

Scripting no Processamento de Linguagens Naturais  
(4º ano de Curso de MiEI)  
**Trabalho Prático 1**  
Relatório de Desenvolvimento

Catarina Cardoso  
(a75037)

Paulo Guedes  
(a74411)

Pedro Cunha  
(a73958)

03/11/2017

# Conteúdo

1	Introdução	2
2	Decisões e Implementação	3
A	Código do Programa	4

# Capítulo 1

## Introdução

### Descrição

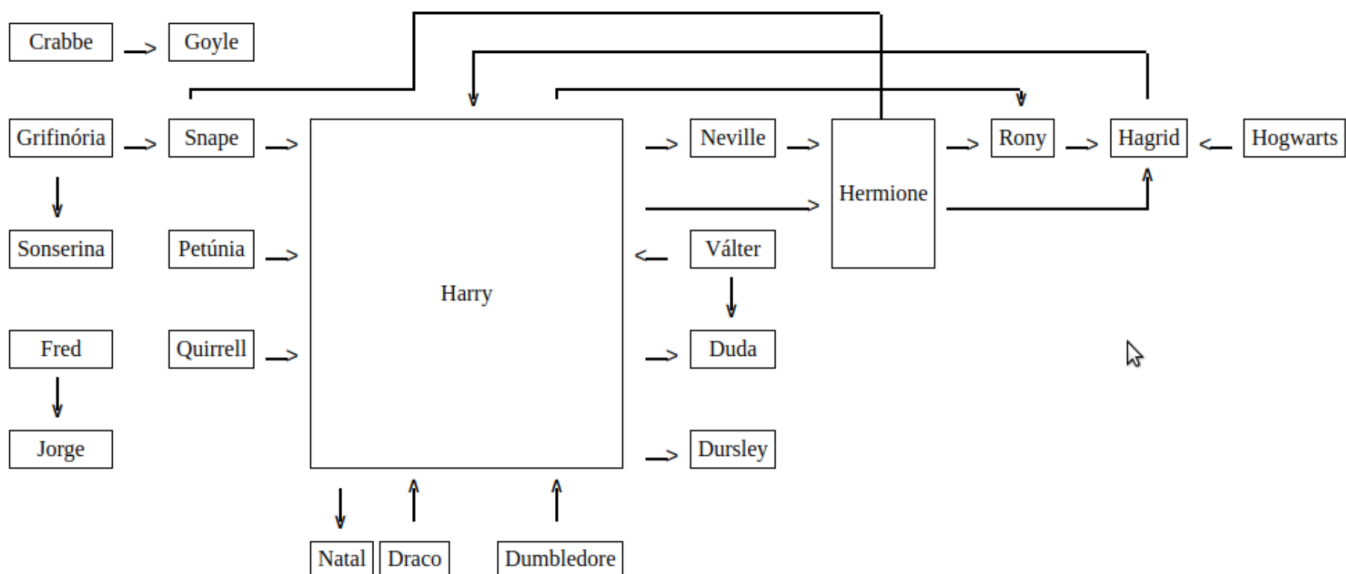
No âmbito da unidade curricular de Scripting no Processamento de Linguagens Naturais, foi proposta a resolução de um trabalho prático usando a linguagem Perl que identificasse os nomes próprios presentes num texto e os relacionasse entre si, originando assim uma tabela de relacionamentos. Nesta tabela que relaciona é possível observar o número de vezes que os nomes se encontram na vizinhança um do outro e, assim, concluir qual o grau de "simpatia" entre eles.

### Estrutura do Relatório

Após o capítulo introdutório, segue-se o capítulo 2 onde se expõe detalhadamente as decisões de implementação tomadas aquando a realização do projeto. No final, no apêndice, pode-se encontrar o ficheiro usado na realização do trabalho prático.

# Decisões e Implementação

Após o tratamento do texto e do preenchimento da tabela de Hash, os nomes próprios nela inseridos são filtrados e separados através de expressões regulares, de modo a serem inseridos no grafo (figura 2.1) que representa todas as ligações encontradas.



3

# Apêndice A

## Código do Programa

```
#!/usr/bin/perl

use Graph::Easy;
use strict;
use warnings;
use utf8::all;

my $graph = Graph::Easy->new();
my $i=25;
my %sortedHash;
my $union;
my $pessoa;
my $fPers;
my $uPers;
my $resto;

my $PM = qr{[A-ZÁÀÃÊËÚÍÓÇ][a-záàãêéúíóç]+};
my $de = qr{d[aoe]s?};
my $s = qr{[\n ]};
my $Pre = qr{Sr\. |Sra\. |Dr\. |Dra\. |Eng\. |Miss\. |Mr\. };
my $np = qr{$Pre? $PM ($s $PM|$s $de $s $PM)*}x;
my $pal = qr{[\wáàãêéúíóç]+};
my $all = qr{.*};
my $allP = qr{.*?};

while(<>){

    s/(^|[\n]|[?!.;:][' "<]|[-]|^--)( ?)($PM)/$1$2_$3/g;
    s/($Pre)(_)( $np)/_ $1_$3/g;
    s/(\b $np)/{$1}/g;
    s/(_)( $Pre)(_)( $np)/{$2$4}/g;

    while(/({($np)}($all){($np)})/g){
        my %pessoas;
        $fPers = $1;
        $uPers = $4;
        $resto = $3;
        $pessoas{$fPers}++;
    }
}
```

```

        if($resto =~ /{($np)}/) {
            while($resto =~ /($allP) \{($np)\}($all)/){
                $pessoa = $2;
                if(!exists $pessoas{$pessoa}){
                    foreach my $key (keys %pessoas){
                        verifica($key, $pessoa);
                    }
                    $pessoas{$pessoa}++;
                }
                $resto = $4;
            }
            if(!exists $pessoas{$uPers}){
                foreach my $key (keys %pessoas){
                    verifica($key, $uPers);
                }
            }
        }

        else{
            verifica($fPers, $uPers);
        }
    }
    s/{($np)}/$1/g;
    s/_//g;
}

for (sort{$sortedHash{$b} <=> $sortedHash{$a}} keys %sortedHash){
    if(/($np)-($np)/g) {
        $union = "$1-$3";
        if ($1 !~ /$3/ && $3 !~ /$1/) {
            $graph->add_edge ($1, $3);
        }
    }
    $i--;
    if ($i eq 0) {last;}
}

print $graph->as_html_file( );

sub verifica {
    my $tempV;
    my $tempV2;
    my ($p1, $p2) = @_;
    $tempV = "$p2-$p1";
    $tempV2 = "$p1-$p2";
    if ($sortedHash{$tempV}) {
        $sortedHash{$tempV}++;
    }
    else {
        $sortedHash{$tempV2}++;
    }
}

```