

Programación Orientada a Objetos

Caracteres y Cadenas

Agenda

- Conceptos Básicos de caracteres y cadenas
- Librería de Manejo de Caracteres (ctype.h)
 - isalpha, isupper, islower, isdigit, isspace, ispunct, isalnum, toupper, tolower

Conceptos Básicos de caracteres y cadenas

- Constantes carácter
 - Valor **int** representado por un carácter entre comillas simples
- Cadena de caracteres
 - Conjunto de caracteres tratados como una sola unidad.
 - Incluye letras, números y caracteres especiales (+, -, *, / y \$)
 - Se escriben entre comillas dobles
- En C una cadena de caracteres es un arreglo de caracteres que terminan con el carácter nulo (`'\0'`)

Conceptos Básicos de caracteres y cadenas

- El valor de la cadena es la dirección del primer carácter
 - Una cadena es un puntero al primer carácter de la cadena
 - `char color[] = "azul";`
 - `const char *ptrColor = "azul";`
 - `char color[] = {'a','z','u','l','\0'};`
 - Omitir el carácter nulo al final de una cadena es un error
 - Imprimir una cadena que no contenga el carácter nulo, es un error
 - Capturar una cadena de caracteres
 - `char palabra[20];`
 - `scanf("%s", palabra);`
 - `cin >> palabra;`

Librería de Manejo de Caracteres (ctype.h)

Prototipo	Descripción
<code>int isdigit(int c);</code>	Devuelve un valor verdadero si c es un dígito; de lo contrario devuelve 0 (falso)
<code>int isalpha(int c);</code>	Devuelve un valor verdadero si c es una letra; de lo contrario devuelve 0 (falso)
<code>int isalnum(int c);</code>	Devuelve un valor verdadero si c es un dígito ó una letra; de lo contrario devuelve 0 (falso)
<code>int isxdigit(int c);</code>	Devuelve un valor verdadero si c es un dígito hexadecimal; de lo contrario devuelve 0 (falso)
<code>int islower(int c);</code>	Devuelve un valor verdadero si c es una letra minúscula; de lo contrario devuelve 0 (falso)
<code>int isupper(int c);</code>	Devuelve un valor verdadero si c es una letra mayúscula; de lo contrario devuelve 0 (falso)

Librería de Manejo de Caracteres (ctype.h)

Prototipo	Descripción
<code>int tolower (int c);</code>	Si <code>c</code> es una letra mayúscula, <code>tolower</code> devuelve <code>c</code> como una letra minúscula. De lo contrario, devuelve el argumento sin cambios.
<code>int toupper(int c);</code>	Si <code>c</code> es una letra minúscula, <code>toupper</code> devuelve <code>c</code> como una letra mayúscula. De lo contrario, devuelve el argumento sin cambios.
<code>int isspace(int c);</code>	Devuelve un valor verdadero si <code>c</code> es un carácter de espacio en blanco, nueva línea (<code>'\n'</code>), espacio (<code>' '</code>), avance de página (<code>'\f'</code>), retorno de carro (<code>'\r'</code>), tabulador horizontal(<code>'\t'</code>), tabulador vertical (<code>'\v'</code>); de lo contrario devuelve 0
<code>int iscntrl(int c);</code>	Devuelve un valor verdadero si <code>c</code> es un carácter de control; de lo contrario devuelve 0 (falso)

Librería de Manejo de Caracteres (ctype.h)

Prototipo	Descripción
<code>int ispunct(int c);</code>	Devuelve un valor verdadero si <code>c</code> es un carácter de impresión diferente de un espacio, un dígito o una letra; de lo contrario devuelve 0 (falso).
<code>int isprint(int c);</code>	Devuelve un valor verdadero si <code>c</code> es un carácter de impresión, incluso el espacio (' '); de lo contrario devuelve 0.
<code>int isspace(int c);</code>	Devuelve un valor verdadero si <code>c</code> es un carácter de impresión, diferente de espacio (' '); de lo contrario devuelve 0.

Funciones de conversión de cadenas (stdlib.h)

Prototipo	Descripción
<code>double atof (const char *ptrN)</code>	Convierte la cadena ptrN a double
<code>int atoi(const char *ptrN)</code>	Convierte la cadena ptrN a int
<code>long atol(const char *ptrN)</code>	Convierte la cadena ptrN a long int
<code>double strtod(const char *ptrN, char **ptrFinal)</code>	Convierte la cadena ptrN a double
<code>long strtol(const char *ptrN, char **ptrFinal, int base)</code>	Convierte la cadena ptrN a long
<code>unsigned long strtoul(const char *ptrN, char **ptrFinal, int base)</code>	Convierte la cadena ptrN a unsigned long

Manejo de cadenas (string.h)

- Biblioteca de manipulación de cadenas -- <string.h>
- Conjunto de Funciones para:
 - Manipular cadenas (copiar y concatenar)
 - Comparar cadenas
 - Buscar caracteres
 - Buscar una cadena dentro de otra
 - Separar cadenas en tokens
 - Longitud de cadena

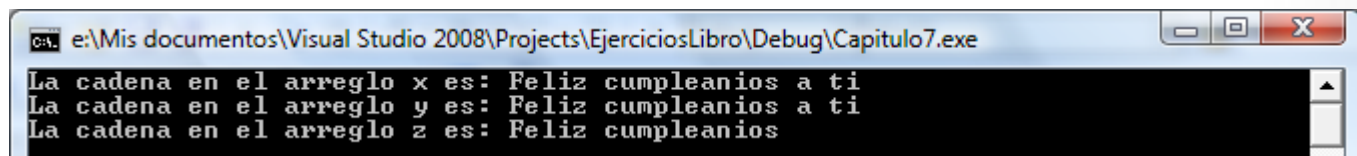
Librería de manejo de cadenas (string.h)

Prototipo	Descripción
<code>char *strcpy(char *s1, const char *s2)</code>	Copia la cadena s2 dentro del arreglo s1
<code>char *strncpy(char *s1, const char *s2, size_t n)</code>	Copia al menos n caracteres de la cadena s2 dentro del arreglo s1. Devuelve el valor de s1
<code>char *strcat(char *s1, const char *s2)</code>	Agrega la cadena s2 al arreglo s1. El primer carácter de s2 sobrescribe al carácter de terminación nulo de s1. Devuelve el valor de s1.
<code>char *strncat(char *s1, const char *s2, size_t n)</code>	Agrega al menos n caracteres de la cadena s2 al arreglo s1. El primer carácter de s2 sobrescribe al carácter de terminación nulo de s1. Devuelve el valor de s1.

Copiar cadenas

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>
#include <iostream>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    char x[]="Feliz cumpleaños a ti"; // Inicializa el arreglo de caracteres
    char y[25]; //Crea arreglo de caracteres y
    char z[18]; //Crea arreglo de caracteres z
    /* contenido de la copia de x dentro de y */
    printf ( "%s%s\n%s%s\n", "La cadena en el arreglo x es: ", x,
    "La cadena en el arreglo y es: ", strcpy( y, x ) );
    /* copia los primeros 17 caracteres de x dentro z. No copian el
    caracter nulo */
    strncpy( z, x, 17 );
    z[17]= '\0'; /* termina la cadena z */
    printf( "La cadena en el arreglo z es: %s\n", z );
    std::cout << "La cadena en el arreglo z es: " << z << std::endl;
    return a.exec();
}
```



```
e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe
La cadena en el arreglo x es: Feliz cumpleaños a ti
La cadena en el arreglo y es: Feliz cumpleaños a ti
La cadena en el arreglo z es: Feliz cumpleaños
```

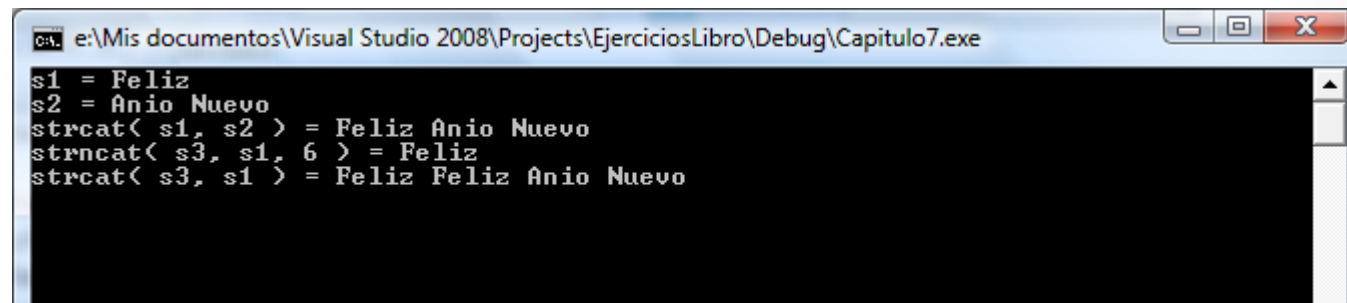
Concatenar cadenas

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    char s1[20] = "Feliz "; // inicializa el arreglo de caracteres s1
    char s2[] = "Año Nuevo "; // inicializa el arreglo de caracteres s2
    char s3[40]=""; // inicializa a vacío el arreglo de caracteres s3
    printf( "s1 %s\ns2 %s\n", s1, s2 ); // concatena s2 y s1
    printf( "strcat( s1, s2 ) = %s\n", strcat( s1, s2 ) );
    /* concatena los primeros 6 caracteres de s1 a s3. Coloque \0'
    después del último carácter */
    printf( "strncat( s3, s1, 6 ) = %s\n", strncat( s3, s1, 6 ) );
    // concatena s1 a s3
    printf( strcat( s3, s1 ), "%s\n", strcat( s3, s1 ) );

    return a.exec();
}
```



The screenshot shows a Windows command prompt window with the title bar "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The window contains the following output:

```
s1 = Feliz
s2 = Año Nuevo
strcat( s1, s2 ) = Feliz Año Nuevo
strncat( s3, s1, 6 ) = Feliz
strcat( s3, s1 ) = Feliz Feliz Año Nuevo
```

Librería de manejo de cadenas (string.h)

Prototipo	Descripción
<code>int strcmp(const char *s1, const char *s2);</code>	Compara la cadena s1 con la cadena s2. La función devuelve 0, menor que 0, o mayor que 0, si s1 es igual, menor, o mayor que s2, respectivamente.
<code>int strncmp(const char *s1, const char *s2, size_t n);</code>	Compara hasta n caracteres de la cadena s1 con la cadena s2. La función devuelve 0, menor que 0, o mayor que 0, si s1 es igual, menor, o mayor que s2, respectivamente.

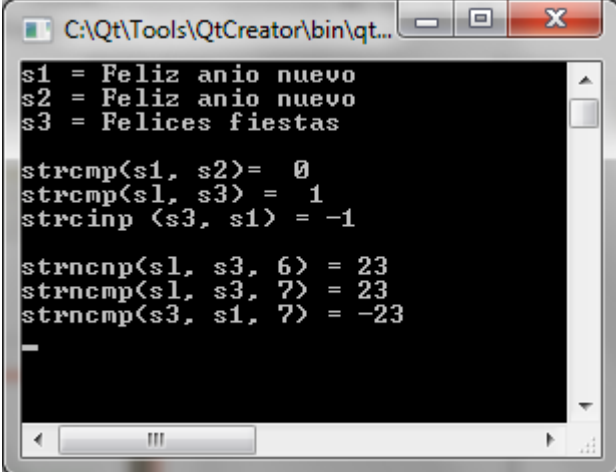
Comparar cadenas

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    const char *s1= "Feliz anio nuevo"; // inicializa el apuntador a char 'I
    const char *s2= "Feliz anio nuevo"; // inicializa el apuntador a char /
    const char *s3="Felices fiestas"; // iniciaiiza ei apuntador a char /
    printf("%s%s\n%s%s\n%s%s\n\n%s%2d\n%s%2d\n%s%2d\n\n",
        "s1 = ", s1, "s2 = ", s2, "s3 = ", s3,
        "strcmp(s1, s2)= ", strcmp( s1, s2 ),
        "strcmp(s1, s3) = ", strcmp( s1, s3 ),
        "strcinp (s3, s1) = ", strcmp( s3, s1 ) );

    printf ("%s%2d\n%s%2d\n%s%2d\n",
        "strncmp(s1, s3, 6) = ", strncmp( s1, s3, 6 ),
        "strncmp(s1, s3, 7) = ", strncmp( s1, s3, 7 ),
        "strncmp(s3, s1, 7) = ", strncmp( s3, s1, 7 ) );

    return a.exec();
}
```



The screenshot shows a Qt Creator console window with the following output:

```
s1 = Feliz anio nuevo
s2 = Feliz anio nuevo
s3 = Felices fiestas

strcmp(s1, s2)= 0
strcmp(s1, s3) = 1
strcinp (s3, s1) = -1

strncmp(s1, s3, 6) = 23
strncmp(s1, s3, 7) = 23
strncmp(s3, s1, 7) = -23
```

Librería de manejo de cadenas (string.h)

Prototipo	Descripción
<code>char *strchr(const char *s, int c);</code>	Localiza la primera ocurrencia del carácter <code>c</code> en la cadena <code>s</code> . Si se localiza a <code>c</code> , se devuelve un apuntador a <code>c</code> en <code>s</code> . De lo contrario devuelve <code>NULL</code> .
<code>size_t strcspn(const char *s1, const char *s2);</code>	Determina y devuelve la longitud del segmento inicial de la cadena <code>s1</code> , que consiste en los caracteres no contenidos en la cadena <code>s2</code> .
<code>size_t strspn(const char *s1, const char *s2);</code>	Determina y devuelve la longitud del segmento inicial de la cadena <code>s1</code> , que consiste sólo en los caracteres contenidos en la cadena <code>s2</code> .
<code>char *strpbrk(const char *s1, const char *s2);</code>	Localiza la primera ocurrencia en la cadena <code>s1</code> de cualquier carácter de la cadena <code>s2</code> . Si localiza un carácter de la cadena <code>s2</code> , se devuelve un apuntador al carácter de la cadena <code>s1</code> . Caso contrario devuelve <code>NULL</code> .

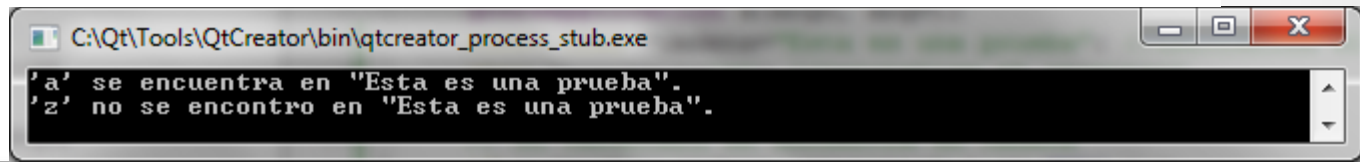
Librería de manejo de cadenas (string.h)

Prototipo	Descripción
<code>char *strchr(const char *s, int c);</code>	Localiza la última ocurrencia de <code>c</code> en la cadena <code>s</code> . Si se localiza a <code>c</code> , se devuelve un apuntador a <code>c</code> en la cadena <code>s</code> . De lo contrario, se devuelve un apuntador <code>NULL</code> .
<code>char *strstr(const char *s1, const char *s2);</code>	Localiza la primera ocurrencia en la cadena <code>s1</code> de la cadena <code>s2</code> . Si se localiza la cadena, se devuelve un apuntador a la cadena en <code>s1</code> . De lo contrario, se devuelve un apuntador <code>NULL</code> .
<code>char *strtok(const char *s1, const char *s2);</code>	Una secuencia de llamadas <code>strtok</code> separa la cadena <code>s1</code> en “tokens” separados por caracteres contenidos en la cadena <code>s2</code> . La primera llamada contiene <code>s1</code> como el primer argumento, y las llamadas siguientes contienen a <code>NULL</code> como el primer elemento para continuar separando la misma cadena. Un apuntador al token actual es devuelto por cada llamada. Si no hay más tokens cuando se llama a la función, se devuelve <code>NULL</code> .

Buscar un carácter en una cadena

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    const char *cadena="Esta es una prueba"; //inicializa el apuntador a char *
    char caracter1='a'; //inicializa el caracter1
    char caracter2='z'; // inicializa el caracter2
    // si caracter1 se encuentra en cadena
    if ( strchr( cadena, caracter1 ) != NULL ) {
        printf( "\\'%c\\' se encuentra en \"%s\".\n", caracter1, cadena );
    } // fin de if
    else { // si no se encuentra caracter1
        printf( "\\'%c\\' no se encontro en \"%s\".\n", caracter1, cadena );
    } // fin de else
    // si caracter2 se encuentra en cadena
    if ( strchr( cadena, caracter2 ) != NULL ) {
        printf( "\\'%c\\' se encontro en \"%s\".\n", caracter2, cadena );
    } // fin de if
    else { // si no se encontro caracter2
        printf( "\\'%c\\' no se encontro en \"%s\".\n", caracter2, cadena );
    } // fin de else
    return a.exec();
}
```



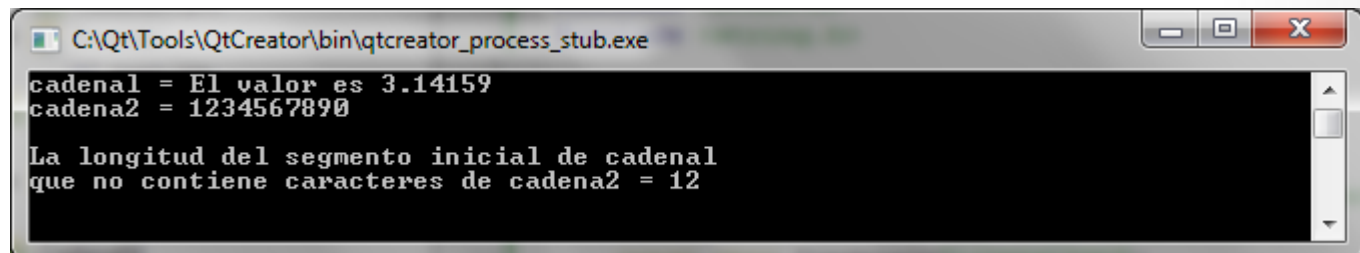
C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe

```
'a' se encuentra en "Esta es una prueba".
'z' no se encontro en "Esta es una prueba".
```

Uso de strchr

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    //Inicialización de las variables char* a crear
    const char *cadena1="El valor es 3.14159";
    const char *cadena2="1234567890";
    printf ( "%s%s\n%s%s\n\n%s\n%s%u",
        "cadena1 = ", cadena1, "cadena2 = ", cadena2,
        "La longitud del segmento inicial de cadena1",
        "que no contiene caracteres de cadena2 = ",
        strchr( cadena1, cadena2 ) );
    return a.exec();
}
```



The screenshot shows a console window titled "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The output text is as follows:

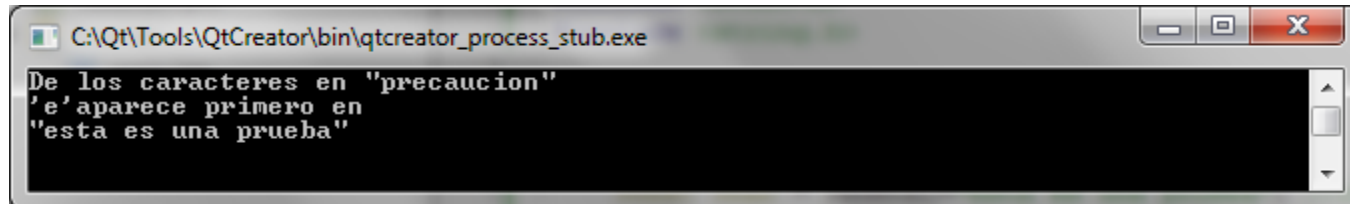
```
cadena1 = El valor es 3.14159
cadena2 = 1234567890

La longitud del segmento inicial de cadena1
que no contiene caracteres de cadena2 = 12
```

Uso de strpbrk

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    const char * cadenal="esta es una prueba"; //inicializa el apuntador a char *
    const char * cadena2="precaucion"; // inicializa el apuntador a char *
    printf( "%s\\%s\\\"\\n'%c'\\%s\\\"\\%s\\\"\\n",
        "De los caracteres en ", cadena2,
        *strpbrk( cadenal,cadena2),
        "aparece primero en ", cadenal );
    return a.exec();
}
```



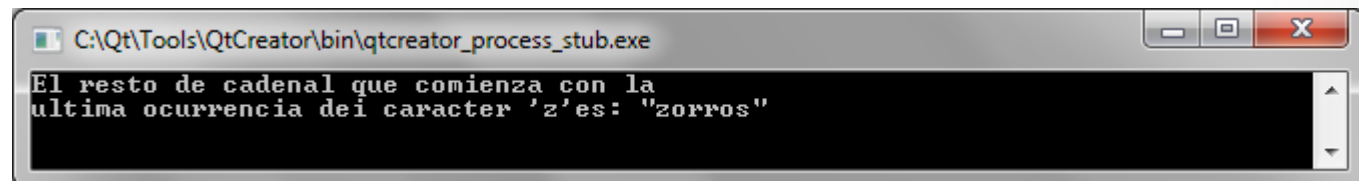
The screenshot shows a console window titled "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The output text is as follows:

```
De los caracteres en "precaucion"
'e' aparece primero en
"esta es una prueba"
```

Uso de strrchr

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    // inicializa el apuntador a char *
    const char * cadenal= "Un zoologico tiene muchos animales incluso los zorros";
    char e='z'; // caracter a buscar
    printf( "%s\n%s'%c'%s\n"%s\n",
    "El resto de cadenal que comienza con la",
    "ultima ocurrencia dei caracter ", e,
    "es: ", strrchr( cadenal, e ) );
    return a.exec();
}
```

A screenshot of a Qt Creator console window. The title bar shows the file path "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The console output displays the result of the strrchr function call: "El resto de cadenal que comienza con la ultima ocurrencia dei caracter 'z'es: "zorros"". The text is white on a black background.

```
C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe
El resto de cadenal que comienza con la
ultima ocurrencia dei caracter 'z'es: "zorros"
```

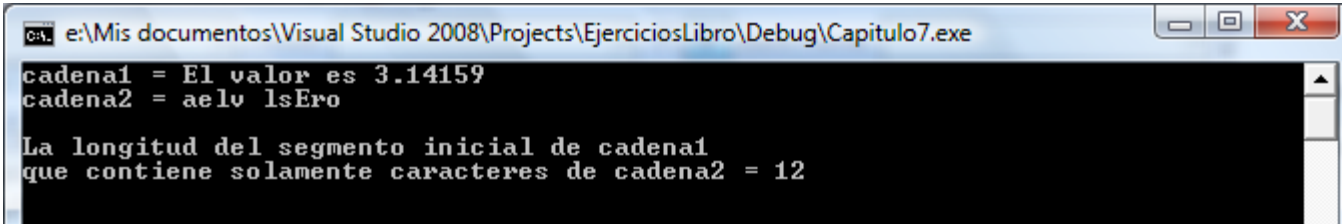
Uso de strspn

```
#include <stdio.h>
#include <string.h>

int main()
{
    /* inicializa dos apuntadores a char */
    const char *cadena1 = "El valor es 3.14159";
    const char *cadena2 = "aelv lsEro";

    printf( "%s%s\n%s%s\n\n%s\n%s%u\n",
        "cadena1 = ", cadena1, "cadena2 = ", cadena2,
        "La longitud del segmento inicial de cadena1",
        "que contiene solamente caracteres de cadena2 = ",
        strspn( cadena1, cadena2 ) );

    return 0; /* indica terminación exitosa */
} /* fin de main */
```



The screenshot shows a Windows command prompt window with the title bar "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The window contains the following output:

```
cadena1 = El valor es 3.14159
cadena2 = aelv lsEro

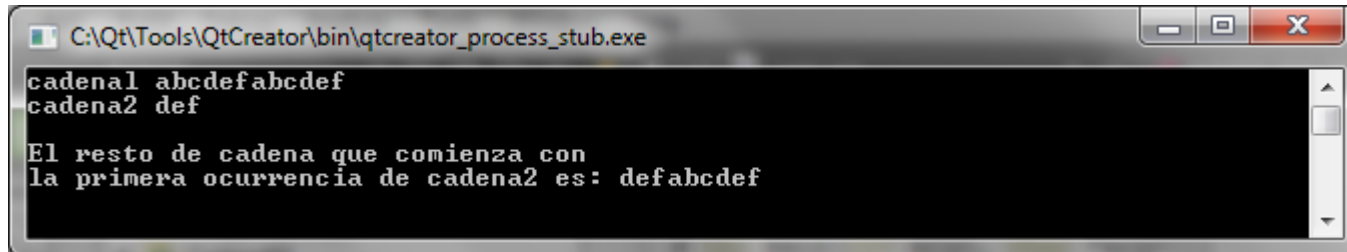
La longitud del segmento inicial de cadena1
que contiene solamente caracteres de cadena2 = 12
```

Uso de strstr

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    const char * cadena1="abcdefabcdef"; // cadena de búsqueda
    const char * cadena2="def"; // cadena a buscar
    printf ( "%s%s\n%s%s\n\n%s\n%s%s\n",
        "cadena1 ", cadena1, "cadena2 ", cadena2,
        "El resto de cadena que comienza con",
        "la primera ocurrencia de cadena2 es: ",
        strstr( cadena1, cadena2 ) );

    return a.exec();
}
```



The screenshot shows a console window titled "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The output of the program is as follows:

```
cadena1 abcdefabcdef
cadena2 def

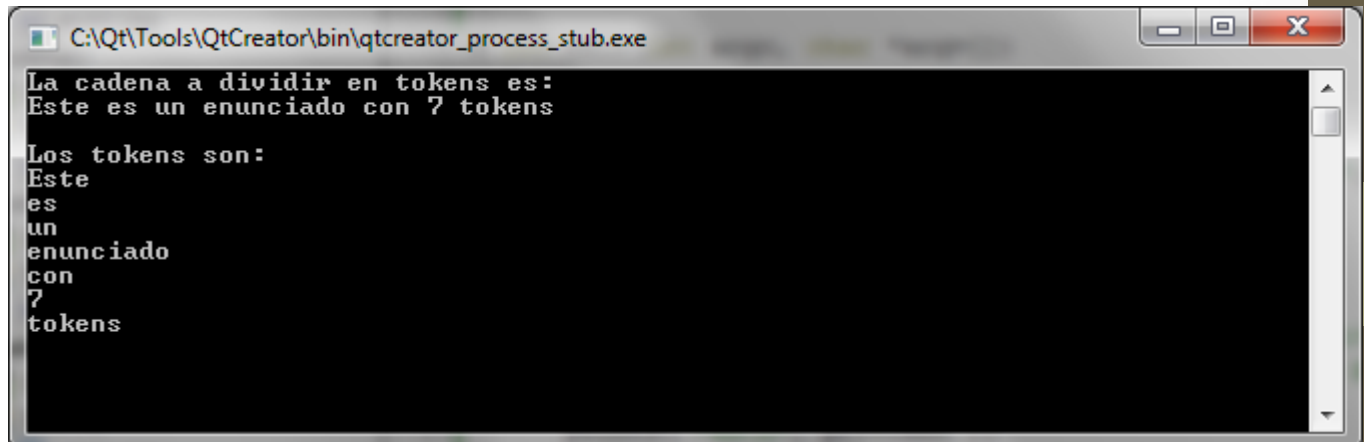
El resto de cadena que comienza con
la primera ocurrencia de cadena2 es: defabcdef
```

strtok

```
#include <QCoreApplication>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);
    // inicializa el arreglo de cadena
    char cadena[]="Este es un enunciado con 7 tokens";
    char * ptrToken; // crea un apuntador char *
    printf ( "%s\n%s\n\n%s\n",
        "La cadena a dividir en tokens es:", cadena,
        "Los tokens son: " );
    ptrToken=strtok( cadena, " " ); // comienza la división en tokens dei enunciado
    // continúa la visualización en tokens hasta que ptrtoken se hace NULL
    while ( ptrToken != NULL ) {
        printf( "%s\n", ptrToken );
        ptrToken =strtok( NULL, " " ); // obtiene ei siguiente token
    } // fin de while

    return a.exec();
}
```



The screenshot shows a Qt Creator console window titled "C:\Qt\Tools\QtCreator\bin\qtcreator_process_stub.exe". The output of the program is displayed in the console area, which has a black background and white text. The output matches the code's intent: it prints the string to be tokenized and then lists its tokens on separate lines.

```
La cadena a dividir en tokens es:
Este es un enunciado con 7 tokens

Los tokens son:
Este
es
un
enunciado
con
7
tokens
```

Funciones de memoria de la biblioteca de manipulación de cadenas

Prototipo	Descripción
<code>void *memcpy(void *s1, const void *s2, size_t n);</code>	Copia n caracteres desde el objeto al que se apunta s2, dentro del objeto al que apunta s1. Devuelve un apuntador al objeto resultante.
<code>void *memmove(void *s1, const void *s2, size_t n);</code>	Copia n caracteres desde el objeto al que apunta s2 dentro del objeto al que apunta s1. La copia se lleva a cabo como si los caracteres primero se copiaran desde el objeto al que apunta s2 en un arreglo temporal y después desde el arreglo temporal hacia el objeto al que apunta s1. Devuelve un apuntador al objeto resultante.
<code>void *memcmp(const void *s1, const void *s2, size_t n);</code>	Compara los n caracteres de los objetos a los que apuntan s1 y s2. La función devuelve un número igual, menor o mayor que 0 si s1 es igual, menor o mayor que s2.

Funciones de memoria de la biblioteca de manipulación de cadenas

Prototipo	Descripción
<code>void *memchr(const void *s, int c, size_t n);</code>	Localiza la primera ocurrencia de <code>c</code> (convertida a <code>unsigned char</code>) en los primeros <code>n</code> caracteres del objeto al que apunta <code>s</code> . Si se encuentra <code>c</code> , devuelve un apuntador a <code>c</code> . De lo contrario, devuelve <code>NULL</code> .
<code>void *memset(void *s, int c, size_t n);</code>	Copia <code>c</code> (convertido en <code>unsigned char</code>) dentro de los primeros <code>n</code> caracteres del objeto al que apunta <code>s</code> . Devuelve un apuntador al resultado.

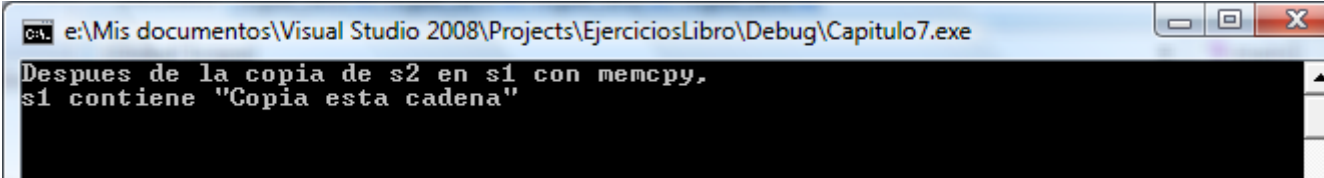
Uso de memcpy

```
#include <stdio.h>
#include <string.h>

int main()
{
    char s1[ 18 ];                /* crea el arreglo de caracteres s1 */
    char s2[] = "Copia esta cadena"; /* inicializa el arreglo de caracteres s2 */

    memcpy( s1, s2, 18 );
    printf( "%s\n%s\n%s\n",
           "Despues de la copia de s2 en s1 con memcpy,",
           "s1 contiene ", s1 );
    getchar();
    return 0; /* indica terminación exitosa */

} /* fin de main */
```



The screenshot shows a Windows command prompt window with the title bar "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The window contains the following text:

```
Despues de la copia de s2 en s1 con memcpy,
s1 contiene "Copia esta cadena"
```

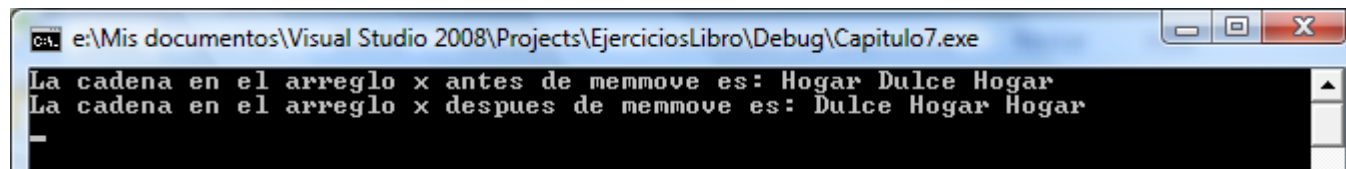
Uso de memmove

```
#include <stdio.h>
#include <string.h>

int main()
{
    char x[] = "Hogar Dulce Hogar"; /* inicializa el arreglo de caracteres x */

    printf( "%s\n", "La cadena en el arreglo x antes de memmove es: ", x );
    printf( "%s\n", "La cadena en el arreglo x despues de memmove es: ",
            memmove( x, &x[ 6 ], 11 ) );

    return 0; /* indica terminación exitosa */
} /* fin de main */
```



The screenshot shows a Windows command prompt window with the title bar "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The window contains two lines of text: "La cadena en el arreglo x antes de memmove es: Hogar Dulce Hogar" and "La cadena en el arreglo x despues de memmove es: Dulce Hogar Hogar". The text is displayed in a monospaced font on a black background.

Uso de memcmp

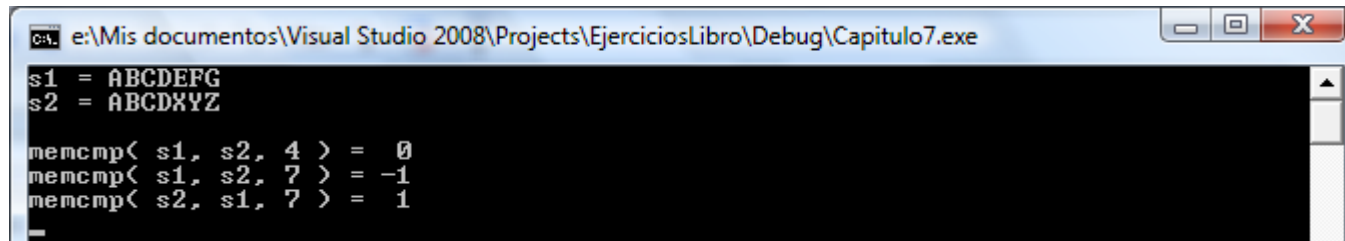
```
#include <stdio.h>
#include <string.h>

int main()
{
    char s1[] = "ABCDEFGH"; /* inicializa el arreglo de caracteres s1 */
    char s2[] = "ABCDXYZ"; /* inicializa el arreglo de caracteres s2 */

    printf( "%s%s\n%s%s\n\n%s%2d\n%s%2d\n%s%2d\n",
           "s1 = ", s1, "s2 = ", s2,
           "memcmp( s1, s2, 4 ) = ", memcmp( s1, s2, 4 ),
           "memcmp( s1, s2, 7 ) = ", memcmp( s1, s2, 7 ),
           "memcmp( s2, s1, 7 ) = ", memcmp( s2, s1, 7 ) );

    return 0; /* indica terminación exitosa */

} /* fin de main */
```



The screenshot shows a command prompt window titled "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The output of the program is displayed as follows:

```
s1 = ABCDEFGH
s2 = ABCDXYZ

memcmp( s1, s2, 4 ) = 0
memcmp( s1, s2, 7 ) = -1
memcmp( s2, s1, 7 ) = 1
```

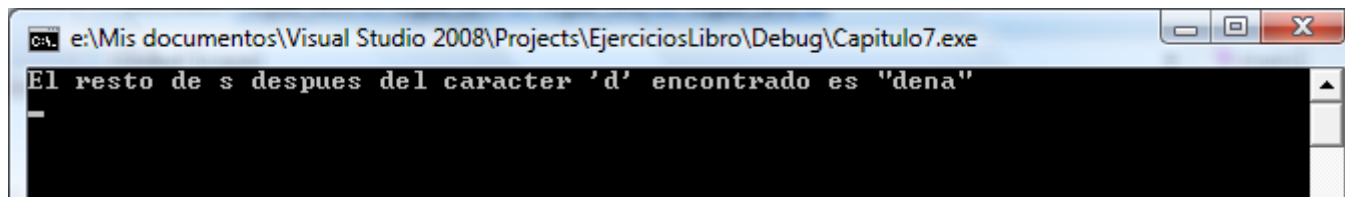
Uso de memchr

```
#include <stdio.h>
#include <string.h>

int main()
{
    const char *s = "Esta es una cadena"; /* inicializa el apuntador a */

    printf( "%s\\'%c\\'%s\\'%s\\\"\\n",
            "El resto de s despues del caracter ", 'd',
            " encontrado es ", memchr( s, 'd', 16 ) );

    return 0; /* indica terminación exitosa */
} /* fin de main */
```

A screenshot of a Windows command prompt window. The title bar shows the file path: "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The command prompt displays the output of the program: "El resto de s despues del caracter 'd' encontrado es "dena"". The text is in a monospaced font on a black background. There is a small cursor at the end of the first line of output.

```
C:\e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe
El resto de s despues del caracter 'd' encontrado es "dena"
```

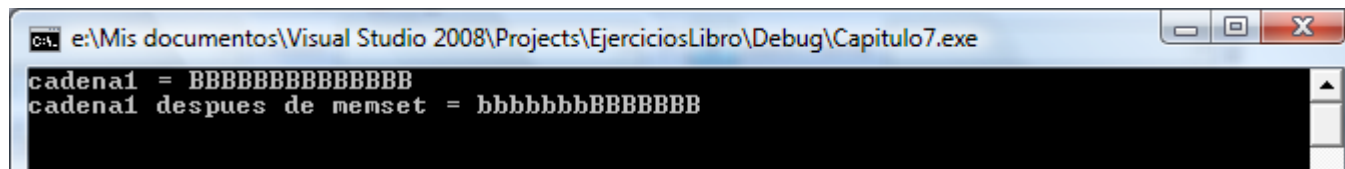
Uso de memset

```
#include <stdio.h>
#include <string.h>

int main()
{
    char cadena1[ 15 ] = "BBBBBBBBBBBBBBB"; /* inicializa cadena1 */

    printf( "cadena1 = %s\n", cadena1 );
    printf( "cadena1 despues de memset = %s\n", memset( cadena1, 'b', 7 ) );

    return 0; /* indica terminación exitosa */
} /* fin de main */
```



The screenshot shows a Windows command prompt window titled "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The window contains the following output:

```
cadena1 = BBBBBBBBBBBBBB
cadena1 despues de memset = hhhhhhhBBBBBBB
```

Funciones de manipulación de cadenas

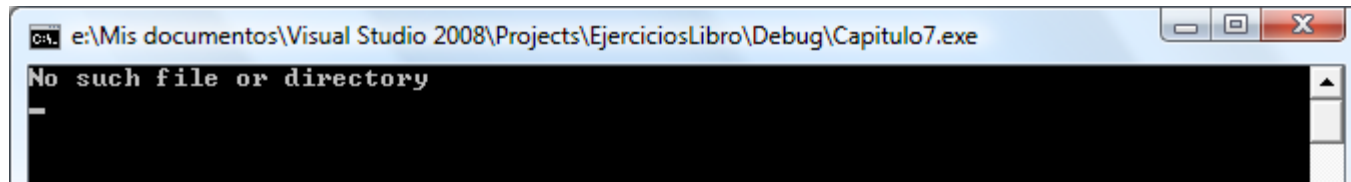
Prototipo	Descripción
<code>char*strerror(int errornum);</code>	Obtiene mediante errornum una cadena de texto del error de manera dependiente de la máquina. Devuelve un apuntador a la cadena.
<code>size_t strlen(const char *s);</code>	Determina la longitud de la cadena s. Devuelve el numero de caracteres que preceden al carácter de terminación nulo.

Uso de strerror

```
#include <stdio.h>
#include <string.h>

int main()
{
    printf( "%s\n", strerror( 2 ) );

    return 0; /* indica terminación exitosa */
} /* fin de main */
```



The screenshot shows a Windows command prompt window with the title bar "e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe". The window contains the text "No such file or directory" followed by a cursor on a new line. This is the output of the program, which calls strerror(2) to get the string representation of the error code 2 (ENOENT).

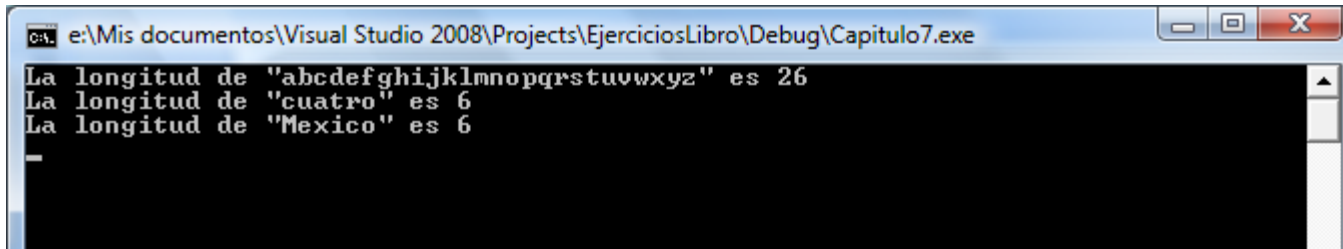
Uso de strlen

```
#include <stdio.h>
#include <string.h>

int main()
{
    /* inicializa los 3 apuntadores a char */
    const char *cadena1 = "abcdefghijklmnopqrstuvwxyz";
    const char *cadena2 = "cuatro";
    const char *cadena3 = "Mexico";

    printf("%s\\\"%s\\\"%s%lu\\n%s\\\"%s\\\"%s%lu\\n%s\\\"%s\\\"%s%lu\\n",
        "La longitud de ", cadena1, " es ", (unsigned long) strlen( cadena1 ),
        "La longitud de ", cadena2, " es ", (unsigned long) strlen( cadena2 ),
        "La longitud de ", cadena3, " es ", (unsigned long) strlen( cadena3 ) );

    return 0; /* indica terminación exitosa */
} /* end main */
```



```
C:\> e:\Mis documentos\Visual Studio 2008\Projects\EjerciciosLibro\Debug\Capitulo7.exe
La longitud de "abcdefghijklmnopqrstuvwxyz" es 26
La longitud de "cuatro" es 6
La longitud de "Mexico" es 6
```

Funciones de entrada/salida de la biblioteca estándar

Prototipo	Descripción
<code>int getchar(void)</code>	Lee el siguiente carácter de la entrada estándar y lo devuelve como un entero
<code>char *gets(char *s);</code>	Lee el siguiente carácter de la entrada estándar y lo coloca en el arreglo <code>s</code> hasta que encuentra un carácter de nueva línea o fin de línea o de fin de archivo. Agregar un carácter de terminación nulo al arreglo
<code>int putchar(int c);</code>	Imprime el carácter almacenado en <code>c</code>
<code>int puts(const char *s)</code>	Imprimir la cadena <code>s</code> seguida por el carácter de nueva línea.
<code>int sprintf(char *s, const char *formato,...);</code>	Equivalente a <code>printf</code> , excepto que la salida se almacena en el arreglo <code>s</code> , en lugar leerlo desde el teclado.
<code>int sscanf(char *s, const char *formato, ...);</code>	Equivalente a <code>scanf</code> , excepto que la entrada se lee desde el arreglo <code>s</code> , en lugar leerlo desde el teclado

Bibliografía

- Fundamentos de programación Cadenas y caracteres