Strings in C++ *Recursive Functions

- · More exploration on a theme
- · Strings in C++
 - · Representing and manipulating text
- · Recursion on Strings
 - · Featuring Cute animals

Thinking Recursively

```
if (The problem is very simple) {
                                                These simple cases
                                                  are called base
  Directly solve the problem.
                                                      cases.
  Return the solution.
} else {
  Split the problem into one or more
  smaller problems with the same
  structure as the original.
  Solve each of those smaller problems.
  Combine the results to get the overall
  solution.
  Return the overall solution.
                                                  These are the
}
                                                 recursive cases.
```

Digit Roots

The digital root is the number you get by repeatedly summing the digits of a number until you're down to a single digit.

C++ Strings

To use strings, you need to add the line #include < string> to the top of your program to import the strings library.

```
/* C++ Version
string s = "Elena Kagan";
s += ", joined " + to_string(2010);
char first = s[0];
char last = s[s.length() - 1]
if (s.find("e") != string::npos) {
   string first = s.substr(0, 5);
   string last = s.substr(7);
}
if (s == "Sonia Sotomayor") {
   cout << "John Roberts" << endl;
}</pre>
```

```
Python Version

s = "Elena Kagan"

s += ", joined " + str(2010)

first = s[0]

la

if C++ strings must be declared using double quotes rather than

if single quotes.
```

You can select an individual character out of a string by

using square brackets. Indices start at zero.

C++ has different types for individual characters (char) and for strings of zero or more characters (string). Check Chapter 1.5 of the textbook for details.

You can use + and += to append to a string. You can only append other strings or characters. Use the to_string function to convert data to strings.

```
/* C++ Version */
string s = "Elena Kagan";
s += ", joined " + to_string(2010);
char first = s[0];
char last = s[s.length() - 1]

if (s.find("e") != string::npos) {
   string first = s.substr(0, 5);
   string last = s.substr(7);
}
if (s == "Sonia Sotomayor") {
   cout << "John Roberts" << endl;
}</pre>
```

The find member function / returns the index of the given pattern if it exists, and the verbosely-named constant string::npos otherwise. This pattern kinda sorta is like the "in" keyword from Python.

C++ doesn't support negative array indices the way that Python does. You can pick the last character of the string by getting its length and subtracting one.

```
let last = s.substring(7);
}
if (s === "Sonia Sotomayor") {
  console.log("John Roberts");
}
```

```
Python Version
           C++ Version
                                           s = "Elena Kagan"
string s = "Elena Kagan";
s += ", joined " + to_string(2010);
                                           s += ", joined " + str(2010)
char first = s[0];
                                           first = s[0]
char last = s[s.length() - 1]
                                           last = s[-1]
if (s.find("e") != string::npos) {
                                           if 'e' in s:
  string first = s.substr(0, 5);
                                              first = s[0:5]
  string last = s.substr(7);
                                              last = s[7:]
if (s == "Sonia Sotomayor") {
                                           if s == "Sonia Sotomayor":
  cout << "John Roberts" << endl;
                                              print("John Roberts")
       JavaScript Version
                                                 JavaScript Version
                                             You can get substrings by
  You can compare strings for
                                             using the substr member
   equality using ==. If you're
                                             function. If you give two
  coming from Python, great!
                                          parameters, the first is a start
 This will feel normal. If you're
                                            index, and the second is a
  coming from Java, hopefully
                                            length, not an end index.
      this will be a welcome
                                             console.log("John Roberts");
             relief.
```

Recap

- · Recursion works by identifying
- · One or more base cases, Simple cases that can be solved directly, and
- · One or more recursive cases, where a larger problem is problem is turned into a smaller one.
 - · C++ Strings have some endearing quirks compared to other