Introdução à Programação de Computadores para Biologia

Manipulação de arquivos

Input & Output (I/O)

Aula 09

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

ESTRUTURAS DE CONTROLE - LOOPS

for, foreach, while

1. *for*

```
for (inicializacao, teste, modificacao) {
   bloco de comandos
}
```

2. *foreach*

```
foreach $escalar(@array) {
   bloco de comandos
}
```

3. *while*

```
while (teste) {
   bloco de comandos
}
```

ESTRUTURAS DE CONTROLE

Repetição

Problema:

> Calcular o tamanho médio de sequências armazenadas em um array.

ESTRUTURAS DE CONTROLE

Repetição

Problema:

> Encontrar um valor específico num hash.

ESTRUTURAS DE CONTROLE - LOOPS

REVISÃO

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como pokemon.pl
4. Copiar **exemplo01** da página da disciplina.
5. Completar o script.

ESTRUTURAS DE CONTROLE - REVISÃO

#Exemplo01; script: pokemon.pl

```
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Pikachu";

# usando for


# usando while


# usando foreach


# sem usar loops


exit;
```

```perl
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Pikachu";

# usando for
@chaves = keys(%evolution);
$numTotal = @chaves;


print "\nExemplo 01 (for):\n";

for ( $i = 0; $i < $numTotal; $i++ ) {
   if ($chaves[$i] eq $pokemon) {
      print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n";


   }
}




exit;
```

```perl
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Bulbasaur";

# usando for
@chaves = keys(%evolution);
$numTotal = @chaves;


print "\nExemplo 01 (for):\n";

for ( $i = 0; $i < $numTotal; $i++ ) {
   if ($chaves[$i] eq $pokemon) {
      print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n";

      last;
   }
}




exit;
```

```perl
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Bulbasur";

# usando for
@chaves = keys(%evolution);
$numTotal = @chaves;
$encontrado = 0;

print "\nExemplo 01 (for):\n";

for ( $i = 0; $i < $numTotal; $i++ ) {
   if ($chaves[$i] eq $pokemon) {
      print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n" ;
      $encontrado = 1;
      last;
   }
}

unless ($encontrado) {
        print "Pokemon \"$pokemon\" nao encontrado\n";
}

exit;
```

```perl
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Bulbasaur";

# usando foreach
@chaves = keys(%evolution);

$encontrado = 0;

print "\nExemplo 01 (foreach):\n";

foreach $nome (@chaves) {
   if ($nome eq $pokemon) {
      print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n" ;
      $encontrado = 1;
      last;
   }
}

unless ($encontrado) {
        print "Pokemon \"$pokemon\" nao encontrado\n";
}

exit;
```

```perl
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Bulbasaur";

# usando while
@chaves = keys(%evolution);

$encontrado = 0;

print "\nExemplo 01 (while):\n";

while (@chaves) {
   $nome = shift(@chaves);
   if ($nome eq $pokemon) {
      print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n" ;
      $encontrado = 1;
      last;
   }
}

unless ($encontrado) {
        print "Pokemon \"$pokemon\" nao encontrado\n";
}

exit;
```

```perl
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Bulbasaur";

# sem loops

print "\nExemplo 01 (sem loops):\n";

if ($evolution{$pokemon}) {
   print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n" ;
} else {
   print "Pokemon \"$pokemon\" nao encontrado\n";
}

exit;
```

HARDCODING

Script: pokemon.pl

```
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

$pokemon = "Bulbasaur";

# sem loops
print "\nExemplo 01 (sem loops):\n";

if ($evolution{$pokemon}) {
   print "Encontrado: $pokemon \-\> $evolution{$pokemon}\n" ;
} else {
   print "Pokemon \"$pokemon\" nao encontrado\n";
}

exit;
```

ENTRADA DE DADOS

STDIN

Script: pokemon.pl

Modificar o exemplo01 para solicitar a informação ao usuário.

No terminal:

```
TatianasMacBook:~ tatiana$ perl pokemon.pl

Qual o Pokemon desejado?
Pikachu


Pikachu -> Raichu
```

ENTRADA DE DADOS

STDIN

Script: pokemon.pl

Modificar o exemplo01 para solicitar a informação ao usuário.

No terminal:

```
print "\nQual o Pokemon desejado?\n";

$infoUsuario = <STDIN>;
chomp $infoUsuario ;

if ($evolution{$infoUsuario}) {
   print "$infoUsuario} \-\> $evolution{$infoUsuario}\n";
} else {
   print "Não encontrado\n" ;
}

exit;
```

ENTRADA DE DADOS

Argumentos do script

Os argumentos podem ser passados para o script na própria linha de comando:

```
TatianasMacBook:~ tatiana$ perl script.pl arg1 arg2 arg3
```

ENTRADA DE DADOS

Argumentos do script

Os argumentos podem ser passados para o script na própria linha de comando:

```
TatianasMacBook:~ tatiana$ perl script.pl arg1 arg2 arg3
```

No script, Perl transforma os argumentos em um array, @ARGV:

```
# A variavel @ARGV (ARGument Values)

print "$ARGV[0]\n"; #imprime o primeiro argumento
print "$ARGV[1]\n"; #imprime o segundo argumento

# ...

print "$ARGV[$#ARGV]\n"; #imprime o ultimo argumento
```

ENTRADA DE DADOS

Argumentos do script

Script: notas.pl

```
#!/usr/bin/perl

$E = 8;
$M = 9;
$H = 10;

$nota_final = ((2*$E)+(3*$M)+(5*$H))/10;

if ($nota_final >= 5) {
   print "Nota: $nota_final, Aluno aprovado.\n";
} else {
   print "Nota: $nota_final, Aluno reprovado.\n";
}

exit;
```

ENTRADA DE DADOS

Argumentos do script

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como notas.pl
4. Copiar **exemplo02** da página da disciplina.

ENTRADA DE DADOS

Argumentos do script

Script: notas.pl

```
#!/usr/bin/perl

($E, $M, $H) = @ARGV;

$nota_final = ((2*$E)+(3*$M)+(5*$H))/10;

if ($nota_final >= 5) {
   print "Nota: $nota_final, Aluno aprovado.\n";
} else {
   print "Nota: $nota_final, Aluno reprovado.\n";
}

exit;
```

ENTRADA DE DADOS

Argumentos do script

Passando o argumento para o script

```
TatianasMacBook:~ tatiana$ perl notas.pl 8 9 10
Nota: 9.3, Aluno aprovado.
```

ENTRADA DE DADOS

Argumentos do script

Passando o argumento para o script

```
TatianasMacBook:~ tatiana$ perl notas.pl 8 9 10
Nota: 9.3, Aluno aprovado.
```

No script, Perl transforma os argumentos em um array, @ARGV:

```
print "$ARGV[0]\n"; #imprime valor 8
print "$ARGV[1]\n"; #imprime valor 9
print "$ARGV[2]\n"; #imprime valor 10
```

ENTRADA DE DADOS

Argumentos do script

Modificar o **#exemplo01**, pokemon.pl para o seguinte input/output:

```
MacBook:~ tatiana$ perl pokemon.pl Pikachu Bulbasaur

Pikachu = Raichu

Bulbasaur = Ivysaur
```

ENTRADA DE DADOS

Argumentos do script

Exemplo01: pokemon.pl

```
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

foreach $pokemon(@ARGV) {
   if ($evolution{$pokemon}) {
      print "$pokemon\=$evolution{$pokemon}\n\n";
   }
}

exit;
```

ENTRADA DE DADOS

Argumentos do script

Exemplo01: pokemon.pl

Modificar o **#exemplo01**, para que, caso o nome não esteja presente, ele imprima NA

```
MacBook:~ tatiana$ perl pokemon.pl Pikachu Caterpie

Pikachu = Raichu

Caterpie = NA
```

ENTRADA DE DADOS

Argumentos do script

Exemplo01: pokemon.pl

```
#!usr/bin/perl

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

foreach $pokemon(@ARGV) {
   if ($evolution{$pokemon}) {
      print "$pokemon\=$evolution{$pokemon}\n\n";
   } else {
      print "$pokemon\=NA\n\n";
   }
}

exit;
```

ENTRADA DE DADOS

Teste de argumentos

Script: notas.pl

```
#!/usr/bin/perl

($E, $M, $H) = @ARGV;

$nota_final = ((2*$E)+(3*$M)+(5*$H))/10;

if ($nota_final >= 5) {
   print "Nota: $nota_final, Aluno aprovado.\n";
} else {
   print "Nota: $nota_final, Aluno reprovado.\n";
}

exit;
```

ENTRADA DE DADOS

Teste de argumentos

Passando o argumento para o script

```
TatianasMacBook:~ tatiana$ perl notas.pl 8 9
Nota: 4.3, Aluno reprovado.
```

ENTRADA DE DADOS

Teste de argumentos

Script: notas.pl

```
#!/usr/bin/perl

unless ($#ARGV == 2) {
   die "Usage: $0 Nota_P1 Nota_P2 Nota_P3\n";
}
($P1, $P2, $P3) = @ARGV;

$nota_final = ((3.5*$P1)+(4.5*$P2)+(2*$P3))/10;

if ($nota_final >= 5) {
   print "Nota: $nota_final, Aluno aprovado.\n";
} else {
   print "Nota: $nota_final, Aluno reprovado.\n";
}

exit;
```

ENTRADA DE DADOS

Teste de argumentos

Exemplo01: pokemon.pl

Modificar o **#exemplo01**, para que, se não for informado NENHUM argumento, o programa restorne a mensagem de ajuda

```
MacBook:~ tatiana$ perl pokemon.pl

Usage: revisao.pl nome_1 nome_2 nome_n
```

ENTRADA DE DADOS

Teste de argumentos

Exemplo01: pokemon.pl

```
#!usr/bin/perl

unless(@ARGV) {
        die "Usage: $0 nome_1 nome_2 nome_n\n";
}

%evolution = ("Bulbasaur",  "Ivysaur",
              "Charmander", "Charmeleon",
              "Squirtle",   "Wartortle",
              "Pikachu",    "Raichu");

foreach $pokemon(@ARGV) {
   if ($evolution{$pokemon}) {
      print "$pokemon\=$evolution{$pokemon}\n";
   } else {
      print "$pokemon\=NA\n";
   }
}

exit;
```

ENTRADA E SAÍDA DE DADOS

Hardcoding

Script: loops.pl

```
@seqs = ("ATGGCGTAGATCG", "TAAGCCCCGGTATATTTGACCCCGAT", "(

$seqs = @seqs; #numero de elementos do array

print "Numero de sequencias = $seqs\n";

foreach $seq (@seqs) {
   $tamanho += length($seq);
}

$media = $tamanho/$seqs;

print "Media de tamanho = $media\n";

exit;
```

ENTRADA E SAÍDA DE DADOS

Arquivos

- Comando *open()*
- filehandle

No script:

```
open(FILEHANDLE, filename);
```

Na linha de comando:

```
Darwin:~ Tatiana$ perl files.pl ~/seq/dmel-gene.fasta
```

ENTRADA E SAÍDA DE DADOS

Arquivos

- Comando *open()*
- filehandle

No script:

```
#!usr/bin/perl

unless(@ARGV) {
   die "Usage: $0 filename \n";
}

$input = $ARGV[0];

open(INPUT, $input);

exit;
```

ENTRADA E SAÍDA DE DADOS

Arquivos

- Comando *open()*
- filehandle

No script:

```
#!usr/bin/perl

unless(@ARGV) {
   die "Usage: $0 filename \n";
}

$output = $ARGV[0];

open(OUTPUT, $output);

exit;
```

ENTRADA E SAÍDA DE DADOS

Arquivos

Na linha de comando:

```
Darwin:~ Tatiana$ perl files.pl input.txt output.txt
```

No script:

```
$input = $ARGV[0];
$output = $ARGV[1];

open(INPUT, $input);
open(OUTPUT, $output);
```

ENTRADA E SAÍDA DE DADOS

Input & Output

```
#!usr/bin/perl

$input   = $ARGV[0];
$output1 = $ARGV[1];
$output2 = $ARGV[2];

open(INPUT,   "<$input");
open(OUTPUT1, ">$output1");
open(OUTPUT2, ">>$output2");

close INPUT
close OUTPUT1;
close OUTPUT2;

exit;
```

ENTRADA E SAÍDA DE DADOS

Input & Output

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como arquivos.pl
4. Copiar **exemplo03** da página da disciplina.

ENTRADA E SAÍDA DE DADOS

Input & Output

Script: arquivos.pl

```
#! usr/bin/perl

$output1 = $ARGV[0];
$output2 = $ARGV[1];

open(OUTPUT1, ">$output1");
open(OUTPUT2, ">>$output2");

print OUTPUT1 "Escrevendo no arquivo 1\n";
print OUTPUT2 "Escrevendo no arquivo 2\n";

close OUTPUT1;
close OUTPUT2;

exit;
```

ENTRADA E SAÍDA DE DADOS

Input & Output

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como arquivos.pl
4. Copiar **exemplo03** da página da disciplina.
5. Executar o script duas vezes.

```
Darwin:~ tatiana$ perl arquivos.pl out1.txt out2.txt
```

6. Verificar o conteúdo dos arquivos out1.txt e out2.txt

ENTRADA E SAÍDA DE DADOS

Saídas

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como saidas.pl
4. Copiar **exemplo04** da página da disciplina.
5. Executar saidas.pl

ENTRADA E SAÍDA DE DADOS

Saídas

Script: saidas.pl

```
#!usr/bin/perl

$output = $ARGV[0];
open(OUTPUT, ">$output");
print OUTPUT "Escrevendo no arquivo usando OUTPUT\n";

print "Onde estou escrevendo sem filehandle?\n";
print STDERR "Onde estou escrevendo com STDERR?\n";
print STDOUT "Onde estou escrevendo com STDOUT?\n";

close OUTPUT;
exit;
```

ENTRADA E SAÍDA DE DADOS

Saídas

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como saidas.pl
4. Copiar **exemplo04** da página da disciplina.
5. Executar saidas.pl
6. Executar saidas.pl novamente da seguinte forma:

```
Darwin:~ tatiana$ perl saida.pl out.txt >direcionado.txt
```

ENTRADA E SAÍDA DE DADOS

Teste de arquivos

Executar saidas.pl novamente da seguinte forma:

```
Darwin:~ tatiana$ perl saida.pl ~/inexistente/out.txt
```

ENTRADA E SAÍDA DE DADOS

Teste de arquivos

Script: saidas.pl

```
#!usr/bin/perl

$output = $ARGV[0];

unless(open(OUTPUT, ">$output")) {
   print STDERR "Nao foi possivel abrir $output\n";
}

print OUTPUT "Escrevendo no arquivo usando OUTPUT\n";

print "Onde estou escrevendo sem filehandle?\n";
print STDERR "Onde estou escrevendo com STDERR?\n";
print STDOUT "Onde estou escrevendo com STDOUT?\n";

close OUTPUT;
exit;
```

ENTRADA E SAÍDA DE DADOS

Teste de arquivos

Script: saidas.pl

```
#!usr/bin/perl

$output = $ARGV[0];

unless(open(OUTPUT, ">$output")) {
   die "Nao foi possivel abrir $output\n";
}

print OUTPUT "Escrevendo no arquivo usando OUTPUT\n";

print "Onde estou escrevendo sem filehandle?\n";
print STDERR "Onde estou escrevendo com STDERR?\n";
print STDOUT "Onde estou escrevendo com STDOUT?\n";

close OUTPUT;
exit;
```

ENTRADA E SAÍDA DE DADOS

Teste de arquivos

Script: saidas.pl

```
#!usr/bin/perl

$output = $ARGV[0];

open(OUTPUT, ">$output") || die "Nao foi possivel abrir $output\n";

print OUTPUT "Escrevendo no arquivo usando OUTPUT\n";

print "Onde estou escrevendo sem filehandle?\n";
print STDERR "Onde estou escrevendo com STDERR?\n";
print STDOUT "Onde estou escrevendo com STDOUT?\n";

close OUTPUT;
exit;
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como entrada.pl
4. Copiar **exemplo05** da página da disciplina.

ENTRADA E SAÍDA DE DADOS

Entrada de dados

Script: entrada.pl

```
#!usr/bin/perl

$input  = $ARGV[0];
$output = $ARGV[1];

open(SEQ, "<$input")  || die "Nao foi possivel abrir o arq $input\n";
open(OUT, ">$output") || die "Nao foi possivel abrir o arq $output\n"

while (<SEQ>) {
   print OUT $_;
}

close SEQ;

exit;
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como entrada.pl
4. Copiar **exemplo05** da página da disciplina.
5. Baixar o arquivo dmel-gene-r5.45.fasta.
6. Executar entrada.pl da seguinte forma:

```
Darwin:~ tatiana$ perl entrada.pl dmel-gene-r5.45.fasta saida.txt
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. O que o script faz?
2. O que faz o operador <>?
3. O que é a variável $_?

```
#!usr/bin/perl

$input  = $ARGV[0];
$output = $ARGV[1];

open(SEQ, "<$input")  || die "Nao foi possivel abrir o arq $input\n";
open(OUT, ">$output") || die "Nao foi possivel abrir o arq $output\n"

while (<SEQ>) {
   print OUT $_;
}

close SEQ;

exit;
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. O que o script faz?
2. O que faz o operador <>?
3. O que é a variável $_?

```
#!usr/bin/perl

$input  = $ARGV[0];
$output = $ARGV[1];

open(SEQ, "<$input")  || die "Nao foi possivel abrir o arq $input\n";
open(OUT, ">$output") || die "Nao foi possivel abrir o arq $output\n"

while ($line = <SEQ>) {
   print OUT $line;
}

close SEQ;

exit;
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como contar_linhas.pl
4. Faça um script em Perl para contar o número de linhas do arquivo sequencias.fasta
5. Confirme o resultado do seu script com o comando *wc* do unix.

ENTRADA E SAÍDA DE DADOS

Entrada de dados

Script: contar_linhas.pl

```
#!usr/bin/perl

$input  = $ARGV[0];

open(SEQ, "<$input")  || die "Nao foi possivel abrir $input\n";

$i = 0;

while (<SEQ>) {
   ++$i;
}

print "O arquivo $input tem $i linhas.\n"

close SEQ;

exit;
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como contar_linhas.pl
4. Faça um script em Perl para contar o número de linhas do arquivo sequencias.fasta
5. Confirme o resultado do seu script com o comando *wc* do unix.
6. Modifique o script contar_linhas.pl para contar o número de sequências e não o número de linhas.
7. Confirme o resultado do seu script com o comando grep.

ENTRADA E SAÍDA DE DADOS

Entrada de dados

Script: contar_linhas.pl

```
#!usr/bin/perl

$input  = $ARGV[0]
open(SEQ, "<$input")  || die "Nao foi possivel abrir $input\n";

$i = 0;

while (<SEQ>) {

    @line = split //, $_;

    if ($line[0] eq "\>") {
        ++$i;
    }
}

print "O arquivo $input tem $i sequencias.\n";

close SEQ;

exit;
```

ENTRADA E SAÍDA DE DADOS

Entrada de dados

1. No Geany, File > New File.
2. File > Save as...
3. Gravar arquivo como contar_linhas.pl
4. Faça um script em Perl para contar o número de linhas do arquivo sequencias.fasta
5. Confirme o resultado do seu script com o comando *wc* do unix.
6. Modifique o script contar_linhas.pl para contar o número de sequências e não o número de linhas.
7. Confirme o resultado do seu script com o comando grep.
8. Mofique o script contar_linhas.pl para criar um array com o nome das sequencias.

```perl
#!usr/bin/perl

$input  = $ARGV[0];

open(SEQ, "<$input")  || die "Nao foi possivel abrir $input\n";

$i = 0;
@names = "";

while (<SEQ>) {

    chomp($_);
    @line = split //, $_;

    if ($line[0] eq "\>") {
        $names[$i] = $_;
    }
}

print "@names\n"

close SEQ;

exit;
```