Universidade de Brasília – UnB
Instituto de Ciências Exatas – IE
Departamento de Ciência da Computação – CIC
CIC0201 - Segurança da Computação - 2023/2
Professora: João Gondim
Aluno: João Vitor Abadio Siqueira

Descritivo - Trabalho de Implementação 1 - Cifra de Vigenère

Para compilar o programa:

gcc vigenere.c -o vigenere

Para executar:

./vigenere

int main():

É responsável por gerenciar os ponteiros para arquivos, a chave e o menu. Os ponteiros para arquivos são três:

- original_plainfile: aponta para um arquivo que contém texto claro
- formated_plainfile: aponta para um arquivo com texto claro formatado
- cipherfile: aponta para um arquivo com texto cifrado

O menu é percorrido com a variável option, a chave é armazenada na *string* key e os nomes dos arquivos na *string* filename.

As opções do menu são:

1. Criptografar um arquivo
2. Descriptografar um arquivo
3. Descobrir a chave e descriptografar um arquivo
4. Sair

Ao selecionar as opção “1” ou “2” será requerido o nome do arquivo de texto para criptografar/descriptografar e a chave, que é formatada para conter apenas letras maiúsculas. A opção “1” tem a particularidade de converter os caracteres do arquivo de texto claro para letra maiúscula e armazenar esse texto claro formatado em um novo arquivo.

Ao selecionar a opção “3” será requerido apenas o nome do arquivo de texto cifrado.

void menu(char*):

Imprime na tela o menu e espera pelo input do usuário entre as opções de 1 a 4 e armazena o input no endereço do parâmetro char*.

void format_plaintext(FILE*):

Recebe o ponteiro para o texto claro. Formata o texto claro para que todas as letras sejam maiúsculas e cria um novo arquivo para armazenar o resultado. Esse novo arquivo é chamado formated_input.txt.

char* format_key (char*):

Recebe a chave. Formata a chave para conter apenas letras maiúsculas e retorna o resultado.

void encrypt_file(FILE*, char*):

Recebe um ponteiro para um arquivo de texto claro e chave. Pede ao usuário o nome do novo arquivo cifrado. Lê cada caractere do arquivo e checa se o carácter é alfabético ou não, fazendo o seguinte:

- É alfabético:
  - $c_i = (m_i + k_i) \bmod 26$, onde c_i é o caractere cifrado, m_i é o caractere da mensagem e k_i é o caractere da chave.
  - O cálculo da cifração leva em conta que os caracteres têm seus valores inteiros tabelados na tabela ASCII, logo uma subtração de 65 é feita em cada caractere antes da equação, para que a equação acima seja feita com valores entre 0 e 25, e o resultado final é somado por 65 e adicionado ao arquivo cifrado.
- Não é alfabético:
  - Adiciona o caractere ao arquivo cifrado.

void decrypt_file(FILE*, char*):

Recebe o ponteiro para o arquivo de texto cifrado e a chave. Pede ao usuário o nome do novo arquivo claro. Lê cada caractere do arquivo e checa se o carácter é alfabético ou não, fazendo o seguinte:

- É alfabético:
  - $m_i = (c_i - k_i) \bmod 26$, onde m_i é o caractere da mensagem, c_i é o caractere cifrado e k_i é o caractere da chave
  - O resultado pode ser negativo, logo:
    - Se é negativo: soma-se 26 ao resultado e adiciona ao arquivo claro
    - Se não é negativo: adiciona o caractere ao arquivo claro
- Não é alfabético:
  - Adiciona o caractere ao arquivo claro.

float index_of_coincidence(float*, size_t):

Recebe um vetor com a quantidade de ocorrências de cada letra do alfabeto no texto e o tamanho do texto. Calcula o índice de coincidência (IOC) de cada letra no vetor e retorna o valor.

void format_ciphertext(FILE* fp):

Recebe ponteiro para arquivo cifrado. Cria um novo arquivo chamado formated_input.txt que contém apenas letras maiúsculas, esse arquivo será utilizado para descobrir a chave.

int key_size(FILE*):

Recebe ponteiro para o arquivo de texto cifrado. Formata o arquivo com **format_ciphertext(FILE*)** e abre o arquivo formated_input.txt. Dado n = 2, divide o texto cifrado em grupos de n grupos e calcula o IOC de cada grupo e em seguida a sua média, acrescenta 1 no valor de n, até atingir 13.

A média dos IOCs são impressas na tela e o usuário escolhe o tamanho de chave que mais faz sentido. Também é impresso a média de IOCs para a língua inglesa e portuguesa. O tamanho da chave é retornado.

char* find_key(FILE*):

Recebe ponteiro para o arquivo de texto cifrado. Calcula o tamanho da chave com **int key_size(FILE*)** e abre o arquivo formated_input.txt. Pede ao usuário para informar o idioma (inglês ou português) e divide o arquivo em “t” grupos, dado que t é o tamanho da chave. Esse grupos são definidos como: t + (n * tamanho_chave). t varia de 0 até tamanho_chave - 1 e essa variância no intervalo caracteriza cada grupo.

A média das ocorrências de cada caractere em um grupo é calculada e deve ser comparada pelo usuário com as ocorrências médias no idioma escolhido. Por exemplo.: o caracter em inglês com maior ocorrência é a letra “E” com 12,70%, o usuário deve encontrar nas probabilidades do texto cifrado qual carácter mais se aproxima desse valor, em seguida calcular quantas rotações foram feitas, o que resulta no caráter da chave.

A cada passo a chave vai sendo formada e no final o texto é decifrado com a chave que foi fornecida passo a passo pelo usuário.

OBS:** deixar a escolha sobre o tamanho da chave e os caracteres da mesma para o usuário foi feita pelo fato das estatísticas serem enviesadas em textos muito curtos. Em textos longos os valores esperados aparecem da forma como o esperado e seria fácil automatizar as decisões e excluir o usuário de todo o processo, mas em textos curtos esse processo se torna difícil e por isso a decisão de envolver o usuário na análise das estatísticas se fez necessário.