

## Tarefa 02

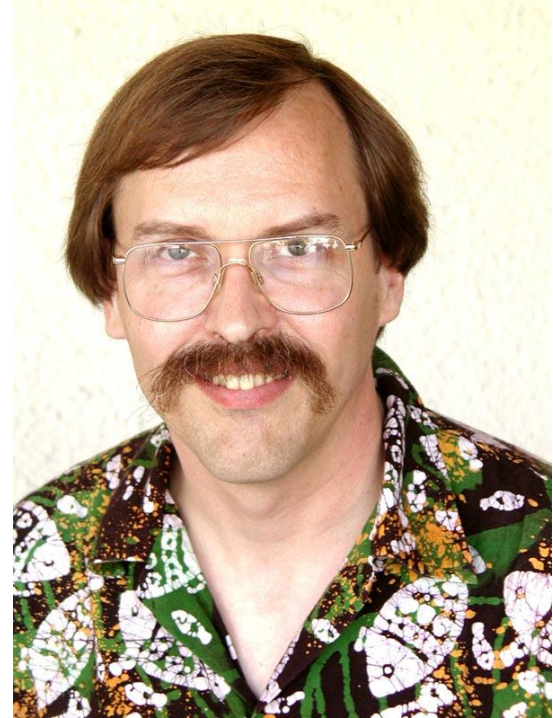
Perl

Felipe Oliveira  
Renan Mendonça

# Introdução

## Perl

- Desenvolvida por Larry Wall em 1987
- Multiparadigma
- Multiplataforma
- Tipagem dinâmica
- Interpretada

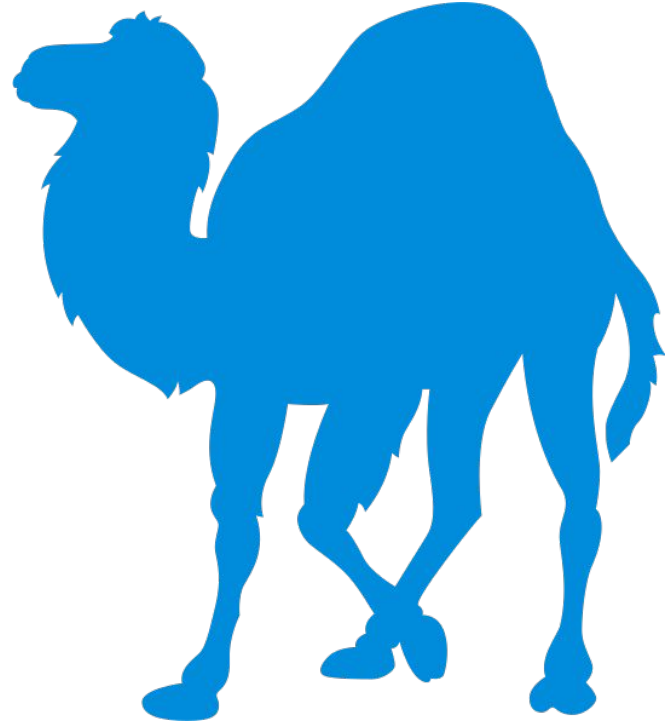


# Origens e influências

- 'sed'
  - AWK
  - C
  - Shell Script
- Ruby on Rails

# Usos

- Desenvolvimento de sites
- Computação gráfica
- Bioinformática
- Administração de sistemas
- Programação de redes
- Finança



# Características

- Strings e expressões regulares
- Contexto de variáveis
  - Escalar e Lista
- Variável padrão \$\_
- Lento
- Maior writability
- Menor readability

```
#!/usr/bin/perl -w
use strict;

# camel code

$_ = 'ev
al("seek\040D
ATA,0,
0:");foreach(1..3)
{<DATA>};my
@camellhump:my$camel;
my$Camel ;while(
<DATA>){$ =sprintf("%-6
9s",$ ).my@romedary
1=split(/:/);if(defined($
p=split(/:/);while(@romeda
ryl){my$camellhump=0
;my$CAMEL=3;if(defined($ _shif
t(@romedary1
))&&/S/){$camellhump+=1<<$CAMEL;}
$CAMEL--;if(d
efined($ _shift(@romedary1))&&/S/){
$camellhump+=1
<<$CAMEL;}$CAMEL--;if(defined($ _shift(
@camellhump))&&/S/){$camellhump+=1<<$CAMEL;}$CAMEL--;if(
defined($ _shift(@camellhump))&&/S/){$camellhump+=1<<$CAME
L;}$Camel.= (split(/,/,"040..m {/J\047\134\L~7FX")){$camellh
ump;}$Camel.="n";@camellhump=split(/n/,$Camel);foreach(@
camellhump){chomp;$Camel=$_.y/LJF7\173\175\047\061\062\063\
064\065\066\067\070/;y/12345678/JL7F\175\173\047 /;$ _=reverse;
print"$ \040$Camel\n";}foreach(@camellhump){chomp;$Camel=$_.y
/LJF7\173\175 \047\12345678/;y/12345678/JL7F\175\173\0 47 /;
$_=reverse;print"\040$ _$Camel\n";}};s/\s+//g;;eval; eval
("seek\040DATA,0,0:");undef$$_;$ _=<DATA>;s/\s+//g;(
);;s
;".*";map{eval"print"$ _".";}/./;4}/g; DATA \124
\1
50\145\040\165\163\145\040\157\1
40\143\141 \155\145\1 54\040\1 51\155\ 141
\147\145\0 40\151\156 \040\141 \163\16 3\
157\143\ 151\141\16 4\151\1 57\156
\040\167 \151\164\1 50\040\ 120\1
45\162\ 154\040\15 1\163\ 040\14
1\040\1 64\162\1 41\144 \145\
155\14 1\162\ 153\04 0\157
\146\ 040\11 7\047\ 122\1
45\15 1\154\1 54\171 \040
\046\ 012\101\16 3\16
3\15 7\143\15 1\14
1\16 4\145\163 \054
\040 \111\156\14 3\056
\040\ 125\163\145\14 4\040\
167\1 51\164\1 50\0 40\160\
145\162 \155\151
\163\163 \151\1
57\156\056
```

# Strings e expressões regulares

## Contexto de variáveis Escalar e Lista

```
1  @nomes = ('Rojas', 'Eustáquio', 'Magela');
2
3  @copia = @nomes;
4  $tamanho = @nomes;
5
6  print "Os nomes são: @copia\n";
7  print "Quantidade de nomes: $tamanho\n";
```

```
1  my @linguagens = ('Pascal', 'Perl', 'Python');
2
3  if (@linguagens) {
4      print "Existem itens nesta lista.\n";
5  } else {
6      print "A lista está vazia.\n";
7  }
```

# Variável padrão

\$ \_



# Avaliação comparativa

## Concatenação de strings

```
1  # Entrada
2  my $str1 = <STDIN>;
3  my $str2 = <STDIN>;
4
5  # Remove as quebras de linha
6  chomp($str1, $str2);
7
8  # Concatena str2 à str1
9  $str1 = $str1 . $str2;
10
11 # Resultado
12 print $str1;
```

```
1  #include <stdio.h>
2  int main() {
3      char str1[100], str2[100], i, j;
4
5      // Entrada
6      scanf("%s", str1);
7      scanf("%s", str2);
8
9      // Calcula o tamanho da string str1
10     for(i = 0; str1[i] != '\0'; ++i);
11
12     // Concatena str2 à str1
13     for(j = 0; str2[j] != '\0'; ++j, ++i) {
14         str1[i] = str2[j];
15     }
16
17     // Adiciona o caracter nulo ao fim da string
18     str1[i] = '\0';
19
20     // Resultado
21     printf("%s\n", str1);
22
23     return 0;
24 }
```

# Avaliação comparativa

## Desempenho

Perl

C

- Interpretado
- Compila a fonte sempre que executado
- Procura módulos

- Compilado
- Mais eficiente