Práctica 14. Argumentos del main y archivos

Objetivos de la práctica:

- Comprender el uso de los argumentos de la función *main*.
- Adquirir la capacidad de resolución de problemas sencillos.
- Adquirir la capacidad de manejar archivos usando el lenguaje C.

1. Argumentos de la función main

A la función main se le pueden pasar argumentos si al ejecutar el programa le pasamos, después del nombre del ejecutable, valores separados por espacios, por ejemplo: `./ejecutable 3 5.3 "cadena de caracteres"`

Cuando se usan argumentos, la función main se escribe así.

```
int main(int argc, char*argv[]){
...
return 0;
}
```

Sus argumentos son:

- `argc`: indica el número de argumentos
- `argv`: array que contiene los argumentos

Por ejemplo, si queremos imprimir los argumentos de la función main, podemos usar este código:

```
#include <stdio.h>
int main(int argc, char*argv[]){
 int i;
 for(i=0;i<argc;i++)
   printf("%s \n", argv[i]);
 return 0;
}
```

Si ejecutamos el programa con `./ejecutable 3 5.3 "cadena de caracteres"` nos mostrará por pantalla:

```
./ejecutable
3
5.3
cadena de caracteres
```

2. Funciones útiles en stdlib.h

2.1. atoi

Convierte una cadena de caracteres a datos de tipo int.
Por ejemplo:

```
#include <stdio.h>
int main(int argc, char*argv[]){
 int i;
 for(i=1;i<argc;i++)
   printf("%d \n", atoi(argv[i]));
 return 0;
}
```

Si ejecutamos el programa con `./ejecutable 3 4 5` nos mostrará por pantalla:

```
3
4
5
```

2.2. atof

Convierte una cadena de caracteres a datos de tipo float.

3. Archivos

3.1. Tipo de datos FILE

La librería `stdio` tiene el tipo de datos `FILE` para trabajar con archivos. Se trabaja con punteros, de forma que tendremos un puntero a `FILE`:

```
FILE *parchivo;
```

3.2. Abrir archivo

Para abrir un archivo se usa la función `FILE *fopen(const char *nombre, const char *modoacceso);` a la cual se le indica:

- su nombre, extensión incluida
- el modo de acceso: lectura (r), escritura (w), añadir información al final (a), abrirlo como binario (rb, wb, ab), etc.

Si el archivo no puede abrirse devuelve `NULL`. Un ejemplo de cómo abrir un archivo en modo escritura (w) sería:

```
FILE *parchivo;
parchivo=fopen("nombre.txt", "w");
if(parchivo==NULL){
  fprintf(stderr, "Ha ocurrido un error al abrir el archivo");
  exit(EXIT_FAILURE);
}
```

El error lo escribimos en `stderr` que es la salida de error estándar (por defecto, la pantalla). Siempre hay que comprobar si ha dado un error la apertura.

3.3. Cerrar archivo

Después de utilizar un archivo hay que cerrarlo con la función `fclose(parchivo)`. Suele haber un límite de archivos abiertos, por eso hay que cerrarlo si no se va a utilizar.

3.4. Escribir en un archivo

Hay diversas funciones para escribir en un archivo, análogas a las de escribir en pantalla:

- Para escribir una letra (`char`) en un archivo: `fputc(letra,parchivo);`
- Para escribir una letra (`char`) por pantalla: `putchar(letra)` o bien `putc(letra,stdout);`
- Para escribir una cadena (vector de `char`) en un archivo: `fputs(cadena,parchivo);`
- Para escribir una cadena (vector de `char`) por pantalla: `puts(cadena);`
- Para escribir una cadena (vector de `char`) en un archivo: `fprintf(puntero, cadena, argumentos);`
- Para escribir una cadena (vector de `char`) por pantalla: `printf(cadena, argumentos);`
- Otras: ver transparencias Entrada y Salida del moodle.

Por ejemplo:

```
#include <stdio.h>
int main(){
 FILE *parchivo;
 char cadena[50]="Cadena a escribir";
 parchivo=fopen("nombre.txt", "w");
 if(parchivo==NULL){
   fprintf(stderr, "Ha ocurrido un error al abrir el archivo");
   exit(EXIT_FAILURE);
 }
 fprintf(parchivo, "La cadena es %s", cadena);
 fclose(parchivo);
 return 0;
}
```

Si ejecutamos el programa anterior, en el archivo `nombre.txt` quedaría escrito `La cadena es Cadena a escribir`. Es decir, en la terminal nos quedaría:

```
$./ejecutable
$cat archivo.txt
La cadena es Cadena a escribir
```

3.5. Leer de un archivo

Hay diversas funciones para leer de un archivo, análogas a las de leer por teclado:

- Para leer una letra (`char`) de un archivo: `fgetc(parchivo);`

- Para leer una letra (`char`) del teclado: `letra=getchar()` o bien `letra=getc(stdin);`
- Para leer una cadena (vector de `char`) de `tamaño` caracteres de un archivo: `fgets(cadena, tamaño,parchivo);` Devuelve `NULL` si se ha acabado el archivo o si hay un error.
- Para leer una cadena (vector de `char`) del teclado: `gets(cadena);`
- Para leer una línea (vector de `char`) de un archivo: `ssize_t getline(char** cadena, size_t *tam, FILE *archivo);`
- Para leer datos de un archivo: `fscanf(parchivo, formato, &variable);`
- Para leer datos de teclado: `scanf(formato, &variable);`
- Otras: ver transparencias Entrada y Salida del moodle

Por ejemplo:

```
#include <stdio.h>
int main(){
 FILE *parchivo;
 int entero1, entero2;
 parchivo=fopen("nombre.dat", "r");
 if(parchivo==NULL){
    fprintf(stderr, "Ha ocurrido un error al abrir el archivo");
    exit(EXIT_FAILURE);
 }
 fscanf(parchivo, "%d%d", &entero1, &entero2);
 printf("Los números leídos son%d y%d", entero1, entero2);
 fclose(parchivo);
 return 0;
}
```

Si ejecutamos el programa anterior en la terminal nos quedaría:

```
$cat archivo.txt
234 6234 345
$./ejecutable
Los números leídos son 234 y 6234
```

3.6. Otras funciones: feof

La función `feof(parchivo);` permite detectar el final del archivo. Devuelve 0 si no se ha llegado al final del archivo y un valor distinto de 0 si se ha llegado al final. Por ejemplo el siguiente programa muestra letra a letra el contenido de un archivo hasta el final:

```
#include <stdio.h>
int main(){
 FILE *parchivo;
 char letra;
 parchivo=fopen("nombre.txt", "r");
 if(parchivo==NULL){
    fprintf(stderr, "Ha ocurrido un error al abrir el archivo");
    exit(EXIT_FAILURE);
```

```
 }
 while(!feof(parchivo)){
    letra=fgetc(parchivo);
    printf("Una letra es%c ", letra);
 }
 fclose(parchivo);
 return 0;
}
```

3.7. Ejemplo de uso de getline para leer el contenido de un archivo y guardarlo en una matriz de char

Cada línea de la matriz será una línea de archivo. Hay más información sobre esta función `getline` en las transparencias de Entradas y Salidas del moodle.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define _GNU_SOURCE
#define MAX_CHAR 100
#define MAX_LINEAS 5
int main(){
 FILE *parchivo;
 char m[MAX_LINEAS][MAX_CHAR];
 char *cadena=NULL;
 size_t tam=0;
 ssize_t bytesleidos;
 int i=0;
 parchivo=fopen("nombre.txt", "r");
 if(parchivo==NULL){
   fprintf(stderr, "Ha ocurrido un error al abrir el archivo");
   exit(EXIT_FAILURE);
 }
 do{
   bytesleidos=getline(&linea,&tam,parchivo);
   if(bytesleidos!=-1){
      strncpy(m[i],linea,bytesleidos-1);
      m[i][bytesleidos-1]='\0';
      i++;
 }while(bytesleidos!=-1);
 free(linea);
 fclose(parchivo);
 return 0;
}
```

4. Ejercicios propuestos

4.1. Ejercicio 1

- Realice un programa `cadenas.c` en lenguaje C al cual se le pasen una cadena y un número como argumentos. La función `main` leerá otra cadena de como máximo 100 caracteres por teclado. A continuación debe calcular:
  - La longitud de la cadena introducida por teclado y mostrarla por pantalla.
  - Concatenar la cadena introducida con la cadena "Cadena añadida" y mostrar la cadena resultante por pantalla.
  - Buscar si contiene la cadena pasada como argumento y en caso de que la tenga, indicarlo por pantalla.
  - Mostrar la letra que ocupa la posición especificada por el número que se le pasa como argumento. Es decir, si la cadena leída es "Hola" y se le pasó el número 2, debe mostrar la letra *o*.

4.2. Ejercicio 2

- Realice un programa `info.c` en lenguaje C que lea información de un archivo sobre un máximo de 5 alumnos y lea una letra y muestre por pantalla el nombre y apellidos del alumno con esa inicial. Después mostrará por pantalla todo el contenido del archivo.
- Para realizarlo, el programa ha de contener una función que lea un archivo y guarde la información en una matriz. Si hay un error en la apertura, se saldrá del programa. NOTA: usa la función `getline` y `strncpy`.
- El archivo `info.txt` contiene información sobre los nombres y apellidos de una serie de personas; cada línea tiene los nombres y apellidos de la persona en cuestión. Como mucho contiene 5 líneas. Por ejemplo:

  David González Pérez
  Javier Sánchez Fernández
  Nerea Álvarez Álvarez
  Pablo Llamazares García
  Paula Martínez Sánchez

- La función `main` primero llamará a la función para leer el archivo y guardarlo en una matriz. A continuación, leerá una letra. Después llamará a otra función que dada una matriz y una letra, imprime todas las filas que empiezan por la letra pasada como argumento. Por ejemplo si se lee la P, se imprimirá por pantalla:

  Pablo Llamazares García
  Paula Martínez Sánchez

- Por último, la función `main` llamará a una función para imprimir toda la matriz con la información sobre los alumnos.