Asignatura: Fundamentos de algoritmia
Evaluación continua. Problema de algoritmos de vuelta atrás.

Profesor: Isabel Pita.

Enunciado: Hace algunos años que varios compañeros tenemos que repartir una serie de paquetes por diferentes casas en estas fechas. Como este año me toca elegir a mi el primero los paquetes que quiero repartir, voy a intentar obtener la mayor cantidad de propinas posibles. Para ello he estimado la propina que me darán en cada casa. Ahora voy a elegir los paquetes que me den la mayor propina posible. El tiempo que tenemos para repartir todos los paquetes es limitado, por lo que antes de elegir los paquetes, nos dan el tiempo que deberíamos tardar en llegar a cada casa desde el punto de partida. Hay que tener en cuenta que los paquetes pueden ser voluminosos, y como voy andando, solo puedo llevar un paquete cada vez. Se supone que volvemos al punto de partida por el mismo camino que utilizamos para ir a la casa, por lo que el tiempo que tardamos en volver coincide con el tiempo de llegar a la casa. Después de entregar el último paquete debemos también volver al punto de partida para notificar si hubo alguna incidencia.

Datos de entrada: Número de paquetes a repartir y tiempo máximo que tenemos para repartirlos. Propina que estimo que me darán en cada casa. Tiempo que se tarda en llegar a cada casa desde el punto de partida.

Datos de salida: Obtener una lista de los paquetes que voy a repartir, es decir, los paquetes que maximizan la propina, sin que se supere el tiempo que tengo para distribuirlos. Si hay varias soluciones óptimas se dará una de ellas.

Se pide:

1. Implementa un programa que resuelva el ejercicio propuesto.

2. Debéis crear vosotros los casos de prueba ya que no se va a utilizar el juez para ello. Crea varios casos de prueba *pequeños* para probar tu programa.

3. La solución del problema debe incluir una solución inicial que permita inicializar la solución mejor. Diseña un caso de prueba *grande* que permita ver la diferencia del tiempo de ejecución cuando se utiliza la inicialización de la solución respecto a cuando se inicializa la solución mejor al máximo/mínimo del tipo. Para ello debes pensar un caso de prueba con el cual la ejecución del programa utilizando la solución inicial consiga podar muchas ramas del árbol de ejecución. Mide el tiempo que tarda el programa en un caso y en otro.

4. La solución del problema debe incluir una estimación del coste de la solución que queda por construir que permita podar adecuadamente el árbol de exploración. Diseña un caso de prueba *grande* que permita ver la diferencia del tiempo de ejecución cuando se utiliza la estimación respecto a cuando no se utiliza. Mide el tiempo que tarda el programa cuando se utiliza la estimación y cuando no se utiliza.

5. Haz un vídeo en el que se explique el árbol de exploración con el que se ha desarrollado el programa y el vector solución que se va a utilizar. El vídeo debe incluir una explicación de la solución inicial que se ha empleado y de la estimación realizada. Se comentarán los tiempos que se han tomado en cada caso. Al principio del vídeo debe salir el alumno para que se vea que lo está realizando él y mostrar su DNI. Los vídeos se destruirán una vez corregidos y no serán empleados con otro fin que evaluar la prueba de vuelta atrás.

6. Debes subir al campus a la tarea abierta para ello el fichero .cpp con la implementación del programa, dos ficheros con los casos de prueba, uno de ellos con los casos de prueba *pequeños* y el otro con el caso de prueba *grande*. Si se necesitan varios casos de prueba grandes se pueden subir varios ficheros uno con cada caso. El link del vídeo realizado se pondrá al comienzo del archivo .cpp con el programa implementado junto con el nombre del alumno. Si el vídeo es pequeño se puede subir directamente al campus en lugar de utilizar la herramienta youtube para compartirlo.

7. Se valorarán los comentarios incluidos en la solución, asi como las decisiones tomadas respecto a los aspectos que no están completamente cerrados en el enunciado.