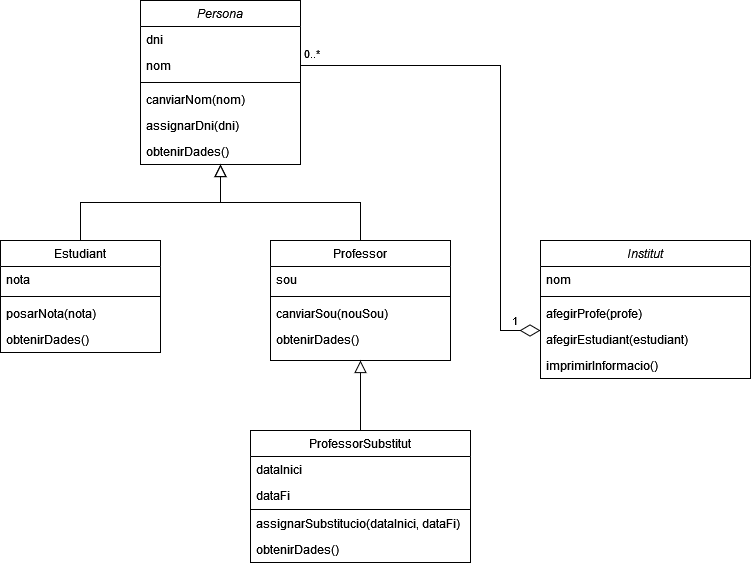
**Implementeu, en un projecte Java, el sistema d’informació corresponent següent diagrama UML**

****

**Indicacions:**

1. S’ha de fer un *main* que provi tota la implementació: Crear un Institut, afegir estudiants i professors, canviar les seves dades… i que imprimeixi la informació de tot l’institut de forma coherent (com el de l’exemple però més llarg).
2. El mètode *obtenirDades* ha de ser polimòrfic i ha de retornar un String que contingui la següent informació:
   1. Per la classe persona, el nom i el DNI.
   2. En el cas dels estudiants, el mateix que *Persona* més la nota de l’estudiant.
   3. En el cas dels profes, el mateix que *Persona* més el sou.
   4. En el cas dels substituts, el mateix que els profes normals, més el dia d’inici i fi de la substitució.
3. Heu d’implementar les següents excepcions:
   1. No té sentit que una persona es pugui canviar el DNI, així que el mètode *assignarDni* de la classe *Persona* ha de llençar una excepció en cas que aquella persona ja tingui DNI.
   2. Els profes són uns pringats, així que no poden cobrar molt. Si intentem pujar-lo a més de 3000, ha de saltar una excepció. També saltarà una excepció si intentem posar sous negatius.
   3. La nota d’un estudiant ha d’estar entre 0 i 10. Si intentem posar una nota no vàlida, ha de saltar una excepció que ho indiqui.
   4. Si intentem obtenir les dades d’una persona sense DNI o sense nom, una excepció ha d’indicar aquest error.
4. Implementar la classe *ProfessorSubstitut* és voluntari (puja nota)

**Exemple de Main.java**

**public class** Main {

**public static void** main(String[] args) **throws** Exception {

Institut ins = **new** Institut(**"La Guineueta"**);

Professor profe1 = **new** Professor(**"Marc"**);

profe1.assignarDni(**"1234"**);

ins.afegirProfe(profe1);

Estudiant estudiant1 = **new** Estudiant();

estudiant1.canviarNom(**"Joan"**);

estudiant1.assignarDni(**"6666"**);

estudiant1.posarNota(7.5);

ins.afegirEstudiant(estudiant1);

ins.imprimirInformacio();

}

}

**Sortida esperada:**

Institut La Guineueta. Llista de persones:

Persona amb nom: Marc i DNI: 1234 (PROFE amb sou: 2500.0)

Persona amb nom: Joan i DNI: 6666 (ESTUDIANT amb nota: 7.5)

Process finished with exit code 0

**Pràctica JAVA. Segona part (ENUNCIAT EN CONSTRUCCIÓ)**

L’objectiu d’aquesta segona part és aconseguir el següent:

* Implementar i provar el funcionament de les excepcions de l’apartat 3. Afegiu al main crides a funcions que sabeu que faran saltar cadascuna de les excepcions.
* La classe Institut, tal com està plantejada a l’UML i com l’heu implementat en Java, té una part del codi redundant. Trobeu quina és aquesta part i elimineu la redundància.
* No té sentit que els estudiants tinguin una única nota. Canvieu la implementació de la classe *Estudiant* per aconseguir el següent:
  + El mètode *posarNota* no ha de substituir la nota anterior, sinó que s’ha d’afegir a un conjunt de notes que ja tenia l’estudiant.
  + S’ha de poder obtenir la nota màxima, mínima i mitjana de qualsevol instància d’estudiant.
* El main hauria de tenir un comportament programàtic, de manera que el programa permeti a l’usuari provar totes les funcionalitats. A continuació un exemple de la consola de l’aplicació en execució (el text en negre és la sortida, i el vermell són les entrades de text que va introduint l’usuari):

Introdueix el número de l’acció que vulguis fer:

1) Veure la llista d’Instituts

2) Veure la informació completa d’un Institut

3) Editar un Institut

4) Eliminar un Institut

5) Crear un Institut

**1**

Encara no hi ha cap Institut

**2**

Encara no hi ha cap Institut

**3**

Encara no hi ha cap Institut

**4**

Encara no hi ha cap Institut

**5**

Escriu el nom de l’Institut que vols crear:

La Guineueta

Institut creat correctament

**1**

1 - La Guineueta

**5**

Escriu el nom de l’Institut:

Jaume Balmes

**1**

1 - La Guineueta

2 - Jaume Balmes

**2**

Escriu el número de l’Institut que vols consultar

**to be continued...**

Process finished with exit code 0