Functions and Error Handling

Functions are groups of code that have a name, and can be called using parentheses.

Two ways of creating functions: the def statement, useful for any type of function, and the lambda statement, useful for creating short anonymous functions.

Keyword arguments that are specified by name. Keyword argument for the print () function is sep, which tells what character or characters should be used to separate multiple items. When non-keyword arguments are used together with keyword arguments, the keyword arguments must come at the end.

When how many arguments the user will pass is unknown, the special form \*args and \*\*kwargs to catch all arguments that are passed. Short for "arguments" and "keyword arguments". The operative difference is that a single \* before a variable means "expand this as a sequence", while a double \*\* before a variable means "expand this as a dictionary".

**Errors and Exceptions**

* *Syntax errors:* Errors where the code is not valid Python (generally easy to fix)
* *Runtime errors:* Errors where syntactically valid code fails to execute, perhaps due to invalid user input (sometimes easy to fix)
* *Semantic errors:* Errors in logic: code executes without a problem, but the result is not what you expect (often very difficult to track-down and fix)

Python handles runtime errors via its *exception handling* framework.

The main tool Python gives you for handling runtime exceptions is the try...except clause. The way to raise one’s own exceptions is with the raise statement.