HDC

C Programming

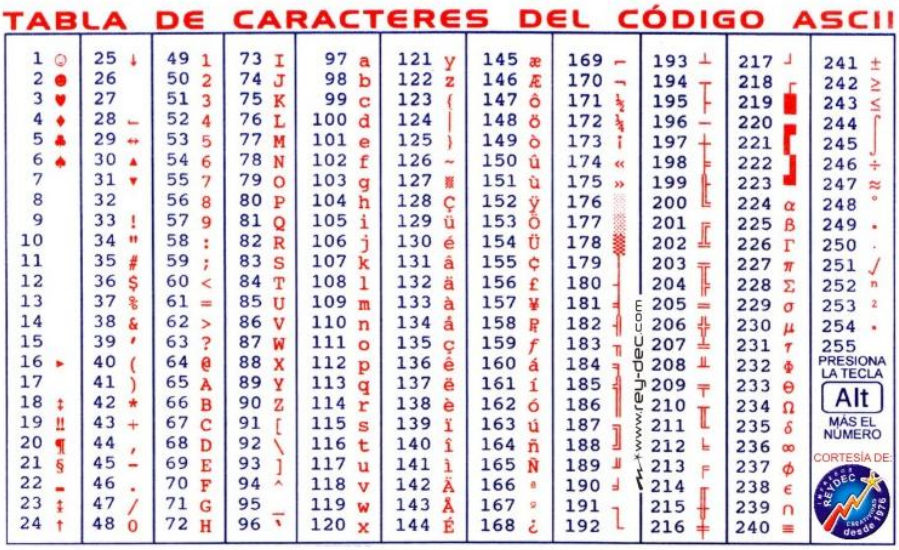
Muy buenas. En el día de hoy vamos a seguir programando en **C**. Si recordáis la última clase sobre punteros, se miró un poco, por encima, cómo funcionan los **Strings**. Pero es un tema muy amplio, el cual vamos a ampliar hoy.

Si podemos recordar de la clase anterior, la forma de **declarar** un string es la siguiente.

```
char string[20] = ”string”;
```

Cada letra se marca con una posición empezando desde el cero, y a esta posición se le asigna un valor el cual representaría la letra, este valor es escogido por el sistema de escritura **ASCII**. Cada letra estaría en la posición **n-1**, estando la primera letra en la posición cero y justo después de la última habría un valor **NULL**.

LETRA	POSICIÓN	VALOR
‘s’	0	115
‘t’	1	116
‘r’	2	114
‘i’	3	105
‘n’	4	110
‘g’	5	103
‘’	6	NULL

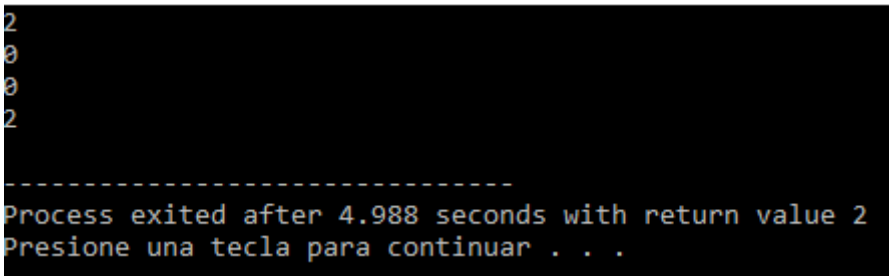


En otras clases se vio dos funciones relacionadas con la impresión y escaneo de strings. Estas fueron printf y scanf, de la **librería stdio.h**. Pero además de estas dos hay muchas otras funciones que merecen la pena ser explicadas.

El primer grupo de estas funciones se tratan en la librería **ctype.h**, la cual nos sirve para comprobar la propiedad de un carácter **devolviendo un número entero en caso de que el resultado sea verdadero y cero en caso de que sea falso**. Vamos primero a ver un ejemplo con **islower**, que **comprueba** que el carácter sea una **letra minúscula** y después mostraré todos los que hay para que podáis comprobarlos vosotros.

```
1.  #include <stdio.h>
2.  #include <ctype.h>
3.
4.  int main()
5.  {
6.
7.      printf("%d\n", islower('a'));
8.      printf("%d\n", islower('D'));
9.      printf("%d\n", islower('2'));
10. printf("%d\n", islower('c'));
11.
12. }
```

El resultado sería el siguiente, en el que se puede ver como **cuando no es una letra minúscula muestra un 0** y cuando si lo es muestra otro número.



Hay muchas más funciones con esta librería y todas funcionan de la misma forma, podéis comprobarlo con más funciones.

isalnum(c)	Alfanumérico (letras ó números)
isalpha(c)	Alfabetico , mayúscula ó minúscula
isascii(c)	Si su valor está entre 0 y 126
iscntrl(c)	Si es un caracter de control cuyo ASCII está comprendido entre 0 y 31 ó si es el código de "delete" , 127 .
islower(c)	Si es un caracter alfabético minuscula.
isupper(c)	Si es un caracter alfabético mayúscula
isdigit(c)	Si es un número comprendido entre 0 y 9
ispunct(c)	Si es un caracter de puntuación
isspace(c)	Si es el caracter espacio, tabulador, avance de línea, retorno de carro, etc.
isxdigit(c)	Si es código correspondiente a un número hexadecimal, es decir entre 0 - 9 ó A - F ó a - f .

El siguiente grupo que vamos a ver de funciones es el que pertenece a la librería **stdlib.h**. Estas funciones sirve para **transformar un valor string en un valor numérico y viceversa**. Las funciones son las siguientes:

1. **atoi**(numero): Transforma de string a decimal.
2. **atol**(numero): Transforma de string a long int.
3. **atof**(numero): Transforma de string a double.
4. **itoa**(numero, string, 10): Transforma de decimal a string.
5. **ultoa**(numero, string, 10): Transforma de unsigned int a string.

De nuevo vamos a ver un ejemplo de cómo funciona.

```c
#include <stdio.h>
#include <ctype.h>
#include <stdlib.h>

int main()
{

    char numero[3] = "254";
    int numero2;
    char numero3[3];
    int i = 0;
    int verdad = 1;

    // Comprobamos que sea un número

    for(i = 0; i < 3; i++){
         if( isdigit(numero[i] ) == 0){
             verdad = 0;
            }
    }

    if (verdad == 1){
         printf("%s\n", numero);
         numero2 = atoi(numero);
         printf("%d\n", numero2);
         itoa(numero2, numero3, 10);
         printf("%s", numero3);
    }

}
```

```
254
254
254
--------------------------------
Process exited after 4.996 seconds with return value 3
Presione una tecla para continuar . . .
```

El último grupo y más amplio que nos falta por ver es el que pertenece a la librería **string.h**. Primero vamos a ver cómo podríamos ver la **longitud de un string**, para esto vamos a usar la función **strlen()**.

```
1.  #include <stdio.h>
2.  #include <string.h>
3.
4.  int main(){
5.      char palabra[20];
6.      printf("Inserta una palabra: ");
7.      scanf("%s", &palabra);
8.      printf("\nEl numero de caracteres es:
%d", strlen(palabra));
9.  }
```

```
Inserta una palabra: HackingDesdeCero

El numero de caracteres es: 16
--------------------------------
Process exited after 70.28 seconds with return value 31
Presione una tecla para continuar . . .
```

La siguiente función sirve para copiar un string en otra variable de string, ya que al tratarse de un array tendríamos que hacerlo letra por letra en un bucle y, esto, lo simplifica mucho. La función para hacer esto es **strcpy(destino, origen).**

```
1.  #include <stdio.h>
2.  #include <string.h>
3.
4.  int main(){
5.      char palabra[20];
6.      char palabra1[20];
```

```
7.          printf("Inserta una palabra: ");
8.          scanf("%s", &palabra);
9.          strcpy(palabra1, palabra);
10.         printf("\nLa palabra copiada es:
%s", palabra1);
11.     }
```

```
Inserta una palabra: HackingDesdeCero

La palabra copiada es: HackingDesdeCero
--------------------------------
Process exited after 16.11 seconds with return value 40
Presione una tecla para continuar . . .
```

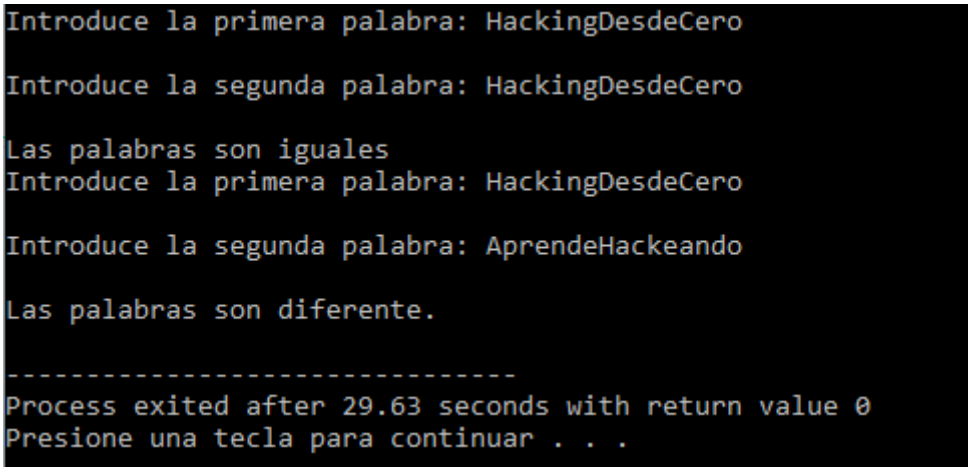
Otra función importante es **strcat(destino, origen)**, que añade el contenido de “origen” al string destino.

```
1.      #include <stdio.h>
2.      #include <string.h>
3.
4.      int main(){
5.          char palabra[20] = "HackingDesde";
6.          char palabra1[20] = "Cero";
7.          strcat(palabra, palabra1);
8.          printf("\nLa palabra concatenada es:
%s", palabra);
9.      }
```

```
La palabra concatenada es: HackingDesdeCero
--------------------------------
Process exited after 5.067 seconds with return value 44
Presione una tecla para continuar . . .
```

La última función que vamos a ver es una función que compara dos strings ya que, como hemos visto antes, lo tendríamos que hacer con un bucle, y esto es más cómodo. Para esto vamos a usar la función **strcmp(string1, string2)**, nos devuelvo 0 en caso de que sean iguales, y otro número en caso de que sean diferentes.

```
1.      #include <stdio.h>
2.      #include <string.h>
3.
4.      int main(){
5.          char palabra[20];
6.          char palabra1[20];
7.          int i;
8.
9.          for (i=0 ; i <2; i++){
10.             printf("Introduce la primera palabra: ");
11.             scanf("%s", &palabra);
12.             printf("\nIntroduce la segunda palabra:
");
13.             scanf("%s", &palabra1);
14.             if (strcmp(palabra, palabra1) == 0){
15.                 printf("\nLas palabras son
iguales\n");
16.             }
17.             else{
18.                 printf("\nLas palabras son
diferente.\n");
19.             }
20.         }
21.     }
```



Con todas estas funciones ya podréis hacer cualquier cosa que necesites en un programa respecto a strings.

Muchas gracias por leer.

Pueden seguirme en Twitter: @RoaddHDC

Contactarse por cualquier duda a: r0add@hotmail.com

Para donaciones, pueden hacerlo en bitcoin en la dirección siguiente:

1HqpPJbbWJ9H2hAZTmpXnVuoLKkP7RFSvw

También recomiendo que se unan al foro: underc0de.org/foro

Escrito por Rollth y revisado por Roadd.