

# String Functions in C

*What are strings?*

*A string in C (also known as C string) is an array of characters, followed by a NULL character. To represent a string, a set of characters are enclosed within double quotes (") or in other words a string is a sequence of characters terminated with a null character \0. For example:*

```
char c[] = "c string";
```

*When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end by default.*

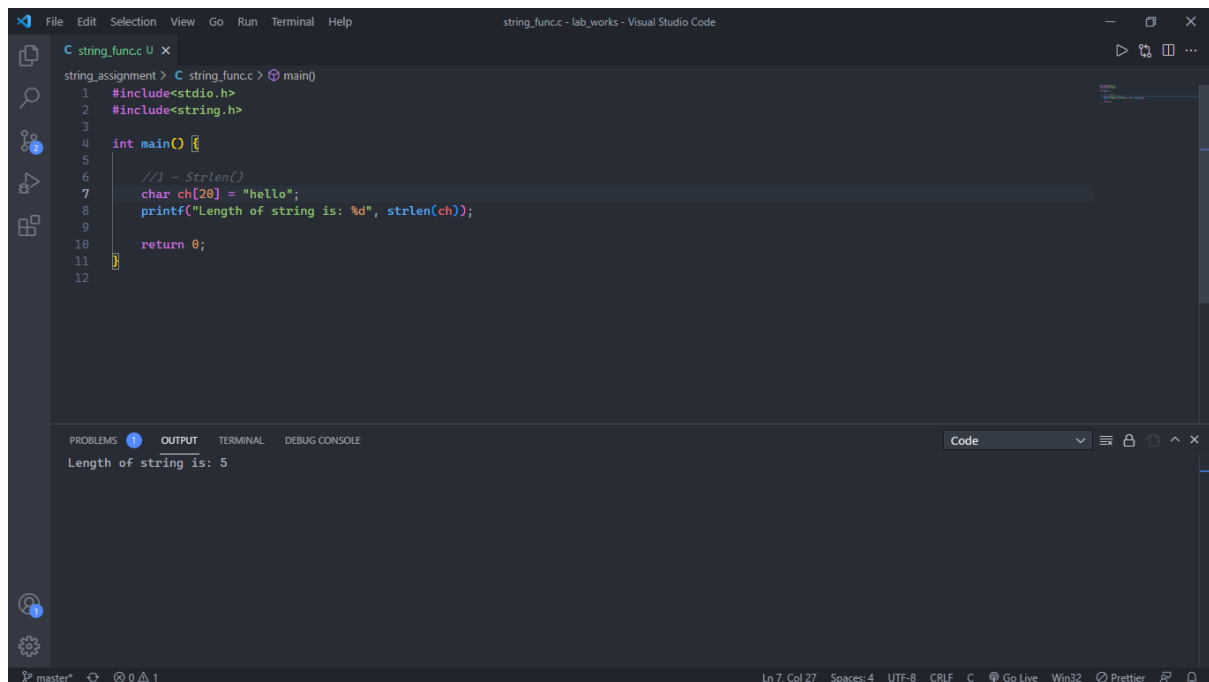
There are about 18 string functions provided by the “string.h” header file in C. They are :

<b>strcat ( )</b>	Concatenates str2 at the end of str1
<b>strncat ( )</b>	Appends a portion of string to another
<b>strcpy ( )</b>	Copies str2 into str1
<b>strncpy ( )</b>	Copies given number of characters of one string to another
<b>strlen ( )</b>	Gives the length of str1
<b>strcmp ( )</b>	Returns 0 if str1 is same as str2. Returns <0 if str1 < str2. Returns >0 if str1 > str2

<b>strcmpi ( )</b>	Same as strcmp() function. But, this function negotiates case. "A" and "a" are treated as same.
<b>strchr ( )</b>	Returns pointer to first occurrence of char in str1
<b>strrchr ( )</b>	last occurrence of given character in a string is found
<b>strstr ( )</b>	Returns pointer to first occurrence of str2 in str1
<b>strrstr ( )</b>	Returns pointer to last occurrence of str2 in str1
<b>strdup ( )</b>	Duplicates the string
<b>strlwr ( )</b>	Converts string to lowercase
<b>strupr ( )</b>	Converts string to uppercase
<b>strrev ( )</b>	Reverses the given string
<b>strset ( )</b>	Sets all character in a string to given character
<b>strnset ( )</b>	It sets the portion of characters in a string to given character
<b>strtok ( )</b>	Tokenizing given string using delimiter

**String function programs of commonly used methods are :**

# 1 - strlen()



The screenshot shows a Visual Studio Code editor window with a C file named `string_func.c`. The code defines a `main` function that includes `stdio.h` and `string.h`. It declares a character array `ch` with the value "hello" and uses `strlen(ch)` to calculate its length, which is then printed. The output window at the bottom shows the result: "Length of string is: 5".

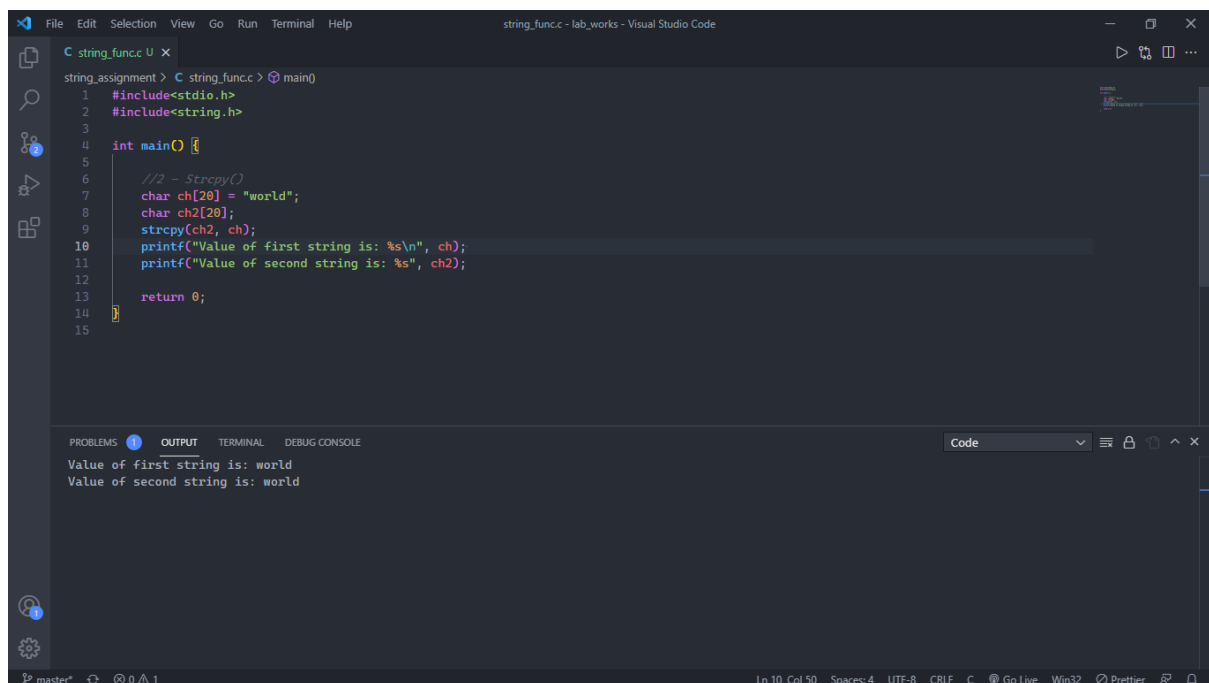
```
string_assignment > C string_func.c > main()
1  #include<stdio.h>
2  #include<string.h>
3
4  int main() {
5
6      //1 - Strlen()
7      char ch[20] = "hello";
8      printf("Length of string is: %d", strlen(ch));
9
10     return 0;
11 }
12
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Length of string is: 5

Ln 7, Col 27 Spaces: 4 UTF-8 CRLF C Go Live Win32 Prettier

# 2 - strcpy()



The screenshot shows a Visual Studio Code editor window with a C file named `string_func.c`. The code defines a `main` function that includes `stdio.h` and `string.h`. It declares two character arrays, `ch` with the value "world" and `ch2`. It then uses `strcpy(ch2, ch)` to copy the contents of `ch` into `ch2`. Both strings are then printed. The output window at the bottom shows the result: "Value of first string is: world" and "Value of second string is: world".

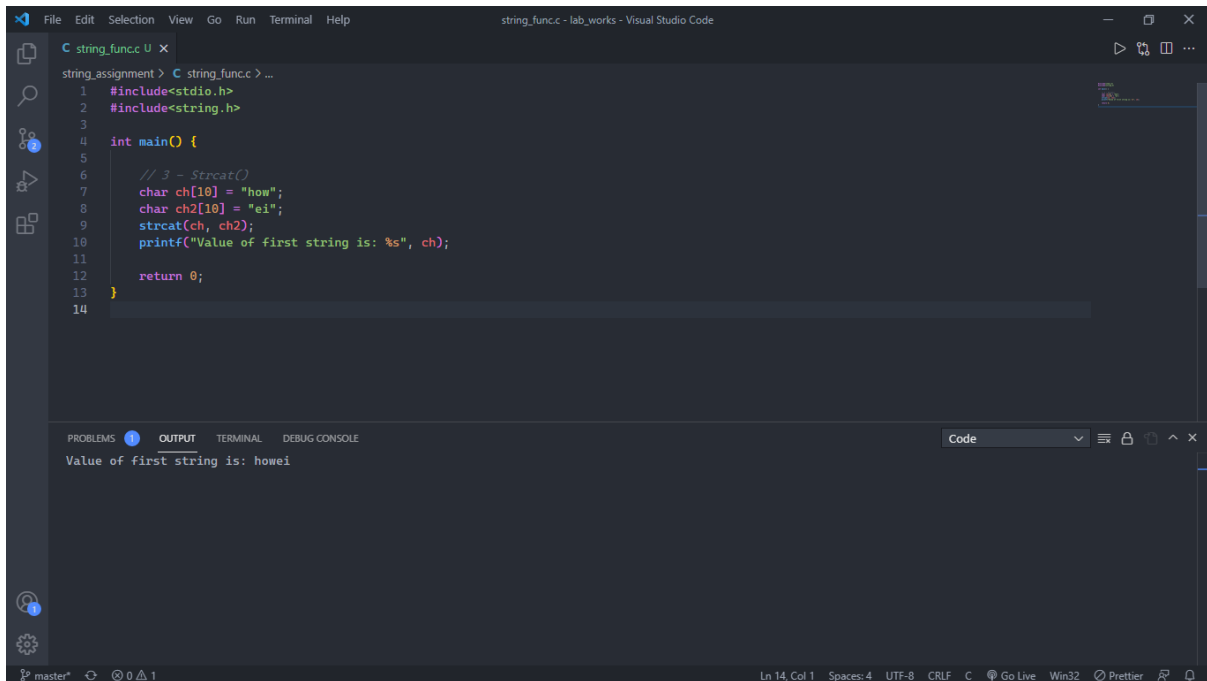
```
string_assignment > C string_func.c > main()
1  #include<stdio.h>
2  #include<string.h>
3
4  int main() {
5
6      //2 - strcpy()
7      char ch[20] = "world";
8      char ch2[20];
9      strcpy(ch2, ch);
10     printf("Value of first string is: %s\n", ch);
11     printf("Value of second string is: %s", ch2);
12
13     return 0;
14 }
15
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Value of first string is: world  
Value of second string is: world

Ln 10, Col 50 Spaces: 4 UTF-8 CRLF C Go Live Win32 Prettier

### 3 - strcat()



The screenshot shows a Visual Studio Code editor window with a C file named `string_func.c`. The code implements the `strcat` function. It includes `<stdio.h>` and `<string.h>`. The `main` function declares two character arrays: `ch[10] = "how"` and `ch2[10] = "ei"`. It then calls `strcat(ch, ch2);` and prints the result using `printf("Value of first string is: %s", ch);`. The output in the terminal shows "Value of first string is: howei".

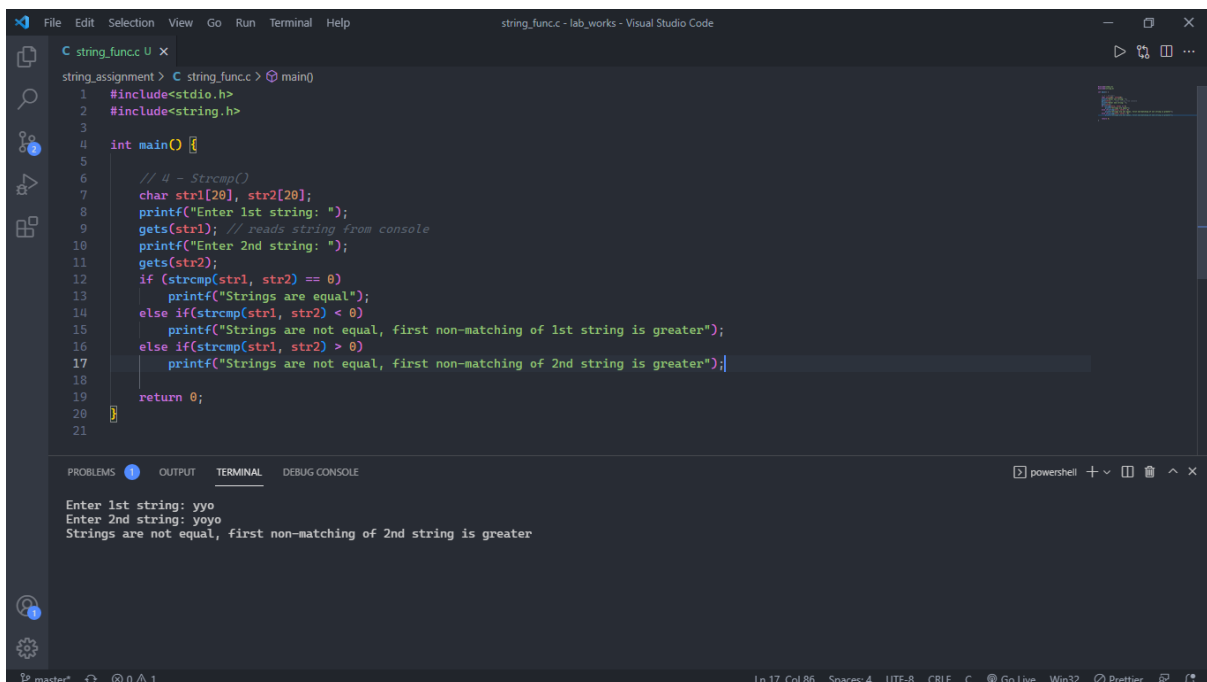
```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5
6     // 3 - Strcat()
7     char ch[10] = "how";
8     char ch2[10] = "ei";
9     strcat(ch, ch2);
10    printf("Value of first string is: %s", ch);
11
12    return 0;
13 }
14
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Value of first string is: howei

Ln 14, Col 1 Spaces: 4 UTF-8 CRLF C Go Live Win32 Prettier

### 4 - strcmp()



The screenshot shows a Visual Studio Code editor window with a C file named `string_func.c`. The code implements the `strcmp` function. It includes `<stdio.h>` and `<string.h>`. The `main` function declares two character arrays: `str1[20]` and `str2[20]`. It prompts the user to enter two strings using `printf` and `gets`. It then compares the strings using `strcmp(str1, str2)` and prints the result based on the comparison. The output in the terminal shows "Enter 1st string: yyo", "Enter 2nd string: yoyo", and "Strings are not equal, first non-matching of 2nd string is greater".

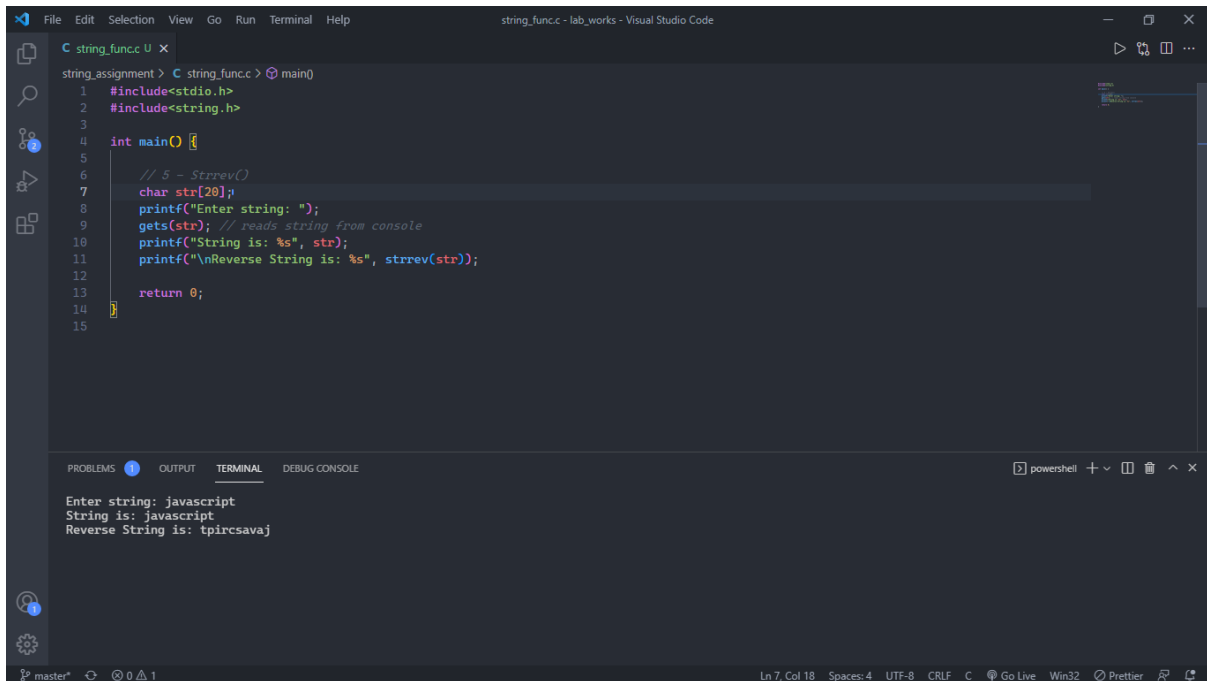
```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5
6     // 4 - Strcmp()
7     char str1[20], str2[20];
8     printf("Enter 1st string: ");
9     gets(str1); // reads string from console
10    printf("Enter 2nd string: ");
11    gets(str2);
12    if (strcmp(str1, str2) == 0)
13        printf("Strings are equal");
14    else if (strcmp(str1, str2) < 0)
15        printf("Strings are not equal, first non-matching of 1st string is greater");
16    else if (strcmp(str1, str2) > 0)
17        printf("Strings are not equal, first non-matching of 2nd string is greater");
18
19    return 0;
20 }
21
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Enter 1st string: yyo  
Enter 2nd string: yoyo  
Strings are not equal, first non-matching of 2nd string is greater

Ln 17, Col 86 Spaces: 4 UTF-8 CRLF C Go Live Win32 Prettier

## 5 - strrev()



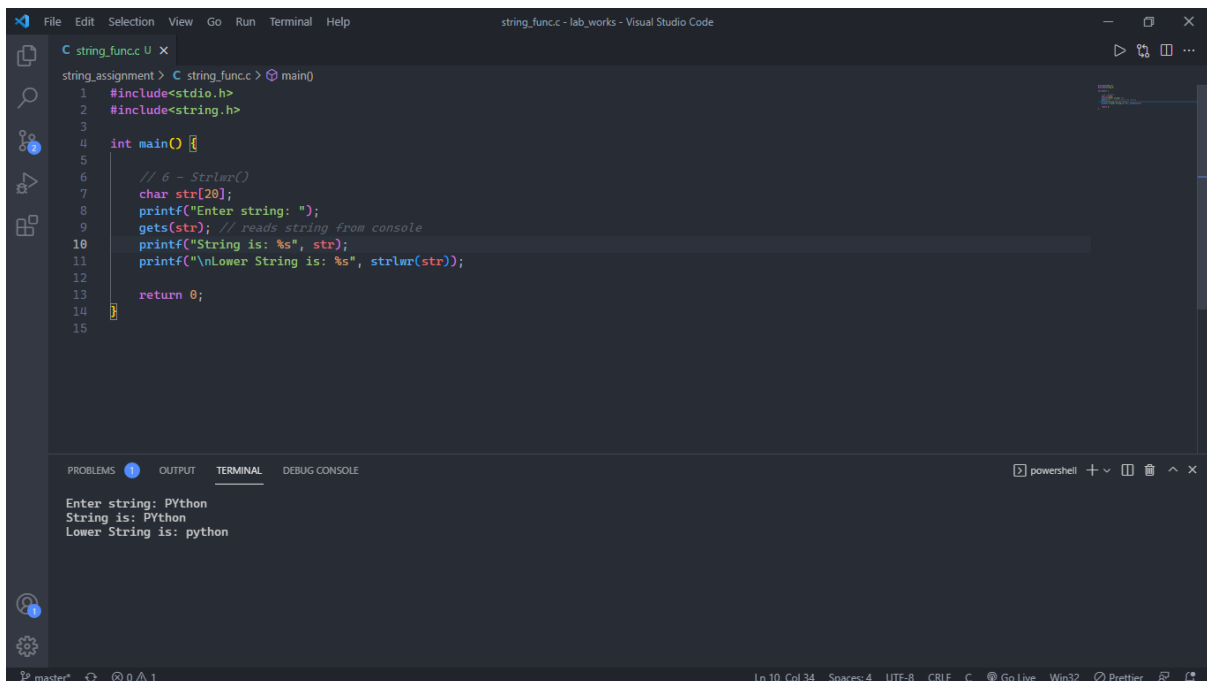
The screenshot shows a Visual Studio Code editor window with a C file named `string_func.c`. The code defines a `main` function that prompts the user to enter a string, reads it using `gets`, and then prints the original string and its reverse using `strrev`. The terminal output shows the user entering "javascript" and the program outputting "String is: javascript" and "Reverse String is: tpircsavaJ".

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 5 - Strrev()
6     char str[20];
7     printf("Enter string: ");
8     gets(str); // reads string from console
9     printf("String is: %s", str);
10    printf("\nReverse String is: %s", strrev(str));
11
12    return 0;
13 }
```

Terminal Output:

```
Enter string: javascript
String is: javascript
Reverse String is: tpircsavaJ
```

## 6 - strlwr()



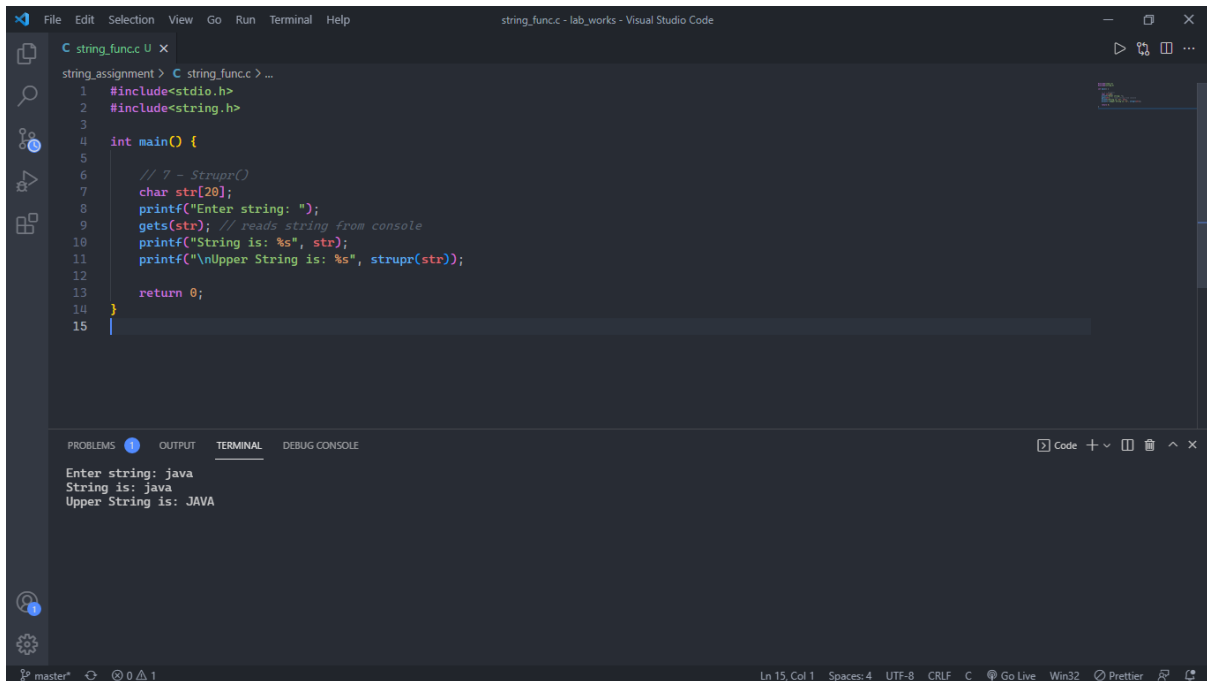
The screenshot shows a Visual Studio Code editor window with a C file named `string_func.c`. The code defines a `main` function that prompts the user to enter a string, reads it using `gets`, and then prints the original string and its lowercase version using `strlwr`. The terminal output shows the user entering "Python" and the program outputting "String is: Python" and "Lower String is: python".

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 6 - Strlwr()
6     char str[20];
7     printf("Enter string: ");
8     gets(str); // reads string from console
9     printf("String is: %s", str);
10    printf("\nLower String is: %s", strlwr(str));
11
12    return 0;
13 }
```

Terminal Output:

```
Enter string: Python
String is: Python
Lower String is: python
```

## 7 -strupr()



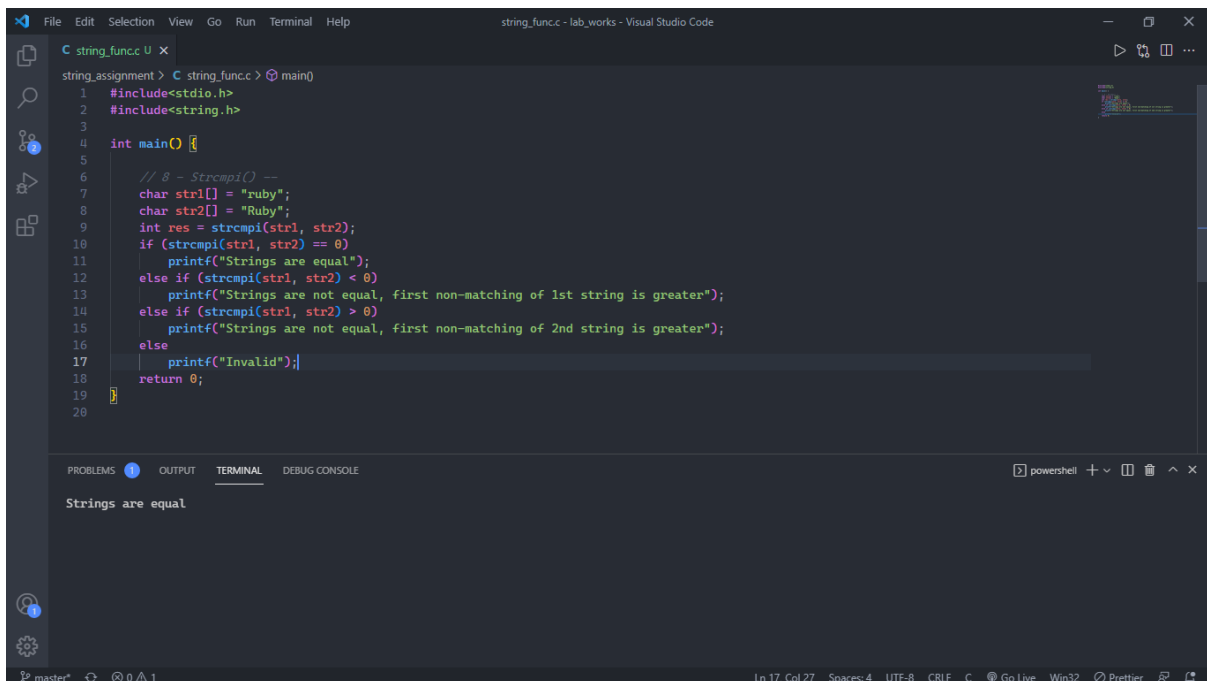
The screenshot shows a Visual Studio Code editor with a C file named `string_func.c`. The code defines a `main` function that prompts the user to enter a string, reads it using `gets`, and then prints the original string and its uppercase version using `strupr`. The terminal window at the bottom shows the execution output.

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 7 - Strupr()
6     char str[20];
7     printf("Enter string: ");
8     gets(str); // Reads string from console
9     printf("String is: %s", str);
10    printf("\nUpper String is: %s", strupr(str));
11
12    return 0;
13 }
14
15
```

TERMINAL

```
Enter string: java
String is: java
Upper String is: JAVA
```

## 8 - strcmpi()



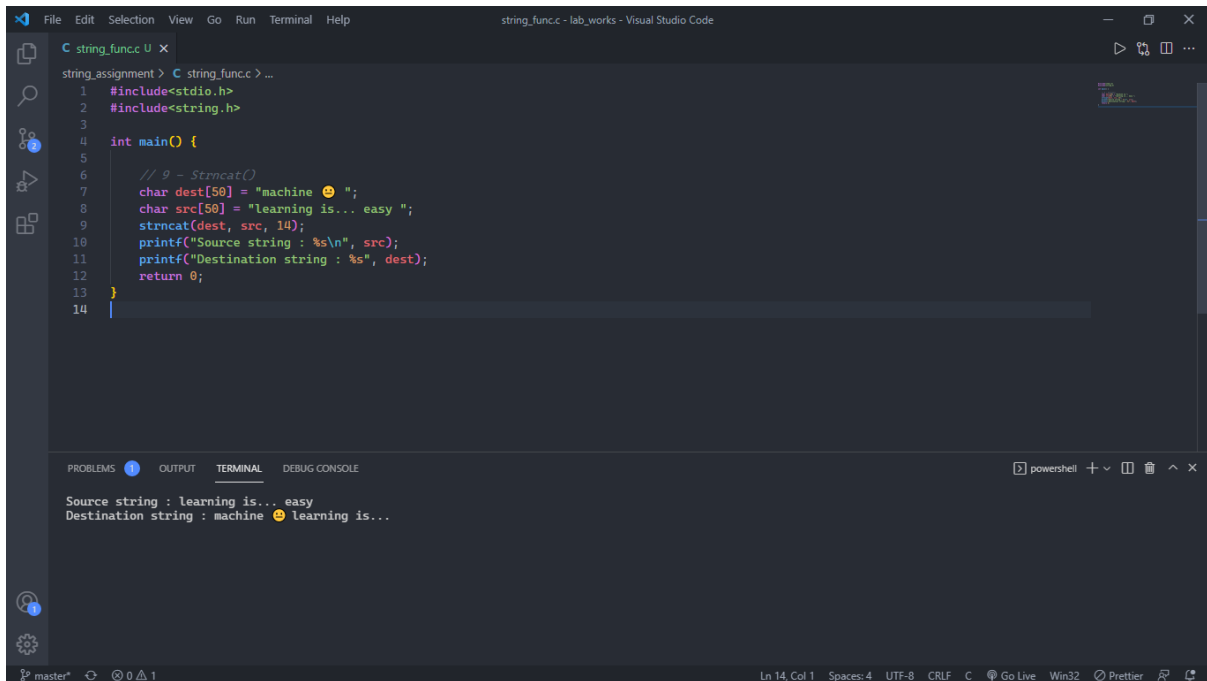
The screenshot shows a Visual Studio Code editor with a C file named `string_func.c`. The code defines a `main` function that compares two strings, `str1` and `str2`, using `strcmpi`. It prints the result of the comparison, indicating if the strings are equal, not equal (and which is greater), or invalid. The terminal window at the bottom shows the execution output.

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 8 - Strcmpi() --
6     char str1[] = "ruby";
7     char str2[] = "Ruby";
8     int res = strcmpi(str1, str2);
9     if (strcmpi(str1, str2) == 0)
10        printf("Strings are equal");
11    else if (strcmpi(str1, str2) < 0)
12        printf("Strings are not equal, first non-matching of 1st string is greater");
13    else if (strcmpi(str1, str2) > 0)
14        printf("Strings are not equal, first non-matching of 2nd string is greater");
15    else
16        printf("Invalid");
17    return 0;
18 }
19
20
```

TERMINAL

```
Strings are equal
```

## 9 - strncat()



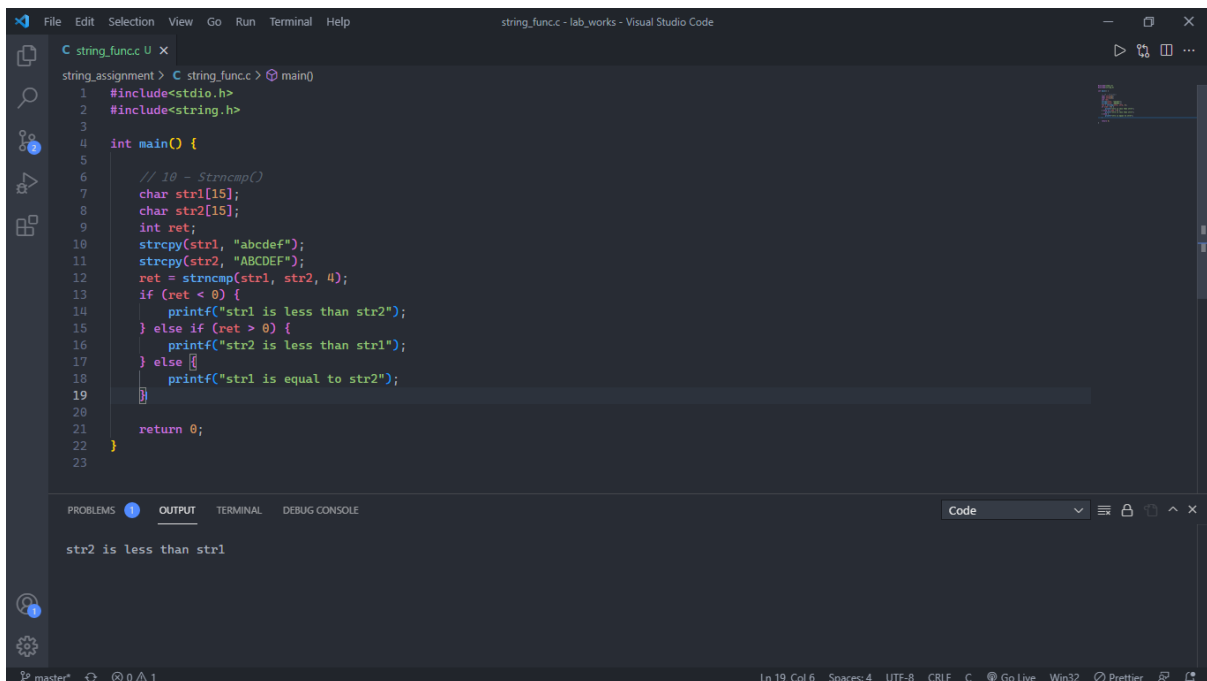
The screenshot shows a Visual Studio Code editor with a C file named `string_func.c`. The code defines a `main` function that uses `strncat` to concatenate a substring of "learning is... easy " into the string "machine 😊 ". The terminal output shows the source and destination strings.

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 9 - Strncat()
6     char dest[50] = "machine 😊 ";
7     char src[50] = "learning is... easy ";
8     strncat(dest, src, 14);
9     printf("Source string : %s\n", src);
10    printf("Destination string : %s", dest);
11    return 0;
12 }
```

Terminal Output:

```
Source string : learning is... easy
Destination string : machine 😊 learning is...
```

## 10 - strncmp()



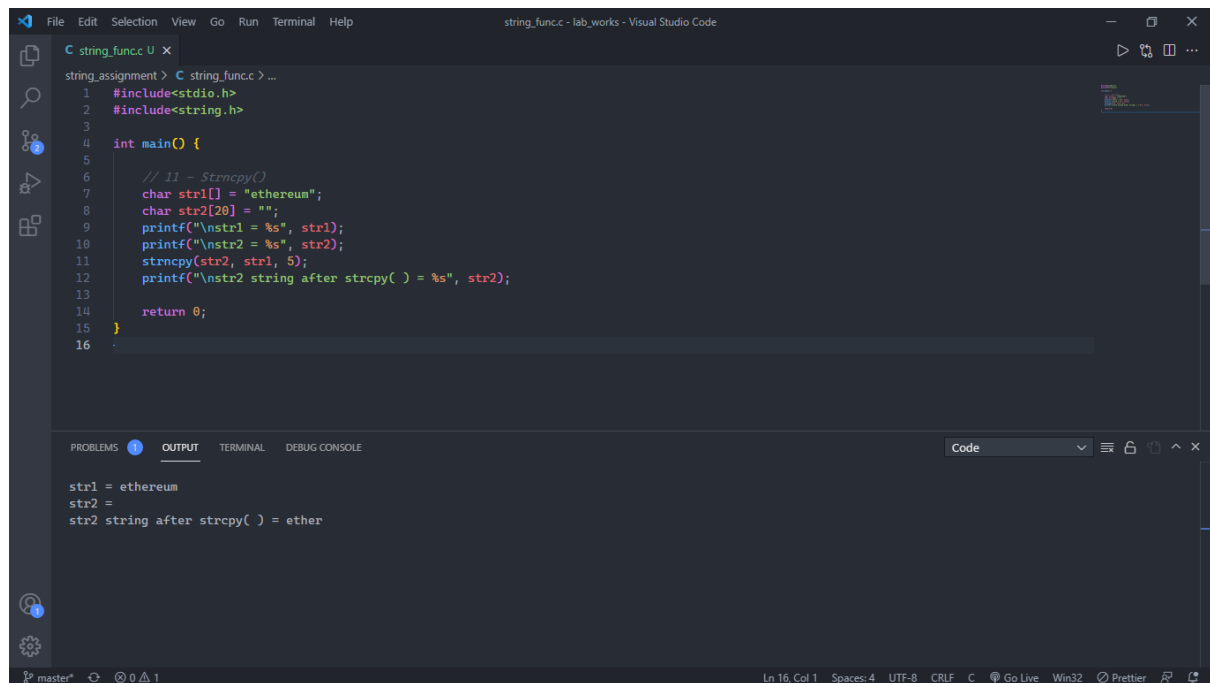
The screenshot shows a Visual Studio Code editor with a C file named `string_func.c`. The code defines a `main` function that compares the first 4 characters of "abcdef" and "ABCDEF" using `strncmp`. The terminal output shows that "str2 is less than str1".

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 10 - Strncmp()
6     char str1[15];
7     char str2[15];
8     int ret;
9     strcpy(str1, "abcdef");
10    strcpy(str2, "ABCDEF");
11    ret = strncmp(str1, str2, 4);
12    if (ret < 0) {
13        printf("str1 is less than str2");
14    } else if (ret > 0) {
15        printf("str2 is less than str1");
16    } else {
17        printf("str1 is equal to str2");
18    }
19    return 0;
20 }
```

Terminal Output:

```
str2 is less than str1
```

## 11 - strncpy()



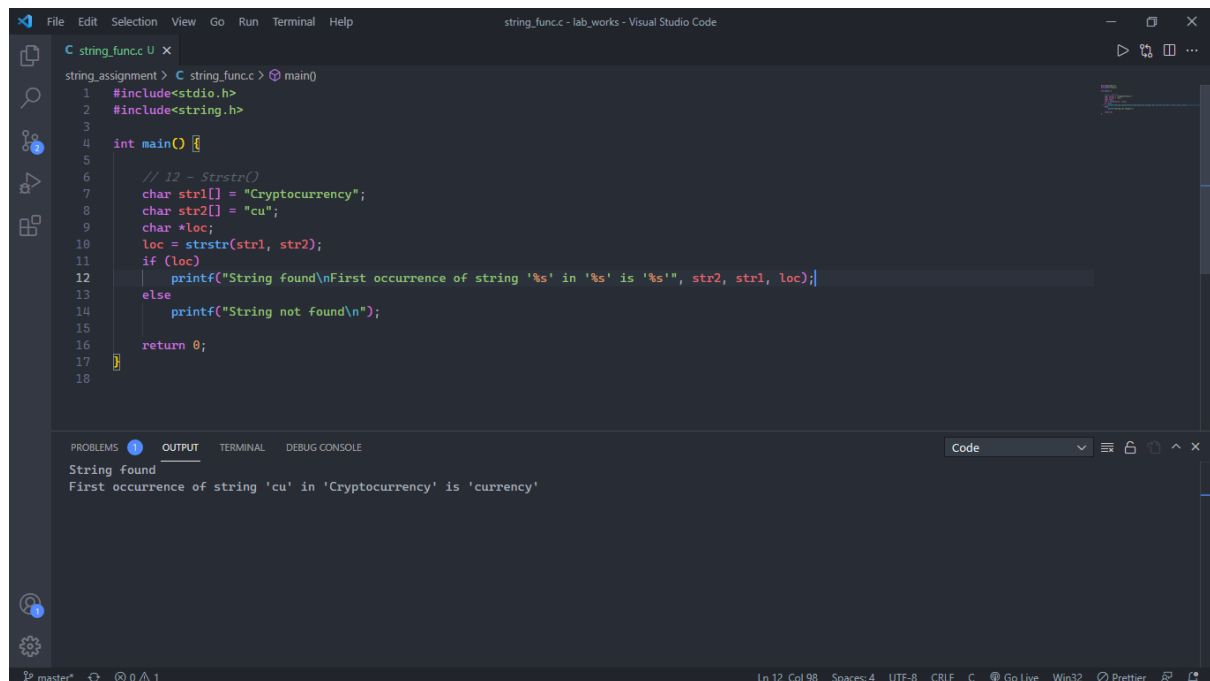
The screenshot shows a Visual Studio Code editor window with a C program named `string_func.c`. The program demonstrates the use of `strncpy()` to copy a substring from one string to another. The code is as follows:

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 11 - strncpy()
6     char str1[] = "ethereum";
7     char str2[20] = "";
8     printf("\nstr1 = %s", str1);
9     printf("\nstr2 = %s", str2);
10    strncpy(str2, str1, 5);
11    printf("\nstr2 string after strcpy( ) = %s", str2);
12
13    return 0;
14 }
```

The output window shows the following results:

```
str1 = ethereum
str2 =
str2 string after strcpy( ) = ether
```

## 12 - strstr()



The screenshot shows a Visual Studio Code editor window with a C program named `string_func.c`. The program demonstrates the use of `strstr()` to find the first occurrence of a substring within a string. The code is as follows:

```
1 #include<stdio.h>
2 #include<string.h>
3
4 int main() {
5     // 12 - strstr()
6     char str1[] = "Cryptocurrency";
7     char str2[] = "cu";
8     char *loc;
9     loc = strstr(str1, str2);
10    if (loc)
11        printf("String found\nFirst occurrence of string '%s' in '%s' is '%s'", str2, str1, loc);
12    else
13        printf("String not found\n");
14
15    return 0;
16 }
```

The output window shows the following results:

```
String found
First occurrence of string 'cu' in 'Cryptocurrency' is 'currency'
```