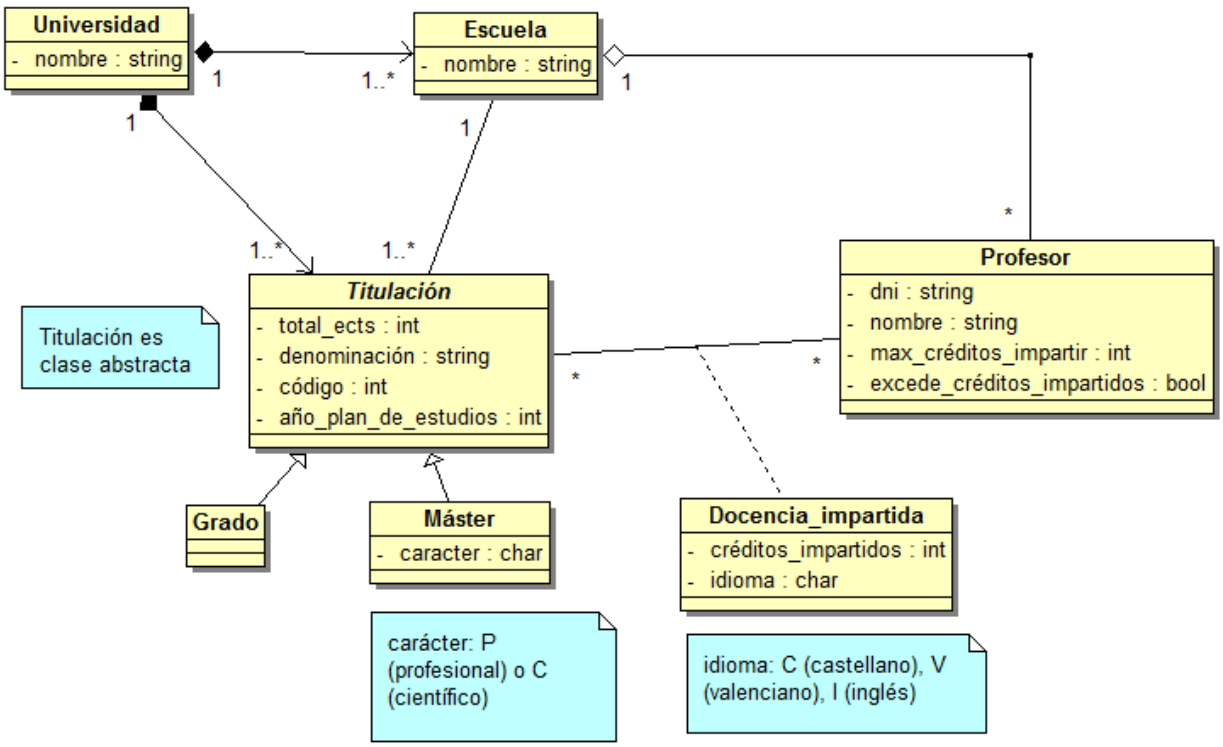
2-dic-2015

ISW: Ejercicio Evaluable – Grupo 3D

Se va a desarrollar una aplicación para gestionar la carga docente del profesorado de la universidad, medida en créditos impartidos. Dado el siguiente diagrama de clases que ha realizado el equipo de desarrollo, se pide:



a. *(3 puntos)* Obtener el diseño en Java de las clases del negocio, siguiendo las técnicas de diseño vistas en clase. Deben declararse los atributos que se deducen del modelo e implementarse los constructores (no es necesario indicar el resto de métodos en las clases diseñadas).

```
public class Universidad {
private String nombre;
private ArrayList<Escuela> listaEscuelas;
private ArrayList<Titulacion> listaTitulaciones;

public Universidad(String nombre) {
  this.nombre = nombre;
  listaEscuelas = new ArrayList<Escuela>();
}
…
}

public class Escuela {
private String nombre;
private ArrayList <Profesor> profAdscritos;
private ArrayList <Titulacion> titulacionesImpartidas;

public Escuela(String nombre) {
  this.nombre = nombre;
  profAdscritos=new ArrayList <Profesor>();
  titulacionesImpartidas=new ArrayList <Titulacion>();
}
…
}
```

```
public abstract class Titulacion {
private int codigo;
private int total_ects;
private String denominacion;
private int plan_de_estudios;
private Escuela escuela;
private ArrayList<DocenciaImpartida> listaDocenciaImpartida;

public Titulacion(int total_ects, String denominacion,int codigo,
       int plan_de_estudios, Escuela escuela) {
  this.codigo = codigo;
  this.total_ects = total_ects;
  this.denominacion = denominacion;
  this.plan_de_estudios = plan_de_estudios;
  this.escuela = escuela;
  listaDocenciaImpartida=new ArrayList<DocenciaImpartida>();
}
…
}

public class Grado extends Titulacion {

  public Grado(int total_ects, String denominacion, int codigo,
              int plan_de_estudios, Escuela escuela) {
       super(total_ects, denominacion, codigo, plan_de_estudios,
escuela);
  }
}

public class Master extends Titulacion {
private char caracter; //P:profesional, C:científico

public Master(int total_ects, String denominacion, int codigo,
       int plan_de_estudios, Escuela escuela, char caracter) {
  super(total_ects, denominacion, codigo, plan_de_estudios, escuela);
  this.caracter = caracter;
}
…
}

public class Profesor {
private String dni;
private String nombre;
private int max_creditos_impartir;
private boolean excede_creditos_impartidos;
private Escuela escuela;
private ArrayList<DocenciaImpartida> listaDocenciaImpartida;

public Profesor(String dni, String nombre, int max_creditos,
       Escuela escuela) {
  this.dni = dni;
  this.nombre = nombre;
  this.max_creditos_impartir = max_creditos;
  this.excede_creditos_impartidos = false;
  this.escuela = escuela;
  listaDocenciaImpartida=new ArrayList<DocenciaImpartida>();
}
…
}
```

```
public class DocenciaImpartida {
private Profesor profesor;
private Titulacion titulacion;
int creditos_impartidos;
char idioma; // C:castellano; V:valenciano; I: inglés
public DocenciaImpartida(Profesor profesor, Titulacion titulacion, int
creditos_impartidos, char idioma) {
        this.profesor = profesor;
        this.titulacion = titulacion;
        this.creditos_impartidos = creditos_impartidos;
        this.idioma = idioma;
  }
…
}
```

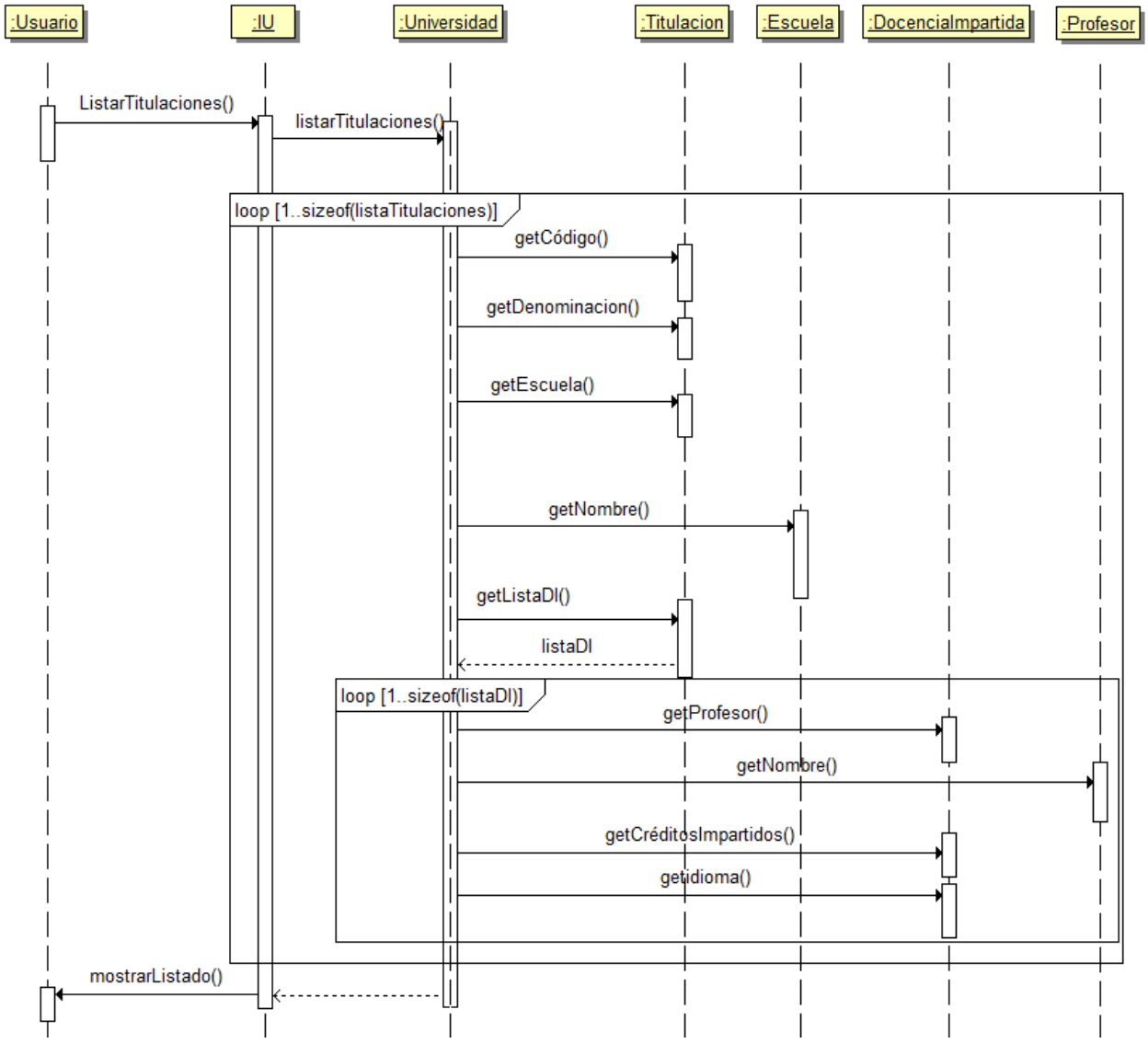
b. *(2 puntos)* Escribe el código en Java para invocar los constructores que consideres necesarios, de forma que el sistema quede inicializado en un estado correcto y consistente (debes crear al menos una instancia de cada clase). Puedes utilizar los valores que desees.

```
public class inicializar {

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


  //Relación Universidad (1) - (1..N) Escuela es Unidireccional, Escuela
no mantiene una referencia a Universidad
  Universidad u=new Universidad("UPV");
  Escuela e=new Escuela("ETSINF"); // La Escuela (e) se inicializa con la
instancia de Universidad (u)
  u.addEscuela(e); //Se añade la Escuela (e) a la lista de Escuelas de la
Universidad (u)

  //Relación Escuela (1) - (*) Profesor Bidireccional
  Profesor p=new Profesor("00000000Z","Antonio Molina",32,e); //El
Profesor (p) se inicializa con la instancia de Escuela (e)
  e.addProfesor(p); //Se añade el Profesor (p) a la lista de Profesores
adscritos a la Escuela (e)

  //Relacion Escuela (1) - (1..N) Titulacion Bidireccional:
inicialización en dos pasos. La Escuela se ha inicializado arriba
  //Titulacion es clase abstracta, se crean instancias de las clases
derivadas Grado y Master
  Titulacion grado=new Grado(240,"Grado en Ingeniería
Informática",156,2009,e); //La Titulación (grado) se inicializa con la
instancia de Escuela (e)
  Titulacion master=new Master(120,"Master en Ingeniería
Informática",2233,2014,e,'P'); //La Titulación (master) se inicializa
con la instancia de Escuela (e)
  e.addTitulacion(grado); //Se añade la Titulacion (grado) a la lista de
Titulaciones impartidas por la Escuela (e)
  e.addTitulacion(master); //Se añade la Titulacion (master) a la lista
de Titulaciones impartidas por la Escuela (e)

  // Relación Universidad (1) - (1..N) Titulación es Unidireccional,
Titulación no mantiene una referencia a Universidad
  u.addTitulacion(grado);
  u.addTitulacion(master);

  //Clase Asociación DocenciaImpartida
  DocenciaImpartida di_grado = new DocenciaImpartida(p,grado,15,'C');
  DocenciaImpartida di_master = new DocenciaImpartida(p,master,6,'C');
```

```
  //Para mantener la bidireccionalidad con Titulacion y Profesor:
  grado.addDI(di_grado); //Se añade la DocenciaImpartida (di_grado) a la
lista de docencia impartida en la Titulacion (grado)
  master.addDI(di_master); //Se añade la DocenciaImpartida (di_master) a
la lista de docencia impartida en la Titulacion (master)
  p.addDI(di_grado); //Se añade la DocenciaImpartida (di_grado) a la
lista de docencia impartida por el Profesor (p)
  p.addDI(di_master); //Se añade la DocenciaImpartida (di_master) a la
lista de docencia impartida por el Profesor (p)

  }
}
```

c. *(5 puntos)* Obtener los diagramas de secuencia asociados a los siguientes escenarios. Puede considerarse la clase Universidad como la clase que realiza tareas de Controlador de la capa de negocio.

c1. “Obtener la relación de todas titulaciones que imparte la Universidad, indicando para cada una de ellas el código de la titulación, su denominación, el nombre de la escuela en la que se imparte y el nombre de los profesores que imparten clase en la titulación con los créditos impartidos y el idioma de impartición”.



c2. “Añadir el profesor con dni: ”11111111A” a la titulación de Grado “156” con “15” créditos impartidos en idioma “valenciano”. Se asume que profesor y titulación ya existen. Si el profesor ya imparte docencia en esa titulación el sistema mostrará un mensaje de error. Si todo es correcto el sistema devolverá un mensaje de éxito. En caso de que el número total de créditos impartidos por el profesor supere el máximo de créditos a impartir (max_créditos_impartir) el sistema registrará este hecho e informará al usuario con un mensaje”