S: Strings

Gareth McCaughan Revision 1.8, May 14, 2001

Credits

For the LaTeX source of this sheet, and for more information on LiveWires and on this course, see the LiveWires web site at http://www.livewires.org.uk/python/

Introduction

A "string" is a piece of text. Strings are made up of "characters"; a character is a single letter, digit, symbol or whatever. Python provides lots of things for doing with strings; this sheet tells you about some of them.

Entering strings

You can put a string into your program by putting it inside quotation marks. Single quotes and double quotes are both fine, but you must use the same sort of quotation mark at the start and at the end!

In these worksheets, I've usually used single quotes. If you prefer double quotes, that's fine. If you want a string that contains a single-quote character (which also serves as the apostrophe), you should surround it with double quotes: `"I'm bored"` rather than `'I'm bored'` (which won't work at all: can you see why?)

Strings as sequences

There are some things that work on lists as well as on strings. They work because both lists and strings are "sequences". Here's a very brief summary. (You might want to compare with with Sheet A (*Lists*).)

```
>>> thing = 'I am the walrus'                  A string to work with
>>> gloop = "Another string"                   And another one
>>> thing+gloop                                String concatenation . . .
'I am the walrusAnother string'                    works as you might guess
>>> 2*thing                                    String replication . . .
'I am the walrusI am the walrus'                   also does what you'd think
>>> thing[0]                                   We start counting at 0
'I'                                                so we get the first character
>>> thing[1:5                                  Characters 1 (inclusive) to 5 (exclusive)
' am '                                             so 4 characters in all
>>> len(thing)                                 How many characters?
15                                             That many!
```

The ‘string’ module

If you say `import string` then after that you can use a whole lot of functions specially for working on strings. Here are a few useful ones:

```
>>> import string                                  Make this stuff available
>>> string.capitalize('walrus')
'Walrus'
>>> string.lower('WalRUs')
'walrus'
>>> string.upper('WalRUs')
'WALRUS'
>>> string.split('I am the walrus')
['I', 'am', 'the', 'walrus']
>>> string.replace('I am a twit', 'a', 'the')
'I them the twit'
```

(The names, with those dots in, are a bit strange. You might like to look at Sheet M (*Modules*).)

There are a *lot* more things you can do with strings if you want to. If there’s something you can’t see how to do, ask a leader.

Other things

It’s possible to convert just about any object into a string that represents it. So, for instance, 1234 would become the string `'1234'`. The function that does this is called `repr` (short for “representation”). So, whatever `x` is, `repr(x)` is a string describing it.

If `repr(x)` is too clumsy or too much effort to type, you can instead say `` `x` `` – notice that those are a different kind of quotation mark: `` ` `` isn’t the same as `'` !