UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ
CAMPUS CASCAVEL
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

TAD FILA USANDO HERANÇA DE UMA LISTA ENCADEADA SIMPLES EM JAVA. ESTRUTURA DE DADOS

GUSTAVO ANTONIO MARTINI
GUSTAVO MACEDO
VINICIUS GILNEK DRAGE

Cascavel-PR
2022

UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ
CAMPUS CASCAVEL
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

TAD FILA USANDO HERANÇA DE UMA LISTA ENCADEADA SIMPLES EM JAVA. ESTRUTURA DE DADOS

GUSTAVO ANTONIO MARTINI
GUSTAVO MACEDO
VINICIUS GILNEK DRAGE

Suas funções, assinaturas e metodologia de teste

Cascavel-PR
2022

Sumário

1 Introdução

Documento desenvolvido com objetivo de explicar as funções implementadas em JAVA, tanto como as decisões de projeto tomadas, para TAD Fila usando herança de um Lista Encadeada Simples. Entre os documento, como solicitado, acompanha o Makefile, o qual monta, compila e executa o programa em JAVA, os resultados são salvos em um arquivo .txt ao final da execução do comando "make run". Sobretudo, ao executar mais de uma vez, ou seja toda subsequência, é necessário executar o comando "make clean", o qual, por decisão de projeto, também remove o .txt. Está decisão foi tomada devido aos testes culparem muito espaço, assim deixando o .txt deveras poluído. O repositório do GitHub é acessado pelo link <https://github.com/gutamen/ED-trab-2>.

2 Metodologia

Código desenvolvido em sincronia com GITHUB, utilizando a IDE NetBeans em conjunto com Visual Studio (Microsoft), os arquivos presentes no GitHub, contém, Makefile, METAINF.MF(necessária para a criação do .jar[executável de programas em java]), pasta com os arquivos .java. Nenhum teste nem .jar acompanha os arquivos no git, é necessário executar o "make run"ao menos uma vez para obter os testes.

3 Desenvolvimento

De modo objetivo apresentarei a discussão sobre como foram divididas as classes, tanto como cada característica específica das mesmas.

A priori, a classe inicial que define a estrutura da Lista, tanto como os gets and sets necessários para o funcionamento, chamada de dataStructLista, apresentada a baixo. Podemos perceber que é uma estrutura básica de fila.

```
1  public class dataStructLista {
2      public dataStructLista next;
3      public int val;
4      public dataStructLista getNext() {
5          return next;
6      }
7      public void setNext(dataStructLista next) {
8          this.next = next;
9      }
10     public int getVal() {
11         return val;
12     }
13     public void setVal(int val) {
14         this.val = val;
15     }
16 }
```

Código da classe "dataStructLista".

Por sua vez, a classe Lista se utiliza do tipo de dado definido na classe dataStricLista para desenvolver as funções de uma Lista. Por decisão de projeto optei por todas as funções de operação, Fila e Lista, estarem presentes na classe Lista, deste modo na classe Fila não são desenvolvidos demais funções.

Temos sete funções no total, as quais são essas lsitadas suas sinaturas no código a baixo.

```
1    public void insertInit(int x)
2    public void removeInit()
3    public void insertFinal(int x)
4    public int[] search(int x)
5    public boolean isEmpty()
6    public void removeFinal()
7    public String printList()
```

Todas as funções presentes na "Lista.java"e suas respectivas assinaturas presentes em "Fila.java".

insertInit: insere um nodo no início da lista.

insertFinal: insere um nodo no final da lista.

Emptylist: retorna verdadeiro se o comprimento da lista for igual a zero.

removeInit: remove o primeiro nodo da lista, se for nulo retorna a lista está vazia.

insertFinal: remove o último nodo da lista, se for nulo retorna a lista está vazia.

search: procura o valor passado pro parâmetro na lista, se encontrar retorna o mesmo e sua posição, caso não encontrar retorna que o número não foi inserido, caso a lista estiver vazia, informa a situação.

printList: printa a lista não há parâmetros nessa função nem retorno.

Para a classe FilaFromLista é onde se encontra a "main", onde a estrutura de teste está montada, utilizando-se da das funções já feitas e explicadas anteriormente.

4 Testes

A posteriori, temos então que, os resultados serão explicados passo a passo no arquivo, ao viés de seus acontecimentos subsequentes. Sendo executados pelo "make run".