# ALGORITMOS Y ESTRUCTURA DE DATOS 2

## Documentación - Obligatorio



Mathias Arcardini

Número estudiante 183036

Matias Hernandez

Número estudiante 151935

Grupo N4B

José Sebastián Grattarola Rizzo

18/11/2015

Índice

[ESTRUCTURA DE DATOS Y ALGORITMOS 1](#_Toc359097655)

[Obligatorio 1](#_Toc359097656)

[Interfaz Sistema: Pre y post condiciones 3](#_Toc359097657)

[Solución escogida 3](#_Toc359097658)

[Diagrama de la estructura de datos 3](#_Toc359097659)

[Justificación 3](#_Toc359097660)

[Testing 3](#_Toc359097661)

# Interfaz Sistema: Pre y post condiciones

\*Precondiciones: la cantidad de esquinas o puntos debe ser mayor 0.

\*Postcondiciones: Crea el sistema con una cantidad limite de esquinas igual a cantPuntos.

**public** Retorno inicializarSistema(**int** cantPuntos)

\*Precondiciones: Debe existir una instancia del sistema ya creada.

\*Postcondiciones: Destruye el sistema de Car Pooling y todos sus elementos, liberando la memoria utilizada.

**public** Retorno destruirSistema()

\*Precondiciones: Debemos recibir un string para matricula y otro para conductor. Se hacen busquedas internas de los objetos con dichos identificadores

\*Postcondiciones: Crea un nuevo movil con id la matricula dada, en estado DISPONIBLE y le asigna el identificador deseado su conductor

**public** Retorno registrarMovil(String matricula, String conductor)

\*Precondiciones: Recibimos un string con la matricula del objeto movil

\*Postcondiciones: Cambia el estado del movil a DESHABILITADO, en caso de que se encuentre el mismo y este en estado DESHABILITADO

**public** Retorno deshabilitarMovil(String matricula)

\*Precondiciones: Recibimos un string con la matricula del objeto movil

\*Postcondiciones: Cambia el estado del movil a DISPONIBLE en caso de que el movil exista y no este disponible

**public** Retorno habilitarMovil(String matricula)

\*Precondiciones: Recibimos un string que representa la matricula del movil

\*Postcondiciones: Destruye el movil identificado con esa matricula, en caso de que el mismo no este asignado.

**public** Retorno eliminarMovil(String matricula)

\*Precondiciones: Recibimos un string que representa la matricula del movil y dos valores double, representando las coordenadas de una esquina del mapa

\*Postcondiciones: Ubica en la esquina de coordenadas x e y, al movil de la matricula pasada por parametro, en caso de que exista la esquina y el movil y ya no haya un movil en la misma

**public** Retorno asignarUbicacionMovil(String matricula, Double coordX, Double coordY)

\*Precondiciones: Recibimos un string que representa la matricula de un movil

\*Postcondiciones: Imprime en pantalla los datos del movil requerido, en caso de que el mismo exista. Formato: [matricula;conductor;estado]

**public** Retorno buscarMovil(String matricula)

\*Precondiciones: No se encuentran precondiciones

\*Postcondiciones: En caso de haber moviles registrados, se imprimen en pantalla [Matricula;Conductor;Estado]

**public** Retorno informeMoviles()

\*Precondiciones: Recibimos dos valores representantes de las coordenadas que forman una esquina.

\*Postcondiciones: Si hay lugar para guardar una esquina mas y la misma no existe en el sistema, se crea una esquina con coordenadas x e y

**public** Retorno registrarEsquina(Double coordX, Double coordY)

\*Precondiciones: Se reciben dos juegos de coordenadas y una cantidad de metros (debe ser numero natural)

\*Postcondiciones: Si la cantidad de metros es coherente, el tramo aun no existe y cada esquina que compondran el tramo, existen, crea un tramo (bidireccionalmente) en el mapa

**public** Retorno registrarTramo(Double coordXi, Double coordYi, Double coordXf, Double coordYf, **int** metros)

\*Precondiciones: Recibimos las coordenadas de una esquina.

\*Postcondiciones: Si la esquina de coordenadas existe y no es parte de otra informacion utilizada en el sistema, la elimina del mapa

**public** Retorno eliminarEsquina(Double coordX, Double coordY)

\*Precondiciones: Recibimos un juego de coordenadas que componen las dos esquinas del tramo que queremos eliminar.

\*Postcondiciones: Si la esquina de coordenadas existe y no es parte de informacion vigente del sistema, la elimina del mapa

**public** Retorno eliminarTramo(Double coordXi, Double coordYi, Double coordXf, Double coordYf)

\*Precondiciones: Recibimos un juego de coordenadas que forman una esquina

\*Postcondiciones: Si existe la esquina, busca el movil mas cercano respecto a las coordenadas pasadas por paramero, que este DISPONIBLE

**public** Retorno movilMasCercano(Double coordX, Double coordY)

\*Precondiciones: Recibimos un juego de coordenadas que forman una esquina y un valor numerico que represente el radio en mts en el que se deben buscar moviles

\*Postcondiciones: Si existe la esquina y el radio es coherente, nos trae la lista de moviles DISPONIBLES que se encuentran dentro de ese radio

**public** Retorno verMovilesEnRadio(Double coordX, Double coordY, **int** radio)

\*Precondiciones: No se encuentran precondiciones

\*Postcondiciones: Muestra la realidad vigente de nuestro sistema, en un mapa desplegado en el navegador

**public** Retorno verMapa()

# Solución escogida

## Diagrama de la estructura de datos

< Pegar imagen del diagrama >

## Justificación

Justificación de las estructuras elegidas para modelar las entidades del problema.

|  |  |
| --- | --- |
| **VENDEDOR** | |
| Estructura |  |
| Justificación |  |

/\*Ídem para todas las entidades del sistema\*/

## Testing

Resumen de las pruebas realizadas y los resultados obtenidos.