

Laboratorios de computación

Salas A y B

Profesor: Claudia Rodríguez Espino.

Asignatura: Fundamentos de Programación

Grupo: 03


No de Práctica(s): 13 practica.

Integrante(s): Pantoja Escareño Oscar Eduardo

Semestre: 2018-2

Fecha de entrega: 29 de Mayo del 2018

Observaciones:

	Manual de prácticas del Laboratorio de Fundamentos de programación	Código:	MADO-17
		Versión:	02
		Página	203/214
		Sección ISO	8.3
		Fecha de emisión	6 de abril de 2018
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Guía práctica de estudio 13: Lectura y escritura de datos



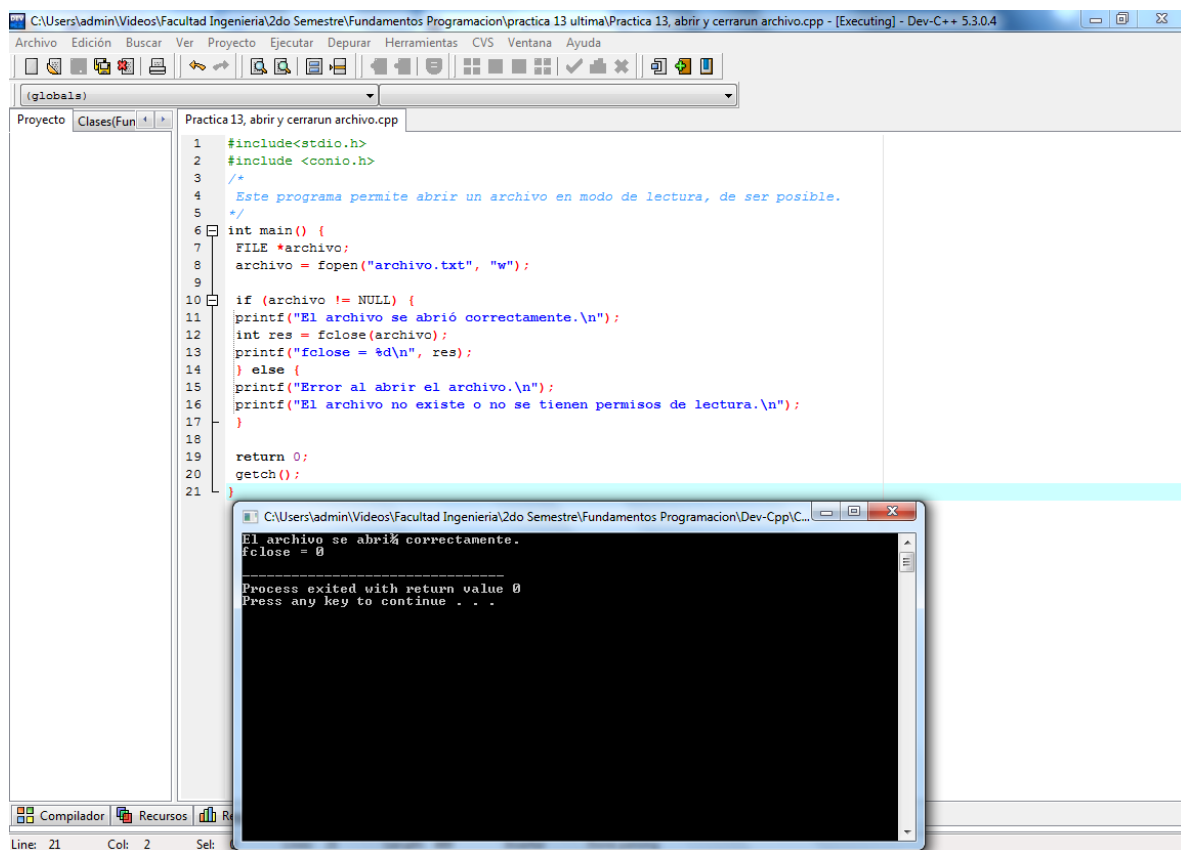
Objetivo

Elaborar programas en lenguaje C que requieran el uso de archivos de texto plano en la resolución de problemas, entendiendo a los archivos como un elemento de almacenamiento secundario.

Desarrollo

Se observaron varias formas de abrir o crear archivos, a través de algunos comandos existentes en C.

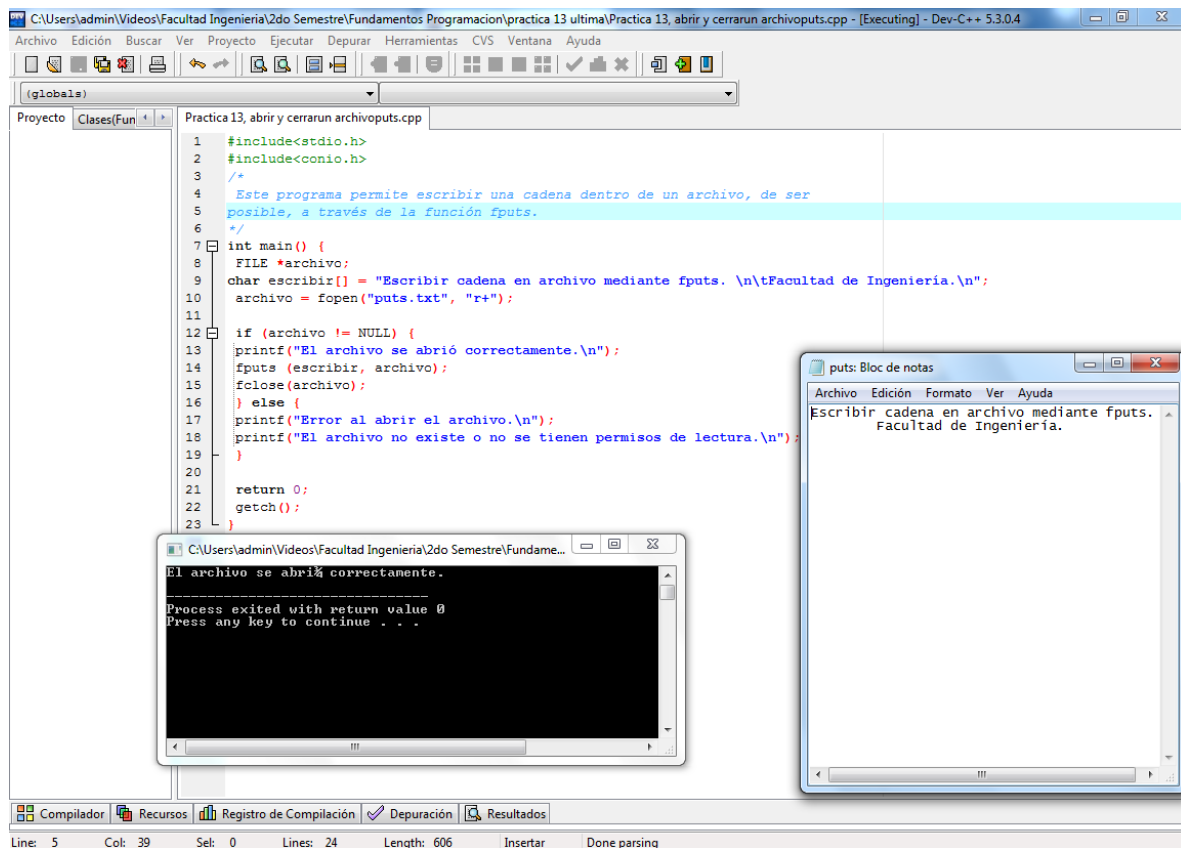
La primera actividad que se realizó fue abrir un archivo y después cerrarlo, aunque este no existiera



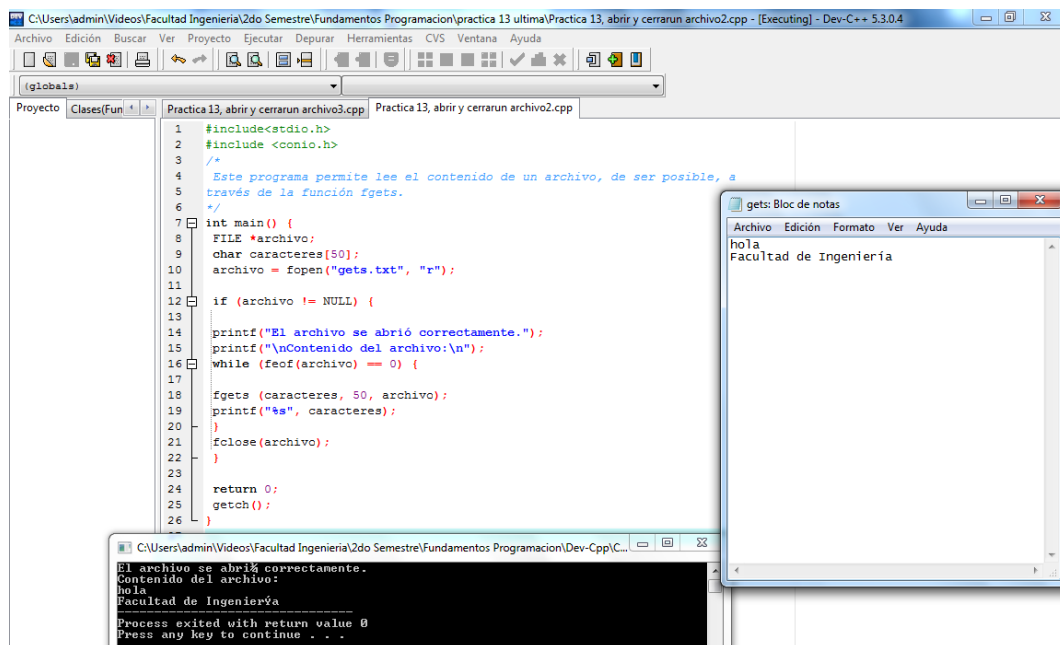
```
1 #include<stdio.h>
2 #include <conio.h>
3 /*
4  * Este programa permite abrir un archivo en modo de lectura, de ser posible.
5  */
6 int main() {
7     FILE *archivo;
8     archivo = fopen("archivo.txt", "w");
9
10    if (archivo != NULL) {
11        printf("El archivo se abrió correctamente.\n");
12        int res = fclose(archivo);
13        printf("fclose = %d\n", res);
14    } else {
15        printf("Error al abrir el archivo.\n");
16        printf("El archivo no existe o no se tienen permisos de lectura.\n");
17    }
18
19    return 0;
20    getch();
21 }
```

El archivo se abrió correctamente.
fclose = 0
Process exited with return value 0
Press any key to continue . . .

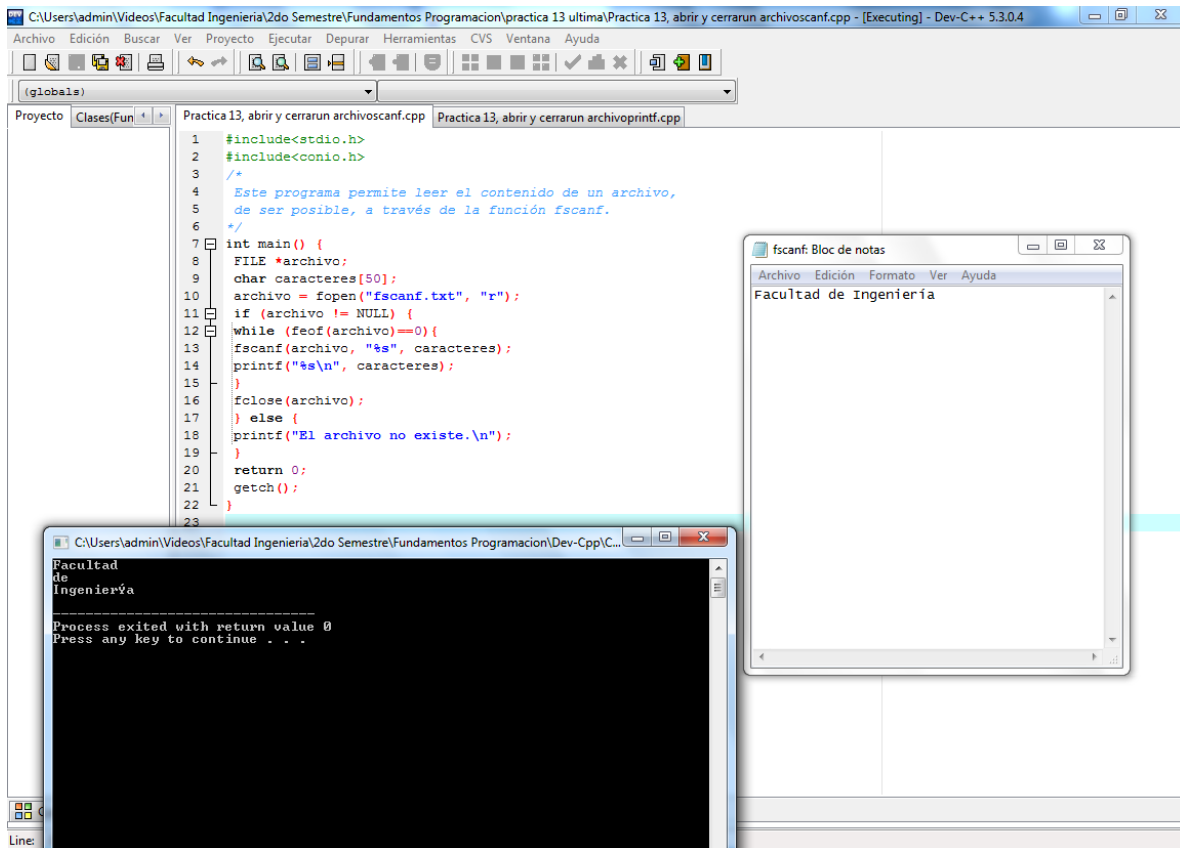
Vimos la acción que realiza la función `fputs()` que es escribir una cadena en un archivo



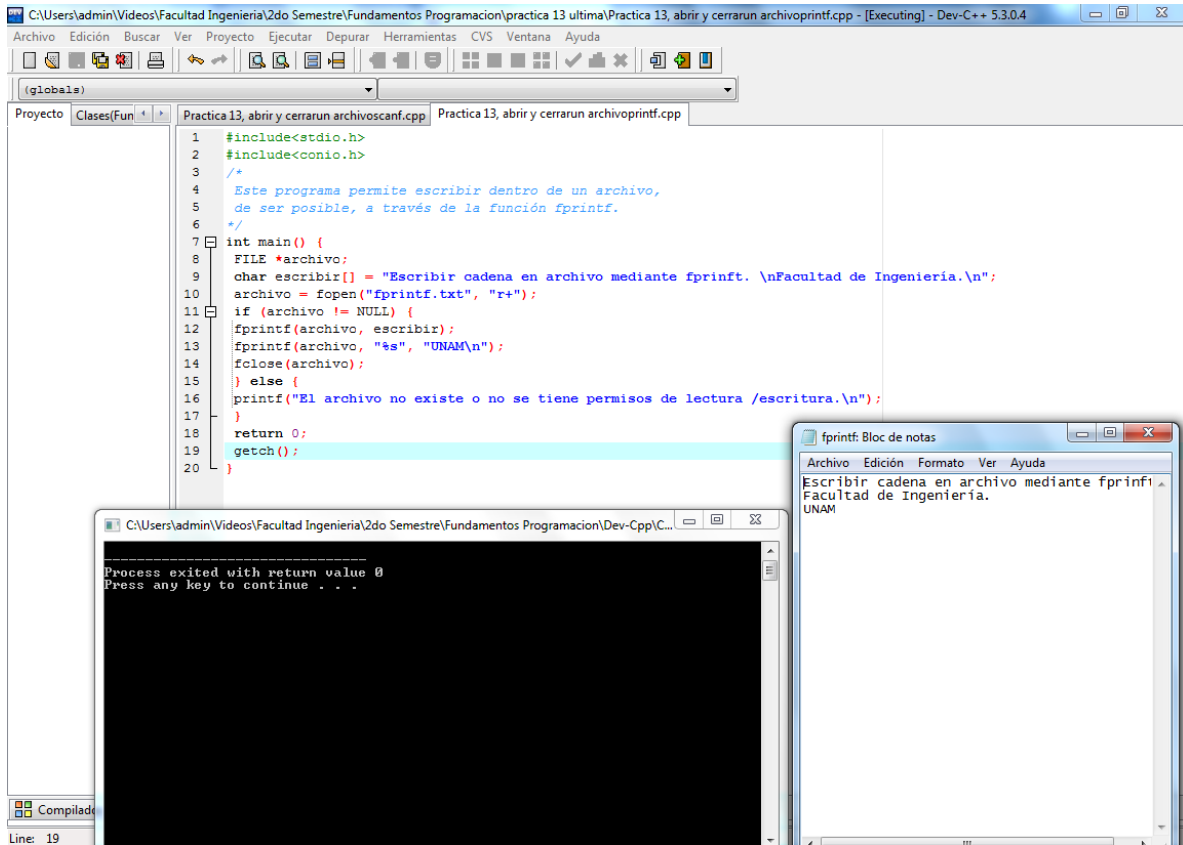
La función `fgets()` lee una cadena desde el archivo especificado. Esta función lee un renglón a la vez.



Las funciones `fprintf()` y `fscanf()` realizan las mismas acciones que el ya conocido `printf()` y `scanf()`, excepción que estos operan sobre archivo



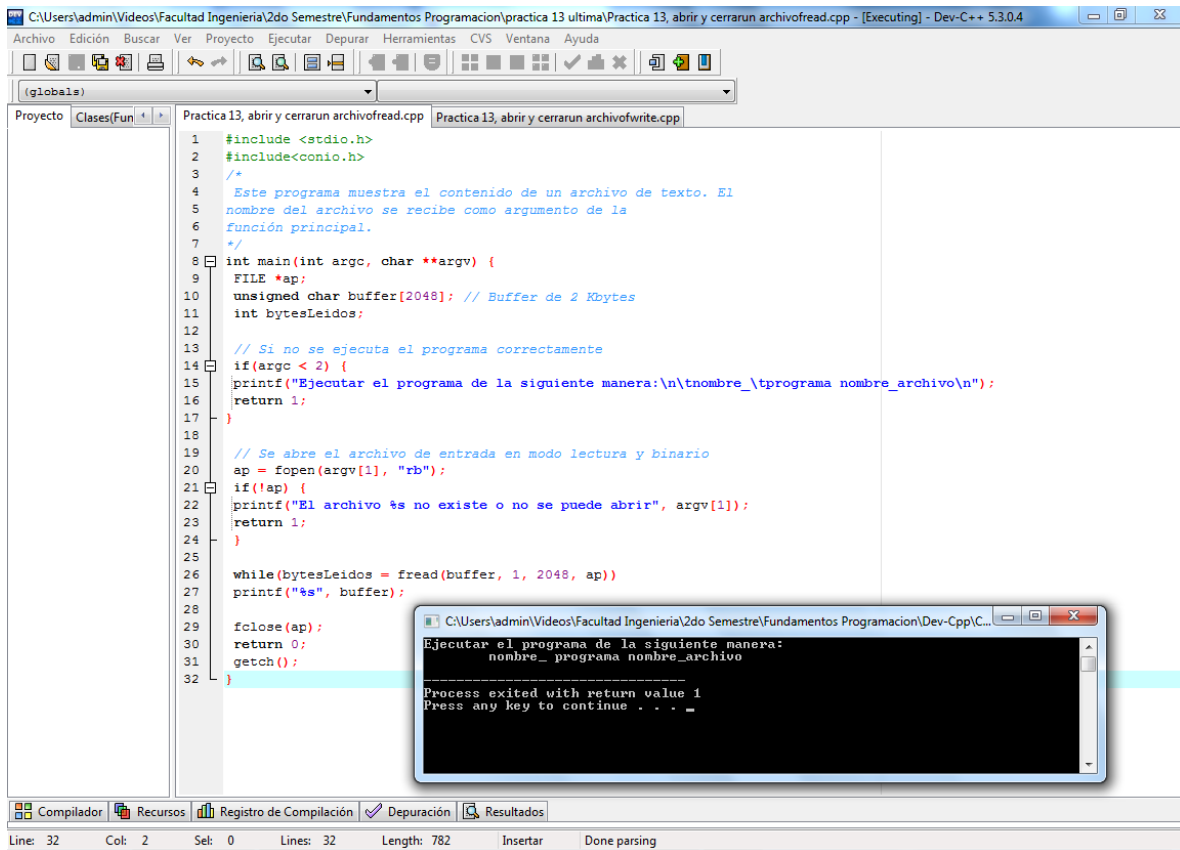
Fread permite leer uno o varios elementos de la misma longitud a partir de una dirección de memoria determinada, es decir un apuntador.



The screenshot displays a C++ development environment with the following components:

- Source Code Editor:** Shows the file `Practica 13, abrir y cerrarun archivoprintf.cpp`. The code includes `<stdio.h>` and `<conio.h>`. It defines a character array `escribir` with the text "Escribir cadena en archivo mediante fprintf. \nFacultad de Ingenieria.\n". The `main` function opens `fprintf.txt` in append mode, writes the string to the file using `fprintf`, and then prints a message to the console using `printf`. The line `getch();` is highlighted in blue.
- Console Window:** Displays the output of the program: "El archivo no existe o no se tiene permisos de lectura /escritura.\n". Below this, it shows "Process exited with return value 0" and "Press any key to continue . . .".
- Text Editor:** A window titled "fprintf: Bloc de notas" shows the content of the file `fprintf.txt`, which contains the text: "Escribir cadena en archivo mediante fprintf. \nFacultad de Ingenieria.\nUNAM".

Fwrite permite escribir hacia un archivo uno o varios elementos de la misma longitud almacenados a partir de una dirección de memoria determinada.



The screenshot displays the Dev-C++ IDE with a C++ program open in the editor. The program is titled "Practica 13, abrir y cerrarun archivofread.cpp". The code includes `<stdio.h>` and `<conio.h>`. It features a `main` function that takes command-line arguments. The program checks if the correct number of arguments is provided (2). If not, it prints a usage message and returns 1. If correct, it opens a file in binary read mode (`"rb"`). If the file cannot be opened, it prints an error message and returns 1. Otherwise, it reads the file content into a buffer of 2048 bytes using `fread` and prints it. The program then closes the file and returns 0.

```
1 #include <stdio.h>
2 #include <conio.h>
3 /*
4  Este programa muestra el contenido de un archivo de texto. El
5  nombre del archivo se recibe como argumento de la
6  función principal.
7  */
8 int main(int argc, char **argv) {
9     FILE *ap;
10    unsigned char buffer[2048]; // Buffer de 2 Kbytes
11    int bytesLeidos;
12
13    // Si no se ejecuta el programa correctamente
14    if (argc < 2) {
15        printf("Ejecutar el programa de la siguiente manera:\n\tnombre _tprograma nombre_archivo\n");
16        return 1;
17    }
18
19    // Se abre el archivo de entrada en modo lectura y binario
20    ap = fopen(argv[1], "rb");
21    if (!ap) {
22        printf("El archivo %s no existe o no se puede abrir", argv[1]);
23        return 1;
24    }
25
26    while (bytesLeidos = fread(buffer, 1, 2048, ap))
27        printf("%s", buffer);
28
29    fclose(ap);
30    return 0;
31    getch();
32 }
```

A terminal window is overlaid on the IDE, showing the output of the program. It displays the usage message and the program's exit status.

```
Ejecutar el programa de la siguiente manera:
nombre _programa nombre_archivo

Process exited with return value 1
Press any key to continue . . . _
```

Conclusiones:

Estas funciones aprendidas nos permiten ejecutar acciones que se enfocan en hacer más que un programa, nos dan la oportunidad de interactuar más con la computadora para poder hacer programa de mayor potencial y mejor estructurado