software carpentry

Programming with Python

Reference

Analyzing Patient Data

- Import a library into a program using `import libraryname`.
- Use the `numpy` library to work with arrays in Python.
- Use `variable = value` to assign a value to a variable in order to record it in memory.
- Variables are created on demand whenever a value is assigned to them.
- Use `print(something)` to display the value of `something`.
- The expression `array.shape` gives the shape of an array.
- Use `array[x, y]` to select a single element from an array.
- Array indices start at 0, not 1.
- Use `low:high` to specify a slice that includes the indices from `low` to `high-1`.
- All the indexing and slicing that works on arrays also works on strings.
- Use `# some kind of explanation` to add comments to programs.
- Use `array.mean()`, `array.max()`, and `array.min()` to calculate simple statistics.
- Use `array.mean(axis=0)` or `array.mean(axis=1)` to calculate statistics across the specified axis.
- Use the `pyplot` library from `matplotlib` for creating simple visualizations.

Repeating Actions with Loops

- Use `for variable in collection` to process the elements of a collection one at a time.
- The body of a for loop must be indented.
- Use `len(thing)` to determine the length of something that contains other values.

Storing Multiple Values in Lists

- `[value1, value2, value3, ...]` creates a list.
- Lists are indexed and sliced in the same way as strings and arrays.
- Lists are mutable (i.e., their values can be changed in place).
- Strings are immutable (i.e., the characters in them cannot be changed).

Analyzing Data from Multiple Files

- Use `glob.glob(pattern)` to create a list of files whose names match a pattern.
- Use `*` in a pattern to match zero or more characters, and `?` to match any single character.

Making Choices

- Use the `ImageGrid` class from the `ipythonblocks` library to create simple “images” made of colored blocks.
- Specify colors use (red, green, blue) triples, each component of which is an integer in the range 0..255.
- Use `if condition` to start a conditional statement, `elif condition` to provide additional tests, and `else` to provide a default.
- The bodies of the branches of conditional statements must be indented.
- Use `==` to test for equality.
- `X and Y` is only true if both X and Y are true.
- `X or Y` is true if either X or Y, or both, are true.
- Zero, the empty string, and the empty list are considered false; all other numbers, strings, and lists are considered true.
- Nest loops to operate on multi-dimensional data.
- Put code whose parameters change frequently in a function, then call it with different parameter values to customize its behavior.

Creating Functions

- Define a function using `def name(...params...)`.
- The body of a function must be indented.
- Call a function using `name(...values...)`.
- Numbers are stored as integers or floating-point numbers.
- Integer division produces the whole part of the answer (not the fractional part).
- Each time a function is called, a new stack frame is created on the **call stack** to hold its parameters and local variables.
- Python looks for variables in the current stack frame before looking for them at the top level.
- Use `help(thing)` to view help for something.
- Put docstrings in functions to provide help for that function.
- Specify default values for parameters when defining a function using `name=value` in the parameter list.
- Parameters can be passed by matching based on name, by position, or by omitting them (in which case the default value is used).

Errors and Exceptions

- Tracebacks can look intimidating, but they give us a lot of useful information about what went wrong in our program, including where the error occurred and what type of error it was.
- An error having to do with the “grammar” or syntax of the program is called a `SyntaxError`. If the issue has to do with how the code is indented, then it will be called an `IndentationError`.
- A `NameError` will occur if you use a variable that has not been defined (either because you meant to use quotes around a string, you forgot to define the variable, or you just made a typo).
- Containers like lists and dictionaries will generate errors if you try to access items in them that do not

exist. For lists, this type of error is called an `IndexError`; for dictionaries, it is called a `KeyError`.
- Trying to read a file that does not exist will give you an `IOError`. Trying to read a file that is open for writing, or writing to a file that is open for reading, will also give you an `IOError`.

Defensive Programming

- Program defensively, i.e., assume that errors are going to arise, and write code to detect them when they do.
- Put assertions in programs to check their state as they run, and to help readers understand how those programs are supposed to work.
- Use preconditions to check that the inputs to a function are safe to use.
- Use postconditions to check that the output from a function is safe to use.
- Write tests before writing code in order to help determine exactly what that code is supposed to do.

Debugging

- Know what code is supposed to do *before* trying to debug it.
- Make it fail every time.
- Make it fail fast.
- Change one thing at a time, and for a reason.
- Keep track of what you've done.
- Be humble.

Command-Line Programs

- The `sys` library connects a Python program to the system it is running on.
- The list `sys.argv` contains the command-line arguments that a program was run with.
- Avoid silent failures.
- The "file" `sys.stdin` connects to a program's standard input.
- The "file" `sys.stdout` connects to a program's standard output.

Glossary

additive color model
A way to represent colors as the sum of contributions from primary colors such as red, green, and blue.

argument
A value given to a function or program when it runs. The term is often used interchangeably (and inconsistently) with parameter.

assertion
An expression which is supposed to be true at a particular point in a program. Programmers typically put assertions in their code to check for errors; if the assertion fails (i.e., if the expression evaluates as false), the program halts and produces an error message. See also: invariant, precondition, postcondition.

assign
To give a value a name by associating a variable with it.

body
(of a function): the statements that are executed when a function runs.

call stack
A data structure inside a running program that keeps track of active function calls.

case-insensitive
Treating text as if upper and lower case characters of the same letter were the same. See also: case-sensitive.

case-sensitive
Treating text as if upper and lower case characters of the same letter are different. See also: case-insensitive.

comment
A remark in a program that is intended to help human readers understand what is going on, but is ignored by the computer. Comments in Python, R, and the Unix shell start with a `#` character and run to the end of the line; comments in SQL start with `--`, and other languages have other conventions.

compose
To apply one function to the result of another, such as `f(g(x))`.

conditional statement
A statement in a program that might or might not be executed depending on whether a test is true or false.

comma-separated values
(CSV) A common textual representation for tables in which the values in each row are separated by commas.

default value
A value to use for a parameter if nothing is specified explicitly.

defensive programming
The practice of writing programs that check their own operation to catch errors as early as possible.

delimiter
A character or characters used to separate individual values, such as the commas between columns in a CSV file.

docstring
Short for “documentation string”, this refers to textual documentation embedded in Python programs. Unlike comments, docstrings are preserved in the running program and can be examined in interactive sessions.

documentation
Human-language text written to explain what software does, how it works, or how to use it.

dotted notation
A two-part notation used in many programming languages in which `thing.component` refers to the `component` belonging to `thing`.

empty string
A character string containing no characters, often thought of as the “zero” of text.

encapsulation
The practice of hiding something’s implementation details so that the rest of a program can worry about *what* it does rather than *how* it does it.

floating-point number
A number containing a fractional part and an exponent. See also: integer.

for loop
A loop that is executed once for each value in some kind of set, list, or range. See also: while loop.

function call
A use of a function in another piece of software.

immutable
Unchangeable. The value of immutable data cannot be altered after it has been created. See also: mutable.

import
To load a library into a program.

in-place operators
An operator such as `+=` that provides a shorthand notation for the common case in which the variable being assigned to is also an operand on the right hand side of the assignment. For example, the statement `x += 3` means the same thing as `x = x + 3`.

index
A subscript that specifies the location of a single value in a collection, such as a single pixel in an image.

inner loop
A loop that is inside another loop. See also: outer loop.

integer
A whole number, such as -12343. See also: floating-point number.

invariant
An expression whose value doesn't change during the execution of a program, typically used in an assertion. See also: precondition, postcondition.

library
A family of code units (functions, classes, variables) that implement a set of related tasks.

loop variable
The variable that keeps track of the progress of the loop.

member
A variable contained within an object.

method
A function which is tied to a particular object. Each of an object's methods typically implements one of the things it can do, or one of the questions it can answer.

object
A collection of conceptually related variables (members) and functions using those variables (methods).

outer loop
A loop that contains another loop. See also: inner loop.

parameter
A variable named in the function's declaration that is used to hold a value passed into the call. The term is often used interchangeably (and inconsistently) with argument.

pipe
A connection from the output of one program to the input of another. When two or more programs are connected in this way, they are called a "pipeline".

postcondition
A condition that a function (or other block of code) guarantees is true once it has finished running. Postconditions are often represented using assertions.

precondition
A condition that must be true in order for a function (or other block of code) to run correctly.

regression
To re-introduce a bug that was once fixed.

return statement
A statement that causes a function to stop executing and return a value to its caller immediately.

RGB
An additive model that represents colors as combinations of red, green, and blue. Each color's value is typically in the range 0..255 (i.e., a one-byte integer).

sequence
A collection of information that is presented in a specific order. For example, in Python, a string is a sequence of characters, while a list is a sequence of any variable.

shape
An array's dimensions, represented as a vector. For example, a 5×3 array's shape is `(5,3)`.

silent failure
Failing without producing any warning messages. Silent failures are hard to detect and debug.

slice
A regular subsequence of a larger sequence, such as the first five elements or every second element.

stack frame
A data structure that provides storage for a function's local variables. Each time a function is called, a new stack frame is created and put on the top of the call stack. When the function returns, the stack frame is discarded.

standard input
A process's default input stream. In interactive command-line applications, it is typically connected to the keyboard; in a pipe, it receives data from the standard output of the preceding process.

standard output
A process's default output stream. In interactive command-line applications, data sent to standard output is displayed on the screen; in a pipe, it is passed to the standard input of the next process.

string
Short for "character string", a sequence of zero or more characters.

syntax error
A programming error that occurs when statements are in an order or contain characters not expected by the programming language.

test oracle
A program, device, data set, or human being against which the results of a test can be compared.

test-driven development
The practice of writing unit tests *before* writing the code they test.

traceback
The sequence of function calls that led to an error.

tuple
An immutable sequence of values.

type
The classification of something in a program (for example, the contents of a variable) as a kind of number (e.g. floating-point, integer), string, or something else.

type of error
Indicates the nature of an error in a program. For example, in Python, an `IOError` to problems with file input/output. See also: syntax error.

while loop
A loop that keeps executing as long as some condition is true. See also: for loop.