Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2133 — Estructuras de Datos y Algoritmos
2022 - 1

Tarea 2

Fecha de entrega código e informe: 2 de Junio del 2022

Objetivos

- Modelar un problema y Solucionarlo de forma eficiente aplicando estructuras de datos
- Analizar distintos algoritmos de hashing y sus propiedades incrementales.
- Aplicar Tablas de Hash a un problema NP-Completo.

Stuart is Sus

Mientras estabas construyendo el DCCRayo de Fotones, mas y mas gente afectada por el frivolo Stuart Little se ha unido a la organización. Ahora llamada "La Orden del Raton Fenix", colaborando en distintas areas para dar fin al reinado del terror de la rata rompe familias. Sin embargo recibiste reportes de que el malvado Little también ha creado su propia organización, y que tienen miembros infiltrados en las mas altas esferas del poder, desde gobiernos hasta incluso la misma Orden.
Entre algunos de sus peligrosos secuaces se encuentran personajes detestables como Scrapy Doo, Sid, Team Rocket, Ademas del temible ~~Winnie the Poo~~ **REDACTED**. Para prevenir que saboteen las operaciones de la Orden, se propuso que diseñaras un sistema computacional con capacidades de detectar y alertar ante el caso de presencia de uno de estos individuos.
Las camaras envian solo una gran imagen de todas las instalaciones de la Orden

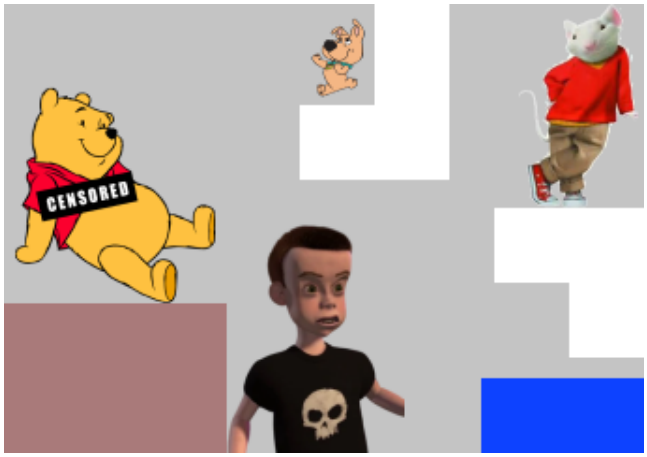

Figura 1: Secuaces de Stuart Little

Lamentablemente, Las camaras son de muy, muy, muy, muy baja resolucion, por lo tanto una imagen mas realista de esta situacion seria la siguiente

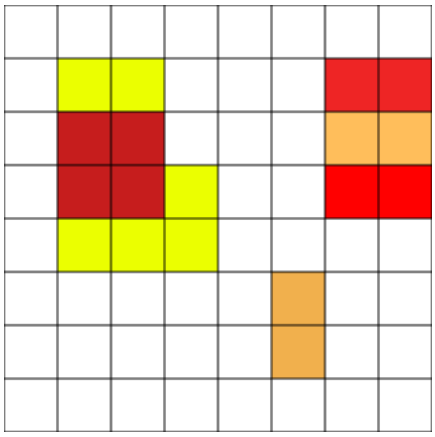

Figura 2: Mapa de Valores capturado

Sin embargo existe otro problema, las camaras son muy lentas. Por lo que pueden capturar distintas instancias de uno de los infiltrados. Por otra parte las cámaras no solo capturan la imagen de los infiltrados, si no que también existirán píxeles asociados a miembros reales de la organizacion

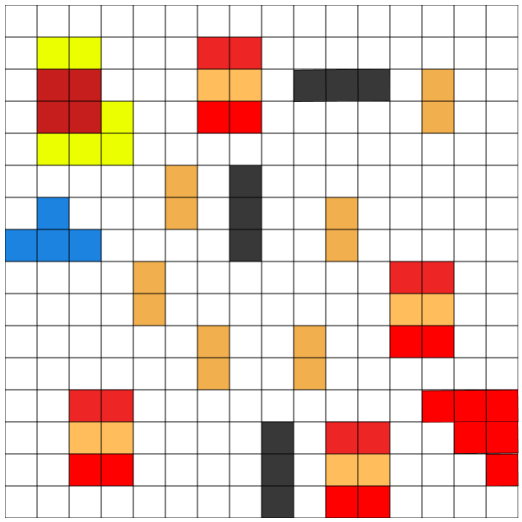

Figura 3: Imagen entregada realmente por la camara

Finalmente, Tu labor sera detectar si alguno de los miembros infiltrados esta presente o no y además contar el numero de ocurrencias.

Problema

Para que resuelvas el problema se entregara un archivo con el mapa base de tamaño $N \times N$ y otro archivo con K imagenes de distintos tamaños. la idea es que retornes en otro archivo si es que uno de estos miembros se encuentra presente e indicar cuantas veces se encuentra en el mapa original.

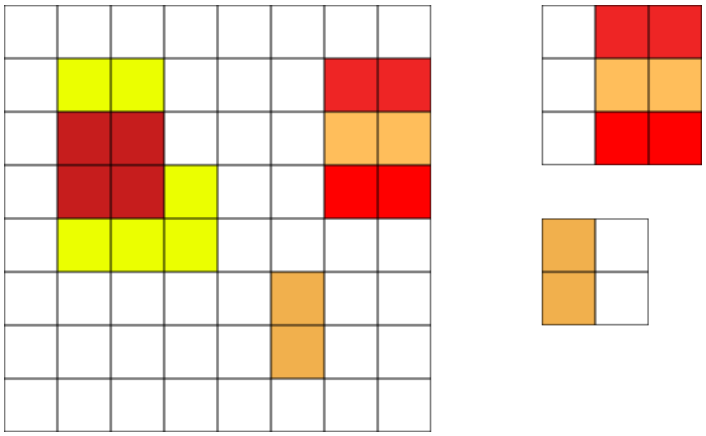

Figura 4: Mapa y ejemplos de busqueda

Sin embargo ahora viene el Plot Twist: Tu gran amigo Pinky, era un traidor. Por lo que el programa que generaste para hacer búsquedas en bodega fue alterado considerablemente y se transformo en un peligroso virus que dejo inaccesibles casi todos los recursos computacionales de tu organización.
Es por esto que el acercamiento de fuerza bruta a este problema es imposible de realizar (ie. revisar en cada consulta todas las combinaciones posibles). Por lo que te enfrentas a un problema de que la busqueda debe ser extremadamente eficiente. Sin embargo en un momento de gran revelacion, recordaste las tablas de hash vistas en clases.

Que deberas hacer

Deberás escribir un programa en **C** que, dada una imagen y un conjunto de `Q` sub-imágenes de sospechosos, Retornara el numero de ocurrencias de cada una de ellas o 0 en caso de que no este presente. Para poder realizar esto de forma eficiente **Deberás** utilizar una tabla de hash, la no utilización de una significara incurrir una calificacion maxima de 4.5 .
Además de la tabla de hash, tendrás que realizar una función de hash facil de calcular y que sea eficiente para el problema presentado. Se recomienda fuertemente diseñar una función de hash que se pueda calcular de forma incremental. Es muy importante que trates de diseñar una solución muy eficiente ya que es parte de la evaluación (Ver sección evaluación)

Aclaraciones

- Todas las imágenes serán de tamaño cuadrado
- Todas las imágenes de búsqueda poseerán un patrón que contiene al menos un píxel no blanco
- Los pixeles seran representados en un rango desde 0 a 255. Con 0 indicando que la celda se encuentra vacia (Blanco en los dibujos anteriores)

Ejecución

Tu programa se debe compilar con el comando `make` y debe generar un binario de nombre `littlefriend` que se ejecuta con el siguiente comando:

```
./littlefriend input.txt querys.txt output.txt
```

Donde `input.txt` Es el archivo que contiene el mapa inicial, `querys.txt` es el archivo que contiene las consultas y output es donde deberas escribir el output del programa

Input

El input está estructurado como sigue:

- Una linea con el valor N que indica la dimension de la grilla (Siendo la grilla entonces de $N \times N$ celdas)
- N lineas donde cada una indica una fila de la imagen. Donde cada fila contiene N enteros entre 0 y 255.

Por ejemplo, El siguiente input seria valido

```
5
0   0   1   1   245
40  0   5   0   15
0   10  97  0   1
40  0   20  5   0
40  0   0   0   0
```

Queries

Las Queries estan estructuradas de la siguiente forma

- Una linea Q que indica la cantidad de subimagenes a buscar
- Q entradas de subimagenes con el mismo formato de entrada que el input

Por ejemplo, El siguiente input seria valido (los # son *comentarios*)

```
2  # Dos consultas
3  # Dimensión primera subimagen
0   0   1
0   0   2
0   1   2
4  # Dimensión segunda subimagen
0   0   4   0
0   0   2   0
0   1   2   4
0   0   2   0
```

Output

Tu output debera consistir de Q lineas, donde cada una indica el numero de matches de la subimagen respectiva.
Por ejemplo, en un caso con $Q = 4$ (los # son *comentarios*)

```
3 # Primera subimagen tiene 3 coincidencias
0 # Segunda subimagen no esta presente
1
2
```

Codigo Base

No habra codigo base como tal, solo se entregara un makefile y un archivo main.c, Es tu deber encargarte de la lectura del archivo y de procesar los inputs entregados.

Evaluación

La evaluación sera solo código en C. No existirá informe ni cuestionario asociado. Sin embargo para poder evaluar la eficiencia de las modelaciones se agregaron un par de consideraciones adicionales a la nota.

Código

Tu código será evaluado con tests de dificultad creciente. Estos tests están separados en 3 categorías:

- Easy: Para debugear manualmente. No serán evaluados.
- Normal: Usados para testear la correctitud de tu implementación. Serán evaluados.
- Hard: Usados para testear la eficiencia de tu implementación. Serán evaluados.

Además, para garantizar su planificación de la tarea, Les aseguramos que estas seran corregidas en un procesador con frecuencia base de 3.7 GHz.

Escala

Además de solicitar que tu programa de la solución, se establecerán ciertos criterios de evaluación sobre la eficiencia de tu modelacion.
La formula para calcular el puntaje por clase de test sera la siguiente:

$$Puntaje = \frac{Q_{Solved}}{Q_{total}} * C_T$$

Donde:

- Q_{Solved} Es el numero de subimagenes con output correcto
- Q_{total} Es el numero total de subimagenes del test
- C_t es un ponderador entre 0 y 1.3 que depende del tiempo del test

Calculo de C_t Dado un test i y su tiempo T_i

- $C = 0$ Si $T_i > 10s$
- $C = 0,7$ Si $T_i \leq 10s$
- $C = 1$ Si $T_i \leq quantile(T, i, 0{,}65)$; con $quantile(T, i, 0{,}65)$ el percentil 65 de los tiempos
- $C = 1{,}3$ Si el tiempo esta en el 20 % mas rapido

Para facilitar el proceso es importante que usen el servidor ya que se publicaran leaderboards parciales. Ademas, la idea es no perjudicarlos por lo que estas ponderaciones estan sujetas a cambios que los puedan beneficiar
Restricción: Para esta tarea también se evaluará el uso no excesivo de memoria. Por esto, tu tarea no debe usar más de 1 GB de RAM, de lo contrario no se asignará puntaje.

Entrega

Código: GIT - Repositorio asignado. Se entrega a más tardar el día de entrega a las 23:59 hora de Chile continental.

Bonus

Los siguientes bonus solo aplican si tu nota correspondiente es mayor o igual a 4.

Manejo de memoria perfecto: +5 % a la nota de código

Recibirás este bonus si `valgrind` reporta que tu programa no tiene ni leaks ni errores de memoria en los tests que logres resolver.

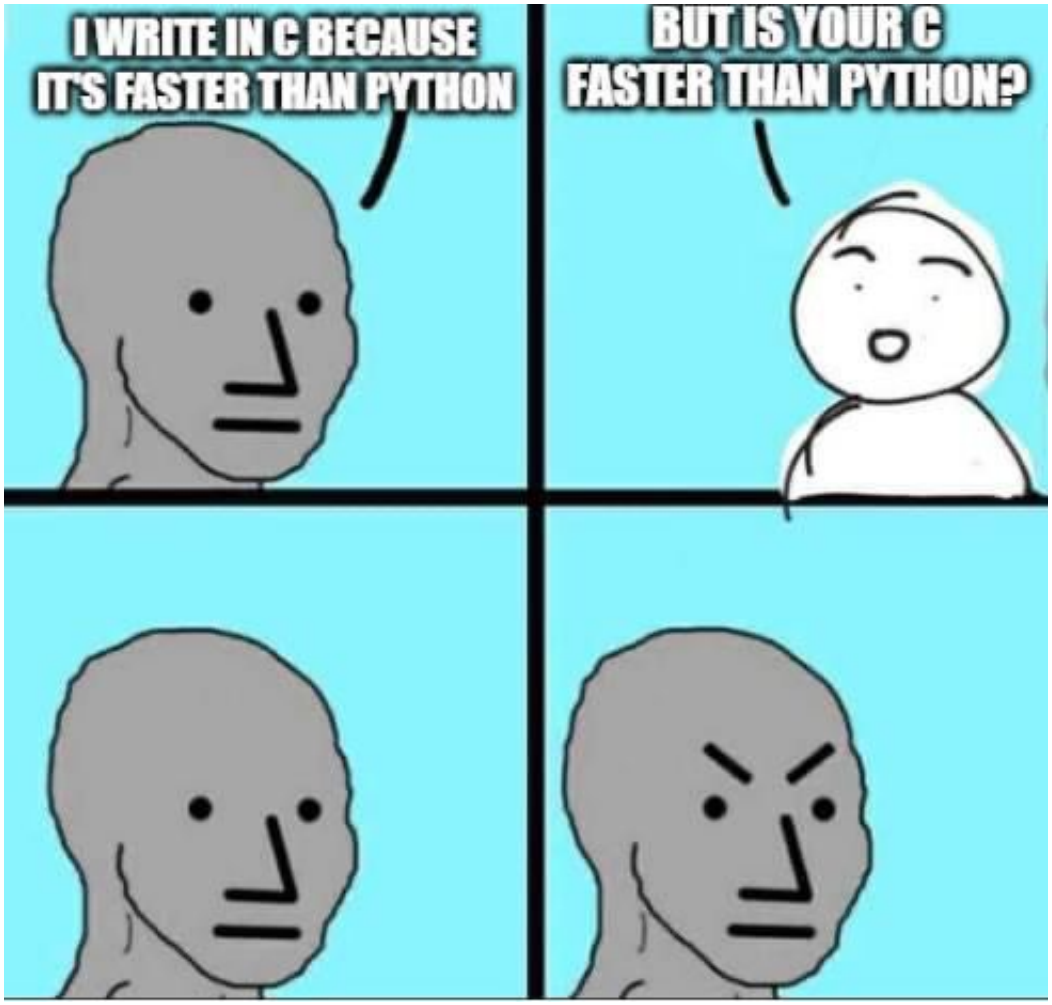

Figura 5: Meme from meme contest