

C Programming

Strings

String functions

C is a very powerful language using strings. It has a rich set of built-in functions that facilitate the use of strings in a program.

The built-in string functions require the following header file when used:

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

1. **Finding the length of a string** (i.e., finding the number of characters in a string)

strlen()

- strlen() returns the number of characters in a string **excluding** the NULL character

```
/*  
Built-in string functions  
*/  
  
#include <stdio.h>  
#include <string.h>  
  
int main(void)  
{  
    char name1[] = "Sharon";  
    char name2[10] = "Mark";  
    char *name3 = "Patrick";  
  
    int length = 0;
```

```

    // length is assigned the number of characters in the string
    inside array name1
    length = strlen(name1);

    // Display the number of characters in the following strings
    printf("\n%d %d %d %d %d", strlen(name1), strlen(name2),
    strlen(name3), strlen("Mary"), length);

    return 0;
}

```

Repl 18.1: <https://replit.com/@michaelTUDublin/181-strlen#main.c>

2. **Copying a string** (i.e., copying from one string to another string)

strcpy(string1, string2)

strcpy(**destination_string**, **source_string**)

- strcpy() copies the contents of the source string to the destination string
- the source string must be NULL terminated
- strcpy() assumes that the destination string is large enough to hold the string being copied to it

```

/*
Built-in string functions
*/

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

int main(void)
{
    char name1[] = "Sharon";

```

```

char name2[10] = "Mark";

// Copy the contents of name1 to name2
strcpy(name2, name1);

// Display the number of characters in the following strings
printf("\n%s", name2);

return 0;

} // end main()

```

Repl 18.2: <https://replit.com/@michaelTUDublin/182-strcpy>

3. **Concatenating a string** (i.e., appending/joining one string to the end of another string)

strcat(string1, string2)

strcat(destination_string, source_string)

- strcat() appends/joins the contents of the source string to the end of the destination string
- **Both** the source string and destination string must be NULL terminated
- strcat() assumes that the destination string is large enough to hold the newly appended string

```

/*
String functions
*/

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

```

```

int main()
{
    char str1[17] = "first and ";
    char str2[] = "second";

    printf("\n%s", str1);
    printf("\n%s", str2);

    // Concatenate / Join the contents of str1 to str2
    strcat(str1, str2);

    // Display the newly concatenated string
    printf("\n%s", str1);

    return 0;

} // end main()

```

Repl 18.3: <https://replit.com/@michaelTUDublin/183-strcat>

4. **Comparing two strings** (i.e., comparing all the characters of one string with another string)

strcmp(string1, string2)

- strcmp() compares two NULL terminated strings
- strcmp() returns an integer
 - **returns 0** if both strings are **identical**
 - returns a non-zero if the strings are different

```

/*
String functions
*/

```

```

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

int main()
{
    char str1[21];
    char str2[21];
    int result = 0;

    printf("Enter first string\n");
    scanf("%s", str1);
    //fgets(str1, 20, stdin);

    printf("Enter second string\n");
    scanf("%s", str2);
    //fgets(str2, 20, stdin);

    // Compare both strings and check if they are the same
    result = strcmp(str1, str2);

    //Check if the strings are the same
    if(result == 0)
    {
        printf("\nStrings are the same");
    } // end if
    else
    {
        printf("\nDifferent strings");
    } // end else

    return 0;

} // end main()

```

Repl 18.4: <https://replit.com/@michaelTUDublin/184-strcmp>

Finally ...

The above built-in functions in C are the most commonly used. C has a rich suite of other built-in string functions such as:

- `strchr(string, character_being_searched);` // Finds the first occurrence of a character in the string
- `strncmp(string1, string2, n);` // compares the first n characters in the two strings
- `strncpy(destination_string, source_string, n);` // copies the first n characters from the source string to the destination string