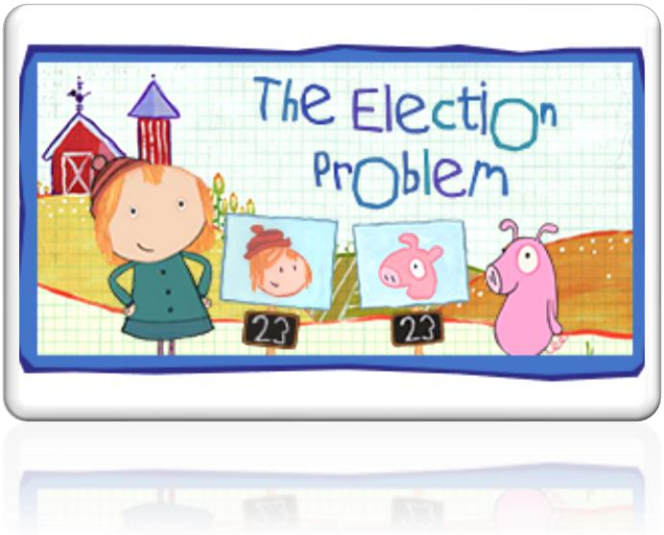
Practice Problems for PE09

For PE09, you'll complete one problem with pointers and C-Style strings. Here are some additional programming problems you can use for practice.



1 THE MYSTRCHR FUNCTION

Write the function **`myStrChr()`**. The function has two parameters: a **`const char * s`** pointing to the first character in a C-style string, and a **`char c`.** Return a pointer to the first appearance of **`c`** appearing inside **`s`** and **`nullptr`** if **`c`** does not appear inside **`s`**.

```
const char * s = "Aardvark";
myStrChr(s, 'a')->&s[1]
myStrChr(s, 'V')->nullptr
```

2 THE FINDFIRST FUNCTION

Write the function **`findFirst()`**. The function has two parameters: a **`const char * s1`** pointing to the first character in a C-style string, and a **`const char * s2`.** Return a pointer to the first appearance of **`s2`** appearing inside **`s1`** and **`nullptr`** (**`0`**) if **`s2`** does not appear inside **`s`**.

```
const char * s = "Aardvark";
findFirst(s, "dv")->&s[3]
findFirst(s, "arki")->nullptr
```

3 THE FINDLAST FUNCTION

Write the function **findLast()**. The function has two parameters: a **const char * s1** pointing to the first character in a C-style string, and a **const char * s2.** Return a pointer to the last appearance of **s2** appearing inside **s1** and **nullptr** (**0**) if **s2** does not appear inside **s**.

```
const char * s = "Aardvark";
findLast(s, "ar")->&s[5]  // "ark"
findLast(s, "K")->nullptr
```

4 THE FINDANY FUNCTION

Write the function **findAny()**.The function has two parameters: a **const char * s** pointing to the first character in a C-style string, and a **const char** * letters. Return the index of the first character in s that matches any letter inside letters. Return -1 if no letters match.

```
const char * s = "This is a test.";
findAny(s, "aeiou")->2      // any vowel
findAny(s, " \t\n\r")->4    // any whitespace
findAny(s, "Bob")->-1       // None found
```

5 THE NUMCHR FUNCTION

Write the function **numChr()**.The function has two parameters: a **const char * s** pointing to the first character in a C-style string, and a **char c**. Return the number of times that **c** appears inside **s**.

```
const char * s = "abracadabra";
numChar(s, 'a')->7
numChar(s, 'A')->0
```

6 THE NUMSTR FUNCTION

Write the function **numStr()**.The function has two parameters: a **const char * s1** pointing to the first character in a C-style string, and a **const char * s2**. Return the number of times that **s2** appears inside **s1**.

```
const char * s = "abracadabra";
numStr(s, "a")->7
numStr(s, "bara")->2
numStr(s, "cad")->1
```

7 THE RFINDANY FUNCTION

Write the function **rFindAny()**.The function has two parameters: a **const char * s** pointing to the first character in a C-style string, and a **const char** * letters. Return the index of the last character in s that matches any letter inside letters. Return -1 if no letters match.

```
const char * s = "This is a test.";
rFindAny(s, "aeiou")->11      // last vowel
rFindAny(s, " \t\n\r")->9    // last whitespace
rFindAny(s, "Bob")->-1       // None found
```

8 THE MYSTRRCHR FUNCTION

Write the function **myStrrChr()**. The function has two parameters: a **const char * s** pointing to the first character in a C-style string, and a **char**, **c**. Return a pointer to the last appearance of c appearing inside **s** and **nullptr** (**0**) if **c** does not appear inside **s**.

```
const char * s = "Aardvark";
myStrrChr(s, 'a')->&s[5]
myStrrChr(s, 'V')->nullptr
```

9 THE MYSTRNCPY FUNCTION

Write the function **myStrNCpy()**. The function has three parameters: a **char * dest**, a **const char * src**, and an int **max**, which represents the maximum size of the destination buffer. Copy the characters in **src** to **dest**, but don't copy more than **max-1** characters. Make sure you correctly terminate the copied string. The function returns a pointer to the first character in **dest**.

10 THE MYSTRNCAT FUNCTION

Write the function **myStrNCat()**. The function has three parameters: a **char * s1**, a **const char * s2**, and an **size_t max**, which represents the maximum size of the **s1** buffer. Append the characters in **s2** to the end of **s1**. Make sure you have only one **'\0'** at the end of the combined characters. Don't go beyond the end of the buffer you are asked to copy to. The function returns a pointer to the first character in **s1**.

11 THE REPLACE PROBLEM

Write the function **replace()**. The function has three parameters: a **char * s**, a **char c1** and a **char** c2. Replace all instances of **c1** with **c2**. Return a pointer to the first character in **s**.

12 The cSubstr Function

Write the function `cSubstr()`. Given 2 pointers, begin and end, which point to the elements in a C-String, copy the characters between the pointers to the destination buffer. However make sure you don't exceed the capacity of the buffer. Make sure you null terminate the destination. Return a pointer to the destination buffer. Use ONLY pointers, and addresses, not integer counters.

13 The cStrFind Function

Write the function `cStrFind()`. The function has two parameters: a `const char * str`, and a `const char * subs`. Find the first location where `subs` is found in `str` (as an `int`). Return `-1` if `subs` is not located in `str`. You may use pointer or array notation, but you cannot use any functions or classes from the standard library.

14 The cStrRFind Function

Write the function `cStrRFind()`. The function has two parameters: a `const char * str`, and a `const char * subs`. Find the last location where `subs` is found in `str` (as an `int`). Return `-1` if `subs` is not located in `str`. You may use pointer or array notation, but you cannot use any functions or classes from the standard library.