Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2233 Programación Avanzada (2018-2)

Tarea 03

Entrega

- **Tarea**
  - **Fecha y hora:** domingo 28 de octubre de 2018, 23:59.
  - **Lugar:** GitHub — Carpeta: `Tareas/T03/`
- **Entregable**
  - **Fecha y hora:** Viernes 19 de octubre de 2018.
  - **Lugar:** Cuestionario de Google Docs
- **README.md**
  - **Fecha y hora:** lunes 29 de octubre de 2018, 23:59.
  - **Lugar:** GitHub — Carpeta: `Tareas/T03/`

Objetivos

- Desarrollar algoritmos para la resolución de problemas complejos.
- Diseñar e implementar estructuras de datos propias basadas en nodos, para modelar y solucionar un problema en concreto.
- Crear, levantar y controlar excepciones.
- Realizar tests unitarios para comprobar el funcionamiento de un programa.

Índice

1. Introducción

Aburrido de jugar a las tragamonedas, Tini decide incursionar en negocios menos conflictivos y decide invertir en la energía eléctrica, negocio que, según las predicciones bursátiles de Nebil, tiene una rentabilidad bastante prometedora. Su empresa quiere desarrollar un programa que pueda levantar información de su red y que así se pueda determinar, por ejemplo, dónde existen problemas de producción, caídas en la red eléctrica, usuarios *colgados*, entre otras cosas. Aquí es donde tú entras para poder solucionar sus problemas.

2. *Electromatic*

Tu programa deberá analizar la estructura de la redes eléctricas chilenas, utilizando estructuras de datos que sean adecuadas para resolver este problema. Para poder modelarlo es necesario entender cómo se compone este complejo sistema, y así poder responder distintas consultas.

En Chile, existen cuatro sistemas eléctricos independientes que suministran energía al país. Estos son el Sistema Interconectado Central (SIC), que cubre el territorio comprendido entre Taltal y la Isla Grande de Chiloé; el Sistema de Aysén (SEA), que atiende el consumo de la Región de Aysén; el Sistema de Magallanes (SEM), que abastece la Región de Magallanes y Antártica Chilena; y finalmente el Sistema Interconectado del Norte Grande (SING). Para lograr abastecer energía estos sistemas reúnen distintas entidades:

1. **Centrales generadoras:** son las encargadas de producir la energía eléctrica. Pueden ser de tipo solar, termoeléctrica o de biomasa.
2. **Estaciones elevadoras:** se encargan de juntar la energía producida por las centrales generadoras. Debe haber como mínimo una en cada sistema eléctrico.
3. **Subestación de transmisión:** transporta la electricidad desde las estaciones elevadoras hasta las subestaciones de distribución. Hay una por provincia.
4. **Subestaciones de distribución:** distribuyen la energía hacia los consumidores. Hay una o más por comuna.
5. **Entidades de consumo:** son los consumidores de energía, como por ejemplo, casas. Por ende, son el punto de término de la red eléctrica. Hay varias entidades de consumo por comuna.

Como podrás notar más adelante por las descripciones que se entregarán, una estación elevadora y una subestación de transmisión tienen una relación 1 a n. Esto implica que una subestación de transmisión puede recibir sólo de una estación elevadora, pero una estación elevadora puede estar conectada a **una o más** subestaciones de transmisión. Lo mismo ocurre entre una subestación de transmisión y una subestación de distribución.

3. Red eléctrica

Como ya se mencionó, las centrales aumentan la capacidad del sistema. En palabras simples, estas son las que *nutren* de energía —o más bien, potencia— eléctrica a la red, la cual es consumida por las casas. Dado esto, es lógico que las centrales entreguen la potencia justa y necesaria para satisfacer la demanda de la red. Por ende, esta *potencia total demandada* debe calcularse, de manera que no exista exceso de energía (u oferta). Es por esto que tu programa debe ser capaz de calcular ese número. Es importante

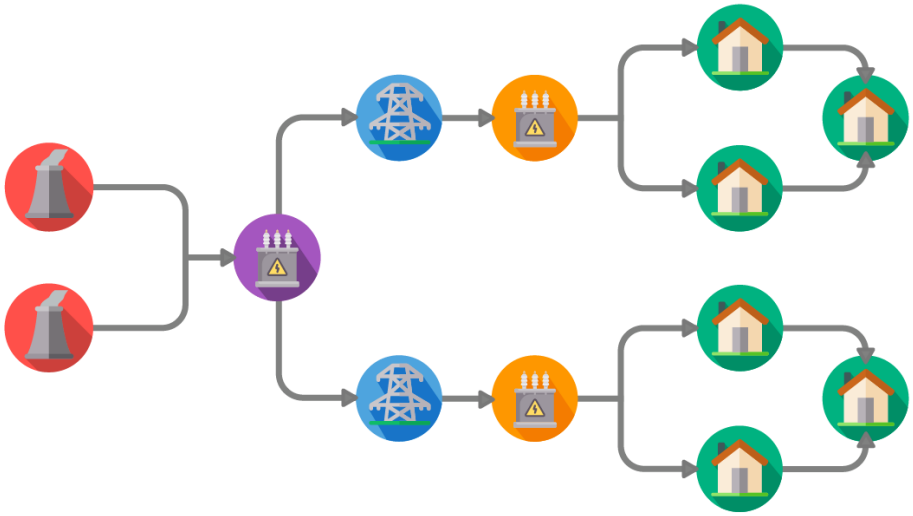

Figura 1: Sistema eléctrico. En rojo están las centrales generadoras; en morado, las estaciones elevadoras; en azul, las subestaciones de transmisión; en naranja, las subestaciones de distribución; y finalmente en verde, las entidades de consumo.

mencionar que a lo largo del camino se va perdiendo potencia, ya que la resistencia de los cables deja de ser despreciable en estas escalas. Por lo tanto, se termina consumiendo ciertas fracciones de la potencia entregada. Todo esto se explica a continuación.

3.1. Cálculo de la demanda de la red

Para explicar esto, nos apoyaremos en el diagrama de la Figura 2. Es importante que leas con detención esta parte. Si bien no es una matemática compleja, entender los pasos que se presentan será vital para luego generalizar el ejemplo a la red real. Para evitar complicaciones innecesarias, se debe tener en mente que **no deberá haber ciclos dirigidos**[1] en la red eléctrica, en específico, en la distribución de casas.

En este diagrama de ejemplo, podemos observar la "última parte de la red". Cada casa tiene una demanda conocida y constante, representado por c_i. Luego, las aristas entre casas representan los cables por donde pasa la energía que consumen las casas, que tienen un largo **en kilómetros** denotado por $L_{i,j}$, donde i es la casa de donde *sale* el cable, y j la casa a donde *llega*. Se asume que la energía fluye en un sólo sentido, desde i hacia j.

Antes de empezar a explicar el calculo de demanda, debemos dejar en claro que para calcular perdidas de potencia en un cable nos basamos en la siguiente formula:

$$P_{perdida} = PR$$

Donde P es la potencia que actualmente fluye por el cable y R es la resistencia del cable.

Para simplificar el problema general, definimos $f_{i,j}$ como la función que determina la pérdida de potencia a través del cable que va desde i hasta j, como:

$$f_{i,j} = M_{i,j}R_{i,j}$$

[1] En un grafo, un ciclo dirigido se define como un camino el cual empieza y termina en el mismo nodo.

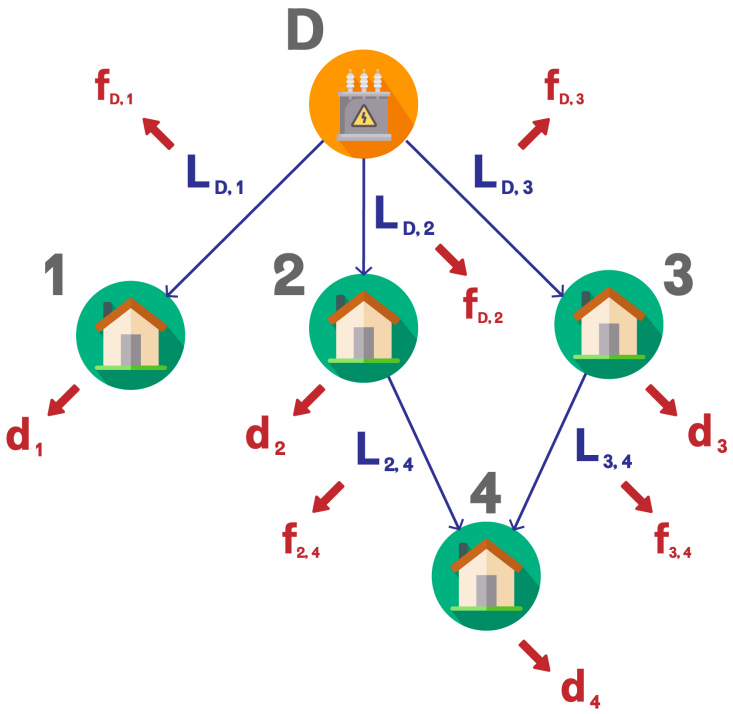

Figura 2: Diagrama de ejemplo distribuidora-casas

Donde $M_{i,j}$ es la potencia que debe enviar la casa i a la casa j:

$$M_{i,j} = P_j + f_{i,j}$$

$$M_{i,j} = P_j + M_{i,j}R_{i,j}$$

Donde P_j es la potencia que necesita el nodo hijo. En palabras simples $M_{i,j}$ es lo que el nodo hijo gasta (P_j) más lo que se pierde en la conexión ($M_{i,j}R_{i,j}$). Sí, esto se ve como una recursión bastante fea, pero se arregla así:

$$M_{i,j} - M_{i,j}R_{i,j} = P_j$$

$$M_{i,j}(1 - R_{i,j}) = P_j$$

$$M_{i,j} = \frac{P_j}{1 - R_{i,j}}$$

Y $R_{i,j}$ es la resistencia del cable que va de i hasta j. A su vez, la resistencia **de un tramo** es calculada por la siguiente fórmula:

$$R_{i,j} = \rho\frac{L_{i,j}}{S}$$

Donde $R_{i,j}$ es la resistencia, ρ la resistividad del material, $L_{i,j}$ es largo del cable (en kilómetros) y S es la sección transversal (o área) del cable. Para efectos de esta tarea, ρ se asumirá como la resistividad del cobre, la cual toma un valor de $\rho = 0,0172\frac{mm^2\Omega}{km}$. Finalmente, la sección transversal está dada por el tipo

de cable que se utiliza, el cual depende de las instalaciones que se están conectando. Un listado de las conexiones, junto al área transversal de sus cables, se señala a continuación:

Emisor	**Receptor**	**Área Transversal**
`Centrales eléctricas generadoras`	`Estaciones elevadoras`	`253 mm`^{2}
`Estaciones elevadoras`	`Subestaciones de transmisión`	`202,7 mm`^{2}
`Subestaciones de transmisión`	`Subestaciones de distribución`	`152 mm`^{2}
`Subestaciones de distribución`	`Casas`	`85 mm`^{2}
`Casas`	`Casas`	`85 mm`^{2}

Para comenzar calculando la demanda total, debemos partir de "abajo hacia arriba", considerando siempre la premisa de que la oferta debe satisfacer la demanda en cada casa. En el ejemplo, d_4 es conocida, por lo que la potencia que debe haber disponible para esa casa será simplemente:

$$P_4 = d_4$$

Luego, para d_3, seguimos con la misma idea, pero ahora considerando que la casa 4 esta *colgada* de la casa 3, es decir, no tiene conexión directa con una subestación. La potencia que la casa 3 requiere es:

$$P_3 = d_3 + \sum_j M_{3,j}$$

Donde j recorre a todos los nodos hijos de la casa 3, como solo tiene un nodo hijo lo resultante es:

$$P_3 = d_3 + M_{3,4}$$
$$P_3 = d_3 + \frac{\frac{P_4}{2}}{1 - R_{3,4}}$$

Como te podrás haber dado cuenta, en esta ecuación existe un supuesto: **la potencia demandada de una casa colgada se distribuye de igual manera entre sus casas *proveedoras*.** En este sentido, si hay una o más conexiones, entonces la distribución sería:

$$\frac{P_i}{n} \qquad n : \text{cantidad de nodos proveedores}$$

Ahora reemplazamos la formula de la resistencia:

$$P_3 = d_3 + \frac{\frac{P_4}{2}}{1 - \frac{\rho L_{3,4}}{S}}$$

Con este paso, ya tenemos una expresión cerrada para la demanda de potencia que tiene cada casa, la cual aumentará a medida que aumente el largo del cable hacia sus casas colgadas. El resultado anterior se puede replicar para el resto de las casas del ejemplo:

$$P_2 = d_2 + \frac{\frac{P_4}{2}}{1 - \frac{\rho L_{2,4}}{S}}$$

$P_1 = d_1$, dado que la casa 1 no tiene casas colgadas.

Ahora subimos un nivel en la red, y (con la misma lógica) obtenemos la demanda de potencia de toda la distribuidora de energía asociada a ese conjunto de casas (P_D):

$$P_D = d_D + \sum_j M_{D,j}$$

$$P_D = d_D + M_{D,1} + M_{D,2} + M_{D,3}$$

$$P_D = d_D + \frac{P_1}{1 - R_{D,1}} + \frac{P_2}{1 - R_{D,2}} + \frac{P_3}{1 - R_{D,3}}$$

$$P_D = d_D + \frac{P_1}{1 - \frac{\rho L_{D,1}}{S}} + \frac{P_2}{1 - \frac{\rho L_{D,2}}{S}} + \frac{P_3}{1 - \frac{\rho L_{D,3}}{S}}$$

Algo importante a considerar es que todas estas son aristas ***dirigidas***, por lo que es diferente decir, por ejemplo, $f_{D,1}$ a decir $f_{1,D}$. Adicionalmente, es importante notar el hecho de que los P_i tienen incluida la demanda de sus hijos en sus respectivas expresiones; luego, al ir subiendo en la red, se va almacenando toda la información necesaria en los nodos padres.

Siguiendo la misma lógica, podemos obtener tanto P_T como P_E, correspondiente a las líneas de transmisión y elevadoras, correspondientemente.

El último problema que faltaría resolver es cómo se distribuye la demanda de P_E para las centrales que le proveen la energía. Para esto queda a tu criterio la manera en que se distribuye el consumo energético de la estación elevadora.

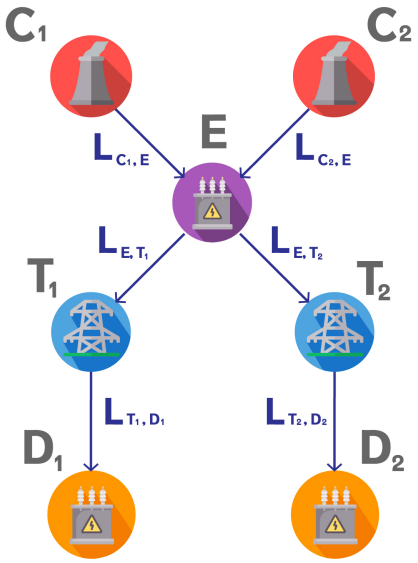

Figura 3: Diagrama Centrales

3.2. Flujo de potencia en la red

Dada una energía generada por la centrales, la red debe ser capaz de distribuirla. Si la cantidad producida es mayor a la cantidad demandada por la red, entonces simplemente se estará satisfaciendo la demanda de todos los nodos y la energía sobrante se perderá.

Ahora bien, si la energía no es la suficiente para satisfacer la demanda de todos los nodos, debes simular el flujo para saber cuanta energía esta llegando realmente a los nodos. Para simular el flujo debes tener en cuenta los siguientes puntos:

- Cuando llega potencia a un nodo debes restarle la demanda propia de ese nodo a la potencia y lo que quede será lo que ira a los nodos hijos.
- Para repartir la potencia entre los nodos hijos debes dividir proporcionalmente la potencia disponible, es decir si hay un nodo con 2 nodos hijos donde el primer hijo necesita 80kW y el segundo 20kW, si al nodo padre le llegasen 50kW entonces deberá repartir 40kW al primer hijo y 10kW al segundo. Para calcular la proporcionalidad ($K_{i,j}$) debes utilizar el valor $M_{i,j}$ que ya habías obtenido cuando hiciste el calculo de la demanda de la red, es posible calcular $M_{i,j}$ cuando no se satisface completamente a la red ya que solo depende de las demandas de los nodos y no del consumo real que están teniendo actualmente.
- Al hacer pasar potencia por una conexión debes restar la perdida correspondiente (PR_{ij}) y lo que quede será lo que efectivamente llegue al nodo hijo.
- Para calcular tanto P_j como $f_{i,j}$ para un nodo i con su hijo j al cual no se le puede cubrir su demanda completa. Se debe calcular $K_{i,j}$, proporcionalmente según fue indicado arriba. Y $K_{i,j}$, que representa la potencia asignada a dicho nodo, reemplazará el valor de $M_{i,j}$ en las ecuaciones entregadas previamente.
- Si un nodo tiene potencia suficiente para suplir su propia demanda, siempre debe proporcionar energía a los nodos a los que este provee, si la potencia no le alcanza para satisfacer su propia demanda entonces gastará toda la potencia en si mismo y entregara 0 hacia sus nodos hijos.

4. Modificación de la red

4.1. Agregar y remover aristas (conexiones)

En el caso de las casas, sólo se debe permitir agregar conexiones entre casas de una misma comuna o casas con la(s) subestación(es) de la comuna.

Por otro lado, se debe poder tratar (y si se puede, efectivamente hacerlo) remover cualquier arista entre cualquier tipo de entidades.

4.2. Agregar nodos

La empresa de Tini quiere conocer qué ocurriría con el sistema eléctrico del país, de un sector o de algunos hogares cuando se añade un nuevo nodo de generación, distribución o consumo a la red.

Cuando se añade un nuevo nodo a la etapa de **generación** aumenta la capacidad del sistema eléctrico al que se incorpora. Este incremento de la energía disponible es recibido por la estación elevadora que pertenezca al mismo sistema eléctrico, la cual mediante un análisis de la red que le subyace, revisa si es

necesaria la aplicación de un mayor flujo de energía hacia la red. Si se agrega un nodo generador pueden ocurrir dos cosas:

- De no necesitar más energía, la red no se verá afectada, simplemente no se utilizará la energía extra.
- En caso de que no se tenga la energía necesaria, se debe distribuir la nueva energía a todos los nodos que la necesiten. Si, por ejemplo, a un nodo consumidor le falta energía, aumenta el flujo que pasa por las líneas de transmisión, la subestación eléctrica y finalmente, el consumidor. La energía eléctrica no se teletransporta entre nodos, va fluyendo en la red.[2]

La energía generada por cada planta está especificada en la base de datos. Sin embargo, para las que se agregan después, se debe pedir la cantidad de energía que esta generará. Esta deberá ser un entero entre 20 y 200 MW. Adicionalmente, se debe pedir que se especifique qué tipo de planta será (termoeléctrica, solar o de biomasa).

Por otro lado, cuando se añade una nueva estación elevadora, éste **debe poder asociarse a alguna(s) generadora(s)** para obtener energía eléctrica. Si se quiere asociar a una generadora que ya está ocupada, la energía de la generadora se divide entre las dos estaciones elevadoras.

Finalmente, cuando se añade un nuevo **consumidor** en la red, la energía que fluya hacia él dependerá de dónde se encuentre. Por ejemplo:

- Si el nuevo nodo está conectado a una subestación, si se agrega una casa "5" a la subestación, entonces a P_D se le agrega:

$$M_{D,5}$$

- Si el nuevo nodo está conectado a una casa, tal como se explicó anteriormente, si una casa "5" está conectada a más casas (como la 3), entonces la casa cambia su potencia a:

$$P_3 = c_3 + M_{3,4} + M_{3,5}$$

Como podrán notar, el hecho de agregar un nuevo nodo a la red puede generar que otros nodos se vean afectados, por lo que deben recalcular la demanda de la red y los flujos de energía.

4.3. Remover nodos

Por otro lado, la empresa de Tini también quiere saber qué ocurre al remover un nodo de un sistema, por lo que debes permitir realizar esta acción sobre un sistema eléctrico. Al quitar un nodo de **generación**, la energía recibida por la estación elevadora disminuye, afectando a los nodos inferiores que estén conectados a esta generadora. La estación elevadora que estaba relacionada a esta generadora ahora tiene menor cantidad de energía fluyendo por ella.

Si se quita un nodo de **elevación**, la energía que le entregaban las centrales generadoras se desvía hacia otra central elevadora. También se elimina la conexión con todos los nodos inferiores. Si no hay más estaciones elevadoras, la energía se pierde.

Si se quita un nodo de **distribución**, la energía que le entregaba la subestación de transmisión se reparte entre el resto de las subestaciones distribuidoras. También se elimina la conexión con todos los nodos infe-

[2]Puedes pensarlo como una red de agua: si se necesita más agua en un nodo al final de la red, entonces pasará más agua por cada nodo anterior.

riores. Si no hay más subestaciones distribuidoras entonces la demanda de la subestación de transmisión será solo la demanda propia. Si se quita un nodo de **transmisión** el efecto es análogo.

Si se quita un **consumidor** conectado a una subestación se reparte nuevamente la energía entre los consumidores de la subestación. Sin embargo, si es que el consumidor está conectado *colgado* a otro nodo consumidor, se elimina la conexión y la energía se redistribuye a los otros nodos que están *colgados* de la misma casa que el nodo removido, si no hay mas nodos *colgados* entonces la energía se queda en el consumidor de origen (el que entrega la energía). En el caso que uno de los nodos exceda el máximo flujo, se debe lanzar el error `ElectricalOverload`, explicado más adelante.

Cuando se remueve un nodo que tiene nodos hijos como es el caso de la casa 2 en la figura 2, si es que el nodo hijo **no** tiene otro nodo padre más que el removido, entonces el nodo hijo es removido también, ya que la única conexión a la red era mediante el nodo removido. Si el nodo hijo **sí** tiene otro nodo padre, como es el caso de la casa 4 que tiene tanto a la casa 2 como a la casa 3 de padres, entonces al remover la casa 2, la casa 4 no será removida ya que sigue estando conectada a la red mediante la casa 3.

Para poder tener un mayor control sobre lo que está pasando, Tini quiere que puedas simular estas acciones, pero también que se le permita efectivamente realizar el cambio, haciendo que realmente se agregue el nodo (en caso de poderse) al sistema eléctrico. Esto se puede entender de la forma en que hay que tener dos opciones "ver qué pasa si se agrega un nodo en cierto lugar" y "efectivamente agregar ese nodo".

5. Consultas

Luego de haberte entregado toda esta información, Tini te da una lista con ciertos datos que hasta ahora no se han podido recolectar en su negocio. También añadió una pequeña explicación de lo que espera que le retornes y en el formato que lo necesita.

- **Energía total consumida en una comuna:** dado un *string* de comuna, se debe retornar el consumo total de esa comuna en kW. Esto es la suma del consumo de energía de cada consumidor dado el `id` de esa subestación y además deben entregar el porcentaje de ese consumo con respecto al consumo total de la red.
- **Clientes con mayor consumo:** dado la `sigla` de una sistema eléctrico, se debe retornar el `id, nombre, provincia y comuna` de la casa con mayor consumo eléctrico en el momento.
- **Clientes con menor consumo:** dado la `sigla` de una sistema eléctrico, se debe retornar el `id, nombre, provincia y comuna` de la casa con menor consumo eléctrico en el momento.
- **Potencia perdida en transmisión:** dado un `id` de una casa, se debe retornar la energía **total** disipada en la transmisión de electricidad que le llega a esa casa actualmente desde la estación elevadora.
- **Consumo de una subestación:** dado el `id` de una subestación de **transmisión** o de **distribución**, se debe retornar el consumo total que esta subestación posee, esto es su consumo propio más los de los nodos hijos más las perdidas de los cables.

IMPORTANTE: Cuando se habla de consumo no necesariamente es el valor entregado en la base de datos porque puede que la potencia generada en la red no satisfaga en su totalidad todas las demandas, el consumo es lo que efectivamente esta llegando a los nodos. Lo mismo sucede con la potencia perdida en la transmisión.

6. Excepciones

Para esta tarea, se espera que puedan manejar y levantar todos los errores posibles mediante el uso correcto de excepciones. Para esta tarea, **una excepción genérica**[3] **no se considerará correcta**.

Tu programa debe, además, implementar las siguientes excepciones:

- `ElectricalOverload`

  Esta excepción se levanta cuando, al intentar modificar la red, se genera una **sobrecarga eléctrica**. Si se estaba intentando remover o agregar un nodo de manera **permanente**, no se debe permitir que se continúe.

  Cuando se levanta este error, se debe indicar la función que la causa, la cantidad de corriente que correría por el nodo fallido y el máximo de corriente. Por ejemplo:

  - `"ElectricalOverload: La acción remover_nodo sobrecarga la red a 120 kW."`
  - `"ElectricalOverload: La acción agregar_nodo sobrecarga la red a 420 kW."`

  En este ejemplo se asume que una función (`remover_nodo`, `agregar_nodo`) causa que la red sobrepase el máximo.

  Límite de potencia en las casas: las casas pueden soportar una tensiones eléctricas muy altas por riesgo de incendio. El límite de potencia eléctrica es de acuerdo a la norma que estipula el DCC, que en este caso tienen un máximo de 30.000kW.

- `ForbiddenAction` Esta excepción se levanta si se intenta realizar una operación prohibida en su estructura de datos.

  Un ejemplo de este error podría ser intentar conectar una casa directamente a una central generadora, lo cual no es posible. Otro ejemplo similar sería intentar conectar una segunda estación elevadora a un sistema eléctrico que ya posee uno.

  Al levantar este error, se debe identificar la función que la causa y el tipo de las entidades que la causan. Por ejemplo:

  - `"ForbiddenAction: Acción agregar_arista no está permitida entre Casa y CentralGen."`
  - `"ForbiddenAction: Acción agregar_arista no está permitida entre casas que tienen" "comunas distintas."`
  - `"ForbiddenAction: Acción agregar_arista no está permitida entre porque se formaría" "un ciclo."`

- `InvalidQuery`

  Esta excepción se levanta si la consulta es inválida, ya que los parametros dados son inválidos. a Ejemplo de esto sería, por ejemplo, buscar la cantidad total de energía que consume una comuna inexistente o buscar la potencia perdida en una casa que no existe.

  Al levantar el error se debe identificar el parámetro incorrecto. Por ejemplo:

  - `"InvalidQuery: La comuna 'Rancagua' no existe."`
  - `"InvalidQuery: No existe una casa con id '20'."`

[3] `except Exception:`

Estas consultas deben ser levantadas cuando sea necesario, sin embargo su programa **no deberá caerse** al levantarse estas excepciones, sino que deben manejarlas ustedes mismos, imprimiendo un mensaje con el error levantado y continuando el funcionamiento del programa normalmente.

7. Testing

En un modulo llamado `testing.py` deberás probar todas las consultas indicadas en la sección consultas y todas las excepciones indicadas en la sección excepciones.

Para testear las consultas, todas deben probarse con un caso y en dos de ellas probar dos casos de uso distintos. Para realizar los tests, debes utilizar la librería de Python `unittest` sobrescribiendo, de ser necesario, los métodos de `setUp` y `tearDown`. Para realizar el testing se podrán crear archivos[4].

Para realizar el testing de las consultas, será necesario que las realices por medio de los **métodos de aserción** correspondientes, testeando que el output de las consultas sea correcto.

```
def test_consulta(self):
    res_consulta = {} # Resultado ideal de la consulta
    self.assertEqual(consulta(), res_consulta)
```

Para realizar testing de una excepción, deberás encontrar una consulta que la lance y testear con ella. Un ejemplo sería:

```
def test_excepcion(self):
    self.assertRaises(Excepcion, consulta, parametros_consulta)
```

8. Interacción con consola

Tu programa debe tener un menú en el cual se pueda:

- Obtener los resultados de las consultas.
- Agregar o remover nodos de cualquier tipo (centrales eléctricas, casas, etcétera). Cabe mencionar que un nodo puede conectarse con más de otro nodo, por lo que se debe poder conectar un nodo nuevo con más de un nodo existente.
- Probar qué pasaría si se realizara cierto cambio en la red, al agregar o remover un nodo.

Adicionalmente, la consola debe ser a prueba de **todo** *string* que pueda colocar el usuario en ella, manejando de manera correcta las excepciones que se levanten.

9. Bases de datos

Centrales generadoras

La información sobre las **centrales generadoras** se encuentra en el archivo `centrales.csv`, donde cada fila representa una central con su información. Cada columna del archivo representa lo listado a continuación:

[4] Los archivos deben tener casos de prueba cuyo resultado es conocido.

Nombre	Tipo de dato	Explicación
id	integer	Identificador único de la central.
nombre	string	Nombre de la central.
sistema_eléctrico	string	Sigla del sistema eléctrico al cual pertenece.
provincia	string	Nombre de la provincia a la que pertenece.
comuna	string	Nombre de la comuna a la que pertenece.
tipo	string	Tipo de central generadora.
potencia	float	Capacidad máxima que produce la central (MW).

Subestaciones

La información sobre las **estaciones elevadoras, subestaciones de transmisión y subestaciones de distribución** se encuentra en tres archivos diferentes, `elevadoras.csv`, `transmision.csv` y `distribucion.csv` respectivamente, donde cada fila representa una estación o subestación con su información. Cada columna de los archivos representa lo listado a continuación:

Nombre	Tipo de dato	Explicación
id	integer	Identificador único de la estación o subestación.
nombre	string	Nombre de la subestación.
sistema_eléctrico	string	Sigla del sistema eléctrico al cual pertenece.
provincia	string	Nombre de la provincia a la que pertenece.
comuna	string	Nombre de la comuna a la que pertenece.
consumo	float	Cantidad de energía consumida por la subestación (MW).

Casas

La información sobre los consumidores, es decir, las **casas** se encuentra en el archivo `casas.csv`, donde cada fila representa una casa con su información. Cada columna del archivo representa lo listado a continuación:

Nombre	Tipo de dato	Explicación
id	integer	Identificador único de cada casa.
sistema_eléctrico	string	Sigla del sistema eléctrico al cual pertenece.
provincia	string	Nombre de la provincia a la que pertenece.
comuna	string	Nombre de la comuna a la que pertenece.
consumo	float	Cantidad de energía gastada por las casas (kW).

Conexiones

La información sobre las conexiones viene en archivos donde sus nombres representan las entidades conectadas, es decir, el archivo que se llama `transmisión_elevadoras.csv` guarda la información de las conexiones entre subestaciones de transmisión y estaciones elevadoras. Cada columna de estos archivos representa lo listado a continuación:

Nombre	Tipo de dato	Explicación
`id_entidad1`	`integer`	Identificador único de la primera entidad.
`id_entidad2`	`integer`	Identificador único de la segunda entidad.
`distancia`	`float`	Distancia entre ambas entidades medida en kilómetros.

Por brevedad se mostró la estructura del archivo de manera genérica. Para el caso de las conexiones entre casas las columnas de los `id` se llaman `id_desde` e `id_hasta` para representar la direccion de la conexión.

10. Entregable

Deberás investigar o crear una forma para calcular la demanda de la red. Para esto debes entregar un documento escrito en Markdown (`.md`) que explique bien el algoritmo a seguir para calcular la demanda de la red, y una justificación de por qué funciona. No es necesario que sea extenso; trata de ser lo más conciso posible.

11. Notas

Como podrás haber notado, en distintas partes del problema se utilizan distintas unidades (*e.g.* kW, MW), por lo que tu programa debe ser robusto frente a esto. En este sentido, es **muy** importante que te fijes en las **unidades** en que se encuentra cada parte, ya que sino se podrían generar inconsistencias.

12. Restricciones y alcances

- Está **prohibido el uso de cualquier tipo de estructura de dato que venga con Python (como listas, diccionarios, tuplas, etcétera), por lo que para poder hacer** uso de este tipo de EDD, se deberá hacer la implementación propia de la estructura deseada. Tampoco se puede hacer cosas que devuelvan estructuras que sean *built-in* en Python. Lo único que sí se puede es desempaquetar una lista (o una tupla, diccionario, etcétera) para usarla, pero en una estructura de datos hecha por ti (en testing sí se puede).
- Esta tarea es **estrictamente individual**, y está regida por el **Código de honor de Ingeniería**.
- Tu programa debe ser desarrollado en Python 3.6.
- Si no se encuentra especificado en el enunciado, asume que el uso de cualquier librería Python está prohibido. Pregunta en la *issue* especial del foro si fuese posible utilizar alguna librería en particular.
- Debes adjuntar un archivo `README.md` **conciso y claro**, donde describas los alcances de tu programa, cómo correrlo, las librerías usadas, los supuestos hechos, y las referencias a código externo. **Tendrás hasta 24 horas después del plazo de entrega** de la tarea para subir el *readme* a tu repositorio.
- Tu tarea podría sufrir los descuentos descritos en la **guía de descuentos**.
- Entregas con atraso de más de 24 horas tendrán calificación mínima (1,0).
- Cualquier aspecto no especificado queda a tu criterio, siempre que no pase por sobre otro.

Las tareas que no cumplan con las restricciones del enunciado obtendrán la calificación mínima (1,0).