Primer Parcial de IIP (ETSInf)
31 de Octubre de 2019. Duración: 1 hora y 30 minutos

Nota: El examen se evalúa sobre 10 puntos, pero su peso específico en la nota final de IIP es de **3,75 puntos**

NOMBRE:	GRUPO:

1. 6 puntos Se quiere diseñar una clase Tipo de Datos denominada `PieceOfNews` para representar una noticia que será publicada en un medio digital el mismo día en el que se produce. Cada noticia tiene asociados los siguientes elementos: hora en la que se produce; enlace al fichero que contiene la información a publicar; número de medios que se han hecho eco de ella el mismo día; tipo, que puede ser texto, vídeo o audio.

   Para representar el instante del día en el que se ha producido una noticia, se dispone de la clase de usuario `TimeInstant`, cuya documentación se muestra -parcialmente- a continuación:

Constructors

Constructor	Description
`TimeInstant()`	Crea un `TimeInstant` con el valor del instante actual UTC (tiempo universal coordinado).
`TimeInstant (int iniHours, int iniMinutes)`	Crea un `TimeInstant` con el valor de las horas y los minutos que recibe como argumentos, `iniHours` y `iniMinutes`, respectivamente.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
`int`	`compareTo (TimeInstant tInstant)`	Compara cronológicamente el instante `this` con `tInstant`. El resultado es negativo si `this` es anterior a `tInstant`, cero si son iguales, y positivo si `this` es posterior a `tInstant`.
`boolean`	`equals (java.lang.Object o)`	Devuelve `true` sii `o` es un objeto de la clase `TimeInstant` y sus horas y minutos coinciden con los del objeto en curso.
`int`	`getHours()`	Devuelve las horas del `TimeInstant`.
`int`	`getMinutes()`	Devuelve los minutos del `TimeInstant`.
`void`	`setHours(int hh)`	Actualiza las horas del `TimeInstant`.
`void`	`setMinutes(int mm)`	Actualiza los minutos del `TimeInstant`.
`java.lang.String`	`toString()`	Devuelve el `TimeInstant` en el formato "hh:mm".

Se pide: implementar la clase `PieceOfNews`, (se supone que en el mismo paquete que la clase `TimeInstant`) con los siguientes atributos y métodos:

a) (0.5 puntos) Tres atributos estáticos, públicos y constantes de tipo `int`, para representar mediante un código numérico los tres tipos de formatos que puede tener el fichero que contiene una noticia: audio, vídeo o texto. Sus identificadores y valores son, respectivamente: `AUDIO`, con valor `0`; `VIDEO`, con valor `1`; `TEXT`, con valor `2`.
   Estas "constantes Java" se deben usar siempre que se requiera en la clase `PieceOfNews`.

b) (0.5 puntos) Cuatro atributos de instancia y privados, para representar los elementos asociados a una `PieceOfNews`. Siguiendo el orden en el que se han descrito previamente, sus identificadores (y tipos Java) son: `instant` (`TimeInstant`); `link` (`String`); `echoedBy` (`int`); `type` (`int`).

c) (0.75 puntos) Método constructor, que crea una `PieceOfNews` que se ha producido en el instante `i`, publicada con enlace `l`, eco en `n` medios y de tipo `t`. Se supone como Precondición que los valores de estos parámetros son correctos.

d) (1.25 puntos) Método `equals`, que sobrescribe el de `Object` y comprueba si una `PieceOfNews` (`this`) es igual a otra, en concreto, si ambas se han producido en el mismo instante, han tenido el mismo eco y son del mismo tipo; los enlaces no se tienen en cuenta.

e) (1.75 puntos) Método `compareTo`, que compara una `PieceOfNews` (`this`) con otra noticia `other` en base a los criterios de popularidad que figuran a continuación y devuelve un `int` negativo si `this` es menos popular que `other`, positivo si `this` es más popular que `other` y `0` si `this` y `other` son igual de populares.
   Criterios de popularidad:
   - En principio, una noticia es menos popular que otra si ha sucedido antes.
   - Si han sucedido en el mismo instante, entonces una noticia es menos popular que la otra si su eco es menor.
   - A igualdad de instante y eco, una noticia es menos popular si está menos elaborada que la otra, entendiéndose que las noticias menos elaboradas vienen como audio, las intermedias vienen como vídeo, y las más elaboradas como texto.

f) (1.25 puntos) Método `toString`, que sobrescribe el de `Object` y que devuelve la descripción de la noticia, es decir, el instante, el enlace, su eco y, entre paréntesis, una palabra que indique su tipo: `text`, `video`, `audio`, como en el siguiente ejemplo:

`10:30 https://media.com/2019/10/31/climate-change2 150 (text)`

Solución:

```
public class PieceOfNews {
    public static final int AUDIO = 0, VIDEO = 1, TEXT = 2;

    private TimeInstant instant;
    private String link;
    private int echoedBy;
    private int type;

    public PieceOfNews(TimeInstant i, String l, int n, int t) {
        instant = i;
        link = l;
        echoedBy = n;
        type = t;
    }

    public boolean equals(Object o) {
        return o instanceof PieceOfNews
            && this.instant.equals(((PieceOfNews) o).instant)
            && this.echoedBy == ((PieceOfNews) o).echoedBy
            && this.type == ((PieceOfNews) o).type;
    }

    public int compareTo(PieceOfNews other) {
        int res = this.instant.compareTo(other.instant);
        if (res == 0) {
            res = this.echoedBy - other.echoedBy;
            if (res == 0) {
                res = this.type - other.type;
            }
        }
        return res;
    }

    public String toString() {
        String res = "";
        res += instant + " " + link + " " + echoedBy + " (";
        switch (type) {
            case TEXT:
                res += "text)"; break;
            case VIDEO:
                res += "video)"; break;
            default:
                res += "audio)";
        }
        return res;
    }
}
```

2. 2 puntos **Se pide**: dada la siguiente clase Programa `TestPieceOfNews`, completar el método `main` para que realice las acciones que se describen a continuación, suponiendo que se ubica en el mismo paquete que las clases `PieceOfNews` y `TimeInstant`. Esta clase programa debe usar las "constantes Java" de las clases del mismo paquete siempre que se requiera.

```
public class TestPieceOfNews {

    /** Devuelve un valor entero aleatorio en [ini, fin], 0 <= ini < fin. */
    private static int random(int ini, int fin) {
        return (int) (Math.random() * (fin - ini + 1) + ini);
    }

    public static void main(String[] args) {
        ...
    }
}
```

a) (0.25 puntos) Crear un `TimeInstant ti` que represente las 10 horas y 30 minutos.

b) (0.25 puntos) Crear una `PieceOfNews n1` que se ha producido en el instante `ti`, de tipo audio, de la que se han hecho eco 200 medios y cuyo enlace es `"https://media.com/2019/10/31/climate-change1"`.

c) (0.25 puntos) Asignar a una variable `echo2` el resultado de generar aleatoriamente un valor en el intervalo [2, 500].
d) (0.25 puntos) Crear una `PieceOfNews n2` que se ha producido en el instante `ti`, de tipo texto, de la que se han hecho eco `echo2` medios y cuyo enlace es `"https://media.com/2019/10/31/climate-change2"`.
e) (0.25 puntos) Asignar a una variable `resC` el resultado de aplicar el método `compareTo` a las variables `n1` y `n2`.
f) (0.75 puntos) En función del valor de `resC`, mostrar por pantalla (con el formato de `toString`) la noticia de mayor popularidad. Si la popularidad fuera la misma, mostrar las dos por pantalla separadas por el símbolo de igualdad.

Solución:

```
public class TestPieceOfNews {

    /** Devuelve un valor entero aleatorio en [ini, fin], 0 <= ini < fin. */
    private static int random(int ini, int fin) {
        return (int) (Math.random() * (fin - ini + 1) + ini);
    }

    public static void main(String[] args) {
        TimeInstant ti = new TimeInstant(10, 30);
        String link = "https://media.com/2019/10/31/climate-change1";
        PieceOfNews n1 = new PieceOfNews(ti, link, 200, PieceOfNews.AUDIO);

        int echo2 = random(2, 500);
        String link2 = "https://media.com/2019/10/31/climate-change2";
        PieceOfNews n2 = new PieceOfNews(ti, link2, echo2, PieceOfNews.TEXT);

        int resC = n1.compareTo(n2);
        if (resC == 0) {
            System.out.println(n1 + " = " + n2);
        }
        else if (resC < 0) {
            System.out.println(n2);
        }
        else { System.out.println(n1); }
    }
}
```

3. 2 puntos Dada la siguiente clase `Exercise3`, en la que se usa la clase `TimeInstant` de la pregunta anterior, se pide escribir lo que muestra por pantalla la ejecución del programa.

```
public class Exercise3 {

    public static void main(String[] args) {
        TimeInstant aux = new TimeInstant(5, 6); int j = 1;
        System.out.println("En main: " + j + " " + aux.getHours() + " " + aux.getMinutes());
        m2(aux, j);
        System.out.println("En main: " + j + " " + aux.getHours() + " " + aux.getMinutes());
    }

    private static void m2(TimeInstant aux, int j) {
        System.out.println("En m2:   " + j + " " + aux.getHours());
        int nH = aux.getHours() + j;
        j++; aux.setHours(nH);
        m1(aux, j);
        System.out.println("En m2:   " + j + " " + aux.getHours());
    }

    private static void m1(TimeInstant aux, int j) {
        System.out.println("En m1:   " + j + " " + aux.getMinutes());
        int nM = aux.getMinutes() + j;
        j++; aux.setMinutes(nM);
        System.out.println("En m1:   " + j + " " + aux.getMinutes());
    }
}
```

Solución:

```
En main: 1 5 6
En m2:   1 5
En m1:   2 6
En m1:   3 8
En m2:   2 6
En main: 1 6 8
```