Pedro Rangel Henriques</P>
<P>Key = FCHV08
Title = <SPAN>How to interconnect operational and behavioral views of web applications</SPAN>
Autore
<P>Key = CHP08i
Title = <SPAN>Strategies for Program Inspection and Visualization</SPAN>
Autores = Daniela da Cruz,
<P>Key = CH07a
Title = <SPAN>anguage</SPAN>
Autores = Daniela da Cruz, Pedro Rangel Henriques</P>
<P>Key = CLH07c
Title = <SPAN>Como ensinar com Mapas de Conceitos: duas abordagens complementares</SPAN>
Autores = D
<P>Key = CH07d
Title = <SPAN>LISS --- The language and the compiler</SPAN>
Autores = Daniela da Cruz, Pedro Rangel H
<P>Key = CFPBH07d
Title = <SPAN>Comparing Generators for Language-based Tools</SPAN>
Autores = Daniela da Cruz, Mari
<P>Key = CHP08a
Title = <SPAN>Documents</SPAN>
Autores = Daniela da Cruz, Maria Joao Varanda Pereira, Pedro Rangel H
<P>Key = CHP08b
Title = <SPAN>DDD</SPAN>
Autores = Daniela da Cruz, Maria Joao Varanda Pereira, Pedro Rangel Henriqu
<P>Key = CPH08c
Title = <SPAN>Properties Preservation during Transformation (short paper)</SPAN>
Autores = Daniela d
<P>Key = FPCH08d
Title = <SPAN>Language in a Model-Based Engineering Environment for Control Systems --- An Approach fo
<P>Key = PMCH08e
Title = <SPAN>: a Visual Interface for an Attribute Grammar based Compiler-Compiler (short paper)</SPA
<P>Key = PMCH08f
Title = <SPAN>Program Comprehension for Domain-Specific Languages</SPAN>
Autores = Daniela da Cruz,
<P>Key = CHP09a
Title = <SPAN>Code Analysis: Past and Present</SPAN>
Autores = Daniela da Cruz, Jorge Sousa Pinto, P
<P>Key = OPCH09a
Title = <SPAN>Visualization of Domain-Specific Programs' Behavior</SPAN>
Autores = Daniela da Cruz,
<P>Key = ira09b
Title = <SPAN>Applying Program Comprehension Techniques to Karel Robot Programs</SPAN>
Autores
<P>Key = ira09c
Title = <SPAN>VisualLISA: Visual Programming Environment for Attribute Grammars Specification</SPA
<P>Key = RSAF09
Title = <SPAN>Influence of domain-specific notation to program understanding</SPAN>
Autores = Danie
<P>Key = FCHGD09a
Title = <SPAN>A Query-by-Example Approach for XML Querying</SPAN>
Autores = Alda Lopes Gancarski,

<P>Key = ORH09a
Title = <SPAN>SMARTCLEAN: uma ferramenta para a limpeza incremental de dados</SPAN>
Autores = Maria
<P>Key = LPH09a
Title = <SPAN>Uma metodologia para Consultas aos Bancos de Dados do NCBI</SPAN>
Autores = Giovani Ru
<P>Key = OPHC09a
Title = <SPAN>Domain Specific Languages: A Theoretical Survey</SPAN>
Autores = Daniela da Cruz, Mar
<P>Key = OPHCC09
Title = <SPAN>: A Domain Specific Visual Language for Attribute Grammars</SPAN>
Autores = Bastian
<P>Key = MKCHCPO09
Title = <SPAN>Comparison of XAML and C\# Forms using Cognitive Dimensions Framework</SPAN>
Autore
<P>Key = OHCP09
Title = <SPAN>XAGra - An XML Dialect for Attribute Grammars</SPAN>
Autores = Daniela da Cruz, Maria
<P>Key = FCHGD09b
Title = <SPAN>GuessXQ, an inference Web-engine for querying XML Documents</SPAN>
Autores = Alda Lo
<P>Key = BHVU09
Title = <SPAN>Instrumentaciones de Programas Escritos en C para Interrelacionar las Vistas Comportament
<P>Key = CH09d
Title = <SPAN>Assessing Databases in .Net: comparing approaches</SPAN>
Autores = Daniela da Cruz, Pe
<P>Key = CH2010a
Title = <SPAN>Exploring, Visualizing and Slicing the Soul of XML Documents</SPAN>
Autores = Daniela
<H3>mastersthesis</H3>
<P>Key = teseambs
Title = <SPAN>Parallel Corpora word alignment and applications</SPAN>
Autores = Alberto Manuel Bra
<H3>misc</H3>
<P>Key = cruz09
Title = <SPAN>GraAL - A Grammar Analyzer</SPAN>
Autores = Daniela da Cruz, Nuno Oliveira, Pedro Rang
<H3>phdthesis</H3>
<P>Key = tesejj
Title = <SPAN>Dicionários dinâmicos multi-fonte</SPAN>
Autores = Jose Joao Dias de Almeida</P>
<H3>techreport</H3>
<P>Key = tlc89
Title = <SPAN>Teoria das Linguagens</SPAN>
Autores = J.B. Barros, Jose Joao Dias de Almeida</P>
<P>Key = estruturasdedados90
Title = <SPAN>Estruturas de Dados</SPAN>
Autores = J.B. Barros, Jose Joao Dias de Almei
<P>Key = Camila
Title = <SPAN><SPAN STYLE="font-variant:small-caps">Camila</SPAN> - A Platform for Software Mathematica
<P>Key = Natura
Title = <SPAN>Natura - Natural language processing</SPAN>
Autores = Jose Joao Dias de Almeida</P>
<P>Key = jspell1
Title = <SPAN>Manual de Utilizador do JSpell</SPAN>
Autores = Jose Joao Dias de Almeida, Ulisses Pi
<P>Key = jj95
Title = <SPAN>NLlex -- a tool to generate lexical analysers for natural language</SPAN>
Autores = Jose
<P>Key = Barbosa95
Title = <SPAN>System Prototyping in <SPAN STYLE="font-variant:small-caps">Camila</SPAN></SPAN>
Au
<P>Key = Barbosa95a
Title = <SPAN><SPAN STYLE="font-variant:small-caps">Camila</SPAN>: A reference Manual</SPAN>
Aut
<P>Key = BA97a
Title = <SPAN>Systems Prototyping in <SPAN STYLE="font-variant:small-caps">Camila</SPAN></SPAN>
Autor
<P>Key = Barbosa95b
Title = <SPAN>Growing Up With <SPAN STYLE="font-variant:small-caps">Camila</SPAN></SPAN>
Autores
<P>Key = Almeida96c
Title = <SPAN>From BiBTeX to HTML semantic nets</SPAN>
Autores = Jose Carlos Ramalho, Jose Joao
<H3>A</H3>
<P>Afonso, S., linguateca</P>
<P>Aires, R., linguateca</P>
<P>Almeida, J. J. D., sepln2004, jspell1, xata07:xmltmx, Barbosa95, epia-music-2007, tlc89, xata:xmldt, elpub2003, APL2k2.
<P>Arnold, G., ALHF02</P>
<P>Augustin, I., LGFSSAH08</P>
<P>Azevedo, R. P., LPRH07-TM, LARH09, LPRH07</P>
<H3>B</H3>
<P>Barbosa, L. S., ABBN98, Barbosa95, ABNO97a, Barbosa95b, BA97a, Barbosa95a, Barbosa2000, ABNO97b</P>
<P>Barreiro, A., linguateca</P>
<P>Barroca, L., BH98</P>
<P>Barros, J. B., ABBN98, tlc89, estruturasdedados90, Almeida96b, Barbosa2000</P>
<P>Beron, M., BCVHU08, BHVU08, BHVU07a, CBHP09, BHVU06b, BHVU06a, BHVU07d, BHVUM06, CFPBH07d, BHVU09, CHLB07a, BUHV08, BH
<P>Bick, E., linguateca, xata05:fs</P>
<H3>C</H3>
<P>Cabral, L., linguateca</P>
<P>Camila, , Camila</P>
<P>Cardoso, N., linguateca</P>
<P>Carvalho, A., elpub06-blind</P>
<P>Castro, J. A., cp3a:terminum2003, APL2k2.Parguess, parguess2002</P>
<P>Chaves, M., linguateca</P>
<P>Costa, L., linguateca</P>
<P>Cramer, B., OPHCC09, OPHCC2010</P>
<P>Crepinsek, M., oliveira09b, KOMPCCH2010</P>
<P>Cruz, D., kosar09, CHV07, oliveira09b, OPCH09a, FCHV08, FPCH08jb, BCVHU08, CHP08b, PMCH08j, OPHCC09, FPCH08d, CHV08ja,
<P>Cunha, E., CHLB07a</P>
<H3>D</H3>
<P>Defude, B., FCHGD09a, FCHGD09b</P>
<P>Doucet, A., GDH06</P>
<H3>F</H3>
<P>Faria, J. L., museums98</P>
<P>Fernandes, A. R., elpub06-blind</P>
<P>Ferreira, E., FPCH08jb, FPCH08d, epia-bio-2007</P>
<P>Ferreira, F. X., FCHGD09b, GFH08, FGH08, FCHGD09a, FH08</P>
<P>Fonseca, J., ALHF02</P>
<P>Fonseca, R., FCHV08, MP07, CFPBH07d</P>
<P>Frankenberg-Garcia, A. F., linguateca</P>
<P>Freitas, L. O., LGFSSAH08</P>
<H3>G</H3>
<P>Gancarski, A. L., GH07a, FCHGD09b, GDH06, FGH08, GFH08, FCHGD09a, GH07b</P>
<P>Gassen, J. B., LRHGT08, LGFSSAH08</P>
<P>Gray, J., HVMLGW05</P>
<P>Guinovart, X. G., sepln2004, xata04:mtd</P>
<H3>H</H3>
<P>Henriques, M. R., RSea99, museums98, Gis99</P>
<P>Henriques, P. R., BHVU08, VH02, LRHGT08, OPCH09a, GRH04, FCHV08, BCVHU08, CHP08b, OPHCC09, FPCH08d, ORH06, CHV08ja, BUH
<H3>J</H3>
<P>Jacinto, M., JGRH04, JGRH03</P>
<H3>K</H3>
<P>Kosar, T., kosar09, MKCHCPO09, oliveira09b, HKMVZ03, KMHVZ04, KOMPCCH2010</P>
<H3>L</H3>
<P>Lenic, M., HVMLGW05, VH02</P>
<P>Librelotto, G. R., JGRH04, JGRH03, LPRH07-TM, LGFSSAH08, ALHF02, LRHGT08, GRH06, LMMVRH08, LARH09, GRH05a, GRH04, LPRH0
<P>Lopes, S. C., CHLB07a, CLH07c</P>
<P>Lourenco, A., epia-bio-2007, epia-music-2007</P>
<H3>M</H3>
<P>Machado, H., LMMVRH08</P>
<P>Maia, B., linguateca</P>
<P>Martins, B., cp3a:terminum2003, cp3a:kvec2003</P>
<P>Martins, F. M., graminteractivas1990</P>
<P>Martins, M., LMMVRH08</P>
<P>Mendes, G., MSH05</P>
<P>Mernik, M., kosar09, PMCH08j, RMHV06, MKCHCPO09, HVMLGW05, VH02, oliveira09b, RMHVC06, RMHCV06, HKMVZ03, PMCH08f, KMHVZ
<P>Montejano, G., BHVUM06</P>
<P>Moreira, S., mp2001</P>
<P>Mota, C., linguateca</P>
<H3>N</H3>
<P>Neves, F. L., ABNO97a, ABNO97b</P>
<P>Neves, L. F., ABBN98</P>
<H3>O</H3>
<P>., linguateca</P>
<P>., OVH05, OHV06</P>
<P>Oliveira, J. N., ABNO97a, ABNO97b</P>
<P>Oliveira, N., kosar09, cruz09, MKCHCPO09, oliveira09b, OPHC09a, OPHCC09, OPCH09a, OHCP09, OPHCC2010, oliveira09c, KOMPC

```
<P>Oliveira, P., ORH06</P>
<P>Oliveria, P. J., ORH09a</P>
<H3>P</H3>
<P>Paulo, R., FPCH08jb, FPCH08d</P>
<P>Pedroso, T., RPA99</P>
<P>Pereira, M. J. V., kosar09, BHVU08, BHVU07a, VH02, CHV07, oliveira09b, BHVUM06, OPCH09a, FCHV08, BHV06, BCVHU08, CHP08b
<P>Pereira, R. T., LPH09a</P>
<P>Pinto, A., linguateca</P>
<P>Pinto, J. S., CHP09a, CPH08c</P>
<P>Pinto, U., Ulisses96, jspell1, Almeida94c</P>
<H3>R</H3>
<P>Ramalho, J. C., LRHGT08, GRH04, JGRH04, LPRH07-TM, SGML97, Ramalho96, Ramalho98, LPRH07, RH98a, RARH98, RRAH99, RAH98,
<P>Ramos, C., RRH02</P>
<P>Ranchhod, E., linguateca</P>
<P>Rebernak, D., RMHVC06, RMHCV06, RMHV06</P>
<P>Reis, R., Reis98</P>
<P>Rocha, I., epia-bio-2007</P>
<P>Rocha, J. G., museums98, RPA99, SGML97, alfarrabio2001, Gis99, RSea99, RARH98, RRAH99, mp2001</P>
<P>Rocha, M., epia-bio-2007</P>
<P>Rocha, P. A., linguateca, freq2002</P>
<P>Rodrigues, M. F., ORH09a, RRH02, ORH06</P>
<H3>S</H3>
<P>Santos, D., linguateca</P>
<P>Sarmento, L., linguateca</P>
<P>Silva, A., RSea99, Gis99</P>
<P>Silva, F. L., LGFSSAH08</P>
<P>Silva, N. A., MSH05</P>
<P>Silva, P., cp3a:terminum2003</P>
<P>Silva, R., linguateca</P>
<P>Silveira, M. C., LGFSSAH08</P>
<P>Simoes, A. M. B., sepln2004, xata07:xmltmx, teseambs, epia-music-2007, xata:xmldt, elpub2003, sepln06, APL2k2.Parguess,
<H3>T</H3>
<P>Turchetti, R. C., LRHGT08</P>
<H3>U</H3>
<P>Uzal, R., BCVHU08, BHVU08, BHVU07a, BHVU06b, BHVU06a, BHVU07d, BHVUM06, BHVU09, BUHV08, BHVU07b, BHVU07c</P>
<H3>V</H3>
<P>Vilela, R., linguateca, xata05:fs</P>
<P>Vizzotto, J., LMMVRH08</P>
<H3>W</H3>
<P>Wu, H., HVMLGW05</P>
<H3>Z</H3>
<P>Zumer, V., KMHVZ04, HKMVZ03</P></BODY>
</HTML>
```

Published from relatorio.pmd using Pweave 0.30.3 on 13-11-2021.