Instituto Politécnico Nacional
Unidad Profesional
Interdisciplinaria De Ingeniería
Campus Zacatecas

Ingeniería en Sistemas Computacionales

Práctica 9 - Forzamiento

Alumna: Vanessa Melenciano Llamas

Boleta: 2020670081

Profesora: Monreal Mendoza Sandra Mireya

Materia: Programación Orientada a Objetos

Grupo: 2CM2

Fecha de entrega: 16 de enero de 2021

Índice

Introducción

En programación orientada a objetos se denomina polimorfismo a la capacidad que tienen los objetos de una clase de responder al mismo mensaje o evento en función de los parámetros utilizados durante su invocación. Un objeto polimórfico es una entidad que puede contener valores de diferentes tipos durante la ejecución del programa.

En JAVA el término polimorfismo también suele definirse como ‘Sobrecarga de parámetros’, que así de pronto no suena tan divertido, pero como veremos más adelante induce a cierta confusión. En realidad, suele confundirse con el tipo de polimorfismo más común, pero no es del todo exacto usar esta denominación. Es decir, se refiere a la idea de "tener muchas formas", ocurre cuando hay una jerarquía de clases relacionadas entre sí a través de la herencia

Por lo general diremos que existen 3 tipos de polimorfismo:

- Sobrecarga: El más conocido y se aplica cuando existen funciones con el mismo nombre en clases que son completamente independientes una de la otra.
- Paramétrico: Existen funciones con el mismo nombre, pero se usan diferentes parámetros (nombre o tipo). Se selecciona el método dependiendo del tipo de datos que se envíe.
- Inclusión: Es cuando se puede llamar a un método sin tener que conocer su tipo, así no se toma en cuenta los detalles de las clases especializadas, utilizando una interfaz común.

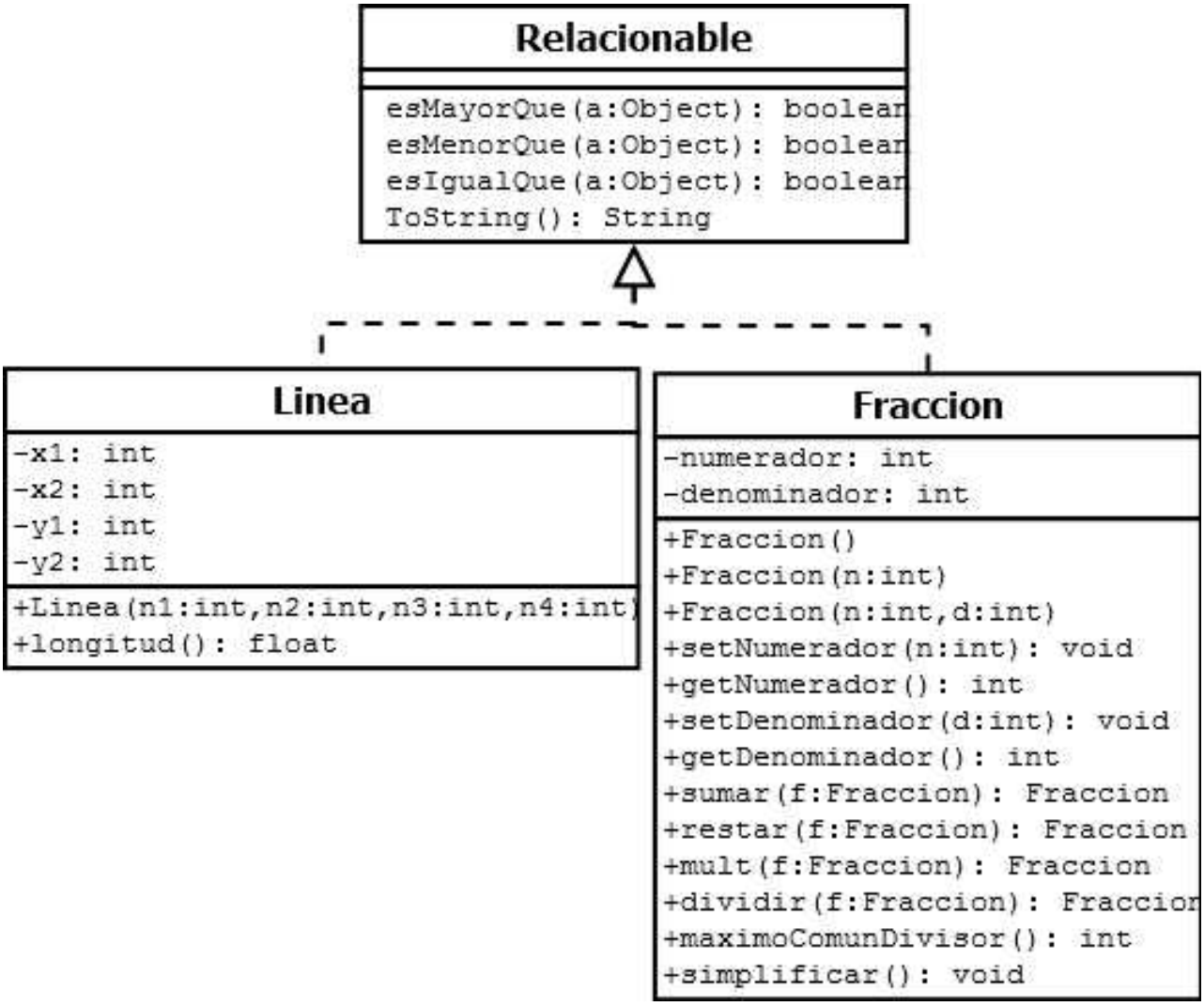
En líneas generales en lo que se refiere a la POO, la idea fundamental es proveer una funcionalidad predeterminada o común en la clase base y de las clases derivadas se espera que provean una funcionalidad más específica.

Objetivos

Usar el polimorfismo con sobreescritura y forzamiento.

Desarrollo

Diseño (UML) y funcionamiento de la solución



Para poder realizar la práctica, se usaron las tres clases que aparecen en el diagrama de clase, donde, tanto la clase Fraccion como Linea, sobreescriben los métodos de la clase Relacionable para tener su correcto funcionamiento cada una.

Funciones

Para comprobar que el programa funcione correctamente, se utilizó una clase llamada TestRelacionable, donde se colocó un main:

```java
public class TestRelacionable {

    public static void main(String args[]){
        Relacionable[] array = new Relacionable[4];

        array[0] = new Linea( n1: 2,  n2: 2,  n3: 4,  n4: 1);
        array[1] = new Linea( n1: 5,  n2: 3,  n3: 1,  n4: 3);
        array[2] = new Fraccion( n: 3,  d: 8);
        array[3] = new Fraccion( n: 7,  d: 5);

        for(Relacionable r: array){
            System.out.println( x: r.ToString());
        }
    }
}
```

Se creó un arreglo de 4 objetos de tipo Relacionables, usando el forzamiento al introducir en el arreglo tipo Linea y Fraccion. Al final, con un for se recorre el arreglo llamando al método heredado “ToString”, para mostrar en pantalla lo siguiente:

```
put - Run (TestRelacionable)  ×
Building POO_P9 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------

--- exec-maven-plugin:1.5.0:exec (default-cli) @ POO_P9 ---
Coordenadas del punto de inicio de la línea: (2,2)
Coordenadas del punto del final de la línea: (4,1)
Distancia de la línea: 3.0
Coordenadas del punto de inicio de la línea: (5,3)
Coordenadas del punto del final de la línea: (1,3)
Distancia de la línea: 2.828427
3/8
7/5
------------------------------------------------------------------
```

Errores detectados

En esta práctica no hubo grandes dificultades ya que nos temas vistos, se habían usado con anterioridad, siendo solo una comprobación de la aplicación de todo.

Posibles mejoras

En este caso, lo que se puede mejorar es seguir practicando la utilización de la sobre escritura aplicando el polimorfismo, pero en general todo claro.

Conclusión

El polimorfismo es una relajación del sistema de tipos, de tal manera que una referencia a una clase (atributo, parámetro o declaración local o elemento de un vector) acepta direcciones de objetos de dicha clase y de sus clases derivadas.

Se logro el objetivo de la práctica al comprender la aplicación del polimorfismo en POO, utilizando la sobreescritura y forzamiento.

Referencia

MARTÍNEZ, G. (2009). PROGRAMACIÓN ORIENTADA A OBJETOS CON APRENDIZAJE ACTIVO. *Scientia Et Technica,* pp. 163-168.