UFRJ – Escola Politécnica – Departamento de Engenharia Eletrônica

Linguagens de Programação:

20/06/2016

Trabalho (parte II) -

Integrantes do grupo: Eduardo Naslausky; Igor Nascimento

1- Introdução

Nosso trabalho consiste em escrever um programa que criptografe, descriptografe e execute funções associadas ao tema, em arquivos de texto fornecidos pelo usuário. Na parte II fizemos a implementação do código em Perl, que consiste nas funções mais básicas de criptografia nos modelos da cifra de Vigenère.

2- Implementação do Programa

A linguagem utilizada é Perl. É utilizada no código as bibliotecas “warning” e “Strict” para aviso e detecção de erros no momento da interpretação. Foi utilizada uma biblioteca própria criada pelo grupo também, para implementação de uma pequena parte do programa. Essa biblioteca é de certa forma ilustrativa, visto que implementa uma função simples e que é utilizada poucas vezes no programa. Ela implementa o “decremento de variáveis de caractéres”, onde decrementar uma variável com valor ‘B’ leva ela no valor ‘A’.

Precisamos implementar tal decremento pois embora o incremento de Perl funcione para letras, o decremento não funcionava como esperado (Isto é, simetricamente ao incremento) [1].

O programa principal em perl, possuí duas funções “sub”: Code e Decode.

Ambas recebem três argumentos, que representam a mensagem, a senha, e o número de caractéres na senha. Elas tem como objetivo codificar e decodificar usando os argumentos como seus nomes sugerem.

O programa aceita senha com letras maíusculas e minúsculas, e tem o mesmo resultado entre si. Com respeito a mensagem, letras maiúsculas, minúsculas e números são aceitos. Espaços são deixados como tal.

3- Casos de Uso

Este programa conta apenas com dois casos de uso: Um de codificação e outro de decodificação.

Para a função code, mudamos no código os argumentos para mensagem: “Trabalho Nota 10”; senha: “Miguel”; comprimento: 6;

Para decode, igualmente, porém com o resultado adquirido no code.

Segue imagens com o resultado:

```
67  my $m= "Trabalho Nota 10";
68  my $s= "Miguel";
69  my $lenght= 6;
70  #print code($m,$s, $lenght), "\n";
71  print decode($m,$s, $lenght), "\n";
```

```
my $m= "Fzgvewtw Tixl 38";
my $s= "Miguel";
my $lenght= 6;
print code($m,$s, $lenght), "\n";
#print decode($m,$s, $lenght), "\n";
```

Pythonmelhor.pl

```
26
27  sub decode{
28      my ($mensagem, $senha, $tamanho) = @_;
29      my @charsMensagem = split("", $mensagem);
30      my $lowerSenha= lc($senha);
31      my @charsSenha = split("", $lowerSenha);
32      my $contador=0;
33      foreach (@charsMensagem){
34          my $i='a';
35          if (/[a-zA-Z]/){
36              while($i ne @charsSenha[$contador % $tamanho]){
37                  for (my $j=0; $j<25; $j++){#HACK PARA FAZER DECREMENTO, JA QUE P
38                      ++$_;
39                  }
40                  ++$i;
41              }
42              my @letra = split("",$_);
43              $_= $letra[-1];
44              $contador++;
45              #print $_,"\n";
46          }
47          elsif(/[0-9]/){
48              while($i ne @charsSenha[$contador % $tamanho]){
49                  for (my $j=0; $j<9; $j++){#HACK PARA FAZER DECREMENTO, JA QUE PERL NÃO CONSEGUE DECREMENTAR
50                      ++$_;
51                  }
52                  ++$i;
53              }
54              my @letra = split("",$_);
55              $_= $letra[-1];
56              $contador++;
57          }
58      }
59      return (@charsMensagem);
```

Git CMD - CrypitoCeasar Direto

```
C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>perl Pythonmelhor.pl
Trabalho Nota 10

C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>
```


```
67  my $m= "Trabalho Nota 10";
68  my $s= "Miguel";
69  my $lenght= 6;
70  #print code($m,$s, $lenght), "\n";
71  print decode($m,$s, $lenght), "\n";
```


```
1   use warnings;
2   use strict;
3   #sdadsad
4   sub code{
5       my ($mensagem, $senha, $tamanho) = @_;
6       my @charsMensagem = split("", $mensagem);
7       my $lowerSenha= lc($senha);
8       my @charsSenha = split("", $lowerSenha);
9       my $contador=0;
10      foreach (@charsMensagem){
11          my $i='a';
12          if (/[a-zA-Z0-9]/){
13              while($i ne @charsSenha[$contador % $tamanho]){
14                  ++$_;
15                  ++$i;
16              }
17              my @letra = split("",$_);
18              $_= $letra[-1];
19              $contador++;
20              #print $_,"\n";
21          }
22      }
23      return (@charsMensagem);
24  }
25
26
27  sub decode{
28      my ($mensagem, $senha, $tamanho) = @_;
29      my @charsMensagem = split("", $mensagem);
30      my $lowerSenha= lc($senha);
```

Git CMD - CrypitoCeasar Direto

```
C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   Pythonmelhor.pl

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Pythonmelhor.pl


C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>git add Pythonmelhor.pl

C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   Pythonmelhor.pl


C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>perl Pythonmelhor.pl
Fzgvewtw Tixl 38

C:\Users\Rita\Desktop\Lig. Prog\Trabalhos\CryptoCaesar>
```

4- Conclusão:
   O trabalho em Perl foi implementado e não é de extrema complexidade. Foram aprendidas algumas funções como lc(), que transforma a string toda em caixa baixa, além de aprendidas também lógicas envolvendo array e passagens para funções Sub.
5- Referências

[1] http://perldoc.perl.org/perlop.html#Auto-increment-and-Auto-decrement

(Onde podemos ler: "The auto-decrement operator is not magical.")