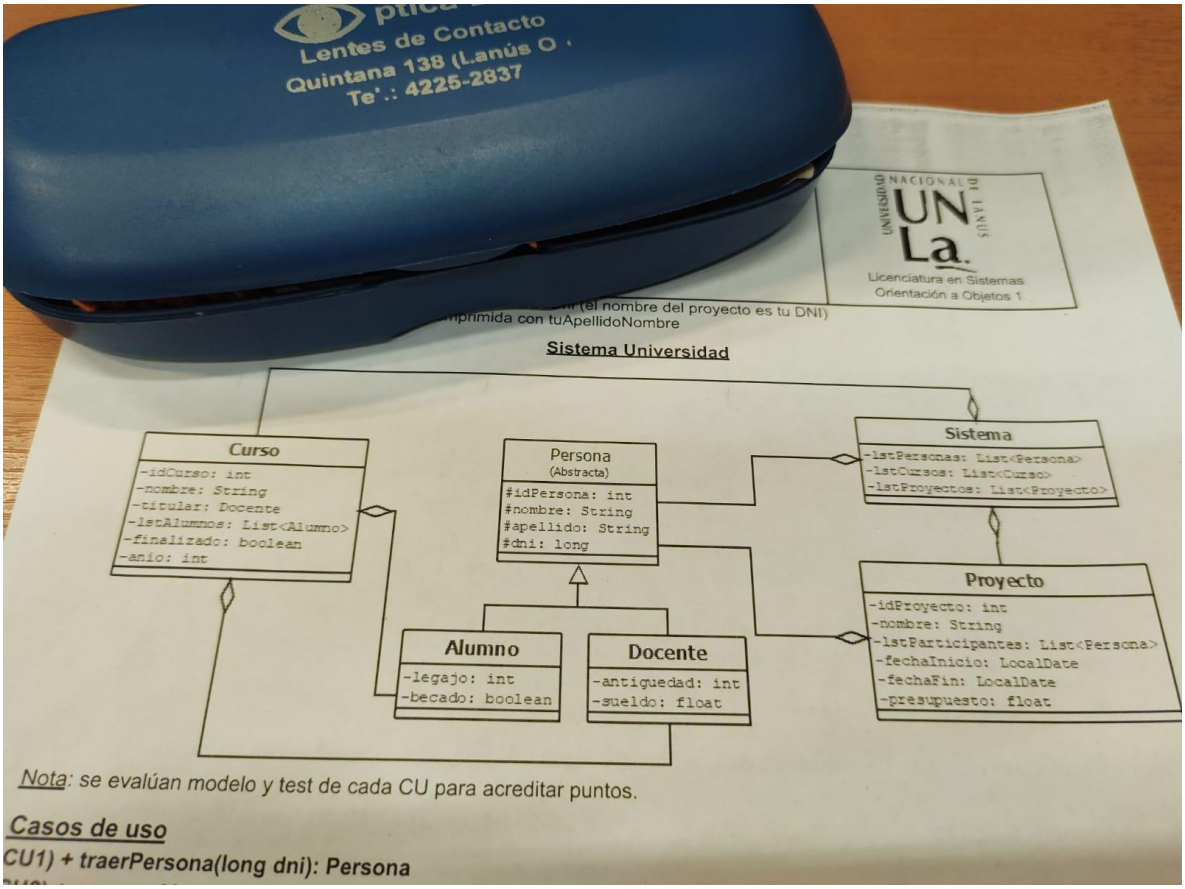
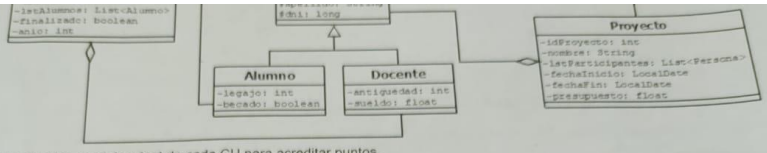

Lentes de Contacto
Quintana 138 (Lanús O
Te'.: 4225-2837
UNIVERSIDAD NACIONAL DE LANÚS
UNLa
Licenciatura en Sistemas
Orientación a Objetos 1
(el nombre del proyecto es tu DNI)
mprimida con tuApellidoNombre
Sistema Universidad
Curso
-idCurso: int
-nombre: String
-titular: Docente
-lstAlumnos: List<Alumno>
-finalizado: boolean
-anio: int
Persona
(Abstracta)
#idPersona: int
#nombre: String
#apellido: String
#dni: long
Sistema
-lstPersonas: List<Persona>
-lstCursos: List<Curso>
-lstProyectos: List<Proyecto>
Proyecto
-idProyecto: int
-nombre: String
-lstParticipantes: List<Persona>
-fechaInicio: LocalDate
-fechaFin: LocalDate
-presupuesto: float
Alumno
-legajo: int
-becado: boolean
Docente
-antiguedad: int
-sueldo: float
Nota: se evalúan modelo y test de cada CU para acreditar puntos.
Casos de uso
CU1) + traerPersona(long dni): Persona



Nota: se evalúan modelo y test de cada CU para acreditar puntos.

Casos de uso

- **CU1) + traerPersona(long dni): Persona**
- **CU2) + agregarAlumno(String nombre, String apellido, long dni, int legajo, boolean becado): boolean**
  // No dispara excepciones. Calcular el id de forma incremental.
- **CU3) + agregarDocente(String nombre, String apellido, long dni, int antiguedad, float sueldo): boolean**
  // No dispara excepciones. Calcular el id de forma incremental.
- **CU4) + traerCurso(String nombre, int anio): Curso**
- **CU5) + agregarCurso(String nombre, Docente titular, boolean finalizado, int anio): boolean**
  // Este método lanza una excepción si ya existe un curso con el mismo nombre y año. Calcular el id de forma incremental. La lista de alumnos se debe inicializar como un ArrayList de Alumnos vacío.
- **CU6) + matricularAlumno(Alumno alumno) : boolean**
  //Agrega un Alumno a un Curso.
- **CU7) + traerProyecto(String nombre): Proyecto**
- **CU8) + agregarProyecto(String nombre, LocalDate fechaInicio, LocalDate fechaFin, float presupuesto): boolean**
  // No dispara excepciones. Calcular el id de forma incremental. La lista de participantes se debe inicializar como un ArrayList de Personas vacío.
- **CU9) + incorporarParticipante(Persona participante): boolean**
  // Agrega una persona, ya sea Alumno o Docente a un proyecto.
- **CU10) - esPresupuestoValido(float presupuesto): boolean**
  // Valida si el valor ingresado como presupuesto es mayor a 0. En caso de ser menor o igual a 0 devuelve "false", lo que disparará el mecanismo de excepción.
- **CU11) ~ calcularIngresos(): float *(Método abstracto)***
  // Calcula los ingresos de una persona: Si es Alumno sus ingresos serán 25.000 si es becado, de lo contrario 0. Si es Docente, los ingresos son el sueldo*antiguedad*0.065.
- **CU12) + sumaIngresosPorProyecto(String nombreProyecto): float**
  // Devuelve la suma de ingresos de los Alumnos y Docentes que participan en un proyecto.
- **CU13) + traerProyectosPorFechaYParticipantes(LocalDate fechaInicioDesde, LocalDate fechaInicioHasta, int cantMinParticipantes): List<Proyecto>**
  // Devuelve una lista de Proyectos que hayan iniciado entre las fechas pasadas como parámetro y que tengan una cantidad de participantes mayor o igual al parámetro cantMinParticipantes. => JUNTARLOS CREADOS.
- **CU14) + cursosPorAlumno(Alumno alumno): List<Curso>**
  // Devuelve una lista de Cursos que está cursando el alumno. Los cursos no deben estar finalizados.

pág

Nota: Al comenzar cada test indicar el número a resolver ej: System.out.println("1"); y luego la implementación del mismo.

Test.java

1) Agregar las siguientes Personas e imprimir la lista de Personas:

- Persona [nombre=Alumno, apellido=Uno, dni=11111111]Alumno [legajo=12345, becado=true]
- Persona [nombre=Alumno, apellido=Dos, dni=22222222]Alumno [legajo=12346, becado=false]
- Persona [nombre=Alumno, apellido=Tres, dni=33333333]Alumno [legajo=12347, becado=true]
- Persona [nombre=Alumno, apellido=Cuatro, dni=44444444]Alumno [legajo=12348, becado=false]
- Persona [nombre=Docente, apellido=Uno, dni=55555555]Docente [antiguedad=2, sueldo=150000]
- Persona [nombre=Docente, apellido=Dos, dni=66666666]Docente [antiguedad=6, sueldo=200000]
- Persona [nombre=Docente, apellido=Tres, dni=77777777]Docente [antiguedad=8, sueldo=300000]

2) Agregar e imprimir los siguientes Cursos:

- Curso [nombre=EPyA, titular=Docente Uno, finalizado=true, anio=2022]
  - **Matricular a**: Alumno Uno, Alumno Dos, Alumno Tres y Alumno Cuatro
- Curso [nombre=IntoBD, titular=Docente Dos, finalizado=false, anio=2023]
  - **Matricular a**: Alumno Uno y Alumno Dos
- Curso [nombre=AyED, titular=Docente Tres, finalizado=false, anio=2023]
  - **Matricular a**: Alumno Uno y Alumno Dos

3) Agregar e imprimir los siguientes Proyectos:

- Proyecto [nombre=Riego Automatizado, fechaInicio=2022-02-01, fechaFin=2022-12-30, presupuesto=1000000]
  - **Incorporar participantes**: Docente Uno y Alumno Uno.
- Proyecto [nombre=Optimización de aulas, fechaInicio=2023-02-01, fechaFin=2023-12-30, presupuesto=2000000.0]
  - **Incorporar participantes**: Docente Dos, Alumno Dos y Alumno Tres.

4) Traer e imprimir la suma de los ingresos de los participantes del proyecto Optimización de aulas.

5) Traer e imprimir todos los Proyectos iniciados entre el 2022-02-01 y el 2023-12-30 con una cantidad de participantes mayor o igual a 3.

6) Traer e imprimir todos los Cursos del alumno "Alumno2"

7) Intentar agregar nuevamente el siguiente Curso:

- Curso [nombre=EPyA, titular=Docente Uno, finalizado=true, anio=2022]

8) Intentar agregar el siguiente Proyecto:

- Proyecto [nombre=Mapa de Calor para Museos, fechaInicio=2023-07-01, fechaFin=2024-07-30, presupuesto=0]