Tradução de códigos em hexadecimal

Autores: Camille Eduarda Cordeiro Felix, Fabiana Souza de Paula, Giovanna Cardoso da Silva, Letícia Delmilio Soares,Quézia Marques Mendes, Vitorya de Almeida Vieira.

Resumo

O algoritmo desenvolvido neste trabalho tem como objetivo converter um código hexadecimal de volta para uma forma legível, decodificando-o para o sistema de codificação da Tabela ASCII (American Standard Code for Information Interchange). A Tabela ASCII é um padrão de codificação de caracteres criado para padronizar a representação de texto e símbolos em computadores e outros dispositivos eletrônicos. O propósito principal do programa é demonstrar o funcionamento do processo de decodificação.

Introdução

A Segunda Guerra Mundial foi um período crucial, marcado por inovações nas comunicações militares. Entre essas inovações, a codificação de mensagens desempenhou um papel vital. Este estudo tem como objetivo decodificar mensagens interceptadas em formato hexadecimal, convertendo-as em texto legível usando a Tabela ASCII. Ao fazer isso, buscamos revelar o conteúdo dessas mensagens e entender melhor as técnicas de comunicação utilizadas na época. Essa pesquisa é relevante, pois não apenas traz à tona aspectos históricos importantes, mas também mostra a evolução das práticas de segurança da informação, que continuam a ser essenciais nos dias de hoje**.**

Metodologia

O código consiste em um algoritmo que recebe o valor de b que é entregue pela inteligência, e uma mensagem em hexadecimal , que é convertida para a tabela ASCII, usando uma função matemática para verificar e ignorar caracteres inválidos.

Primeiros nós adicionamos as bibliotecas, “#include <stdio.h>” para entrada e saída de dados,”#include <math.h>” para fazer as operações matemáticas, e o “#include <string.h>” para manipular as strings.

Foi usado o tipo Double, para armazenar os valores das posições, pois ela garante mais precisão e armazena números com mais casas decimais.

Então criamos uma função para calcular f(x,b) para calcular as constantes por meio da variável b que será dada pelo agente da inteligência a cada interceptação, e pela variável x que é a posição atual do caractere, começando por 1.

Esta função verifica se o valor decimal do caractere corresponde ao resultado da função polinomial para essa posição. Se corresponder, o caractere é considerado válido.

Em “Int main” criamos as variáveis b que vai ser a variável que recebe a quantidade de testes, e a variável “MsgHex” que vai ser a mensagem lida para que deverá ser descodificada.Criamos também o “int x” que vai ser a posição do caractere e o “int i” que vai ser o elemento que vai ser usado para andar pela “MsgHex”.E então pedimos para o usuário digitar o valor de b e a mensagem em Hexadecimal.

Usamos o int para armazenar o valor de x, pois não é necessário números com casas decimais.

Então, fazemos um loop para percorrer a string da mensagem em hexadecimal para contar quantos caracteres tem na mensagem e incrementamos o “i” para iterar em pares, código precisa processar os caracteres em pares para fazer a conversão corretamente.Então convertemos este par hexadecimal para um valor inteiro. Por meio de um array de 3 posições, onde os dois primeiros elementos são os dois caracteres da string da mensagem hexadecimal , e o terceiro elemento é o caractere nulo “\0” para finalizar a string. Isso transforma o par de caracteres hexadecimais em uma string válida.

A função que converte valores hexadecimais (strtol) precisa de uma string como entrada, então criamos uma string temporária com os dois caracteres que queremos converter.

A função strtol usa o par hexadecimal e a converte de formato hexadecimal para um número inteiro.O valor convertido é armazenado na variável “Ascii”.

Este valor representa o código ASCII do caractere correspondente ao par de caracteres hexadecimais.

Depois o código verifica se o valor convertido de hexadecimal para ASCII é 0 , e se for 0 significa que é o fim da mensagem, então o código para de processar a mensagem, interrompendo o loop com o comando break.

Chamamos a função para calcular (x, b), que vai realizar um cálculo baseado na posição atual do caractere e em um valor b fornecido anteriormente. E o resultado da função é armazenado em outra variável,para determinar se o caractere atual deve ser ignorado ou não, com base no resultado deste cálculo.Ele verifica se a função é um número inteiro, e se o resultado da função for um número inteiro, isso indica que o caractere atual deve ser ignorado.

Se o resultado desta função for um número inteiro, o código incrementa a variável x, que representa a posição do caractere na mensagem, e então usa o “continue” para pular o restante do loop e ir para o próximo caractere.

E se resultado da função estiver certa, o caractere correspondente ao valor de “Ascii” é impresso na tela, apresentando o resultado.

Então a posição x do caractere é sempre incrementada no final do loop assim permitindo que o próximo caractere na sequência passe na próxima passagem do loop.

Resultado

O resultado da execução do algoritmo faz as mensagens interceptadas e codificadas em hexadecimal, serem convertidas para seus correspondentes caracteres ASCII.Onde para cada mensagem inserida, o código processou em pares de caracteres hexadecimais, fez a validação das mensagens por meio de uma função e decodificou os valores de acordo com o que foi oferecido.

Discussões

Foi uma surpresa para o nosso grupo descobrir que simplesmente converter os valores não era suficiente para nós iniciantes; então estudamos e discutimos mais o uso de uma função para verificar a validade dos caracteres e adicionar uma camada extra de complexidade nesse desenvolvimento do algoritmo para decodificar mensagens em hexadecimal.

Essa experiência nos trouxe uma compreensão mais profunda sobre como as mensagens podem ser codificadas e decodificadas com base na Tabela ASCII.

No começo parecia fácil transformar números hexadecimais em letras compreensíveis; porém, conforme avançamos com o projeto, percebemos que há várias etapas importantes para garantir que o procedimento ocorra de forma mais eficiente possível.

A elaboração da função para autenticar os caracteres com ajuda de uma fórmula matemática foi algo novo e muito interessante.

Isso representou um desafio bem legal para o nosso grupo, pois, normalmente, nossa lógica de início consiste em converter valores e finalizar.

Porém ao estudarmos novas maneiras de enxergar através de outros ângulos e maneiras distintas de criar o passo a passo da solução, acrescentamos ao nosso trabalho uma camada adicional de conhecimento.

Conclusão

O algoritmo descrito neste trabalho converte códigos hexadecimais em texto legível, utilizando a codificação ASCII. A Tabela ASCII padroniza a representação de caracteres em dispositivos eletrônicos. O programa procura demonstrar o processo de decodificação, transformando mensagens codificadas em hexadecimal nos seus caracteres correspondentes. Para isso, ele processa pares de caracteres hexadecimais, valida as mensagens e realiza a decodificação dos valores.

Referências:

MIZRAHI , Victorine Viviane. Treinamento em Linguagem C: 2º Edição. São Paulo: Pearson Prentice Hall, 2008.

Learn, Microsoft. Exemplo de programa C, codificação, decodificação e verificação de uma mensagem. Microsoft Learn, 2023. Disponivel em:(https://learn.microsoft.com/pt-br/windows/win32/seccrypto/example-c-program-signing-encoding-decoding-and-verifying-a-message). Acesso em: 22/09/2024.

Evaristo, Jaime - Aprendendo a Programar, programando em linguagem C, 2001 editora Book Express Ltda.

Feofiloff, Paulo - Algoritmos em linguagem C, editora Elsevier, 2009.

Apêndice:

Repositório do Github: https://github.com/Lehtche/Trabalho-Algoritmo.git