Grupo 2213
Marzo de 2019

Álvaro Sánchez
Nicolás Serrano

alvaro.sanchezromero@estudiante.uam.es
nicolas.serranos@estudiante.uam.es

Análisis y Diseño de Software

Practica 3

Álvaro Sánchez
Nicolás Serrano
alvaro.sanchezromero@estudiante.uam.es
nicolas.serranos@estudiante.uam.es

Tabla de contenido

Álvaro Sánchez
Nicolás Serrano
alvaro.sanchezromero@estudiante.uam.es
nicolas.serranos@estudiante.uam.es

Apartado 1. Las clases básicas Camino

Cuando se nos pidió un método para obtener en cualquier momento obtener el total acumulado hasta ese instante del coste energético de todos los caminos creados lo declaramos el getter de ese atributo estático como un método estático también.

Apartado 2. La clase básica Posada

A la hora de implementar los constructores, el constructor que tuviera menos argumentos, en este caso el constructor que recibe como único parámetro el nombre de la posada dentro llamará al constructor que recibe como parámetro el mayor número de parámetros, en este caso el que recibe como segundo parámetro el valor de recuperación energética. Más tarde mantendremos el diseño para el constructor de la posada con el parámetro luz.

Apartado 3. La clase básica Explorador

En el método recorre(Camino) para además de utilizar los puedeRecorrerCamino(camino) y puedeAlojarseEn(Posada) y comprobar si el camino existe para saber si el explorador puede recórrelo, comprobamos que el camino que se nos pasa por argumento pertenece a la posada actual en la que está el explorador.

Luego, en recorre(Posada …) para comprobar si algún camino no se ha podido realizar utilizamos un booleano “error” inicializado a false que si en algún momento falla la iteración de recorre(Camino) se pone a true, permitiendo la iteración de recorre(Camino) pero guardando que se ha producido un error para luego devolver false al finalizar el método.

Apartado 4. Ejemplo de uso y pruebas de las clases anteriores

Hemos separado los testers en uno por clase, por lo que tenemos TesterCamino, TesterPosada y TesterExplorador aparte de EjemploDeExploradoresBasicos que se nos da y los de los siguientes apartados.

Cada uno de los testers viene comentado paso a paso cada uno de los métodos que usan y los resultados que esperan, imprimiéndose estos en algunos casos por pantalla.

Apartado 5. Clase Simulacion a partir de archivos de texto

La clase Simulación tiene como atributos una lista de posadas y un explorador si guardar ella misma los caminos que se crean pues se almacenan tras crearlos directamente en las posadas.

Para simplificar y clarificar el código el constructor además de recibir por argumento un string con la ruta de los ficheros a abrir llama a los métodos privados leerPosadas, leerCaminos y leerExplorador en el que cada uno abre el fichero que se le pasa por argumento y almacena la información de estos. Además, leerExplorador aparte de guardar la información del explorador va recorriendo los caminos de posada en posada como se nos pide e imprime el explorador tras recorrer un camino.

Álvaro Sánchez
Nicolás Serrano
alvaro.sanchezromero@estudiante.uam.es
nicolas.serranos@estudiante.uam.es

También encontramos el método privado Posada buscarPosada(String posada) que busca una posada en la clase con el nombre que se le pasa por argumento, así nos es más sencillo para almacenar las posada origen y destino de un camino.

Por último en EjemploDeUsoSimulación ha sido modificado para en vez de pasarse por argumento a new Simulación("POSADAS.txt", "CAMINOS.txt", "EXPLORADOR.txt") se les pasa la ruta de estos archivos cuando se ejecuta el programa desde el directorio apropiado.

Apartado 6. Otras clases de caminos, posadas y exploradores

Creamos la clase trampa como una clase que hereda de camino, añadiéndole los atributos facorCoste y probRetorno y sobrescribiendo algunas funciones como getDestino, costeEspecial, esTrampa y toString.

En la implementación de la clase Luz para que formase un atributo de Posada, hemos modificado los contructores como hemos indicado en el apartado 2.

Mago es una clase que hereda de explorador que se sobrescribe algunas clases de Explorador como puedeAlojarseEn(Posada) o recorre(Camino). Durante el desarrollo pensamos en crear dos clases que heredaran de mago que fuera hada y hechicero, pero por simplicidad y trabajar con menos ficheros fuente decidimos crear hada y hechicero como la enumeración TipoDeMago, siendo ese un atributo dentro de la clase Mago. Por eso observamos puedeAlojarseEn(Posada) que hacemos un if/else comprobando el tipo de mago que es y realizando la acción esperada.

Detectamos a la hora de crear un mago pues este puede crearse en una posición en la que no puede alojarse. Pero decidimos mantener este diseño porque las alternativas eran o que la posición fuese null o que no se creara el explorador y no nos parecieron buenas alternativas para tan pequeña incongruencia.

Por último, como en el apartado 4, desarrollamos un tester para cada clase, siendo estos TesterMago, TesterLuz y TesterTrampa comentado paso a paso cada uno de los métodos que usan y los resultados que esperan, imprimiéndose estos en algunos casos por pantalla.

Apartado 7 (opcional). Otras clases de caminos, posadas y exploradores

Este apartado no lo logramos terminar del todo y por eso no está incluido en los ficheros fuente, pero si en la memoria explicando el funcionamiento del programa, como era lo que teníamos.

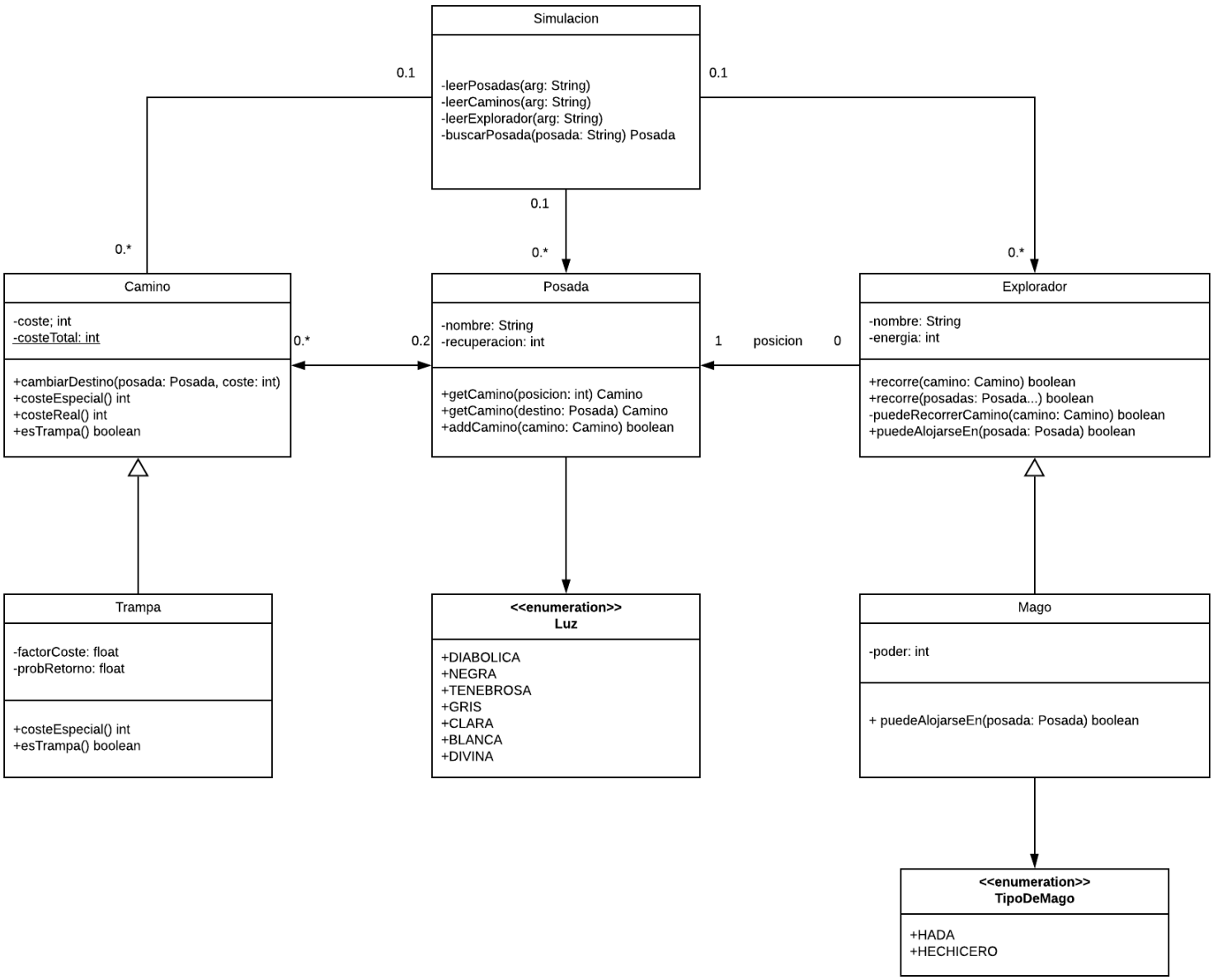
Habíamos creado el fichero EXPLORADORES.txt para que al principio leerExplorador() leyera todos los exploradores que íbamos a tener con su nombre, la energía y la posada de inicio de esos exploradores como en EXPLORADOR.txt normal y estos se almacenaban en una lista de exploradores.

Después en vez de tener en cada línea simplemente el nombre de la posada que se iba a recorrer teníamos también el nombre del explorador que iba a ir a esa posada utilizando una función parecida a buscarPosada(String posada) que ya utilizábamos antes en el apartado 5.

Álvaro Sánchez
Nicolás Serrano
alvaro.sanchezromero@estudiante.uam.es
nicolas.serranos@estudiante.uam.es

También en para añadir la nueva funcionalidad de POSADAS.txt tras la recuperación pusimos el tipo de luz de esta que dentro de leerPosadas(), llamaba a buscarLuz(String luz) que recibía el nombre del tipo de luz y devolvía el elemento de la enumeración llamando así al constructor con el atributo luz.

Diagrama de clases



Nota

Documentación en ./doc, código fuente en ./src y ficheros utilizados en ./txt

Compilar y ejecutar desde el directorio ./src