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.

PARTICIPATION ACTIVITY 8.4.1: strcmp() and strcpy() string functions.

Start 2x speed

Function signatures:

```
int strcmp(const char* str1, const char* str2);

char* strcpy(char* destination, const char* source);
```

Usage example:

```
int cmp1, cmp2;
char string1[10] = "abcxyz";
char string2[10] = "xyz";
char newText[10];
char* subStr = nullptr;

cmp1 = strcmp(string1, string2);
cout << "strcmp of \"" << string1 << "\" and \"";
cout << string2 << "\" returned " << cmp1 << endl;

subStr = &string1[3];

cmp2 = strcmp(subStr, "xyz");
cout << "strcmp of \"" << subStr;
cout << "\" and \"xyz\" returned ";
cout << cmp2 << endl;

strcpy(newText, subStr);
cout << "newText is now \"" << newText << "\"";
cout << endl;
```

Console:

```
strcmp of "abcxyz" and "xyz" returned -23
strcmp of "xyz" and "xyz" returned 0
newText is now "xyz"
```

Feedback?

PARTICIPATION ACTIVITY 8.4.2: C string library functions.

1) A variable declared as `char* substringAt5 = &myString[5];` cannot be passed as an argument to strcmp(), since strcmp() requires const char* arguments.
   - ○ True
   - ○ False

2) A character array variable declared as `char myString[50];` can be passed as either argument to strcpy().
   - ○ True
   - ○ False

3) A variable declared as `const char* firstMonth = "January";` could be passed as either argument to strcpy().
   - ○ True
   - ○ False

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.
		`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`
		`}`

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).
		`if (strrchr(orgName, 'D') != nullptr) {   // 'D' exists in orgName?`
		`   subString = strrchr(orgName, 'D');     // Points to last 'D'`
		`   strcpy(newText, subString);`
		`// newText now "Dept."`
		`}`

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.
		`subString = strstr(orgName, "Dept"); // Points to first 'D'`
		`if (subString != nullptr) {`
		`   strcpy(newText, subString);`
		`// newText now "Dept. of Redundancy Dept."`
		`}`

Feedback?

PARTICIPATION ACTIVITY 8.4.3: C string search functions.

1) What does fileExtension point to after the following code?

```
const char* fileName =
"Sample.file.name.txt";
const char* fileExtension =
strrchr(fileName, '.');
```

- ".file.name.txt"
- ".txt"
- "Sample.file.name"

2) `strstr(fileName, ".pdf")` is non-null only if the fileName string ends with ".pdf".

- True
- False

3) What is true about fileName if the following expression evaluates to true?

```
strchr(fileName, '.') ==
strrchr(fileName, '.')
```

- The '.' character occurs exactly once in fileName.
- The '.' character occurs 0 or 1 time in fileName.
- The '.' character occurs 1 or more times in fileName.

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>
#include <cstring>
using namespace std;

int main() {
   const int MAX_USER_INPUT = 100;        // Max input size
   char userInput[MAX_USER_INPUT];        // User defined
string
   char* stringPos = nullptr;             // Index into
string

   // Prompt user for input
   cout << "Enter a line of text: ";
   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;

   return 0;
}
```

```
Enter a line of text: Hello!
Censored: Hello.
...
Enter a line of text: Boo hoo to
you!
Censored: --- hoo to you.
...
Enter a line of text: Booo!
Boooo!!!!
Censored: ---o. Boooo!!!!
```

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

3) Assuming `char* lastR;` is already declared, store in lastR a pointer to the last instance of an 'r' in the char* variable userInput.

Check Show answer

4) Assuming `char* firstQuit;` is already declared, store in firstQuit a pointer to the first instance of "quit" in the char* variable userInput.

Check Show answer

Feedback?

CHALLENGE ACTIVITY 8.4.1: Enter the output of the string functions.

Start

Type the program's output.

```
#include <iostream>
#include <cstring>
using namespace std;

int main() {
   char nameAndTitle[50];
   char* stringPointer = nullptr;

   strcpy(nameAndTitle, "Dr. Ann Hill");

   stringPointer = strchr(nameAndTitle, 'X');
   if (stringPointer != nullptr) {
      cout << "a" << endl;
   }
   else {
      cout << "b" << endl;
   }

   return 0;
}
```

b

1 2 3 4

Check Next

Feedback?

CHALLENGE ACTIVITY 8.4.2: Find char in C string

Assign a pointer to any instance of searchChar in personName to searchResult.

```
3  using namespace std;
4
5  int main() {
6     char personName[100];
7     char searchChar;
8     char* searchResult = nullptr;
9
10    cin.getline(personName, 100);
11    cin >> searchChar;
12
13    /* Your solution goes here  */
14
15    if (searchResult != nullptr) {
16       cout << "Character found." << endl;
17    }
18    else {
19       cout << "Character not found." << endl;
20    }
21
22    return 0;
23 }
```

Run

View your last submission

Feedback?

CHALLENGE ACTIVITY 8.4.3: Find C string in C string.

Assign the first instance of "The" in movieTitle to movieResult.

```
3  using namespace std;
4
5  int main() {
6     char movieTitle[100];
7     char* movieResult = nullptr;
```

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

Run ✓ All tests passed

✓ 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?