Any substring of a given string is specified if we know the index in the string of the substring's first character and the index of its last character. For example, the substring "term" of the string "determine" starts at the character with index 2 and ends at the character with index 5. To extract a particular substring from a string, Java provides the substring method. This is invoked by sending to the string a message of the form:

substring( m, n )

where m is the index in the string of the substring's first character and n is one more than the index in the string of the substring's last character. For example, the expression

1. Consider the string "abcde". What should the arguments m and n be so that "abcde".substring( m, n ) evaluates to the substring

1. "bcd"
2. "de"
3. "abcd"
4. "bcde"
5. "c"

2. What is the value of "abcde".substring( 2, 2 )?

1. The required arguments are:
   1. 1 and 4
   2. 3 and 5
   3. 0 and 4
   4. 1 and 5
   5. 2 and 3
2. The value is "", that is, the empty string, the string that has nothing between the starting and ending double-quote marks.

Each such tail of a string can be specified by using the length of the string as the second argument to the substring method. However, as a convenience to programmers, Java (and other similar programming languages) permits a shorthand instruction, obtained by simply omitting the second argument altogether. In other words, if we supply the substring method with only one argument, then it behaves as if a second argument has been supplied equal to the length of the string. Thus, for example, "abcde".substring( 2 ) evaluates to "cde". Familiarize yourself with how this works by experimenting in the code fragment toward the top of this page.

The substring method is useful if you know exactly where the substring you want is located within the string that includes it. But suppose you are not sure whether or not the substring you want is part of a particular string. Java provides the indexOf method to help us search within a string in hopes of finding a given substring. (The capitalization of the "O" in this method's name is important.)

The method takes a single argument, namely the substring that we hope to find. In the following code fragment, we search the string "abcde" for the substring "cde". This substring appears in "abcde" starting at the character with index 2, so the value of the expression "abcde".indexOf( "cde" ) is 2. Verify this by running the code.

If the substring is not found, indexOf returns -1. The value returned by indexOf is always an int.