CS150 - Final "Cheat Sheet"

1 Input/Output

- Reading input from the user

  `raw_input(message)`: Displays *message* to the user and return what the user typed as a string

- Reading from a file

```
file = open(filename, "r")

for line in file:
    # do something

file.close()
```

- Writing to a file

  – Opening a file for writing:

```
file = open(filename, "w")  #  begin writing from the beginning
# or
file = open(filename, "a")   # append to the end of the existing contents
```

  – the `write` method writes strings and other objects to the opened file (without and carriage return)

- Reading from URLs (e.g. web pages)

```
import urllib

web_page = urllib.urlopen(some_url)

for line in web_page:
    # do something
```

- Command-line arguments

  `import sys`

  `sys.argv` is a list containing the command-line arguments.

2 Strings

- The following functions are built-in and answer questions about strings
  - `len(string)`: Returns the number of characters in the string
  - `int(string) float(string)`: Converts a string to an `int` or `float`
- String object methods
  - `upper() lower()`: Returns a new string that is upper or lower cased
  - `find(some_string)`: Returns the index that *some_string* occurs at in the string or -1 if it does not occur.
  - `find(some_string, index)`: Same as above, but starts searching at *index*
  - `replace(old, new)`: Return a copy of the string with all occurrences of *old* substituted with *new*
  - `startswith(prefix)`: Returns `True` if the string starts with *prefix*, `False` otherwise
  - `endswith(prefix)`: Returns `True` if the string ends with *prefix*, `False` otherwise
  - `strip()`: Returns a copy of the string with leading and trailing whitespace removed
  - `split()`: Return a list of the words in the string using a space as the delimiter
- String operators
  - `string1 + string2`: Returns a new string that is the concatenation of *string1* and *string2*
  - `string * int`: Returns a new string that is *string* repeated *int* times

3 Lists

- Creating new lists
  - `[]` creates an empty list
  - `[object1, object2, object3, ...]` Creates a list containing the objects
- The following functions are built-in and answer questions about lists
  - `len(list)`: Returns the number of entries in the list

- List object methods
  - `append(x)`: Adds x to the end of the list
  - `extend(other_list)`: Adds all of the elements in *other_list* to the end of the list
  - `find(item)`: Return the index of the first occurrence of `item` in the list, -1 if `item` does not occur in the list
  - `insert(index, x)`: Insert x at *index* in the list
  - `pop()`: Removes the item at the end of the list and returns it
  - `pop(index)`: Removes item at *index* from the list and returns it
  - `reverse()`: Reverses the elements in the list
  - `sort()`: sorts the elements in the list
- List operators
  - `list1 + list2`: Returns a new list that contains the elements of *list1* followed by the elements of *list2*
  - `list * int`: Returns a new list that contains the items in *list* repeated *int* times
  - `item in list` Returns `True` if `item` is in `list`, `False` otherwise

4 Sets

- Creating new sets
  - `set()` creates an empty set
  - `set(iterable)` can also be called with any `iterable` object (e.g. strings or lists) and it will create a new set with each item in that iterable object
- The following functions are built-in and answer questions about sets
  - `len(set)`: Returns the number of entries in the set
- Set object methods
  - `add(...)`: Add an element to the set
  - `clear()`: Remove all elements from the set
  - `difference(...)`: Returns the difference of two sets
  - `intersection(...)`: Returns the intersection of two sets
  - `pop()`: Remove an arbitrary element from the set
  - `remove(item)`: Removes the item from the set
  - `union(...)`: Returns the union of two sets
- Set operators
  - `item in set` Returns `True` if `item` is in `set`, `False` otherwise

5 Dictionaries

- Creating dictionaries
  - {} creates a new empty dictionary
  - {`key1:value1, key2:value2, ...`} creates a new dictionary with key value pairs
- The following functions are built-in and answer questions about dictionaries
  - `len(dict)`: Returns the number of entries (key/value pairs) in the dictionary
- Dictionary object methods
  - `clear`: Remove all of the items in the dictionary
  - `items`: Returns a list of key/value tuples of the key/value pairs in the dictionary
  - `keys`: Returns a list of the keys in the dictionary
  - `values`: Returns a list of the values in the dictionary
- Dictionary operators
  - `item in dict` Returns `True` if `item` is in the keys of `dict`, `False` otherwise

6 Tuples

- Creating new tuples
  - `()` creates an empty tuple
  - `(object1, object2, object3, ...)` Creates a list containing the objects
- The following functions are built-in and answer questions about lists
  - `len(tuple)`: Returns the number of entries in the tuple
- Tuple operators
  - `tuple1 + tuple2`: Returns a new tulple that contains the elements of *tuple1* followed by the elements of *tuple2*
  - `item in tuple` Returns `True` if `item` is in `tuple`, `False` otherwise

7 Modules

- `turtle` module
  - `forward(distance)`: Move the turtle forward by the specified *distance*
  - `right(angle) left(angle)`: Turn the turtle right/left by *angle*
  - `goto(x, y)`: Move turtle to position x, y
  - `setheading(angle)`: Set the turtles heading to *angle*
  - `circle(radius)`: Draw a circule with radius *radius*
  - `penup()`: Pull the pen up – no drawing when moving
  - `pendown()`: Put the pen down – drawing when moving
  - `fillcolor(color)`: Change the fill color to *color*, where *color* is a string
  - `begin_fill()`: Start filling
  - `end_fill()`: Fill in the shape drawn since the last call to `begin_fill`
- `random` module
  - `randint(a, b)`: Return a random integer N such that $a \leq N \leq b$
  - `uniform(a, b)`: Return a random floating point number N such that $a \leq N \leq b$
- `math` module
  - `sqrt(num)`: Return the square root of *num*
- `matplotlib` module

  Importing: `from matplotlib import pyplot`
  - `pyplot.plot(x, y)`: add data in lists `x` and `y` to the plot
  - `pyplot.show()`: display the graph
  - `pyplot.xlabel(string)`: label the x-axis with `string` (similarly `pyplot.ylabel`)
  - `pyplot.title(string)`: set `string` as the title of the plot