Un refactorizador simple

José Antonio Leiva Izquierdo
Plettac Seguridad y Sistemas, S.A.

<jleiva@isys.dia.fi.upm.es>

I Concurso Universitario de Programación de la Comunidad Autónoma de Madrid (CUPCAM 2003): enunciado del problema D

Actualmente están surgiendo nuevas tendencias en el desarrollo de software encaminadas a cambiar el modo en que trabajan los programadores. La refactorización es un ejemplo de ello. Con este término se pretende definir el proceso mediante el cual se cambia la estructura interna del software para facilitar su comprensión y posterior mantenimiento, siempre sin alterar su comportamiento.

La metodología consiste en la aplicación de una serie de patrones llamados 'patrones de refactorización'. La mayoría de ellos son muy sencillos de aplicar, como por ejemplo, cambiar el nombre de una clase, obtener una interfaz a partir de métodos públicos, etc. El problema que planteamos aquí consiste en aplicar uno de estos patrones conocido como 'moción de código' a un fragmento de programa previamente analizado. El cometido de este patrón es minimizar el acoplamiento entre módulos. De este modo, el principal objetivo del problema es obtener un programa óptimo en términos de acoplamiento mediante la aplicación reiterada de este patrón. Se utilizará una forma bastante inocente de medir el acoplamiento de un programa, cuantificando la interacción entre los módulos que lo componen. Esto es, el número de veces que un método llama a métodos de módulos diferentes del suyo.

Un fragmento de programa estará siempre compuesto por varios módulos. Cada módulo poseerá métodos y cada método realizará una serie de llamadas a otros métodos. No importa si el método llamante y el método llamado pertenecen o no al mismo módulo. En ocasiones, un método del módulo A llamará a más métodos de otro módulo, digamos B, que a métodos de A. Este comportamiento aumenta el acoplamiento entre los módulos A y B complicando la comprensión y el futuro mantenimiento del programa. El patrón del problema tomaría el método mencionado y lo trasladaría de A al módulo B.

Descripción de la entrada

Tanto módulos como métodos están identificados por un número entre 0 y 100. El formato para especificar un método es el identificador del correspondiente módulo, seguido por el carácter '.' y un identificador único dentro de este módulo. Se definen todos los métodos, tanto si interactúan con métodos de otros módulos como si no.

La entrada estará compuesta por un conjunto de fragmentos de código previamente analizados. Cada fragmento se define como una serie de líneas. La primera indica la cantidad de módulos del programa seguida por la cantidad de métodos que pertenecen a cada módulo. Las siguientes líneas indican las llamadas entre métodos. Cada una de estas líneas consiste en el identificador de un método seguido de los identificadores de los métodos a los que llama. Cuando un método no llame a ningún otro, sólo aparecerá su identificador.

Descripción de la salida

Para cada fragmento de código, la salida deberá mostrar el valor mínimo de interacción entre módulos tras la aplicación del patrón indicado.

Ejemplo de entrada

```
2 5 5
0.0
0.1
0.2
0.3 0.1 0.2 0.4 1.0
0.4 0.0 0.1 1.0 1.1 1.2 1.4
1.0
1.1
1.2
1.3 0.0 0.1 0.2 0.3
1.4 1.1 1.2 1.3
2 3 2
0.0 0.2
0.1 0.0 0.2
0.2 1.0 1.1
1.0 0.0 0.1 1.1
1.1
3 2 1 1
0.0 0.1
0.1 0.0 2.0
1.0 0.0 0.1
2.0 0.0
```

Ejemplo de salida

```
5
0
0
```