Primeiro Trabalho Prático

Neste trabalho prático de Introdução à Criptografia, cada aluno (individualmente) deverá apresentar um programa completo de sua autoria para implementar a cifragem e decifragem simulando uma máquina de rotor.

O Problema

Nós vimos em aula o funcionamento das máquinas de rotor, que são basicamente uma sequência de substituições monoalfabéticas usando permutações distintas (cada uma é um rotor). Após um símbolo ter sido cifrado, uma ou mais das permutações são modificadas, fazendo uma rotação por um elemento à esquerda ou à direita.

Um dos maiores problemas práticos em usar este tipo de criptografia é que a chave é grande e difícil de ser memorizada. No caso da criptografia clássica, deve-se memorizar uma permutação das letras de A a Z para cada rotor. Já para o mundo digital, cada permutação contém todos os números entre 0 e 255.

Neste trabalho, os rotores terão 256 posições, uma para cada possível byte e usaremos uma estratégia diferente para o usuário memorizar facilmente a chave. O usuário fornece uma frase qualquer com até 256 bytes. Então, executaremos a fase de inicialização do RC4 usando a frase escolhida pelo usuário. Ao final disto, o vetor S do RC4 contém uma permutação dos números de 0 a 255, que sera usada para inicializar um rotor.

Para relembrá-los, o RC4 tem a seguinte inicialização, na qual K contém a frase escolhida pelo usuário e tamk será o tamanho da frase (`strlen(frase)`). Cada caractere da frase é visto como um inteiro de 0 a 255. Use o conteúdo de S como a permutação inicial do rotor.

```
Seja S um vetor de 256 inteiros (de 0 a 255 cada)
Seja K um vetor com tamk números (de 0 a 255 cada)

para i de 0 a 255 faça
  S[i] = i;

j = 0;
para i de 0 a 255 faça
  j = (j + S[i] + T[i mod tamk]) mod 256;
  troque S[i] com S[j]
```

Cada rotor será inicializado com uma frase distinta, fornecida pelo usuário. O número de rotores vai de 1 a 5. Além disso, para cada rotor o usuário determina dois valores k e l. A cada k símbolos cifrados, o rotor é movido l posições à direita (o valor 0 mantém o rotor parado). Os valores permitidos para k vão de 1 a 10^9 e, para l, de 0 a 255.

Note que não é preciso rotacionar os elementos do rotor (que provavelmente estarão em um vetor) cada vez que k símbolos forem cifrados. Basta armazenar em uma variável, digamos p, quantas posições o rotor foi deslocado (à direita) até o momento. Então para cifrar um símbolo x pode-se cifrar o símbolo $x + p \bmod 256$ usando o rotor original.

Já para a decifragem, lembre-se que as permutações dos rotores devem ser invertidas e que os rotores devem ser aplicados do último para o primeiro (na ordem inversa da cifragem).

O mesmo argumento da cifragem se aplica à decifragem, não sendo necessário rotacionar e inverter rotores repetidamente. Apenas armazena-se a o número de posições p (à direita) que o rotor foi deslocado e para decifrar y, primeiro se aplica a permutação inversa do rotor original, obtendo o símbolo x, e depois se calcula o símbolo original fazendo $x - p \bmod 256$.

Em ambos os processos, se p ficar maior que 256, pode-se fazer $p = p \bmod 256$, para evitar problemas numéricos.

Seu algoritmo deverá ser capaz de cifrar e decifrar qualquer arquivo, incluindo binários como imagens, sons, executáveis, etc.

Solução

Você deve produzir um programa chamado rotor, escrito em linguagem C, C++ ou Python 3 (nenhuma outra será aceita) e compilado e executado em uma máquina Linux. Seu programa será executado da seguinte forma:

Seu programa será compilado da seguinte forma (para C e C++):

```
gcc -std=c17 -pedantic -Wall -o rotor *.c
g++ -std=c++17 -pedantic -Wall -o rotor *.cpp
```

E executado como (para compilados, ou interpretado via Python 3.9):

```
./rotor modo n frase1 frase2 frasen k1 l1 k2 l2 kn ln entrada saida
python3 rotor.py modo n frase1 frase2 frasen k1 l1 k2 l2 kn ln entrada saida
```

Os parâmetros são os seguintes:

- modo: Identifica a operação a ser realizada, C para cifragem e D para decifragem;
- n: Número de rotores, de 1 a 5;
- frase1 a frasen: As frases usadas para inicializar os rotores de 1 a n;
- k1 a kn: A cada quantos símbolos os rotores se movem;
- l1 a ln: Quantas posições os rotores se movem;
- entrada: Nome do arquivo de entrada para cifragem ou decifragem; e
- saida: Nome do arquivo de saida para cifragem ou decifragem.

Antes de executar a cifragem, imprima o conteúdo inicial de cada rotor. Identifique o rotor com uma linha contendo o modo de uso, o número do rotor e os valores da frase, tamk, l e k para aquele rotor.

Para imprimir o estado do rotor, imprima uma matriz 16×16, preenchida da esquerda para direita e de cima para baixo, com o valor de cada posição do rotor. Para melhor apresentação visual, imprima os números sempre com 3 posições e um espaçamento entre eles.

No caso de decifragem, imprima os rotores já invertidos.

Por exemplo, considere a seguinte linha de comando:

```
./rotor C 2 GIROSCOPIO BAUNILHA 1 1 17 3 carta.txt carta.bin
```

Seu programa deve cifrar o conteúdo do arquivo carta.txt, armazenando o resultado em carta.bin usando dois rotores. O primeiro rotor é inicializado usando a frase GIROSCOPIO e se move 1 posição à direita a cada símbolo cifrado. Já o segundo rotor é inicializado com a frase BAUNILHA e move-se 3 posições à direita a cada 17 símbolos cifrados.

O rotor inicializado com a palavra BAUNILHA para cifragem deve ser impresso como segue:

```
C 1 BAUNILHA 1 1
206  58  47 151 121 104 124  52 110  50  72  32 132 209  30  60
242 213 172 108 174  10 136 180  94 130  70 232 245   6  29  97
 80  39  41 154  91  87  55  27 173  82 101 229  63 184  46 159
223 139 244  74  36  22 252 126 248 135  12 113  85 152 142 237
228 235 187  76 217 233  56 128 119 188 131 204  88  26 111 146
 16  64  89 177 249 239 141 120  59   0 246 103 153  57  19  45
227 171  13   1   4 122 179  96  67 175 133  14 182 149 220  71
155 215 147   7 164 210 144  83 105  17  37  78  28 167 192 253
193  25  61  53 211 163 225 114 186  43 255 216  73 168 176 222
150 202 247 250  54 224 243 221 198  20 212 189 162 129 161 226
199  35  44 148 160 181  31  84 127 208 145 195 165  62 201 190
  5  42 157 241 109 205  40 143  38 115 254 214   2 194 238 236
230 125 158  21 185   8 183 196 219 231 137 117  34 234  15  48
251  90  75 203  79 116  18 134  33 207  98   9  11 156  68  99
200  51 112 178  93 118 170  77   3 106  92 102 140 218 107  66
166 169  49  23  69 191 123 240  65 100  81  86  24  95 138 197
```

O mesmo rotor, inicializado para a decifragem fica como abaixo:

```
D 1 BAUNILHA 1 1
 89  99 188 232 100 176  29 115 197 219  21 220  58  98 107 206
 80 121 214  94 153 195  53 243 252 129  77  39 124  30  14 166
 11 216 204 161  52 122 184  33 182  34 177 137 162  95  46   2
207 242   9 225   7 131 148  38  70  93   1  88  15 130 173  44
 81 248 239 104 222 244  26 111  10 140  51 210  67 231 123 212
 32 250  41 119 167  60 251  37  76  82 209  36 234 228  24 253
103  31 218 223 249  42 235  91   5 120 233 238  19 180   8  78
226  59 135 185 213 203 229  72  87   4 101 246   6 193  55 168
 71 157  25  74  12 106 215  57  22 202 254  49 236  86  62 183
118 170  79 114 163 109 144   3  61  92  35 112 221 178 194  47
164 158 156 133 116 172 240 125 141 241 230  97  18  40  20 105
142  83 227 102  23 165 108 198  45 196 136  66  73 155 175 245
126 128 189 171 199 255 152 160 224 174 145 211  75 181   0 217
169  13 117 132 154  17 187 113 139  68 237 200 110 151 143  48
149 134 159  96  64  43 192 201  27  69 205  65 191  63 190  85
247 179  16 150  50  28  90 146  56  84 147 208  54 127 186 138
```

O rotor inicializado com a palavra GIROSCOPIO para cifragem deve ser impresso como segue:

```
C 1 GIROSCOPIO 1 1
 59  27 178  80   8 212 165 177 211   0  48 208 104 147 235 132
240 152 175 168  21  83 218 238 102 130  82 219  72  57  37 121
232 189  45  54 169 253 141 151  13 228  96 159 204 167 229 129
  5 143 108 111 245  31 239 244 217  40 119 123 201 158 230 116
227 103 183 139 128  29 110 150  23 180  81 223  79  64 174  49
 28  65 246 188  99 251  17  15 233 185 120 100  84  10   4 250
 36 207 140 226 181  73 193 118 144   9 149   3 225 157  67 153
 98 190  70  35 125  41  76 155  97 187 213 127  61 221  69 173
154 194 101  44  24  87 148  11 249 210 105  19  38 214 137 106
161  56 198 136  33 234   7  22  30  16  47 241  42  89  86 184
195  32  63 236 200 166  92 135  85  52  62 179  74 222  14 160
 43  25  12 196   6 192 170 206  20 199 172 216  66  91 163 255
176  68 171  88 131   1 209 220 182 215  50 138  34 254 117  90
107 247 224 162  26  53  95 242  93 115  55 145  71  75 133 142
237 243  18  39 248  60 231 191  78 134 113  46 146 126  77   2
202  51 205 156  94 203 252 122 114  58 124 186 112 109 197 164
```

O mesmo rotor, inicializado para a decifragem fica como abaixo:

```
D 1 GIROSCOPIO 1 1
  9 197 239 107  94  48 180 150   4 105  93 135 178  40 174  87
153  86 226 139 184  20 151  72 132 177 212   1  80  69 152  53
161 148 204 115  96  30 140 227  57 117 156 176 131  34 235 154
 10  79 202 241 169 213  35 218 145  29 249   0 229 124 170 162
 77  81 188 110 193 126 114 220  28 101 172 221 118 238 232  76
  3  74  26  21  92 168 158 133 195 157 207 189 166 216 244 214
 42 120 112  84  91 130  24  65  12 138 143 208  50 253  70  51
252 234 248 217  63 206 103  58  90  31 247  59 250 116 237 123
 68  47  25 196  15 222 233 167 147 142 203  67  98  38 223  49
104 219 236  13 134 106  71  39  17 111 128 119 243 109  61  43
175 144 211 190 255   6 165  45  19  36 182 194 186 127  78  18
192   7   2 171  73 100 200  66 159  89 251 121  83  33 113 231
181 102 129 160 179 254 146 185 164  60 240 245  44 242 183  97
 11 198 137   8   5 122 141 201 187  56  22  27 199 125 173  75
210 108  99  64  41  46  62 230  32  88 149  14 163 224  23  54
 16 155 215 225  55  52  82 209 228 136  95  85 246  37 205 191
```

Algumas dicas para C/C++. Os parâmetros que o usuário digitou na linha de comando podem ser consultados usando os argumentos `argc` e `argv` da função `main`. Abra os arquivos de entrada e saída em modo binário `fopen(nome, "rb")` e leia (ou escreva) um byte por vez com `fgetc` e `fputc`. O problema em usar `fscanf` e `fprintf` é que as cadeias de símbolos podem conter o byte 0, que é confundido com o fim da cadeia. Se desejar ler e escrever mais de um byte por vez, use `fread` e `fwrite`.

Provavelmente, situações semelhantes ocorrem em Python. Tome cuidado e pesquise soluções. As funções `ord` e `chr` são úteis para converter entre caracteres e inteiros.

Entrega e Avaliação

A programa deve ser entregue pelo Ava até às 23:55 do dia 24/10/2023 por um arquivo compactado contendo todos os arquivos fontes.

A minha opinião sobre seu código pode ser influenciada pelo quantidade e gravidade dos avisos que o compilador emitir.

A avaliação considerará o código fonte e sua execução correta. Comente claramente o que cada trecho de código faz. Quanto mais claro e simples, melhor. A nota vai de 0,0 a 10,0 sendo que 3 pontos correspondem a avaliação do código fonte e 7 pontos a completude da tarefa, distribuídos da forma abaixo:

Condição	Pontuação
Inicializa o rotor para cifragem	1,0
Cifra com 1 rotor e $k = 1$ e $l = 1$	1,0
Cifra com 1 rotor e quaisquer k e l válidos	0,5
Cifra com múltiplos rotores e quaisquer k e l válidos	1,0
Inicializa o rotor para decifragem	1,0
Decifra com 1 rotor e $k = 1$ e $l = 1$	1,0
Decifra com 1 rotor e quaisquer k e l válidos	0,5
Decifra com múltiplos rotores e quaisquer k e l válidos	1,0