

The background is a deep blue gradient. A bright, glowing light source is visible in the lower-left quadrant, creating a lens flare effect. Numerous white, spherical particles of varying sizes are scattered across the frame, some appearing to move along faint, curved paths. The overall aesthetic is scientific and futuristic.

C
a
d
e
n
a
s

Funciones relacionadas con strings ***Librería string. h***

strcpy()	copia dos strings;
strcat()	agrega una cadena a otra;
strcmp()	compara una cadena con otra;
strchr()	localiza primera instancia de un caracter dentro de un string;
strstr()	localiza la primera ocurrencia de un string dentro de otro;
strlen()	determina la longitud de un string.

Lenguaje de Programación C

isalnum(character): devuelve cierto (un entero cualquiera distinto de cero) si character es una letra o dígito, y falso (el valor entero 0) en caso contrario.

isalpha(character): devuelve cierto si character es una letra, y falso en caso contrario.

isblank(character): devuelve cierto si character es un espacio en blanco o un tabulador.

isdigit(character) devuelve cierto si character es un dígito, y falso en caso contrario.

isspace(character): devuelve cierto si character es un espacio en blanco, un salto de línea, un retorno de carro, un tabulador, etc., y falso en caso contrario.

islower(character): devuelve cierto si character es una letra minúscula, y falso en caso contrario.

isupper(character): devuelve cierto si character es una letra mayúscula, y falso en caso contrario.

toupper(character): devuelve la mayúscula asociada a character, si la tiene; si no, devuelve el mismo character.

Lenguaje de Programación C

tolower(caracter): devuelve la minúscula asociada a caracter, si la tiene; si no, devuelve el mismo caracter.

isalnum(caracter): devuelve cierto (un entero cualquiera distinto de cero) si caracter es una letra o dígito, y falso (el valor entero 0) en caso contrario.

isalpha(caracter): devuelve cierto si caracter es una letra, y falso en caso contrario.

isblank(caracter): devuelve cierto si caracter es un espacio en blanco o un tabulador.

isdigit(caracter) devuelve cierto si caracter es un dígito, y falso en caso contrario.

isspace(caracter): devuelve cierto si caracter es un espacio en blanco, un salto de línea, un retorno de carro, un tabulador, etc., y falso en caso contrario.

islower(caracter): devuelve cierto si caracter es una letra minúscula, y falso en caso contrario.

isupper(caracter): devuelve cierto si caracter es una letra mayúscula, y falso en caso contrario.

toupper(caracter): devuelve la mayúscula asociada a caracter, si la tiene; si no, devuelve el mismo caracter.

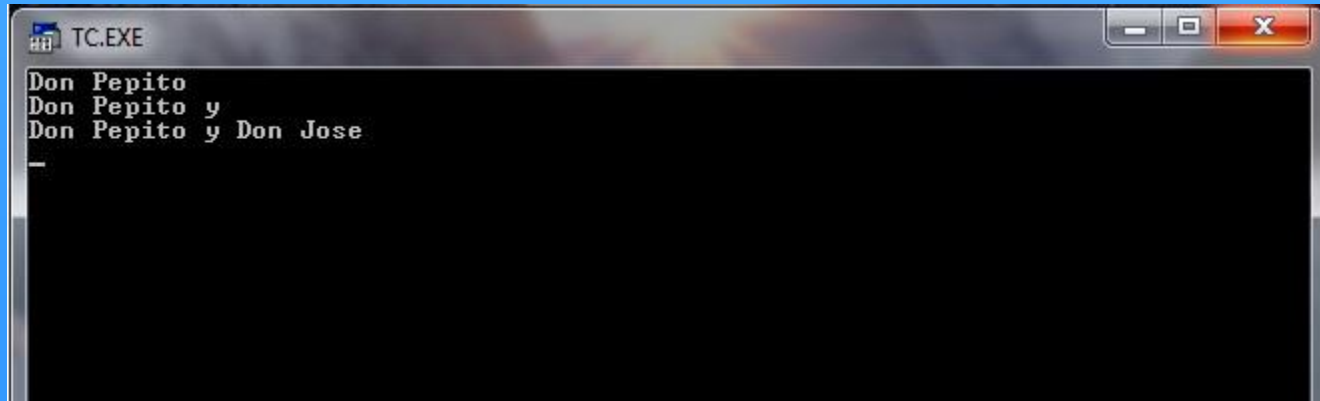
tolower(caracter): devuelve la minúscula asociada a caracter, si la tiene; si no, devuelve el mismo caracter.

strcat

Copia la *cadena2* al final de la *cadena1*.

Se declara de la siguiente manera:

```
#include<stdio.h>
#include<string.h>
int main()
{
char texto1[]="Don Pepito";
char texto2[]=" y ";
char texto3[]="Don Jose";
printf("%s\n",texto1);
strcat(texto1,texto2);
printf("%s\n",texto2);
strcat(texto1,texto3);
printf("%s\n",texto2);
getchar();
return 0;
}
```



```
TC.EXE
Don Pepito
Don Pepito y
Don Pepito y
Don Pepito y Don Jose
-
```

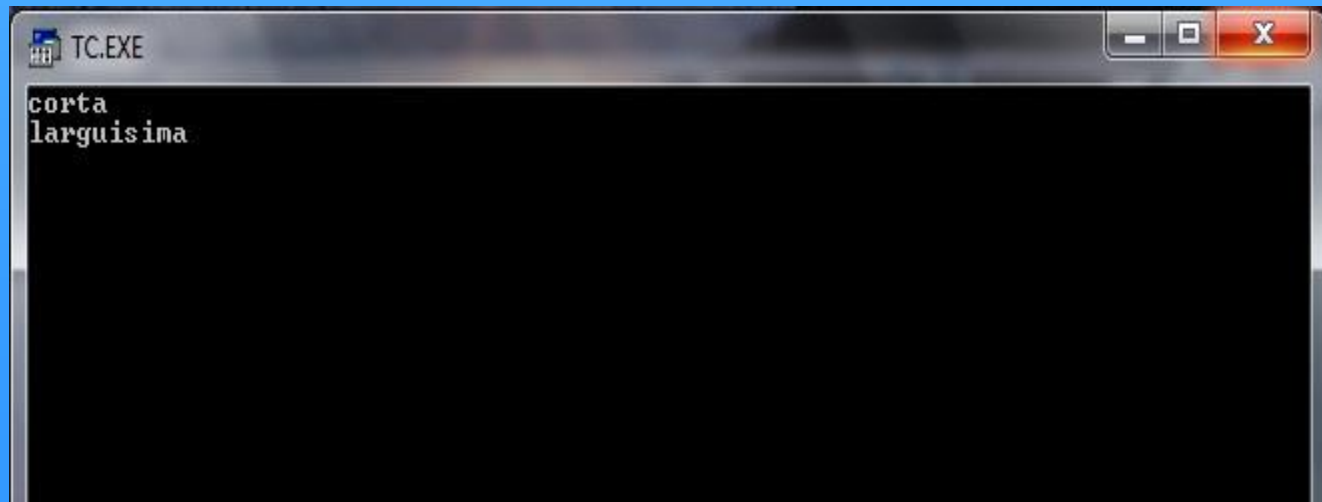
strcpy

Se utiliza para copiar una cadena de caracteres (fuente) en el lugar que ocupaba otra (destino). Esta copia es destructiva, o sea, que todos los caracteres que estaban en la cadena destino desaparecen, aunque la cadena destino fuera más larga que la cadena fuente. La cadena destino se pone como primer argumento de la función y la cadena fuente como segundo.

Copia el contenido de *cadena2* en *cadena1*. *cadena2* puede ser una variable o una cadena directa (por ejemplo "hola"). Debemos tener cuidado de que la cadena destino (*cadena1*) tenga espacio suficiente para albergar a la cadena origen (*cadena*). 2

strcpy

```
#include<stdio.h>
#include<string.h>
int main()
{
char texto1[]="corta";
char texto2[]="mediana";
char texto3[]="larguísima";
strcpy(texto2,texto1);
printf("%s\n",texto2);
strcpy(texto2,texto3);
printf("%s\n",texto2);
getch();
return 0;
}
```



strcmp

Compara dos cadenas alfabéticamente.

La comparación es lexicográfica, es decir de manera alfabética de la A a la Z, además las letras minúsculas son mayores que las mayúsculas

Se declara de la siguiente manera:

```
#include <string.h>
int strcmp(const char *cadena1, const char *cadena2);
```

Compara *cadena1* y *cadena2*. Si son iguales devuelve 0. Un número negativo si *cadena1* va antes que *cadena2* y un número positivo si es al revés:

cadena1 == *cadena2* -> 0

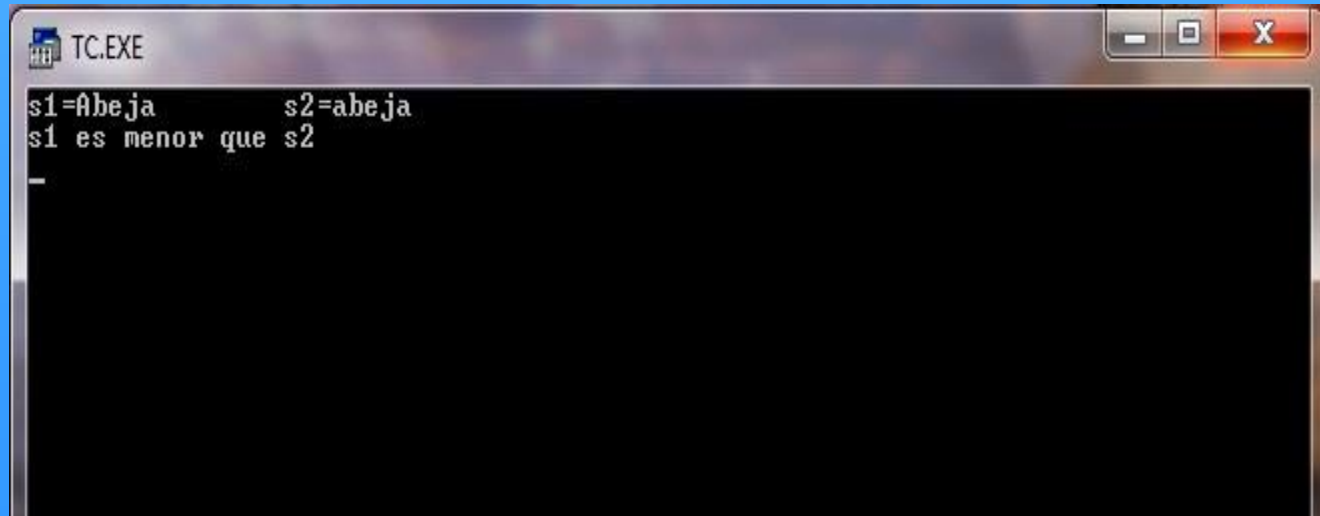
cadena1 < *cadena2* -> número negativo

cadena1 > *cadena2* -> número positivo

Valor de Retorno	
Si s1 es ...	valor de retorno es ...
menor que s2	<0
el mismo que s2	== 0
mayor que s2	> 0

strcmp

```
#include <stdio.h>
#include <string.h>
int main()
{
    char s1[6] = "Abeja";
    char s2[6] = "abeja";
    int i; printf( "s1=%s\t", s1 );
    printf( "s2=%s\n", s2 );
    i = strcmp( s1, s2 );
    printf( "s1 es " );
    if( i < 0 ) printf( "menor que" );
    else if( i > 0 ) printf( "mayor que" );
    else printf( "igual a" );
    printf( " s2\n" );
    return 0;
}
```



TC.EXE

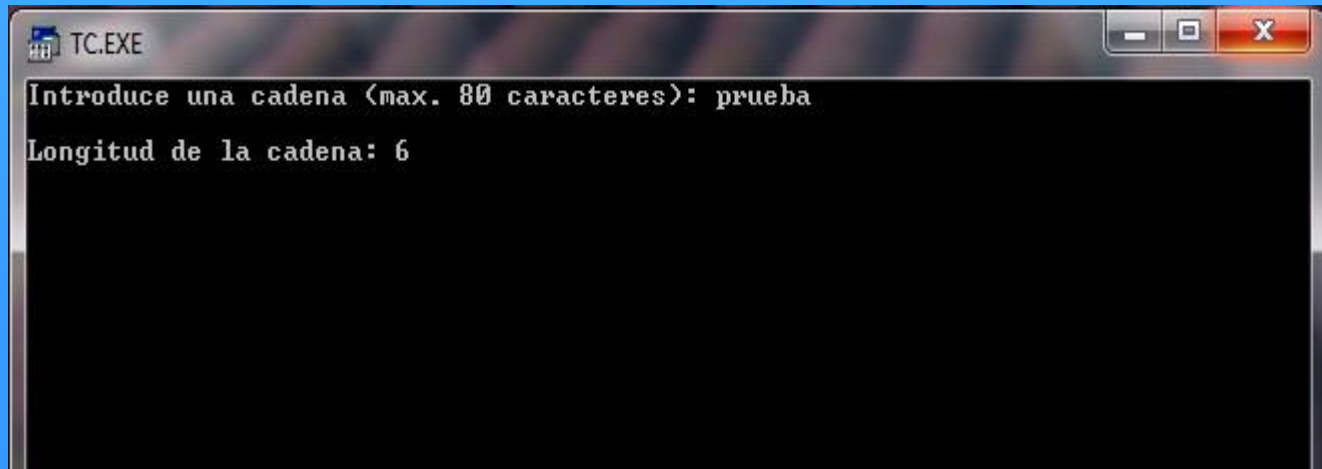
```
s1=Abeja      s2=abeja
s1 es menor que s2
-
```

Lenguaje de Programación C

strlen

Esta función nos devuelve el número de caracteres que tiene la cadena (sin contar el '\0').

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
#define MAXLON 80
int main()
{
    char a[MAXLON+1];
    int longitud;
    printf ("Introduce una cadena (max. %d caracteres): ", MAXLON);
    scanf("%s",&a);
    longitud = strlen(a);
    printf ("\nLongitud de la cadena: %d\n", longitud);
    getch();
    return 0;
}
```



```
TC.EXE
Introduce una cadena (max. 80 caracteres): prueba
Longitud de la cadena: 6
```