CUANDO PROGRAMAMOS EN C, HAY QUE TENER EN CUENTA LO SIGUIENTE

Manejo de errores

- Es muy importante **detectar a tiempo que ha habido un** error cuando se ejecuta nuestro programa y en ese momento terminar la ejecución para evitar qué seguir con ella nos lleve a errores más graves que puedan "colgar" la máquina, por ejemplo.
- Es recomendable que la **salida del programa vaya a stdout (o salida estándar) y los errores vayan a stderr.** Por ejemplo:
  fprintf(stdout, "La palabra es aceptada\n");
  fprintf(stderr, "Los argumentos de entrada son invalidos\n");

En la lectura de ficheros

Cuando se accede a un fichero para leerlo nos pueden ocurrir los siguientes casos que debemos evitar;

- Se intenta leer **un fichero que no existe** y en vez de **dar error y salir del programa** se continúa y al final el programa termina fallando.
- Nuestro programa no detecta el final de fichero y se cuelga. **Con las funciones de detección de final de fichero y con las funciones de lectura de fichero debemos poder detectar cuando terminamos de leer un fichero.**
- Cuando se está leyendo un fichero no se comprueba que las cosas que se desean leer estén como se espera y esto puede producir que nuestro programa "cuelgue" la máquina. **Cuando se está leyendo de un fichero y rellenando con lo que lee unos datos** se **debe comprobar que** se **lee lo deseado para evitar problemas.**

En el paso de argumentos al main

- **Si nuestro programa puede recibir parámetros** a la hora de ejecutarse y **estos no son introducidos,** lo más lógico es, **o bien pedirlos a continuación, o dar un error diciendo cual es la sintaxis de lo que se debería haber introducido** - Y para completar podría el programa mostrar una ayuda de los parámetros que acepta y su uso a través del parámetro -? o -h tras el nombre del programa.
- **Si** el programa espera argumentos en línea de comandos, hay que asegurarse de que el usuario los ha introducido **comprobando el valor de argc.**

Trabajo con memoria dinámica

- Lo primero que tenemos que tener en cuenta es que tenemos en nuestro fichero los includes: **<alloc.h> y <stdlib.h>.**
- **Siempre que usemos una variable definida como puntero hay que reservar memoria,** el siguiente ejemplo sería erróneo ya que no se le da memoria en ningún sitió al puntero a char:

  char *cadena;

  y después:

  scanf("%s", cadena);

- Se tiene que reservar la memoria para la cadena de caracteres utilizada.
- Si el programa tiene que parar la ejecución por un error, se debe liberar la memoria reservada hasta el momento.

Trabajo con includes (.h);

- . No se puede poner ningún include del tipo
- Indicando el path completo al archivo: #include “a:\mifichero.h” Se pone de forma relativa:

  #include “mifichero.h”
- De un archivo de código: #include “mifichero.c” Nunca un .c en un include.
- De un archivo nuestro, entre **"<"** y ">": #include <mifichero.h> Se pone con comillas: #include

"mifichero.h"

Siendo mifichero.h o mifichero.c ficheros creados por nosotros. Lógicamente no nos referimos a stdlib.h o stdio.h, u otros includes del sistema.

- **No olvidar poner ningún include,** es típico olvidar <alloc.h>, <stdlib.h> (sin estos includes el programa puede dejar la maquina "colgada" a la hora de ejecutarse). Para saber si nos olvidamos poner algún include hay que configurar el Turbo C para que nos avise con un "warning" de ello. Lo más cómodo y recomendable es activar en las opciones del compilador Turbo C (o el que utilicemos) para que avise con un mensaje en todos los casos que nos ofrece.
- Cuando creamos nuestro fichero de cabecera (.h) tenemos que tener las siguientes instrucciones en éste:

  ```
  #ifndef__IDENTIFICADOR
  #define__IDENTIFICADOR
  (...)
  #endif
  ```