PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

LENGUAJES DE PROGRAMACIÓN 1
5ta. práctica (tipo b)
(Primer Semestre 2017)

Indicaciones Generales:

- Duración: 110 minutos.
- Se podrá usar como material de consulta solo sus apuntes de clase.
- No se pueden emplear **variables globales**, **estructuras, ni objetos** (con excepción de "cin" y "cout"). Tampoco se podrán emplear las funciones malloc, realloc, strdup, strtok, sscanf, sprintf ni fopen.
- Deberá modular correctamente el proyecto en archivos independientes. El archivo **main.cpp** solo podrá contener la función **main** de cada proyecto.
- Se tomará en cuenta en la calificación el uso de comentarios relevantes.
- Los programas que presenten errores de sintaxis o de concepto se calificarán en base al 40% de puntaje de la pregunta. Los que no den resultados coherentes en base al 60%.
- **La presentación, la ortografía y la gramática de los trabajos influirá en la calificación.**

Puntaje total: 20 puntos

Cuestionario:

Las publicaciones de la red social de microblogging Twitter ("tweets") son textos cortos donde las personas pueden manifestar y expresar sus ideas u opiniones sobre cualquier tópico o tema de coyuntura. Para ello, Twitter les permite hacer uso de unas entidades particulares en los textos, entre las cuales destacan: las **menciones** a otros usuarios de la red social (inician con el caracter '**@**': "@pucp"), y la marcación de tópicos o palabras clave denominados "**hashtags**" (inician con el caracter '**#**': "#100PUCP").

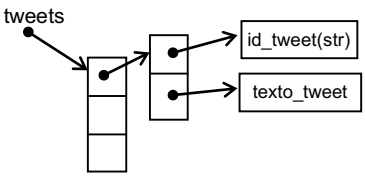
El volumen de los tweets que se generan en esta plataforma es muy alto (millones por segundo), por lo que es necesario que se adopten ciertas medidas para el almacenamiento y administración de los datos si es que se quiere optimizar tareas posteriores de búsquedas y extracción de información:

Pregunta 1 (10 puntos)

En primer lugar, se debe procesar la primera parte del archivo de pruebas recibido (hasta llegar a la línea que contiene un "0"). Cada línea posee la información de un tweet publicado, exactamente 2 campos: ID y texto de la publicación (todos los caracteres serán **ASCII**):

```
862812068025643008  100 años son solo el comienzo. #100PUCP #Se100tePUCP
862786895905869824  No te pierdas la exposición "Vivir para crear" en la @pucp
…
0
```

En base a esto, se solicita que esta información se almacene en 3 estructuras de datos para su posterior explotación. La primera es un arreglo de tweets (**`char*** tweets`**), donde por cada tweet leído se guardarán los campos del ID_Tweet y el Texto de la publicación (ambos como cadenas de caracteres).



En las otras dos estructuras, se desea construir un arreglo de los nombres de usuarios (**`char** users`**) extraídos de los textos, y de igual manera con los "hashtags" (**`char** hashtags`**). Estos dos arreglos no deben poseer elementos duplicados, y deben estar ordenadas ascendentemente por el código ASCII (queda a criterio suyo si es que las ordena en el momento de la inserción o al final de la lectura).

Como parte final de este primer pedido, debe implementar una pequeña función de prueba que imprima un reporte simple de los datos leídos (importante para la calificación).

```
Total tweets: 100
Total usuarios mencionados: 10
Lista de usuarios:
@CCULTURALPUCP @pucp …
Total hashtags: 20
Lista de hashtags:
#100PUCP #EnVivo …
```

Pregunta 2 (8 puntos)

Después de procesar el histórico de publicaciones de la primera parte del archivo, ahora Twitter necesita que se implemente unas estructuras conocidas como "vectores invertidos de documentos".

En palabras simples, se desea identificar el listado de índices de los tweets (del arreglo char*** tweets) en cuyos textos se hace mención a cada usuario (char** users) y "hashtag" (char** hashtag), y lo almacene en nuevas estructuras que les permita realizar búsquedas más rápidas sobre estas entidades: **`int** indexUsers`** y **`int** indexHashtags`**

(Se explicará solo el caso de "indexUsers", ya que el procesamiento de "indexHashtags" es similar) El tamaño del arreglo "indexUsers" será el mismo que el arreglo "users", y cada uno de sus elementos es una lista de valores enteros. Si tomamos el primer elemento (indexUsers[0]), en esa lista se guardarán los **índices** (no el ID_Tweet) del arreglo "tweets" en cuyos Textos (de uno o más tweets) aparezca el nombre de usuario almacenado en users[0]. El tamaño de estas listas debe ser exacto.

Por ejemplo, según el cuadro de la pág. 1, si el usuario en users[0] fuera "@pucp", entonces la lista de indexUsers[0] deber contener al índice "1" (posición en "tweets" donde estará el tweet que lo menciona).

Pregunta 3 (2 puntos)

Después de implementar el proceso de indexación solicitado, se solicita que realice unas pruebas. Para eso, terminará de leer la segunda parte del archivo (después del "0"), donde por cada línea encontrará o un nombre de usuario o un "hashtag".

Lo que debe realizar su función es imprimir un reporte, donde, **usando las estructuras de índices**, acceda rápidamente al contenido de los tweets donde se mencionó a dicho usuario o "hashtag". En este paso, **ya NO se debe buscar el término en el texto del tweet**, debido a que los índices nos permitirán ahora hacer más eficiente el acceso a la información textual almacenada.

```
Búsqueda 1: #100PUCP
123862812068025643008:  100 años son solo el comienzo. #100PUCP #Se100tePUCP
…
Total tweets: 4

Búsqueda 2: @pucp
862786895905869824  No te pierdas la exposición "Vivir para crear" en la @pucp
Total tweets: 1
…
```

Finalmente, y en resumen, **el programa principal** de su proyecto debe contener:

Nota: Los arreglos deberán usar **memoria dinámica** y tener **tamaños exactos** al final del procesamiento.

```
int main(void) {
  char *** tweets, ** users, ** hashtags;
  int ** indexUsers, ** indexHashtags;
  leerTweets (tweets, users, hashtags);
  reporteLectura(tweets, users, hashtags);
  generarIndices (tweets, users, hashtags, indexUsers, indexHashtags);
  buscarEnIndices(tweets, users, hashtags, indexUsers, indexHashtags);
  return 0;
}
```

Consideraciones finales:

- Cree en el computador una carpeta de trabajo con la siguiente ruta: c:\temp\Laboratorio5. En ella colocará los proyectos que den solución a los problemas planteados.
- **En cada archivo que implemente en los proyectos (.h y .cpp) deberá colocar un comentario en el que coloque claramente su nombre y código. De no hacerlo se le descontará 0.5 puntos por archivo.**

Al finalizar la práctica, comprima[1] la carpeta ***laboratorio5*** en un archivo con nombre <código del alumno con 8 dígitos>.<extensión del archivo comprimido> y súbalo a la Intranet del curso, en el enlace Documentos, en la carpeta \Laboratorio5\<código del horario>\<aula>.

Profesores del curso: Arturo Oncevay
Miguel Guanira

San Miguel, 11 de mayo del 2017.

[1] Para evitar problemas en la corrección de la prueba, utilice el programa de compresión que viene por defecto en el Windows (Zip).