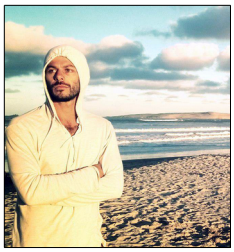
ESTRUCTURA DE DATOS Y ALGORITMOS 1

Obligatorio 1



Guzmán López Orrego
205873

M3A

Michel Camarotta
24 de noviembre de 2016

Índice

Interfaz Sistema: Pre y post condiciones

```
package sistema;

public interface ISistema {

	// PRE: -
	// POS: crea un sistema de reservas con la cantidad de ciudades definida por parámetro
	public Sistema.TipoRet crearSistemaReservas(Integer cantCiudades);

	// PRE: debe existir un sistema de reserva
	// POS: elimina un sistema de reservas reiniciando sus atributos a sus valores originales (listas vacías o cantidad de ciudades en cero)
	public Sistema.TipoRet destruirSistemaReservas();

	// PRE: el ingreso de una nueva ciudad no puede superar la capacidad del sistema
	// PRE: la ciudad no debe existir previamente en el sistema
	// POS: agrega una ciudad a la lista de ciuades del sistema
	public Sistema.TipoRet registrarCiudad(String ciudad);

	// PRE: el hotel a registrarse no puede tener una cantiad de estrellas menor a 1 o mayor a 5
	// PRE: el hotel a registrarse no puede tener una capacidad menor a cero
	// PRE: el hotel a registrarse tiene que tener una ciudad que ya haya sido ingresada al sistema
	// PRE: no puede existir el hotel a ingresarse en el sistema
	// POS: agrega un hotel a la lista de hoteles del sistema y a la lista de hoteles de la ciudad a la cual pertenece el hotel
	public Sistema.TipoRet registrarHotel(String ciudad, String nombre, int estrellas, int capacidad);

	// PRE: tiene que existir el hotel para la ciudad en el cual quiere registrarse el servicio
	// POS: se registra un servicio para un determinado hotel-ciudad: se agrega el servicio al hotel-ciudad en la lista de hoteles del sistema y en la lista de hoteles de la ciudad
	public Sistema.TipoRet ingresarServicio(String ciudad, String hotel, String servicio);
```

// PRE: debe existir el servicio para el hotel en la ciudad
// POS: se borra el servicio especificado para un determinado hotel-ciudad: se borra el servicio al hotel-ciudad en la lista de hoteles del sistema y en la lista de hoteles de la ciudad
public Sistema.TipoRet borrarServicio(String ciudad, String hotel, String servicio);

// PRE: el ranking ingresado debe ser mayor a cero o menor a cinco
// PRE: tiene que existir el hotel-ciudad en el sistema para el cual se quiere ingresar el comentario
// POS: se registra un comentario para un determinado hotel-ciudad: se agrega el comentario al hotel-ciudad en la lista de hoteles del sistema y en la lista de hoteles de la ciudad
// POS: se reordena la lista de Hoteles por Ranking
public Sistema.TipoRet ingresarComentario(String ciudad, String hotel, String comentario, int ranking);

// PRE: debe existir el hotel-ciudad para el cual quiere ingresarse una reserva
// POS: se ingresa una reserva para el cliente en el hotel-ciudad especificado si el hotel no está lleno (en el hotel-ciudad de la lista de hoteles del sistema y en la lista de hoteles para la ciudad)
// POS: se ingresa la reserva a una lista de espera si el hotel está lleno (en el hotel-ciudad de la lista de hoteles del sistema y en la lista de hoteles para la ciudad)
public Sistema.TipoRet realizarReserva(int cliente, String ciudad, String hotel);

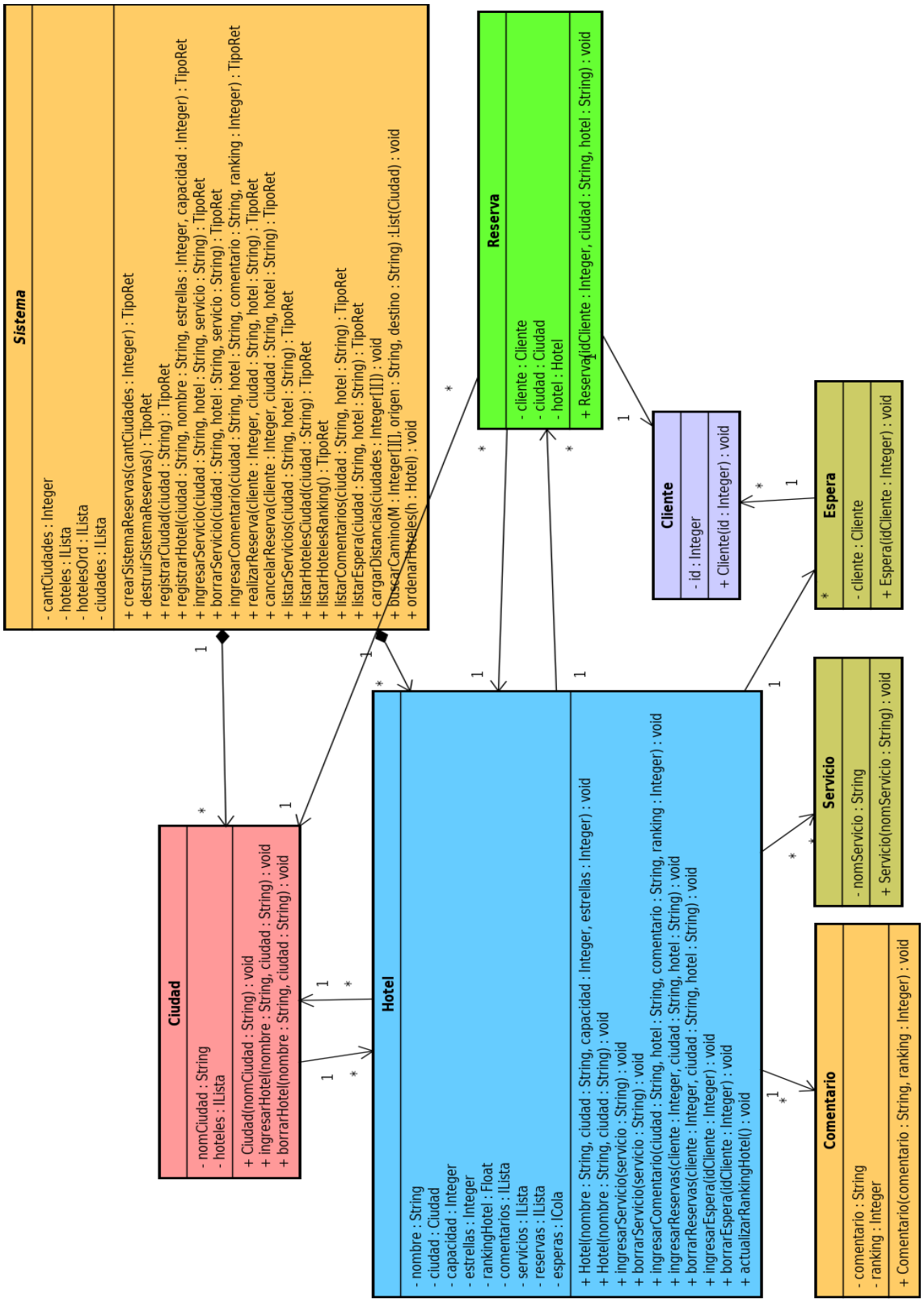
// PRE: debe existir la reserva para el cliente-hotel-ciudad en la lista de reservas o en la lista de esperas del hotel-ciudad
// POS: se elimina la reserva para el cliente de la lista de reservas o la lista de espera del hotel-ciudad
public Sistema.TipoRet cancelarReserva(int cliente, String ciudad, String hotel);

// PRE: debe existir el hotel-ciudad para el cual se desea listar los servicios
// POS: lista los servicios del hotel-ciudad especificado (se muestran en pantalla)
public Sistema.TipoRet listarServicios(String ciudad, String hotel);

```
    // PRE: debe existir la ciudad para la cual se desea listar los
hoteles
    // POS: se listan los hoteles en la ciudad (se muestran en
pantalla)
    public Sistema.TipoRet listarHotelesCiudad(String
ciudad);

    // PRE: -
    // POS: lista los hoteles de la lista de hoteles del sistema
ordenados por ranking de forma descendente (se muestran en
pantalla)
    public Sistema.TipoRet listarHotelesRanking();

    // PRE: debe existir el hotel-ciudad para la cual se desea listar
los comentarios
    // POS: lista los comentarios del hotel-ciudad (se muestran en
pantalla)
    public Sistema.TipoRet listarComentarios(String ciudad,
String hotel);

    // PRE: debe existir el hotel-ciudad para la cual se desea listar
las reservas en la lista de espera
    // POS: lista las reservas en la lista de espera del hotel-ciudad
(se muestran en pantalla)
    public Sistema.TipoRet listarEspera(String ciudad,
String hotel);

    // Para el ejercicio que utiliza la matriz agregar
    public Sistema.TipoRet CargarDistancias(int[][] ciudades);

    public Sistema.TipoRet buscarCamino(int[][] ciudades, String
ciudadOrigen, String ciudadDestino);

}
```

Solución escogida

Diagrama de la estructura de datos



Justificación

ColaDinamica	
Estructura	private NodoCola frente; private NodoCola cola; private int largo; private int tope;
Implementación en Java	public void enqueue(Object o) public void dequeue() public Object obtenerPrimerElemento() public Object obtenerElementoI(Integer i) public Object obtenerPrimerElementoYDequeue() public int cantidadElementos() public void vaciarLista() public boolean esVacia() public boolean esLLena() public boolean existeElemento(Object o) public void borrarElemento(Object o)
Justificación	Dicha clase permite mantener una lista de cola, es decir, que los objetos se agregan por la cola y se eliminan por el frente. Es necesaria para mantener la lista de espera. Las nuevas reservas en espera entran a la cola de la lista mientras que las más viejas se eliminan por el frente (pasan a la lista de reservas).

ListaSEIni	
Estructura	private NodoLista inicio; private NodoLista fin; private int cant;
Implementación en Java	public boolean esVacia() public void agregarInicio(Object dato) public void agregarFinal(Object dato) public void insertar(Object dato) public void agregarI(Object dato, Integer i) public void borrarNodoIni() public void borrarNodoFin() public void vaciarLista() public void borrarElemento(Object dato) public void mostrarElemento() public boolean existeElemento(Object dato) public Object obtenerElemento(Object dato) public Object obtenerElementoI(Integer i) public Object obtenerPrimerElemento() public int cantidadElementos()
Justificación	Dicha clase permite mantener una lista simplemente encadenada, es decir, que los objetos se pueden agregar al inicio, al final o en cualquier parte de la lista gestionando los punteros de la misma. Es necesaria para mantener las listas de hoteles, ciudades, comentarios, servicios y reservas.

ListaOrd

Estructura	public NodoLista inicio; private Comparator<Object> comp;
Implementación en Java	public boolean esVacia() public void agregarInicio(Object dato) public void agregarFinal(Object dato) public void insertar(Object dato) public void agregarI(Object dato, Integer i) public void borrarNodoIni() public void borrarNodoFin() public void vaciarLista() public void borrarElemento(Object dato) public void mostrarElemento() public boolean existeElemento(Object dato) public Object obtenerElemento(Object dato) public Object obtenerElementoI(Integer i) public Object obtenerPrimerElemento() public int cantidadElementos()
Justificación	Esta clase permite mantener ordenada la lista de hoteles por ranking. Permite que los objetos se inserten a la lista de hoteles en orden descendente según su ranking. Se implementa para poder listar los hoteles por ranking imprimiéndolos en pantalla.

NodoCola	
Estructura	private Object dato; private NodoCola siguiente;
Implementación en Java	-
Justificación	Permite definir los punteros (nodos) en la lista de ColaDinamica

NodoLista	
Estructura	private Object dato; private NodoLista siguiente;
Implementación en Java	
Justificación	Permite definir los punteros (nodos) en la lista ListaSEIni

ICola	
Estructura	-
Implementación en Java	public void enqueue(Object o) public void dequeue() public Object obtenerPrimerElemento() public Object obtenerPrimerElementoYDequeue() public int cantidadElementos() public void vaciarLista() public boolean esVacia() public boolean esLLena() public boolean existeElemento(Object o) public void borrarElemento(Object o) Object obtenerElementoI(Integer i)
Justificación	Es la interfaz que es implementada por ColaDinamica

ILista	
Estructura	-

Implementación en Java	public boolean esVacia();
	public void agregarInicio(Object dato);
	public void agregarFinal(Object dato);
	void agregarI(Object dato, Integer i);
	void insertar(Object dato);
	public void borrarNodoIni();
	public void borrarNodoFin();
	public void vaciarLista();
	public void borrarElemento(Object dato);
	public void mostrarElemento();
	public boolean existeElemento(Object dato);
	public Object obtenerElemento(Object dato);
	public Object obtenerPrimerElemento();
	public int cantidadElementos();
	Object obtenerElementoI(Integer i);

Justificación	Es la interfaz que es implementada por ListaOrd y ListaSEIni

ListaOrd	
Estructura	public NodoLista inicio; private Comparator<Object> comp;
Implementación en Java	public boolean esVacia() public void agregarInicio(Object dato) public void agregarFinal(Object dato) public void insertar(Object dato) public void agregarI(Object dato, Integer i) public void borrarNodoIni() public void borrarNodoFin() public void vaciarLista() public void borrarElemento(Object dato) public void mostrarElemento() public boolean existeElemento(Object dato) public Object obtenerElemento(Object dato) public Object obtenerElementoI(Integer i) public Object obtenerPrimerElemento() public int cantidadElementos()
Justificación	Esta clase permite mantener ordenada la lista de hoteles por ranking. Permite que los objetos se inserten a la lista de hoteles en orden descendente según su ranking. Se implementa para poder listar los hoteles por ranking imprimiéndolos en pantalla.

NodoCola	
Estructura	private Object dato; private NodoCola siguiente;
Implementación en Java	-
Justificación	Permite definir los punteros (nodos) en la lista de

	ColaDinamica

NodoLista	
Estructura	private Object dato; private NodoLista siguiente;
Implementación en Java	
Justificación	Permite definir los punteros (nodos) en la lista ListaSEIni

ICola	
Estructura	-
Implementación en Java	public void enqueue(Object o) public void dequeue() public Object obtenerPrimerElemento() public Object obtenerPrimerElementoYDequeue() public int cantidadElementos() public void vaciarLista() public boolean esVacia() public boolean esLLena() public boolean existeElemento(Object o) public void borrarElemento(Object o) Object obtenerElementoI(Integer i)
Justificación	Es la interfaz que es implementada por ColaDinamica

RankingHotelComparator	
Estructura	-
Implementación en Java	public int compare(Object obj1, Object obj2)
Justificación	Implementa Comparator<Object> para comparar los hoteles por el ranking de los mismos y poder insertarlos ordenados en una lista

ISistema	
Estructura	-

Implementación en Java	public Sistema.TipoRet crearSistemaReservas(Integer cantCiudades);
	public Sistema.TipoRet destruirSistemaReservas();
	public Sistema.TipoRet registrarCiudad(String ciudad);
	public Sistema.TipoRet registrarHotel(String ciudad, String nombre, int estrellas, int capacidad);
	
	public Sistema.TipoRet ingresarServicio(String ciudad, String hotel, String servicio);
	public Sistema.TipoRet borrarServicio(String ciudad, String hotel, String servicio);
	public Sistema.TipoRet ingresarComentario(String ciudad, String hotel, String comentario, int ranking);
	public Sistema.TipoRet realizarReserva(int cliente, String ciudad, String hotel);
	public Sistema.TipoRet cancelarReserva(int cliente,

	String ciudad, String hotel);
	public Sistema.TipoRet listarServicios(String ciudad, String hotel);
	public Sistema.TipoRet listarHotelesCiudad(String ciudad);
	public Sistema.TipoRet listarHotelesRanking();
	public Sistema.TipoRet listarComentarios(String ciudad, String hotel);
	public Sistema.TipoRet listarEspera(String ciudad, String hotel);
	public Sistema.TipoRet CargarDistancias(int[][] ciudades);
	public Sistema.TipoRet buscarCamino(int[][] ciudades, String ciudadOrigen, String ciudadDestino);

Justificación	Es la interfaz que es es implementada por el Sistema

Sistema	

Estructura	private int cantCiudades = 0;
	private ILista hoteles;
	private ILista hotelesOrd = new ListaOrd(new RankingHotelComparator());
	private ILista ciudades;

Implementación en Java	public Sistema.TipoRet crearSistemaReservas(Integer cantCiudades);
	public Sistema.TipoRet destruirSistemaReservas();
	public Sistema.TipoRet registrarCiudad(String ciudad);
	public Sistema.TipoRet registrarHotel(String ciudad, String nombre, int estrellas, int capacidad);
	public Sistema.TipoRet ingresarServicio(String ciudad, String hotel, String servicio);
	public Sistema.TipoRet borrarServicio(String ciudad, String hotel, String servicio);
	public Sistema.TipoRet ingresarComentario(String ciudad, String hotel, String comentario, int ranking);
	public Sistema.TipoRet realizarReserva(int cliente, String ciudad, String hotel);
	public Sistema.TipoRet cancelarReserva(int cliente, String ciudad, String hotel);
	public Sistema.TipoRet listarServicios(String ciudad, String hotel);
	public Sistema.TipoRet listarHotelesCiudad(String ciudad);
	public Sistema.TipoRet listarHotelesRanking();
	public Sistema.TipoRet listarComentarios(String ciudad, String hotel);
	public Sistema.TipoRet listarEspera(String ciudad, String hotel);
	public Sistema.TipoRet CargarDistancias(int[][] ciudades);
	public Sistema.TipoRet buscarCamino(int[][] ciudades, String ciudadOrigen, String ciudadDestino);

Justificación	El sistema es el que permite toda la funcionalidad de la

	aplicación. Mantiene el sistema de reservas en los hoteles, asigna reservas a clientes, los pone en lista de espera si no hay más lugar, registra hoteles, ciudaes, ingresa comentarios, verifica que no existan errores, etc.