OBJETIVOS

- Diseño de tipos: creación de clases. Atributos, constructores, métodos get, set y `toString()`.
- Tests de creación de objetos.

CREACIÓN DE LAS CLASES

Cree las clases que implementan cada una de las interfaces definidas en la fase anterior y defina en cada clase los métodos constructores y los métodos get y set que procedan para cada una de sus propiedades; para ello tenga en cuenta si la propiedad es básica o derivada, y si es consultable y/o modificable. Añada también un método `toString()` teniendo en cuenta la representación como cadena de cada entidad. Para ello utilice las descripciones que se proporcionan.

Las **asignaturas** serán representadas textualmente en la aplicación mediante el código de la misma entre paréntesis, seguido del nombre, por ejemplo, "(0000230) Fundamentos de Programación". Se requiere que la clase que implemente al tipo contenga un constructor que permita crear objetos para representar asignaturas a partir de todas las propiedades básicas que las definen. Nota: aún no sabemos cómo calcular el acrónimo de la asignatura, así que de momento haga que el método devuelva `null` y coloque un *TODO* para completarlo más adelante.

Con respecto a las **becas**, la aplicación las representará textualmente mediante el código y el tipo, separados por comas y colocados entre corchetes (por ejemplo, "[ABB2024, movilidad]"). Se requiere que la clase que implemente al tipo contenga dos constructores: uno que permita crear becas una vez conocidas todas sus propiedades básicas, y otro que permita crear becas de las que no se conoce aún su cuantía ni su duración. Se supondrán en este caso la cuantía y duración mínimas, que son 1500,0 € y 1 mes, respectivamente (tenga en cuenta que la cuantía mínima podría variar en futuras versiones de la aplicación).

Una **persona** se representará mediante el DNI, seguido de un guión, los apellidos, una coma, el nombre, otro guión y la fecha de nacimiento. Por ejemplo: "28864657W – García Vaquero, Pascual – 15/09/1998". Se requiere que la clase que implemente al tipo contenga dos constructores, uno que permita crear objetos para representar personas a partir de todas sus propiedades básicas, y otro con todas las propiedades básicas excepto el email, que se inicializará con una cadena vacía.

Un **espacio** se representará mediante su nombre seguido de la planta entre paréntesis. Por ejemplo, "A3.10 (planta 3)". La clase que implemente el tipo tendrá un único constructor con todas sus propiedades básicas.

Una **nota** se representará mediante la asignatura, el curso académico (formado por el año de inicio del curso seguido de un guión y de los dos últimos dígitos del año de final del curso), la convocatoria, el valor numérico y la calificación, separados por comas. Por ejemplo, "(0000230) Fundamentos de Programación, 2014-15, PRIMERA, 7.5, NOTABLE". La clase que implemente el tipo tendrá dos constructores, uno que permita crear objetos para representar notas a partir de todas sus propiedades básicas, y otro con todas las propiedades básicas excepto la mención de honor, que se inicializará con un valor falso.

Una **tutoría** se representará por un carácter que indica el día (L, M, X, J o V), seguido de la hora de comienzo, un guión y la hora de fin. Por ejemplo, "X 10:30-12:30". La clase que implemente el tipo tendrá dos constructores, uno con el día de la semana, la hora de comienzo y la hora de fin, y otro con el día, la hora de comienzo y la duración.

REALIZACIÓN DE TESTS DE CREACIÓN DE OBJETOS

Una vez creadas las clases, implemente un test para cada una de ellas. El test debe crear un objeto por cada constructor disponible, y posteriormente debe mostrar en la consola el objeto completo y también cada una de sus propiedades consultables por separado.

En concreto, para realizar el test del tipo T, cree una clase de test con el nombre `TestT`. Esta clase deberá contener los siguientes métodos:

- Un método privado `mostrarT()` que recibirá como parámetro un objeto del tipo T y mostrará en la consola el objeto completo y todas sus propiedades consultables.
- Un método privado por cada constructor definido en el tipo T. Por ejemplo, si el tipo posee dos constructores, escriba dos métodos privados de nombres `testConstructor1()` y `testConstructor2()`. Cada uno de ellos creará un objeto del tipo usando el constructor correspondiente y posteriormente lo mostrará en la consola llamando al método `mostrarT()`.
- Un método `main()` que invocará sucesivamente a cada uno de los métodos definidos en el apartado anterior.