1. Haz una clase llamada **Password** que siga las siguientes condiciones:

* Que tenga los atributos**longitud** y **contraseña**. Por defecto, la longitud sera de 8.
* Los constructores serán los siguiente:
  + Un constructor por defecto.
* Un constructor con la longitud que nosotros le pasemos. Generará una contraseña aleatoria con esa longitud.
* Los métodos que implementa serán:
  + **esFuerte()**: devuelve un booleano si es fuerte o no, para que sea fuerte debe tener más de ***2 mayúsculas, más de 1 minúscula y más de 5 números.***
  + **generarPassword()**:  genera la contraseña del objeto con la longitud que tenga.
  + Método get para contraseña y longitud.
  + Método set para longitud.

Ahora, crea una clase clase ejecutable:

* Crea un array de Passwords con el tamaño que tú le indiques por teclado.
* Crea un bucle que cree un objeto para cada posición del array.
* Indica también por teclado la longitud de los Passwords (antes de bucle).
* Crea otro array de booleanos donde se almacene si el password del array de Password es o no fuerte (usa el bucle anterior).
* Al final, muestra la contraseña y si es o no fuerte (usa el bucle anterior). Usa este simple formato:

contraseña1 valor\_booleano1

contraseña2 valor\_bololeano2

1. Crea una ***clase llamada Libro*** que guarde la información de cada uno de los libros de una biblioteca. La clase debe guardar el ***título del libro, autor, número de ejemplares del libro y número de ejemplares prestados***. La clase contendrá los siguientes métodos:

* Constructor por defecto.
* Constructor con parámetros.
* Métodos Setters/getters
* Método *préstamo* que incremente el atributo correspondiente cada vez que se realice un préstamo del libro. No se podrán prestar libros de los que no queden ejemplares disponibles para prestar. Devuelve true si se ha podido realizar la operación y false en caso contrario.
* Método *devolución* que decremente el atributo correspondiente cuando se produzca la devolución de un libro. No se podrán devolver libros que no se hayan prestado. Devuelve true si se ha podido realizar la operación y false en caso contrario.
* Método *toString* para mostrar los datos de los libros. Este método se heredada de Object y lo debemos modificar (override) para adaptarlo a la clase Libro.
* ***Escribe un programa para probar el funcionamiento de la clase Libro.***

1. Nos piden hacer una un programa que gestione empleados.

Los empleados se definen por tener:

* Nombre
* Edad
* Salario

También tendremos una constante llamada PLUS, que tendrá un valor de 300€

Tenemos dos tipos de empleados: repartidor y comercial.

El comercial, aparte de los atributos anteriores, tiene uno más llamado comisión (double).

El repartidor, aparte de los atributos de empleado, tiene otro llamado zona (String).

Crea sus constructores, getters and setters y toString (piensa como aprovechar la herencia).

No se podrán crear objetos del tipo Empleado (la clase padre) pero si de sus hijas.

Las clases tendrán un método llamado plus, que según en cada clase tendrá una implementación distinta. Este plus básicamente aumenta el salario del empleado.

* En comercial, si tiene más de 30 años y cobra una comisión de más de 200 euros, se le aplicara el plus.
* En repartidor, si tiene menos de 25 y reparte en la “zona 3”, este recibirá el plus.

Puedes hacer que devuelva un booleano o que no devuelva nada, lo dejo a tu elección.

Crea una clase ejecutable donde crees distintos empleados y le apliques el plus para comprobar que funciona.