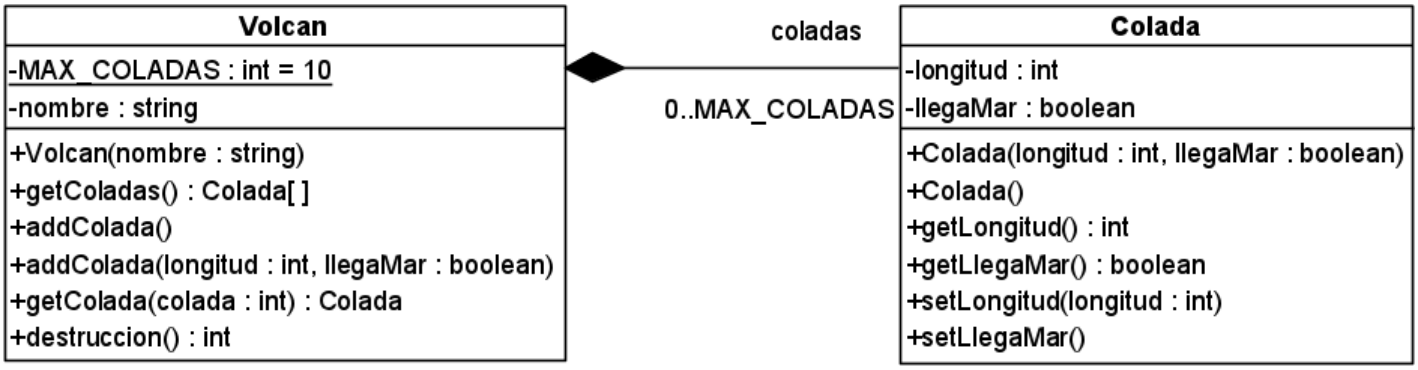
Programación y Diseño Orientado a Objetos

Ejercicio tipo examen de prácticas P1, P2 y P3

Descripción de la actividad

Se desea implementar una aplicación, en Java, para la gestión de las coladas de volcanes según el siguiente diagrama UML. *En el diagrama no se han añadido todos los constructores por simplicidad*.



Ejercicios

1.- (0,5 puntos) Asegure un **diseño coherente** y **bien estructurado** de su código en todo momento, como la generación de constructores con **todos los atributos**, uno por **defecto** y uno de **copia**; y no repita código innecesariamente (reutilización).

2.- (0,25 puntos) Evite usar **múltiples retornos** en sus métodos, optando en cambio por un único punto de retorno, por razones de diseño.

3.- (0,25 puntos) Evite el uso de **«números mágicos»** en su código. Además, en lugar de asignar valores booleanos directamente como *true* o *false*, construya expresiones booleanas a partir de las variables y condiciones pertinentes.

4.- (1 puntos) Declare la clase Colada según se indica en el diagrama de clases. **Implemente los constructores** indicados, teniendo en cuenta que el constructor por defecto debe crear, **utilizando el constructor parametrizado**, una colada de longitud 0 y que no llega al mar.

5.- (1 punto) Implemente los métodos necesarios para **consultar y establecer los atributos** de una **Colada**, tal y como se indica en el diagrama de clases. Si la longitud de la colada es menor de 0 se debe escribir un mensaje por consola con el error y no se modificará el atributo.

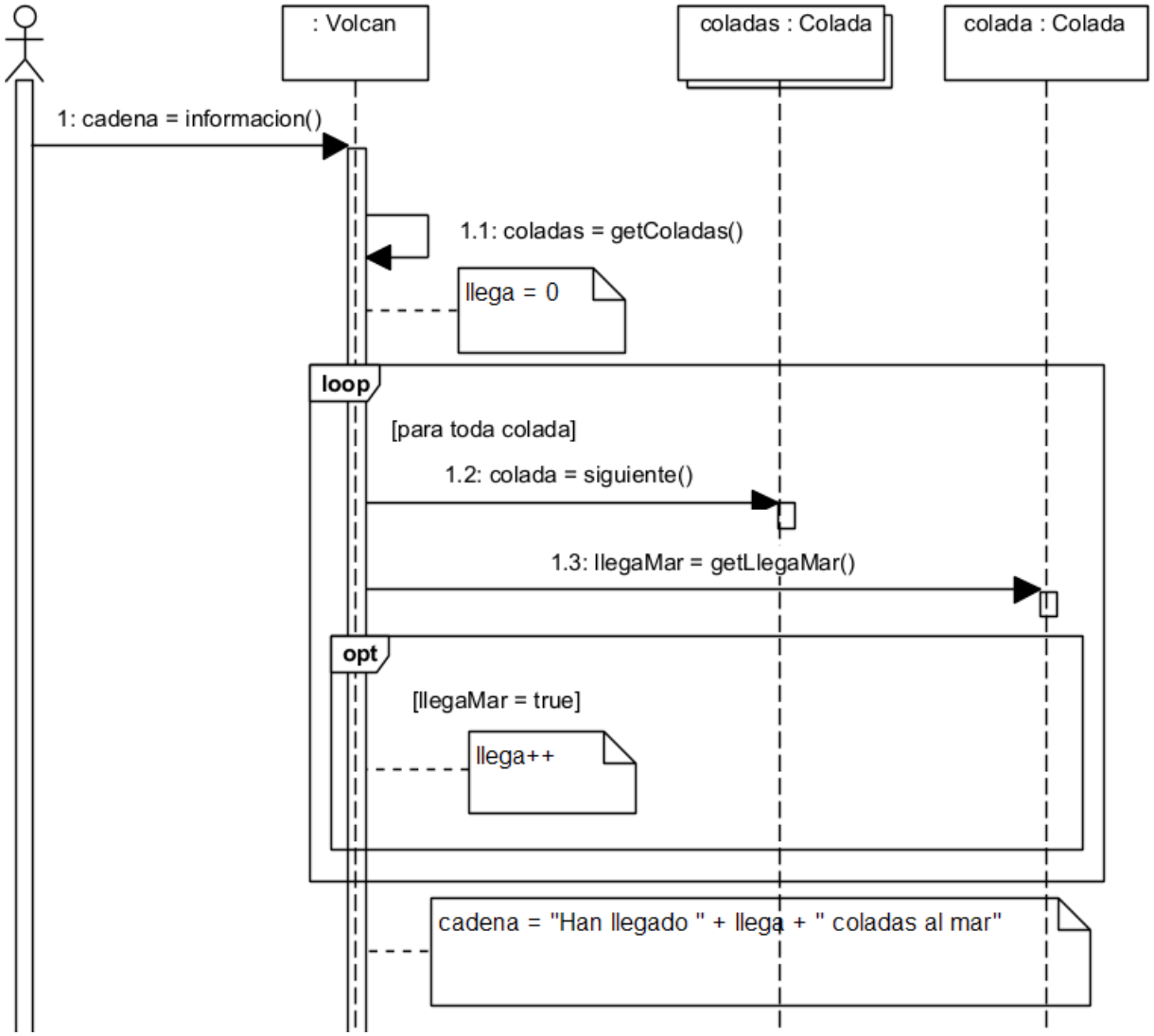
6.- (**1 puntos**) **Declare la clase Volcan** según se indica en el diagrama de clases. Implemente el **constructor parametrizado** que se indica en el diagrama y, además, el método que devuelve todas las coladas que tiene el volcán, **getColadas**.

7.- (1.5 puntos) Implemente los métodos **addColada** de la clase **Volcan** tal y como se indica en el diagrama de clases, teniendo en cuenta que al añadir no puede sobrepasarse el máximo de coladas permitido. Si no se pudiera añadir una colada, se mostrará un mensaje por consola indicando el error. Si no se pasan parámetros a **addColada**, entonces se añade una **Colada** por defecto.

8.- (1 punto) Implemente el método **getColada** para **consultar coladas** de un **Volcan**, tal y como se indica en el diagrama. Las coladas se consultan por el orden en el que se crearon (la 1, la 2, etc.). Si la colada a consultar no existe, se mostrará un error por consola y se devolverá un valor nulo.

9.- (1.5 puntos) Implemente el método **destruccion** de la clase **Volcan** tal y como se indica en el diagrama. Este método devuelve la longitud total destruida por el volcán, es decir, la suma de la longitud de todas sus coladas.

10.- (1 punto) Implemente el método especificado en el siguiente diagrama:



11.- (1 punto) Complete el **programa principal** para que realice las siguientes operaciones:

a) Declare un volcán de nombre «Cumbre Vieja» y añada al volcán tres coladas: la primera por defecto, la segunda con una longitud de 10 y que ha llegado al mar y la tercera con una longitud de 12 y que no ha llegado al mar.

b) Partiendo del volcán, muestre por consola la longitud de su segunda colada, utilizando el método implementado en el ejercicio 5.

c) Partiendo del volcán, establezca que la tercera colada ha llegado al mar.

d) Muestre por consola la información (método *informacion()*) del volcán.