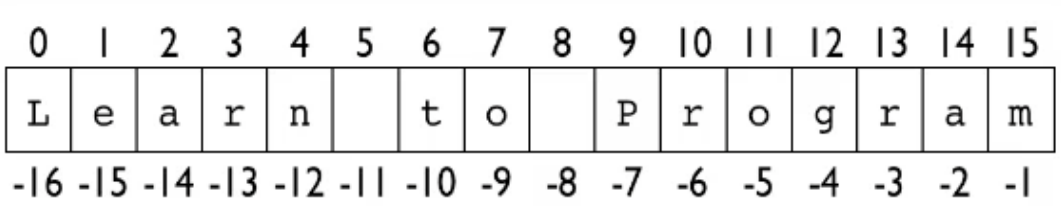
`str`: indexing and slicing

Indexing

An index is a position within the string. Positive indices count from the left-hand side with the first character at index 0, the second at index 1, and so on. Negative indices count from the right-hand side with the last character at index -1, the second last at index -2, and so on. For the string `"Learn to Program"`, the indices are:



The first character of the string is at index 0 and can be accessed using this bracket notation:

```
>>> s[0]
'L'
>>> s[1]
'e'
```

Negative indices are used to count from the end (from the right-hand side):

```
>>> s[-1]
'm'
>>> s[-2]
'a'
```

Slicing

We can extract more than one character using slicing. A slice is a substring from the start index up to but not including the end index. For example:

```
>>> s[0:5]
'Learn'
>>> s[6:8]
'to'
>>> s[9:16]
'Program'
```

More generally, the end of the string can be represented using its length:

```
>>> s[9:len(s)]
'Program'
```

The end index may be omitted entirely and the default is `len(s)`:

```
>>> s[9:]
'Program'
```

Similarly, if the start index is omitted, the slice starts from index 0:

```
>>> s[:]
'Learn to Program'
>>> s[:8]
'Learn to'
```

Negative indices can be used for slicing too. The following three expressions are equivalent:

```
>>> s[1:8]
'earn to'
>>> s[1:-8]
'earn to'
>>> s[-15:-8]
'earn to'
```

Modifying Strings

The slicing and indexing operations do not modify the string that they act on, so the string that `s` refers to is unchanged by the operations above. In fact, we cannot change a string. Operations like the following result in errors:

```
>>> s[6] = 'd'
Traceback (most recent call last):
        File <"pyshell#19", line 1, in <module>
                s[6] = 'd'
TypeError: 'str' object does not support item assignment
```

Imagine that we want to change string `s` to refer to `'Learned to Program'`. The following expression evaluates to that `'Learned to Program'`: `s[:5] + 'ed' + s[5:]`

Variable `s` gets the new string: `s = s[:5] + 'ed' + s[5:]`

Notice that the string that `s` originally referred to was not modified: strings cannot be modified. Instead a new string was created and `s` was changed to point to that string.

Jennifer Campbell • Paul Gries
University of Toronto
