ICC 2 – Trabalho 04

Prof. Moacir Ponti (Eng. Comp.) e Rodrigo Mello

Estagiários PAE

Martha Dais Ferreira

Fausto Guzzo da Costa

Gabriela Thumé (Eng. Comp.)

Data Máxima para Perguntas: 25/09/2014

Data de Entrega do Trabalho: 29/09/2014

Fórum para Envio de Perguntas: https://groups.google.com/d/forum/icc2-2014

Descrição

Implemente o método de criptografia descrito a seguir. Suponha a seguinte mensagem criptografada:

```
>ydobbr,?
```

Agora considere a seguinte tabela para a conversão desses caracteres (utilize a mesma em seu programa):

	a	b	c	d	e	f	g	h	i	j	k	l	m	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
o	p	q	r	s	t	u	v	w	x	y	z	A	B	C
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
S	T	U	V	W	X	Y	Z	0	1	2	3	4	5	6
45	46	47	48	49	50	51	52	53	54	55	56	57	58	59
7	8	9	:	;	<	=	>	?	@	!	“	#	$	%
60	61	62	63	64	65	66	67	68	69	70	71	72	73	74
&	'	(	)	*	+	,	-	.	/	[	\	]	_	{
75	76	77	78	79	80	81	82	83	84	85	86	87	88	89
|	}													
90	91													

Quebre essa mensagem criptografada em pedaços de tamanho k, por exemplo, para k = 3 e convertendo cada pedaço para uma coluna de números usando a tabela acima temos a matriz:

$$Y = \begin{bmatrix} 67 & 15 & 18 \\ 25 & 2 & 81 \\ 4 & 2 & 68 \end{bmatrix}$$

Sabe-se que a mensagem original X foi criptograda realizando um produto pela matriz M:

$$M = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Ou seja:

$$X\ M = Y$$

Assim, para convertermos novamente a mensagem criptografada Y na original X, precisamos resolver:

$$X = M^{-1}\ Y$$

Ou seja, se fizermos o produto da matriz inversa de M, i.e., M^{-1} pela mensagem criptograda Y, retornamos para a mensagem original.

Seu programa irá receber como entrada o nome de um arquivo que contém um texto que sempre formará uma matriz quadrada com k linhas e k colunas e, em seguida, a matriz M que também tem dimensões k x k.

No caso acima, a mensagem retornada seria:

Tudo bem?

Dicas

1) Leia as páginas 88 a 90 do livro https://www.dropbox.com/s/aa71ogpk8xski1j/gaalt1.pdf;
2) A maneira de se encontrar a matriz inversa M^{-1} é apresentada no livro https://www.dropbox.com/s/aa71ogpk8xski1j/gaalt1.pdf nas páginas 77 a 82;
3) Implemente com alocação dinâmica!

Motivação

1) Exemplo simples de criptografia simétrica, que motiva a compreensão futura de algoritmos como TLS, DES, RSA, SHA, MD5, etc;
2) O livro http://www.fmf.uni-lj.si/~lavric/Tattersall%20-%20Elementary%20number%20theory%20in%20nine%20chapters.pdf (Elementary Number Theory In Nine Chapters por James J. Tattersall) traz diversas motivações sobre criptografia em seu Capítulo 7. Diversos usos de algoritmos similares a este do trabalho eram utilizados pelos Babilônios. VALE MUITO A PENA LER AS PÁGINAS 210 A 218 !!!!!!!!!!!!!!!!!!!

Formato para os Casos de Teste

```
Formato da entrada:
<nome de um arquivo com o texto>
<valor na linha 1 coluna 1>
<valor na linha 1 coluna 2>
...
```

```
<valor na linha 1 coluna k>
<valor na linha 2 coluna 1>
<valor na linha 2 coluna 2>
...
<valor na linha 2 coluna k>
...
<valor na linha k coluna 1>
<valor na linha k coluna 2>
...
<valor na linha k coluna k>
```

Formato da saída:
<texto retornado>

Exemplo de Caso de Teste

Entrada fornecida:
arquivo.txt
1
1
0
0
1
1
0
0
1

OBS: O conteúdo do arquivo será ">ydobbr,?"

Saída esperada:
Tudo bem?

Informações Importantes

1) Um dos objetivos da disciplina de ICC2 é o aprendizado individual dos conceitos de programação. A principal evidência desse aprendizado está nos trabalhos, que são individuais neste curso. Você deverá desenvolver seu trabalho sem copiar trechos de código de outros alunos ou da Internet, nem codificar em conjunto. Portanto, compartilhem ideias, soluções, modos de resolver o problema, mas não o código.

1.1) O plágio vai contra o código de ética da USP;

1.2) Quando autores e copiadores combinam, estão ludibriando o sistema de avaliação, por isso serão utilizadas ferramentas de análise plágio tais como o MOSS (http://moss.stanford.edu/);

1.3) O trabalho em grupo e a cooperação entre colegas é em geral benéfico e útil ao aprendizado. Para ajudar um colega você pode lhe explicar estratégias e ideias. Por exemplo, pode explicar que é preciso usar dois loops para processar os dados, ou que para poupar memória basta usar uma certa estrutura de dados, etc. O que você não deve fazer é mostrar o seu código. Mostrar/compartilhar o código pode prejudicar o aprendizado de seu colega:

1.3.1) Depois seu colega ver seu código, será muito mais difícil para ele imaginar uma solução original e própria;

1.3.2) O seu colega não entenderá realmente o problema: a compreensão passa pela prática da codificação e não pela imitação/cópia.

1.4) Um colega que tenha visto a sua solução pode eventualmente divulgá-la a outros colegas, deixando você numa situação muito complicada, por tabela;

1.5) O texto acima foi baseado e adaptado da página http://www.ime.usp.br/~mac2166/plagio/, da qual recomendo a leitura completa.

2) Todos os códigos fontes serão comparados por um (ou mais) sistema(s) de detecção de plágio, e os trabalhos com alta similaridade detectada terão suas notas zeradas, tanto aqueles relativos ao código de origem quanto do código copiado. A detecção de plágio será reportada à Seção de Graduação para providências administrativas;
3) A avaliação incluirá a porcentagem de acertos verificada pelo Sistema de Submissão de Trabalhos e também a análise do seu código, incluindo identação, comentários, bom uso da memória e práticas de programação. Portanto faça seu código com cuidado, da melhor forma possível.

Sobre o sistema de submissão:

1. Seu código deverá incluir arquivo fonte .c .h e Makefile – todos os arquivos deverão obrigatoriamente conter no início um comentário com seu nome, número USP, turma e data da entrega do trabalho;

2. A data/hora de entrega do trabalho é aquela estipulada no sistema. Trabalhos entregues por email NÃO serão aceitos, mesmo que dentro da data/hora estipulada. Faça seu trabalho com antecedência para evitar entregar em cima da hora e ter problemas de submissão;

2.1. A submissão é de responsabilidade do aluno, e os problemas comuns à entrega próxima ao fechamento do sistema também. Portanto: problemas de acesso à rede não serão aceitos como desculpa para entrega por email ou fora do prazo;

3. A compilação e execução do código é feita no sistema pelos comandos:

make all
make run

4. A saída do seu programa deve ser exatamente igual à saída esperada, incluindo: espaços em branco, quebras de linha e precisão decimal;

5. Há um limite em segundos para a execução dos casos de teste e um limite de memória total para ser utilizado. Você deverá gerenciar bem o tempo de execução e o espaço ocupado para que seu programa fique dentro desses limites, para evitar uso excessivo, pois o sistema irá invalidar o caso em que o limite foi excedido;

6. Ao enviar, aguarde sem recarregar a página nem pressionar ESC, para obter a resposta. Caso demore mais do que 2 minutos para dar uma resposta, feche e abra novamente a página do servidor. Verifique se seu programa tem algum problema de entrada de dados (ou se tem algum loop infinito ou parada para pressionamento de tecla). Caso não tenha, aguarde alguns minutos e tente novamente;

7. O erro de “Violação de Memória” significa acesso indevido a arranjo ou arquivo. Use compilação com g e o programa valgrind para tentar detectar a linha de código com erro.

7.1. Exemplo 1 de violação de memória:

```
int **mat = (int **)malloc(sizeof(int *) * 3);
```

```
mat[0] = (int *)malloc(sizeof(int)*10);
for (i = 1; i < 3; i++) {
   free(mat[i]);
}
// apenas a posicao 0 de mat foi alocada as outras nao
// portanto esta liberando regiao nao alocada
// gerando violacao de memoria
```

7.2) Exemplo 2 de violação de memória:

```
int B[5] = {5, 6, 7, 8, 9};
int N = 5;
int *A = malloc(N*sizeof(int));
int j = 0, i = 1; // 'j' inicializado em 0, 'i' inicializado em 1
while (j < N){
A[i] = B[j]; // 'j' e 'i' sao indices diferentes
j++;         // quando j = (N1) i = (N1)+1 e
i++;
     // havera escrita indevida em A
}
```