- Implementar todos los constructores.
- A los constructores con super enviarle todos los parámetros del super.
- Para test seguir la estructura de la imagen y declarar a qué clase pertenece cada test y un valor ejemplo con su resultado.
- Conforme vaya haciendo los test de un método ir testeando los métodos que contenga.
- Terminar de leer bien la consigna, hacer un pseudoUML con las clases, sus relaciones, atributos y métodos que necesitaran.
- Ir método por método solicitado y analizar cómo va a ser su solución e identificar qué otros métodos necesitaré implementar en otras clases para realizar el cálculo solicitado.
- Importante no tener envidia de atributos "Si una clase tiene los atributos necesarios para realizar un cálculo, delegarle el método a esa clase e invocarlo cuando lo necesite" (Hará que no tenga clases anémicas).
- "NUNCA modificar la colección de una clase en otra clase, siempre delegar a la clase dueña.
- No pasar fechas.now por parametro.
- NUNCA repetir código. En caso de jerarquía modularizar
- Verificar que los métodos realmente cumplan con lo solicitado.
- Los constructores de subclases siempre van a necesitar por parámetro los parámetros que requiera los constructores de la superclase, agregando los propios.
- Si no tiene métodos ni atributos, entonces es una interfaz.
- Si tengo que utilizar un atributo de una clase pasarla como objetos y hacerle un método que lo cambie él mismo, sino terminaría rompiendo el polimorfismo.
- en el diagrama UML los métodos abstractos también ponerlos en la subclases (si una clase abstracta hereda a otra clase abstracta, no reescribir el método abstracto heredado).
- hacer una hoja por cada clase e ir haciendo punto por punto, implementado los métodos que necesite en cada clase para cada punto del parcial.
- utilizar flechas bidireccionales, aclarar cardinalidades de ambos lados y recurrir a la composición.
- atento a las pequeñas aclaraciones de las consignas, que marcan MUCHO cómo será el modelado del sistema, tener en cuenta futuras cosas que debe de respetar a la hora de modelar, por ej reemplazo de clases o futuro escalamiento en cuanto a cómo crece una jerarquía.
- UML: flecha de composición es --->, no la punta negra rellena
En las cardinalidades no solo decir *, si es un grupo por ejemplo hacer 2..asterisco.
- NO REPETIR CÓDIGO, SI PUEDO FILTRAR ALGO EN EL STREAM HACERLO AHÍ.
- PONER EN EL SUPER() DE LOS CONSTRUCTORES LOS PARÁMETROS DEL CONSTRUCTOR DE LA SUPER CLASE.
- Averiguar el uso de coleccion.forEach y agregarlo al apunte. Buscar otras que podrían servirme en streams o no.
- Si se envía un parámetro y no lo usan todas las clases no es tan grave, pero se puede solucionar jugando con qué método invoca, lo del lookup. También se puede invocar otro método que obtenga el valor faltante.
- poner en protected las variables que estén en una herencia para poder usarlas en una subclase.

MÉTODOS A TESTEAR SIEMPRE:
-LOS QUE TENGAN UN IF
-LOS QUE TENGAN UNA COLECCIÓN (POSIBLEMENTE VACÍA)