OBJETIVO

Este documento contiene material de apoyo para el boletín T11 del trabajo práctico. En concreto, el documento incluye algunos métodos creacionales y propiedades poblacionales de la factoría Grados. El profesor las explicará en la sesión de laboratorio y responderá sus dudas al respecto. El alumno, por su parte, debe añadir el código a su proyecto de curso, y realizar por su cuenta el resto de ejercicios del boletín.

FACTORIA GRADOS

Clase `Grados`

```
public class Grados {

	// ********* Tipo "Asignatura" ***************
	private static Map<String, Asignatura> asignaturasPorCodigo =
		new HashMap<String, Asignatura>();

	public static Integer getNumAsignaturasCreadas() {
		return asignaturasPorCodigo.size();
	}
	public static Set<Asignatura> getAsignaturasCreadas() {
		return new HashSet<Asignatura>(asignaturasPorCodigo.values());
	}
	public static Set<String> getCodigosAsignaturasCreadas() {
		return new HashSet<String>(asignaturasPorCodigo.keySet());
	}
	public static Asignatura getAsignaturaCreada(String codigo) {
		return asignaturasPorCodigo.get(codigo);
	}

	private static void actualizaPoblacionales(Asignatura a){
		asignaturasPorCodigo.put(a.getCodigo(), a);
	}

	public static Asignatura createAsignatura(String nombre, String codigo,
			Double creditos, TipoAsignatura tipo,
			Integer curso, Departamento departamento) {
		Asignatura res = new AsignaturaImpl(nombre, codigo, creditos,
			tipo, curso, departamento);
		actualizaPoblacionales(res);
		return res;
	}

	public static Asignatura createAsignatura(String asignatura) {
		Asignatura res = new AsignaturaImpl(asignatura);
		actualizaPoblacionales(res);
		return res;
	}

	public static List<Asignatura> createAsignaturas(String nombreFichero) {
		List<Asignatura> res = leeFichero(nombreFichero,
			s -> createAsignatura(s));
		return res;
	}
```

Clase `Grados` (cont)

```
	// ********* Tipo "Beca" ***************
	private static Set<Beca> becas = new HashSet<Beca>();
	private static Integer[] numBecasPorTipo = { 0, 0, 0 };

	public static Integer getNumBecasCreadas() {
		return becas.size();
	}

	public static Set<Beca> getBecasCreadas() {
		return new HashSet<Beca>(becas);
	}

	public static Integer getNumBecasTipo(TipoBeca tipo) {
		return numBecasPorTipo[tipo.ordinal()];
	}

	private static void actualizaPoblacionales(Beca b) {
		becas.add(b);
		numBecasPorTipo[b.getTipo().ordinal()]++;
	}

	public static Beca createBeca(String codigo, Double cuantiaTotal,
			Integer duracion, TipoBeca tipo) {
		Beca res = new BecaImpl(codigo, cuantiaTotal, duracion, tipo);
		actualizaPoblacionales(res);
		return res;
	}

	public static Beca createBeca(String codigo, TipoBeca tipo) {
		Beca res = new BecaImpl(codigo, tipo);
		actualizaPoblacionales(res);
		return res;
	}

	public static Beca createBeca(Beca beca) {
		Beca res = new BecaImpl(beca.getCodigo(), beca.getCuantiaTotal(),
			beca.getDuracion(), beca.getTipo());
		actualizaPoblacionales(res);
		return res;
	}

	public static Beca createBeca(String beca) {
		Beca res = new BecaImpl(beca);
		actualizaPoblacionales(res);
		return res;
	}

	public static List<Beca> createBecas(String nombreFichero) {
		List<Beca> res = leeFichero(nombreFichero, s -> createBeca(s));
		return res;
	}
```

Clase `Grados` (cont)

```
	// ********* Tipo "Profesor" ***************
	private static Boolean usarImplementacionMap = true;
	private static Set<Profesor> profesores = new HashSet<Profesor>();

	public static Integer getNumProfesoresCreados() {
		return profesores.size();
	}
	public static Set<Profesor> getProfesoresCreados() {
		return new HashSet<Profesor>(profesores);
	}

	public static void setUsarImplementacionMapProfesor(Boolean b) {
		usarImplementacionMap = b;
	}

	private static void actualizaPoblacionales(Profesor p){
		profesores.add(p);
	}

	public static Profesor createProfesor(String dni, String nombre,
			String apellidos, LocalDate fechaNacimiento, String email,
			Categoria categoria, Departamento departamento) {
		Profesor res = null;
		if (usarImplementacionMap) {
			res = new ProfesorImpl2(dni, nombre, apellidos,
				fechaNacimiento, email, categoria, departamento);
		} else {
			res = new ProfesorImpl(dni, nombre, apellidos,
				fechaNacimiento, email, categoria, departamento);
		}
		actualizaPoblacionales(res);
		return res;
	}

	public static Profesor createProfesor(Profesor profesor) {
		Profesor res = createProfesor(profesor.getDNI(),
			profesor.getNombre(), profesor.getApellidos(),
			profesor.getFechaNacimiento(), profesor.getEmail(),
			profesor.getCategoria(), profesor.getDepartamento());
		copiaAsignaturasProfesor(res, profesor);
		copiaTutoriasProfesor(res, profesor);
		return res;
	}

	private static void copiaAsignaturasProfesor(Profesor res,
			Profesor profesor) {
		for (Asignatura a : profesor.getAsignaturas()) {
			res.imparteAsignatura(a, profesor.dedicacionAsignatura(a));
		}
	}

	private static void copiaTutoriasProfesor(Profesor res,
			Profesor profesor) {
		for (Tutoria t : profesor.getTutorias()) {
			res.nuevaTutoria(t.getHoraComienzo(),
				t.getDuracion(), t.getDiaSemana());
		}
	}
```

Clase `Grados` (cont)

```
	// ********* Tipo "Departamento" ***************
	private static Set<Departamento> departamentos =
		new HashSet<Departamento>();

	public static Integer getNumDepartamentosCreados() {
		return departamentos.size();
	}
	public static Set<Departamento> getDepartamentosCreados() {
		return new HashSet<Departamento>(departamentos);
	}

	private static void actualizaPoblacionales(Departamento d){
		departamentos.add(d);
	}

	public static Departamento createDepartamento(String nombre) {
		Departamento res = new DepartamentoImpl(nombre);
		actualizaPoblacionales(res);
		return res;
	}


	// otros métodos públicos de utilidad
	public static <T> List<T> leeFichero(String nombreFichero,
			Function<String, T> funcion_deString_aT) {
		List<T> res = null;
		try {
			res = Files.lines(Paths.get(nombreFichero)).map(
				funcion_deString_aT).collect(Collectors.toList());
		} catch (IOException e) {
			System.out.println(
				"Error en lectura del fichero: " + nombreFichero);
		}
		return res;
	}
```