

# Programación básica

**Alma González**  
**Septiembre 1 2021**



# ABRIR/CERRAR ARCHIVOS, Y LEER/ESCRIBIR CONTENIDO

- En muchas otras ocasiones queremos que la salida de nuestros códigos se guarde en archivos de texto (u otros formatos) para que puedan ser utilizados posteriormente. Ej. Para realizar gráficos.
- Cuando la salida es texto, y el programa es simple podemos hacerlo directamente desde la terminal al ejecutar nuestros códigos. Ejemplo:
  - En la carpeta Sep15 tenemos el código “Ejemplo3\_for.c” podemos ejecutar el código con la instrucción: `ejemplo3_for.o > salida.txt` , y en lugar de imprimir el resultado en la pantalla lo guardará en el archivo `salida.txt` (Nota que el archivo de salida puede tener cualquier nombre que nosotros definamos)
- En muchas ocasiones necesitaremos que nuestros códigos puedan leer información a partir de un archivo y usarla durante la ejecución del mismo.
- En programas sencillos esto podemos hacerlo desde la terminal al ejecutar nuestros códigos. Ejemplo:
  - En la carpeta Sep10 tenemos el código `ejemplo_switch.c`, podemos ejecutarlo con la instrucción `./ejemplo_switch < entrada.txt` , donde el archivo “`entrada.txt`” tiene las opciones que el código requiere para funcionar.

ABRIR Y CERRAR ARCHIVOS.  
PARA LEER SU CONTENIDO, O ESCRIBIR  
CONTENIDO DESDE EL PROGRAMA MISMO.

- Debemos declarar una variable que nos permita manipular el archivo, la sintaxis para declararla es:
- `FILE *fp;` (fp es el nombre de la variable, podemos llamar de otra forma, pero fp es estándar)

# ABRIR Y CERRAR ARCHIVOS.

- Para abrir un archivo, una vez declarada la variable:

`fp=fopen("nombre_archivo", "r");` //El argumento "r" indica que es de solo lectura, otras opciones son: "w" (write, escribir), "a" (append, añadir). También se pueden usar "r+", "w+", "a+"

- Para cerrar el archivo, una vez que termine de leerlo o de escribir en él:

`fclose(fp);`

# ESCRITURA

- Una vez que el archivo ha sido abierto en modo escritura, podemos añadir contenido de la siguiente forma:
  - `fprintf(fp, "%f %f\n", var1, var2);` //Por ejemplo, suponiendo que escribimos una línea que tendrá dos números dados por las variables `var1, var2`. Podemos usar `fprintf` con variables del tipo `char, string, int, float, etc...`
  - `fprintf(fp, "Esta es una prueba de fprintf...\n");`
  - `fputs("Esta es una prueba de fputs...\n", fp);` //Nos permite escribir un string.
  - `fputc("a", fp);` //Nos permite escribir un carácter.

# EJEMPLO ESCRITURA DE ARCHIVO

```
#include <stdio.h>

int main() {

    FILE *archivo;
    float var1,var2;

    var1=0.15;
    var2=100.8;

    archivo = fopen("test.txt", "w");
    fputs("Esta es una prueba de fputs...\n", archivo);
    fprintf(archivo,"fprintf...\n");
    fprintf(archivo, "%.3f %.3f \n",var1,var2);
    fclose(archivo);
    return(0);
}
```

# LECTURA

- Una vez que el archivo ha sido abierto en modo lectura, podemos leer su contenido de la siguiente forma:
  - `fscanf(fp, "%f %f", &var1, &var2);` //Por ejemplo, suponiendo que estamos leyendo la primera linea que tiene dos numeros y asignamos cada numero a una variable.
  - `fgets(var3, 255, (FILE*)fp);` //Suponiendo que la linea es una cadena de caracteres y que se guardara en una variable previamente definida como:
    - `char var3[255];` //el numero 255 es el tamaño máximo para definir una cadena de caracteres (string).

# EJEMPLO LECTURA DE ARCHIVO

```
#include <stdio.h>

int main() {

    FILE *archivo;
    float var1,var2;
    char var[255];

    archivo = fopen("test.txt", "r");
    fgets(var,255,(FILE*)archivo);
    printf("%s",var);
    fscanf(archivo,"%s",var);
    printf("%s\n",var);
    fscanf(archivo,"%f %f",&var1,&var2);
    printf("%f %f\n",var1,var2);
    fclose(archivo);

    return(0);
}
```



# EJERCICIOS:

- parabolac: escribir un programa, que lea de un archivo el número de veces que se quiere evaluar una función, así como el limite inferior y superior en el que se evaluará. Usar la función  $x^2+1$ , y escribir el resultado en un archivo llamado parabolatxt ()
- Modificar el programa Sep15/funciones.c para escribir el resultado a un archivo, en lugar de escribirlo a la pantalla. El nombre del programa deberá ser funciones.c pero estará en la carpeta Sep17. Pueden elegir cualquier nombre para el archivo que se crea.