Intermediate Concepts

Built-In Functions

Printing

The print() function prints to stdout (the screen, or the command line)

- print(“message”) → prints ”message” to the screen
- print(mystring) → prints the contents of mystring to the screen.
- print(myint) → converts myint to a string (if possible), and prints it to the screen.
- print(mystring1 + mystring2) → concats mystring1 with mystring2, and prints to the screen
- Print(“there are {} items”.format(myint)) → inserts myint into the placeholder {} and prints to the screen

Taking User Inputs

The input() function prints a prompt to stdout, then waits for a user input.

- input(“enter here:”) → prints ”enter here:” to the screen, then waits for an input.
- A user input ends when the user enters the [ENTER] key
- All inputs are returned as a string. If another type is needed, they must be converted.
- No entry (just hitting [ENTER]) returns an empty string.
- The user input can be assigned to a variable, for example:
  - Userinput = input(“enter something here: “)

Checking Variables

Introducing a few useful/common built in functions:

- len() <- returns the length of that object
  - len('mystring') → 8
- type() <- returns the type of that object
  - type(2.0) → float

Built-in Functions, Under the Hood

Everything in Python is an object!

- Each object has various methods - some of these are ‘named’ methods that will work on many different object (any object with that method defined)
- For example:
  - len() calls the object method named object.__len__()
- This can be extended to your own objects by defining that __len__() method in them

Variable Names

Built in functions should NEVER be used as variable names. Python will allow this, and will in turn overwrite the built in function.

For example:

input = 25

will overwrite the input() function until the Python interpreter is restarted (or reaches the end of the program), and input() will no longer work!

Help with Functions

The help() function returns the docstring for a function, which typically describes what it is. For example, help(print) will print out a short message about the print() function.

Do not put parentheses after the function name when passing it to the help() function.

Math and Logic

sum() returns the sum of all elements passed into it.

max() returns the maximum value

min() returns the minimum value

all() returns True if **all** elements passed into it are True.

any() returns True if **any** elements passed into it are True.

Working with Lists and Other Iterables

map() applies a function to each element in a list or iterable

all() returns True if **all** elements passed into it are True.

any() returns True if **any** elements passed into it are True.

Repackaging Lists and Other Iterables

zip() takes in multiple iterables (i.e., lists), a series of tuples where each tuple contains the n-th element from each inputted iterable

enumerate() returns each item in a list or iterable, along with an int that represents its index number. Each result is returned in a tuple with the conter int first and the element second.

Counting

range() returns an iterator (a list that yields one element at a time), from the starting value (optional), to the ending value (required, exclusive), by a given step size (optional)

range(5) yields: 0,1,2,3,4

range(2,5) yields: 2,3,4

range(2,5,2) yields: 2,4

range(5,0,-1) yields: 5,4,3,2,1

We'll come back to this when we introduce FOR loops. For now, just be aware that range() is for counting.

All Variables

The following are variable types. When they are called as a function (with parantheses after the name), they return the default instance of that variable. For example, int() returns the integer 0, and str() returns an empty string (‘’).

int(), float(), complex(), bool(), hex(), tuple(), list(), str(), dict(), set()

When an argument is passed to these, the input is converted to that type and returned. For example, str(myint) would return the value of myint converted to a string.

All the Rest

The following doc lists all the built in functions.

https://docs.python.org/3/library/functions.html