Laboratorio 1 - Hash

Leguajes de programación
Universidad Andrés Bello

2 de abril de 2020

1. Tablas hash

Una tabla hash es una estructura de datos muy eficiente para realizar búsqueda de datos. Su funcionamiento consiste en asignar una clave calculada a cada elemento, llamada hash, obtenida a través de una función llamada función de hash.

Una función de hash evitará producir una misma clave para dos elementos distintos. Cuando esto ocurre, a este comportamiento se le conoce como colisión, sin embargo, evitar colisiones suele requerir mayores esfuerzos y puede ser computacionalmente costoso.

Las tablas hash tienen diversas aplicaciones, entre las cuales destacan su utilización en algoritmos de búsqueda para determinar rutas ya recorridas, sistemas de corrección ortográfica y sistemas de clasificación de colecciones como bibliotecas o supermercados.

2. Tarea

En esta tarea, a usted se le solicita implementar una tabla hash simplificada utilizando arreglos, la cual contendrá las palabras existentes en un párrafo de texto, y será útil para determinar fácilmente si la palabra existe en el texto y conocer cuan repetida está.

Por cada palabra w de largo l, usted deberá calcular una función de hashing $hash(w)$ la cual debe estar definida por la siguiente formula, donde $w[0]$ corresponde al valor en ascii de la primera letra de la palabra w y k es un valor arbitrario definido por el usuario en el archivo de entrada.

$$hash(w, k) = w[0] * k^0 + w[1] * k^1 + w[2] * k^2 + ... + w[l] * k^l$$

Luego, en un arreglo A de tamaño 1.000.000 y de tipo `int`, deberá almacenar en la posición $hash(s)$ un contador que indique la cantidad de veces que esa palabra se encuentra en el texto.

Por lo tanto, como entrada su programa recibirá un archivo que tiene en su primera linea el valor del exponente k utilizado para calcular el hash, en la segunda linea contendrá

un párrafo de texto que corresponderá al texto de entrada, en la tercera linea contendrá la cantidad de palabras $|Q|$ por las que se pregunta; y en las lineas siguientes, desde la linea 4 hasta la n, contendrá las palabras a preguntar $Q_0, ..., Q_n$.

Como salida, su programa debe escribir un archivo llamado `salida.txt` el cual tendrá $|Q|$ lineas, cada una de las cuales contendrá primero el hash correspondiente a la palabra y luego, separado por un espacio, la cantidad de veces que esta palabra aparece en el texto

3. Ejemplo de archivo entrada.txt

En este ejemplo, se define $k = 3$ y se pregunta por 6 palabras ($|Q| = 6$)

```
3
It is a long established fact that a reader will be distracted by the
readable content of a page when looking at its layout. The point of
using Lorem Ipsum is that it has a more-or-less normal distribution
of letters, as opposed to using Content here, content here, making
it look like readable English.
6
or
a
Content
content
hola
that
```

4. Ejemplo de archivo de salida.txt

Cada linea tiene el hash de la palabra a preguntar. En el ejemplo $hash(or, 3) = 453$, $hash(a, 3) = 97$, etc

```
453 1
97 3
123997 1
124029 1
4028 0
4433 2
```

5. Instrucciones

- Fecha de entrega: Sábado 11 de abril, 2020 a las 23:59.
- Método de entrega: Su repositorio privado creado a través del link de la tarea

- Para comenzar su tarea, clone su repositorio y utilice el archivo `main.c` con código pre hecho. Ese código le servirá para separar las palabras existentes en el archivo de entrada. Puede editarlo con libertad.
- Su repositorio de la tarea solo debe contener un archivo el cual contendrá su programa, y este será llamado `main.c`. No incluya los archivos de entrada o salida en su repositorio. Utilice un archivo `.gitgnore` para no subirlo a su repositorio remoto.
- Su programa debe recibir y generar los archivos entrada.txt y salida.txt en la misma ubicación en la que se encuentra el programa ejecutable generado por la compilación.
- Solo puede utilizar bibliotecas estandar. Verifique que su programa compila al ejecutar el comando `gcc main.c -o main.o`.
- El proceso de revisión será automatizado. Es importante respetar el formato establecido para los archivos de entrada y salida. Este formato no es modificable. Por lo tanto, si su archivo de salida no corresponde al formato preestablecido, su calificación será deficiente.
- Las preguntas sobre la tarea deben ser formuladas como un Issue en el repositorio del laboratorio ubicado en el siguiente link `https://github.com/INS125/Laboratorio/issues`

6. Recomendaciones

- Se recomienda hacer avances parciales. Si su archivo de salida no contiene la cantidad de veces que la palabra se muestra en el texto pero si contiene el hash por cada palabra, puede obtener puntaje parcial.
- Si su programa no crea el archivo de salida o no compila, será evaluado con la nota mínima.
- En el repositorio oficial del laboratorio puede encontrar dos ejemplos de archivos de entrada y su correspondiente salida. `https://github.com/INS125/Laboratorio/`
- Recuerde solicitar unirse a github student. Si no lo hace, no podrá hacer su código privado y cualquiera podría visualizar su tarea. Es su responsabilidad cuidar su tarea.

7. Código de honor

Toda persona inscrita en este curso se compromete a:

- Actuar con honestidad, rectitud y buena fe frente a sus profesores y compañeros.
- No presentar trabajos o citas de otras personas como propias o sin su correspondiente citación, ya sea de algún compañero, libro o extraídos de internet como también a no reutilizar trabajos presentados en semestres anteriores como trabajos originales.

- No copiar a compañeros ni hacer uso de ayudas o comunicaciones fuera de lo permitido durante las evaluaciones.

Cualquier alumno o alumna que no respete el código de honor durante una evaluación (sea este la entrega de una tarea o el desarrollo de una prueba o control tanto durante la cátedra como el laboratorio) será evaluado con la nota mínima y será virtud de profesor, de acuerdo con la gravedad de la falta, las acciones siguientes a tomar.