TAD

**Classe QueueNode:**

package br.unicap.filas;

class QueueNode <T>{

private T info;

private QueueNode<T> prox;

QueueNode(T obj){

this.info = obj;

}

void setInfo(T obj){

this.info = obj;

}

T getInfo(){

return this.info;

}

void setProx(QueueNode<T> prox){

this.prox = prox;

}

QueueNode<T> getProx(){

return this.prox;

}

}

**Classe Queue:**

package br.unicap.filas;

public class Queue<T> {

private QueueNode<T> head;

private QueueNode<T> fim;

public void enQueue(T obj) { //so quando a fila nao esta cheia

QueueNode<T> novo = new QueueNode<>(obj);

if (this.isEmpty()) {

this.head = this.fim = novo;

} else {

this.fim.setProx(novo);

this.fim = novo;

}

}

public T deQueue() {// so quando a fila nao esta vazia

QueueNode<T> retirado = head;

if (this.head == this.fim) {

this.head = this.fim = null;

} else {

head = head.getProx();

}

return retirado.getInfo();

}

public T head() {

return head.getInfo();

}

public boolean isEmpty() {

return head == null;

}

public boolean isFull() {

return false;

}

}

**Classe QueueComLimite:**

package br.unicap.filas;

public class QueueComLimite<T> {

private QueueNode<T> head;

private QueueNode<T> fim;

private int tam;

private int qtde;

public QueueComLimite(int tam){

this.tam = tam;

}

public void enQueue(T obj) { //so quando a fila nao esta cheia

QueueNode<T> novo = new QueueNode<>(obj);

if (this.isEmpty()) {

this.head = this.fim = novo;

this.qtde++;

} else if(!this.isFull()){

this.fim.setProx(novo);

this.fim = novo;

this.qtde++;

}

}

public T deQueue() {// so quando a fila nao esta vazia

QueueNode<T> retirado = head;

if (this.head == this.fim) {

this.head = this.fim = null;

} else {

head = head.getProx();

}

this.qtde--;

return retirado.getInfo();

}

public T head() {

return head.getInfo();

}

public boolean isEmpty() {

return head == null;

}

public boolean isFull() {

return this.qtde==tam;

}

}

Questão 01

**Classe Pedido:**

package br.unicap.dados;

public class Pedido {

private int numero;

private String descricao;

public Pedido(int numero, String descricao) {

this.numero = numero;

this.descricao = descricao;

}

public int getNumero() {

return numero;

}

public void setNumero(int numero) {

this.numero = numero;

}

public String getDescricao() {

return descricao;

}

public void setDescricao(String descricao) {

this.descricao = descricao;

}

@Override

public String toString() {

return "Pedido{" + "numero=" + numero + ", descricao=" + descricao + '}';

}

}

**Classe Lanchonete:**

package br.unicap.controles;

import br.unicap.dados.Pedido;

import br.unicap.filas.Queue;

import java.util.Scanner;

public class Lanchonete {

/\* Que TAD você utilizará para representar os dados do seu problema? A queue (Fila).

Por que?

A fila assim como a lanchonete obedece a regra do FIFO (primeiro a entrar é o primeiro a sair), logo utilizá-la

para este problema facilitou no desenvolvimento do programa e no mantimento desta regra de entrada e saída.

\*/

Queue<Pedido> pedidos = new Queue<>();

public void incluirPedido() {

if (!pedidos.isFull()) {

Pedido p = anotarPedido();

pedidos.enQueue(p);

} else {

System.out.println("Lanchonete lotada!");

}

}

public void atenderPedido() {

if (pedidos.isEmpty()) {

System.out.println("Sem pedidos no momento!");

} else {

Pedido atendido = pedidos.deQueue();

System.out.println(atendido + "-> Atendido");

}

}

private Pedido anotarPedido() {

Scanner in = new Scanner(System.in);

System.out.println("Número do pedido");

int num = in.nextInt();

in.nextLine();

System.out.println("Descrição do pedido");

String desc = in.nextLine();

return new Pedido(num, desc);

}

}

**Classe LanchoneteJoaoEMaria:**

package br.unicap.TestaFila;

import br.unicap.controles.Lanchonete;

import java.util.Scanner;

public class LanchoneteJoaoEMaria {

public static void main(String[] args) throws CloneNotSupportedException {

Scanner in = new Scanner(System.in);

Lanchonete cc = new Lanchonete();

char op;

System.out.println("### Lanche do Joao e da Maria ###");

do {

menuOpcoes();

System.out.print("Informe a opção: ");

op = in.next().charAt(0);

in.nextLine();

switch (op) {

case '1':

cc.incluirPedido();

break;

case '2':

cc.atenderPedido();

break;

case '0':

System.out.println("Fim do programa!");

break;

default:

System.out.println("Opção inválida!");

}

} while (op != '0');

}

public static void menuOpcoes() {

System.out.println("[1] - Inserir pedido\n"

+ "[2] - Atender pedido\n"

+ "[0] - Sair do programa");

}

}

Questão 02

**Classe Projeto:**

package br.unicap.dados;

public class Projeto {

private String nome;

private String descricao;

public Projeto(String nome, String descricao) {

this.nome = nome;

this.descricao = descricao;

}

public String getNome() {

return nome;

}

public void setNome(String nome) {

this.nome = nome;

}

public String getDescricao() {

return descricao;

}

public void setDescricao(String descricao) {

this.descricao = descricao;

}

@Override

public String toString() {

return "Projeto{" + "nome=" + nome + ", descricao=" + descricao + '}';

}

}

**Classe ControllerProjetos:**

package br.unicap.controles;

import br.unicap.dados.Projeto;

import br.unicap.filas.Queue;

import java.util.Scanner;

public class ControllerProjetos {

/\* Que TAD você utilizará para representar os dados do sue problema? A queue (Fila).

Por que?

A fila assim como a lista de projetos obedece a regra do FIFO (primeiro a entrar é o primeiro a sair), logo utiliza-la

para este problema facilitou no desenvolvimento do programa e no mantimento desta regra de entrada e saida.

\*/

Queue<Projeto> projetos = new Queue<>();

public void incluirProjeto() {

if (!projetos.isFull()) {

Projeto p = anotarPedido();

projetos.enQueue(p);

} else {

System.out.println("Sem espaço para mais projetos!!");

}

}

public void avaliarProjeto() {

if (projetos.isEmpty()) {

System.out.println("Sem projetos no momento!");

} else {

Projeto atendido = projetos.deQueue();

System.out.println( atendido + "-> Avaliado!");

}

}

private Projeto anotarPedido() {

Scanner in = new Scanner(System.in);

System.out.println("Nome do projeto");

String nome = in.nextLine();

System.out.println("Descrição do projeto");

String desc = in.nextLine();

return new Projeto(nome, desc);

}

}

**Classe ProjetosJose:**

package br.unicap.TestaFila;

import br.unicap.controles.ControllerProjetos;

import java.util.Scanner;

public class ProjetosJose {

public static void main(String[] args) throws CloneNotSupportedException {

Scanner in = new Scanner(System.in);

ControllerProjetos cp = new ControllerProjetos();

char op;

System.out.println("### Lanche do Joao e da Maria ###");

do {

menuOpcoes();

System.out.print("Informe a opção: ");

op = in.next().charAt(0);

in.nextLine();

switch (op) {

case '1':

cp.incluirProjeto();

break;

case '2':

cp.avaliarProjeto();

break;

case '0':

System.out.println("Fim do programa!");

break;

default:

System.out.println("Opção inválida!");

}

} while (op != '0');

}

public static void menuOpcoes() {

System.out.println("[1] - Incluir Projeto\n"

+ "[2] - Avaliar Projeto\n"

+ "[0] - Sair do programa");

}

}

Questao 03

**Classe Cliente:**

package br.unicap.dados;

public class Cliente {

private int senha;

private String nome;

public Cliente(String nome) {

this.nome = nome;

}

public Cliente(int senha, String nome) {

this.senha = senha;

this.nome = nome;

}

public int getSenha() {

return senha;

}

public void setSenha(int senha) {

this.senha = senha;

}

public String getNome() {

return nome;

}

public void setNome(String nome) {

this.nome = nome;

}

@Override

public String toString() {

return "Cliente{" + "senha=" + senha + ", nome=" + nome + '}';

}

}

**Classe Banco:**

package br.unicap.controles;

import br.unicap.dados.Cliente;

import br.unicap.filas.QueueComLimite;

import java.util.Scanner;

public class Banco {

int qtdeClientes;

QueueComLimite<Cliente> clientes = new QueueComLimite<>(10);

public void fornecerSenha() {

if (!clientes.isFull()) {

Cliente c = registrarCliente();

clientes.enQueue(c);

} else {

System.out.println("Sem espaço para mais clientes!!");

}

}

public void atenderSenha() {

if (clientes.isEmpty()) {

System.out.println("Sem clientes no momento!");

} else {

Cliente atendido = clientes.deQueue();

System.out.println( atendido + "-> Atendido!");

}

}

private Cliente registrarCliente() {

Scanner in = new Scanner(System.in);

System.out.println("Nome do cliente");

String nome = in.nextLine();

int desc = qtdeClientes++;

return new Cliente(desc,nome);

}

}

**Classe BancoView:**

package br.unicap.TestaFila;

import br.unicap.controles.Banco;

import java.util.Scanner;

public class BancoView {

public static void main(String[] args) throws CloneNotSupportedException {

Scanner in = new Scanner(System.in);

Banco cp = new Banco();

char op;

System.out.println("### Banco ###");

do {

menuOpcoes();

System.out.print("Informe a opção: ");

op = in.next().charAt(0);

in.nextLine();

switch (op) {

case '1':

cp.fornecerSenha();

break;

case '2':

cp.atenderSenha();

break;

case '0':

System.out.println("Fim do programa!");

break;

default:

System.out.println("Opção inválida!");

}

} while (op != '0');

}

public static void menuOpcoes() {

System.out.println("[1] - Fornecer senha\n"

+ "[2] - Atender senha\n"

+ "[0] - Sair do programa");

}

}