UNIVERSIDAD ORT
Uruguay

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70

EVALUACION	**Obligatorio**	**GRUPO**	**Todos**	**FECHA**	**26/09/2017**
MATERIA	**Algoritmos y Estructuras de Datos 2**				
CARRERA	**Analista Programador - ATI**				
CONDICIONES	**- Puntos:** Máximo: 50 Mínimo: 1 **- Fecha máxima de entrega:** **- Horario de Coordinación adjunta para entregas:** de lunes a jueves de 8:00 a 12:30 y de 17:30 a 19:00 hs. **IMPORTANTE** - Los grupos deben estar conformados por **dos personas**. - Inscribirse (sacar la "boleta de entrega"). - Entregar en carpeta con elástico. - Entregar 2 copias en cd, donde se incluya la documentación en pdf con foto de los integrantes. - No entregue el código impreso. - Etiquetar documentación y cd con nombre, nº estudiante, carrera, grupo, materia y docente				

Obligatorio :: Gestión de Plantaciones

Introducción

El MGAP está interesado en contar con un sistema que registre las plantaciones de grano del país.

Una plantación tiene un propietario, un tipo (soja, trigo, arroz, etc.) y una producción mensual, la cual se mide en cantidad de toneladas por mes.

Luego que la cosecha es recolectada, es enviada a silo donde se almacena, hasta quedar lista para su distribución a las diferentes industrias que la utilizan como insumo (el maíz se vende al molino, etc.).

El sistema deberá contar con un mapa en donde se registrarán las ciudades, las plantaciones y los silos.

El sistema ayudará a los productores a saber a cuál silo les conviene llevar sus camiones para almacenar el grano. Actualmente la distribución de carga entre los silos no es homogénea, siendo que algunos están sobrepasados de capacidad y otros están ociosos, lo cual repercute en atrasos para los productores.

Generalidades

Se define una clase Retorno, la cual se utilizará como tipo de retorno para todas las operaciones del sistema. Dicha clase contiene:

- Un resultado, que especifica si la operación se pudo realizar correctamente (OK), o si ocurrió algún error (según el número de error).
- Un valor entero, para las operaciones que retornen un número entero.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70

- Un valor String, para las operaciones que retornen un String, o un valor más complejo (por ejemplo una lista o clase), la cuál será formateada según lo indicado en el Anexo I de este documento.

Se provee: una interfaz llamada ISistema, la cual no podrá ser modificada en ningún sentido, y una clase sistema que la implementa, donde el estudiante deberá completar la implementación de las operaciones solicitadas.

Además, se proveen los siguientes tipos de datos que deberán ser respetados.

Sistema	`public class Sistema{`
	
	`enum TipoPunto {CIUDAD, PLANTACIÓN, SILO};`
	
	`/* Aquí van las operaciones del sistema */`
	
	`}`

Retorno	`public class Retorno {`
	
	`enum Resultado {OK, ERROR_1, ERROR_2, ERROR_3,`
	
	`ERROR_4, ERROR_5, NO_IMPLEMENTADA};`
	
	`int valorEntero;`
	`string valorString;`
	`Resultado resultado;`
	
	`}`

Pueden definirse tipos de datos (clases) auxiliares.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70

Funcionalidades

1. Operaciones globales

1.1. Inicializar Sistema

Firma: `Retorno inicializarSistema (int cantPuntos);`

Descripción: Inicializa las estructuras necesarias para representar el sistema especificado, capaz de albergar como máximo *`cantPuntos`* en el mapa. Los puntos del mapa serán las ciudades, plantaciones, o silos.

Retornos posibles	
OK	• Si el sistema pudo ser inicializado exitosamente.
ERROR	• 1. Si *`cantPuntos`* es menor o igual a 0.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

1.2. Destruir Sistema

Firma: `Retorno destruirSistema();`

Descripción: Destruye el sistema de todos sus elementos y estructuras, liberando la memoria utilizada.

Retornos posibles	
OK	• Siempre retorna OK.
ERROR	• No hay errores posibles.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

2. Operaciones relativas al mapa

2.1. Registrar productor

Firma: `Retorno registrarProductor(String cedula, String nombre, String dirección, String email, String celular);`

Descripción: Registra al productor de granos de cédula “cedula” en el sistema. La operación deberá controlar que el email, el número de celular y la cédula de identidad tengan un formato correcto.

Esta operación deberá realizarse en orden (log n) promedio.

Retornos posibles	
OK	• Si el productor pudo ser registrado exitosamente.
ERROR	• 1. Si la cédula *cédula* no cumple el formato N.NNN.NNN-N • 2. Si el número de celular *celular* no cumple el formato 09NNNNNNN • 3. Si el email *email* no cumple el formato de direcciones de e-mail • 4. Si el productor de cédula *cedula* ya está registrado en el sistema.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Se recomienda el uso de **expresiones regulares** para lograr fácilmente el control de los formatos. Ver Anexo con links de interés.

2.2. Registrar ciudad

Firma: `Retorno registrarCiudad(String nombre, Double coordX, Double coordY);`

Descripción: Registra la ciudad de nombre *nombre* y coordenadas *coordX, coordY* en el sistema.

Retornos posibles	
OK	• Si la ciudad fue registrada exitosamente.
ERROR	• 1. Si en el sistema ya hay registrados *cantPuntos* puntos. • 2. Si ya existe un punto en las coordenadas *coordX*, *coordY* del sistema.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Esta operación deberá realizarse en orden (1) promedio.

2.3. Registrar plantación

Firma: `Retorno registrarPlantación(String nombre, Double coordX, Double coordY, String cedula_productor, int capacidad);`

Descripción: Registra una plantación de nombre *nombre* en el sistema, la cual está a cargo del productor de cédula *cedula_productor* y tiene una capacidad de producción de *capacidad* toneladas de granos por mes.

Retornos posibles	
OK	• Si la plantación pudo ser registrada exitosamente.
ERROR	• 1. Si en el sistema ya hay registrados *cantPuntos* puntos. • 2. Si *capacidad* es menor o igual a 0. • 3. Si el punto de coordenadas *coordX, coordY* ya está registrado en el sistema.

	• 4. Si el productor de cédula *cedula_productor* no existe en el sistema.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Esta operación deberá realizarse en orden (1) promedio.

2.4. Registrar silo

Firma: `Retorno registrarSilo(String nombre, Double coordX, Double coordY, int capacidad);`

Descripción: Registra un silo de nombre *nombre* en el sistema, con una capacidad de procesamiento de *capacidad* toneladas de granos por mes.

Retornos posibles	
OK	• Si el silo pudo ser registrado exitosamente.
ERROR	• 1. Si en el sistema ya hay registrados *cantPuntos* puntos. • 2. Si *capacidad* es menor o igual a 0. • 3. Si el punto de coordenadas *coordX, coordY* ya está registrado en el sistema.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Esta operación deberá realizarse en orden (1) promedio.

2.5. Registrar tramo

Firma: `Retorno registrarTramo(Double coordXi, Double coordYi, Double coordXf, Double coordYf, int peso);`

Descripción: Registra un tramo en el sistema desde la coordenada inicio (*coordXi, coordYi*) hasta la coordenada destino (*coordXf, coordYf)* de peso *peso.*

Nota: Se considerará que los tramos son navegables en ambos sentidos. O sea que si agregamos el tramo para ir del punto A al punto B, también se podrá navegar del punto B al punto A.

Retornos posibles	
OK	• Si el tramo pudo ser registrado exitosamente.
ERROR	• 1. Si *peso* es menor o igual a 0. • 2. Si no existe *coordi* o *coordf*. • 3. Si ya existe un tramo registrado desde *coordi a coordf*.

NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

2.6. Eliminar tramo

Firma: `Retorno eliminarTramo(Double coordXi, Double coordYi, Double coordXf, Double coordYf);`

Descripción: Elimina el tramo desde la coordenada inicio (coordXi, coordYi) hasta la coordenada destino (coordXf, coordYf).

Retornos posibles	
OK	• Si el tramo pudo ser eliminado exitosamente.
ERROR	• 1. Si no existen algunos de los puntos *coordi* o *coordf* • 2. Si no existe un tramo registrado desde *coordi a coordf*.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Nota: Como los tramos son no dirigidos, al eliminar el tramo desde el punto A al punto B, también deberá eliminarse el tramo del punto B al punto A.

2.7. Eliminar punto del mapa

Firma: `Retorno eliminarPunto(Double coordX, Double coordY);`

Descripción: Elimina el punto (coordXi, coordYi) del mapa. El punto puede ser un silo, una ciudad o una plantación. Si existen tramos conectados con el punto, deberán ser eliminados también.

Retornos posibles	
OK	• Si el punto pudo ser eliminado exitosamente.
ERROR	• 1. Si no existe el punto de coordenadas *coordX, coordY* en el mapa
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

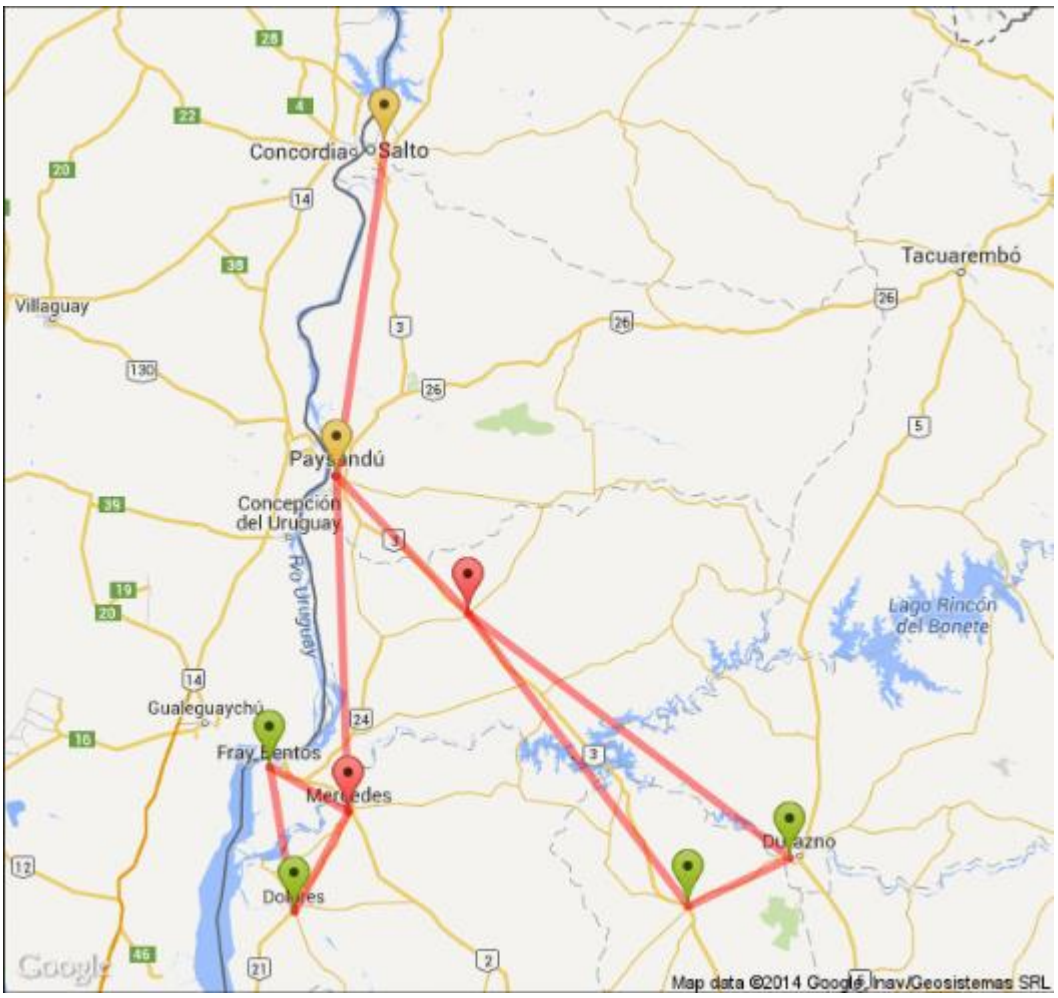
3. Reportes y Rutas

3.1. Mapa de estado

Firma: `Retorno mapaEstado();`

Descripción: Abre una ventana del navegador y muestra en un mapa de Google Maps todos los puntos registrados en el sistema: plantaciones, ciudades y silos. Se deberá utilizar la siguiente referencia de colores para diferenciarlos: ciudades en rojo, plantaciones en amarillo y silos en verde.

Retornos posibles	
OK	• Siempre retorna OK.
ERROR	• No hay errores posibles.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.



3.2. Mejor ruta a silo (7 puntos)

Firma: `Retorno rutaASiloMasCercano(Double coordX, Double coordY);`

Descripción: Cuando una plantación llega a su capacidad de producción, el productor responsable deberá llevar sus granos al silo más cercano que tenga la capacidad de procesar la cantidad de toneladas de granos que produjo la plantación.

El productor entonces utilizará la operación rutaASiloMasCercano, la cual deberá retornar una lista de coordenadas para indicar el camino elegido al silo más cercano con capacidad suficiente. La lista deberá ser retornada en el campo valorString de tipo retorno siguiendo el formato de retorno establecido.

En caso de que la operación sea exitosa el silo deberá reservar la capacidad necesaria para procesar los granos de la plantación, y no podrá utilizarla para otra plantación. A su vez si el silo más cercano no tiene suficiente capacidad para atender el pedido de la plantación, no deberá ser tenido en cuenta y se deberá continuar con la búsqueda.

Retornos posibles	
OK	• Si la ruta pudo ser generada exitosamente.
ERROR	• 1. Si la plantación de coordenadas *coordX, coordY* no existe en el sistema. • 2. Si no encuentra ningún silo que pueda satisfacer la necesidad de procesamiento de la plantación.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Formato: Ejemplo de una ruta con cuatro puntos:

CoordX1;CoordY1|CoordX2;CoordY2|CoordX3;CoordY3|CoordX4;CoordY4

CooordX1,CoordY1 siempre serán las coordenadas de la plantación, y CoordX4,CoordY4 serán las del silo elegido.

3.3. Listado de plantaciones en ciudad (8 puntos)

Firma: `Retorno listadoDePlantacionesEnCiudad(Double coordX, Double coordY);`

Descripción: De cada ciudad se registra el km 0, pero los límites de una ciudad se consideran a los 20 kms a la redonda del km 0. Este listado deberá devolver una lista con todas las plantaciones registradas dentro de los límites de la cuidad de coordenadas coordX, coordY. Se deberá retornar en el campo valorString un String que contenga las coordenadas de las plantaciones consideradas.

Nota: Se considerarán las plantaciones que sean alcanzables recorriendo 20 kms por tramos definidos en el mapa, en lugar de los que estén geométricamente en un radio de 20 kms.

Retornos posibles	
OK	• Si el listado se pudo generar correctamente.

ERROR	• 1. Si la ciudad de coordenadas *coordX, coordY* no existe en el sistema.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Formato: Ejemplo con dos plantaciones

CoordX1;CoordY1|CoordX2;CoordY2

3.4. Listado de silos

Firma: `Retorno listadoDeSilos();`

Descripción: Retorna la lista de todos los silos registrados en el sistema con su capacidad original y su capacidad remanente. La lista deberá ser retornada en el campo valorString de tipo retorno siguiendo el formato de retorno establecido.

Retornos posibles	
OK	• Siempre retorna OK.
ERROR	• Nunca retorna error.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Formato: Ejemplo con dos silos

CoordX1;CoordY1;CapacidadOriginal1;CapacidadRemanente1|CoordX2;CoordY2;CapacidadOriginal2;CapacidadRemanente2

3.5. Listado de productores

Firma: `Retorno listadoProductores();`

Descripción: Retorna la lista de todos los productores ordenados por cédula en manera ascendente. La lista deberá ser retornada en el campo valorString de tipo retorno siguiendo el formato de retorno establecido.

Retornos posibles	
OK	• Si el reporte pudo ser generado exitosamente.
ERROR	• Nunca retorna error.
NO_IMPLEMENTADA	• Cuando aún no se implementó. Es el tipo de retorno por defecto.

Formato: Ejemplo con dos productores

cedula1;nombre1;celular1|cedula2;nombre2;celular2

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70

Información importante

- Se deberán **respetar los formatos de retorno** dados para las operaciones que devuelven datos.
- **Ninguna** de las operaciones deben imprimir **nada** en consola.
- El sistema no debe requerir ningún tipo de interacción con el usuario por consola.
- Es obligación del estudiante mantenerse al tanto de las aclaraciones que se realicen en clase o a través del foro de aulas.
- **Se valorará la selección adecuada de las estructuras para modelar el problema y la eficiencia en cada una de las operaciones.** Deberá aplicar la metodología vista en el curso.
- Deberá entregar dos copias en CD: conteniendo el código fuente y la documentación (en pdf con las fotos de los integrantes) que entregó impresa.
- Puntos acerca de la documentación impresa:
    - **Incluir las pre y post condiciones de los métodos solicitados**
    - **Incluir un diagrama de la estructura de datos que se implementó para representar el sistema junto con una breve explicación indicando por qué eligió dichas estructuras**
    - **Comentar las pruebas realizadas y los resultados obtenidos**
    - **La documentación tiene un valor de 10 puntos del total del obligatorio, que podrán restarse en caso de estar incompleta o mal presentada.**
    - **No incluir el código fuente en la documentación**
- Para la presentación de la documentación se publicará en aulas.ort.edu.uy un template. (El uso de este template es obligatorio).
- El proyecto será implementado en lenguaje JAVA sobre una interfaz ISistema que se publicará en el sitio de la materia en aulas.ort.edu.uy (El uso de esta interfaz es obligatorio).
- El proyecto entregado debe compilar y ejecutar correctamente en Eclipse.
- No se contestarán dudas sobre el obligatorio en las 48 horas previas a la entrega.
- No se contestarán dudas a través del mail del docente. Las preguntas se deberán hacer en el foro de consultas de aulas.

Facultad de Ingeniería
Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel 902 15 05 Fax 908 13 70

Anexo I: Formatos de retorno

Para las operaciones en las que se debe retornar un tipo complejo (varios valores, o una colección de valores), se define el siguiente formato de manera de serializar el valor y encapsularlo en un único String.

- Si el valor a retornar es una colección de datos, se separarán cada ítem de la colección por un carácter "|".
- Si el valor a retornar es un tipo complejo y tiene más de un atributo (por ejemplo la CI y nombre), se separarán ambos valores por un carácter ";"

Ejemplos:

Retornar la CI y el nombre de una persona:

32551567;Fernando

Retornar una colección de nombres:

Fernando|Esteban|Fabián

Retornar una colección de personas, con sus CI y nombres:

32551567;Fernando|1234567;Esteban|98765432;Fabián

Anexo II: Información útil

Expresiones regulares:
http://www.mkyong.com/regular-expressions/how-to-validate-email-address-with-regular-expression/
http://regexpal.com/

Crear mapas de Google Maps con marcadores:
https://developers.google.com/maps/documentation/staticmaps/?csw=1
Ejemplo:
http://maps.googleapis.com/maps/api/staticmap?center=Montevideo,Uruguay&zoom=13&size=1200x600&maptype=roadmap&markers=color:blue%7Clabel:1%7C-34.90,-56.16&markers=color:red%7Clabel:2%7C-34.91,-56.17&markers=color:green%7Clabel:3%7C-34.905,-56.19&sensor=false

Parsear un string:
http://stackoverflow.com/questions/3481828/how-to-split-a-string-in-java