

## TAREA N°2

### Programación en C: Aplicación de Grafos.

**Fecha de envío:** Lunes 05 de junio, a las 23:55 hrs.

**Modalidad:** Trabajo individual

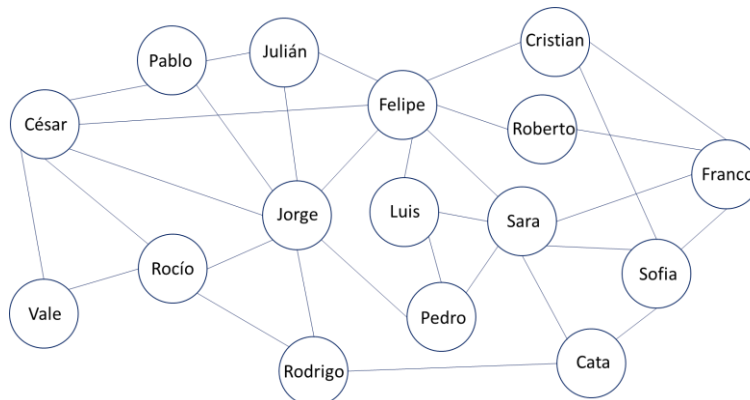
#### I. OBJETIVOS.

El objetivo del presente laboratorio es evaluar tu capacidad para:

1. Llevar a cabo un programa completo en el lenguaje de programación C, aplicando las buenas prácticas en programación (orden, comentarios, identificadores representativos) y de diseño modular.
2. Ser capaz de cumplir con una interfaz concreta solicitada para el programa.
3. Diseñar e implementar una estructura lineal dinámica para resolver un problema que requiera búsqueda de información.

#### II. ENUNCIADO.

Una empresa emergente quiere competir con las grandes redes sociales actuales, a través de su novedosa propuesta **BookFace**. Para esto, se requiere de tu ayuda para proveer que *BookFace* entregue algunas funcionalidades a sus usuarios. En la figura 1 se muestra un ejemplo de esta red social.

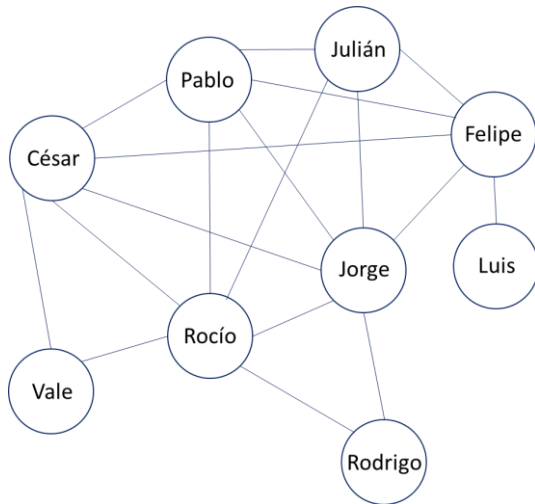


**Figura 1.** Ejemplo de la red social BookFace.

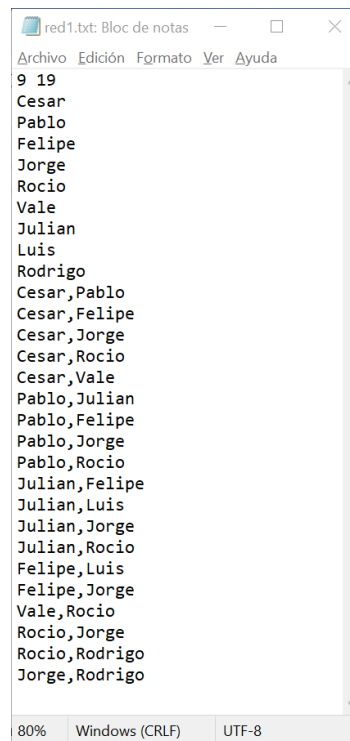
Esta información estará almacenada en un archivo de texto donde con el siguiente formato:

- La primera línea tendrá 2 valores enteros: cuántas personas pertenecen a la red (valor llamado *n* en este documento) y cuántas relaciones de amistad existen en total en BookFace (valor llamado *m* en este documento).
- Las siguientes *n* líneas contendrán los nombres de las personas pertenecientes a red social.
- Las siguientes *m* líneas tendrán el detalle de cada relación de amistad: donde aparecerá el nombre de ambas personas, separadas por una coma.

En la figura 2 aparece un ejemplo de *BookFace* y su correspondiente almacenamiento.



(a)



```

red1.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
9 19
Cesar
Pablo
Felipe
Jorge
Rocio
Vale
Julian
Luis
Rodrigo
Cesar,Pablo
Cesar,Felipe
Cesar,Jorge
Cesar,Rocio
Cesar,Vale
Pablo,Julian
Pablo,Felipe
Pablo,Jorge
Pablo,Rocio
Julian,Felipe
Julian,Luis
Julian,Jorge
Julian,Rocio
Felipe,Luis
Felipe,Jorge
Vale,Rocio
Rocio,Jorge
Rocio,Rodrigo
Jorge,Rodrigo
80% Windows (CRLF) UTF-8

```

(b)

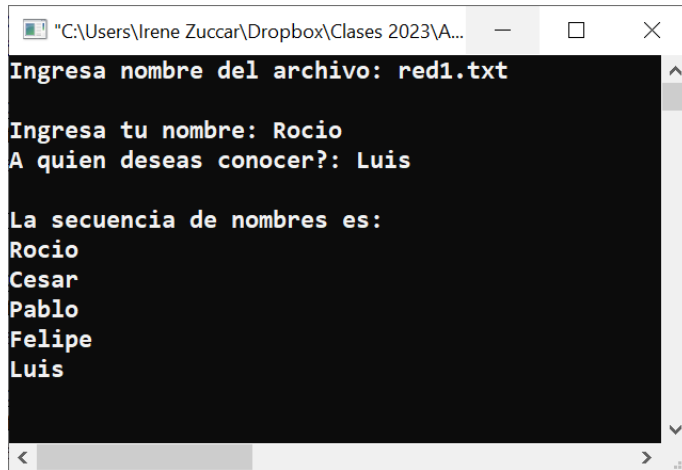
**Figura 2.** (a) Ejemplo de la red social *BookFace* (b) Almacenamiento en archivo de texto.

### Se pide:

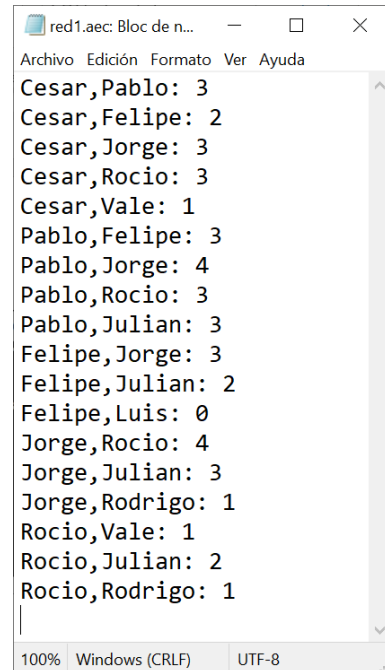
Debes construir un programa en C que:

1. Le pida al usuario el nombre del archivo de entrada (con el formato recién descrito).
2. Luego, el programa deberá solicitar por teclado dos nombres: el primero corresponde al nombre del usuario, y el segundo nombre es la persona que desea conocer. Como respuesta, el programa deberá mostrar UNA secuencia de nombres de personas que se deben seguir para llegar a conocer a la persona de interés. (NOTA: Puede haber más de una secuencia correcta).
3. Además, el programa debe generar un archivo de salida con el mismo nombre que el de entrada, pero con extensión “.aec”. En este archivo debe estar almacenada cada relación de amistad, indicando para cada una la cantidad de amigos en común que posea.

En la figura 3.a aparece un ejemplo con la interfaz solicitada. Observa que se ingresó como archivo de prueba el archivo “**red1.txt**” (figura 2.a). En la figura 3.b se muestra el archivo “**red1.aec**” generado con la información de los amigos en común para cada relación de amistad dentro de la red social.



(a)



(b)

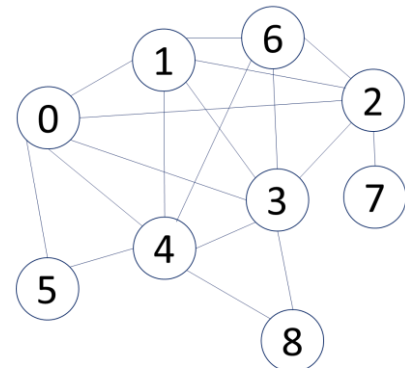
**Figura 3.** (a) interfaz del programa solicitado con el archivo “red1.txt” como entrada (ver figura 2). (b) Almacenamiento en archivo de texto de salida con el número de amigos en común para cada relación existente.

### Sugerencias para la implementación:

1. Podrías usar una matriz de caracteres para almacenar los nombres de las personas, donde cada fila de la matriz corresponda a un nombre. De esta forma, la fila en la que quede cada nombre se asociará a un número y ese será el número del nodo que lo almacene. En la figura 4 se muestra la matriz mencionada para el ejemplo desarrollado en este enunciado.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	C	e	s	a	r															
1	P	a	b	l	o															
2	F	e	l	i	p	e														
3	J	o	r	g	e															
4	R	o	c	i	o															
5	V	a	l	e																
6	J	u	l	i	a	n														
7	L	u	i	s																
8	R	o	d	r	i	g	o													

(a)



(b)

**Figura 4.** (a) Matriz cuyas filas almacenan los nombres de las personas en la red social. Es una forma en la que puedes asociar un nombre a un número de nodo. (b) Representación del grafo (figura 2.a) usando la enumeración conseguida en la matriz.

2. Otra forma de resolver lo anterior: puedes implementar una lista dinámica donde cada nodo posea el nombre de cada persona y un número correlativo (entre 0 y  $n-1$ ).
3. Para construir el camino entre 2 nodos, puedes usar el algoritmo de Dijkstra entregado en cátedra, o podrías trabajar con una modificación del recorrido en profundidad (o DFS):

```
void GeneraCamino(int nodo, int destino)
{
    int i;

    Visitados[nodo] = true;
    Camino[Pos] = nodo;
    Pos++;
    if (nodo == destino)
    {
        CaminoTerminado = true;
        return;
    }
    for (i=0; i<numNodos; i++)
    {
        if (Grafo[nodo][i] != 0 && !Visitados[i])
        {
            GeneraCamino(i, destino);
            if (CaminoTerminado)
                return;
            else
                Pos--;
        }
    }
}
```

En el arreglo “**Camino**” (variable global) quedará el camino entre el “**nodo**” y el “**destino**”, desde la posición 0 hasta la posición “**Pos**”. Se recomienda que el arreglo lo definas según el valor de  $n$  (o “**numNodos**”) del grafo.

### III. CONSIDERACIONES EN LA REVISIÓN.

#### Sobre el código:

1. Tu programa debe trabajar sobre un grafo en forma interna para resolver esta tarea.
2. Tu programa **debe seguir exactamente** las **reglas e interfaces** explicadas en este documento.
3. Debes construir **una función para cada tarea que realice tu código**, cuidando los tipos de datos de las entradas y de las salidas.
4. Debes usar identificadores representativos para tus variables, parámetros de entrada y funciones.
5. Debes comentar cada una de tus funciones, variables globales, estructuras y tipos de datos que definas. Para las funciones, debes indicar una descripción de la labor que lleva a cabo, sus entradas, y su salida. En la figura 5 aparece un ejemplo de cómo debes hacerlo.

```
/*  
Entrada: número entero que se revisará.  
Salida: número entero que corresponde al número de divisores.  
Proceso: función que recibe un entero y calcula y retorna el total  
         de divisores que posee.  
*/  
int numeroDivisores(int n)  
{  
    int i=2, cont=0;  
  
    while (i < n)  
    {  
        if (n%i == 0)  
        {  
            cont++;  
        }  
        i++;  
    }  
    return cont+2;  
}
```

**Figura 5:** Ejemplo de cómo debes comentar una función en esta tarea.

1. Tu código debe estar correctamente *indentado* (uso de sangrías para cada sub-bloque de instrucciones), esto incluye el correcto alineamiento de las llaves (“{” y “}”) que enmarcan tales bloques. Se penalizarán códigos desordenados en este aspecto.
2. Tu código no puede presentar más de 1 línea en blanco. Se penalizarán códigos que no cumplan con este punto.
3. Puedes trabajar con el IDE y compilador de C que más te acomode. No obstante, tu código debe poder ser compilado y ejecutado sin problemas en Windows.
4. El sistema debe ser **robusto**, se penalizarán los errores no manejados de cualquier tipo. En particular: Tu programa debe avisar al usuario si el archivo de entrada no existe; También

debe revisar si los nombres digitados existen en la red, y avisar en tal caso. En ambos casos el programa deberá terminar su ejecución.

#### IV. SOBRE LA ENTREGA, ATRASOS Y FALTAS A LA ÉTICA.

1. Debes subir tu trabajo **al aula virtual de tu curso de cátedra** dentro de la plataforma <https://canvas.unab.com/>, en una casilla que se habilitará especialmente para esto.
2. **NO SE ACEPTARÁN TRABAJOS ATRASADOS.**
3. Tu trabajo es **individual**, no se aceptarán trabajos en grupo.
4. Debe subir un archivo con el código fuente de tu programa.
5. El nombre de tu archivo debe ser **“paterno\_materno\_nombre.c”**
6. Si el programa no se puede ejecutar, tendrá la nota mínima: **1.0**. Si tu programa funciona, se evaluará su ejecución, y también el código fuente.
7. Ante el escenario de existir sospecha de copia (con otros compañeros, o desde internet), será interrogado acerca de su trabajo, para aclarar dudas de su entendimiento y autoría. Si se confirma la sospecha, el trabajo será evaluado con nota **1.0**.
8. Las consultas las debe realizar directamente a los profesores de taller, presencialmente en clases o a los correos y en los horarios que ellos establezcan.