

LENGUAJE DE PROGRAMACION - Caracteres, Cadenas y Textos

Aunque se hable de que hay un lenguaje propio de programación de Arduino, esto no es del todo cierto, ya que la programación se hace en C++, adicionalmente, Arduino ofrece un paquete de librerías, también llamado core, que facilitan la programación de los pines de entrada, salida y de los puertos de comunicación, así como otras librerías, para operaciones específicas. En muchos casos el propio IDE, ya incluye estas librerías de forma automática y no es necesario declararlas expresamente. Otra diferencia respecto al C++ standard, es la estructura del programa, ya que no usa la función "main()", sino que con Arduino usamos las funciones "setup()" y "loop()".

En la mayoría de los casos, se puede hacer un proyecto Arduino de cierta complejidad con la librería que nos ofrece el core y no es necesario añadir más instrucciones ni tipos de datos que los que hay en el core. Pero cuando queremos algo más complejo o más rápido o menos consumo de memoria, etc... es necesario usar funciones y estructuras no disponibles en el core de Arduino. Y acá es donde nos valemos de librerías externas.

RECONOCIMIENTO DE CARACTERES

A continuación veremos una serie de funciones, que comenzaremos a usar en el momento que enviemos órdenes a nuestro dispositivo, ya sea comandos enviados por el puerto serie, leyendo un archivo con ordenes pregrabadas (Secuencia de comandos) y/o archivos de inicialización.

Cuando leemos archivos, leemos caracteres, así que primero veremos como reconocerlos, así sabremos que hacer con ellos, luego armaremos y/o leeremos las cadenas de caracteres (string) y ahí veremos las funciones para manejar y procesarlas (strings). Acá les Dejo las funciones que podemos usar para el reconocimientos de caracteres y un ejemplo para cada uno.



IMPORTANTE: el reconocimiento de caracteres, puede realizarse por medio del código **Ascii** de cada uno (En todos los ejemplos se muestra el código **Ascii** de cada carácter analizado). No siempre es imprescindible usar las funciones que provee el IDE de Arduino.

isDigit(dato)

Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un dígito (número). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.

Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char"

Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.

(Programa "110_Texto_01_Reconoce_Caracter_isDigit_A")

```
1 char dato;
2 int d;
3
4 void setup(){
5   Serial.begin(9600);
6   while (!Serial) {
7     ; // Esperamos que el puerto serie este abierto.
8   }
9   Serial.println("Envíe un carácter por el Puerto Serie");
10
11 void loop () {
12   if (Serial.available() > 0) {
13     dato = Serial.read(); //Se lee dato / carácter enviado.
14     d = dato;
15   }
16   Serial.print("Este carácter: ");
```

	<pre> 15 if(isDigit(dato)){ 16 Serial.print(dato); 17 Serial.print(" es un Numero"); 18 }else{ 19 Serial.print(dato); 20 Serial.print(" NO NO NO"); 21 } 22 - 23 Serial.print(" ("); 24 Serial.print(d); 25 Serial.println(" "); 26 } 27 }</pre>
isAlpha(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter Alfabético (letra). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isAlpha_A")</p> <pre> 1 char dato; 2 int d; 3 - 4 void setup(){ 5 Serial.begin(9600); 6 while (!Serial) { 7 ; // Esperamos que el puerto serie este abierto. 8 } 9 Serial.println("Envíe un caracter por el Puerto Serie"); 10 - 11 void loop () { 12 if (Serial.available() > 0) { 13 dato = Serial.read(); //Se lee dato/carácter enviado. 14 d = dato; 15 - 16 Serial.print("Este caracter: "); 17 if(isAlpha(dato)){ 18 Serial.print(dato); 19 Serial.print(" es una Letra"); 20 }else{ 21 Serial.print(dato); 22 Serial.print(" NO NO NO"); 23 } 24 - 25 Serial.print(" ("); 26 Serial.print(d); 27 Serial.println(" "); 28 } 29 }</pre>
isAlphaNumeric(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter Alfanumérico (es una letra o un número). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isAlphaNumeric_A")</p> <pre> 1 char dato; 2 int d;</pre>

	<pre> - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 } - 10 void loop (){ 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; 14 15 Serial.print("Este carácter: "); 16 if(isAlphaNumeric(dato)){ 17 Serial.print(dato); 18 Serial.print(" es Alfanumérico"); 19 }else{ 20 Serial.print(dato); 21 Serial.print(" NO NO NO"); 22 } 23 24 Serial.print(" ("); 25 Serial.print(d); 26 Serial.println(" "); 27 } 28 }</pre>
isAscii(dato)	<p>Descripción: . Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter ASCII (es una letra o un número). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isAscii_A")</p> <pre> 1 char dato; 2 int d; - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un caracter por el Puerto Serie"); 9 } - 10 void loop (){ 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/caracter enviado. 13 d = dato; 14 15 Serial.print("Este caracter: "); 16 if(isAscii(dato)){ 17 Serial.print(dato); 18 Serial.print(" es isAscii"); 19 }else{ 20 Serial.print(dato); 21 Serial.print(" NO NO NO"); 22 } 23 24 Serial.print(" ("); 25 Serial.print(d); 26 Serial.println(" "); 27 } 28 }</pre>

	<pre> 24 Serial.println(" "); 25 } 26 }</pre>
isControl(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter de control. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa. Estos caracteres, nu pueden ser recibidos desde el puerto serie, sin embrago, se encuentran con relativa frecuencia cuando leemos archivos.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isControl_A")</p> <pre> 1 char dato; 2 int d; - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 } - 10 void loop (){ 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; 14 } 15 Serial.print("Este carácter: "); 16 if(isControl(dato)){ 17 Serial.print(dato); 18 Serial.print(" es un Carácter de Control"); 19 }else{ 20 Serial.print(dato); 21 Serial.print(" NO NO NO"); 22 } 23 Serial.print(" ("); 24 Serial.print(d); 25 Serial.println(" "); 26 }</pre>
isGraph(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter imprimible con algún contenido (el espacio se puede imprimir pero no tiene contenido). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isGraph_A")</p> <pre> 1 char dato; 2 int d; - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 }</pre>

	<pre> - 10 void loop () { 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; - 14 Serial.print("Este carácter: "); 15 if(isGraph(dato)){ 16 Serial.print(dato); 17 Serial.print(" es un Carácter Imprimible - no es espacio en blanco "); 18 }else{ 19 Serial.print(" NO Es un Carácter Imprimible"); 20 } - 21 Serial.print(" ("); 22 Serial.print(d); 23 Serial.println(")"); 24 } 25 }</pre>
isHexadecimalDigit(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un dígito hexadecimal válido (números con formato Hexadecimal - 0 - 9, a - f, o A - F). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isHexadecimalDigit_A")</p> <pre> 1 char dato; 2 int d; - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 } - 10 void loop () { 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; - 14 Serial.print("Este carácter: "); 15 if(isHexadecimalDigit(dato)){ 16 Serial.print(dato); 17 Serial.print(" es un dígito Hexadecimal válido"); 18 }else{ 19 Serial.print(dato); 20 Serial.print(" NO es un dígito Hexadecimal válido"); 21 } - 22 Serial.print(" ("); 23 Serial.print(d); 24 Serial.println(")"); 25 } 26 }</pre>
isLowerCase(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es una letra y está en Minúscula. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p>

	<p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isLowerCase_A")</p> <pre> 1 char dato; 2 int d; 3 4 void setup(){ 5 Serial.begin(9600); 6 while (!Serial) { 7 ; // Esperamos que el puerto serie este abierto. 8 } 9 Serial.println("Envíe un carácter por el Puerto Serie"); 10 11 void loop (){ 12 if (Serial.available() > 0) { 13 dato = Serial.read(); //Se lee dato/carácter enviado. 14 d = dato; 15 16 Serial.print("Este caracter: "); 17 if(isLowerCase(dato)){ 18 Serial.print(dato); 19 Serial.print(" es una Letra y esta con Minúscula"); 20 }else{ 21 Serial.print(dato); 22 Serial.print(" NO es una Letra o Esta con Mayúscula"); 23 } 24 25 Serial.print(" ("); 26 Serial.print(d); 27 Serial.println(")"); 28 } 29 }</pre>
<p>isPrintable(dato)</p>	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter es imprimible (es decir, cualquier carácter que produzca un resultado, incluso un espacio en blanco). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa. Estos caracteres, no pueden ser recibidos desde el puerto serie, sin embargo, se encuentran con relativa frecuencia cuando leemos archivos.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isPrintable_A")</p> <pre> 1 char dato; 2 int d; 3 4 void setup(){ 5 Serial.begin(9600); 6 while (!Serial) { 7 ; // Esperamos que el puerto serie este abierto. 8 } 9 Serial.println("Envíe un carácter por el Puerto Serie"); 10 11 void loop (){ 12 if (Serial.available() > 0) { 13 dato = Serial.read(); //Se lee dato/carácter enviado. 14 d = dato; 15 16 Serial.print("Este carácter: "); 17 if(isPrintable(dato)){ 18 Serial.print(dato); 19 } 20 }</pre>

	<pre> 17 Serial.print(" es un es un carácter es imprimible"); 18 }else{ 19 Serial.print(dato); 20 Serial.print(" NO es un es un carácter es imprimible "); 21 } 22 - 23 Serial.print(" ("); 24 Serial.print(d); 25 Serial.println(" "); 26 } 27 }</pre>
isPunct(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter o signo de puntuación (Punto, coma, etc). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isPunct_A")</p> <pre> 1 char dato; 2 int d; 3 - 4 void setup(){ 5 Serial.begin(9600); 6 while (!Serial) { 7 ; // Esperamos que el puerto serie este abierto. 8 } 9 Serial.println("Envíe un carácter por el Puerto Serie"); 10 } 11 - 12 void loop (){ 13 if (Serial.available() > 0) { 14 dato = Serial.read(); //Se lee dato/carácter enviado. 15 d = dato; 16 - 17 Serial.print("Este carácter: "); 18 if(isPunct(dato)){ 19 Serial.print(dato); 20 Serial.print(" es un Signo de Puntuación"); 21 }else{ 22 Serial.print(dato); 23 Serial.print(" NO NO NO"); 24 } 25 - 26 Serial.print(" ("); 27 Serial.print(d); 28 Serial.println(" "); 29 } 30 }</pre>
isSpace(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un carácter espaciador (espacio en blanco). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isSpace_A")</p> <pre> 1 char dato; 2 int d; 3 -</pre>

	<pre> 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 } - 10 void loop (){ 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; 14 Serial.print("Este carácter: "); 15 if(isSpace(dato)){ 16 Serial.print(dato); 17 Serial.print(" es un carácter espaciador (espacio en blanco)"); 18 }else{ 19 Serial.print(dato); 20 Serial.print(" NO NO NO"); 21 } 22 Serial.print(" ("); 23 Serial.print(d); 24 Serial.println(")"); 25 } 26 }</pre>
<p>isUpperCase(dato)</p>	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es una letra y está en Mayúscula. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isUpperCase_A")</p> <pre> 1 char dato; 2 int d; - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 } - 10 void loop (){ 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; 14 Serial.print("Este carácter: "); 15 if(isUpperCase(dato)){ 16 Serial.print(dato); 17 Serial.print(" es una letra en Mayúscula"); 18 }else{ 19 Serial.print(dato); 20 Serial.print(" NO NO NO"); 21 } 22 Serial.print(" ("); 23 Serial.print(d); 24 Serial.println(")"); 25 }</pre>

	26 }
isWhitespace(dato)	<p>Descripción: Específicamente evalúa si el dato que se le entrega como parámetro, es un espacio en blanco, avance de forma ('\f'), nueva línea ('\n'), retorno de carro ('\r'), pestaña horizontal ('\t') y pestaña vertical ('\v'), es decir caracteres que se encuentran leyendo un archivo. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser analizado, debe ser del tipo "char".</p> <p>Retorno: la función retorna "true" o "false" (Verdadero o falso), según sea el caso.</p> <p>(Programa "110_Texto_01_Reconoce_Caracter_isWhitespace_A")</p> <pre> 1 char dato; 2 int d; - 3 void setup(){ 4 Serial.begin(9600); 5 while (!Serial) { 6 ; // Esperamos que el puerto serie este abierto. 7 } 8 Serial.println("Envíe un carácter por el Puerto Serie"); 9 } - 10 void loop (){ 11 if (Serial.available() > 0) { 12 dato = Serial.read(); //Se lee dato/carácter enviado. 13 d = dato; 14 } 15 Serial.print("Este carácter: "); 16 if(isWhitespace(dato)){ 17 Serial.print(dato); 18 Serial.print(" es un espacio en Blanco"); 19 }else{ 20 Serial.print(dato); 21 Serial.print(" NO NO NO"); 22 } 23 Serial.print(" ("); 24 Serial.print(d); 25 Serial.println(")"); 26 }</pre>

toupper(dato)	<p>Descripción: Esta función transforma el carácter recibido como parametro (si es letra) a Mayúscula. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser transformado, debe ser del tipo "char".</p> <p>Retorno: El Caracter Transformado. Ver Ejemplo.</p>
tolower(dato)	<p>Descripción: Esta función transforma el carácter recibido como parametro (si es letra) a Minuscula. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: "dato" es el carácter a ser transformado, debe ser del tipo "char".</p> <p>Retorno: El Caracter Transformado. Ver Ejemplo.</p>

A tener en cuenta:



- El carácter '5' tiene el valor entero 53 - Si restamos los caracteres '5'-'0' o lo que es igual, sus valores enteros 53 - 48, entonces el resultado será el Numero 5
- Con las letras mayúsculas y minúsculas pasa algo similar que con los números. Si al valor de cada letra Mayúscula le sumamos 32 obtendremos la misma letra pero minúscula. $c = 'B' + 32 ==> 'b'$

110_Texto_02_Tratamiento_de_Cadenas_toupper_tolower_A	110_Texto_02_Tratamiento_de_Cadenas_toupper_tolower_B
1 char Car_Org = 'a',	1 char Car_Org = 'a',
2 Car_01,	2 Car_01,
3 Car_02;	3 Car_02;
-	-
4 void setup(){	4 void setup(){
5 Serial.begin(9600);	5 Serial.begin(9600);
6 while (!Serial) {	6 while (!Serial) {
7 ; // Espera que puerto serie abra.	7 ; // Espera que puerto serie abra.
8 }	8 }
9 Car_01 = toupper(Car_Org);	9 Car_01 = Car_Org - 32;
10 Serial.print("Carácter Convertido: ");	10 Serial.print("Caracter Convertido: ");
11 Serial.println(Car_01);	11 Serial.println(Car_01);
12 Car_02 = tolower(Car_01);	12 Car_02 = Car_01 + 32;
13 Serial.print("Nuevamente Convertido: ");	13 Serial.print("Nuevamente Convertido: ");
14 Serial.println(Car_02);	14 Serial.println(Car_02);
15 }	15 }
16 void loop() {	16 void loop() {
17 // Esta vez hacemos nada.	17 // Esta vez hacemos nada
18 }	18 }

CADENA DE CARACTERES (string)

Un **"string"** es una cadena de caracteres, o lo que es lo mismo, un string es un array (vector) de chars. Cuando se trabaja con grandes cantidades de texto, es conveniente usar un strings.

Una características de los strings, es el carácter de terminación, que nos indica dónde acaba la cadena de caracteres (texto contenido). Podemos definir un string de tamaño 500 para almacenar una cadena de caracteres, pero algunas veces puede tener una sola palabra y otras, una frase completa. Este carácter de terminación, permite a funciones como **"Serial.print()"** saber hasta dónde debe procesar el string y mostrar por pantalla solamente el texto que debe mostrar.

El carácter de terminación o **"null termination"** se representa como **'\0'** y corresponde al código 0 (cero) de ASCII. Esto significa que, un string para almacenar la palabra "Hola" debe tener un tamaño de 5 y no de 4, este espacio extra es necesario para guardar el **"null termination"** o carácter de terminación.

Cadena

H	o	l	a	\0					
0	1	2	3	4	5	6	7	8	9

Algo a tener en cuenta, es que las posiciones, comienzan desde la 0 (cero), tal como se muestra en la imagen.



Los strings siempre se definen entre dobles comillas “hola”, mientras que los caracteres siempre se definen con comillas simples ‘h’.

Los Strings usan mucho espacio en memoria, pero son muy útiles y los vamos a utilizar mucho en la parte de comunicación, por este motivo es importante aprender a manejarlos.

Tener en cuenta que los string o cadenas de caracteres, podremos usarlos de varias formas. Comenzando por lo más rudimentario, en donde nosotros hacemos todo, o ayudados por las clases y métodos que Arduino nos brinda, en donde un String no es un tipo de dato propiamente dicho, sino una clase preprogramada, por lo tanto, contendrá funciones asociadas (métodos), operadores y propiedades.

CADENA SIMPLES

EL lenguaje de programación que usamos, es muy flexible, y nos permite declarar estas cadenas en formas variadas, según nos convenga.

<code>char cadena_01[200];</code>	Declaración de un string (arreglo o vector de caracteres) sin inicializar, en el que podremos guardar hasta 199 caracteres (debe quedar uno libre para el carácter nulo requerido), luego haremos lo que haga falta. En el ejemplo usamos cadena_01 , y el tamaño o longitud del string, puede ser hasta 255, aunque hay formas de ampliarlos.
<code>char cadena_02[9] = {'R', 'o', 'b', 'o', 't', 'i', 'c', 'a'};</code>	Declaración de un string (arreglo o vector de caracteres), en el que guardamos 8 caracteres. Luego el compilador agregará el carácter nulo requerido. En el ejemplo usamos cadena_02 .
<code>char cadena_03[9] = {'R', 'o', 'b', 'o', 't', 'i', 'c', 'a', '\0'};</code>	Declaración de un string (arreglo o vector de caracteres), en el que guardamos 8 caracteres. Agregue explícitamente el carácter nulo (el compilador ya no necesita agregarlo). En el ejemplo usamos cadena_03 .
<code>char cadena_04[] = "Robotica";</code>	Declaración de un string constante con una cadena entre comillas; el compilador dimensionará el arreglo y lo ajusta al tamaño de la constante (cadena) y adicionará un carácter nulo de terminación. En el ejemplo usamos cadena_04 .
<code>char cadena_05[9] = "Robotica";</code>	Declaración de un string con un tamaño explícito. En el ejemplo usamos cadena_05 .
<code>char cadena_06[15] = "Robotica";</code>	Declaración de un string , dejando espacio adicional para una cadena más grande. En el ejemplo usamos cadena_06 .

1) Cadenas de Caracteres. Ej_001

Habiendo analizado diversas formas de declarar un string, veamos las primeras acciones, concatenar un texto, carácter a carácter.

(Programa “110_Texto_02_Tratamiento_de_Cadenas_A”)



1	<code>char Cadena_01[200] = "Estudio Robotica";</code>
-	
2	<code>void setup(){</code>
3	<code> Serial.begin(9600);</code>
4	<code> while (!Serial) {</code>
5	<code> ; // Esperamos que el puerto serie este abierto.</code>
6	<code> }</code>
7	<code> Serial.println(Cadena_01);</code>
8	<code> Cadena_01[15] = ' ';</code>
9	<code> Cadena_01[16] = 'y';</code>
10	<code> Cadena_01[17] = ' ';</code>
11	<code> Cadena_01[18] = 'm';</code>
12	<code> Cadena_01[19] = 'e';</code>
13	<code> Cadena_01[20] = ' ';</code>

```
14 Cadena_01[21] = 'g';
15 Cadena_01[22] = 'u';
16 Cadena_01[23] = 's';
17 Cadena_01[24] = 't';
18 Cadena_01[25] = 'a';
19 Cadena_01[26] = 0; // termino el string
20 Serial.println(Cadena_01);
21 }
-
22 void loop(){
23     // Por esta vez no hay que hacer nada.!
24 }
```

2) Cadenas de Caracteres. Ej_002

Nuevamente vamos a concatenar un texto, armando una frase, pero esta vez, los caracteres serán leídos desde el puerto serie. El texto debe leerse carácter a carácter, y será ensamblado en un string, o arreglo de caracteres. **Cada frase se da por terminada con un punto.**



(Programa "110_Texto_02_Tratamiento_de_Cadenas_B")

```
1 char Cadena_01[200];
2 char Caracter;
3 int Posicion;
-
4 void setup(){
5     Serial.begin(9600);
6     while (!Serial) {
7         ; // Esperamos que el puerto serie este abierto.
8     }
9     Posicion = 0;
10    Serial.println("Envie un Mensaje y termine con un Punto");
11 }
-
12 void loop(){
13     while (Serial.available()){
14         Caracter = Serial.read();
15         if(Caracter != '.'){
16             Cadena_01[Posicion] = Caracter;
17             Posicion++;
18         }else{
19             Serial.println(Cadena_01);
20             Posicion = 0;
21         }
22     }
23 }
```

"sizeof()" - "strlen()"

Analizaremos dos funciones que nos permiten analizar las cadenas o arreglos de caracteres. Una nos permite saber el tamaño de la Variable "**sizeof()**" y la otra cuando ocupa el contenido de la variable, dentro de la variable "**strlen()**"

sizeof(Cadena)	Descripción: Esta función retorna el número de bytes que realmente ocupa una variable. Para el caso de una cadena, retorna el tamaño máximo de

	<p>posiciones disponibles. Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: Recibe una variable de cualquier tipo valido.</p> <p>Retorno: la función retorna un numero entero.</p>
strlen(Cadena)	<p>Descripción: La función "strlen()" se usa para obtener la longitud de la cadena. La longitud de la cadena es solo para los caracteres imprimibles y no incluye el terminador nulo.</p> <p>Parámetros: Recibe una cadena de caracteres.</p> <p>Retorno: la función retorna un numero entero.</p>

(Programa "110_Texto_02_Tratamiento_de_Cadenas_sizeof_strlen_A")

1	char Cadena_01[200];
2	char Cadena_02[9] = {'R', 'o', 'b', 'o', 't', 'i', 'c', 'a'};
3	char Cadena_03[9] = {'R', 'o', 'b', 'o', 't', 'i', 'c', 'a', '\0'};
4	char Cadena_04[] = "Estudio Robotica";
5	char Cadena_05[9] = "Robotica";
6	char Cadena_06[15] = "Robotica";
-	
7	int i,
8	Largo,
9	Cantidad;
-	
10	void setup(){
11	Serial.begin(9600);
12	while (!Serial) {
13	; // Esperamos que el puerto serie este abierto.
14	}
15	}
-	
16	void loop() {
17	Cantidad = sizeof(Cadena_01);
18	Largo = strlen(Cadena_01);
19	Serial.print("El string Cadena_01 ocupa : ");
20	Serial.print(Cantidad);
21	Serial.print(" Bytes y contiene: ");
-	Serial.print(Largo);
22	Serial.println(" Caracteres");
-	
23	delay(1000); //Hace mas lento el programa
-	
24	Cantidad = sizeof(Cadena_02);
25	Largo = strlen(Cadena_02);
26	Serial.print("El string Cadena_02 ocupa : ");
27	Serial.print(Cantidad);
28	Serial.print(" Bytes y contiene: ");
29	Serial.print(Largo);
30	Serial.println(" Caracteres");
-	
31	delay(1000); //Hace mas lento el programa
-	
32	Cantidad = sizeof(Cadena_03);
33	Largo = strlen(Cadena_03);
34	Serial.print("El string Cadena_03 ocupa : ");
35	Serial.print(Cantidad);
36	Serial.print(" Bytes y contiene: ");
37	Serial.print(Largo);

38	Serial.println(" Caracteres");
-	
39	delay(1000); //Hace más lento el programa
-	
40	Cantidad = sizeof(Cadena_04);
41	Largo = strlen(Cadena_04);
42	Serial.print("El string Cadena_04 ocupa : ");
43	Serial.print(Cantidad);
44	Serial.print(" Bytes y contiene: ");
45	Serial.print(Largo);
46	Serial.println(" Caracteres");
-	
47	for (i = 0; i < Cantidad - 1; i++){
48	Serial.print(i, DEC);
49	Serial.print(" = ");
50	Serial.println(Cadena_04[i]);
51	}
-	
52	delay(1000); //Hace mas lento el programa
-	
53	Cantidad = sizeof(Cadena_05);
54	Largo = strlen(Cadena_05);
55	Serial.print("El string Cadena_05 ocupa : ");
56	Serial.print(Cantidad);
57	Serial.print(" Bytes y contiene: ");
58	Serial.print(Largo);
59	Serial.println(" Caracteres");
-	
60	delay(1000); //Hace mas lento el programa
-	
61	Cantidad = sizeof(Cadena_06);
62	Largo = strlen(Cadena_06);
63	Serial.print("El string Cadena_06 ocupa : ");
64	Serial.print(Cantidad);
65	Serial.print(" Bytes y contiene: ");
66	Serial.print(Largo);
67	Serial.println(" Caracteres");
-	
68	Serial.println("-----");
69	delay(5000); //Hace mas lento el programa
70	}

"strcpy()" - "strcat()"

Dos funciones más, una copia el contenido de una cadena en otra "**strcpy()**", y la segunda, agrega al contenido de la primera, el contenido de la segunda "**strcat()**".

strcpy(Cadena_01, Cadena_02)	<p>Descripción: Esta función "strcpy()" copia el contenido de una cadena (Cadena_02) dentro de otra, borrando del contenido previo de la cadena receptora (Cadena_01). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: Recibe dos string o arreglo de caracteres, las que llamaremos Cadena_01 (receptora) y Cadena_02 (emisora).</p> <p>Retorno: Nada.</p>
-------------------------------------	--

strlen(Cadena_01, Cadena_02)	<p>Descripción: Esta función " strlen() Concatena o suma el contenido de una cadena (Cadena_02) dentro de otra (Cadena_01). Esta función es provista por el IDE y no es necesario incluir alguna biblioteca, simplemente se usa.</p> <p>Parámetros: Recibe dos string o arreglo de caracteres, las que llamaremos Cadena_01 (receptora) y Cadena_02 (emisora).</p> <p>Retorno: Nada.</p>
-------------------------------------	---

(Programa "110_Texto_02_Tratamiento_de_Cadenas_strcpy_strcat_A")

1	char Cadena_01[200];
2	char Cadena_02[] = "Estudio Robotica";
3	char Cadena_03[15] = "me gusta";
-	
4	int Largo,
5	Cantidad;
-	
6	void setup(){
7	Serial.begin(9600);
8	while (!Serial) {
9	; // Esperamos que el puerto serie este abierto.
10	}
11	}
-	
12	void loop() {
13	Cantidad = sizeof(Cadena_01);
14	Largo = strlen(Cadena_01);
15	Serial.print("El string Cadena_01 ocupa: ");
16	Serial.print(Cantidad);
17	Serial.print(" Bytes y contiene: ");
18	Serial.print(Largo);
19	Serial.println(" Caracteres");
-	
20	delay(1000); //Hace mas lento el programa
-	
21	// Copia contenido de Cadena_02 en Cadena_01
22	strcpy(Cadena_01, Cadena_02);
23	Serial.println(Cadena_01);
-	
24	// Adiciona o concatena la cadena entre comillas dentro de Cadena_01
25	strcat(Cadena_01, " y ");
26	Serial.println(Cadena_01);
-	
27	// Adiciona o concatena el contenido de Cadena_03 dentro de Cadena_01
28	strcat(Cadena_01, Cadena_03);
29	Serial.println(Cadena_01);
-	
30	Cantidad = sizeof(Cadena_01);
31	Largo = strlen(Cadena_01);
32	Serial.print("Aora Cadena_01 ocupa: ");
33	Serial.print(Cantidad);
34	Serial.print(" Bytes y contiene: ");
35	Serial.print(Largo);
36	Serial.println(" Caracteres");
-	
37	Serial.println("-----");
38	delay(5000); //Hace mas lento el programa

"strcmp()

strcmp(Cadena_01, Cadena_02)

Descripción: Recordando que cada carácter tiene un código (Ver Apéndice A - Tabla de Códigos ASCII), entonces una cadena de caracteres (Una palabra por ejemplo), tendría un valor numérico, si reemplazamos cada letra por su código (su código ASCII). Entonces el Carácter "A" < "B" ya que el código de "A" es 65 y el código de la "B" es 66. y debido a que en la Tabla ASCII las letras mayúsculas están antes que las minúsculas, entonces "A" < "a". Si se ha comprendido bien esto, entonces podremos afirmar que con esta función podremos ordenar alfabéticamente cadenas de texto o strings.

Parámetros: Recibe dos string o arreglo de caracteres, las que llamaremos **Cadena_01** y **Cadena_02**.

Retorno: Número Entero.

(Programa "110_Texto_02_Tratamiento_de_Cadenas_strcmp_A")

```
char Cadena_01[ ] = "Primero";
char Cadena_02[15] = "Segundo";
int res;
void setup(){
  Serial.begin(9600);
  while (!Serial) {
    ; // Esperamos que el puerto serie este abierto.
  }
  res = strcmp( Cadena_01, Cadena_02 );
  if (res == 0){
    Serial.println("Las cadenas son Idénticas");
  } else if (res < 0){
    Serial.println("Cadena_01 es la menor");
  } else {
    Serial.println("Cadena_02 es la menor");
  }
}
void loop() {
}
```

Esta función Compara dos Cadenas (la comparación es de acuerdo a los códigos ASCII que componen las cadenas), y retorna un número entero que puede tomar tres estados:

- Si el número retornado es 0 (cero) Las cadenas comparadas son iguales (idénticas).
- Si el número retornado es Negativo, la primera cadena es Menor que la segunda cadena.
- Si el número retornado es Positivo, la primera cadena es Mayor que la segunda cadena

Es muy importante destacar, que la comparación realizada, es sobre los contenidos de cada variable, y no interesa el largo con el que se haya declarado la cadena, solo importa el contenido.

3) Transformar Toda una Cadena de Mayúscula a Minúscula y Viceversa.

Para realizar esta transformación, usaremos funciones que ya habíamos usado antes, "**toupper()**" que transforma un carácter (que está en Minúscula) a Mayúscula y luego la otra función "**tolower()**", que transforma un carácter (que está en Minúsculas) a Mayúscula.



(Programa "110_Texto_02_Tratamiento_de_Cadenas_toupper_tolower_C")

```
1 char Cadena_Org[ ] = "estudio robótica y me GUSTA mucho";
2 char Cadena_01[100];
3 char Cadena_02[100];
4
5 int i,
6   Largo;
7
8 void setup(){
```



```
7 Serial.begin(9600);
8 while (!Serial) {
9   ; // Espera que puerto serie abra.
10 }
11 Serial.print("Cadena original: ");
12 Serial.println(Cadena_Org);
13
14 Largo = strlen(Cadena_Org);
15
16 for(i=0; i<Largo; i++){
17   Cadena_01[i] = toupper(Cadena_Org[i]);
18 }
19 Serial.print("Cadena Convertida: ");
20 Serial.println(Cadena_01);
21
22 for(i=0; i<Largo; i++){
23   Cadena_02[i] = tolower(Cadena_01[i]);
24 }
25 Serial.print("Nuevamente Convertida: ");
26 Serial.println(Cadena_02);
27
28 void loop() {
29   // Esta vez hacemos nada
30 }
```

CADENAS MÚLTIPLES O VECTOR DE CADENAS

Con frecuencia es conveniente, cuando se trabaja con grandes cantidades de texto, como un proyecto con una pantalla LCD, configurar un arreglo o vector de cadenas. Debido a que las cadenas en sí mismas son arreglos o vectores, un vector de vectores es en realidad una matriz bidimensional.

En el siguiente código, el asterisco después del tipo de datos **char** "**char ***" indica que se trata de un vector o arreglo de "**punteros**".

Cada puntero apunta a una cadena, que en el ejemplo son de igual largo, pero podrían ser todas con tamaños distintos.

Cada una de las cadenas, son en realidad un vector, por lo tanto, al crear un vector de vectores, estamos usando una matriz (Bi-Dimensional).

Los punteros son una de las partes más misteriosas y esotéricas del C.

Aunque en realidad, no es necesario comprender como funcionan los punteros en detalle, para que todos podamos usarlos de manera fácil y efectiva, al menos, por ahora.

(Programa "110_String_03_Cadenas_Multiples_A")

```
char* miStrings[]={ "01 - String 01",
                    "02 - String 02",
                    "03 - String 03",
                    "04 - String 04",
                    "05 - String 05",
                    "06 - String 06" };

int i;
void setup(){
  Serial.begin(9600);
  while (!Serial) {
    ; // Esperamos que el puerto serie este abierto.
  }
}
void loop(){
  for (i = 0; i < 6; i++){
    Serial.println(miStrings[i]);
    delay(500);
  }
}
```

Continuará - (esta en edición)

Apéndice A - Tabla de Códigos ASCII

Breve historia del Código ASCII: El código ASCII (siglas en ingles para American Standard Code for Information Interchange, es decir Código Americano (Je! lease estadounidense...) Estándar para el intercambio de Información) (se pronuncia Aski).

Fue creado en 1963 por el Comité Estadounidense de Estándares o "ASA", este organismo cambio su nombre en 1969 por "Instituto Estadounidense de Estándares Nacionales" o "ANSI" como se lo conoce desde entonces.

Este código nació a partir de reordenar y expandir el conjunto de símbolos y caracteres ya utilizados en aquel momento en telegrafía por la compañía Bell. En un primer momento solo incluía letras mayúsculas y números, pero en 1967 se agregaron las letras minúsculas y algunos caracteres de control, formando así lo que se conoce como US-ASCII, es decir los caracteres del 0 al 127.

Así con este conjunto de solo 128 caracteres fue publicado en 1967 como estándar, conteniendo todos lo necesario para escribir en idioma ingles.

En 1981, la empresa IBM desarrolló una extensión de 8 bits del código ASCII, llamada "pagina de código 437", en esta versión se reemplazaron algunos caracteres de control obsoletos, por caracteres gráficos. Además se incorporaron 128 caracteres nuevos, con símbolos, signos, gráficos adicionales y letras latinas, necesarias para la escrituras de textos en otros idiomas, como por ejemplo el español. Así fue como se sumaron los caracteres que van del ASCII 128 al 255.

IBM incluyó soporte a esta página de código en el hardware de su modelo 5150, conocido como "IBM-PC", considerada la primera computadora personal. El sistema operativo de este modelo, el "MS-DOS" también utilizaba el código ASCII extendido.

Casi todos los sistemas informáticos de la actualidad utilizan el código ASCII para representar caracteres, símbolos, signos y textos (74) .

A tener en cuenta:



- El carácter '5' tiene el valor entero 53 - Si restamos los caracteres '5'-'0' o lo que es igual, sus valores enteros 53 - 48, entonces el resultado será el Numero 5
- con las letras mayúsculas y minúsculas pasa algo similar que con los números. Si al valor de cada letra Mayúscula le sumamos 32 obtendremos la misma letra pero minúscula. $c = 'B' + 32 ==> 'b'$

Como utilizar el código ASCII:

Sin saberlo lo utilizas todo el tiempo, cada vez que utilizas algún sistema informático; pero si lo que necesitas es obtener algunos de los caracteres no incluidos en tu teclado debes hacer lo siguiente, por ejemplo:

Como escribir con el teclado, o tipear : Letra EÑE mayúscula - letra N con tilde - ENIE

WINDOWS: en computadoras con sistema operativo como Windows 8, Win 7, Vista, Windows Xp, etc. Para obtener la letra, carácter, signo o símbolo "Ñ" : (Letra EÑE mayúscula - letra N con tilde - ENIE) en ordenadores con sistema operativo Windows:

1) Presiona la tecla "Alt" en tu teclado, y no la sueltes.

- 2) Sin dejar de presionar "Alt", presiona en el teclado numérico el número "165", que es el número de la letra o símbolo "Ñ" en el código ASCII.
- 3) Luego deja de presionar la tecla "Alt" y... ¡ Ya está listo !

Lista completa de caracteres, letras, signos y símbolos del código ASCII

Caracteres de control ASCII no imprimibles :

código ascii 00 = NULL (Carácter nulo)
código ascii 01 = SOH (Inicio de encabezado)
código ascii 02 = STX (Inicio de texto)
código ascii 03 = ETX (Fin de texto, palo corazón barajas inglesas de póker)
código ascii 04 = EOT (Fin de transmisión, palo diamantes barajas de póker)
código ascii 05 = ENQ (Consulta, palo tréboles barajas inglesas de póker)
código ascii 06 = ACK (Reconocimiento, palo picas cartas de póker)
código ascii 07 = BEL (Timbre)
código ascii 08 = BS (Retroceso)
código ascii 09 = HT (Tabulador horizontal)
código ascii 10 = LF (Nueva línea - salto de línea)
código ascii 11 = VT (Tabulador vertical)
código ascii 12 = FF (Nueva página - salto de página)
código ascii 13 = CR (ENTER - retorno de carro)
código ascii 14 = SO (Desplazamiento hacia afuera)
código ascii 15 = SI (Desplazamiento hacia adentro)
código ascii 16 = DLE (Escape de vínculo de datos)
código ascii 17 = DC1 (Control dispositivo 1)
código ascii 18 = DC2 (Control dispositivo 2)
código ascii 19 = DC3 (Control dispositivo 3)
código ascii 20 = DC4 (Control dispositivo 4)
código ascii 21 = NAK (Confirmación negativa)
código ascii 22 = SYN (Inactividad sincrónica)
código ascii 23 = ETB (Fin del bloque de transmisión)
código ascii 24 = CAN (Cancelar)
código ascii 25 = EM (Fin del medio)
código ascii 26 = SUB (Sustitución)
código ascii 27 = ESC (Esc - escape)
código ascii 28 = FS (Separador de archivos)
código ascii 29 = GS (Separador de grupos)
código ascii 30 = RS (Separador de registros)
código ascii 31 = US (Separador de unidades)
código ascii 127 = DEL (DEL - Suprimir, borrar, eliminar)

Caracteres ASCII alfanumericos imprimibles:






















código ascii 32 = espacio (Espacio en blanco)
código ascii 33 = ! (Signos de exclamacion, signo de admiracion)
código ascii 34 = " (Comillas dobles , comillas altas o inglesas)
código ascii 35 = # (Signo numeral o almohadilla)
código ascii 36 = \$ (Signo pesos)
código ascii 37 = % (Signo de porcentaje - por ciento)
código ascii 38 = & (Y - ampersand - et latina)
código ascii 39 = ' (Comillas simples, apóstrofe)
código ascii 40 = ((Abre paréntesis)
código ascii 41 =) (Cierra paréntesis)
código ascii 42 = * (Asterisco)
código ascii 43 = + (Signo mas, suma, positivo)
código ascii 44 = , (Coma)

codigo ascii 45 = - (Signo menos , resta , negativo , guión medio)
codigo ascii 46 = . (Punto)
codigo ascii 47 = / (Barra inclinada, división, operador cociente)
codigo ascii 48 = 0 (Número cero)
codigo ascii 49 = 1 (Número uno)
codigo ascii 50 = 2 (Número dos)
codigo ascii 51 = 3 (Número tres)
codigo ascii 52 = 4 (Número cuatro)
codigo ascii 53 = 5 (Número cinco)
codigo ascii 54 = 6 (Número seis)
codigo ascii 55 = 7 (Número siete)
codigo ascii 56 = 8 (Número ocho)
codigo ascii 57 = 9 (Número nueve)
codigo ascii 58 = : (Dos puntos)
codigo ascii 59 = ; (Punto y coma)
codigo ascii 60 = < (Menor que)
codigo ascii 61 = = (Signo igual, igualdad, igual que)
codigo ascii 62 = > (Mayor que)
codigo ascii 63 = ? (Cierra signo interrogación)
codigo ascii 64 = @ (Arroba)
codigo ascii 65 = A (Letra A mayúscula)
codigo ascii 66 = B (Letra B mayúscula)
codigo ascii 67 = C (Letra C mayúscula)
codigo ascii 68 = D (Letra D mayúscula)
codigo ascii 69 = E (Letra E mayúscula)
codigo ascii 70 = F (Letra F mayúscula)
codigo ascii 71 = G (Letra G mayúscula)
codigo ascii 72 = H (Letra H mayúscula)
codigo ascii 73 = I (Letra I mayúscula)
codigo ascii 74 = J (Letra J mayúscula)
codigo ascii 75 = K (Letra K mayúscula)
codigo ascii 76 = L (Letra L mayúscula)
codigo ascii 77 = M (Letra M mayúscula)
codigo ascii 78 = N (Letra N mayúscula)
codigo ascii 79 = O (Letra O mayúscula)
codigo ascii 80 = P (Letra P mayúscula)
codigo ascii 81 = Q (Letra Q mayúscula)
codigo ascii 82 = R (Letra R mayúscula)
codigo ascii 83 = S (Letra S mayúscula)
codigo ascii 84 = T (Letra T mayúscula)
codigo ascii 85 = U (Letra U mayúscula)
codigo ascii 86 = V (Letra V mayúscula)
codigo ascii 87 = W (Letra W mayúscula)
codigo ascii 88 = X (Letra X mayúscula)
codigo ascii 89 = Y (Letra Y mayúscula)
codigo ascii 90 = Z (Letra Z mayúscula)
codigo ascii 91 = [(Abre corchetes)
codigo ascii 92 = \ (Barra invertida , contrabarra , barra inversa)
codigo ascii 93 =] (Cierra corchetes)
codigo ascii 94 = ^ (Intercalación - acento circunflejo)
codigo ascii 95 = _ (Guión bajo , subrayado , subguión)
codigo ascii 96 = ` (Acento grave)
codigo ascii 97 = a (Letra a minúscula)
codigo ascii 98 = b (Letra b minúscula)
codigo ascii 99 = c (Letra c minúscula)

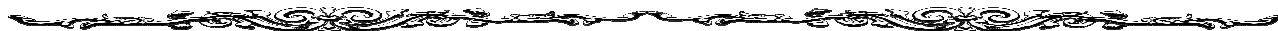
codigo ascii 100 = d (Letra d minúscula)
codigo ascii 101 = e (Letra e minúscula)
codigo ascii 102 = f (Letra f minúscula)
codigo ascii 103 = g (Letra g minúscula)
codigo ascii 104 = h (Letra h minúscula)
codigo ascii 105 = i (Letra i minúscula)
codigo ascii 106 = j (Letra j minúscula)
codigo ascii 107 = k (Letra k minúscula)
codigo ascii 108 = l (Letra l minúscula)
codigo ascii 109 = m (Letra m minúscula)
codigo ascii 110 = n (Letra n minúscula)
codigo ascii 111 = o (Letra o minúscula)
codigo ascii 112 = p (Letra p minúscula)
codigo ascii 113 = q (Letra q minúscula)
codigo ascii 114 = r (Letra r minúscula)
codigo ascii 115 = s (Letra s minúscula)
codigo ascii 116 = t (Letra t minúscula)
codigo ascii 117 = u (Letra u minúscula)
codigo ascii 118 = v (Letra v minúscula)
codigo ascii 119 = w (Letra w minúscula)
codigo ascii 120 = x (Letra x minúscula)
codigo ascii 121 = y (Letra y minúscula)
codigo ascii 122 = z (Letra z minúscula)
codigo ascii 123 = { (Abre llave curva - llaves curvas)
codigo ascii 124 = | (Barra vertical, pleca , linea vertical)
codigo ascii 125 = } (Cierra llave - llaves curvas)
codigo ascii 126 = ~ (Signo de equivalencia , tilde o virgullilla de la ñ)

Caracteres ASCII extendidos imprimibles :

codigo ascii 128 = Ç (Letra C cedilla mayúscula)
codigo ascii 129 = ü (Letra u minúscula con diéresis)
codigo ascii 130 = é (Letra e minúscula con acento agudo)
codigo ascii 131 = â (Letra a minúscula con acento circunflejo)
codigo ascii 132 = ä (Letra a minúscula con diéresis)
codigo ascii 133 = à (Letra a minúscula con acento grave)
codigo ascii 134 = å (Letra a minúscula con anillo)
codigo ascii 135 = ç (Letra c cedilla minúscula)
codigo ascii 136 = ê (Letra e minúscula con acento circunflejo)
codigo ascii 137 = ë (Letra e minúscula con diéresis)
codigo ascii 138 = è (Letra e minúscula con acento grave)
codigo ascii 139 = ï (Letra i minúscula con diéresis)
codigo ascii 140 = î (Letra i minúscula con acento circunflejo)
codigo ascii 141 = ì (Letra i minúscula con acento grave)
codigo ascii 142 = Ä (Letra A mayúscula con diéresis)
codigo ascii 143 = Å (Letra A mayúscula con anillo)
codigo ascii 144 = É (Letra E mayúscula con acento agudo)
codigo ascii 145 = æ (Diptongo latino ae minúscula)
codigo ascii 146 = Æ (Diptongo latino AE mayúscula)
codigo ascii 147 = ô (Letra o minúscula con acento circunflejo)
codigo ascii 148 = ö (Letra o minúscula con diéresis)
codigo ascii 149 = ò (Letra o minúscula con acento grave)
codigo ascii 150 = û (Letra u minúscula con acento circunflejo)
codigo ascii 151 = ù (Letra u minúscula con acento grave)
codigo ascii 152 = ÿ (Letra y minúscula con diéresis)
codigo ascii 153 = Ö (Letra O mayúscula con diéresis)

codigo ascii 154 = Û (Letra U mayúscula con diéresis)
codigo ascii 155 = ø (Letra o minúscula con barra inclinada)
codigo ascii 156 = £ (Signo Libra Esterlina)
codigo ascii 157 = Ø (Letra O mayúscula con barra inclinada)
codigo ascii 158 = × (Signo de multiplicación)
codigo ascii 159 = f (Símbolo de función, florín neerlandés)
codigo ascii 160 = á (Letra a minúscula con acento agudo)
codigo ascii 161 = í (Letra i minúscula con acento agudo)
codigo ascii 162 = ó (Letra o minúscula con acento agudo)
codigo ascii 163 = ú (Letra u minúscula con acento agudo)
codigo ascii 164 = ñ (Letra eñe minúscula - letra n con tilde - enie)
codigo ascii 165 = Ñ (Letra EÑE mayúscula - letra N con tilde - ENIE)
codigo ascii 166 = ^a (Ordinal femenino, indicador de genero femenino)
codigo ascii 167 = ^o (Ordinal masculino, indicador de genero masculino)
codigo ascii 168 = ¿ (Abre signo interrogacion)
codigo ascii 169 = ® (Símbolo de Marca Registrada)
codigo ascii 170 = ¬ (Signo de negacion)
codigo ascii 171 = ½ (Un medio, mitad, fracción)
codigo ascii 172 = ¼ (Un cuarto, cuarta parte, fracción)
codigo ascii 173 = ¡ (Abre signos de exclamacion, signo de admiracion)
codigo ascii 174 = « (Abre comillas bajas, angulares, latinas o españolas)
codigo ascii 175 = » (Cierra comillas bajas, angulares, latinas o españolas)
codigo ascii 176 =  (Bloque color tramado densidad baja, carácter gráfico)
codigo ascii 177 =  (Bloque color tramado densidad media, gráfico)
codigo ascii 178 =  (Bloque color tramado densidad alta, carácter gráfico)
codigo ascii 179 =  (Línea simple vertical de recuadro gráfico)
codigo ascii 180 =  (Línea vertical con empalme de recuadro gráfico)
codigo ascii 181 = Á (Letra a mayúscula con acento agudo)
codigo ascii 182 = Â (Letra A mayúscula con acento circunflejo)
codigo ascii 183 = Ã (Letra A mayúscula con acento grave)
codigo ascii 184 = © (Símbolo Copyright, bajo derecho de autor)
codigo ascii 185 =  (Doble línea vertical empalme izquierdo, gráfico)
codigo ascii 186 =  (Líneas doble vertical de recuadro gráfico, verticales)
codigo ascii 187 =  (Línea doble esquina superior derecha de recuadro)
codigo ascii 188 =  (Línea doble esquina inferior derecha de recuadro)
codigo ascii 189 = ¢ (Signo centavo, céntimo o centésimo)
codigo ascii 190 = ¥ (Signo monetario YEN japonés, YUAN chino)
codigo ascii 191 =  (Línea simple esquina de recuadro gráfico)
codigo ascii 192 =  (Línea simple esquina de recuadro gráfico)
codigo ascii 193 =  (Línea horizontal con empalme de recuadro gráfico)
codigo ascii 194 =  (Línea horizontal con empalme de recuadro gráfico)
codigo ascii 195 =  (Línea vertical con empalme de recuadro gráfico)
codigo ascii 196 = - (Línea simple horizontal de recuadro gráfico)
codigo ascii 197 =  (Líneas simples empalmes de recuadro gráfico)
codigo ascii 198 = ã (Letra a minúscula con tilde)
codigo ascii 199 = Ã (Letra A mayúscula con tilde)
codigo ascii 200 =  (Línea doble esquina inferior izquierda de recuadro)
codigo ascii 201 =  (Línea doble esquina superior izquierda de recuadro)
codigo ascii 202 =  (Doble línea horizontal empalme arriba, recuadro)
codigo ascii 203 =  (Doble línea horizontal empalme abajo, recuadro)
codigo ascii 204 =  (Doble línea vertical empalme derecho, recuadro)
codigo ascii 205 = == (Líneas doble horizontales de recuadro gráfico)
codigo ascii 206 =  (Líneas dobles cruce de líneas de recuadro gráfico)
codigo ascii 207 = ₡ (Signo monetario - divisa general)
codigo ascii 208 = ð (Letra eth latina minúscula)

codigo ascii 209 = Ð (Letra eth latina mayúscula)
codigo ascii 210 = Ê (Letra E mayúscula con acento circunflejo)
codigo ascii 211 = Ë (Letra E mayúscula con diéresis)
codigo ascii 212 = È (Letra E mayúscula con acento grave)
codigo ascii 213 = ì (Letra minúscula i sin punto)
codigo ascii 214 = Í (Letra i mayúscula con acento agudo)
codigo ascii 215 = Î (Letra I mayúscula con acento circunflejo)
codigo ascii 216 = Ï (Letra i mayúscula con diéresis)
codigo ascii 217 = ⌋ (Línea simple esquina de recuadro gráfico)
codigo ascii 218 = ⌌ (Línea simple esquina de recuadro gráfico)
codigo ascii 219 = █ (Bloque color pleno solido, carácter gráfico)
codigo ascii 220 = ▀ (Medio bloque negro, mitad inferior, carácter gráfico)
codigo ascii 221 = ¦ (Barra vertical partida)
codigo ascii 222 = Ì (Letra I mayúscula con acento grave)
codigo ascii 223 = ▀ (Medio bloque negro, mitad superior, carácter gráfico)
codigo ascii 224 = Ó (Letra o mayúscula con acento agudo)
codigo ascii 225 = ß (Letra alemana eszett o ese-zeta)
codigo ascii 226 = Ô (Letra O mayúscula con acento circunflejo)
codigo ascii 227 = Ò (Letra O mayúscula con acento grave)
codigo ascii 228 = õ (Letra o minúscula con tilde)
codigo ascii 229 = Õ (Letra O mayúscula con tilde)
codigo ascii 230 = µ (Signo micro)
codigo ascii 231 = þ (Letra latina thorn minúscula)
codigo ascii 232 = Þ (Letra latina thorn mayúscula)
codigo ascii 233 = Ú (Letra U mayúscula con acento agudo)
codigo ascii 234 = Û (Letra U mayúscula con acento circunflejo)
codigo ascii 235 = Ù (Letra U mayúscula con acento grave)
codigo ascii 236 = ý (Letra y minúscula con acento agudo)
codigo ascii 237 = Ý (Letra Y mayúscula con acento agudo)
codigo ascii 238 = ¯ (Macron (marca larga), superguión, guión alto)
codigo ascii 239 = ´ (Acento agudo)
codigo ascii 240 = ≡ (Símbolo matemático de congruencia, equivalencia)
codigo ascii 241 = ± (Signo mas menos)
codigo ascii 242 = ¨ (ASCII 242)
codigo ascii 243 = ¾ (Tres cuartos, fracción)
codigo ascii 244 = ¶ (Fin de párrafo - signo de calderón)
codigo ascii 245 = § (Signo de sección)
codigo ascii 246 = ÷ (Signo de división)
codigo ascii 247 = , (Cedilla , virgulilla baja)
codigo ascii 248 = ° (Signo de grado, anillo)
codigo ascii 249 = ¨ (Diéresis)
codigo ascii 250 = · (Punto centrado, punto medio, coma georgiana)
codigo ascii 251 = ¹ (Superíndice uno)
codigo ascii 252 = ³ (Superíndice tres , potencia tres , al cubo)
codigo ascii 253 = ² (Superíndice dos , al cuadrado)
codigo ascii 254 = ■ (Cuadrado negro, caracter gráfico)
codigo ascii 255 = nbsp (Espacio sin separación - non breaking space)





Si tienes algunas Correcciones y/o Sugerencias, por favor contáctame.