



UNIVERSIDADE NOVE DE JULHO - UNINOVE
CUBO DE RUBIK (VIDE CLASSROOM)

EXCLUÍDOS OS DADOS SOBRE OS AUTORES EM ATENDIMENTO A LGPD - LEI GERAL DE
PROTEÇÃO DE DADOS

CUBO DE RUBIK

São Paulo
2024

**EXCLUÍDOS OS DADOS SOBRE OS AUTORES EM ATENDIMENTO A LGPD -
LEI GERAL DE PROTEÇÃO DE DADOS**

CUBO DE RUBIK

Projeto apresentado a Universidade Nove de Julho - UNINOVE, como parte dos requisitos obrigatórios para obtenção do título de Bacharel em Ciência da Computação.

Prof. Orientador: Edson Melo de Souza, Dr.

**São Paulo
2024**

RESUMO

Este documento delinea as especificações e o funcionamento de uma aplicação de criptografia inovadora que emprega um sistema de matrizes inspirado no renomado quebra-cabeça tridimensional, o Cubo de Rubik. Este método de encriptação capitaliza sobre a complexidade inerente ao cubo, utilizando o horário do sistema da máquina do usuário como a chave criptográfica essencial.

ABSTRACT

Contextualization: Nisi mollit anim consequat deserunt tempor laboris fugiat sit do. Pariatur dui est incididunt deserunt pariatur quis sint. Consectetur aliqua reprehenderit laborum aute id dolor fugiat. In consequat pariatur officia dolor esse pariatur sit reprehenderit. Duis nostrud proident occaecat non adipisicing officia sit sunt aute. Nulla eiusmod labore elit velit. Labore fugiat dolor sint irure Lorem irure est voluptate dolor magna commodo. **Objective:** Consequat velit incididunt laborum sit reprehenderit ea ex cillum ut ut incididunt veniam veniam id. **Method:** Deserunt labore labore reprehenderit fugiat dolor Lorem enim consequat ea. Reprehenderit officia id eiusmod voluptate dolor excepteur. Ut adipisicing occaecat laboris minim laborum dolore tempor. Veniam aliqua ad exercitation aute Lorem veniam laborum voluptate anim sunt enim. **Results:** Deserunt labore labore reprehenderit fugiat dolor Lorem enim consequat ea. Reprehenderit officia id eiusmod voluptate dolor excepteur. Ut adipisicing occaecat laboris minim laborum dolore tempor. Veniam aliqua ad exercitation aute Lorem veniam laborum voluptate anim sunt enim. **Conclusion:** Deserunt labore labore reprehenderit fugiat dolor Lorem enim consequat ea. Reprehenderit officia id eiusmod voluptate dolor excepteur. Ut adipisicing occaecat laboris minim laborum dolore tempor. Veniam aliqua ad exercitation aute Lorem veniam laborum voluptate anim sunt enim.

Keywords: Keyword 1, Keyword 2, Keyword 3, Keyword 4, Keyword 5, Keyword 6.

SUMÁRIO

Lista de Ilustrações	6
Lista de Tabelas	7
Lista de Quadros	8
Lista de Abreviaturas	9
1 Introdução	10
1.1 Citações diretas e indiretas	10
1.1.1 Citação Direta.....	10
1.1.2 Citação Indireta	10
1.2 Montagem de Tabela	10
1.3 Montagem de Quadro	12
1.4 Montagem de Equação	12
1.5 Montagem de Algoritmo	13
1.6 Inclusão de Figura	13
1.6.1 Subseção	13
2 Fundamentação Teórica	14
2.1 Visão Geral	14
2.2 Conteúdo 1	14
3 Metodologia	15
3.1 Visão Geral	15
3.2 Conteúdo 1	15
4 Análise dos Resultados	16
4.1 Item 1	16
5 Conclusões	17
Referências Bibliográficas	18
Apêndices	19
A : Título	19
Anexos	20

A	: Título	20
LISTA DE ILUSTRAÇÕES		
<hr/>		
1.1	Descrição da figura.	13

LISTA DE TABELAS

1.1 Descrição da tabela	10
1.2 Formatação no modo paisagem para textos grandes.	11

LISTA DE QUADROS

1.1 Descrição dos dados contidos no quadro.	12
--	----

LISTA DE ABREVIATURAS

MM Morfologia matemática

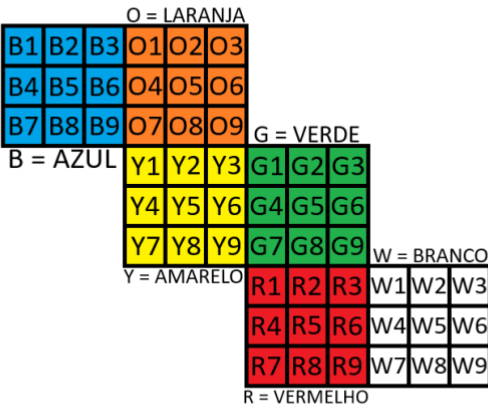
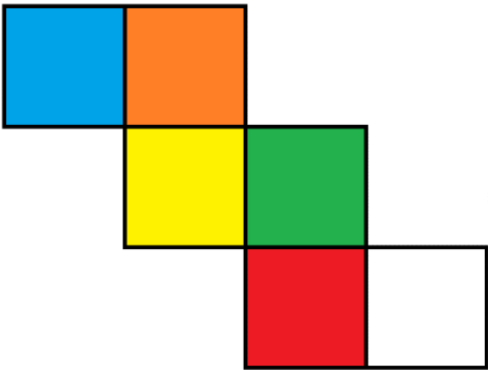
CC Componente conexo

EE Elemento estruturante

1 INTRODUÇÃO

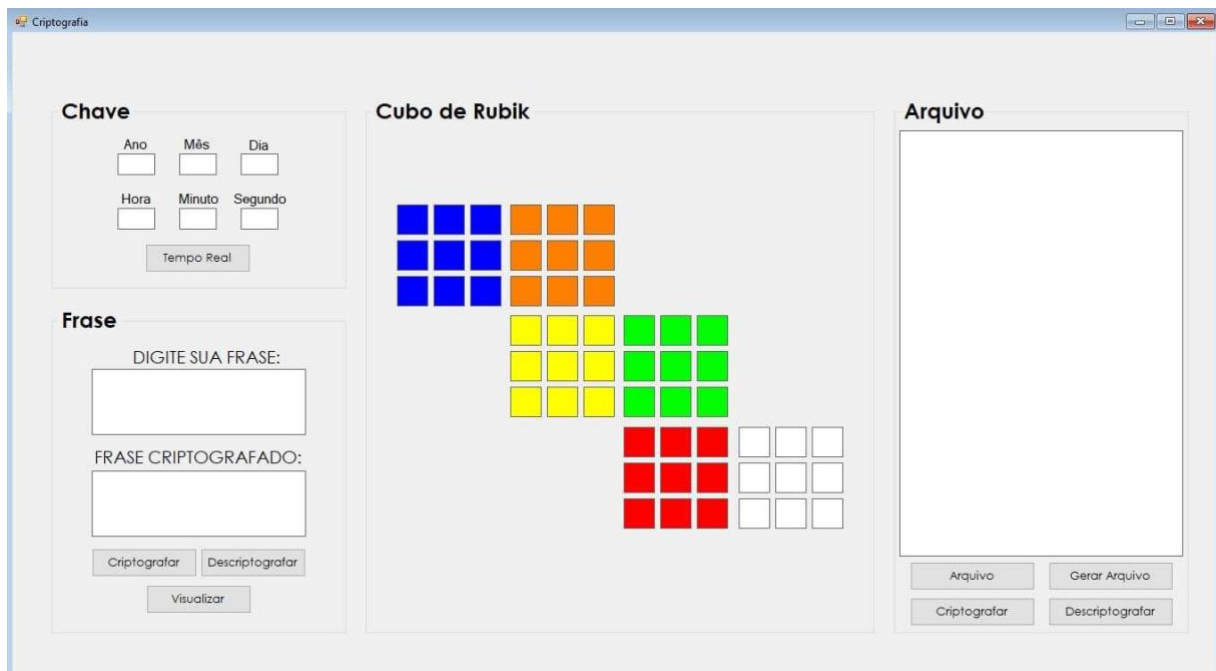
Resumo do capítulo

O design da aplicação foi concebido e desenvolvido pela presente equipe, que empregou um modelo de mapeamento exclusivo, desenhado com a assistência de um Cubo de Rubik físico para uma autenticidade e precisão meticolosas. No coração desta aplicação, reside a representação de cada segmento do cubo, categorizado de acordo com as seis cores distintas presentes em suas respectivas faces. Apesar de um movimento, ou "giro", do cubo resultar na alteração da posição das peças, as identidades (cores) desses elementos permanecem inalteradas. O processo de criptografia é impulsionado pela chave do usuário, composta por seis componentes temporais: ano, mês, dia, hora, minuto e segundo. A inserção desses dados pelo usuário inicia uma sequência de seis giros meticolosos, correspondentes a cada unidade de tempo, aplicados às seis faces do cubo simulado. Esta operação é a essência da criptografia: cada "giro" representa uma permutação dos valores textuais, uma dança coordenada de caracteres que trocam de posições de maneira controlada e complexa. Para viabilizar esta abordagem, estabelece-se um arranjo matricial que captura e cataloga os valores de cada elemento do cubo, armazenando cinquenta e quatro unidades de dados em seis matrizes distintas. Essas matrizes servem como um registro organizado e acessível dos dados processados. Com a solicitação de criptografia pelo usuário, o sistema ativa o mecanismo de "giros" do cubo, baseando-se nas chaves definidas para transposição dos dados textuais entre as matrizes. A engenhosidade da aplicação não se limita à criptografia. A descriptografia é também uma função integral deste sistema, revertendo o texto ao seu estado original. Tal processo é alcançado ao executar os "giros" em sentido inverso, desembaralhando a sequência de movimentos previamente aplicados e restaurando a ordem inicial dos caracteres. A habilidade do algoritmo de reverter com precisão cada ação criptográfica é um testemunho da sua sofisticação e robustez, garantindo a integridade dos dados e a confidencialidade das informações do usuário. A documentação a seguir oferece uma visão detalhada sobre a implementação técnica, a lógica subjacente e as instruções operacionais deste sistema de criptografia, refletindo nosso compromisso com a excelência em segurança de dados e inovação em criptografia.



B1	→	B3	O1	→	O3	Y1	→	Y3	G1	→	G3	R1	→	R3	W1	→	W3
B2	→	B6	O2	→	O6	Y2	→	Y6	G2	→	G6	R2	→	R6	W2	→	W6
B3	→	B9	O3	→	O9	Y3	→	Y9	G3	→	G9	R3	→	R9	W3	→	W9
B4	→	B2	O4	→	O2	Y4	→	Y2	G4	→	G2	R4	→	R2	W4	→	W2
B5	→	B5	O5	→	O5	Y5	→	Y5	G5	→	G5	R5	→	R5	W5	→	W5
B6	→	B8	O6	→	O8	Y6	→	Y8	G6	→	G8	R6	→	R8	W6	→	W8
B7	→	B1	O7	→	O1	Y7	→	Y1	G7	→	G1	R7	→	R1	W7	→	W1
B8	→	B4	O8	→	O4	Y8	→	Y4	G8	→	G4	R8	→	R4	W8	→	W4
B9	→	B7	O9	→	O7	Y9	→	Y7	G9	→	G7	R9	→	R7	W9	→	W7
R7	→	W7	B3	→	W3	B7	→	O7	Y3	→	O3	Y7	→	G7	R3	→	G3
R8	→	W8	B6	→	W6	B8	→	O8	Y6	→	O6	Y8	→	G8	R6	→	G6
R9	→	W9	B9	→	W9	B9	→	O9	Y9	→	O9	Y9	→	G9	R9	→	G9
W7	→	O1	W3	→	G1	O7	→	G1	O3	→	W1	G7	→	W1	G3	→	O1
W8	→	O4	W6	→	G2	O8	→	G4	O6	→	W2	G8	→	W4	G6	→	O2
W9	→	O7	W9	→	G3	O9	→	G7	O9	→	W3	G9	→	W7	G9	→	O3
O1	→	Y1	G1	→	Y1	G1	→	R1	W1	→	R1	W1	→	B1	O1	→	B1
O4	→	Y4	G2	→	Y2	G4	→	R4	W2	→	R2	W4	→	B4	O2	→	B2
O7	→	Y7	G3	→	Y3	G7	→	R7	W3	→	R3	W7	→	B7	O3	→	B3
Y1	→	R7	Y1	→	B3	R1	→	B7	R1	→	Y3	B1	→	Y7	B1	→	R3
Y4	→	R8	Y2	→	B6	R4	→	B8	R2	→	Y6	B4	→	Y8	B2	→	R6
Y7	→	R9	Y3	→	B9	R7	→	B9	R3	→	Y9	B7	→	Y9	B3	→	R9

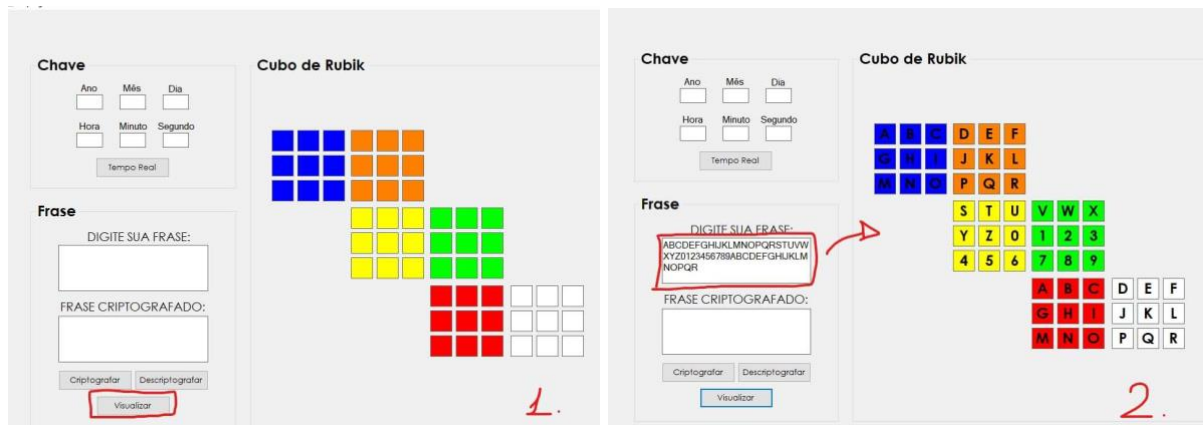
2 INTERFACE DA APLICAÇÃO



Interface

A interface da aplicação é intuitiva e objetiva, o que facilita a navegação e a operação pelo usuário. A tela principal é claramente dividida em três seções principais, cada uma com uma função distinta, refletindo uma abordagem modular ao design da interface. Abaixo detalha-se cada área e suas funcionalidades:

3 Botão Visualizar



Este botão trata-se do mecanismo pelo qual a aplicação prepara o texto para ser visualizado e potencialmente criptografado, garantindo que ele tenha o tamanho correto para ser processado pelo algoritmo. A interface visual associada mostra como o texto é mapeado para o Cubo de Rubik, proporcionando ao usuário uma representação gráfica das operações de criptografia. A aplicação se beneficia de validar a entrada antes de proceder com a criptografia, evitando erros e garantindo que o texto seja apropriadamente manipulado. Esta atenção aos detalhes é fundamental em uma aplicação de segurança, onde a precisão da entrada afeta diretamente a saída do processo de criptografia. Para uma compreensão concisa, cada artifício será separado por tópicos e subtópicos:

- **Declaração de Matrizes**

- O sistema começa com a declaração de várias matrizes de strings, cada uma representando um lado do Cubo de Rubik.
- No contexto do código, `string[,]` define uma matriz de strings com duas dimensões, e `blue` é uma instância dessa matriz, tendo 3 linhas e 3 colunas, perfazendo um total de 9 strings para armazenamento.
- Cada matriz é inicializada para conter 9 posições (`new string[3, 3]`), e cada posição é destinada a armazenar um caractere ou conjunto de caracteres, possivelmente representando as cores de uma peça do Cubo de Rubik.

- **String 'caesar'**

- Uma string especial chamada `caesar` contém a sequência de caracteres que serão alocados nas matrizes declaradas. Esta string é uma referência à Cifra de César, uma técnica clássica de criptografia que desloca letras ao longo do alfabeto.

- **Condicional para Verificação e Preenchimento**

- Uma condição verifica se o comprimento do texto fornecido pelo usuário é menor que 54 caracteres.
- Se for menor, o código calcula quantos caracteres adicionais são necessários (`caracteresRestantes`) e concatena esses caracteres ao `Texto.Text`. A cadeia usada para o preenchimento vem da string `caesar` e é recortada para o número exato de caracteres necessários para alcançar o total de 54.

```

if (Texto.Text.Length < 54)
{
    int caracteresRestantes = 54 - Texto.Text.Length;
    Texto.Text +=
    "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
    ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789".Substring(0, caracteresRestantes);
}

```

- **Exceção para Textos Longos**

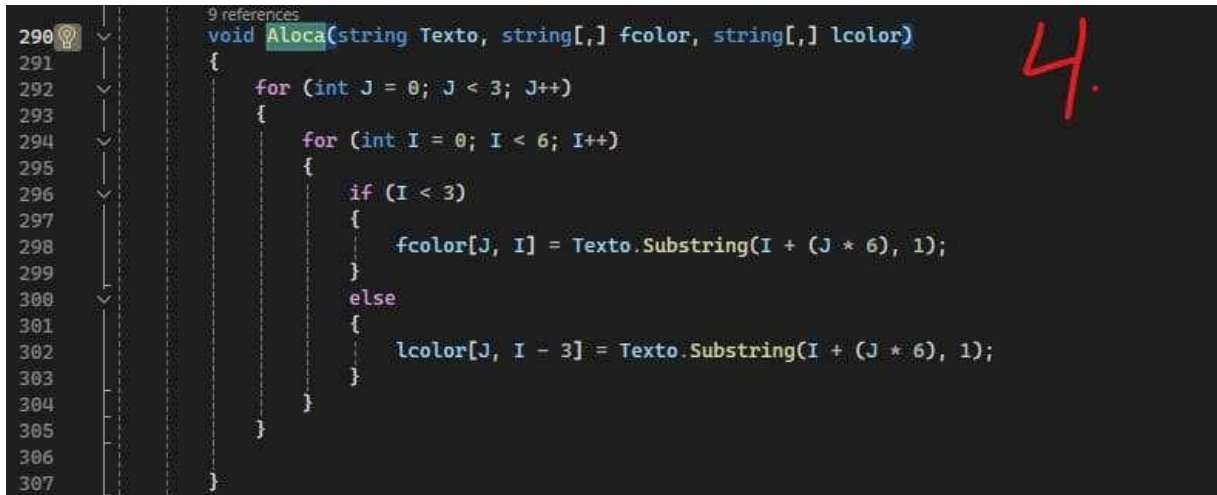
- Se o texto exceder 54 caracteres, uma mensagem de erro é exibida, informando ao usuário que a mensagem é longa demais para ser criptografada. O código então executa a função LimparTextBoxes(), que limpa os campos de texto, e sai da função para impedir a execução de mais código.
- Adiciona um loop onde caso o total de caracteres ser maior que 54, uma mensagem de erro será exibida para o usuário.
- Após isso o void LimparTextBoxes() é acionado e caixa de texto é limpa.

```

else
{
    while (Texto.Text.Length > 54)
    {
        MessageBox.Show("Sua Mensagem longa demais para criptografar.",
        "Erro", MessageBoxButtons.OK, MessageBoxIcon.Error);
        LimparTextBoxes();
        Texto.Text = "";
        return;
    }
}

```

3.1 MÉTODO ALOCA



```
290 void Aloca(string Texto, string[,] fcolor, string[,] lcolor)
291 {
292     for (int J = 0; J < 3; J++)
293     {
294         for (int I = 0; I < 6; I++)
295         {
296             if (I < 3)
297             {
298                 fcolor[J, I] = Texto.Substring(I + (J * 6), 1);
299             }
300             else
301             {
302                 lcolor[J, I - 3] = Texto.Substring(I + (J * 6), 1);
303             }
304         }
305     }
306 }
307
```

- **Estrutura do Método**

- O método Aloca é definido para receber três argumentos: uma string Texto e duas matrizes bidimensionais fcolor e lcolor, que representam, respectivamente, as faces frontal e lateral do cubo no contexto da criptografia.
- fcolor é preenchida com os caracteres da primeira metade de cada conjunto de seis caracteres, enquanto lcolor recebe a segunda metade.

- **Lógica de Alocação**

- A função começa iterando sobre as linhas (J) e colunas (I) das matrizes. A iteração ocorre primeiramente através de cada linha (J) e depois por cada um dos seis caracteres (I) nos grupos de seis da string Texto.
- Para as colunas de 0 a 2 ($I < 3$), os caracteres são alocados na matriz fcolor. Para as colunas de 3 a 5, os caracteres são alocados na matriz lcolor, ajustando o índice da coluna com $I - 3$.
- Com a lógica aplicada, converte-se uma matriz bidimensional em unidimensional.

```
void Aloca(string Texto, string[,] fcolor, string[,] lcolor)
{
    for (int J = 0; J < 3; J++)
    {
        for (int I = 0; I < 6; I++)
        {
            if (I < 3)
            {
                fcolor[J, I] = Texto.Substring(I + (J * 6), 1);
            }
            else
            {
                lcolor[J, I - 3] = Texto.Substring(I + (J * 6), 1);
            }
        }
    }
}
```

- **Função na Criptografia**

- O preenchimento das matrizes é equivalente a um "giro" do cubo na criptografia. Cada matriz fcolor e lcolor recebe um conjunto específico de caracteres, assim como cada face de um Cubo de Rubik teria um conjunto específico de cores.
- Este método não apenas prepara o texto para a criptografia, mas também oferece uma representação visual das peças do cubo que seriam giradas durante a criptografia física, estabelecendo a base para o subsequente embaralhamento dos dados.
- Ao visualizar os dados alocados nas matrizes, o usuário pode obter uma compreensão mais clara de como o texto é segmentado e preparado para a criptografia, refletindo as rotações de um Cubo de Rubik real.

3.2 MÉTODO INSERE

```
private void Insere()
{
    b1.Text = blue[0, 0];
    b2.Text = blue[0, 1];
    b3.Text = blue[0, 2];

    b4.Text = blue[1, 0];
    b5.Text = blue[1, 1];
    b6.Text = blue[1, 2];

    b7.Text = blue[2, 0];
    b8.Text = blue[2, 1];
    b9.Text = blue[2, 2];

    o1.Text = orange[0, 0];
    o2.Text = orange[0, 1];
    o3.Text = orange[0, 2];

    o4.Text = orange[1, 0];
    o5.Text = orange[1, 1];
    o6.Text = orange[1, 2];

    o7.Text = orange[2, 0];
    o8.Text = orange[2, 1];
    o9.Text = orange[2, 2];

    y1.Text = yellow[0, 0];
    y2.Text = yellow[0, 1];
    y3.Text = yellow[0, 2];

    y4.Text = yellow[1, 0];
    y5.Text = yellow[1, 1];
    y6.Text = yellow[1, 2];

    y7.Text = yellow[2, 0];
    y8.Text = yellow[2, 1];
    y9.Text = yellow[2, 2];

    g1.Text = green[0, 0];
}
```

Basicamente essa função é chamada após o Aloca, foi montada baseada nas matrizes 3 x 3 e insere o caractere armazenado pelo Aloca em sua respectiva aba, de uma forma que fique visível ao usuário.

3.3 MÉTODO COLETAINFEATTEXTOCRIP

```

2 referências
private void ColetaInfAtTextoCrip()
{
    StringBuilder textoCripBuilder = new StringBuilder();
    for (int grupo = 0; grupo < 3; grupo++)
    {
        ColetaInfGrupo(textoCripBuilder, PegaMatrizIndex(grupo * 2), PegaMatrizIndex(grupo * 2 + 1));
    }
    textoCrip.Text = textoCripBuilder.ToString();
}

2 referências
private void ColetaInfGrupo(StringBuilder builder, string[,] matrizDireita, string[,] matrizEsquerda)
{
    for (int i = 0; i < 3; i++)
    {
        AddInfStringBuilder(builder, matrizDireita, i, true);
        AddInfStringBuilder(builder, matrizEsquerda, i, false);
    }
}

2 referências
private void AddInfStringBuilder(StringBuilder builder, string[,] matriz, int i, bool isMatrizDaDireita)
{
    for (int j = 0; j < 3; j++)
    {
        if (isMatrizDaDireita)
        {
            builder.Append(matriz[i, j]);
        }
        else
        {
            builder.Append(matriz[i, j]);
        }
    }
}

4 referências
private string[,] PegaMatrizIndex(int index)
{
    switch (index)
    {
        case 0: return blue;
        case 1: return orange;
        case 2: return yellow;
        case 3: return green;
        case 4: return red;
    }
}

```

- O método `ColetaInfEAtTextoCrip()` e as funções auxiliares que ele invoca são a espinha dorsal da transformação do texto na aplicação.
- Decompondo o processo para entender o mecanismo por trás de cada passo e como eles se encaixam dentro do contexto maior da criptografia pode ser:

```
private void ColetaInfEAtTextoCrip()
{
    StringBuilder textoCripBuilder = new StringBuilder();
    for (int grupo = 0; grupo < 3; grupo++)
    {
        ColetaInfGrupo(textoCripBuilder, PegaMatrizIndex(grupo * 2),
PegaMatrizIndex(grupo * 2 + 1));
    }
    textoCrip.Text = textoCripBuilder.ToString();
}
```


4 ANÁLISE DOS RESULTADOS

Descrever as conclusões do trabalho... bla bla bla.

4.1 Item 1

Bla bla bla

5 CONCLUSÕES

Descrever as conclusões do trabalho... bla bla bla.

REFERÊNCIAS BIBLIOGRÁFICAS

ABBASI, A.; ALTMANN, J.; HOSSAIN, L. **Identifying the effects of co-authorship networks on the performance of scholars: A correlation and regression analysis of performance measures and social network analysis measures.** *Journal of Informetrics*, Elsevier Ltd, v. 5, n. 4, p. 594–607, 2011. ISSN 17511577. doi:[10.1016/j.joi.2011.05.007](https://doi.org/10.1016/j.joi.2011.05.007). Citado na pág. 12.

D'UGGENTO, A. M.; RICCI, V.; TOMA, E. **An indicator proposal to evaluate research activities based on Scimago institutions ranking (SIR) data: An application for italian high education institutions.** *Electronic Journal of Applied Statistical Analysis*, v. 9, n. 4, p. 655–674, 2016. ISSN 20705948. doi:[10.1285/i20705948v9n4p655](https://doi.org/10.1285/i20705948v9n4p655). Citado na pág. 10.

MITCHELL, T. M. et al. ***Machine learning***. [S.l.]: McGraw-hill New York, 1997. 432 p. Citado na pág. 10.

VILLASEÑOR-ALMARAZ, M. et al. **Impact factor correlations with Scimago Journal Rank, Source Normalized Impact per Paper, Eigenfactor Score, and the CiteScore in Radiology, Nuclear Medicine & Medical Imaging journals.** *La radiologia medica*, Springer Milan, v. 124, n. 6, p. 495–504, jun 2019. ISSN 0033-8362. doi:[10.1007/s11547-019-00996-z](https://doi.org/10.1007/s11547-019-00996-z). Citado na pág. 10.

APÊNDICES

A : Título

Segundo a ABNT (Associação Brasileira de Normas Técnicas), os apêndices são textos criados "pelo próprio autor" para complementar sua argumentação.

Descrição

1. Conteúdo

ANEXOS

A : Título

Segundo a ABNT (Associação Brasileira de Normas Técnicas), os anexos são documentos criados por terceiros, e usados pelo autor.

Publicação

1. DE SOUZA, E. M.; STOROPOLI, J. E. ; ALVES, W. A. L. **FERRAMENTA DE EXTRAÇÃO DE DADOS PARA A WEB OF SCIENCE**. In: SETII - Seminário em Tecnologia da Informação Inteligente, 2019, São Paulo. Universidade Nove de Julho.