Trabalho de Programação 2

Processador CESAR

1. Descrição Geral

Você deverá desenvolver um programa em assembler do CESAR para decriptar uma frase, usando a cifragem de Affine (http://en.wikipedia.org/wiki/Affine_cipher).

A frase a ser decriptada é chamada de "texto cifrado" e o resultado da decriptação é o "texto claro" (ou texto decriptado).

A frase a ser decriptada deverá ser fornecida pelo usuário através do teclado. O mesmo pode acontecer com o parâmetro "*a*", necessário para o processo de decodificação.

Para realizar a decriptação o programa deve escolher valores para os parâmetros "a" e "b". Inicialmente, o programa atribui os valores iniciais aos dois parâmetros e realiza a decriptação, fornecendo o resultado no visor.

Então, o usuário pode decidir como o programa deverá se comportar: (1) se deve passar, automaticamente, para uma nova combinação de valores de "a" e "b" e fornecer no visor a decriptação correspondente ou (2) se deve solicitar ao usuário que altere o valor dos parâmetros para então fornecer no visor a decriptação correspondente.

Essa implementação corresponde ao processo de criptoanálise (ou quebra de código) empregando a técnica de "força bruta", onde todas as combinações possíveis são tentadas, até que se encontre aquela que fornece um texto claro com significado.

2. Especificação do Trabalho

O texto cifrado será formado apenas pelas letras maiúsculas, representadas pelos seus códigos ASCII ('A'=41H, 'B'=42H, ..., 'Z'=5AH). Existem 26 letras maiúsculas na tabela ASCII.

Para determinar o valor de cada caractere do texto claro deverá ser utilizado o seguinte procedimento:

1) Converter o código ASCII das letras 'A', 'B', ... 'Z' do texto cifrado em números 0, 1, ..., 25. Esses números serão colocados em um vetor chamado de "*Y*".

2) Considerando-se "*i*" como a posição do número "y_i" no vetor "*Y*", calcula-se o número "x_i", a ser colocado na posição "*i*" de um vetor "*X*". O cálculo a ser feito é o seguinte (conforme a decifragem de Affine):

$$x_i = a^{-1}(y_i - b) \ mod \ 26$$

"*mod* 26" representa o resto da divisão por 26.
Assim, "*n mod* 26" é o resto da divisão de "*n*" por 26.

3) Converter os números do vetor "*X*" (valores que estão entre 0 e 25) para o código ASCII das letras 'A', 'B', ... 'Z'. O resultado dessa operação será o texto claro.

Detalhamento da Decifragem

A expressão de cálculo da decifragem é obtida pela inversão da expressão de cifragem, conforme abaixo:

$$\begin{aligned} y_i &= (a.x_i + b) \ mod \ 26 \\ (y_i - b) &= (a \ x_i) \ mod \ 26 \\ a^{-1} (y_i - b) &= a^{-1} (a \ x_i) \ mod \ 26 \\ a^{-1} (y_i - b) &= a^{-1} \ a \ (x_i) \ mod \ 26 \end{aligned}$$

Nesse ponto percebe-se que só é possível isolar o valor de "x_i" se "$1 = a.a^{-1}$", com essa multiplicação sendo realizada em módulo 26. Dessa forma, se "$1 = a.a^{-1} \ mod \ 26$", a expressão final torna-se a expressão de decifragem de Affine:

$$x_i = a^{-1} (y_i - b) \ mod \ 26$$

Sendo assim, apesar do programa receber e apresentar o valor do parâmetro *a*, a expressão de decodificação utiliza a^{-1}. Logo, o programa deve determinar o valor de "a^{-1}" a partir do valor de "*a*" escolhido, satisfazendo a seguinte expressão:

$$1 = a.a^{-1} \ mod \ 26.$$

Entretanto, a análise dessa expressão revela que ela só pode ser satisfeita para alguns valores de *a*. Ou seja, existem valores de "*a*" para os quais não é possível encontrar um valor para "a^{-1}" que tornem verdadeira a expressão (para

verificar isso, por exemplo, tente encontrar o valor de "a^{-1}" para "a=2"). Os valores de "a" que admitem "a^{-1}", segundo a expressão, estão listados na Tabela 1 a seguir.

a	1	3	5	7	9	11	15	17	19	21	23	25
a^{-1}	1	9	21	15	3	19	7	23	11	5	17	25

Tabela 1 – Valores válidos de "a" e os correspondentes a^{-1}

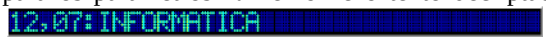
3. Procedimentos do programa

A seguir é apresentada a sequência de procedimentos a serem implementados no programa:

0,5 [1] Ao iniciar, o programa deve apresentar no visor a identificação do aluno (nome e número do cartão).
Deve aguardar digitar uma tecla para prosseguir

2,0 [2] O programa deve solicitar que o usuário digite o texto cifrado.
   a. Deve ser colocada uma mensagem no visor informando que o usuário deve digitar o texto cifrado.
   b. Assim que o usuário iniciar a digitar o texto, o visor deve ser apagado e o texto digitado deve ser colocado a partir da posição mais à esquerda do visor.
   c. O texto cifrado pode ter no máximo 30 caracteres. O programa não pode permitir a digitação de mais de 30 caracteres.
   d. O usuário deve digitar o texto cifrado usando apenas letras, que serão apresentadas no visor.
   e. Letras minúsculas devem ser convertidas para maiúsculas.
   f. O programa deve permitir o uso do <BS> (back-space) para eventuais correções.
   g. Ao encerrar a digitação do texto cifrado o usuário deverá digitar <ENTER>.
   h. Outros caracteres devem ser ignorados.
Sempre será fornecido texto!
Jamais entrará ENTER sem caracteres

[3] Então, o programa deve atribuir o valor "1" para o parâmetro "a" e o valor "0" (zero) para o parâmetro "b".

4,0 [4] Com os valores selecionados para os parâmetros "a" e "b", o programa deve realizar a decriptação e apresentar o resultado no visor.
   a. No visor devem ser apresentados, em ordem, os valores dos parâmetros "a" e "b" e o texto decriptado.
   b. Por exemplo, se tivermos os valores 12 e 7 para os parâmetros "a" e "b" e o texto decriptado for "INFORMATICA", no visor será apresentado: 

[5] Então, o programa deve aguardar que o usuário digite uma tecla de comando.
   a. As teclas não devem ser ecoadas (escritas) no visor, apenas lidas do teclado.
   b. As teclas de comando são: <ENTER>, "A", "a", "N", "n", "F" ou "f". As outras teclas devem ser ignoradas.

0,5 [6] Se for digitado <ENTER>, o programa deve alterar os valores de "a" e "b" para o próximo par de valores válidos. Então, deve apresentar o novo texto decriptado.
   a. O valor de "b" deve ser incrementado. Se ultrapassar 25, deverá ser zerado (circular do 25 para o zero).
   b. Sempre que o parâmetro "b" passar de 25 para zero, o parâmetro "a" deve ser incrementado, passando para o próximo valor válido, de acordo com a Tabela 1.
   c. Se o valor de "a" for o último da tabela, ao incrementá-lo ele deverá circular para o primeiro valor da Tabela 1 (no caso, "1").
   d. Então, o programa deve continuar no passo [4].

3,0 [7] Se for digitado "A" ou "a", o programa deve solicitar ao usuário que forneça, através do teclado, um novo valor para o parâmetro "a".
   a. O programa deve colocar mensagem no visor, solicitando o novo valor de "a".
   b. Na entrada do novo valor devem ser usados apenas números e deve ser tratado o BS (*back-space*), para eventuais correções.
   c. Ao final da entrada do novo valor, o usuário deverá digitar <ENTER>.
   d. Depois de confirmado o novo valor, o programa deverá validá-lo. Caso o usuário entre um valor diferente dos valores válidos (valores de "a" da Tabela 1), o programa deve colocar uma mensagem informando esse fato, aguardar que o usuário digite qualquer tecla, e retornar para o passo [4].

[8] Se for digitado "N" ou "n", o programa deverá solicitar novo texto a ser decriptado. Para isso, o programa deve retornar para o passo [2].

[9] Finalmente, se for digitado "F" ou "f", o programa deverá ser encerrado.

4. Programa "minimamente operacional"

O programa será considerado minimamente operacional se for capaz de ler o texto cifrado e listar o resultado da decifragem, segundo a Cifra de Cesar (A Cifra de Cesar é obtida através da Cifra de Affine com *a*=1 e *b*=3).

Por exemplo, o texto cifrado "**`LQIRUPDWLFD`**" (com *a*=1 e *b*=3) deve fornecer como resultado o texto claro "**`INFORMATICA`**".

5. Entregáveis: o que deve ser entregue?

A entrega do trabalho será realizada em duas partes:

- a primeira entrega é uma implementação em "C";
- a segunda entrega é a implementação em assembler.

Primeira entrega

Nessa primeira entrega deve ser enviado um único arquivo em "C". Nessa implementação não é necessário seguir, rigorosamente, as especificações de entrada e saída. Entretanto, as funcionalidades especificadas no trabalho devem estar implementadas:

- Possibilidade de entrar o texto cifrado, via teclado;
- Escolher valores de *a* e *b*, via teclado;
- Apresentar o resultado da decifragem na tela, usando os valores de *a* e *b*;

Segunda entrega

Devem ser entregues dois arquivos: o arquivo fonte (.CED) e um relatório em formato PDF.

A implementação entregue no arquivo .CED deve seguir, rigorosamente, a especificação do trabalho no que diz respeito aos procedimentos de entrada e saída de dados pelo teclado e pelo visor, respectivamente.

Para gerar arquivos PDF a partir de qualquer editor no Windows, pode-se usar o "primo pdf" (http://www.primopdf.com/) ou o "doPDF" (http://www.dopdf.com/br/).

No relatório da implementação devem estar presentes os seguintes elementos:

- Identificação do aluno;
- Principais dificuldades encontradas;
- Descrição do programa implementado (pode ser feita sobre a implementação em "C").

Caso tenha sido necessário alterar o programa "C" ou este não tenha sido entregue na primeira entrega, o fonte "C" deverá ser entregue na segunda entrega.

Observação: não colocar a listagem completa dos programas no relatório.

6. Avaliação

O trabalho só será considerado entregue se forem enviados os arquivos solicitados e o programa estiver **minimamente operacional**.

O trabalho será avaliado da seguinte forma:

- 10,0 pontos: entrega do arquivo "C";
- 20,0 pontos: entrega e avaliação do relatório;
- 70,0 pontos: correção da operação da implementação em assembler.

O arquivo ".C" deverá ser entregue até a **data prevista para a primeira entrega**. A não entrega desse arquivo na data prevista acarretará na perda dos pontos de avaliação do arquivo "C".

A implementação final do trabalho deverá ser entregue até a **data prevista para a segunda entrega**. Admite-se essa entrega com até uma semana de atraso. Nesse caso a nota final do trabalho será obtida pela diminuição de 20,0 pontos (de um total de 100,00) da nota alcançada na avaliação do mesmo. Não serão aceitos trabalhos entregues além dos prazos estabelecidos.

7. Observações

Recomenda-se a troca de ideias entre os alunos. Entretanto, a identificação de cópias de trabalhos acarretará na aplicação do Código Disciplina Discente e a tomada das medidas cabíveis para essa situação.

O professor da disciplina reserva-se o direito, caso necessário, de solicitar uma demonstração do programa, onde o aluno será arguido sobre o trabalho como um todo. Nesse caso, a nota final do trabalho levará em consideração o resultado da demonstração.