Departamento de Lenguajes y Sistemas Informáticos

Javaluador

Departamento de Lenguajes y Sistemas Informáticos

Usuario: np31

Instrucciones generales

Cada problema de la colección tiene un identificador nummérico `n` (por ejemplo, el identificador del problema P001 es 001).

Para resolver el problema `n = 001, 002,...` debe crearse un fichero llamado `Pn.java` en lenguaje `Java 1.8` cuya primera línea debe ser obligatoriamente

```
package AA;
```

Además, el fichero definirá la clase pública

```
public class Pn {}
```

- Los enunciados:
  - están formato **PDF** y especifican qué función o funciones debe contener como mínimo esta clase, así como su interpretación;
  - cuyo identificador es menor que 10 corresponden a problemas de difícil optimización y tienen por objetivo mostrar algunos ejemplos donde **NO** se pueden aplicar las técnicas estudiadas en la asignatura;
  - cuyo **identificador** es **mayor o igual a 10** corresponden a problemas donde se puede aplicar alguna de las **técnicas de optimización** estudiadas como `programación dinámica` o `ramificación y poda`;
  - marcados con un asterisco (*) corresponde a problemas con mayor dificultad.
- El programa no debe contener instrucciones de apertura, cierre, lectura o escritura de ficheros ni cláusulas `throw`. Tampoco llamadas a las funciones de la clase `System` como `exec`, `exit` o `halt`.
- **Javaluador** comprobará el código con varios ejemplos de prueba. Los mensajes de error que pueden recibirse son los siguientes:
  - Error de presentación: el código enviado incluye funciones de acceso a ficheros, no declara `package AA;` o no contiene la clase `Pn`.
  - Error de compilación: el programa no es Java 1.8 válido o no contiene la función requerida.

    - Memoria máxima excedida: el programa requiere más memoria de la permitida (100MB).
    - Tiempo máximo excedido: el tiempo máximo de ejecución (10 segundos) ha sido superado.
    - Error de ejecución: error durante la ejecución de las pruebas.
    - Código incorrecto: el programa no resuelve adecuadamente algunos de los casos de prueba.

- Cada enunciado contiene al menos un ejemplo de entrada y salida correctos. La mayor parte de los problemas requiere procesar una cadena de texto de entrada (este formato facilita su supervisión en el laboratorio). Por ello, se recomienda usar `String.split("\\p{Space}+")` para filtrar las entradas de texto.
- La mayor parte de los problemas requieren optimizar un valor (mediante una función llamada `best` o `bestValue`) u obtener una solución óptima (mediante una función llamada `bestSolution` o similar). En este último caso, si existen varias soluciones óptimas, cualquiera de ellas será considerada correcta.
