Lenguaje C - Resumen rápido

(Un resumen para aprendices, para ayudar a recordar funciones que pueden resultar complejas, como las de cadenas y ficheros)

Un programa elemental

```
#include <stdio.h>
main() {
 printf("Hola");
}
```

Tipos de datos básicos

- Números enteros: int
- Números reales: float (simple precisión) y double (doble precisión)
- Caracteres: char
- Operaciones habituales: suma (+), resta (-), multiplicación (*), división (/), resto de la división o "módulo" (%).
- Operaciones abreviadas: incremento en una unidad (++), decremento en una unidad, incremento en varias unidades (x+=2), decremento en varias unidades (z-=3) y abreviaturas similares para las demás operaciones (x*=5; z/=4; y%=2;).

Estructuras básicas

if

- Comprueba una condición.
- Ejemplo: if (x==5) printf("Vale 5");
- Tipos de condiciones posibles: Mayor que (>), menor que (<), mayor o igual que (>=), menor o igual que (<=), distinto de (!=), igual a (==).
- Si no se cumple la condición: else -> if (x==5) printf("Vale 5"); else printf("No vale 5");
- Para enlazar varias condiciones: y (&&), o (||), no (!): if ((x==5) && (y==1)) printf("Son 5 y 1");

while

- Repite mientras se cumpla una condición (la condición se comprueba antes de empezar a repetir).
- Ejemplo: while (x!=correcto) { printf("Vuelve a intentar"); scanf("%d", &x); }
- Tipos de condiciones y forma de enlazarlas: igual que para "if".

do..while

- Repite mientras se cumpla una condición (la condición se comprueba después de haber dado la primera "vuelta").
- Ejemplo: do { printf("Vuelve a intentar"); scanf("%d", &x); } while (x!=correcto);
- Tipos de condiciones y forma de enlazarlas: igual que para "if".

Manejo básico de pantalla y teclado

printf

- Escritura formateada en pantalla
- Ejemplo: printf("El resultado es %d", x);
- Formatos más habituales: %d = número entero en decimal, %f = número real (coma flotante), %c = caracter, %s = cadena de texto, %x = número entero en hexadecimal
- Devuelve: el número de caracteres escritos

scanf

- Lectura formateada desde teclado
- Ejemplo: scanf("%d", &x);
- Formatos más habituales: similares a "printf"
- Devuelve: el número de datos leidos (0 = ninguno, EOF = error)
- Observaciones: en general, el nombre de la variable se debe preceder de & (no es necesario si se trata de un array)

puts

- Escritura de una cadena en pantalla (con avance de línea)
- Ejemplo: puts("Hola");
- Devuelve: EOF en caso de error; un valor no negativo si todo ha ido bien

gets

- Lectura de una cadena desde teclado
- Ejemplo: gets(nombre);
- Devuelve: NULL en caso de error; la cadena si todo es correcto

putchar

- Escritura de una letra en pantalla
- Ejemplo: putchar('a');
- Devuelve: EOF en caso de error; el carácter en caso contrario

getchar

- Lectura de una letra desde teclado
- Ejemplo: letra = getchar();
- Devuelve: EOF en caso de error; el carácter en caso contrario
- Observaciones: no se analizan las letras tecleadas hasta que se pulsa Intro

Manejo de cadenas

strlen

- Devuelve la longitud (número de letras almacenadas) en una cadena de texto, sin contar el carácter nulo final.
- Include: string.h
- Parámetros: la cadena a analizar
- Devuelve: un número entero
- Ejemplo: int longitud = strlen("hola");

strcpy

- Asigna una valor a una cadena de texto (no se permite usar construcciones como *cadena="hola"* salvo cuando se declara una variable).
- Include: string.h
- Parámetros: la cadena de destino y la cadena de origen
- Devuelve: la cadena de destino, modificada (se puede ignorar el valor devuelto, porque el parámetro se modifica también)
- Ejemplo: strcpy(nombre, "juan");

strcat

- Añade un texto al final de una cadena.
- Include: string.h
- Parámetros: la cadena de destino y la cadena a añadirle
- Devuelve: la cadena de destino, modificada (se puede ignorar el valor devuelto, porque el parámetro se modifica también)
- Ejemplo: strcat(saludo, " y familia");

strstr

- Comprueba si una cadena contiene un texto.
- Include: string.h
- Parámetros: cadena y 'texto a buscar'
- Devuelve NULL si no la contiene
- Ejemplo: if(strstr(cadena,"pepe")!=NULL);

strcmp

- Compara dos cadenas
- Include: string.h
- Parámetros: cadena1 y cadena2
- Devuelve: un número entero (0 si son iguales, 1 si no lo son).

- Ejemplo: if(strcmp(cadena1,cadena2)==0);

...

Manejo de ficheros

1. Resumen de ficheros

Para manejar ficheros, siempre deberemos realizar tres operaciones básicas:

- Abrir el fichero.
- Leer datos de él o escribir datos en él.
- Cerrar el fichero.
- Para declarar fichero: **FILE* fichero;**
- Para abrir un fichero: **fichero = fopen(nombreFichero, modo)**; Los modos son:
    - **r** Abrir sólo para lectura.
    - **w** Crear para escribir. Sobreescribe el fichero si existiera ya (borrando el original).
    - **a** Añade al final del fichero si existe, o lo crea si no existe.
    - **+** Se escribe a continuación de los modos anteriores para indicar que también queremos modificar. Por ejemplo: r+ permite leer y modificar el fichero
    - **t** Abrir en modo de texto.
    - **b** Abrir en modo binario.
- Obtener un carácter del fichero: **caracter = fgetc(fichero)**
- Escribir un carácter en el fichero: **fputc(caracter,fichero)**
- Escribir datos formateados en el fichero: **fprintf(fichero,"formato", argumento1, argumento2...)**
- Leer de fichero: **fscanf(fichero, "formato", argumento1,....)**
- Cerrar fichero: **fclose(fichero)**
- Distinto de 0 si acaba el fichero: **feof(fichero)**
- Leer linea en cadena: **fgets(cadena, cantidad , fichero)**
- Escribir cadena: **fputs(cadena, fichero)**
- Saltar a una posicion:**fseek(fichero, salto, desde)** (el "desde" puede ser: SEEK_SET, SEEK_CUR, SEEK_END, para principio, actual o final del fichero)
- Saber la posicion actual: **donde = ftell(fichero)**
- Leer un bloque de datos: **fread(&donde,tamaño_dato,cuantos_datos,fichero);**
- Escribir un bloque de datos: **fwrite(&donde,tamaño_dato,cuantos_datos,fichero);**

2. Explicación detallada de funciones de ficheros

(Todas ellas en "stdio.h")

fopen

- Intenta abrir un fichero.
- Parámetros: el nombre físico del fichero y el modo de apertura
- Devuelve: un puntero a fichero (FILE*) si se ha podido abrir; NULL si no se ha conseguido.
- Ejemplo: fichero = fopen("datos.txt", "rt");
- Modos de apertura habituales: "rt" = lectura de un fichero de texto, "rb" = lectura de un fichero binario; "r+" (r+b o r+t) en vez de r si se quiere leer pero también escribir; "a" (at, ab) para añadir al final del fichero; "w" (wt, wb) para crear un fichero (borrando el actual, si existiera).

fclose

- Cierra un fichero.
- Parámetros: el identificador del fichero
- Devuelve: 0 si se ha podido cerrar; EOF si no se ha podido
- Ejemplo: fclose(fichero);

fgets

- Lee una cadena desde un fichero.
- Parámetros: la cadena, la longitud máxima, el identificador del fichero
- Devuelve: NULL en caso de error; la cadena, si todo ha sido correcto
- Ejemplo: fgets(texto, 20, fichero);
- Observaciones: conserva el avance de línea (\n) al final de la cadena leída.

fputs

- Guarda una cadena en un fichero.
- Parámetros: la cadena, la longitud máxima, el identificador del fichero
- Devuelve: EOF en caso de error; el último carácter escrito, si todo ha ido bien
- Ejemplo: fputs(texto, fichero);
- Observaciones: no añade avance de línea (\n) al final de la cadena.

fgetc

- Lee un carácter desde un fichero.
- Parámetros: el identificador del fichero
- Devuelve: EOF en caso de error; el carácter leido (convertido a int), si todo ha ido bien
- Ejemplo: letra = fgetc(fichero);

fputc

- Guarda un carácter en un fichero.
- Parámetros: el carácter, el identificador del fichero
- Devuelve: EOF en caso de error; el carácter escrito, si todo ha ido bien
- Ejemplo: fputc('a', fichero);

fscanf

- Lee datos con formato desde un fichero, de forma similar a "scanf"
- Ejemplo: fscanf("%d%d", &num1, &num2);

fprintf

- Guarda datos con formato en un fichero, de forma similar a "printf"
- Ejemplo: fprintf(fichero, "Edad: %d", edad);

fread

- Lee datos desde un fichero.
- Parámetros: el bloque de datos, el tamaño de cada dato, la cantidad de datos, el identificador del fichero
- Devuelve: el número de datos leído (en caso de error, será menor que lo esperado)
- Ejemplo: fread(&ficha, sizeof(ficha), 1, fichero);

fwrite

- Guarda datos en un fichero.
- Parámetros: el bloque de datos, el tamaño de cada dato, la cantidad de datos, el identificador del fichero
- Devuelve: el número de datos guardados (en caso de error, será menor que lo esperado)
- Ejemplo: fwrite(&arrayDeInt, sizeof(int), 100, fichero);

fseek

- Salta a una cierta posición de un fichero.
- Parámetros: el identificador del fichero, el desplazamiento, el origen
- Devuelve: 0 si se ha podido saltar; otro valor si no se ha podido
- Ejemplo: fseek(fichero, -10, SEEK_CUR);
- Valores de "origen" del salto: SEEK_SET (0, inicio), SEEK_CUR (1, actual), SEEK_END (2, final)

ftell

- Devuelve la posición actual en un fichero.
- Parámetros: el identificador del fichero
- Devuelve: un entero largo (-1 en caso de error)
- Ejemplo: posicion = ftell(fichero);

(www.aprendeaprogramar.com
Versión actual de este resumen: 0.11)