Librerías Básicas

(Notas / Apéndice)

Los archivos de cabecera en donde están definidas algunas de las funciones que son utilizadas habitualmente (trabajando p.ej. sobre plataforma Windows) son:

Librería stdio.h

printf
Función: Escribe en la salida estándar con formato.
Sintaxis1: printf(Mensaje y/o formato , arg1 , ...);
Sintaxis2: printf("... Mensaje de salida ...");

scanf
Función: Lee de la salida estándar con formato.
Sintaxis: scanf(formato , arg1 , ...);

puts
Función: Escribe una cadena y salto de línea.
Sintaxis: puts(cadena);

gets
Función: Lee y guarda una cadena introducida por teclado.
Sintaxis: gets(cadena);

fopen
Función: Abre un archivo en el modo indicado.
Sintaxis: pf=fopen(fichero , modo);
Donde **pf** es un apuntador hacia un archivo.

fclose
Función: Cierra un archivo cuyo apuntador le indicamos.
Sintaxis: fclose(pf);

fprintf
Función: Escribe con formato en un archivo.
Sintaxis: fprintf(pf , formato , arg1 , ...);

fgets
Función: Lee una cadena de un archivo.
Sintaxis: fgets(cadena , longitud , pf);

Librería stdlib.h

atof
Función: Convierte una cadena de texto en un valor de tipo float.
Sintaxis: numflo=atof(cadena);

atoi
Función: Convierte una cadena de texto en un valor de tipo entero.
Sintaxis: nument=atoi(cadena);

atol
Función: Convierte una cadena de texto en un valor de tipo long.
Sintaxis: numlong=atol(cadena);

itoa
Función: Convierte un valor numérico entero en una cadena de texto. La base generalmente será 10, aunque se puede indicar otra distinta.
Sintaxis: itoa(número , cadena , base);

exit
Función: Termina la ejecución y abandona el programa.
Sintaxis: exit(estado); /* Normalmente el estado será cero 0 para indicar que el programa finalizara sin ningún problema */

Librería conio.h

clrscr
Función: Borra o limpia la pantalla.
Sintaxis: clrscr( );

clreol
Función: Borra desde la posición del cursor hasta el final de la línea.
Sintaxis: clreol( );

gotoxy
Función: Cambia la posición del cursor a las coordenadas indicadas.
Sintaxis: gotoxy(columna , fila ó renglón);
NOTA: En un monitor estándar se tienen aproximadamente 78 columnas x 25 renglones.

textcolor
Función: Selecciona el color de texto (0 - 15).
Sintaxis: textcolor(color);
(ver tabla de colores)

textbackground
Función: Selecciona el color de fondo (0 - 7).
Sintaxis: textbackground(color);
(ver tabla de colores)

Observación: En caso de usar la sentencia **clrscr()**, se recomienda que ésta última función sea posterior a las funciones de formato de color.

Tabla de colores
(prototipo: librería CONIO.H.)

MODALIDADES DE COLOR (text mode)			
COLORES	Valor (color)	Plano de Fondo FONDO	Plano Frontal TEXTO
BLACK	0	✓	✓
BLUE	1	✓	✓
GREEN	2	✓	✓
CYAN	3	✓	✓
RED	4	✓	✓
MAGENTA	5	✓	✓
BROWN	6	✓	✓
LIGHTGRAY	7	✓	✓
DARKGRAY	8	No aplica (idem 0)	✓
LIGHTBLUE	9	No aplica (idem 1)	✓
LIGHTGREEN	10	No aplica (idem 2)	✓
LIGHTCYAN	11	No aplica (idem 3)	✓
LIGHTRED	12	No aplica (idem 4)	✓
LIGHTMAGENTA	13	No aplica (idem 5)	✓
YELLOW	14	No aplica (idem 6)	✓
WHITE	15	No aplica (idem 7)	✓
BLINK	128	No aplica	***

*** **BLINK** (parpadear) es un parámetro adicional (opcional) para la función **textcolor(BLINK +** N° de color**);** para desplegar o mostrar los caracteres con un cierto parpadeo sin importar la modalidad de color.

BLINK es aplicable únicamente en modo texto, es decir sobre el primer plano. Sin embargo, muchas de estas funciones y parámetros relacionadas con el formato en cuanto a colores, solo son portables para Ms-DOS, trabajando bajo una versión de Turbo C++ por ejemplo.

wherex
Función: Retorna la columna en la que se encuentra el cursor.
Sintaxis: col=wherex( );

wherey
Función: Retorna la fila en la que se encuentra el cursor.
Sintaxis: fila=wherey( );

getch
Función: Lee y retorna un único carácter introducido mediante el teclado por el usuario. No muestra el carácter por la pantalla.
Sintaxis: letra=getch( );

getche
Función: Lee y retorna un único carácter introducido mediante el teclado por el usuario. Muestra el carácter por la pantalla.
Sintaxis: letra=getche( );

Librería string.h

strlen
Función: Calcula la longitud de una cadena.
Sintaxis: longitud=strlen(cadena);

strcpy
Función: Copia el contenido de una cadena sobre otra.
Sintaxis: strcpy(copia , original);

strncpy
Función: Copia en la mayor parte de **n** caracteres de la cadena **original** sobre la cadena **copia**.
Sintaxis: strncpy(copia , original);

strcat
Función: Concatena dos cadenas.
Sintaxis: strcat(cadena1 , cadena2);

strcmp
Función: Compara el contenido de dos cadenas. Si **cadena1** < **cadena2** retorna un número negativo. Si **cadena1** > **cadena2**, un número positivo, y si **cadena1** es igual que **cadena2** retorna **0** (o **NULL**).
Sintaxis: valor=strcmp(cadena1 , cadena2);

strncmp
Función: Compara hasta **n** caracteres de la **cadena1** con la **cadena2**. La función regresa **0**, menos que **0**, o mayor que **0**, si **cadena1** es igual a, menor que o mayor que **adena2**, respectivamente.
Sintaxis: strncmp(cadena1 , cadena2,n);

strchr
Función: Encuentra la primera ocurrencia de un carácter **c** en la cadena **s**. Si el carácter **c** se encuentra, regresa un apuntador al carácter. De lo contrario regresa un apuntador **NULL**.
Sintaxis: strchr(s , c);

strrchr
Función: La función busca la última ocurrencia de un carácter **c** especificado en una cadena **s**. Si encuentra dicho carácter, regresa un apuntador al carácter de la cadena, de lo contrario regresa **NULL**.
Sintaxis: strrchr(s , c);

strstr
Función: Busca la primera ocurrencia de su segundo argumento de cadena, en su primer argumento de cadena. Si **cadena2** se encuentra contenida en **cadena1**, regresa un apuntador a la posición de la cadena en el primer argumento.
Sintaxis: strstr(cadena1 , cadena2);

strtok
Función: Una secuencia de llamadas a ***strtok*** divide la **cadena1** en ***tokens*** (o subcadenas), que están separados por caracteres contenidos en **cadena2**. La primera llamada contiene a **cadena1** como primer argumento, y las llamadas subsecuentes para continuar la división de la misma cadena contienen **NULL** como primer argumento.
Para cada llamada se devuelve un apuntador al ***token*** actual. Si ya no existen más ***tokens*** cuando la función es llamada, regresa un apuntador **NULL**.
Sintaxis: strtok(cadena1 , cadena2);

strdup
Función: Función que duplica una cadena. Regresa un apuntador a hacia el duplicado o hacia la cadena-copia.
Sintaxis: strdup(cadena);

strrev
Función: Función que invierte el orden de los caracteres de una cadena.
Sintaxis: strrev(cadena);

strlwr
Función: Función que convierte una cadena a minúsculas.
Sintaxis: strlwr(cadena);

strupr
Función: Función que convierte una cadena a mayúsculas.
Sintaxis: strupr(cadena);

strset
Función: Función que modifica todos los caracteres de una cadena a un carácter específico.
Sintaxis: strset(cadena);

Librería math.h

Funciones matemáticas.

Dentro del fichero de cabecera "**math.h**" tenemos acceso a muchas funciones matemáticas predefinidas en C, como:

acos(x): Arco coseno ó coseno inverso de un ángulo
asin(x): Arco seno ó seno inverso de un ángulo
atan(x): Arco tangente ó tangente inversa de un ángulo
atan2(y,x): Arco tangente de y/x (por si x o y son 0)
ceil(x): El valor entero superior a x y más cercano a él
cos(x): Coseno de un ángulo
cosh(x): Coseno hiperbólico de un ángulo
exp(x): Exponencial o antilogaritmo natural de x (e elevado a x)
fabs(x): Valor absoluto

floor(x): El valor mayor entero que es menor que x
fmod(x,y): Resto o residuo de la división x/y
log(x): Logaritmo natural (o neperiano, en base e)
log10(x): Logaritmo en base 10
pow(x,y): x elevado a y
pow10(n): calcula 10 a la potencia n
sin(x): Seno de un ángulo
sinh(x): Seno hiperbólico de un ángulo
sqrt(x): Raíz cuadrada de su argumento
tan(x): Tangente de un ángulo
tanh(x): Tangente hiperbólica de un ángulo
ceil(x): Redondea un número de punto flotante a entero hacia arriba.
labs(x): Obtiene el valor absoluto de un entero largo
modf(x): Descompone un número de punto flotante en parte entera y parte decimal

(todos ellos usan parámetros **X** de tipo "double") y una serie de constantes como:

M_E, el número "**e**", con un valor de **2.71828...**
M_PI, el número "**Pi**", **3.14159...**

NOTA: En **C++**, tendremos también posiblemente definida una clase "complex", para números complejos (con dos componentes, **X** e **Y**).

Funciones Interesantes

fflush(stdin)
Función: Limpia el ***buffer*** (memoria temporal) para el teclado.
Sintaxis: fflush(stdin);
Prototipo: stdio.h

sizeof
Función: Operador que retorna el tamaño en ***bytes*** de una variable.
Sintaxis: tamaño=sizeof(variable);

cprintf
Función: Funciona como el **printf** pero escribe en el color que hayamos activado con la función **textcolor** sobre el color activado con **textbackground**.
Sintaxis: cprintf(formato , arg1 , ...);
Prototipo: conio.h

kbhit
Función: Espera la pulsación de una tecla para continuar la ejecución.
Sintaxis: while (!kbhit( )) /* Mientras no pulsemos una tecla... */
Prototipo: conio.h

random
Función: Retorna un valor aleatorio entre 0 y num-1.
Sintaxis: valor=random(num); /* También necesitamos la función **randomize** */
Prototipo: stdlib.h

randomize
Función: Inicializa el generador de números aleatorios. Deberemos llamarlo al inicio de la función en que utilicemos el **random**. También deberemos utilizar el include **time.h**, ya que **randomize** hace una llamada a la función **time**, incluída en este último archivo.
Sintaxis: randomize( );
Prototipo: stdio.h

free
Función: Libera bloques de memoria asignados.
Sintaxis: free(apuntador a memoria);
Prototipo: stdlib.h

system
Función: Ejecuta el comando indicado. Esto incluye tanto los comandos del sistema operativo, como cualquier programa que nosotros le indiquemos. Al acabar la ejecución del comando, volverá a la línea de código situada a continuación de la sentencia **system**.
Sintaxis: system(comando); /* p.ej: system("arj a programa"); */
Prototipo: stdlib.h

Nota Importante:
Muchas de las funciones expuestas anteriormente NO cumplen con el estándar de ANSI C, por lo que en un momento dado puede que no presenten portabilidad en otros sistemas, es decir, que no funcionen, o que ni si quiera sean reconocidas en diversas plataformas. Así por ejemplo, la librería conio.h , y por default todas las funciones que define, NO son compatibles con sistemas que trabajen bajo plataforma UNIX.