

Trabajo Final de Estructuras de Datos – FAI – UNCOMA

Red de Distribución de Agua 2025

Descripción general

Se desea desarrollar un sistema en Java que muestre soluciones óptimas de distribución de agua en una red que une a varias ciudades. El principal objetivo es que se entregue agua a todas las ciudades con un aprovechamiento acorde a ciertos criterios de distribución. Los criterios dependen del caudal de las cañerías y la población.

Las ciudades son turísticas, por lo que la cantidad de habitantes que consumen agua varía mes a mes. Al respecto, se conoce una estadística mensual por cada ciudad de los últimos 10 años. Del mismo modo se conoce el aprovisionamiento mensual por cada ciudad de los últimos 10 años.



De las **ciudades** se guarda la siguiente información:

- Nombre
- Cantidad de habitantes por mes y año
- Nomenclatura
 - Formato CI1234:
 - CI son las dos primeras letras del nombre de la ciudad o las iniciales del nombre de la ciudad si el nombre tiene más de una palabra.
 - El número es un valor entre 3000 y 4000)
Ejemplo: para la ciudad de Neuquén, la nomenclatura sería NE3001
- Superficie (m2)
- Cantidad de metros cúbicos promedio consumidos por persona por día

De las **tuberías** se guarda la siguiente información:

- Nomenclatura
 - Formato: NomenclaturaCiudadDesde - NomenclaturaCiudadHasta, por ejemplo: si las ciudades son Neuquén y Cupral-Có sería (NE3001, CC3002)
- Caudal mínimo en metros cúbicos por hora
- Caudal máximo en metros cúbicos por hora
- Diámetro de la tubería en milímetros
- Estado: ACTIVO, EN REPARACIÓN, EN DISEÑO, INACTIVO.

Los criterios de filtro de distribución de agua por ciudad son:

- A discreción, se listan las ciudades que quedan fuera de la distribución en un mes dado.
- Cantidad de habitantes de la ciudad ese mes
- Cantidad máxima de caudal de cañería a cubrir.
- Cantidad mínima de caudal de cañería a cubrir.

Implementación

Estructuras

Para la implementación se deberá utilizar un dígrafo etiquetado (con etiqueta de tipo numérica) para el mapa de las ciudades conectadas por tuberías. La etiqueta será el caudal máximo de la tubería (Esta clase debe ser implementada por el equipo)

Las ciudades se guardarán en una tabla de búsqueda implementada con un árbol AVL, ordenada alfabéticamente por el nombre de la ciudad (Esta clase debe ser implementada por el equipo).

Para el mapeo se deberá utilizar HashMap de Java: como dominio se utilizará una clase con los atributos *nom1* y *nom2* (*no es un String, es una clase de dominio con dos atributos String, que debe implementar el método equals y el método hashCode*), donde *nom1* es la nomenclatura de la ciudad desde donde sale el agua y *nom2* es la nomenclatura de la ciudad hasta donde llega el agua en esa tubería. Como rango la información de la tubería correspondiente según el detalle anterior.

Para calcular la cantidad de agua aprovisionada por mes se debe considerar:

1. La cantidad de agua requerida por cantidad de habitantes ese mes.
2. El camino de aprovisionamiento de agua elegido para ese mes. Este camino elegido será a discreción.
 - El camino elegido define el caudal de la tubería.
 - El camino elegido debe ser un camino activo
 - El cálculo a partir del camino es la cantidad de agua calculada transportando agua sin interrupciones por la cantidad de días del mes correspondiente.
3. El agua aprovisionada para esa ciudad es el valor mínimo entre 1 y 2

Clase principal

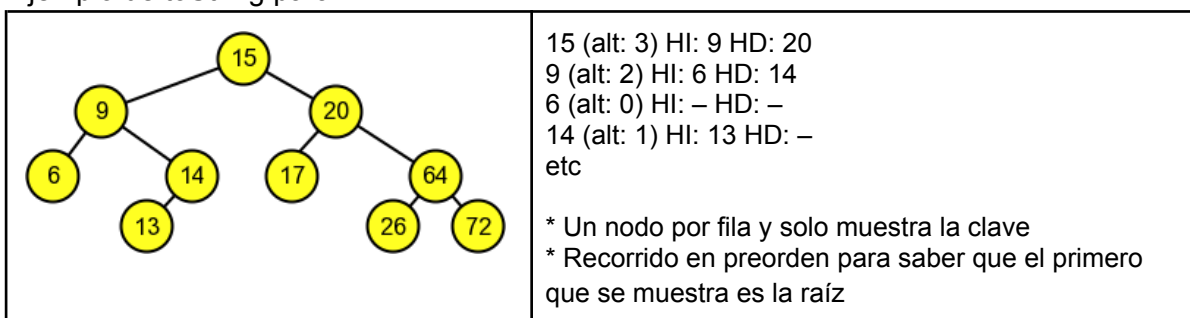
Desarrollar la clase Transporte de Agua con un menú de opciones para realizar las siguientes tareas:

1. Altas, bajas y modificaciones de ciudades
2. Altas, bajas y modificaciones de tuberías (1º versión: solo una tubería por par de ciudades en el mismo sentido)
3. Alta de información de la cantidad de habitantes para año dado y ciudad dada
4. Consultas sobre ciudades:
 - Dada un nombre de ciudad, mes y año determinado mostrar
 - la cantidad de habitantes y
 - el volumen de agua que se habría distribuido en dicha ciudad en dicho mes y año (considerando cantidad de habitantes y metros cúbicos promedio consumidos por persona por día)
 - Dadas dos subcadenas *minNomb* y *maxNomb* (nombres de ciudades), y dos volúmenes de agua (*minVol* y *maxVol*) devolver todas las ciudades cuyo nombre esté en el rango [*minNomb*, *maxNomb*] que en un mes y año determinado hayan consumido un volumen de agua en el rango [*minVol*, *maxVol*].
5. Consultas sobre transporte de agua:

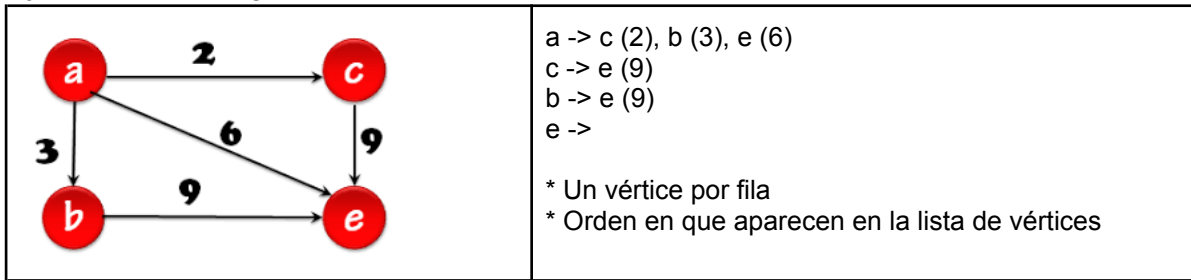
Dada una ciudad A y una ciudad B:

- Obtener el camino que llegue de A a B tal que el caudal pleno del camino completo sea el mínimo entre los caminos posibles. El caudal pleno es el caudal definido por la tubería más pequeña del camino. Decir en qué estado se encuentra el camino.
 - El estado del camino se define en función del estado de todas sus tuberías.
 - Una tubería en reparación indica que todo el camino está en reparación.
 - Una tubería en diseño indica que todo el camino está en diseño.
 - Una tubería inactiva indica que todo el camino está inactivo.
 - Si hay una tubería en diseño y otra en reparación o inactiva, el camino se considera en diseño.
 - Si hay una tubería en reparación y otra inactiva, el camino se considera inactivo.
 - Obtener el camino que llegue de A a B pasando por la mínima cantidad de ciudades. Una vez obtenido el camino decir cuál es el estado.
 - (*) El camino que llegue de A a B utilizando una tubería que trabaje con el mínimo caudal y que no pase por una ciudad C dada
 - (*) Obtener todos los caminos posibles para llegar de A a B, mostrarlos y luego filtrar y mostrar solo los que haya posibilidad de transportar agua hasta un máximo caudal
7. Dado un año, generar y mostrar un listado de las ciudades ordenadas por consumo de agua anual de mayor a menor.
- El consumo de agua anual se calcula como la sumatoria de los consumos mensuales de dicho año.
 - Utilice una estructura de datos auxiliar que considere apropiada para crear el listado que asegure la eficiencia al momento de la consulta. No debe guardar la estructura ni el listado después de mostrarlo.
8. Mostrar sistema: es una operación de debugging que permite ver todas las estructuras utilizadas con su contenido (grafo, AVL y Mapeo) para verificar, en cualquier momento de la ejecución del sistema, que se encuentren cargadas correctamente. En este punto se deberá utilizar el método *toString()* para mostrar claramente cómo están almacenados los datos en la estructura, no es solo un listado. Por ejemplo, para el AVL debe indicarse cuál es la raíz y que para cada nodo se vea quién es su hijo izquierdo y hijo derecho, y la altura de cada nodo. En el Grafo, que se vean los vértices en el orden que estén almacenados y los arcos que lo tienen por origen.

Ejemplo de *toString* para AVL



Ejemplo de toString para Grafo



Clases para investigar

- Leer sobre StringTokenizer o split de String (para fraccionar un String con un carácter separador)
- Abrir archivo de texto para lectura (FileReader, BufferedReader)
- Escribir en archivo de texto (FileWriter)

Requisitos importantes

- El programa debe permitir la ejecución por separado de cada una de las operaciones especificadas mediante un menú.
- El programa debe ser eficiente: Debe recorrer todas las estructuras sólo lo necesario y haciendo buen uso de la memoria.
- Las estructuras deben estar implementadas de forma genérica para elementos de tipo Object o Comparable de Java, según el propósito de la estructura, a menos que se indique lo contrario.
- La carga inicial del sistema debe hacerse en forma de lote, a partir de un archivo de texto. Por ejemplo, la información se puede ingresar en un único archivo, de a un elemento por línea, indicando el tipo (C:Ciudad, T:Tubería, etc), o en archivos diferentes para cada colección.

LA PRE-CARGA LA DEFINE CADA GRUPO ASEGURANDO QUE HACEN TODAS LAS ROTACIONES POSIBLES AL INSERTAR EN EL AVL Y QUE CUENTAN CON UN GRAFO INTERESANTE PARA HACER CONSULTAS. [ATENCIÓN: VER EJEMPLO \(AQUI ENLACE\) DEL CASO ESPECIAL DE PRUEBA EN ELIMINAR DE AVL](#)

- Mantener un archivo .LOG (archivo de texto) para guardar la siguiente información:
 - Estado del sistema al momento de terminar la carga inicial (toString de todas las estructuras)
 - Listado de operaciones que se realizan a lo largo de la ejecución (Ej: "Se insertó la ciudad X", "Se eliminó la tubería Y", "Se cargaron los datos del año X de la ciudad Y", etc)
 - Estado del sistema al momento de terminar de ejecutarse (toString de todas las estructuras)

Condiciones y fechas de entrega

- El programa debe realizarse de manera individual o grupal y deberá ser defendido antes los docentes antes de subirlo a pedco. En la defensa se deberá indicar claramente quién

escribió qué parte del código. Una vez aprobado deberá subirlo a la plataforma PEDCO para su archivo, en el curso llamado “para alumnos preparando final”.

- Al momento de la defensa, deberán presentar un dibujo (en papel o digital) del diagrama del mapa de ciudades (grafo) y de la tabla de equipos (AVL) subido en la carga inicial desde archivo de texto.
- Los estudiantes que promocionan la materia no deberán realizar los módulos marcados con (*) y tendrán tiempo para defenderlo hasta el viernes 1 de agosto de 2025. Deberán desarrollarlo en grupos de 3 estudiantes que estén todos en condición de promocionar la materia. Durante la defensa del trabajo, que será en forma oral y grupal, se tendrá en cuenta que todos demuestren tener conocimiento en profundidad del código desarrollado y que la distribución del trabajo haya sido balanceada en complejidad y código.
- Los estudiantes que no promocionan deberán realizar todos los puntos del trabajo de manera individual o en grupos de 2 personas como máximo. Podrán defenderlo en cualquier momento, pero como mínimo 2 semanas antes de presentarse a rendir el examen final regular. Durante la defensa del trabajo, que será en forma oral (y grupal en caso de ser 2 integrantes), se tendrá en cuenta que demuestren un conocimiento en profundidad del código desarrollado.

Material de estudio

Además de las estructuras vistas hasta el momento, tienen el siguiente material en forma de apuntes y videos, para usar durante el desarrollo del trabajo final:

- [Apunte 5 - Grafos.pdf](#)
 - [EDAT - Video 5.1 - Grafos \(Parte 1\)](#)
 - [EDAT - Video 5.2 - Grafos \(Parte 2\)](#)
- [Apunte 6 - TDAs de Propósito Específico.pdf](#)
 - [EDAT - Video 6.1 - TDAs especiales](#)
- [Apéndice B - Cómo implementar estructuras para un tipo de elemento genérico.pdf](#)
 - [EDAT - Video 4.2 - Clase especial - Cómo crear clases comparables](#)
- [Apéndice C - Como devolver más de un resultado en un método Java.pdf](#)
 - [Video extra: estructuras provistas por Java](#)
- [Apéndice D - Cómo implementa Java las estructuras de datos vistas en la materia.pdf](#)
- PORQUE NUNCA ESTÁ DE MÁS REPETIRLO
- [Apéndice A - Buenas Prácticas de Programación.pdf](#)

Listado de Ciudades

Cada ciudad incluye:

- Nombre
- Nomenclatura
- Superficie (m²)
- Consumo promedio (m³/persona/día)
- Habitantes históricos (no detallados aquí por brevedad, pero deben incluirse en tu implementación)

Nombre	Nomenclatura	Superficie (m ²)	Consumo Promedio (m ³ /persona/día)
Neufuén	NE3001	1200000	0.25
Cupral-Có	CC3002	800000	0.30
Maravélica	MA3003	900000	0.22
Solandina	SO3004	1100000	0.28
Terraviva	TE3005	750000	0.26
Piedraluna	PI3006	1000000	0.29
Valleverde	VA3007	950000	0.27
Monteaurora	MO3008	1050000	0.24
Rionegra	RI3009	890000	0.23
Fuenteluz	FU3010	780000	0.21
Ventisquero Alto	VE3011	920000	0.25
Cielo flor	CI3012	970000	0.28
Bosquemonte	BO3013	860000	0.26

Listado de Tuberías

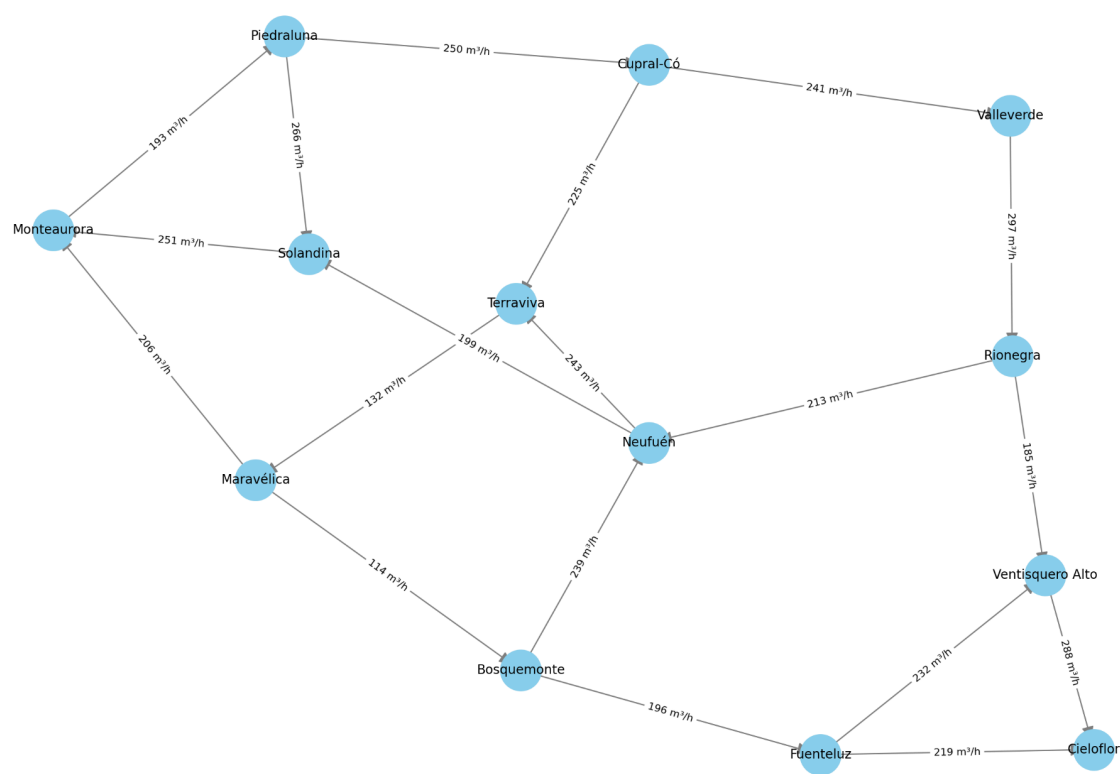
Cada tubería tiene:

- Caudal mínimo (m³/h)
- Caudal máximo (m³/h)
- Diámetro (mm)
- Nomenclatura

Ciudad	Habitantes Prom.	Consumo Total (m³)	Aprovisionado (m³)	Cobertura (%)
Rionegra	11,616	12,003,435	11,755,209	97.93
Terraviva	10,948	14,174,039	13,811,340	97.44
Valleverde	11,239	12,461,847	11,989,397	96.21
Cupral-Có	11,377	13,837,848	13,091,481	94.61
Piedraluna	11,294	15,773,760	14,330,989	90.85
Solandina	10,987	16,995,550	14,870,467	87.50
Monteaurora	11,209	18,262,184	15,503,908	84.90
Fuenteluz	10,624	13,116,456	9,369,003	71.43
Maravélica	11,123	18,208,773	11,359,524	62.38
Bosquemonte	11,381	16,362,879	9,899,859	60.50
Ventisquero Alto	11,120	17,553,696	9,932,555	56.58
Cieloflor	11,037	17,734,735	9,872,552	55.67
Neufuén	11,313	11,188,801	0	0.00

Grafo dirigido

Grafo Dirigido de Aproveccionamiento de Agua (Caudal Mnimo)



Ciudad_Desde	Ciudad_Hasta	Caudal_Max_m3h
Neufun	Rionegra	274
Neufun	Solandina	387
Neufun	Terraviva	443
Neufun	Bosquemonte	148
Cupral-C	Piedraluna	312
Cupral-C	Terraviva	431
Cupral-C	Valleverde	268
Maravlica	Terraviva	456
Maravlica	Bosquemonte	164
Maravlica	Monteaurora	460