|  |  |
| --- | --- |
| *NOME: GUSTAVO HENRIQUE DE MEIRA* | *RA: 18015* |
| *NOME: GUSTAVO HENRIQUE BÉRA* | *RA: 18180* |

# PROJETO MATRIZ ESPARSA

# Sumário

[Introdução 2](#_Toc5746578)

[Desenvolvimento 3](#_Toc5746579)

[Conclusão 5](#_Toc5746580)

# Introdução

O projeto se trata do desenvolvimento de uma classe que simule uma matriz (um tipo de tabela, que na programação é utilizada para armazenar dados, efetuar operações matemáticas etc.) a partir de uma Lista Cruzada (sem utilizar o objeto pronto, visto nos semestres passados e que é instanciado como se fosse um vetor), na qual todos os dados estão conectados, isto é, cada ponteiro dessa lista aponta para um item seguinte e o último aponta para o primeiro.

Ele tem como objetivo primordial o de aprimorar nossos conceitos a respeito de Listas Ligadas, que teve seus ponteiros alterados para se assemelhar a uma matriz e desenvolver nossas habilidades com as operações relacionadas à matriz, como *procurar um item existente em seu interior, verificar a existência de um elemento, somar e multiplicar as matrizes, além de somar uma determinada constante à uma coluna específica da matriz.*

Vale a pena ressaltar que muitos conceitos apreendidos em Técnicas de Programação ainda têm muita utilidade nesse projeto, como por exemplo o método de realizar a leitura de um arquivo de texto, contendo as informações da matriz, tais como sua proporção e seus dados, contendo a sua posição e seu valor.

# Desenvolvimento

|  |  |
| --- | --- |
| ***Datas*** | ***Modificações*** |
| **25/03/2019** | * Criação das classes Célula e Matriz Esparsa (atributos, propriedades etc.); * Codificação do método “CriarNosCabecas()”, que nos permite percorrer a matriz para inserir e remover valores. |
| **27/03/2019** | * Codificação dos métodos “Inserir()” e “ExisteDado()”, que ajustará os ponteiros na matriz para que possamos realizar a maioria das operações. |
| **29/03/2019** | * Codificação dos métodos “RemoverCelula()” e “LeituraDeArquivo()”; * Correção de erros nos ponteiros da classe MatrizEsparsa, que gerava um loop ao percorrê-la; * Correção do ponteiro da “Direita” da última coluna de uma linha. |
| **01/04/2019** | * Codificação do método “Exibir()”; * Configuração do DataGridView. |
| **02/04/2019** | * Codificação do método “Procurar()”; * Correções aplicadas aos métodos “Inserir()”, “RemoverCelula()” e “ExisteDado()”, pois entrava em loop durante um momento do percurso na matriz; * Codificação do método “SomarConstante()” – Início das operações na MatrizEsparsa. |
| **05/04/2019** | * Codificação do método “SomarMatrizes()”; * Correção aplicada ao método ExisteDado() – A inserção de um dado não se sobrepunha a outro já existente em uma determinada célula; |
| **07/04/2019** | * Codificação do método MultiplicarMatrizes(); * Codificação do método CriarMatriz() – forma manual, sem leitura de arquivo – e do método EsvaziarMatriz(); * Método CriarNosCabecas() melhorado – está mais rápido; * Tratamentos de Exceções; * Melhorias na interface visual; |

# Conclusão

A partir da realização desse projeto, é possível dizer que um dos métodos que tem um alto nível de dificuldade e de importância para as outras tarefas que o programa desempenha é o *ExisteDado()*, e a justificativa para tal ocorrido é simples: era necessário compreender a forma que percorremos a matriz e, além disso, esse método configura os ponteiros para que uma inserção possa ser efetuada e para que uma remoção possa ser concluída com sucesso, pois não é possível remover um elemento que nem se quer está contido na matriz. As operações matemáticas perderam a sua dificuldade devido a esse método.