Homework

10

Two Recursion Problems

Using the problem-solving technique called recursion, you can solve large problems by reducing them to smaller problems of the same form. To solve a recursive problem, you use the following **template**, called the **recursive paradigm**, taking a **recursive leap of faith**. Apply this technique by:

1. Identifying the simple case(s) for which the answer is easily determined.
2. Identifying a simpler problem **of the same form**.

Recursive Find

Use recursion to implement a function: **`find(s, t)`** that tests whether the **`string t`** is contained in a **`string s`**. For instance, calling the function like this:

```
1 | find("Mississippi", "sip");
```

A recursive find.

returns **`true`**, since **`"sip"`** is contained in **`"Mississippi"`**.

You **must use recursion to complete this**, not loops; you obviously may not use the **`string find()`** member function either. Here's the basic strategy:

- If the text starts with the **`string`** you want to match, then you are done.
- If the **`string s`** is smaller than the **`string t`**, you are also done.
- If not, check the sentence without the first character.

String Cleaning

The function **`stringClean(str)`** returns a new **`string`** where adjacent duplicate characters have been reduced to a single character. So **`"yyzzza"`** yields **`"yza"`**. Here are some examples:

```
stringClean("yyzzza") -> "yza"
stringClean("abbbcdd") -> "abcd"
stringClean("Hello") -> "Helo"
```

A string cleaning.

Again, you need to use recursion, not loops to complete your function. If you get stuck, well, you know what to do.