Cliente: Prefeitura

Aluno: Gabriel Schweder Piske - Técnico em Desenvolvimento de Sistemas - T DESI 2024/1 N1

Tópico A - Descrever justificativas para o desenvolvimento do algoritmo:

Descrição: A necessidade de desenvolver um algoritmo para calcular a quantidade de passageiros que circulam nos ônibus de uma linha durante as viagens realizadas em horários de pico surge da necessidade da empresa de transporte do município em identificar as linhas mais sobrecarregadas. Este projeto visa fornecer uma solução confiável para a tomada de decisões de investimento, otimizando a alocação de recursos e garantindo um transporte público mais eficiente para a população.

Tópico B - Incluiu o fluxograma do algoritmo no arquivo LeiaMe:

Link para acesso do Fluxograma:
https://drive.google.com/drive/folders/19RZ2clabXchAE0dgje5V6WPWPHzLfWBX?usp=drive_link
Caso as imagens não estejam legíveis, segue link de acesso do Fluxograma no Miro:
https://miro.com/welcomeonboard/Zkh1NXFtZmU0RmpQVWlIdE0yd21PemxoSjl1Q0hrdmN3OVd5UElvTEp2OUszaVFIWFZkRGI1d0RXTU4yTDZjNnwzNDU4NzY0NTc4MjkxNDYzNDYyfDI=?share_link_id=726340500933

Tópico C- Incluiu o algoritmo no arquivo LeiaMe

Main.java:

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.InputMismatchException;
import java.util.Scanner;

/**
 * @author gabriel_piske
 */
public class Main {

   //Variavéis Globais
   public static Scanner ler = new Scanner(System.in); //Scanner Global
```

```
public static ArrayList<Onibus> listaOnibus = new ArrayList<>();
public static ArrayList<Linha> listaLinha = new ArrayList<>();
public static ArrayList<Viagem> listaViagem = new ArrayList<>();

public static void main(String[] args) throws IOException {
   try {
      //recupera todos arquivos de texto
      recuperarOnibus();
      recuperarViagem();
      recuperarLinha();
   } catch (Exception e) {
      System.err.println("Erro ao recuperar arquivos: " + e.getMessage());
   }

   int opcao;
   do {
      //Menu
      System.out.println("Menu: ");
      System.out.println("1 - Cadastro Onibus");
      System.out.println("2 - Cadastro Linha");
      System.out.println("3 - Cadastro Viagem");
      System.out.println("4 - Sair");
      System.out.print("Entrada: ");
      opcao = ler.nextInt();
      try {
         switch (opcao) {
            case 1:
               cadastrarOnibus();
               break;
            case 2:
               cadastrarLinha();
               break;
            case 3:
               cadastrarViagem();
               break;
            case 4:
               System.out.println("Programa Finalizado.");
               break;
            default:
               System.out.println("Opcao Invalida, Tente Novamente");
         }
      } catch (IOException | InputMismatchException e) {
         System.err.println("Erro: " + e.getMessage());
         ler.next(); // Limpa a entrada incorreta do Scanner
```

```
            }

        } while (opcao != 4);

    }

    //Tela para cadastramento do objeto onibus e já colocando-o na lista
    public static void cadastrarOnibus() throws IOException {
        try {
            //Gravando Informações da Placa e Capacidade para o Construtor
            System.out.println("Cadastrar Onibus: ");
            System.out.println("Informe a Placa do Onibus: ");
            String placa = ler.next();
            System.out.println("Informe a Capacidade maxima do Onibus: ");
            int cpMax = ler.nextInt();

            //Intanciando Objeto e adicionando a lista
            Onibus onibus = new Onibus(placa, cpMax);
            listaOnibus.add(onibus);
            System.out.println("Onibus Cadastrado com sucesso!");

            //Gravando o arquivo txt
            FileWriter arquivo = new FileWriter("registroOnibus.txt", true);
            PrintWriter gravador = new PrintWriter(arquivo);
            gravador.println(placa + "," + cpMax);
            gravador.close();
        } catch (InputMismatchException e) {
            System.err.println("Erro de entrada, insira um numero valido.");
            ler.next();
        }

    }

    //Tela para cadastramento do objeto linha e já colocando-a na lista
    public static void cadastrarLinha() throws IOException {
        try {
            //Gravando Informações do Terminal e numero de paradas da linha para
o Construtor
            System.out.println("Cadastrar Linha: ");
            System.out.println("Informe o terminal da Linha: ");
            String terminal = ler.next();
            System.out.println("Informe o numero de paradas: ");
            int nmParadas = ler.nextInt();
```

```
            //Instanciando Objeto e adicionando na Lista
            Linha linha = new Linha(nmParadas, terminal);
            listaLinha.add(linha);
            System.out.println("Linha Cadastrada com sucesso!");

            //Gravando no Arquivo txt
            FileWriter arquivo = new FileWriter("registroLinha.txt", true);
            PrintWriter gravador = new PrintWriter(arquivo);
            gravador.println(nmParadas + "," + terminal);
            gravador.close();
        } catch (InputMismatchException e) {
            System.err.println("Erro de entrada, insira um numero valido.");
            ler.next();
        }

    }

    //Tela para cadastramento do objeto viagem já puxando onibus e linha correspondente e adicionando a lista no final
    public static void cadastrarViagem() throws IOException {
        try {
            System.out.println("Cadastrar Viagem: ");

            //Verifica se há onibus cadastrados
            if (listaOnibus.isEmpty()) {
                System.err.println("Nao ha onibus Cadastrados.");
                return;
            }
            //Lista todos os onibus possiveis para escolha
            System.out.println("Selecione o Onibus: ");
            for (int i = 0; i < listaOnibus.size(); i++) {
                System.out.println((i + 1) + "." + listaOnibus.get(i).getPlaca());
            }
            System.out.print("Entrada: ");
            int onibusSelect = ler.nextInt();

            //Instancia o objeto onibus escolhido para a viagem
            Onibus onibus = listaOnibus.get(onibusSelect - 1);

            //Verifica se há linhas cadastrados
            if (listaLinha.isEmpty()) {
                System.err.println("Nao ha linhas Cadastradas.");
                return;
            }
```

```
            //Lista todos as linhas possiveis para escolha
            System.out.println("Selecione a Linha: ");
            for (int i = 0; i < listaLinha.size(); i++) {
               System.out.println((i + 1) + "." + listaLinha.get(i).getTerminal());
            }
            System.out.print("Entrada: ");
            int linhaSelect = ler.nextInt();

            //Instancia o objeto linha escolhida para a viagem
            Linha linha = listaLinha.get(linhaSelect - 1);

            //Entrada da data e hora para o cadrasto
            System.out.println("Informe a Data da Viagem: ");
            String data = ler.next();
            System.out.println("Informe a Hora da Viagem: ");
            String hora = ler.next();
            Viagem viagem = new Viagem(data, hora, onibus, linha);
            listaViagem.add(viagem);
            System.out.println("Viagem Cadastrada com Sucesso!");

            Viagem viagemRetornado = decorrerViagem();

            //Gravando no Arquivo txt
            FileWriter arquivo = new FileWriter("registroViagem.txt", true);
            PrintWriter gravador = new PrintWriter(arquivo);
            gravador.println(data + "," + hora + "," + onibus.getPlaca() + "," +
onibus.getCapacidadeMaxima() + "," + linha.getTerminal() + "," +
linha.getNmParadas());
            //gravador.println(viagemRetornado);
            gravador.close();
         } catch (InputMismatchException e) {
            System.err.println("Erro de entrada, insira um numero valido.");
            ler.next();
         }

      }

      //Faz a parte lógica do decorrimento da viagem instanciando os onibus/linhas
próprio da viagem
      public static Viagem decorrerViagem() throws IOException {
         try {
            System.out.println("Decorer Viagem: ");
```

```
        //Lista as Viagens cadastradas posteriormente e as recuperadas nos arquivos txt
        System.out.println("Selecione a Viagem:");
        for (int i = 0; i < listaViagem.size(); i++) {
           Viagem viagem = listaViagem.get(i);
           System.out.println((i + 1) + "- Data: " + viagem.getData() + ", Hora: " + viagem.getHora() + ", Onibus: " + viagem.getOnibus().getPlaca() + ", Linha: " + viagem.getLinha().getTerminal());
        }
        System.out.print("Entrada: ");
        int viagemSele = ler.nextInt();

        //Instancia o objeto escolhido
        Viagem viagem = listaViagem.get(viagemSele - 1);

        //instanciando objetos proprios da viagem
        Onibus onibus = viagem.getOnibus();
        Linha linha = viagem.getLinha();

        //Variavéis para controle
        int totalSubiram = 0;
        int totalDesceram = 0;

        //Criando e Gravando no txt
        FileWriter arquivoBalanco = new FileWriter("balancoViagem.txt", true);
        PrintWriter gravadorBalanco = new PrintWriter(arquivoBalanco);
        gravadorBalanco.println("Viagem: Data: " + viagem.getData() + ", Hora: " + viagem.getHora() + ", Onibus: " + onibus.getPlaca() + ", Linha: " + linha.getTerminal());

        //Decorendo paradas da linha
        for (int i = 0; i < linha.getNmParadas(); i++) {
           System.out.println("Parada " + (i + 1) + ": ");

           //Subida
           System.out.println("Quantos Passageiros Subiram? ");
           int subiram = ler.nextInt();
           //Verifica para não subir passageiros a mais da capacidade
           if (onibus.getPassageirosAtual() + subiram > onibus.getCapacidadeMaxima()) {
              System.out.println("Impossivel Subir Todos os Passageiros");
              subiram = onibus.getCapacidadeMaxima() - onibus.getPassageirosAtual();
           }
```

```
            onibus.setPassageirosAtual(onibus.getPassageirosAtual() + subiram);
            totalSubiram += subiram;

            // Descida
            int desceram = 0;
            if (i == 0) {
               System.out.println("Primeira parada: Ninguem pode descer");
            } else if (i == linha.getNmParadas() - 1) {
               desceram = onibus.getPassageirosAtual();
               System.out.println("Ultima parada: Todos devem descer");
            } else {
               System.out.println("Quantos Passageiros Desceram? ");
               desceram = ler.nextInt();
               if (desceram > onibus.getPassageirosAtual()) {
                  desceram = onibus.getPassageirosAtual();
               }
               totalDesceram += desceram;
            }

            onibus.setPassageirosAtual(onibus.getPassageirosAtual() - desceram);

            System.out.println("Passageiros Atuais no onibus: " +
onibus.getPassageirosAtual());
            gravadorBalanco.println("Parada " + (i + 1) + ": Subiram " + subiram + ",
Desceram " + desceram + ", Passageiros Atuais " + onibus.getPassageirosAtual());
         }
         // Balanço Geral
         System.out.println("Viagem Concluida!!!");
         System.out.println("Total de Passageiros que subiram: " + totalSubiram);
         System.out.println("Total de Passageiros que desceram: " +
totalDesceram);

         //Gravando o Balanço no arquivo txt
         gravadorBalanco.println("Total de passageiros que subiram: " +
totalSubiram);
         gravadorBalanco.println("Total de passageiros que desceram: " +
totalDesceram);
         gravadorBalanco.println("-----------------------------------------------");
         gravadorBalanco.close();

         return viagem;
      } catch (InputMismatchException e) {
         System.err.println("Erro de entrada, insira um numero valido.");
         ler.next();
```

```
        }
        return null;
    }

    //Faz a recuperação dos Dados do Arquivo registroOnibus.txt para utilizar no programa
    private static void recuperarOnibus() throws IOException {
        String aarq = "registroOnibus.txt";
        String linha = "";
        File arq = new File(aarq);
        if (arq.exists()) {

            try {
                FileReader abrindo = new FileReader(aarq);
                BufferedReader leitor = new BufferedReader(abrindo);
                while (true) {
                    linha = leitor.readLine();
                    if (linha == null) {
                        break;
                    }
                    //Separa dados da linha do arquivo de texto pela ,
                    String[] linhaAtualOnibusArquivo = linha.split(",");
                    //Cria objeto onibus passando parametros do arquivo de texto parametro 0 é placa e 1 é capacidade máxima
                    Onibus onibus = new Onibus(linhaAtualOnibusArquivo[0], Integer.parseInt(linhaAtualOnibusArquivo[1]));
                    //Adiciona na lista de onibus
                    listaOnibus.add(onibus);
                }
                leitor.close();
            } catch (Exception erro) {
                System.err.println("Erro ao recuperar dados do arquivo registroOnibus.txt: " + erro.getMessage());
            }

        }
    }

    //Faz a recuperação dos Dados do Arquivo registroViagem.txt para utilizar no programa
    private static void recuperarViagem() throws IOException {
        String aarq = "registroViagem.txt";
        String linha = "";
        File arq = new File(aarq);
```

```
      if (arq.exists()) {
         try {
            FileReader abrindo = new FileReader(aarq);
            BufferedReader leitor = new BufferedReader(abrindo);
            while (true) {
               linha = leitor.readLine();
               if (linha == null) {
                  break;
               }
               //Separa dados da linha do arquivo de texto pela ,
               String[] linhaAtualViagemArquivo = linha.split(",");
               Linha linhaHist = new
Linha(Integer.parseInt(linhaAtualViagemArquivo[5]),
linhaAtualViagemArquivo[4]);
               Onibus onibusHist = new Onibus(linhaAtualViagemArquivo[2],
Integer.parseInt(linhaAtualViagemArquivo[3]));

               //Cria objeto onibus passando parametros do arquivo de texto
parametro 0 é data e 1 é hora 2 é Objeto Onibus e 3 e Objeto Linha
               Viagem viagem = new Viagem(linhaAtualViagemArquivo[0],
linhaAtualViagemArquivo[1], onibusHist, linhaHist);
               //Adiciona na lista de onibus
               listaViagem.add(viagem);
            }
            leitor.close();
         } catch (Exception erro) {
            System.err.println("Erro ao recuperar dados do arquivo
registroViagem.txt: " + erro.getMessage());
         }

      }
   }

   //Faz a recuperação dos Dados do Arquivo registroLinha.txt para utilizar no
programa
   private static void recuperarLinha() throws IOException {
      String aarq = "registroLinha.txt";
      String linha = "";
      File arq = new File(aarq);
      if (arq.exists()) {

         try {
            FileReader abrindo = new FileReader(aarq);
            BufferedReader leitor = new BufferedReader(abrindo);
```

```
                while (true) {
                    linha = leitor.readLine();
                    if (linha == null) {
                        break;
                    }
                    //Separa dados da linha do arquivo de texto pela ,
                    String[] linhaAtualLinhaArquivo = linha.split(",");
                    //Cria objeto onibus passando parametros do arquivo de texto
        parametro 0 é nmParads e 1 é terminal
                    Linha linhaHist = new
        Linha(Integer.parseInt(linhaAtualLinhaArquivo[0]), linhaAtualLinhaArquivo[1]);
                    //Adiciona na lista de onibus
                    listaLinha.add(linhaHist);
                }
                leitor.close();
            } catch (Exception erro) {
                System.err.println("Erro ao recuperar dados do arquivo de
        registroLinha.txt: " + erro.getMessage());
            }

        }
    }

}
```

Onibus.java:

```
        /**
         * @author gabriel_piske
         */
        public class Onibus {

            private String placa;
            private int capacidadeMaxima;
            private int passageirosAtual;

            //------------> Construtores
            public Onibus() {

            }

            public Onibus(String placa, int cpMax) {
                this.placa = placa;
                this.capacidadeMaxima = cpMax;
```

```
        this.passageirosAtual = 0;
    }

    public Onibus(int cpMax, String placa, int pasAtual) {
        this.capacidadeMaxima = cpMax;
        this.passageirosAtual = pasAtual;
        this.placa = placa;
    }

    //------------> Gets e Sets
    public int getCapacidadeMaxima() {
        return capacidadeMaxima;
    }

    public void setCapacidadeMaxima(int capacidadeMaxima) {
        this.capacidadeMaxima = capacidadeMaxima;
    }

    public String getPlaca() {
        return placa;
    }

    public void setPlaca(String placa) {
        this.placa = placa;
    }

    public int getPassageirosAtual() {
        return passageirosAtual;
    }

    public void setPassageirosAtual(int passageirosAtual) {
        this.passageirosAtual = passageirosAtual;
    }

    @Override
    public String toString() {
        return placa + "," + capacidadeMaxima;
    }
}
```

Viagem.java:

```
/**
 * @author gabriel_piske
```

```
 */
public class Viagem {

    private String data;
    private String hora;
    private Onibus onibus;
    private Linha linha;
    private int qtdParadas;

    //Construtores
    public Viagem() {

    }

    public Viagem(String data, String hora, Onibus onibus, Linha linha) {
        this.data = data;
        this.hora = hora;
        this.onibus = onibus;
        this.linha = linha;
        this.qtdParadas = 0;
    }

    //Gets e Sets
    public String getData() {
        return data;
    }

    public void setData(String data) {
        this.data = data;
    }

    public String getHora() {
        return hora;
    }

    public void setHora(String hora) {
        this.hora = hora;
    }

    public Onibus getOnibus() {
        return onibus;
    }

    public void setOnibus(Onibus onibus) {
```

```
        this.onibus = onibus;
    }

    public Linha getLinha() {
        return linha;
    }

    public void setLinha(Linha linha) {
        this.linha = linha;
    }

    public int getQtdParadas() {
        return qtdParadas;
    }

    public void setQtdParadas(int qtdParadas) {
        this.qtdParadas = qtdParadas;
    }

    @Override
    public String toString() {
        return data + "," + hora + "," + this.onibus + "," + this.linha ;
    }
}
```

Linha.java:

```
/**
 * @author gabriel_piske
 */
public class Linha {

    private int nmParadas;
    private String terminal;

    //Construtores
    public Linha() {

    }

    public Linha(int nmPar, String term) {
        this.nmParadas = nmPar;
        this.terminal = term;
    }
```

```
        //Gets e Sets
        public int getNmParadas() {
            return nmParadas;
        }

        public void setNmParadas(int nmParadas) {
            this.nmParadas = nmParadas;
        }

        public String getTerminal() {
            return terminal;
        }

        public void setTerminal(String terminal) {
            this.terminal = terminal;
        }

        @Override
        public String toString() {
            return terminal + "," + nmParadas;
        }
}
```

Tópico D - Utilizou software próprio de fluxogramas para desenvolvimento do gráfico:

Descrição: Ferramenta utilizada para desenvolvimento do fluxograma foi o MIRO.

Tópico F - Descrever no arquivo LeiaMe qual a linguagem foi utilizada no desenvolvimento do algoritmo.

Descrição: Conforme instruído e solicitado o algoritmo foi totalmente desenvolvido na linguagem Java.

Tópico G - Descrever no arquivo LeiaMe, qual IDE foi utilizada no desenvolvimento do algoritmo

Descrição: O projeto foi totalmente desenvolvido utilizando a IDE Apache NetBeans.

Tópico H - Descrever no arquivo LeiaMe, infraestrutura de arquivos é necessário para funcionar o algoritmo

Descrição: Para instalar o algoritmo em sua máquina basta abrir o link deste repositório do GitHub (https://github.com/gabrielpiske/contadorPassageiros). E instalar todo o projeto.

Tópico I - Instruiu no arquivo LeiaMe como se configura os arquivos de execução do algoritmo (crítico)

Descrição: Após concluir o tópico H, somente é necessário abrir a pasta "SA_Passageiros_GabrielPiske" na IDE Apache NetBeans para execução.