

TAP 2020-21. Laboratorio.

4. Usando las ayudas de Netbeans para completar una aplicación

Este es el ejercicio de examen de la convocatoria extraordinaria del curso 2018-19. No se parece a los ejercicios anteriores de examen y no fue demasiado bien entendido, aunque es más concreto y dirigido que aquellos, ... veamos cómo no es difícil.

El objetivo de este ejercicio es que el programa de prueba que se proporciona en la siguiente página funcione correctamente, dando una salida razonablemente similar a mostrada a continuación.

```
Dado de alta el abono
  {653455433X, Pedro Pérez Pereda, zona=A, año=1989, Fam. numerosa, cad.=vie, 26 jul 2019}
Cantidad a pagar 150.0
Dado de alta el abono
  {306732876J, Ana Antón Andujar, zona=C, año=2005, Fam. numerosa no, cad.=vie, 26 jul 2019}
Cantidad a pagar 20.0
Dado de alta el abono
  {43745724R, Berta Bereziartua Bershmidt, zona=B, año=1969, Fam. numerosa no, cad.=vie, 26 jul 2019}
Cantidad a pagar 35.0
El abonado 306732876J pasa canceladora en Abando
El abono asociado al DNI 653455433X es rechazado en Plencia por estar fuera de zona
***caducando 653455433X
El abono asociado al DNI 653455433X es rechazado en Casco Viejo por estar caducado
Renovado el abono
  {653455433X, Pedro Pérez Pereda, zona=A, año=1989, Fam. numerosa, cad.=mar, 26 may 2020}
Cantidad a pagar 142.5
El abonado 653455433X pasa canceladora en Casco Viejo
```

NOTA.- El camino más sencillo para resolver el ejercicio consiste en copiar el programa de prueba de la página siguiente y pegarlo en NetBeans para ir aceptando las sugerencias que este nos dará para crear clases y métodos. Se sugiere además comenzar desactivando el código con marcas de comentario de línea para ir activándolo parcialmente y resolviendo lo necesario para que se ejecute correctamente la parte activa.

Enunciado:

Se quiere implementar la gestión de abonos de metro de tipo mensual y anual. Cada abono estará asociado a una zona tarifaria (A, B -que incluye a A- o C -que incluye a las anteriores-), y a una persona, de la que se dispondrá de varios datos tanto para su personalización como para la aplicación de diferentes tarifas (nombre, apellidos, DNI, año de nacimiento, familia numerosa (si/no)).

El sistema debe permitir gestionar un solo abono por persona, que podrá ser mensual o anual. La primera vez que la persona solicita un abono, se dará de alta y será válido desde ese mismo instante hasta una vez pasado el periodo correspondiente. Un abono caducado seguirá existiendo y podrá ser renovado en cualquier momento.

El mismo sistema estará conectado a la gestión de entradas y salidas del metro de modo que cada vez que se use un abono se compruebe si está en la zona correspondiente y no está caducado para aceptarlo.

El cálculo de la tarifa aplicable de un abono se hará en un método específico dentro de una clase ad-hoc.

- Las tarifas de referencia se establecen para un mes y dependen de las zonas: A= 30€/mes, B= 35€/mes, C=40€/mes.
- Las tarifas anuales equivaldrán al coste de 10 meses.
- Los menores de 18 años y las familias numerosas tendrán un descuento del 50% (no acumulables)
- Los abonos anuales acumulan una promoción con cada renovación que consiste en un 5% de descuento, con un máximo de 5 renovaciones (a partir de ahí queda fijado en el 25%). Este descuento se aplica después del indicado en el punto anterior si procede.
(i.e. 1ª renovación: -5%, 2ª: -10%, 3ª: -15%, 4ª: -20%, 5ª y siguientes: -25%)

Sugerencias:

Las fechas (caducidad de los abonos, cálculo de mayoría de edad) son fáciles de manejar con la clase `GregorianCalendar`

El formato de fecha mostrado en el cuadro superior se obtiene con la clase `SimpleDateFormat` y se corresponde con la especificación `"EEE, d MMM yyyy"`

```
class Precios{
    //...
    static double getPrecio(Abono a) {
        //...
    }
}
```

```

public class MetroBilbao {

    private static final TubeManager MANAGER=TubeManager.getTheManager(); //El gestor es un singleton

    //TODO añadir todas las estaciones (no es parte del ejercicio)
    private static final String[] ESTACIONES_A = {"Casco Viejo", "Abando"},
        ESTACIONES_B = {"Erandio", "Axpe"},
        ESTACIONES_C = {"Plencia", "Sopelana"};

    static { //Inicializacion del sistema: estaciones y zonas
        for (String s : ESTACIONES_A) MANAGER.addEstacion(s, Zona.A);
        for (String s : ESTACIONES_B) MANAGER.addEstacion(s, Zona.B);
        for (String s : ESTACIONES_C) MANAGER.addEstacion(s, Zona.C);
    }

    // PRUEBAS
    public static void main(String[] args) {
        pruebaAlta(true, Zona.A, "Pedro", "Pérez Pereda", "653455433X", 1989, true);
        pruebaAlta(false, Zona.C, "Ana", "Antón Andujar", "306732876J", 2005, false);
        pruebaAlta(false, Zona.B, "Berta", "Bereziartua Bershmidt", "43745724R", 1969, false);

        pruebaUso("306732876J", "Abando");
        pruebaUso("653455433X", "Plencia");

        //Prueba de uso de abono caducado. ("caducar" no iría en la aplicación final)
        MANAGER.caducar("653455433X");
        pruebaUso("653455433X", "Casco Viejo");

        pruebaRenovacion("653455433X");
        pruebaUso("653455433X", "Casco Viejo");
        pruebaRenovacion("306732876J");
    }

    // RUTINAS DE PRUEBAS
    private static void pruebaAlta(boolean anual, Zona zona, String nombre, String apellidos, String dni,
        int año, boolean famNum)
    {
        try {
            Abono a = anual ? new AbonoAnual(zona, nombre, apellidos, dni, año, famNum)
                : new AbonoMensual(zona, nombre, apellidos, dni, año, famNum);
            System.out.println("Dado de alta el abono\n " + a);
            System.out.println(" Cantidad a pagar " + Precios.getPrecio(a));
        } catch (MetroGestionException ex) {
            System.out.println(ex.getMessage());
        }
    }

    private static void pruebaUso(String dni, String estacion) {
        try {
            MANAGER.acceso(dni, estacion);
            System.out.println("El abonado " + dni + " pasa canceladora en " + estacion);
        } catch (MetroGestionException ex) {
            System.out.println(ex.getMessage());
        }
    }

    private static void pruebaRenovacion(String dni) {
        try {
            Abono.renueva(dni);
            System.out.println("Renovado el abono\n " + Abono.getByDni(dni));
            System.out.println(" Cantidad a pagar " + Precios.getPrecio(Abono.getByDni(dni)));
        } catch (MetroGestionException ex) {
            System.out.println(ex.getMessage());
        }
    }
}

```