



### 1. Filtrado de un fichero de texto

Se dispone de ficheros de texto que responden a la siguiente sintaxis expresada en notación de Backus-Naur:

```
<fichero-alumnos> ::= { <alumno> }  
<alumno> ::= <nip> <separador> <grupo> <separador> <nombreCompleto> fin_línea  
<nip> ::= literal_entero  
<grupo> ::= literal_entero  
<nombre-completo> ::= literal_string  
<separador> ::= " "
```

Por ejemplo, se supone la existencia de un fichero denominado «alumnos.txt» que sigue dicha sintaxis y cuyo contenido aparece a continuación (la negrita no forma parte del fichero de texto, se ha añadido exclusivamente para facilitar su lectura en este enunciado):

```
747752 411 JORGE RODRIGUEZ SIPAN  
705318 412 PEDRO CUERDA ESCUER  
580368 412 LORETO BERGUA MONGE  
600107 411 JOSE ANDRES ZAMORA BADENES  
502491 411 TIBURCIO JOSA GALO  
742611 411 JACINTO BORAO BERNARDEZ  
513678 411 ESCOLASTICO MATEU FRANCES  
666327 412 ROLDAN ANTUNEZ ZAMORA  
509239 412 AZUCENA AZNAR POBLADOR  
761907 412 ROLDAN MARTINEZ NAVARRO  
614206 411 LIDIA BERNARDEZ MOSQUERA  
710384 412 RODRIGO AZANZA SIPAN  
700648 411 RUBEN CANDAS ADANEZ  
516733 411 FERNANDO CARPINTERO DUQUE  
564584 411 JUAN IGNACIO ABADIA AZANZA  
587194 412 IVAN REMACHA JUAN  
609680 411 LUCAS GALO OTANO  
573900 411 RUBEN JULIAN MOLINS  
630947 411 REMEDIOS MONGE MURIEL  
623042 411 JACINTA ZAPATERO ALONSO
```

Se pide completar el diseño de la función `filtrar()` cuya especificación aparece a continuación:

```
/*  
 * Pre: «nombreOrigen» es el nombre de un fichero de texto donde cada línea  
 *       corresponde a los siguientes datos de un alumno, separados por espacios en  
 *       blanco: NIP, grupo y nombre completo.  
 * Post: Ha filtrado un fichero cuyo nombre es el valor de «nombreOrigen» para dejar,  
 *       en un fichero cuyo nombre es el del valor del parámetro «nombreDestino»,  
 *       solo el NIP y el nombre completo de aquellos alumnos que pertenecen  
 *       al grupo indicado por el valor del parámetro «grupo».  
 */  
void filtrar(const char nombreOrigen[], const char nombreDestino[], int grupo);
```

Es decir, la invocación `filtrar("alumnos.txt", "alumnos-tardes.txt", 412)` debería generar un fichero denominado «alumnos-tardes.txt» con el siguiente contenido:

```
705318 PEDRO CUERDA ESCUER  
580368 LORETO BERGUA MONGE  
666327 ROLDAN ANTUNEZ ZAMORA  
509239 AZUCENA AZNAR POBLADOR  
761907 ROLDAN MARTINEZ NAVARRO  
710384 RODRIGO AZANZA SIPAN  
587194 IVAN REMACHA JUAN
```



# Problemas de Programación 1

## Trabajo con ficheros de texto

### 2. Notación Backus-Naur

Describe la sintaxis de los ficheros producto de las invocaciones a la función filtrar anterior utilizando notación Backus-Naur.

### 3. Cálculo de las frecuencias de aparición de las letras del alfabeto

Se pide diseñar las dos funciones siguientes:

```
/*  
 * Pre: «nombreFichero» es el nombre de un fichero de texto válido y el número de  
 * componentes de la tabla «frecuencias» es igual al número de letras del  
 * alfabeto inglés.  
 * Post: Ha asignado a la tabla «frecuencias» el número de apariciones de cada una de  
 * las letras del alfabeto inglés en el fichero cuyo nombre es «nombreFichero»,  
 * no distinguiendo entre mayúsculas y minúsculas.  
 */  
void analizar(const char nombreFichero[], int frecuencias[]);  
  
/*  
 * Pre: ---  
 * Post: Ha escrito en la pantalla una tabla con las frecuencias de apariciones de  
 * las letras según los valores de la tabla «frecuencias». La primera  
 * componente de la tabla es el número de veces que aparece la letra A, la  
 * segunda, la letra B y así sucesivamente.  
 */  
void escribirFrecuencias(int frecuencias[]);
```

A modo de ejemplo, la siguiente secuencia de declaraciones e instrucciones debería escribir en la pantalla el resultado que se reproduce a continuación:

```
const int NUM_LETRAS = 'Z' - 'A' + 1;  
int frecuencias[NUM_LETRAS];  
analizar("quijote.txt", frecuencias);  
escribirFrecuencias(frecuencias);
```

```
A: 193344  
B: 24146  
C: 59435  
D: 87237  
E: 221979  
F: 7581  
G: 17225  
H: 19920  
I: 77615  
J: 10530  
K: 0  
L: 89141  
M: 44658  
...
```

```
...  
N: 108440  
O: 153359  
P: 35464  
Q: 32483  
R: 100953  
S: 125726  
T: 61749  
U: 78193  
V: 17855  
W: 2  
X: 377  
Y: 25115  
Z: 6491
```