

IIP Primer Parcial - ETSInf

10 de Noviembre de 2014. Duración: 1 hora y 30 minutos.

Se desea hacer una aplicación para representar un juego con bloques de distintos colores y dimensiones, apilables en torres. Cada uno de estos bloques tiene asociado un **color** (azul o rojo) y una **dimensión** (número entero entre 1 y 50, ambos incluidos).

Las reglas de este juego indican que:

- Los bloques apilados en una torre deben seguir colores alternos (encima de un bloque azul solo puede haber un bloque rojo y viceversa).
- Encima de un bloque de dimensión x solo puede haber un bloque de dimensión y , donde $y \leq x$ (la torre se estrecha hacia la punta, es decir, se ensancha hacia la base).
- Un bloque puede ser un **comodín**, en cuyo caso puede ir encima de cualquier otro bloque independientemente de su color. Ahora bien, un bloque comodín debe de respetar, como cualquier otro, la regla de la dimensión.

1. 6 puntos Se pide implementar la clase **Bloque**, para lo que se debe:

- (0.5 puntos) Definir los atributos de clase públicos y constantes que representan los dos colores posibles de un bloque: **AZUL** y **ROJO**, con valores enteros 0 y 1, respectivamente. Estos atributos deberán ser utilizados siempre que se requiera (tanto en la clase **Bloque** como en la clase **TorreBloques**).
- (0.5 puntos) Definir los atributos de instancia privados **color** (**int**), **dimension** (**int**), **comodin** (**boolean**).
- (1.5 punto) Implementar dos constructores:
 - Un constructor general con los parámetros apropiados para inicializar todos los atributos de instancia.
 - Un constructor por defecto que crea un **Bloque** azul que no es comodín y cuya dimensión se determina aleatoriamente dentro del rango [1, 50].
- (0.5 puntos) Escribir el método **consultor** y el método modificador del atributo **dimension**.
- (1 punto) Escribir el método **equals** (que sobrescribe el de **Object**) para comprobar si un bloque es igual a otro, i.e. si otro es también un bloque y los atributos de uno y otro coinciden uno a uno.
- (1 punto) Escribir el método **toString** (que sobrescribe el de **Object**) para que el resultado tenga un formato como el mostrado en los siguientes ejemplos: “(Color: rojo, dimensión: 22 y SÍ es comodín)”, “(Color: azul, dimensión: 15 y NO es comodín)”.
- (1 punto) Escribir un método **puedeEstarEncimaDe(Bloque b)** que compruebe si un bloque puede, aplicando las reglas ya mencionadas, estar encima de otro **b** que recibe como argumento. Por ejemplo, dadas dos variables **a** y **b** de tipo **Bloque**, **a.puedeEstarEncimaDe(b)** será cierto si y solo si la dimensión del bloque **a** es menor o igual que la del bloque **b** y, o bien **a** es un comodín, o bien los colores de **a** y **b** son distintos.

2. 4 puntos Se pide implementar el programa **TorreBloques**, cuyo objetivo es poder hacer pruebas con pequeñas torres de algunos bloques. Para ello, se pide implementar dicha clase con un método **dimensionValida** que:

- (1.25 puntos) limite el valor de la dimensión **d** que se le pasa como parámetro a uno de los valores del intervalo [1, 50]. Esto es: si **d** es menor que 1, devuelve 1; si **d** es mayor que 50, devuelve 50; en cualquier otro caso, devuelve **d** sin modificarlo.

y con un método **main** que realice las siguientes acciones:

- (0.25 puntos) Crear un **Bloque b1** con el constructor por defecto.
- (0.5 puntos) Crear un **Bloque b2** de color azul, dimensión 30 y que sea comodín (con el constructor general).
- (1 punto) Tras leer su color y dimensión desde teclado, crear un **Bloque b3** que no sea comodín.
El color se solicitará al usuario como un **String** con valores “rojo” o “azul”; si el usuario introduce cualquier otro valor, el color del **Bloque** será rojo.
El valor de dimensión se solicitará al usuario como un entero en [1,50] y, por si el usuario introduce cualquier otro valor, se deberá invocar al método **dimensionValida** para garantizarlo.
- (0.25 puntos) Mostrar por pantalla los tres objetos creados.
- (0.75 puntos) Comprobar si se puede formar una torre situando el bloque **b3** encima del **b2** y este último encima del bloque **b1**. Después, mostrar por pantalla un mensaje con el resultado de dicha comprobación, i.e. si se ha podido formar tal torre o no.

Se deben importar las clases que se consideren necesarias y utilizar las constantes definidas en la clase **Bloque** donde sea oportuno.

Solución:

```
public class Bloque {
    private int color;
    private int dimension;
    private boolean comodin;

    public static final int AZUL = 0;
    public static final int ROJO = 1;

    public Bloque() {
        this.color = AZUL;
        this.dimension = (int) (1 + Math.random() * 50);
        this.comodin = false;
    }

    public Bloque(int color, int dimension, boolean comodin) {
        this.color = color;
        this.dimension = dimension;
        this.comodin = comodin;
    }

    public int getDimension() {
        return dimension;
    }

    public void setDimension(int dim) {
        dimension = dim;
    }

    public boolean puedeEstarEncimaDe(Bloque b) {
        return this.dimension <= b.dimension
            && (this.comodin || this.color != b.color);
    }

    public boolean equals(Object o) {
        return o instanceof Bloque
            && this.color == ((Bloque) o).color
            && this.dimension == ((Bloque) o).dimension
            && this.comodin == ((Bloque) o).comodin;
    }

    public String toString() {
        String col = "azul";
        if (this.color == ROJO) col = "rojo";
        String com = "NO";
        if (this.comodin) com = "SÍ";
        return "(Color: " + col + ", dimensión: " + dimension + ", "
            + com + " es comodín)";
    }
}
```

Solución:

```
import java.util.Scanner;
public class TorreBloques {

    public static int dimensionValida(int dimension) {
        if (dimension < 1) return 1;
        else if (dimension > 50) return 50;
        else return dimension;
    }

    public static void main(String[] args) {
        Scanner teclado = new Scanner(System.in);

        Bloque b1 = new Bloque();

        Bloque b2 = new Bloque(Bloque.AZUL, 30, true);

        System.out.print("Introduce el color del bloque 3 (azul/rojo): ");
        String color = teclado.next().trim().toLowerCase();
        int codColor = Bloque.ROJO;
        if (color.equals("azul")) codColor = Bloque.AZUL;

        System.out.print("Introduce la dimensión del bloque 3 [1,50]: ");
        int dimension = dimensionValida(teclado.nextInt());

        Bloque b3 = new Bloque(codColor, dimension, false);

        System.out.println("Bloque 1: " + b1 + "\nBloque 2: " + b2 + "\nBloque 3: " + b3);

        boolean res = b3.puedeEstarEncimaDe(b2) && b2.puedeEstarEncimaDe(b1);
        System.out.print("\nLa torre formada con esos tres bloques");
        if (!res) System.out.print(" NO");
        System.out.println(" es válida");
    }
}
```