Introdução à Programação de Computadores para Biologia

Arrays e Hashes

Aula 05b

https://tttorres.github.io/introprog2021/

PROBLEMA

> No arquivo dmel-gene-r5.45.fasta há um conjunto de genes presentes no genoma de *Drosophila melanogaster*. Queremos armazenar cada sequência em uma variável com um número identificador único.

1. Como implementar em Perl?

2. Quantas variáveis seriam geradas?

PROBLEMA

> No arquivo dmel-gene-r5.45.fasta há um conjunto de genes presentes no genoma de *Drosophila melanogaster*. Queremos armazenar cada sequência em uma variável com um número identificador único.

1. Como implementar em Perl?

```
$gene00000001 = "ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGC";
$gene00000002 = "AAGAGAGAGGATAGAGAGATCTTCTCTCTCGGGGTAGC";
```

2. Quantas variáveis seriam geradas?

```

```

PROBLEMA

> No arquivo dmel-gene-r5.45.fasta há um conjunto de genes presentes no genoma de *Drosophila melanogaster*. Queremos armazenar cada sequência em uma variável com um número identificador único.

1. Como implementar em Perl?

```
$gene00000001 = "ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGC";
$gene00000002 = "AAGAGAGAGGATAGAGAGATCTTCTCTCTCGGGGTAGC";
```

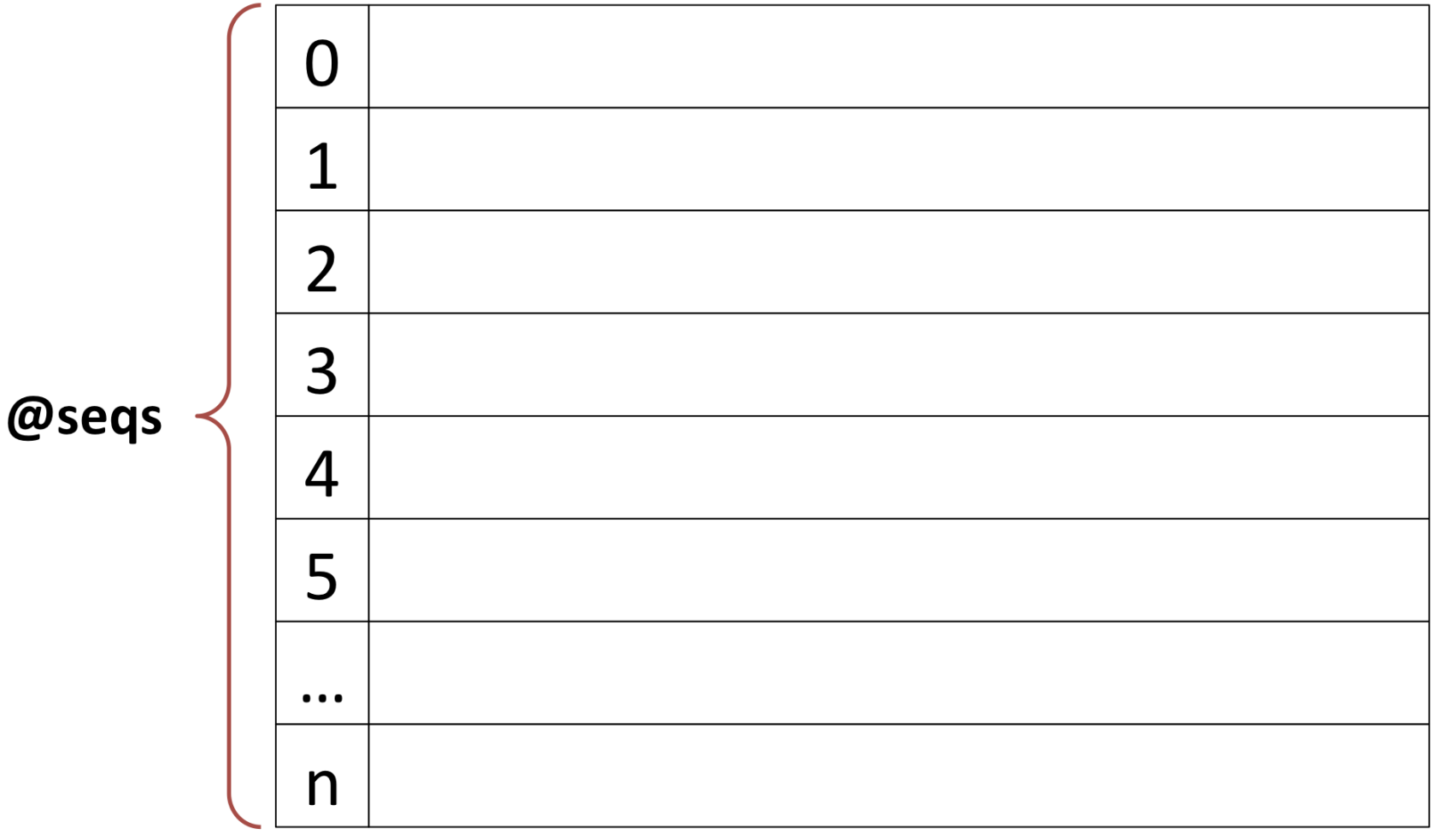
2. Quantas variáveis seriam geradas?

```
$gene00000001 = "ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGC";
$gene00015608 = "AAGAGAGAGGATAGAGAGATCTTCTCTCTCGGGGTAGC";
```

VETORES ("ARRAYS")

0	
1	
2	
3	
4	
5	
...	
n	

VETORES ("ARRAYS")



VETORES ("ARRAYS")

@seqs

0	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
1	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
2	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
3	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
4	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
5	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
...	...
n	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA

VETORES ("ARRAYS")

@genes

0	for
1	Mvl
2	Cyp6g1
3	Cut
4	Ovo
5	Npf
...	...
n	white

VETORES ("ARRAYS")

Na prática

1. No terminal, descobrir quais são os cinco primeiros genes do arquivo **dmel-gene-r5.45.fasta**

VETORES ("ARRAYS")

Na prática

1. No terminal, descobrir quais são os cinco primeiros genes do arquivo **dmel-gene-r5.45.fasta**

```
grep ">" dmel-all-gene-r5.45.fasta | head -5
>FBgn0033056 loc=2R:complement(1944862..1947063) name=CG7856; length=
release=r5.45; species=Dmel;
>FBgn0037888 type=gene; loc=3R:complement(7065763..7066713); name=scp
length=951; release=r5.45; species=Dmel;
>FBgn0034742 type=gene; loc=2R:complement(18487910..18493140); name=C
length=5231; release=r5.45; species=Dmel;
>FBgn0032640 type=gene; loc=2L:17471603..17474773; name=Sgt; length=3
release=r5.45; species=Dmel;
>FBgn0259204 type=gene; loc=X:6905390..6906170; name=CG42308; length=
release=r5.45; species=Dmel;
```

VETORES ("ARRAYS")

Na prática

1. No terminal, descobrir quais são os cinco primeiros genes do arquivo **dmel-gene-r5.45.fasta**

```
grep ">" dmel-gene-r5.45.fasta | head -5
```

2. No Geany, File > New File.
3. File > Save as...
4. Gravar arquivo como arrays.pl
5. Copiar **exemplo03** da página da disciplina e substituir o nome dos genes.

VETORES ("ARRAYS")

Na prática

Script: arrays.pl

```
# exemplo03
#! /usr/bin/perl
# script para entender arrays

# criando o array de nomes
@gene_names = ("gene1", "gene2", "gene3", "gene4", "gene5"

exit;
```

VETORES ("ARRAYS")

Na prática

Script: arrays.pl

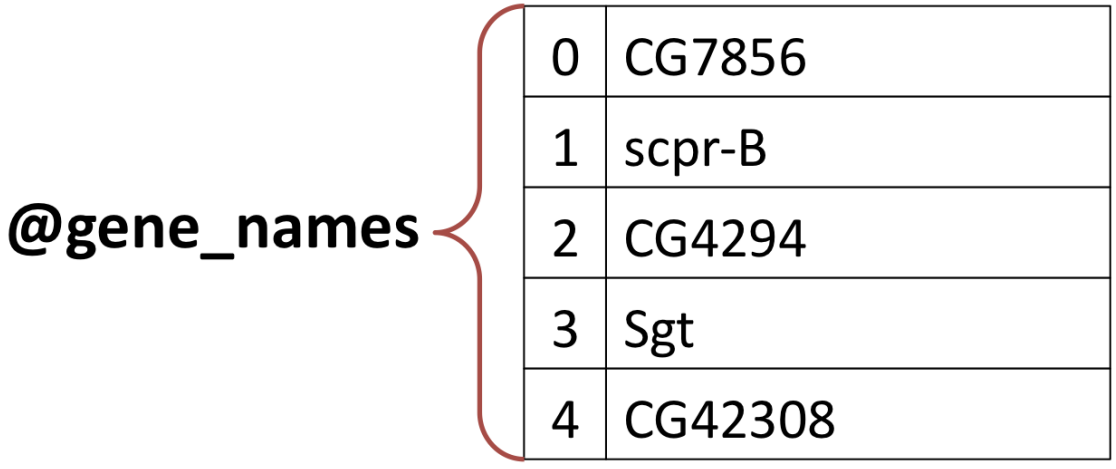
```
# exemplo03
#! /usr/bin/perl
# script para entender arrays

# criando o array de nomes
@gene_names = ("CG7856", "scpr-B", "CG4294", "Sgt", "CG42

exit;
```

VETORES ("ARRAYS")

Na prática



VETORES ("ARRAYS")

Na prática

Script: arrays.pl

Copiar e colar **exemplo04** da página

```
# exemplo01
#! /usr/bin/perl
# script para entender arrays

# criando o array de nomes
@gene_names = ("CG7856", "scpr-B", "CG4294", "Sgt",
               "CG42308");

# criando o array de IDs
@gene_IDs = ("FBgn0033056", "FBgn0037888", "FBgn0034742",
             "FBgn0032640", "FBgn0259204" );

exit;
```

VETORES ("ARRAYS")

Na prática

@gene_names

0	CG7856
1	scpr-B
2	CG4294
3	Sgt
4	CG42308

@gene_IDs

0	FBgn0033056
1	FBgn0037888
2	FBgn0034742
3	FBgn0032640
4	FBgn0259204

VETORES ("ARRAYS")

Acessando elementos individuais de um array

Script: arrays.pl

```
# criando o array de IDs
@gene_IDs = ( "FBgn0033056", "FBgn0037888", "FBgn0034742"
              "FBgn0032640", "FBgn0259204" );

# acessando os elementos individuais do array
print "$gene_IDs[0] \= $gene_names[0]\;\n";




exit;
```

VETORES ("ARRAYS")

Acessando elementos individuais de um array

Script: arrays.pl

Copiar e colar **exemplo05** da página

```
# criando o array de IDs
@gene_IDs = ( "FBgn0033056", "FBgn0037888", "FBgn0034742"
              "FBgn0032640", "FBgn0259204" );

# acessando os elementos individuais do array
print "$gene_IDs[0] \= $gene_names[0]\;\n";
print "$gene_IDs[1] \= $gene_names[1]\;\n";
print "$gene_IDs[2] \= $gene_names[2]\;\n";
print "$gene_IDs[3] \= $gene_names[3]\;\n";
print "$gene_IDs[4] \= $gene_names[4]\;\n";

exit;
```

VETORES ("ARRAYS")

Atribuição de valores individuais

Script: arrays.pl

Copiar e colar **exemplo06** da página

```
print "$gene_IDs[4] \= $gene_names[4]\;\n";

# exemplo06
# continuacao do script para entender arrays

#adicionando elementos no array
$gene_names[5] = "CG15556";
$gene_IDs[5]   = "FBgn0039821";

print "$gene_IDs[5] \= $gene_names[5]\;\n";

exit;
```

VETORES ("ARRAYS")

Atribuição de valores individuais

Script: arrays.pl

Copiar e colar **exemplo07** da página

```
$gene_IDs[5]   = "FBgn0039821";

print "$gene_IDs[5] \= $gene_names[5]\;\n";

# exemplo07
# adicionando mais elementos no array
$gene_names[7] = "CG9773";
$gene_IDs[7]   = "FBgn0037609";

print "$gene_IDs[6] \= $gene_names[6]\;\n";
print "$gene_IDs[7] \= $gene_names[7]\;\n";

exit;
```

VETORES ("ARRAYS")

Atribuição de valores individuais

Script: arrays.pl

Copiar e colar **exemplo08** da página

```
# exemplo07
# adicionando mais elementos no array
$gene_names[7] = "CG9773";
$gene_IDs[7]   = "FBgn0037609";

# exemplo08
# adicionando mais elementos no array
$gene_names[6] = "CG7059";
$gene_IDs[6]   = "FBgn0038957";

print "$gene_IDs[6] \= $gene_names[6]\;\n";
print "$gene_IDs[7] \= $gene_names[7]\;\n";

exit;
```

VETORES ("ARRAYS")

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo09** da página

1. Função push()

```
print "$gene_IDs[7] \= $gene_names[7]\;\n";

# exemplo09
# o que faz a funcao push()?
push(@gene_names, "RabX4");
push(@gene_IDs, "FBgn0051118");

print "\nExemplo07\:\n";
print "$gene_IDs[0] \= $gene_names[0]\;\n";
...
print "$gene_IDs[8] \= $gene_names[8]\;\n";

exit;
```

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo10** da página

2. Função pop()

```
# o que faz a funcao pop()?
pop(@gene_names);
pop(@gene_IDs);

print "\nExemplo10\:\n";
print "$gene_IDs[0] \= $gene_names[0]\;\n";
print "$gene_IDs[1] \= $gene_names[1]\;\n";
print "$gene_IDs[2] \= $gene_names[2]\;\n";
print "$gene_IDs[3] \= $gene_names[3]\;\n";
print "$gene_IDs[4] \= $gene_names[4]\;\n";
print "$gene_IDs[5] \= $gene_names[5]\;\n";
print "$gene_IDs[6] \= $gene_names[6]\;\n";
print "$gene_IDs[7] \= $gene_names[7]\;\n";
print "$gene_IDs[8] \= $gene_names[8]\;\n";

exit;
```

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo11** da página

3. Função shift()

```
# o que faz a funcao shift()?
$gene_name01 = shift(@gene_names);
$gene_id01   = shift(@gene_IDs);

print "\nExemplo11\:\n";
print "$gene_id01 \= $gene_name01\;\n";
print "Novo array\:\n";
print "$gene_IDs[0] \= $gene_names[0]\;\n";
print "$gene_IDs[1] \= $gene_names[1]\;\n";
print "$gene_IDs[2] \= $gene_names[2]\;\n";
print "$gene_IDs[3] \= $gene_names[3]\;\n";
print "$gene_IDs[4] \= $gene_names[4]\;\n";
print "$gene_IDs[5] \= $gene_names[5]\;\n";
print "$gene_IDs[6] \= $gene_names[6]\;\n";
print "$gene_IDs[7] \= $gene_names[7]\;\n";
print "$gene_IDs[8] \= $gene_names[8]\;\n";
```

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo12** da página

4. Função unshift()

```
# o que faz a funcao unshift()?
unshift(@gene_names, $gene_name01);
unshift(@gene_IDs, $gene_id01);

print "\nExemplo12\:\n";
print "$gene_IDs[0] \= $gene_names[0]\;\n";
print "$gene_IDs[1] \= $gene_names[1]\;\n";
print "$gene_IDs[2] \= $gene_names[2]\;\n";
print "$gene_IDs[3] \= $gene_names[3]\;\n";
print "$gene_IDs[4] \= $gene_names[4]\;\n";
print "$gene_IDs[5] \= $gene_names[5]\;\n";
print "$gene_IDs[6] \= $gene_names[6]\;\n";
print "$gene_IDs[7] \= $gene_names[7]\;\n";
print "$gene_IDs[8] \= $gene_names[8]\;\n\n";
```

VETORES ("ARRAYS")

Contexto de lista ou contexto escalar?

Script: arrays.pl

Atribuindo o array a uma variável escalar

```
# continuacao do script para entender arrays
# contexto lista ou escalar?

$n = @gene_names;
print "n \= $n!\n";




exit;
```

VETORES ("ARRAYS")

Contexto de lista ou contexto escalar?

Script: arrays.pl

Atribuindo o array a uma variável escalar

```
# continuacao do script para entender arrays
# contexto lista ou escalar?

$n = @gene_names;
print "n \= $n!\n";

print @gene_names;




exit;
```

VETORES ("ARRAYS")

Contexto de lista ou contexto escalar?

Script: arrays.pl

Atribuindo o array a uma variável escalar

```
# continuacao do script para entender arrays
# contexto lista ou escalar?

$n = @gene_names;
print "n \= $n!\n";

print @gene_names;

print "\n@gene_names\n";



exit;
```

VETORES ("ARRAYS")

Contexto de lista ou contexto escalar?

Script: arrays.pl

Atribuindo o array a uma variável escalar

```
# continuacao do script para entender arrays
# contexto lista ou escalar?

$n = @gene_names;
print "n \= $n!\n";

print @gene_names;

print "\n@gene_names\n";

print join(", ", @gene_names);

exit;
```

VETORES ("ARRAYS")

Valores sequenciais

Script: arrays.pl

Atribuindo valores sequenciais a um array

```
# continuacao do script para entender arrays
# valores sequenciais

@var_10 = (1..10);
@var_20 = (10..20);
@var_abc = (a..z);

print "@var_10\n";
print "@var_20\n";
print "@var_abc\n";

exit;
```

VETORES ("ARRAYS")

Subconjuntos

Script: arrays.pl

Selecionando subconjuntos em arrays

```
# continuacao do script para entender arrays
# subconjuntos

@subset_genes = @gene_names[2..4];

print "\n@subset_genes\n";



exit;
```

VETORES ("ARRAYS")

Subconjuntos

Script: arrays.pl

Selecionando subconjuntos em arrays

```
# continuacao do script para entender arrays
# subconjuntos

@subset_genes = @gene_names[2,4];

print "\n@subset_genes\n";



exit;
```

VETORES ("ARRAYS")

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo13** da página

5. Função splice()

```
# exemplo13
# funcao splice() para substituicao
# uso: splice(@array, inicio, tamanho, @lista_substituir)

@nums = (1..20);

print "\nExemplo13\:\n";
print "Antes - @nums\n";

splice(@nums,5 ,5 , 21..25);

print "Depois - @nums\n";

exit;
```

VETORES ("ARRAYS")

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo14** da página

5. Função split()

```
# funcao split()
# transformar string em array

# strings
$var_music="Rain-Drops-On-Roses-And-Whiskers-On-Kittens";
$var_Mando="This is the way";

# transformar strings em arrays
@music = split('-', $var_music);
@Mando  = split(' ', $var_Mando);

print "$music[5]\n";
print "$Mando[3]\n";
```

VETORES ("ARRAYS")

Funções para a manipulação de arrays

Script: arrays.pl

Copiar e colar **exemplo15** da página

5. Função join()

```
# funcao join()
# transformar array em string

$string1 = join( ' ', @music );
$string2 = join( '-', @Mando );

print "\nExemplo15\:\n";
print "$string1\n";
print "$string2\n";

exit;
```

VETORES ("ARRAYS")

Problema: criar uma agenda de endereços

@agenda_nomes

0	Ahsoka Tano
1	Sabine Wren
2	Hera Syndulla
3	Asajj Ventress
4	Rey
5	Leia Organa
6	Bo-Katan Kryze
7	Jyn Erso

VETORES ("ARRAYS")

Problema: criar uma agenda de endereços

@agenda_enderecos

0	Shili
1	Mandalore
2	Ryloth
3	Dathomir
4	Jakku
5	Alderaan
6	Mandalore
7	Vallt

VETORES ("ARRAYS")

@gene_names

0	CG7856
1	scpr-B
2	CG4294
3	Sgt
4	CG42308

@gene_IDs

0	FBgn0033056
1	FBgn0037888
2	FBgn0034742
3	FBgn0032640
4	FBgn0259204

VETORES ASSOCIATIVOS (HASHES)

genes

CG7856	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCT
scpr-B	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCT
CG4294	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCA
Sgt	ATAGCGCTAGCTGAGCTAGCTGGTAGCTGAGCTGAGTATA
CG42308	ATAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
CG15556	ATAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
CG7059	ATAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
CG9773	ATAGCGCTAGCTGAGCTAGCTGAGCTAGCTGAGCTGAGT

VETORES ASSOCIATIVOS (HASHES)

%genes

CG7856	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCT
scpr-B	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCT
CG4294	ATAGCGCTAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCA
Sgt	ATAGCGCTAGCTGAGCTAGCTGGTAGCTGAGCTGAGTATA
CG42308	ATAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
CG15556	ATAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
CG7059	ATAGCTGAGCTAGCTGAGCTGCGTAGCTGAGCTGAGTATA
CG9773	ATAGCGCTAGCTGAGCTAGCTGAGCTAGCTGAGCTGAGT

keys **value**

VETORES ASSOCIATIVOS (HASHES)

Na prática

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como hashes.pl
4. Copiar **exemplo16** da página da disciplina e substituir o nome dos genes.

VETORES ASSOCIATIVOS (HASHES)

Na prática

Script: hashes.pl

```
# exemplo16
#! /usr/bin/perl
# script para entender hashes

# criando o hash de genes
%genes = ("FBgn0033056", "CG7856",
          "FBgn0037888", "scpr-B",
          "FBgn0034742", "CG424",
          "FBgn0032640", "Sgt",
          "FBgn0259204", "CG42308",
          "FBgn0039821", "CG15556",
          "FBgn0038957", "CG7059",
          "FBgn0037609", "CG9773");

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Na prática

Script: hashes.pl

```
# exemplo16
#! /usr/bin/perl
# script para entender hashes

# criando o hash de genes
%genes = ("FBgn0033056","CG7856","FBgn0037888","scpr-B",
"FBgn0034742","CG424","FBgn0032640","Sgt","FBgn0259204",
"CG42308","FBgn0039821","CG15556","FBgn0038957","CG7059",
"FBgn0037609","CG9773");




exit;
```

VETORES ASSOCIATIVOS (HASHES)

Na prática

Script: hashes.pl

```
# exemplo16
#! /usr/bin/perl
# script para entender hashes

# criando o hash de genes
%genes = ("FBgn0033056" => "CG7856",
          "FBgn0037888" => "scpr-B",
          "FBgn0034742" => "CG424",
          "FBgn0032640" => "Sgt",
          "FBgn0259204" => "CG42308",
          "FBgn0039821" => "CG15556",
          "FBgn0038957" => "CG7059",
          "FBgn0037609" => "CG9773");

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Acessando elementos individuais de um hash

Script: hashes.pl

```
# exemplo1
#! /usr/bin/perl
# script para entender hashes

# criando o hash de genes
%genes = ("FBgn0033056" => "CG7856",
          "FBgn0037888" => "scpr-B",
          "FBgn0034742" => "CG424",
          "FBgn0032640" => "Sgt",
          "FBgn0259204" => "CG42308",
          "FBgn0039821" => "CG15556",
          "FBgn0038957" => "CG7059",
          "FBgn0037609" => "CG9773");



exit;
```

VETORES ASSOCIATIVOS (HASHES)

Acessando elementos individuais de um hash

Script: hashes.pl

```
# criando o hash de genes
%genes = ("FBgn0033056" => "CG7856",
          "FBgn0037888" => "scpr-B",
          "FBgn0034742" => "CG424",
          "FBgn0032640" => "Sgt",
          "FBgn0259204" => "CG42308",
          "FBgn0039821" => "CG15556",
          "FBgn0038957" => "CG7059",
          "FBgn0037609" => "CG9773");

# acessando elementos individuais do hash
$gene01 = $genes{FBgn0033056};
$gene02 = $genes{FBgn0037888};

print "$gene01 e $gene02\n";

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Criando uma nova entrada em um hash

Script: hashes.pl

```
# acessando elementos individuais do hash
$gene01 = $genes{FBgn0033056};
$gene02 = $genes{FBgn0037888};

print "$gene01 e $gene02\n";




exit;
```

VETORES ASSOCIATIVOS (HASHES)

Criando uma nova entrada em um hash

Script: hashes.pl

```
# acessando elementos individuais do hash
$gene01 = $genes{FBgn0033056};
$gene02 = $genes{FBgn0037888};

print "$gene01 e $gene02\n";

# criar uma nova entrada
$genes{"FBgn0051118"} = "RabX4";

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de hashes

Script: hashes.pl

Recuperando as chaves dos hashes

1. Função keys()

```
print "$gene01 e $gene02\n";

# criar uma nova entrada
$genes{"FBgn0051118"} = "RabX4";

# recuperar os IDs
@gene_IDs = keys(%genes);



exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de hashes

Script: hashes.pl

Recuperando as chaves dos hashes

1. Função keys()

```
print "$gene01 e $gene02\n";

# criar uma nova entrada
$genes{"FBgn0051118"} = "RabX4";

# recuperar os IDs
@gene_IDs = keys(%genes);

print "@gene_IDs\n";

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de hashes

Script: hashes.pl

Recuperando os valores dos hashes

2. Função values()

```
# recuperar os IDs
@gene_IDs = keys(%genes);

print "@gene_IDs\n";

# recuperar os valores
@gene_values = values(%genes);



exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de hashes

Script: hashes.pl

Recuperando os valores dos hashes

2. Função values()

```
# recuperar os IDs
@gene_IDs = keys(%genes);

print "@gene_IDs\n";

# recuperar os valores
@gene_values = values(%genes);

print "@gene_values\n";

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de arrays/hashes

Script: hashes.pl

Ordenando arrays

3. Função sort()

```
# recuperar os valores
@gene_values = values(%genes);

print "@gene_values\n";

# recuperar os IDs em ordem alfabética



exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de arrays/hashes

Script: hashes.pl

Ordenando arrays

3. Função sort()

```
# recuperar os valores
@gene_values = values(%genes);

print "@gene_values\n";

# recuperar os IDs em ordem alfabética
@gene_IDs = sort(keys(%genes));

print "@gene_IDs\n";

exit;
```

VETORES ASSOCIATIVOS (HASHES)

Funções para a manipulação de arrays/hashes

Script: hashes.pl

Ordenando arrays

4. Função reverse()

```
# recuperar os IDs em ordem alfabética
@gene_IDs = sort(keys(%genes));

print "@gene_IDs\n";

# o que faz a funcao reverse()
@gene_values_rev = reverse(@gene_values);

print "@gene_values_rev\n";

exit;
```