Práctica PRO2

Otoño 2019

1. Introducción

El objetivo de la práctica de este cuatrimestre es desarrollar un conjunto de módulos que serían componentes esenciales de un programa de gestión de fotografías y álbumes digitales. Un programa de estas características va mucho más allá de lo que se puede hacer en la asignatura y buena parte de sus componentes no se abordan en esta práctica, p.e., visualización de las imágenes, interfície gráfica de usuario, almacenaje en la nube, compartición de imágenes y álbumes con otros usuarios y aplicaciones, módulos de edición y retoque digital, etc.

Aún así, en la práctica desarrollaréis algunas componentes y funcionalidades útiles. También habréis de desarrollar un programa que permita utilizar y comprobar el correcto funcionamiento de los módulos de esta práctica.

N.B. En las descripciones que vienen a continuación, cuando se discute un grupo de operaciones no debemos dar por sentado que sólo hay un módulo implicado, ni que todas las operaciones descritas pertenecen a un mismo módulo, ni que son las únicas que van a aparecer en los diferentes módulos. Asimismo las operaciones descritas en las secciones 2 a 5 no necesariamente dan lugar a funcionalidades del programa principal (**main**); la sección 6 detalla dichas funcionalidades.

2. Fotografías

Un elemento esencial del sistema son, obviamente, las fotografías. Cada fotografía tiene un identificador único (un string) y de ella se guarda una serie de datos adicionales:

- La fecha (dd mm aaaa) en que se ha tomó la fotografía
- Un número variable (mayor o igual que 0) de *etiquetas*. Cada etiqueta es un string y puede empezar con cualquier símbolo, pero para favorecer la legibilidad usaremos en este enunciado el prefijo '`#`' y un tipo de letra distintivo para identificarlas: p.e., una foto podría estar etiquetada con las etiquetas `#gatitos` y `#Mireia`, otra con las etiquetas `#crepe`, `#torre_Eiffel` y `#vacaciones2019`, etc. Para simplificar asumiremos que todas las etiquetas consisten en una única "palabra", es decir, están formadas exclusivamente por letras, dígitos y guiones de subrayado. No contendrán ningún espacio en blanco, separador u otros símbolos no alfanuméricos. Se distinguen mayúsculas y minúsculas, así, por ejemplo, `#mireia` $\neq$ `#Mireia` $\neq$ `#MIREIA`.

En una aplicación real guardaríamos probablemente muchos más datos, por ejemplo, el nombre y ruta del archivo de imagen, el formato del archivo de imagen, su tamaño, el modelo de la

cámara con la que se tomó, otros datos EXIF, ... pero para nuestros propósitos los datos mencionados bastarán. De igual forma se debería registrar no solo la fecha en la que se tomó la fotografía sino el instante del día en que la foto se tomó.

Las operaciones que tendremos sobre fotos incluyen:

- `crea_foto`: dado un nombre (string) y una fecha con el formato `dd mm aaaa`, crea una foto con dicho identificador y la fecha dada como fecha en la que se tomó la foto
- `agrega_etiqueta`, `elimina_etiqueta`: agrega o elimina una etiqueta dada t de la foto dada
- `contiene_etiqueta`: consultora que devuelve cierto si y sólo si la foto dada contiene la etiqueta dada
- `id_foto`: devuelve el identificador de la foto dada
- `fecha_toma`: devuelve la fecha en la que se tomó la foto dada
- `nr_etiquetas`: devuelve el número de etiquetas asignadas a la foto dada
- `imprime_foto`: escribe toda la información de la foto: identificador, fecha en la que se tomó y la lista de etiquetas, éstas en orden lexicográfico ascendente

3. Álbumes

Los álbumes son otro componente fundamental en nuestro sistema. Un álbum no es más que un conjunto de fotos. Cada álbum tiene un identificador único. Es un string cualquiera, pero cuando queramos agregar un álbum a la colección (véase la siguiente sección) el álbum no puede llamarse "`ALL`".

Como en el caso de las fotografías, en una aplicación real habría mucha más información asociada a cada álbum: descripción, fotografía de la portada, enlaces para compartir el álbum, etc. pero aquí sólo tendremos el identificador del álbum y el conjunto de fotos que lo compone.

Las operaciones que tendremos sobre álbumes incluyen:

- `agrega_foto`: agrega una foto a un álbum; devuelve un valor booleano para indicar si la operación ha tenido o no éxito: no se pueden agregar a un mismo álbum dos fotos con igual identificador (aunque difieran en otros atributos, p.e., la fecha en que fueron tomadas)
- `elimina_foto`: elimina del álbum la foto cuyo identificador se da; no hace nada si el álbum no contiene una foto con dicho identificador
- `obten_foto`: dado el identificador de una foto devuelve una indicación de si el álbum contiene o no una foto con el identificador dado, y en caso afirmativo retorna la foto en cuestión
- `nr_fotos`: devuelve el número de fotos que contiene el álbum
- `lee_album`: lee el identificador y la secuencia de fotos que componen un álbum y lo crea
- `imprime_album`: escribe el identificador y la secuencia de fotos, éstas en orden ascendente de identificador

4. Colección de Imágenes

Una *colección de imágenes* (*colección*, para abreviar) consta de 0 o más fotos agrupadas en álbumes. Notad que eso no impide que haya álbumes que no pertenezcan a la colección.

Es importante resaltar que una misma fotografía puede pertenecer a muchos álbumes y ello deberá ser tenido muy en cuenta en la fase de implementación, cuando tengáis que escoger una representación apropiada para la colección.

Además de los álbumes agregados explícitamente por el usuario, toda colección contiene un álbum especial cuyo nombre es `ALL` que contiene todas las fotos que se han agregado a la colección, bien como miembros de un álbum "normal", bien directamente por el usuario (y que no pertenecen pues a ningún álbum "normal").

Un primer bloque de operaciones que deberemos ofrecer es:

- `agrega_album`: dado un álbum (cuyo identificador es distinto de `ALL`) se agrega una copia del mismo a la colección y se sincronizan las fotos, dando prioridad a las que ya estaban en la colección; si una foto del álbum no estaba en la colección (es decir, la colección no contiene ninguna foto con el mismo identificador) dicha foto se añade a la colección; en caso contrario, es decir, si en la colección ya había una foto con igual identificador, entonces la foto que estaba en la colección pasa a formar parte del álbum agregado en sustitución de la que estaba en el original; si la colección ya contiene un álbum con el mismo identificador que el dado, entonces la operación no hace nada
- `elimina_album`: dado el identificador de un álbum, lo elimina de la colección, pero las fotos que lo formaban **no** se eliminan de la colección; no se puede eliminar el álbum `ALL`; no se hace nada si no existe ningún álbum en la colección con el identificador dado
- `contiene_album`: dado un identificador de álbum, devuelve cierto si y solo si la colección contiene un álbum con el identificador dado
- `obten_album`: dado el identificador de álbum devuelve una indicación de si la colección contiene o no un álbum con el identificador dado, y en caso afirmativo retorna el álbum en cuestión
- `nr_albumes`: devuelve el número de álbumes en la colección (incluído `ALL`)
- `lista_albumes`: escribe la lista de los identificadores de los álbumes de la colección (incluído `ALL`), en orden lexicográfico ascendente

Luego tenemos un bloque de operaciones sobre las fotos que hay en la colección (observad que, como consecuencia, no todas las fotos de una colección proceden de agregar álbumes):

- `agrega_foto`: dada una foto la agrega a la colección si no existe ninguna foto en la colección con idéntico identificador (pero no la agrega a ningún álbum); no hace nada en caso contrario
- `elimina_foto`: dado el identificador de una foto, elimina la foto de la colección (y en particular de todos los álbumes de la colección a los que dicha foto pertenecía); no hace nada si no hay ninguna foto en la colección con el identificador dado

- `modifica_foto`: dado el identificador x de una foto y una nueva foto f con identificador x, si en la colección existe una foto con identificador x la reemplaza por f (el cambio afecta a todos los álbumes de la colección a los que pertenecía dicha foto); no hace nada si no había ninguna foto con identificador x en la colección

- `nr_albumes_foto`: devuelve el número de álbumes de la colección (incluido `ALL`) a los que pertenece la foto cuyo identificador se nos da

- `lista_albumes_foto`: escribe, en orden lexicográfico ascendente, la lista de los identificadores de los álbumes de la colección (incluído `ALL`) que contienen a la foto cuyo identificador se nos da

Finalmente tendremos dos operaciones de búsqueda.

- `busca_por_fecha`: dados un rango de fechas y el identificador de un álbum contenido en la colección, el resultado de la consulta es un nuevo álbum que contiene el subconjunto de fotografías pertenecientes al álbum dado cuya fecha está dentro del rango indicado. Para obtener todas las fotos anteriores a una fecha se puede usar la fecha 1/1/1700 como límite inferior del rango, y de manera parecida, para obtener todas las fotos posteriores a una fecha se usa la fecha 31/12/2299 como límite superior del rango. El identificador del álbum sobre el que se realiza la búsqueda puede ser `ALL`. El identificador del álbum resultado se suministra como parámetro de la operación

- `busca_por_etiqueta`: Dada una etiqueta t y el identificador de un álbum contenido en la colección (que puede ser `ALL`) el resultado es un nuevo álbum que contiene el subconjunto de fotos pertenecientes al álbum dado que contienen la etiqueta t. El identificador del álbum resultado se suministra como parámetro de la operación

5. Consultas booleanas

Las colecciones de fotos, tal como hemos visto en la sección anterior, ofrecen unas operaciones de búsqueda *básicas*, por rangos de fechas y por etiquetas.

En nuestra práctica tendremos también consultas booleanas.

Una consulta booleana es una expresión que combina operadores lógicos (**not**, **and**, **or**) y consultas básicas (por fechas o por etiqueta). Diremos que las consultas por fechas son del tipo DATE y las consultas por etiqueta son del tipo TAG. Siguiendo este estilo, diremos que las consultas booleanas son del tipo BOOL.

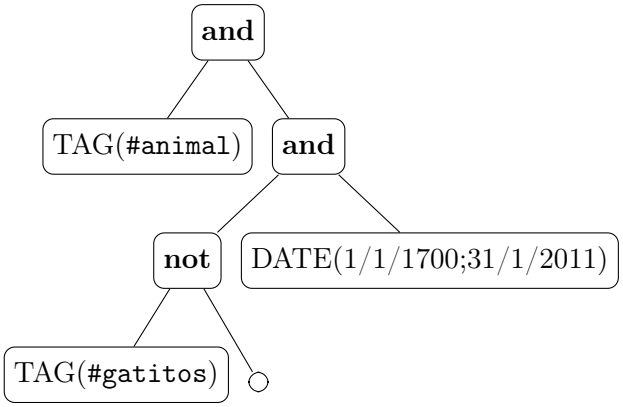
Dada una consulta booleana la operación fundamental será `evalua_consulta`. Dado un identificador de un álbum de la colección (que puede ser `ALL`), devolverá un nuevo álbum con el subconjunto de fotografías pertenecientes al álbum dado que satisfacen la expresión lógica.

Las consultas booleanas se representarán mediante árboles binarios en cuyas hojas (nodos con ambos subárboles vacíos) hay consultas básicas y en cuyos nodos internos tenemos un operador lógico: **not**, **or** o **and**. Todas las consultas en las hojas se hacen sobre el álbum cuyo identificador se da en `evalua_consulta`. En los nodos internos de negación (**not**) sólo se emplea el subárbol izquierdo, el subárbol derecho se deja vacío.

Así podríamos hacer, por ejemplo, una consulta booleana aplicada sobre el álbum `ALL` que nos permite hallar todas las fotografías con la etiqueta `#animal` tomadas antes de febrero de 2011 pero sin la etiqueta `#gatitos`; podríamos describirla así:

BEGIN_QUERY TAG `#animal` **and not** TAG `#gatitos` **and** DATE 1 1 1700 31 1 2011
END_QUERY

Y el árbol correspondiente:



Además de la operación `evaluar_consulta`, se deberán ofrecer las operaciones necesarias para crear consultas booleanas, de manera incremental. Por ejemplo, dadas dos consultas A y B crear otra C que sea el **and** de A y B.

Más adelante se os proporcionará el código necesario (quizás con alguna pequeña adaptación por vuestra parte) para leer consultas booleanas siguiendo el formato sugerido en el ejemplo.

6. Programa principal

Con carácter general, el programa puede asumir que los datos de entrada son sintácticamente correctos y, el caso de la fechas, no se superan los límites del rango previamente mencionado. La entrada del programa consiste en una secuencia de "comandos" aplicados sobre una colección que se encuentra vacía al principio de la ejecución—pero tiene el álbum especial `ALL`, sin fotos, desde el inicio.

El programa también dispone de un álbum que no pertenece a la colección, al cual nos referiremos como **álbum en curso**, que al igual que la colección se encuentra vacío al principio de la ejecución (y su identificador es el string vacío).

Cada comando empieza con "su" nombre (p.e., `lee_album` o `elimina_foto_coleccion` y a continuación vienen los parámetros correspondientes, según se describe en esta sección.

Cada comando producirá unos ciertos efectos sobre la colección y/o el álbum en curso y una cierta salida. En ocasiones, dicha salida será simplemente un mensaje de confirmación, en otras un mensaje de error y en otras una lista de resultados. Todo lo que no se especifique a continuación se podrá derivar del juego de pruebas público de la práctica.

Estos son los comandos que el programa principal debe reconocer. Obviamente, cada uno de ellos podrá usar una o más de las operaciones presentadas en las secciones 2-5.

1. `lee_album`: lee un identificador, un número de fotos y a continuación una secuencia de fotos (cada foto consta de identificador, dia, mes, año, un número de etiquetas y a continuación las etiquetas) y el álbum en curso pasa a ser el formado por esos contenidos. Por ejemplo:

```
lee_album Paris 5
img11 15 8 2019 0
img12 15 8 2019 2 Juan Laura
img13 15 6 2019 1 Laura
img14 15 8 2019 2 Juan ChampsElysees
img23 16 8 2019 3 Laura Juan TorreEiffel
```

2. `agrega_foto`: lee los datos de una foto (como en el ejemplo anterior) y agrega dicha foto al álbum en curso
3. `elimina_foto`: lee un identificador de foto y la elimina del álbum en curso
4. `obten_foto`: lee un identificador de foto, lo busca en el álbum en curso y escribe la información de la foto o una indicación de que dicha foto no está en el álbum en curso
5. `nr_fotos`: escribe el número de fotos en el álbum en curso
6. `agrega_album`: agrega el álbum en curso a la colección; escribe un mensaje de error si el identificador corresponde a un álbum ya existente o es `ALL`
7. `elimina_album`: lee un identificador de álbum y lo elimina de la colección
8. `obten_album`: lee un identificador de álbum y, si la colección contiene un álbum con el identificador dado, escribe el contenido de dicho álbum y el álbum en curso pasa a ser una copia del mismo; en caso contrario escribe una indicación al respecto
9. `nr_albums`: escribe el número de álbumes de la colección
10. `lista_albumes`: escribe los identificadores de los álbumes de la colección, en orden lexicográfico ascendente
11. `agrega_foto_coleccion`: lee la información de una foto y agrega dicha foto a la colección (si no existe ninguna foto con el mismo identificador)
12. `elimina_foto_coleccion`: lee un identificador de una foto y elimina la foto de la colección (y de todos los álbumes de la colección a los que dicha foto perteneciese)
13. `modifica_foto_coleccion`: lee la información de una foto; si una foto con ese identificador existe en la colección, ésta se reemplaza por la foto recién leída; todos los álbumes de la colección a los cuales perteneciese la foto reflejan la modificación de la foto
14. `albumes_foto`: lee un identificador de foto y escribe los identificadores de los álbumes que incluyen la foto, en orden lexicográfico ascendente, o una indicación de que la foto no pertenece a la colección
15. `busca_por_fecha`: lee dos fechas (dia, mes, año), el identificador del álbum de la colección donde hacer la búsqueda y el identificador del álbum resultado; escribe el álbum resultante de la búsqueda, que pasa a ser el álbum en curso; si el identificador del álbum dónde hacer la búsqueda no existe, escribe un mensaje y el álbum en curso no cambia

16. `busca_por_etiqueta`: lee una etiqueta, el identificador del álbum de la colección donde hacer la búsqueda y el identificador del álbum resultado; escribe el álbum resultante de la búsqueda, que pasa a ser el álbum en curso; si el identificador del álbum dónde hacer la búsqueda no existe, escribe un mensaje y el álbum en curso no cambia

17. `evalua_consulta_booleana`: lee el identificador del álbum de la colección dónde hacer la búsqueda, el identificador del álbum resultante y una expresión booleana[1] según el formato explicado en la sección 5; escribe el álbum resultante de la búsqueda, que pasa a ser el álbum en curso; si el identificador del álbum dónde hacer la búsqueda no existe, escribe un mensaje y el álbum en curso no cambia

18. `imprime_album_en_curso`: sin datos; escribe el contenido del álbum en curso; en concreto, imprime el identificador, un salto de línea y a continuación las fotos que contiene el álbum, una por línea, según el formato de los juegos de pruebas

19. `imprime_coleccion`: sin datos; escribe el contenido de todos los álbumes de la colección (`ALL` incluido)

20. `acabar`: sin datos; cesa la ejecución del programa

[1] Finalizada la fase de especificación, se os proporcionará una función que lee la expresión booleana (una secuencia de strings) y devuelve el árbol de expresión