UNIVERSIDAD SIMÓN BOLÍVAR
Dept. Computación y Tecnología de la Información
Laboratorio de Algoritmos y Estructuras II
Abr-Jul 2016

Proyecto II
El misterio del robo de DACE

Versión 1.0

Los empleados de DACE han atravesado por días difíciles dado a los lamentables hechos acaecidos durante la semana. Los estudiantes del Laboratorio de Algoritmos y Estructuras II voluntariamente ayudarán a los trabajadores de DACE a ordenar toda la información perdida para así evitar una tragedia: tener que cursar todo nuevamente.

Los papeles encontrados en el suelo eran documentos y pagos respectivos de dichos documentos. Al momento de leerlos algunos se encontraban con modificaciones en los nombres de las personas. Mientras los estudiantes ordenaban los papeles se dieron cuenta que entre ellos se encontraban unas pistas que los llevaría a descubrir el posible nombre del bandido que se robó las computadoras.

Pese a que el bandido sigue suelto, DACE debe seguir trabajando. Todos los días las personas hacen una cola para retirar documentos. Las personas que al momento de llegar a la taquilla no hayan pagado el documento que solicitaron no lo recibirán.

Como siguen las investigaciones del misterioso caso, los trabajadores de DACE se mantendrán atentos para visualizar si el nombre de las personas que se encuentren en la cola para retirar documentos se parece a la palabra que se reveló en las pistas.

¿Cuáles serán los sospechosos entre las personas que hacen la cola de DACE?

1. Documentos y Pagos

Los documentos poseerán la siguiente información: nombre y tipo de documento. Los nombres de los documentos serán únicos mientras que el tipo de documento podrá verse repetido en varios casos.

Los pagos poseerán la misma información que los documentos. La cantidad no necesariamente deberá coincidir con la cantidad de documentos.

Tanto los documentos como los pagos deberán estar guardados en dos Tablas de Hash diferentes. La clave para poder guardarlos será la primera letra del nombre de la persona. Cada nombre tendrá una lista de documentos o pagos, dependiendo de la tabla, correspondientes a dicha persona.

La técnica que deberá implementar para la resolución de colisiones es el *Hashing Cuco* el cual utiliza dos funciones de hash en vez de una.

Los documentos y pagos podrían tener modificaciones en los nombres de las personas por lo que se deberá comprobar si el nombre se encuentra correctamente escrito. Correctamente escrito se define como (i) es de al menos 3 caracteres de longitud, (ii) no deberá tener dígitos o caracteres especiales (iii) deberá empezar en mayúscula.

En caso de que el nombre tenga una modificación usted lo deberá corregir de la siguiente manera: (i) si no posee 3 caracteres de longitud inmediatamente ese documento o pago no será válido y en consecuencia no se colocará en la Tabla de Hash, (ii) si posee dígitos o caracteres especiales deberá removerlos del nombre y formar el nombre sin caracteres especiales o dígitos (iii) en caso de no empezar por mayúscula deberá reemplazar la minúscula por la mayúscula.

2. Pistas

Las pistas serán una cadena de caracteres que cuando se junten deberán de ser descifrados para poder hallar el posible nombre del bandido y tendrán un orden. Para poder descifrar el mensaje deberá de transformar la expresión con paréntesis en forma de Notación Polaca inversa.

La Notación Polaca Inversa consiste es una notación matemática en la que cada operador sigue todos sus operandos. También se conoce como notación posfija y no necesita ningún paréntesis. En la notación polaca inversa los operadores siguen a sus operandos. Por ejemplo, sumar 3 y 4, se podría escribir "3 4 +.[en] lugar de "3 + 4". Si hay varias operaciones, el operador se coloca inmediatamente después de su segundo operando; por lo que la expresión escrita "3-4 + 5.[en] la notación convencional, en la Notación Polaca Inversa se escribiría "3 4 - 5 +": 4 se resta a 3 y luego se suma 5 [2].

En este proyecto, los operadores de dos argumentos serán letras: r, e, l, s (prioridad desde el más bajo al más alto donde r es menor prioridad y s mayor prioridad), entre paréntesis ( ). Los operandos serán el resto de las letras del abecedario excluyendo las letras que son operadores. Supongamos que sólo hay una forma de Notación Polaca Inversa (no hay expresiones como a e u r c). Un ejemplo de dos pistas sería:

Pista 1: (ae

Pista 2: (nrd))
Pista Completa: (ae(nrd))
Pista Descifrada: andre

No necesarimente la palabra tiene que tener sentido. Un ejemplo de esto sería:

Pista 1: ((ce
Pista 2: a)r(t
Pista 3: e
Pista 4: h))
Pista Completa: ((cea)r(teh))
Pista Descifrada: caether

Para resolver el problema de las pistas deberá usar una pila. Cuando la pista ya esté descifrada, ésta será una palabra similar al nombre del bandido. El criterio de similaridad por el cual se guiará será la *Distancia de Levenshtein*. La *Distancia de Levenshtein* es el número mínimo de operaciones requeridas para transformar una cadena de caracteres en otra. Se entiende por operación, bien una inserción, eliminación o la sustitución de un carácter.

Por ejemplo, la distancia de Levenshtein entre 'casa' y 'calle' es de 3 porque se necesitan al menos tres ediciones elementales para cambiar uno en el otro.

casa → cala (sustitución de 's' por 'l')
cala → calla (inserción de 'l' ente 'l' y 'a')
calla → calle (sustitución de 'a' por 'e') [1]

Aquellos nombres que estén en la cola con menor distancia con respecto a la palabra descrifrada son potenciales sospechosos.

No se podrá usar la función *dict()* para dicha implementación.

3. Entrada

La entrada del programa serán dos archivos. Dichos archivos deberán ser argumentos de la línea de comando pasados al script:

python3 proyecto.py archivos.txt cola.txt

El primer archivo contendrá los documentos, pagos y pistas. En la primera línea se encontrará el número de pistas que se encuentran en ese archivo. En las línes restantes podrá haber un documento

con su nombre y el tipo de documento, un pago con su nombre y el tipo de documento o una pista con el número de la pista y un string. La primera columna de esas líneas tendrá la cadena doc, pista o pago. La segunda columna si es un documento o un pago tendrá un nombre y si es una pista tendrá un número. La tercera columna tendrá en el caso del documento y los pagos el tipo de documento y en el caso de las pistas la cadena de caracteres que posea esa pista. Un ejemplo de cómo estaría estructurado un archivo de entrada sería el siguiente:

npistas
doc nombre1 tipodoc
pista numpista1 string
doc nombre2 tipodoc
doc nombre1 tipodoc
pag nombre1 tipodoc
pista numpista2 string

El segundo archivo contendrá la cola de las personas que desean retirar un documento. Cada línea del archivo es el nombre de una persona. Un ejemplo de cómo estaría estructurado un archivo de entrada sería el siguiente:

nombre1
nombre2
nombre3

4. Salida

La salida deberá verse reflejada en un archivo. En la primera línea del archivo se encontrará el posible nombre del ladrón decodificado y los sosprechosos que se encuentran en la cola. La segunda línea estará en blanco. Desde la tercera línea, la primera columna tendrá todas aquellas personas que pudieron obtener sus documentos dado a que pagaron (en el orden en el que fueron atendidos) y en la segunda columna el nombre del respectivo documento. Luego se colocará una línea en blanco y se colocarán en la primera columna el nombre de todos aquellos que no se les pudo dar el documento por falta de pago (en el orden en el que fueron atendidos) y en la segunda columna se colocará el tipo de documento. En ambos casos se deberá imprimir con el nombre bien escrito.

Un ejemplo de como estaría estructurado un archivo de salida sería el siguiente:

nombredescifrado nombresospechoso1 nombresospechoso2 nombre1 tipodoc

nombre2 tipodoc
nombre3 tipodoc
nombre4 tipodoc

nombre5 tipodoc
nombre6 tipodoc
nombre1 tipodoc

5. Informe

- Portada, la cual debe contener:
  - Arriba a la izquierda: el nombre de la universidad, la carrera y la asignatura.
  - Centrado: el nombre del proyecto.
  - Abajo a la izquierda: nombre del profesor del laboratorio.
  - Abajo a la derecha: nombre y carné de los integrantes del equipo.
- Introducción:
  - Breve descripción del problema atacado en el proyecto.
  - Descripción del contenido del resto del informe.
- Diseño:
  - Decisiones de diseño tomadas por Ud. en relación con cualquier aspecto del proyecto, que Ud. considere conveniente explicar en el informe.
- Estado actual:
  - Operatividad del programa: señalar si funciona perfectamente o no; en caso negativo, describir las anomalías.
  - Manual de operación: nombre de los archivos correspondientes a todas las clases que conforman el programa, nombre del archivo a ejecutar (esto es: la clase con el programa principal), y modo de operar el programa.
- Conclusiones:
  - Experiencias adquiridas.
  - Dificultades encontradas durante el desarrollo.
  - Recomendaciones.
- Bibliografía: libros, revistas o cualquier otro recurso consultado (que puede incluso ser conversaciones o consultas con amigos o expertos).

  Una referencia bibliográfica debe incluir la siguiente información:
  - Libro: autores, nombre, editorial y año.
  - Artículo: autores, nombre, revista, volumen, número y año.
  - Página web: autores, nombre, URL, fecha de la última visita.
  - Conversaciones personales o consultas: interlocutor y fecha.

6. Comentarios Finales

- Cualquier error que a posteriori sea hallado en este enunciado, así como cualquier otro tipo de observación adicional sobre el desarrollo del proyecto, serán publicados como fe de erratas en el Aula Virtual.
- El proyecto es un trabajo en equipo de dos personas. Si bien lleva dos semanas de trabajo, se considera similar a un examen en cuanto a que no puede haber intercambios contrarios a la ética con otros equipos.
- El proyecto que no corra o no esté bien comprimido no obtendrá ningún puntaje.
- Tanto las entradas como las salidas deben de poseer el formato planteado en el enunciado.
- Toda la implementación será realizada en el lenguaje de programación Python3.
- Deberá hacer uso de colas o colas de prioridad, pilas, listas y tablas de hash. En caso de no hacer uso de alguno de estos tipos de datos será penalizado su proyecto.
- Todas las operaciones que lo requieran deben hacer uso de excepciones.
- Deberá implementar su proyecto haciendo uso de POO(Programación Orientada a Objetos)
- Las estructuras de datos deberán ser implementadas por usted. El hacer uso de funciones predeterminadas de Python como estructuras de datos será penalizado en el proyecto.

Entrega Final - sábado 2 de julio de 2016

Cree un archivo comprimido del tipo "tgz" llamado ProyII-X-Y.tgz, donde X es su número de carné y Y el de su compañero. Su contenido serán los archivos usados para la implementación de su proyecto y un informe que describa el proceso de desarrollo y las preguntas adicionales que fueron dejadas en este enunciado. Deberá entregar su proyecto por el Aula Virtual hasta el día sábado de la semana 12 a las 10:00 am.

Referencias

[1] *Distancia de Levenshtein.* `https://es.wikipedia.org/wiki/Distancia_de_Levenshtein`

[2] *Notación Polaca Inversa.* `https://es.wikipedia.org/wiki/Notación_polaca_inversa`