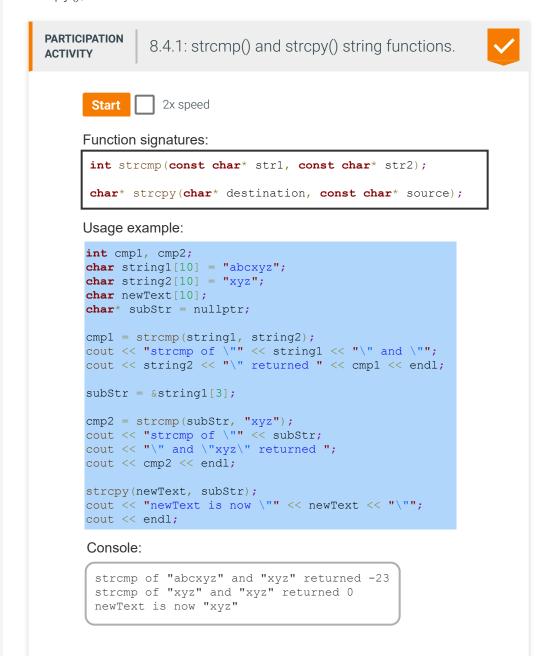
8.4 String functions with pointers

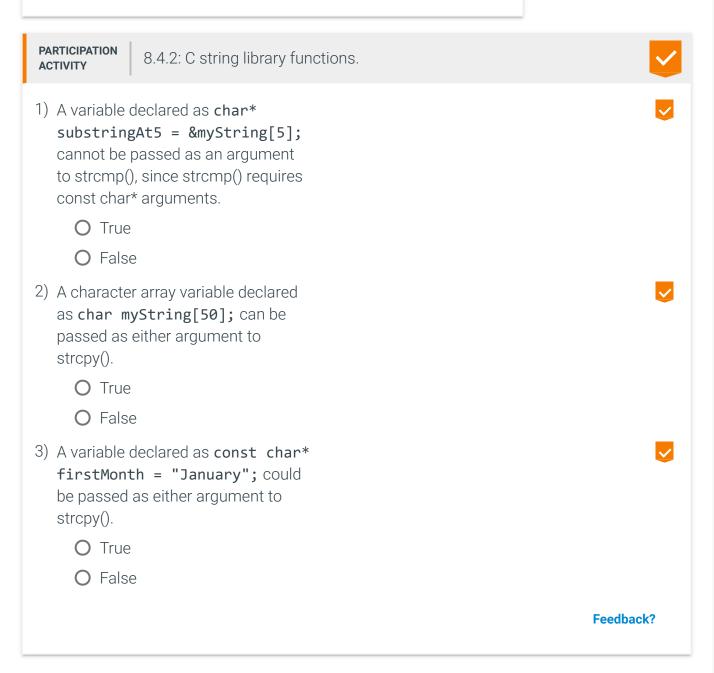
C string library functions

The C string library, introduced elsewhere, contains several functions for working with C strings. This section describes the use of char pointers in such functions. The C string library must first be included via: #include <cstring>.

Each string library function operates on one or more strings, each passed as a char* or const char* argument. Strings passed as char* can be modified by the function, whereas strings passed as const char* arguments cannot. Examples of such functions are strcmp() and strcpy(), introduced elsewhere.



Feedback?



C string search functions

strchr(), strrchr(), and strstr() are C string library functions that search strings for an occurrence of a character or substring. Each function's first parameter is a const char*, representing the string to search within.

The strchr() and strrchr() functions find a character within a string, and thus have a char as the second parameter. strchr() finds the first occurrence of the character within the string and strrchr() finds the last occurrence.

strstr() searches for a substring within another string, and thus has a const char* as the second parameter.

Table 8.4.1: Some C string search functions.

Given:

```
char orgName[100] = "The Dept. of Redundancy Dept.";
char newText[100];
char* subString = nullptr;
```

strchr()	strchr(sourceStr, searchChar) Returns a null pointer if searchChar does not exist in sourceStr. Else, returns pointer to first occurrence.	<pre>if (strchr(orgName, 'D') != nullptr) { // 'D' exists in orgName? subString = strchr(orgName, 'D'); // Points to first 'D' strcpy(newText, subString); // newText now "Dept. of Redundancy Dept." } if (strchr(orgName, 'Z') != nullptr) { // 'Z' exists in orgName? // Doesn't exist, branch not taken }</pre>
strrchr()	strrchr(sourceStr, searchChar) Returns a null pointer if searchChar does not exist in sourceStr. Else, returns pointer to LAST occurrence (searches in reverse, hence middle 'r' in name).	<pre>if (strrchr(orgName, 'D') != nullptr) { // 'D' exists in orgName? subString = strrchr(orgName, 'D'); // Points to last 'D' strcpy(newText, subString); // newText now "Dept." }</pre>
strstr()	strstr(str1, str2) Returns a null pointer if str2 does not exist in str1. Else, returns a char pointer pointing to first occurrence of string str2 within string str1.	<pre>subString = strstr(orgName, "Dept"); // Points to first 'D' if (subString != nullptr) { strcpy(newText, subString); // newText now "Dept. of Redundancy Dept." }</pre>

Feedback?

PARTICIPATION ACTIVITY

8.4.3: C string search functions.



		g	
af c	hat does fileExtension point to ter the following code? const char* fileName = Sample.file.name.txt"; const char* fileExtension = trrchr(fileName, '.');		✓
	O ".file.name.txt"		
	O ".txt"		
	O "Sample.file.name"		
no	on-null only if the fileName string ands with ".pdf".		<u>~</u>
	O True		
	O False		
fc tr	hat is true about fileName if the llowing expression evaluates to ue? trchr(fileName, '.') ==		✓
	trrchr(fileName, '.')		
	O The '.' character occurs exactly once in fileName.		
	O The '.' character occurs 0 or 1 time in fileName.		
	O The '.' character occurs 1 or more times in fileName.		
		1	Feedback?

Search and replace example

The following example carries out a simple censoring program, replacing an exclamation point by a period and "Boo" by "---" (assuming those items are somehow bad and should be censored). "Boo" is replaced using the strncpy() function, which is described elsewhere.

Note that only the first occurrence of "Boo" is replaced, as strstr() returns a pointer to the first occurrence. Additional code would be needed to delete all occurrences.

Figure 8.4.1: String searching example.

```
#include <iostream>
                                                           Enter a line of text: Hello!
#include <cstring>
                                                          Censored: Hello.
using namespace std;
                                                          Enter a line of text: Boo hoo to
int main() {
  Censored: --- hoo to you.
string
                                                          Enter a line of text: Booo!
  char* stringPos = nullptr;
                                      // Index into
                                                          Boooo!!!!
string
                                                          Censored: ---o. Boooo!!!!
  // Prompt user for input
  cout << "Enter a line of text: ";</pre>
  cin.getline(userInput, MAX_USER_INPUT);
  // Locate exclamation point, replace with period
  stringPos = strchr(userInput, '!');
  if (stringPos != nullptr) {
      *stringPos = '.';
  // Locate "Boo" replace with "---"
  stringPos = strstr(userInput, "Boo");
  if (stringPos != nullptr) {
     strncpy(stringPos, "---", 3);
  // Output modified string
  cout << "Censored: " << userInput << endl;</pre>
  return 0;
}
```

Feedback?

PARTICIPATION ACTIVITY

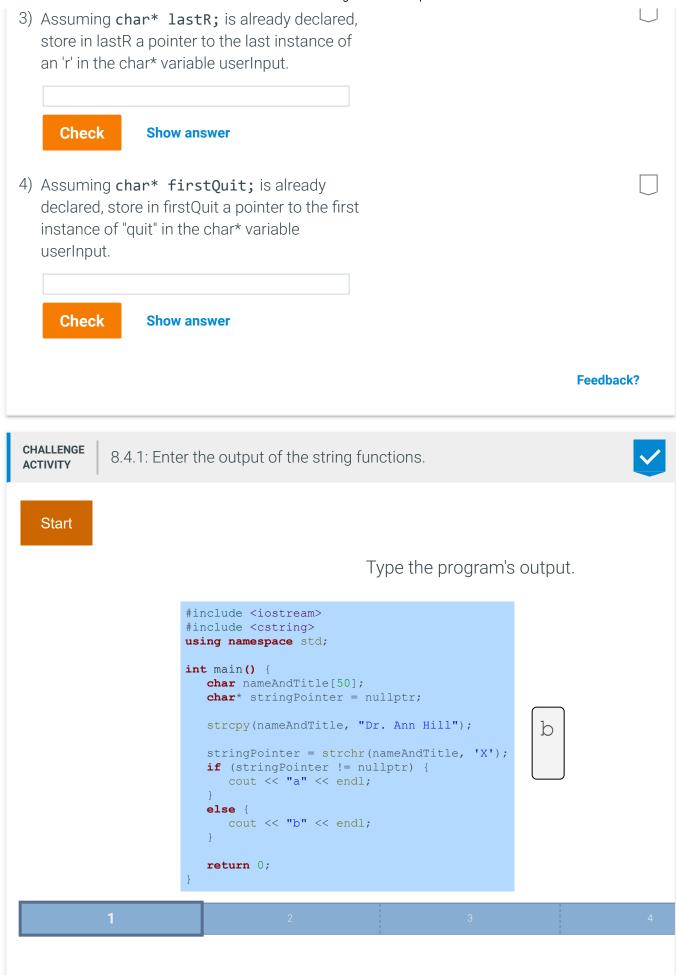
8.4.4: Modifying and searching strings.

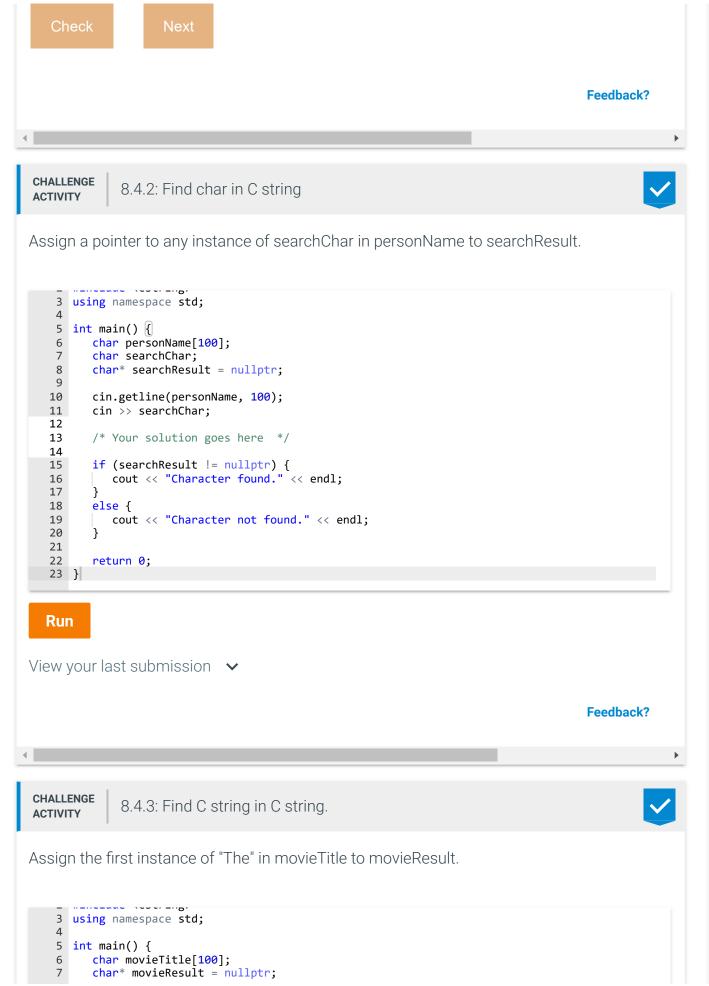
1) Declare a char* variable named charPtr.

Check Show answer

2) Assuming char* firstR; is already declared, store in firstR a pointer to the first instance of an 'r' in the char* variable userInput.

Check Show answer





```
8
        cin.getline(movieTitle, 100);
   9
  10
        /* Your solution goes here */
  11
        movieResult = strstr(movieTitle, "The");
  12
  13
        cout << "Movie title contains The? ";</pre>
  14
        if (movieResult != nullptr) {
  15
           cout << "Yes." << endl;</pre>
  16
  17
        else {
  18
           cout << "No." << endl;</pre>
  19
  20
  21
        return 0;
  22
  23 }
           All tests passed
  Run

✓ Testing: The Lion King

           Your output
                           Movie title contains The? Yes.

✓ Testing: Airplane II: The Sequel

           Your output
                           Movie title contains The? Yes.

✓ Testing: Frozen

           Your output
                           Movie title contains The? No.
                                                                                Feedback?
```