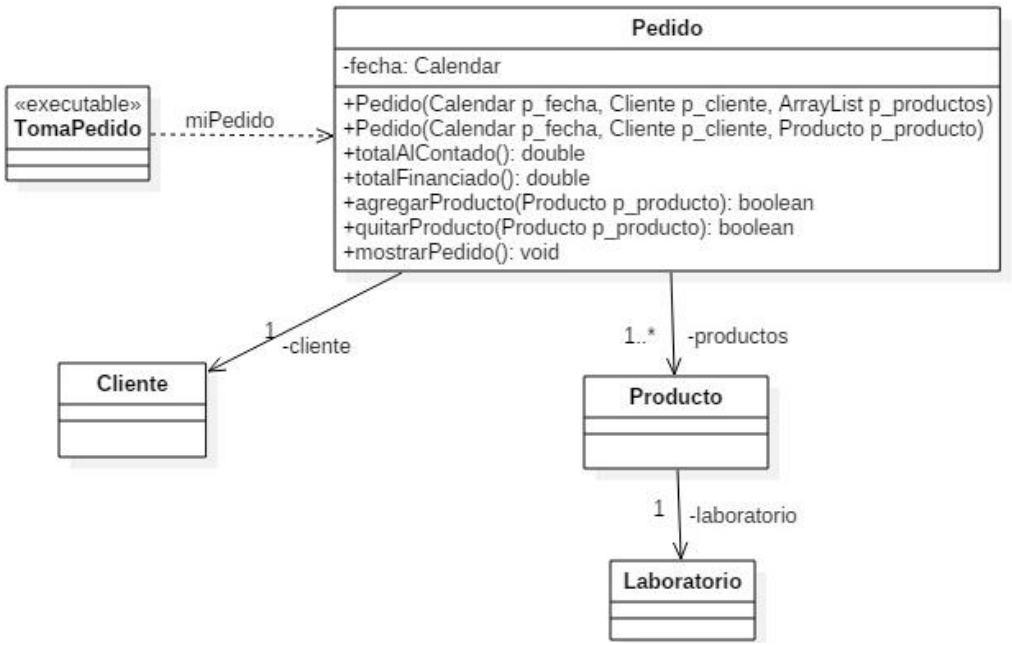
Objetivo: Que el alumno:
- Aprecie la ventaja de la reutilización del código, a través de la instanciación de **clases predefinidas de Java** (colecciones)
- Aprenda a manejar (agregar, eliminar y recuperar un elemento) colecciones de objetos de una clase
    - Colecciones estáticas/dinámicas
    - Colecciones homogéneas/heterogéneas
    - Colecciones indexadas

Notas:
- En todos los casos, al trabajar con colecciones, definir 2 constructores:
    - el que recibe una colección, representando el conocimiento ***** ó **n**
    - el que representa el caso **0** ó **1** (es decir **0..*** ó **1..***)
        - ✓ **caso 0**: el constructor no recibe nada → crea la colección con **this.setMiColeccion(new ArrayList())**
        - ✓ **caso 1**: el constructor recibe un objeto elemento de la colección → crea la colección y le adiciona el objeto
- Se solicita resolver algunos ejercicios con genéricos y otros utilizando cast al recuperar elementos
- En todos los casos, reutilizar las clases existentes
- Implementar al menos 1 ejercicio con menú para selección de opciones

1. Crear una clase ejecutable llamada **ArrayDePuntos**, donde se haga lo siguiente:
   1.1. Crear un contenedor estático de seis elementos de tipo Punto, llamado **puntos**.
   1.2. Agregar como elementos instancias de la clase **Punto**. Ingresar los datos por teclado.
   1.3. Recorrer el array e imprimir las coordenadas de cada elemento, utilizando el método previsto en la clase Punto.
   1.4. Imprimir en pantalla la distancia cada 2 elementos consecutivos del contenedor (Ej: distancia pto1-pto2, distancia pto2-pto3, etc)

2. Una empresa desea implementar una aplicación para automatizar la toma de pedidos de sus clientes. El modelo de la aplicación es el siguiente:



La clase Pedido es la responsable de mantener la lista de productos solicitados, por lo tanto, debe proveer la interfaz necesaria para agregar o quitar un producto a la lista de productos solicitados. Hay que tener en cuenta que esta colección de pedidos presenta la característica de que cada elemento se agrega al final (no se agregan productos en el medio de la lista) y se mantiene el orden en que se van agregando, por tal motivo se sugiere usar una clase ArrayList (que implementa la interface List y reúne el comportamiento de una lista). Un punto a considerar es que se trabaja por unidad, es decir, si un producto se pide en más de una unidad, se lo agrega la cantidad de veces solicitada a la lista.

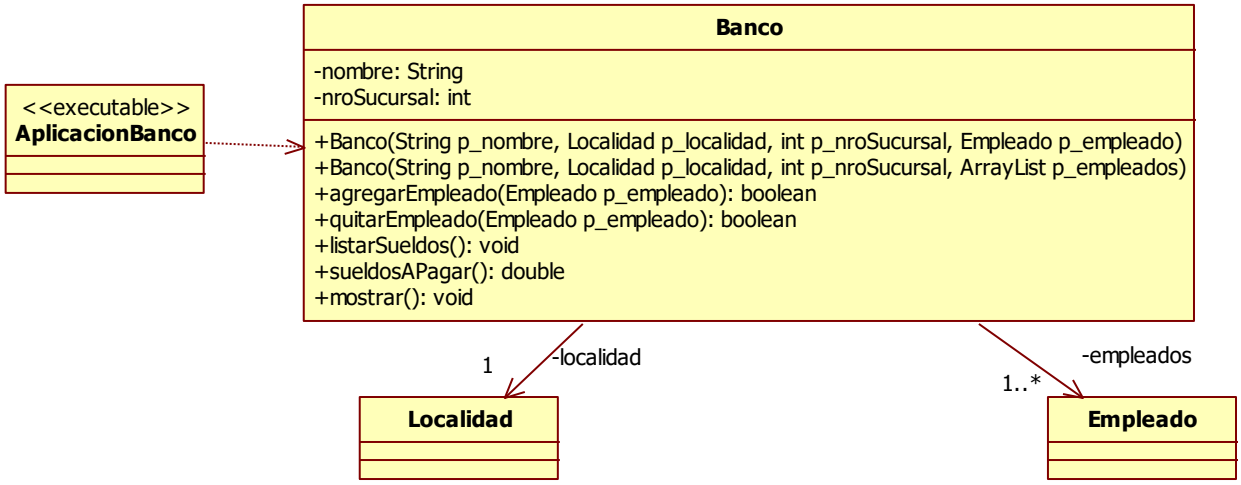
Se debe diseñar una solución que permita armar la lista de productos solicitados y luego permita obtener un detalle de los mismos. El listado debe informar precio de lista y de contado, y al final, un total para cada precio, con el siguiente formato:

****** Detalle del pedido ****** Fecha: 14 de agosto de 2023.

Producto	Precio Lista	Precio Contado
Pendrive	10857.00	9870.00
Libro-POO	10450.00	9500.00
Revista-user	5830.00	5300.00
*** Total ------	27137.00	24670.00

Para obtener esta información es importante tener en cuenta que el Total al Contado será el resultado de calcular la suma del precio de contado de cada elemento de la lista. El total Financiado por su parte surge de la suma del precio de lista de cada elemento de la colección de productos solicitados.
Para demostrar que la solución fue planteada de manera correcta, simular un pedido de productos para un determinado cliente. Listar los totales, quitar un producto del pedido, volver a listar los totales y emitir el detalle del pedido.

3. Una entidad bancaria modeló una aplicación para liquidar los sueldos de sus empleados, según se observa en el diagrama de clases. El banco tiene la restricción de tener **al menos 1** empleado al momento de crearse **(1..*).**



El diseño de la solución debe prever que el banco pueda listar los sueldos de cada uno de sus empleados (CUIL, apellido y nombre, y sueldo neto), e informar el monto total a pagar en concepto de sueldos, así como los datos que identifican a la entidad (nombre, número de sucursal y localidad). La nómina de empleados debe mostrar todos los datos con el siguiente diseño:

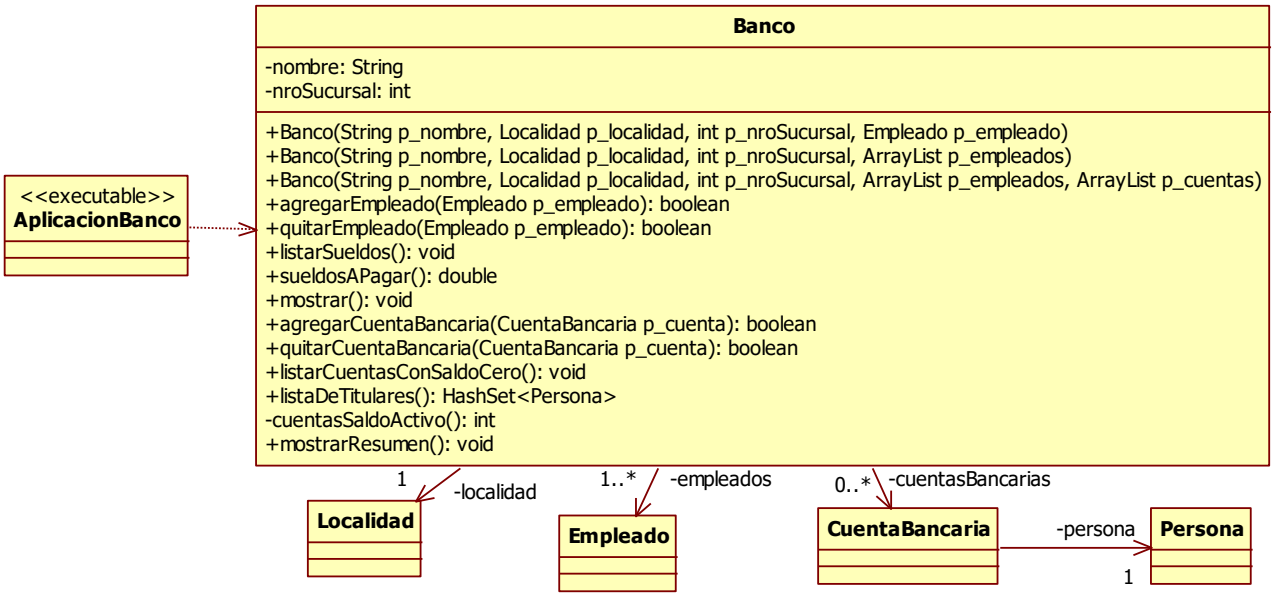
Banco**: RIO** Sucursal**: 3**
Localidad**: Corrientes** Provincia**: Corrientes**

27267504235 Perez, Lorena, ----------$121200.0
20159462638 Dominguez, Pedro ----------$265000.4

Total a Pagar----------$386200.4

El Banco tiene también la responsabilidad de contratar y despedir empleados.
En la clase AplicacionBanco crear los objetos necesarios para realizar una prueba de la implementación.

4. La misma entidad bancaria desea agregar funcionalidades a su aplicación. Como sabemos, el banco realiza variadas operaciones, por lo que no es necesario que posea una cuenta al momento de crearse, pero sin embargo la cuenta bancaria sí debe tener un titular al momento de crearse.
   El banco cuenta un listado de todos los titulares de cuentas (tener presente que, si una persona tiene más de una cuenta, solo debe aparecer una vez en el listado). También puede listar las cuentas cuyo saldo sea cero. Además, debe conocer cuántas cuentas están activas (una cuenta activa es aquella cuyo saldo sea mayor a 0).



Es responsabilidad del banco emitir un resumen, que tendrá el siguiente formato:

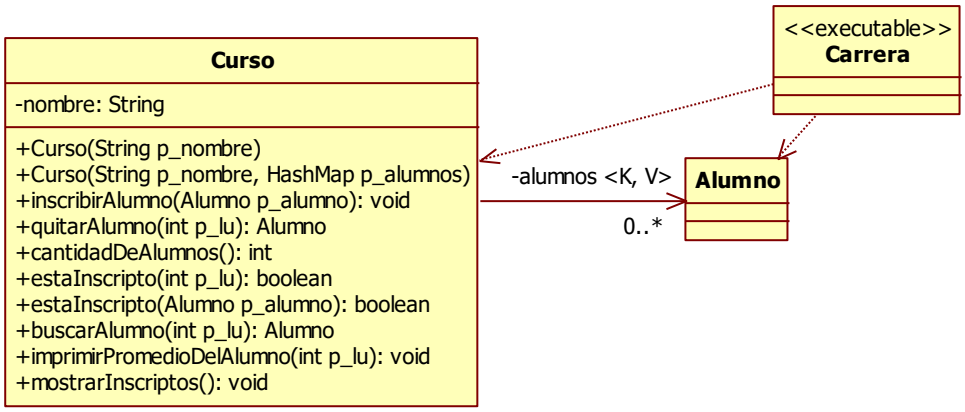
Banco: Rio - **Sucursal**: 3
Localidad: Saladas **Provincia**: Corrientes
****
RESUMEN DE CUENTAS BANCARIAS
****
Número total de Cuentas Bancarias: 2510
Cuentas Activas: 2352
Cuentas Saldo Cero: 158.

Cuentas sin saldo:
--- Cuenta ---------- Apellido y Nombre ----------
14526387 Gomez, Marisa Esther
23145698 Villalba, Martín

Listado de Clientes: Gomez, Marisa Esther; Villalba, Martín; Zalazar, Ernesto
---

Nota: En la clase AplicacionBanco agregar las instrucciones necesarias a fin de probar las nuevas funcionalidades.

5. Una institución educativa desea administrar información de los cursos que dicta. El siguiente diagrama de clases modela el diseño de la situación



La institución registra los alumnos que se inscriben al curso en una colección, por lo tanto, debe proveer una interface que contemple el inscribir y el eliminar un alumno del curso. Tenga en cuenta que en este caso la colección presenta la característica de que cada alumno (valor) de la misma puede ser recuperado mediante el número de LU (clave), por tal motivo se sugiere usar una clase HashMap (implementa la interface Map, reúne el comportamiento de un dictionary, trabajando con pares de clave-valor).
Entre las tareas que debe realizar la administración del Curso están las siguientes: conocer la cantidad de alumnos inscriptos a un curso, indicar si un alumno figura o no inscripto al curso (ya sea si se conoce al objeto alumno, o sólo su LU), obtener un alumno de la colección proporcionando su LU, mostrar sólo el promedio de un alumno en particular, del cual se conoce su LU, y listar todos los alumnos inscriptos al curso, indicando su LU y nombre y apellido.
5.1. En la clase ejecutable se debe:

5.1.1. Crear una instancia de Curso y varias de la clase Alumno
5.1.2. Asignarles notas de parciales a los alumnos
5.1.3. Inscribir los alumnos al curso creado anteriormente.
5.1.4. Imprimir la cantidad y la lista de alumnos inscriptos al curso
5.1.5. Dar de baja un alumno del curso, y luego verificar que no esté inscripto
5.1.6. Imprimir nuevamente la lista de alumnos para ver como que queda definitivamente y la cantidad total de alumnos inscriptos en el curso
5.1.7. Buscar un alumno por su libreta. Una vez encontrado, mostrarlo con el método apropiado.
5.1.8. Mostrar el promedio del alumno solicitado, según libreta

Al ejecutar la aplicación (clase Carrera), deberá obtener la siguiente salida:

****-- Cantidad de inscriptos: 4
32555 Pedro Gomez
23564 Maria Vasquez
30123 Juan Perez
32655 Marcela Martinez

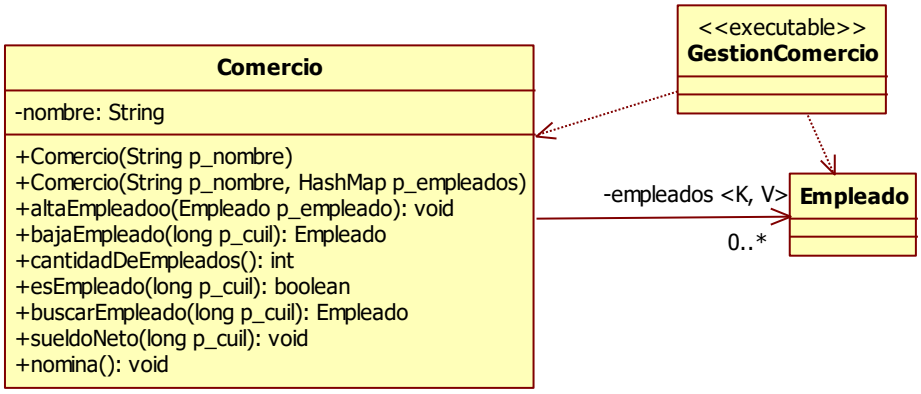
****-- Se da de baja a Pedro porque abandona el curso --****
Está Pedro Gomez inscripto ?? --> false

****-- Alumnos inscriptos actualmente: 3
23564 Maria Vasquez
30123 Juan Perez
32655 Marcela Martinez

****-- Busca y muestra el alumno con numero de libreta 30123 --****
Apellido y nombre: Juan Perez
LU: 30123 Notas: 7.0, 9.0
Promedio: 8.0 Aprobado

****-- Mostrar promedio del alumno 23564 --****
Promedio: 5.5

6 Un Comercio desea automatizar el registro de sus empleados (colección empleados), y para ello debe contar con una interface que contemple el alta y la baja de los mismos. Tenga en cuenta que en este caso la colección presenta la característica de que cada empleado (valor) puede ser recuperado mediante el número de CUIL (clave), por lo que resulta conveniente usar una clase HashMap. Para gestionar los RRHH, es necesario contar con métodos que permitan conocer la cantidad de empleados, consultar por medio del CUIL si un empleado es parte de la empresa, buscar un empleado determinado, conociendo su CUIL, visualizar por pantalla el sueldo neto del empleado cuyo CUIL coincida con el parámetro introducido, y por último, emitir una nómina de los empleados, que debe ser presentado ante la AFIP, con el siguiente formato:



**** Nomina de empleados de Avanti SRL ****
30100623 Gonzalez, Juan---------- $102750.00
37045987 Martinez, Mercedes------ $100719.00
32550096 Gomez, Virginia---------- $150120.00

Implementar una clase ejecutable GestionComercio, donde se instancie un comercio y varios empleados, que serán contratados por el comercio. Emitir una nómina, y verificar el correcto funcionamiento de las demás funcionalidades implementadas.