Universidad de Burgos
Departamento de Ingeniería Informática
Lenguajes y Sistemas Informáticos

Algoritmia
3º Grado en Ingeniería Informática
Programación dinámica

Práctica Obligatoria 3
Programación dinámica

Créditos: 0.6 (15 horas); **Peso: 10 %**

Tras la resolución de los problemas anteriores con gran maestría y dedicación, tu fama como programador ha llegado a la empresa Datos Perdidos S.L. Esta empresa se dedica a la recuperación de datos en dispositivos defectuosos, te plantea desarrollar una nueva funcionalidad que creen que sería de utilidad para buscar cadenas de datos utilizando palabras clave que el cliente pueda recordar en una base de datos corrupta.

Del mismo modo, el Ministerio de Transporte, Movilidad y Agenda Urbana, al estar muy contento con el servicio anterior, quiere plantearte un nuevo problema para coordinar las rutas de autobús: encontrar la ruta menos costosa para todos los viajes, teniendo en cuenta que el origen y el destino puede ser cualquier comarca.

1. Problemas a resolver

1.1. Recuperación de datos

Datos Perdidos S.L. tiene acceso a cadenas de texto donde los datos que se quieren recuperar se mezclan con información ruidosa. Quieren que desarrolles un algoritmo capaz de buscar que información conocida de antemano (Ej.: nombres, número de teléfono. Etc.) se encuentra en la base de datos corrupta. Adicionalmente, se pide que este algoritmo permita ignorar un carácter seleccionable por el usuario como espacios en blanco.

El algoritmo a implementar debe ser eficiente, por lo tanto, el uso de un método de fuerza bruta debe ser descartado.

1.2. Calcular todas las rutas entre comarcas

Dado un grafo dirigido de comarcas junto con el coste del viaje entre cada una de ellas se pide:

1. Diseñar un algoritmo que calcule y almacene todas las rutas que deberíamos elegir si quisiéramos de ir de cualquier comarca a cualquier otra.
2. Ser capaz de indicar para cualquier par de comarcas, origen y destino, cuál es la ruta menos costosa.
3. Ser capaz de indicar el coste de la mejor ruta entre cualesquier 2 comarcas.

Como Ministerio de Transporte va a necesitar consultar constantemente las rutas que se realizan, se pide que, para evitar el tiempo de cómputo, se almacenen todas ellas en memoria y proporcione una forma de consultar las rutas y sus costes.

2. Informe

Junto con la solución planteada se pide que se respondan a las siguientes preguntan que se plantean en el *Notebook* de la práctica:

1. Diga la complejidad temporal y espacial de los algoritmos que se pidan.
2. Responda a las preguntas:
   - Respecto a la recuperación de datos:
     - Describa brevemente como sería un método de fuerza bruta (que no se debería de haber implementado en la práctica por su ineficiencia) para resolver este problema.
     - ¿Cómo sería de utilidad el algoritmo implementado en un software de control de versiones como git, subversion, bazaar...?

- Respecto a calcular todas las rutas entre comarcas:
  - ¿La solución es óptima (minimiza siempre el valor pedido) o es aproximada (encuentra un mínimo local)?
  - Existiendo el camino a->b con coste 10 y el camino óptimo a->c->b con coste total de 9, describe como tu algoritmo encuentra este segundo.

3. Rúbrica

Si no se cumple con los requisitos, con el enunciado, se utilizan librerías no nativas de *Python* o no se presenta informe, se tendrá una puntación de cero en toda la práctica.

No se pueden cambiar las cabeceras de las clases y funciones del *Notebook* provisto, pero sí se puede y se recomienda crear funciones y clases adicionales para facilitar el buen diseño de la solución.

3.1. Código 6/10

- Código repetido, inútil o no general: -0.25 puntos por cada ocurrencia.

3.1.1. Recuperación de datos (50 %)

- Funcionalidad correcta (supera todos los tests): 7 puntos
- Complejidad temporal óptima: 2 puntos
- Complejidad espacial óptima: 1 punto

3.1.2. Rutas entre comarcas (50 %)

- Funcionalidad correcta (supera todos los test): 7 puntos
- Complejidad temporal óptima: 1.5 puntos
- Complejidad espacial óptima: 1.5 puntos

3.2. Informe 4/10

- Análisis de complejidad temporal y espacial: 60 %
- Preguntas: 40 %

4. Entrega

La entrega de la práctica consistirá en un fichero .ipynb con el *Notebook* con el código y las preguntas respondidas.

Los ficheros deberán tener el siguiente nombre:
`<ApellidosPrimerAlumno>_<ApellidosSegundoAlumno>_pdinam.ipynb`
Así, si los alumnos fueran “José Luis Garrido Labrador” e “Ismael Ramos Pérez”, la entrega sería:

- GarridoLabrador_RamosPerez_pdinam.ipynb

Nota importante 1: si el documento no tiene este formato de nombre, la práctica entera tendrá una penalización de **dos puntos** sobre el total.

Nota importante 2: en caso de que se entregue por parejas **ambas personas** deberán hacer la entrega y deberán entregar **el mismo fichero**, si uno de la pareja no hace la entrega se evaluará como ”No presentado”. En caso de que estos ficheros sean diferentes (bajo verificación por SHA256 y manual) se tendrá la penalización de **dos puntos** y se tendrá en cuenta cada práctica por separado.

Nota importante 3: en el caso de que se suba un fichero que no es un ***Jupyter Notebook***, **esté corrupto** o **tenga fallos de sintaxis**, alguna parte del código, la práctica tendrá una nota de 0. Por tanto, verificar que la entrega ha sido correcta. Tampoco se permite en el código llamadas a funciones del sistema, de tener alguna de estas llamadas la práctica tendrá también una nota de 0.