

Programación 1

Tema 14

Ficheros de texto



Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza





Índice

- **Textos y ficheros de texto**
- **Herramientas de C++ para trabajar con ficheros de texto**
- Resolución de problemas básicos con ficheros de texto
 - **Recorrido de un fichero** con información textual
 - **Creación de un fichero** con información textual



Texto

- Texto
 - Información estructurada mediante una secuencia de líneas (0, 1 o más líneas)
 - Cada línea está integrada por una secuencia de caracteres (0, 1 o más caracteres)
- Ejemplos
 - Teclado, pantalla, contenido de ficheros de texto



Texto

- Implementación
 - Secuencia de caracteres donde cada línea termina con uno o varios caracteres de control:
 - Linux
 - Carácter LF (*line feed*), de código ASCII 10
 - Mac OS Classic (pre-Mac OS X)
 - Carácter CR (*carriage return*), de código ASCII 13
 - Windows y algunos protocolos de internet
 - Caracteres CR+LF
 - En C++ representaremos ese carácter (o caracteres) como ‘\n’ o endl, dependiendo del contexto.



Texto

Un soneto me manda hacer Violante
que en mi vida me he visto en tanto aprieto;
catorce versos dicen que es soneto;
burla burlando van los tres delante.

Yo pensé que no hallara consonante,
y estoy a la mitad de otro cuarteto;
mas si me veo en el primer terceto,
no hay cosa en los cuartetos que me espante.

Por el primer terceto voy entrando,
y parece que entré con pie derecho,
pues fin con este verso le voy dando.

Ya estoy en el segundo, y aun sospecho
que voy los trece versos acabando;
contad si son catorce, y está hecho.



Ficheros de texto

- Texto almacenado en un fichero
- Interpretación de la secuencia de bytes de un fichero como caracteres

escalera.txt - Bloc de notas

Archivo Edición Formato Ver Ayuda

INSTRUCCIONES PARA SUBIR UNA ESCALERA

Julio Cortázar

Nadie habrá dejado de observar que con frecuencia el suelo se pliega de manera tal que una parte sube en ángulo recto con el plano del suelo, y luego la parte siguiente se coloca paralela a este plano, para dar paso a una nueva perpendicular, conducta que se repite en espiral o en línea quebrada hasta alturas sumamente variables. Agachándose y poniendo la mano izquierda en una de las partes verticales, y la derecha en la horizontal correspondiente, se está en posesión momentánea de un peldaño o escalón. Cada uno de estos peldaños, formados como se ve por dos elementos, se situó un tanto más arriba y adelante que el anterior, principio que da sentido a la escalera, ya que cualquiera otra combinación producirá formas quizás más bellas o pintorescas, pero incapaces de trasladar de una planta baja a un primer piso.

Las escaleras se suben de frente, pues hacia atrás o de costado resultan particularmente incómodas. La actitud natural consiste en mantenerse de pie, los brazos colgando sin esfuerzo, la cabeza erguida aunque no tanto que los ojos dejen de ver los peldaños inmediatamente superiores al que se pisa, y respirando lenta y regularmente. Para subir una escalera se comienza por levantar esa parte del cuerpo situada a la derecha abajo, envuelta casi siempre en cuero o gamuza, y que salvo excepciones cabe exactamente en el escalón. Puesta en el primer peldaño dicha parte, que para abreviar llamaremos pie, se recoge la parte equivalente de la izquierda (también llamada pie, pero que no ha de confundirse con el pie antes citado), y llevándola a la altura del pie, se le hace seguir hasta colocarla en el segundo peldaño, con lo cual en éste descansará el pie, y en el primero descansará el pie. (Los primeros peldaños son siempre los más difíciles, hasta adquirir la coordinación necesaria. La coincidencia de nombre entre el pie y el pie hace difícil la explicación. Cuídese especialmente de no levantar al mismo tiempo el pie y el pie).

Llegando en esta forma al segundo peldaño, basta repetir alternadamente los movimientos hasta encontrarse con el final de la escalera. Se sale de ella fácilmente, con un ligero golpe de talón que la fija en su sitio, del que no se moverá hasta el momento del descenso.



Ficheros de texto

Código	Carácter												
0	NUL	16	DLE	32	!	48	0	64	@	80	P	96	'
1	SOH	17	DC1	33	"	49	1	65	A	81	Q	97	a
2	STX	18	DC2	34	#	50	2	66	B	82	R	98	b
3	ETX	19	DC3	35	\$	51	3	67	C	83	S	99	c
4	EOT	20	DC4	36	%	52	4	68	D	84	T	100	d
5	ENQ	21	NAK	37	&	53	5	69	E	85	U	101	e
6	ACK	22	SYN	38	'	54	6	70	F	86	V	102	f
7	BEL	23	ETB	39	(55	7	71	G	87	W	103	g
8	BS	24	CAN	40)	56	8	72	H	88	X	104	h
9	HT	25	EM	41	*	57	9	73	I	89	Y	105	i
10	LF	26	SUB	42	+	58	:	74	J	90	Z	106	j
11	VT	27	ESC	43	,	59	;	75	K	91	[107	k
12	FF	28	FS	44	-	60	<	76	L	92	\	108	l
13	CR	29	GS	45	.	61	=	77	M	93]	109	m
14	SO	30	RS	46	/	62	>	78	N	94	^	110	n
15	SI	31	US	47	?	63	?	79	O	95	_	111	o
		48		64		80		96		100		112	p
		49		65		81		97		101		113	q
		50		66		82		98		102		114	r
		51		67		83		99		103		115	s
		52		68		84		100		104		116	t
		53		69		85		101		105		117	u
		54		70		86		102		106		118	v
		55		71		87		103		107		119	w
		56		72		88		104		108		120	x
		57		73		89		105		109		121	y
		58		74		90		106		110		122	z
		59		75		91		107		111		123	{
		60		76		92		108		112		124	-
		61		77		93		109		113		125	}
		62		78		94		110		114		126	~
		63		79		95		111		115		127	DEL
		64		80		96		112		116		128	
		65		81		97		113		117		129	
		66		82		98		114		118		130	
		67		83		99		115		119		131	
		68		84		100		116		120		132	
		69		85		101		117		121		133	
		70		86		102		118		122		134	
		71		87		103		119		123		135	
		72		88		104		120		124		136	
		73		89		105		121		125		137	
		74		90		106		122		126		138	
		75		91		107		123		127		139	
		76		92		108		124		128		140	
		77		93		109		125		129		141	
		78		94		110		126		130		142	
		79		95		111		127		131		143	
		80		96		112		128		132		144	
		81		97		113		129		133		145	
		82		98		114		130		134		146	
		83		99		115		131		135		147	
		84		100		116		132		136		148	
		85		101		117		133		137		149	
		86		102		118		134		138		150	
		87		103		119		135		139		151	
		88		104		120		136		140		152	
		89		105		121		137		141		153	
		90		106		122		138		142		154	
		91		107		123		139		143		155	
		92		108		124		140		144		156	
		93		109		125		141		145		157	
		94		110		126		142		146		158	
		95		111		127		143		147		159	
		96		112		128		144		148		160	
		97		113		129		145		149		161	
		98		114		130		146		150		162	
		99		115		131		147		151		163	
		100		116		132		148		152		164	
		101		117		133		149		153		165	
		102		118		134		150		154		166	
		103		119		135		151		155		167	
		104		120		136		152		156		168	
		105		121		137		153		157		169	
		106		122		138		154		158		170	
		107		123		139		155		159		171	
		108		124		140		156		160		172	
		109		125		141		157		161		173	
		110		126		142		158		162		174	
		111		127		143		159		163		175	
		112		128		144		160		164		176	
		113		129		145		161		165		177	
		114		130		146		162		166		178	
		115		131		147		163		167		179	
		116		132		148		164		168		180	
		117		133		149		165		169		181	
		118		134		150		166		170		182	
		119		135		151		167		171		183	
		120		136		152		168		172		184	
		121		137		153		169		173		185	
		122		138		154		170		174		186	
		123		139		155		171		175		187	
		124		140		156		172		176		188	
		125		141		157		173		177		189	
		126		142		158		174		178		190	
		127		143		159		175		179		191	
		128		144		160		176		180		192	
		129		145		161		177		181		193	
		130		146		162		178		182		194	
		131		147		163		179		183		195	
		132		148		164		180		184		196	
		133		149		165		181		185		197	
		134		150		166		182		186		198	
		135		151		167		183		187		199	
		136		152		168		184		188		200	
		137		153		169		185		189		201	
		138		154		170		186		190		202	
		139		155		171		187		191		203	
		140		156		172		188		192		204	
		141		157		173		189		193		205	
		142		158		174		190		194		206	
		143		159		175		191		195		207	
		144		160		176		192		196		208	
		145		161		177		193		197		209	
		146		162		178		194		198		210	
		147		163		179		195		199		211	
		148		164		180		196		200		212	
		149		165		181		197		201		213	
		150		166		182		198		202		214	
		151		167		183		199		203		215	
		152		168		184		200		204		216	
		153		169		185		201		205		217	
		154		170		186		202		206		218	
		155		171</									



Ficheros de texto

01001001	01001110	01010011	01010100
01010010	01010101	01000011	01000011
01001001	01001111	01001110	01000101
01010011	00100000	01010000	01000001
01010010	01000001	00100000	01010011
01010101	01000010	01001001	01010010
00100000	01010101	01001110	01000001
00100000	01000101	01010011	01000011
01000001	01001100	01000101	01010010
01000001			



Ficheros de texto

73	78	83	84	82
85	67	67	73	79
78	69	83	32	80
65	82	65	32	83
85	66	73	82	32
85	78	65	32	69
83	67	65	76	69
82	65			



Ficheros de texto

I	N	S	T	R
U	C	C	I	O
N	E	S		P
A	R	A		S
U	B	I	R	
U	N	A		E
S	C	A	L	E
R	A			



Ficheros de texto

- Secuencias de *bytes* interpretadas como caracteres
- Estructurados en líneas

Ficheros de texto

- Secuencias de *bytes* intercalados de caracteres
- Estructurados en líneas

pirata.txt - Bloc de notas

Archivo Edición Formato Ver Ayuda

CANCIÓN DEL PIRATA
José de Espronceda

Con diez cañones por banda,
viento en popa, a toda vela,
no corta el mar, sino vuela
un velero bergantín.
Bajel pirata que llaman,
por su bravura, El Temido,
en todo mar conocido
del uno al otro confín.

La luna en el mar riela
en la lona gime el viento,
y alza en blando movimiento
olas de plata y azul;
y va el capitán pirata,
cantando alegre en la nona.

analisisTranvia.cc - Bloc de notas

Archivo Edición Formato Ver Ayuda

```
#include <iostream>
#include <iomanip>
#include <cstring> // funciones strlen(...) y strcmp(...)

#include "../Tranvia/tranvia.h" // módulo tranvia
#include "../operaciones/operaciones.h" // módulo operaciones

using namespace std;

/*
 * Repertorio de códigos de órdenes válidas
 */
const char FIN[] = "FIN";
const char AYUDA[] = "AYUDA";
const char DATOS[] = "DATOS";
const char DIA[] = "DIA";
const char TOTAL[] = "TOTAL";
const char MIN[] = "MINIMO";
const char MAX[] = "MAXIMO";
const char VIAJEROS[] = "VIAJEROSDIA";
const char ACUMULADOS[] = "VIAJEROSACUMULADOS";
```

Ficheros de te

Problemas 2010-11-04.doc - Bloc de not

Archivo Edición Formato Ver Ayuda

Programación de Aplicaciones de la Standard API Ed. 5 (disponible en [docs/api/index.html](http://download-llnw.oracle.com/javase/1.5.0/docs/api/index.html)). **Import** java.io.File; **/**/ * Clase que contiene un método estático para renombrar en masa ficheros de una determinada extensión dentro de un directorio determinado. **public class Renombrar {*/
/* Post: Ha renombrado los ficheros del directorio [nombreDirectorio] que tienen nombre que terminan en [extensión] por [nombreNuevo] seguido de un número consecutivo, manteniendo la extensión */ **public static void renombrarvarios(String nombreDirectorio, String extension, String nombreNuevo) {**
/* Ejecuta [renombrarvarios] en la carpeta D:\temp\fotos\viaje, para cambiar el nombre por "Verano 2010" de los ficheros con extensión jpg */ **public static void main(String[] args) {**
renombrarvarios("D:\\temp\\fotos\\viaje", ".jpg", "Verano 2010"); **}******

Programación de Aplicaciones de Escritorio

java.io.File se pide completar el código del método renombrarVarios de la clase Renombrar para que permita los nombres de los ficheros determinada extensión en masa. El método renombrarVarios para resolver el problema de las imágenes JPG del directorio D:\temp\fotos, se facilita impresión en papel la clase java.io.File y una selección de la Documentación del API.

HYPERLINK "http://download-llnw.oracle.com/technetwork/java/javase/1.5.0/index.html" ::Interfaz de Programación de Aplicaciones de Escritorio standar

HYPERLINK "http://download-llnw.oracle.com/technetwork/java/javase/1.5.0/docs/api/index.html"

::import java.io.File; :: * clase

/* * Clase que contiene un método estático para renombrar en masa ficheros de una determinada extensión en un directorio determinado directorio. */ public class Renombrar {

/* * Post: Ha renombrado los ficheros del directorio [nombreDirectorio] que terminan en [extension] por [nombreNuevo] seguido de un número consecutivo, manteniendo la extensión. */ public static void renombrarVarios(String nombreDirectorio, String extension, String nombreNuevo) {

/* * Ejecuta [renombrarVarios] en la carpeta D:\temp\fotos\viaje, para cambiar el nombre por "Verano 2010" de los ficheros con extensión jpg. */ public static void main(String[] args) { renombrarVarios("D:\\temp\\fotos\\viaje", ".jpg", "Verano 2010"); } }



Problema 1

```
/*
 * Pre: «nombreFichero» es el nombre de un fichero de
 *       texto existente y accesible para su lectura.
 * Post: Si el fichero de nombre «nombreFichero» se puede
 *       leer, asigna a «nLineas» el número de Líneas que
 *       contiene del fichero, a «nCaracteres» el número
 *       de caracteres del mismo y a «lecturaOk», el valor
 *       «true». En caso contrario, asigna «false» a «LecturaOk»
 *       e informa del error ocurrido por «cerr».
 */
void contabilizar(const string nombreFichero,
                  unsigned &nLineas,
                  unsigned &nCaracteres,
                  bool &lecturaOk);
```



Una solución, leyendo carácter a carácter

```
void contabilizar(const string nombreFichero,
                  unsigned &nLineas, unsigned &nCaracteres, bool &lecturaOk){
    ifstream f;
    f.open(nombreFichero);
    if (f.is_open()) {
        nLineas = 0;
        nCaracteres = 0;
        char c;
        while (f.get(c)) {
            nCaracteres++;
            if (c == '\n') {
                nLineas++;
            }
        }
        f.close();
        lecturaOk = true;
    } else {
        cerr << "No se ha podido... \" " << nombreFichero << "\\" . " << endl;
        lecturaOk = false;
    }
}
```



Una solución, leyendo carácter a carácter

```
void contabilizar(const string nombreFichero,
                  unsigned &nLineas, unsigned &nCaracteres,
                  bool &lecturaOk) {
    ...
    nLineas = 0;
    nCaracteres = 0;
    char c;
    while (f.get(c)) {
        nCaracteres++;
        if (c == '\n') {
            nLineas++;
        }
    }
    ...
}
```



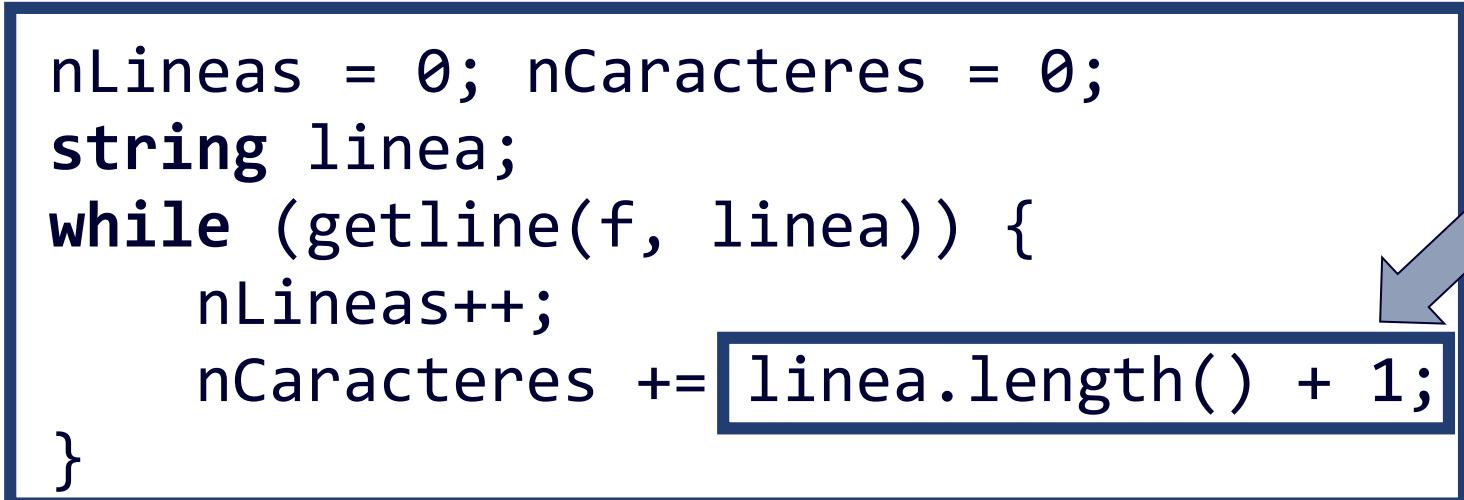
Una solución, leyendo línea a línea

```
void contabilizar(const string nombreFichero,
                  unsigned &nLineas, unsigned &nCaracteres, bool &lecturaOk){
    ifstream f;
    f.open (nombreFichero);
    if (f.is_open()) {
        nLineas = 0;
        nCaracteres = 0;
        string linea;
        while (getline(f, linea)) {
            nLineas++;
            nCaracteres += linea.length() + 1;
        }
        f.close();
        lecturaOk = true;
    } else {
        cerr << "No se ha podido abrir el fichero \""
           << "\"." << endl;
        lecturaOk = false;
    }
}
```



Una solución, leyendo línea a línea

```
void contabilizar(const string nombreFichero,  
                  unsigned &nLineas, unsigned &nCaracteres,  
                  bool &lecturaOk) {  
    ...  
    nLineas = 0; nCaracteres = 0;  
    string linea;  
    while (getline(f, linea)) {  
        nLineas++;  
        nCaracteres += linea.length() + 1;  
    }  
    ...  
}
```





Ficheros de NIF

```
struct Nif {  
    int dni;  
    char letra;  
};  
  
...  
  
const unsigned MAX_NUM_NIF = 700000;  
Nif vector[MAX_NUM_NIF];  
unsigned n;
```



Ficheros de NIF

- Se desea dar persistencia a vectores de datos de tipo Nif:
 - Definición de la sintaxis de un fichero de texto que almacena NIF
 - Diseño del código de dos funciones
 - Una función lea los datos de los NIF y almacene en un vector aquellos que sean válidos (su letra se corresponde con su DNI)
 - Otra función que escriba en un fichero los NIF presentes en un vector



Ficheros de NIF

Sintaxis

```
<fichero-nif> ::= { <nif> }
<nif> ::= <dni> <separador> <letra> <fin-línea>
<dni> ::= literal-entero
<letra> ::= "A" | "B" | "C" | "D" | ...
           | "X" | "Y" | "Z"
<separador> ::= "-"
<fin-línea> ::= "\n"
```



Ficheros de NIF

Ejemplo

23087654-R

23208481-D

...

82413711-L

82534538-G



Ficheros de NIF

Escritura

```
/*
 * Pre: ---
 * Post: Crea un fichero de texto de nombre
 *       «nombreFichero» en el que almacena Los NIF de
 *       Las primeras «n» componentes de «T», a razón de
 *       un NIF por Línea, separando el número de DNI de
 *       la Letra mediante un guion. Si el fichero se
 *       puede crear, asigna «true» a «escrituraOk». En
 *       caso contrario, asigna «false» y escribe un
 *       mensaje de error por «cerr».
 */
void escribirFicheroNif(const string nombreFichero,
                        const Nif T[], const unsigned n,
                        bool &escrituraOk);
```



Ficheros de NIF

Escritura

```
void escribirFicheroNif(const string nombreFichero,
    const Nif T[], const unsigned n, bool &escrituraOk) {
    ofstream f;
    f.open(nombreFichero);
    if (f.is_open()) {
        for (unsigned i = 0; i < n; i++) {
            f << T[i].dni << "-" << T[i].letra << endl;
        }
        f.close();
        escrituraOk = true;
    } else {
        cerr << "No se ha podido escribir el fichero \""
            << nombreFichero << "\\".\\n" << endl;
        escrituraOk = false;
    }
}
```



Ficheros de NIF

Lectura

```
/*
 * Pre: El contenido del fichero de nombre «nombreFichero» sigue la sintaxis
 * de la regla <fichero-nif> y el número de NIF válidos almacenados en
 * el fichero «nombreFichero» es menor o igual a la dimensión del
 * vector «T».
 * Post: Asigna a «nDatos» el número de NIF válidos del fichero y almacena en
 * las primeras «nDatos» componentes del vector «T» la información de
 * los NIF válidos almacenados en el fichero. A «nErroneos» le asigna
 * el número total de NIF del fichero no válidos. Si el fichero se
 * puede abrir, asigna «true» a «lecturaOk». En caso contrario, asigna
 * «false» y escribe un mensaje de error por «cerr».
 */
void leerFicheroNif(const string nombreFichero, Nif T[],
                     unsigned &nDatos, unsigned &nErroneos, bool &lecturaOk);
```



Ficheros de NIF

Lectura

```
void leerFicheroNif(const string nombreFichero, Nif T[],  
                     unsigned &nDatos, unsigned &nErroneos, bool &lecturaOk) {  
    ifstream f;  
    f.open(nombreFichero);  
    if (f.is_open()) {  
        nDatos = 0;  
        nErroneos = 0;  
        char ignorarGuion;  
        while (f >> T[nDatos].dni >> ignorarGuion >> T[nDatos].letra) {  
            if (esValido(T[nDatos])) {  
                nDatos++;  
            } else {  
                nErroneos++;  
            }  
        }  
        f.close();  
        lecturaOk = true;  
    } else {  
        cerr << "No se ha podido leer del fichero \" " << nombreFichero << "\"" << endl;  
        lecturaOk = false;  
    }  
}
```



Ficheros de NIF

Lectura

```
void leerFicheroNif(const string nombreFichero, Nif T[],  
                     unsigned &nDatos, unsigned &nErroneos,  
                     bool &lecturaOk) {  
    ...  
    nDatos = 0;  
    nErroneos = 0;  
    char ignorarGuion;  
    while (f >> T[nDatos].dni >> ignorarGuion  
           >> T[nDatos].letra) {  
        if (esValido(T[nDatos])) {  
            nDatos++;  
        } else {  
            nErroneos++;  
        }  
    }  
    ...  
}
```



¿Cómo se puede estudiar este tema?

- Repasando estas transparencias
- Trabajando con el código de estas transparencias
 - <https://github.com/prog1-eina/tema-14-ficheros-de-texto>
- Leyendo
 - Capítulo 14 de los apuntes del profesor Martínez, adaptados al curso 2021-22
 - Disponibles en Moodle
 - Tutoriales de *Cplusplus.com* (2000–2017)
 - «Basic Input/Output»: http://wwwcplusplus.com/doc/tutorial/basic_io/
 - «Input/output with files»: <http://wwwcplusplus.com/doc/tutorial/files/>
 - En ambos casos se introducen y explican más conceptos de los que se van a ver en este curso
- Problemas de las próximas clases
- Práctica 6