Paradigmas de Programación

"Ardillas"

Prueba práctica competitiva

Enunciado

En esta ocasión tendremos dos opciones para realizar la práctica competitiva: una **opción A** (con un valor de 0,4 puntos) y otra **opción B** (con un valor máximo de 1,8 puntos). **Solo se puede elegir una opción**. A la **opción A** solo se puede concurrir de forma individual; pero la **opción B** admite grupos de hasta 3 personas. En la **opción A** hay que entregar un archivo de nombre `squirrel_jumps.ml`, mientras que en la **opción B** el archivo entregado debe llamarse `squirrel_dist.ml`.

Opción A

Se trata de optimizar la función

```
shortest_tour: int -> int -> (int * int) list -> int -> (int * int) list
```

del **ejercicio 2 de la práctica 10**, para que sea lo más rápida posible en su ejecución. Como referencia aproximada puede pensarse en su aplicación a tableros de tamaño 200 x 200 con una lista aleatoria de 10.000 árboles: en un equipo portátil típico actual debería tardar menos de 1 segundo en encontrar la solución. Las propuestas se comprobarán con una serie de tableros de prueba. De entre las 30 primeras propuestas recibidas, **las 15 más rápidas** serán calificadas con **0,4 puntos** siempre que alcancen el nivel de optimización requerido, sean **originales**, estén implementadas sin salirse del **paradigma funcional** y el código venga convenientemente comentado para su comprensión.

La implementación se escribirá en un archivo `squirrel_jumps.ml` que debe compilar correctamente con la interfaz `squirrel_jumps.mli`, suministrada con este enunciado (`ocamlc -c squirrel_jumps.mli squirrel_jumps.ml`). La primera línea del archivo `squirrel_jumps.ml` debe contener, como comentario, el nombre completo del autor.

Opción B

En el **ejercicio 2 de la práctica 10**, se pide la definición de una función

```
shortest_tour: int -> int -> (int * int) list -> int -> (int * int) list
```

que encuentre un camino para la ardilla a través de los árboles, que minimice el número de saltos realizados. Se trata ahora de modificar la definición de esta función de modo que se **minimice la distancia recorrida a lo largo del camino**. Como en las pruebas prácticas anteriores, la implementación **debe ceñirse al paradigma funcional** (evitando el uso de variables, vectores y bucles).

Debe procurarse, asimismo, en la medida de lo posible optimizar el código para una ejecución lo más rápida posible y evitar, también en la medida de lo posible, problemas con la pila de recursividad (que pudiesen provocar errores de "*stack overflow*"). Tenga en cuenta que las implementaciones proporcionadas serán probadas con tableros relativamente grandes y con un número de árboles también posiblemente grande (estaría bien, por ejemplo que fuese "capaz" de resolver el problema sobre mapas de hasta 600 filas y 600 columnas, con hasta 100.000 árboles).

Se sugiere basar la implementación en el **algoritmo de Dijkstra** para encontrar el camino más corto entre dos nodos de un grafo.

La implementación se escribirá en un archivo `squirrel_dist.ml` que debe compilar correctamente con el fichero de interfaz `squirrel_dist.mli`, suministrado con este enunciado (`ocamlc -c squirrel_dist.mli squirrel_dist.ml`). La primera línea del archivo `squirrel_dist.ml` debe contener, como comentario, el nombre completo de los autores.

Deben incluirse en el código los comentarios convenientes para su fácil comprensión.

Las propuestas recibidas serán chequeadas con una serie de casos de prueba y serán rechazadas aquellas que fallen en alguno de los casos.

Las implementaciones que superen los casos de prueba serán valoradas con hasta **1,8 puntos** (correspondientes al apartado de pruebas prácticas), según su originalidad, sencillez y nivel de optimización. Los plagios detectados serán rechazados.

Puede realizarse individualmente o en grupos de hasta tres personas. En el caso de grupos la puntuación que correspondiese a la implementación será repartida entre los miembros como en la primera prueba práctica competitiva.

Normas y fecha límite de entrega (ambas opciones)

La entrega se realizará en el Moodle de la asignatura. Dentro de la sección "Pruebas Prácticas", se ha creado una tarea específica para tal efecto. La tarea permanecerá abierta hasta las 20:00 horas del miércoles 21 de diciembre o hasta que se detecten **18 propuestas** correctas para la **opción B** (lo que antes ocurra). Diariamente se actualizará el número de propuestas recibidas.

Debe entregarse únicamente el archivo `squirrel_jumps.ml` o el archivo `squirrel_dist.ml`. La entrega requiere explícitamente que se pulse el botón de "Enviar tarea", y solo se podrá realizar un envío. Es decir, una vez realizado el envío, la tarea ya no podrá editarse (no se podrá eliminar, ni se podrán realizar cambios en la misma). Sólo un miembro de cada grupo debe realizar el envío.