Below is a table listing some common string functions used in C programming, along with their descriptions and examples. These functions are available in the string.h header file.

Function	**Description**	**Example**
`strlen`	Calculates the length of a string (excluding null character).	`int len = strlen("Hello, World!");`
`strcpy`	Copies a string to another string.	`strcpy(dest, src);`
`strncpy`	Copies a specified number of characters from one string to another.	`strncpy(dest, src, 5);`
`strcat`	Concatenates two strings.	`strcat(dest, src);`
`strncat`	Concatenates a specified number of characters from two strings.	`strncat(dest, src, 5);`
`strcmp`	Compares two strings lexicographically.	`int result = strcmp(str1, str2);`
`strncmp`	Compares a specified number of characters from two strings lexicographically.	`int result = strncmp(str1, str2, 5);`
`strchr`	Finds the first occurrence of a character in a string.	`char *ptr = strchr(str, 'o');`
`strrchr`	Finds the last occurrence of a character in a string.	`char *ptr = strrchr(str, 'o');`
`strstr`	Finds the first occurrence of a substring in a string.	`char *ptr = strstr(str, "World");`
`strspn`	Finds the length of the initial segment of a string consisting of characters in a given set.	`size_t len = strspn(str, "abc");`
`strcspn`	Finds the length of the initial segment of a string consisting of characters not in a given set.	`size_t len = strcspn(str, "abc");`
`strpbrk`	Finds the first occurrence in a string of any character from a given set.	`char *ptr = strpbrk(str, "aeiou");`
`strtok`	Splits a string into tokens based on a delimiter.	`char *token = strtok(str, " ,.!");`
`strerror`	Returns a string describing an error code.	`char *err_msg = strerror(errno);`
`memchr`	Searches a memory block for the first occurrence of a character.	`void *ptr = memchr(arr, 'x', 10);`
`memcmp`	Compares two memory blocks.	`int result = memcmp(arr1, arr2, 10);`
`memcpy`	Copies a memory block to another memory block.	`memcpy(dest, src, 10);`
`memmove`	Copies a memory block to another memory block, with overlapping support.	`memmove(dest, src, 10);`
`memset`	Sets a memory block to a specified character.	`memset(arr, '0', 10);`