SESIÓN 02

La práctica corresponde al:
Tema 1: Introducción a EDAT en Java (Colecciones Abstractas)

Peso en la nota final:
0%

Objetivos

Con esta práctica se pretende que el estudiante trabaje y se familiarice con los siguientes conceptos:

- Clase, Objeto, Herencia, "`interface`", "`implements`", etc.
- Sintaxis de Genericidad en Java
- Clase `AbstractCollection`
- Interfaz `Iterator`

Conceptos Teóricos

Una vez se ha practicado el trabajo con un iterador en la práctica anterior, en la actual se quiere comprobar cómo se trabaja con un iterador dentro de una colección.
En la práctica se pretende que el alumno se familiarice con la estructura interna de una colección en Java. Este tipo de clases son las que implementan las diferentes estructuras de datos en java. Para forzar a que tengan ciertos comportamientos unificados y por tanto ser más sencillas de emplear, todas se inscriben en la jerarquía que se inicia con el interfaz `Collection`.
Para facilitar la creación de nuevas clases que implementen este interfaz, se facilita la clase abstracta `AbstractCollection`. En ella aparecen predefinidos varios métodos del interfaz `Collection`. Al ser implementaciones muy genéricas (basadas en la iteración), es probable que el programador tenga que sobrescribir alguna para ganar en eficiencia.
Se pueden encontrar explicaciones detalladas de estas características en el material asociado a esta sesión de prácticas de la asignatura en el correspondiente espacio en UBUVirtual.

Trabajo del Alumno

Durante la sesión de prácticas, se explicarán las distintas técnicas necesarias para la resolución del problema y que el estudiante utilizará para obtener el nuevo código que constituirá la entrega de la práctica.
Dicha entrega debe de incluir el conjunto de test y las modificaciones de código que el estudiante estime necesario para cumplir con los requisitos solicitados, incluyendo la salida de la utilidad "`javadoc`".

1. Programa

Se pide en esta práctica el implementar una colección abstracta que permita el manejo de matrices de datos contiguas en memoria como si se tratara de una colección lineal. Se incluirá este comportamiento en la clase `ColeccionArray2D`, incluida en el material.
Para poder realizar las operaciones necesarias para trabajar como colección, deberá incluir:

- Un método constructor que permitirá almacenar la información bi-dimensional en el interior de la estructura de datos. Se contará con que la estructura NO cambiará de tamaño a lo largo de su uso en los programas. La clase será una vista sobre los datos, de forma que, si cambian estos, el cambio se verá reflejado en los datos proporcionados por la clase.

- Una clase interna que implemente `Iterator` y que permita acceder a cada uno de los elementos de forma secuencial (se recorrerá primero la fila completa correspondiente de la matriz antes de pasar a la siguiente). El iterador deberá implementar el método `remove()`.

- Un método

  ```
  E set (int posición, E dato)
  ```

  que permita establecer el dato guardado en la estructura, dada su posición en una supuesta iteración secuencial de su contenido. Se devuelve como resultado el antiguo valor que se ha sustituido.

2. Informe

No se solicita en esta práctica.

Condiciones de Entrega

- La práctica se realizará en grupos de 1 o 2 personas.
- La entrega se hará SOLAMENTE por medio de la plataforma UBUVirtual. Cada miembro del grupo deberá subir su propia copia de la solución. No se admitirán soluciones fuera de plazo o por otros medios de entrega (e-mail, mensajería interna, …).
- La entrega incluirá la información de quienes son los autores de la misma, tanto en los ficheros de código fuente como en los documentos asociados.
- Cada entrega consistirá en un fichero comprimido (formato .zip), con la estructura de nombre

"`Apellidos1Nombre1_Apellidos2Nombre2`"

- Qué deberá incluir el fichero comprimido:
  - Código fuente de la solución (dentro de la estructura de paquetes necesaria)
  - Documentación en formato Javadoc
  - Informe en formato PDF (en aquellas prácticas que se solicite).
- No hace falta entregar ficheros binarios

Criterios de Evaluación

- **Corrección del funcionamiento**. Se valorará como aspecto más importante que el funcionamiento de la estructura de datos implementada cumpla con todas las condiciones solicitadas, tanto en la documentación oficial como en el enunciado. Se facilitan al alumno un subconjunto de los tests que se emplearán en la corrección, pero se aconseja que el alumno realice de forma adicional los que crea conveniente.
- **Corrección del código**. Ausencia de *warnings* u otros elementos no deseables como variables no utilizadas, código que no se emplea, catch vacíos, ausencia de marcas sobre los métodos (*@Override*, @...) etc.
- **Documentación**. Las explicaciones y razonamientos que se incluyan a modo de comentarios en el código que entrega el alumno. Se solicita que el alumno entregue adicionalmente la documentación generada en formato javadoc, junto con los ficheros de código fuente.
- En este caso, no se evaluará complejidad algorítmica del código.