Guilherme Quadros da Silva

RU: 3282910

Uninter - Matemática Computacional: AP Criptografia Simétrica com XOR

Atividade Prática (AP) da disciplina de Matemática Computacional do curso de Engenharia de Computação da Uninter:

Codificar a mensagem “APROVADO” por criptografia simétrica pelo algoritmo elementar XOR utilizando como chave criptográfica o seu RU ou parte dele. Após a obtenção da cifra decodificá-la comprovando a reciprocidade do processo.

Instruções

A pasta **"source"** contém o código-fonte do programa criado para resolver o problema proposto. A solução do projeto foi escrita em .NET 4.6 com a linguagem de programção C# utilizando a IDE "Rider" versão 2020.1.3 da Jetbrains, mas a última versão do Visual Studio Community deve conseguir abrir o projeto normalmente. Caso queira somente visualizar o código-fonte basta abrir o arquivo "Program.cs" em um bloco de notas.

A pasta **"exe"** contém um executável do programa desenvolvido compatível .NET 4.6, o Windows 10 deve suportar essa versão por padrão: https://docs.microsoft.com/en-us/archive/blogs/astebner/mailbag-what-version-of-the-net-framework-is-included-in-what-version-of-the-os

Mas se o seu sistema operacional não suportar o executável você conseguirá facilmente achar na internet uma versão de .NET Framework ou Mono compatível.

O código é escrito em inglês por uma preferência minha e costume mesmo.

O Programa

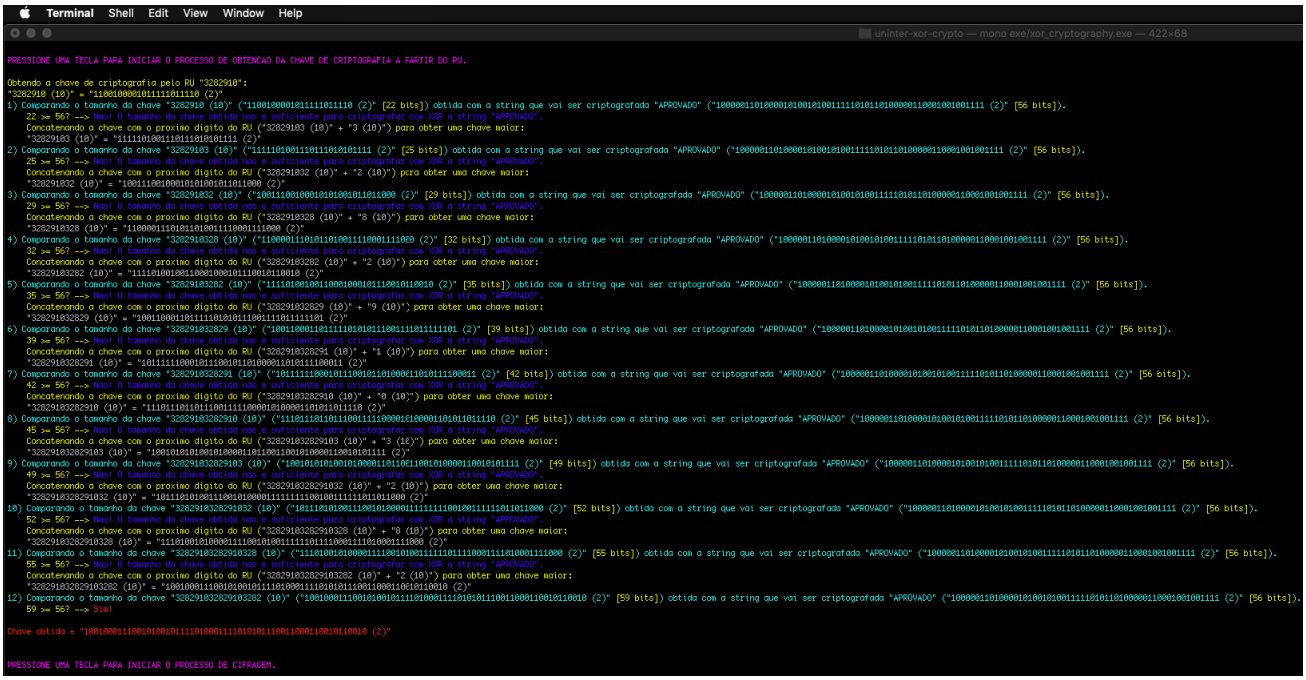
1. O programa pode ser iniciado rodando o arquivo "xor_cryptography.exe" dentro da pasta "exe". Cada etapa do processo ele espera um comando do usuário para prosseguir.

```
guiquadros@iMac848 uninter-xor-crypto (master) $ ls -la
total 40
drwxr-xr-x   8 guiquadros  staff   256 Aug 31 04:34 .
drwxrwxrwx  29 guiquadros  staff   928 Aug 29 23:59 ..
-rw-r--r--@  1 guiquadros  staff  6148 Aug 31 04:33 .DS_Store
drwxr-xr-x  16 guiquadros  staff   512 Aug 31 04:34 .git
-rw-r--r--   1 guiquadros  staff  6042 Aug 29 23:59 .gitignore
-rw-r--r--   1 guiquadros  staff  1456 Aug 31 04:34 README.md
drwxr-xr-x   5 guiquadros  staff   160 Aug 31 04:30 exe
drwxr-xr-x   4 guiquadros  staff   128 Aug 30 00:00 source
guiquadros@iMac848 uninter-xor-crypto (master) $ mono exe/xor_cryptography.exe
Uninter - Matematica Computacional: AP Criptografia Simetrica com XOR
Autor: Guilherme Quadros da Silva

PRESSIONE UMA TECLA PARA INICIAR O PROGRAMA DE CRIPTOGRAFIA E DESCRIPTOGRAFIA.
```

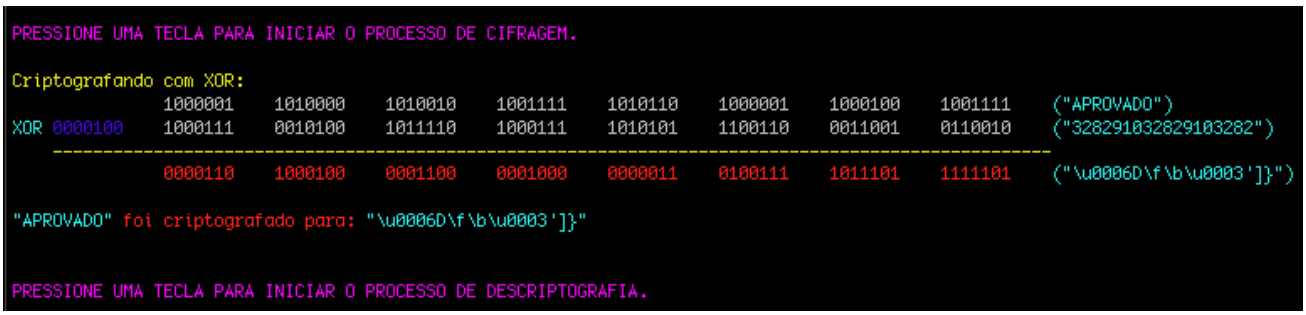
2. A primeira parte é a criptografia, que é iniciada percorrendo cada caracter da palavra "APROVADO" e obtendo seu valor na tabela ASCII em decimal e depois convertendo cada valor decimal para o seu correspondente em binário:

```
PRESSIONE UMA TECLA PARA INICIAR O PROGRAMA DE CRIPTOGRAFIA E DESCRIPTOGRAFIA.

Encriptando "APROVADO" com o RU "3282910"...

Buscando valores de "APROVADO" na tabela ASCII:
"{character}" = "{valor ASCII em decimal}" e "{valor ASCII em binario}"
       "A" =  "65 (10)" e "1000001 (2)"
       "P" =  "80 (10)" e "1010000 (2)"
       "R" =  "82 (10)" e "1010010 (2)"
       "O" =  "79 (10)" e "1001111 (2)"
       "V" =  "86 (10)" e "1010110 (2)"
       "A" =  "65 (10)" e "1000001 (2)"
       "D" =  "68 (10)" e "1000100 (2)"
       "O" =  "79 (10)" e "1001111 (2)"

"APROVADO" = "1000001101000010100101001111101011010000011000100100111 1 (2)"

PRESSIONE UMA TECLA PARA INICIAR O PROCESSO DE OBTENCAO DA CHAVE DE CRIPTOGRAFIA A PARTIR DO RU.
```

3. Em seguida é obtida a chave de cripografia a partir do RU "3282910". Como o número binário gerado convertendo "3282910" é muito pequeno é feita uma concatenação com cada dígio de "3282910" repetidas vezes até se chegar em uma chave suficientemente grande para cifrar a palavra "APROVADO" toda. A conversão é feita sempre no número resultado de uma vez só e não dígito por digíto, isso permite que a string cifrada gerada seja mais protegida do que em outra abordagens que poderiam utilizar de muitos zeros para a cifragem (como converter dígito a dígito do RU por exemplo).



4. O próximo passo é aplicar o operador XOR entre "APROVADO" em binário e a chave obtida. Note que os primeiros bits da chave não são usados na conversão (a chave obtida no passo anterior tem 59 bits enquanto a palavra "APROVADO" em binário tem 56 bits na conversão utilizada no passo 2).



5. A descriptografia segue os mesmos passos da criptografia, ela só passa a string cifrada ao invés de "APROVADO" para a mesma rotina, utilizando o mesmo RU ("3282910") como base para obtenção da chave. Abaixo a conversão de cada caracter cifrado para binário:

```
PRESSIONE UMA TECLA PARA INICIAR O PROCESSO DE DESCRIPTOGRAFIA.

Descriptando "\u0006D\f\b\u0003']}" com o RU "3282910"...
Buscando valores de "\u0006D\f\b\u0003']}" na tabela ASCII:
"{character}" = "{valor ASCII em decimal}" e "{valor ASCII em binario}"
  "\u0006" =   "6 (10)" e "0000110 (2)"
       "D" =  "68 (10)" e "1000100 (2)"
      "\f" =  "12 (10)" e "0001100 (2)"
      "\b" =   "8 (10)" e "0001000 (2)"
  "\u0003" =   "3 (10)" e "0000011 (2)"
       "'" =  "39 (10)" e "0100111 (2)"
       "]" =  "93 (10)" e "1011101 (2)"
       "}" = "125 (10)" e "1111101 (2)"

"\u0006D\f\b\u0003']}" = "00001101000100000110000010000000011010011110111011111101 (2)"

PRESSIONE UMA TECLA PARA INICIAR O PROCESSO DE OBTENCAO DA CHAVE DE CRIPTOGRAFIA A PARTIR DO RU.
```

6. Obtenção da chave a partir do número do RU seguindo a mesma lógica anterior:

```
PRESSIONE UMA TECLA PARA INICIAR O PROCESSO DE OBTENCAO DA CHAVE DE CRIPTOGRAFIA A PARTIR DO RU.

Obtendo a chave de criptografia pelo RU "3282910":
"3282910 (10)" = "1100100001011111011110 (2)"
1) Comparando o tamanho da chave "3282910 (10)" ("1100100001011111011110 (2)" [22 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   22 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("32829103 (10)" + "3 (10)") para obter uma chave maior:
   "32829103 (10)" = "1111101001110111010101111 (2)"
2) Comparando o tamanho da chave "32829103 (10)" ("1111101001110111010101111 (2)" [25 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   25 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("328291032 (10)" + "2 (10)") para obter uma chave maior:
   "328291032 (10)" = "10011100100010101001011011000 (2)"
3) Comparando o tamanho da chave "328291032 (10)" ("10011100100010101001011011000 (2)" [29 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   29 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("3282910328 (10)" + "8 (10)") para obter uma chave maior:
   "3282910328 (10)" = "11000011101011010011110001111000 (2)"
4) Comparando o tamanho da chave "3282910328 (10)" ("11000011101011010011110001111000 (2)" [32 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   32 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("32829103282 (10)" + "2 (10)") para obter uma chave maior:
   "32829103282 (10)" = "11110100100110001000101110010110010 (2)"
5) Comparando o tamanho da chave "32829103282 (10)" ("11110100100110001000101110010110010 (2)" [35 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   35 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("328291032829 (10)" + "9 (10)") para obter uma chave maior:
   "328291032829 (10)" = "100110001101111101010111001111011111101 (2)"
6) Comparando o tamanho da chave "328291032829 (10)" ("100110001101111101010111001111011111101 (2)" [39 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   39 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("3282910328291 (10)" + "1 (10)") para obter uma chave maior:
   "3282910328291 (10)" = "101111110001011100101101000011010111100011 (2)"
7) Comparando o tamanho da chave "3282910328291 (10)" ("101111110001011100101101000011010111100011 (2)" [42 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   42 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("32829103282910 (10)" + "0 (10)") para obter uma chave maior:
   "32829103282910 (10)" = "111011101101110011111000010100001101011011110 (2)"
8) Comparando o tamanho da chave "32829103282910 (10)" ("111011101101110011111000010100001101011011110 (2)" [45 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   45 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("328291032829103 (10)" + "3 (10)") para obter uma chave maior:
   "328291032829103 (10)" = "1001010101001010000110110011001010000110010101111 (2)"
9) Comparando o tamanho da chave "328291032829103 (10)" ("1001010101001010000110110011001010000110010101111 (2)" [49 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   49 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("3282910328291032 (10)" + "2 (10)") para obter uma chave maior:
   "3282910328291032 (10)" = "1011101010011100101000011111111001001111111011011000 (2)"
10) Comparando o tamanho da chave "3282910328291032 (10)" ("1011101010011100101000011111111001001111111011011000 (2)" [52 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   52 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("32829103282910328 (10)" + "8 (10)") para obter uma chave maior:
   "32829103282910328 (10)" = "1110100101000011110010100111111011110001111010001111000 (2)"
11) Comparando o tamanho da chave "32829103282910328 (10)" ("1110100101000011110010100111111011110001111010001111000 (2)" [55 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   55 >= 56? --> Nao! O tamanho da chave obtida nao e suficiente para criptografar com XOR a string "\u0006D\f\b\u0003']}".
   Concatenando a chave com o proximo digito do RU ("328291032829103282 (10)" + "2 (10)") para obter uma chave maior:
   "328291032829103282 (10)" = "10010001110010100101111010001111010101110011000110010110010 (2)"
12) Comparando o tamanho da chave "328291032829103282 (10)" ("10010001110010100101111010001111010101110011000110010110010 (2)" [59 bits]) obtida com a string que vai ser criptografada "\u0006D\f\b\u0003']}" ("00001101000100000110000010000000011010011110111011111101 (2)" [56 bits]).
   59 >= 56? --> Sim!

Chave obtida = "10010001110010100101111010001111010101110011000110010110010 (2)"


PRESSIONE UMA TECLA PARA INICIAR O PROCESSO DE CIFRAGEM.
```

7. O resultado final usando o operador XOR, provando que é possível obter a string "APROVADO" novamene a partir da string cifrada.

```
PRESSIONE UMA TECLA PARA INICIAR O PROCESSO DE CIFRAGEM.

Criptografando com XOR:
              0000110   1000100   0001100   0001000   0000011   0100111   1011101   1111101   ("\u0006D\f\b\u0003']}")
XOR 0000100   1000111   0010100   1011110   1000111   1010101   1100110   0011001   0110010   ("328291032829103282")
   ------------------------------------------------------------------------------------------
              1000001   1010000   1010010   1001111   1010110   1000001   1000100   1001111   ("APROVADO")

"\u0006D\f\b\u0003']}" foi descriptografado para: "APROVADO"


PRESSIONE QUALQUER TECLA PARA FECHAR A EXECUCAO DO PROGRAMA.
```