

Python Summary

The textbook has many useful tables describing various functions and what they do. Here I have tried to bring everything together so that you can have it as an easy reference. While I have included everything that was in the tables in the book, I have rearranged some parts to avoid duplication and added some extra commands where I thought they may be useful.



Contents

1	Common Syntax	1
2	Built-in Functions	2
3	Common Libraries	2
4	Portable Graphics Library	3
5	String Methods	7
6	List Methods	9
7	Dictionary Methods	10
8	Set Methods	10



1 Common Syntax

If statements

```
if condition:
    statements
```

```
if condition:
    statements
else:
    statements
```

```
if condition1:
    statements
elif condition2:
    statements
elif condition3:
    statements
else:
    statements
```

Loop statements

While statements:

```
while condition:
    statements
```

For statements:

```
for var in seq:
    statements
```

```
for var in range(limit):
    statements
```

Other statements

Function definition

```
def name(parameters):
    statements
```

Return statement

```
return value
```

Try statement

```
try:
    body
except type:
    response
```

2 Built-in Functions

Built-in Python Functions

Built-in functions

<code>abs(<i>x</i>)</code>	Returns the absolute value of <i>x</i> .
<code>max(<i>x</i>, <i>y</i>, ...)</code>	Returns the largest of the arguments.
<code>min(<i>x</i>, <i>y</i>, ...)</code>	Returns the smallest of the arguments.
<code>round(<i>x</i>)</code>	Returns the closest integer to <i>x</i> .
<code>int(<i>x</i>)</code>	Converts <i>x</i> to an integer.
<code>float(<i>x</i>)</code>	Converts <i>x</i> to a floating-point number.
<code>len(<i>s</i>)</code>	Returns the length of the string argument <i>s</i> .
<code>str(<i>x</i>)</code>	Converts <i>x</i> to a string.
<code>str(<i>value</i>)</code>	Converts <i>x</i> to a string.

3 Common Libraries

Selections from the math library

Mathematical constants

<code>pi</code>	The mathematical constant π .
<code>e</code>	The mathematical constant <i>e</i> (the base for natural logarithm).

General mathematical functions

<code>sqrt(<i>x</i>)</code>	Returns the square root of <i>x</i> .
<code>floor(<i>x</i>)</code>	Returns the largest integer less than or equal to <i>x</i> .
<code>ceil(<i>x</i>)</code>	Returns the smallest integer greater than or equal to <i>x</i> .
<code>copysign(<i>x</i>,<i>y</i>)</code>	Returns <i>x</i> with the sign of <i>y</i> .

Logarithmic and exponential functions

<code>exp(<i>x</i>)</code>	Returns the exponential function of <i>x</i> (e^x).
<code>log(<i>x</i>)</code>	Returns the natural logarithm (base <i>e</i>) of <i>x</i> .
<code>log10(<i>x</i>)</code>	Returns the common logarithm (base 10) of <i>x</i> .

Trigonometric functions

<code>cos(<i>theta</i>)</code>	Returns the cosine of the radian angle <i>theta</i> .
<code>sin(<i>theta</i>)</code>	Returns the sine of the radian angle <i>theta</i> .
<code>tan(<i>theta</i>)</code>	Returns the tangent of the radian angle <i>theta</i> .
<code>atan(<i>x</i>)</code>	Returns the principal arctangent of <i>x</i> , which lies between $-\pi/2$ and $+\pi/2$.
<code>atan2(<i>y</i>,<i>x</i>)</code>	Returns the angle between the <i>x</i> -axis and the line from the origin to (<i>x</i> , <i>y</i>).
<code>radians(<i>angle</i>)</code>	Converts an angle measured in degrees to radians.
<code>degrees(<i>angle</i>)</code>	Converts an angle measured in radians to degrees.

Selections from the random library

Random integers

<code>randint(min, max)</code>	Returns a random integer between <i>min</i> and <i>max</i> , inclusive.
<code>randrange(limit)</code>	Returns a random integer from 0 up to but not including <i>limit</i> .
<code>randrange(start, limit)</code>	Returns a random integer from <i>start</i> up to but not including <i>limit</i> .

Random floating-point numbers

<code>random()</code>	Returns a random floating-point number in the range between 0 and 1.
<code>uniform(min, max)</code>	Returns a random floating-point number between <i>min</i> and <i>max</i> .

Random functions on lists

<code>choice(list)</code>	Returns a random element from the specified list.
<code>sample(list, k)</code>	Returns a list with <i>k</i> elements randomly chosen from <i>list</i> .
<code>shuffle(list)</code>	Rearranges the list in a random order.

Initialization functions

<code>seed()</code>	Randomizes the internal number generator.
<code>seed(k)</code>	Sets the internal state of the random number generator so that it generates the same sequence for any specific value of the integer <i>k</i> .

4 Portable Graphics Library

Methods for GWindow objects

<code>GWindow(width, height)</code>	Creates a new GWindow objects of the specified size.
<code>gw.get_width()</code>	Returns the width of the graphics window.
<code>gw.get_height()</code>	Returns the height of the graphics window.
<code>gw.add(obj)</code>	Adds the object to the graphics window.
<code>gw.add(obj, x, y)</code>	Repositions the object to <i>(x, y)</i> and adds it to the window.
<code>gw.remove(obj)</code>	Removes the object from the graphics window.
<code>gw.clear()</code>	Removes all objects from the graphics window.
<code>gw.get_element_at(x, y)</code>	Returns the topmost graphical object covering the point <i>(x, y)</i> . If no such object exists then <code>None</code> is returned.
<code>gw.add_event_listener(type, func)</code>	Prepares the window to respond to events of the specified <i>type</i> by calling <i>func</i> .
<code>gw.set_interval(func, delay)</code>	Prepares the window to <u>repeatedly</u> call <i>func</i> every <i>delay</i> milliseconds.
<code>gw.set_timeout(func, delay)</code>	Prepares the window to call <i>func</i> <u>once</u> after waiting <i>delay</i> milliseconds.

Creating Graphical Objects

Creating graphical objects

<code>GRect(<i>x</i>, <i>y</i>, <i>width</i>, <i>height</i>)</code>	Creates a <code>GRect</code> object with the specified dimensions.
<code>GRect(<i>width</i>, <i>height</i>)</code>	Creates a <code>GRect</code> object an (0,0) with the specified size.
<code>G Oval(<i>x</i>, <i>y</i>, <i>width</i>, <i>height</i>)</code>	Creates a <code>G Oval</code> object that fits inside the corresponding rectangle of the specified size.
<code>G Oval(<i>width</i>, <i>height</i>)</code>	Creates a <code>G Oval</code> object in which the oval ifts inside a rectangle of the specified size. The origin of the <code>G Oval</code> is at (0,0).
<code>GLine(<i>x</i>₁, <i>y</i>₁, <i>x</i>₂, <i>y</i>₂)</code>	Creates a <code>GLine</code> object connection (<i>x</i> ₁ , <i>y</i> ₁) and (<i>x</i> ₂ , <i>y</i> ₂).
<code>GLabel(<i>str</i>, <i>x</i>, <i>y</i>)</code>	Creates a <code>GLabel</code> object containing the specified string with its baseline origin at the point (<i>x</i> , <i>y</i>).
<code>GLabel(<i>str</i>)</code>	Creates a <code>GLabel</code> object containing the specified string with its baseline origin at the point (0,0).
<code>GArc(<i>x</i>, <i>y</i>, <i>width</i>, <i>height</i>, <i>start</i>, <i>sweep</i>)</code>	Creates a <code>GArc</code> object at the specified point and dimensions which starts at <i>start</i> degrees and extends counterclockwise <i>sweep</i> degrees.
<code>GPolygon()</code>	Creates an empty <code>GPolygon</code> object, which then needs vertices to be added.
<code>GCompound()</code>	Creates an empty <code>GCompound</code> object, into which other objects can then be added.

Methods available to all objects

Methods that control the location

<code>object.set_location(<i>x</i>, <i>y</i>)</code>	Sets the location of this object to (<i>x</i> , <i>y</i>).
<code>object.move(<i>dx</i>, <i>dy</i>)</code>	Moves the object using the displacements <i>dx</i> and <i>dy</i> .
<code>object.move_polar(<i>r</i>, <i>theta</i>)</code>	Moves the object <i>r</i> pixels in the direction <i>theta</i> .

Methods that control the appearance

<code>object.set_color(<i>color</i>)</code>	Sets the color used to display this object.
<code>object.set_line_width(<i>width</i>)</code>	Sets the width of the lines (in pixels) used to draw the object.
<code>object.set_visible(<i>flag</i>)</code>	Sets whether the object is visible, where <i>flag</i> is a boolean.
<code>object.rotate(<i>theta</i>)</code>	Rotates the object <i>theta</i> degrees about its origin.
<code>object.scale(<i>sf</i>)</code>	Scales the object by <i>sf</i> both horizontally and vertically.

Methods that control the stacking order

<code>object.send_backward()</code>	Moves this object one step backward in the stacking order.
<code>object.send_forward()</code>	Moves this object one step forward in the stacking order.
<code>object.send_to_back()</code>	Moves this object to the back of the stacking order.
<code>object.send_to_front()</code>	Moves this object to the front of the stacking order.

Methods that return properties

<code>object.get_x()</code>	Returns the <i>x</i> coordinate of the object.
<code>object.get_y()</code>	Returns the <i>y</i> coordinate of the object.
<code>object.get_width()</code>	Returns the width of the object.
<code>object.get_height()</code>	Returns the height of the object.
<code>object.get_color()</code>	Returns the color used to display this object.
<code>object.get_line_width()</code>	Returns the width of the lines used to draw the object.
<code>object.is_visible()</code>	Returns a boolean indicating whether the object is currently visible.
<code>object.contains(<i>x</i>, <i>y</i>)</code>	Check to see whether the point (<i>x</i> , <i>y</i>) is inside the object.

Methods only available to GFillableObject objects

GFillableObject objects include GRects, GObvals, GArCs, and GPolygons

<code>object.set_filled(<i>bool</i>)</code>	Sets whether the object is filled.
<code>object.set_fill_color(<i>color</i>)</code>	Sets the color used to fill the interior of the object.
<code>object.get_fill_color()</code>	Returns the color used to display the interior of the object.
<code>object.is_filled()</code>	Returns a boolean indicating whether the object is currently filled.

Methods only available to GLabel objects

<code>object.set_font(<i>str</i>)</code>	Sets the font for the label. The format of the font specification is a CSS string as described in the text.
<code>object.set_label(<i>str</i>)</code>	Sets the text of the label to the provided string.
<code>object.get_label()</code>	Returns the text of the label as a string.
<code>object.get_ascent()</code>	Returns the <i>font ascent</i> (the maximum distance above the baseline).
<code>object.get_descent()</code>	Returns the <i>font descent</i> (the maximum distance below the baseline).

Methods only available to GLine objects

<code>object.set_start_point(<i>x</i>, <i>y</i>)</code>	Changes the starting point of the line to (<i>x</i> , <i>y</i>) without changing the end.
<code>object.set_end_point(<i>x</i>, <i>y</i>)</code>	Changes the end point of the line to (<i>x</i> , <i>y</i>) without changing the start.
<code>object.get_start_point()</code>	Returns the starting point of the line.
<code>object.get_end_point()</code>	Returns the end point of the line.

Methods only available to GRect or GObval objects

<code>object.set_size(<i>width</i>, <i>height</i>)</code>	Sets the size of the object to the specified width and height.
<code>object.set_bounds(<i>x</i>, <i>y</i>, <i>width</i>, <i>height</i>)</code>	Sets both the location of the object and the size of the object.

Methods only available to GArc objects

<code>arc.set_start_angle(<i>start</i>)</code>	Sets the start angle to <i>start</i> degrees.
<code>arc.get_start_angle()</code>	Returns the start angle.
<code>arc.set_sweep_angle(<i>sweep</i>)</code>	Sets the sweep angle to <i>sweep</i> .
<code>arc.get_sweep_angle()</code>	Returns the sweep angle.
<code>arc.get_start_point()</code>	Returns the coordinate of the starting point of the arc.
<code>arc.get_end_point()</code>	Returns the coordinate of the ending point of the arc.

Methods only available to GPolygon objects

<code>poly.add_vertex(<i>x</i>, <i>y</i>)</code>	Adds a vertex at the point (<i>x</i> , <i>y</i>).
<code>poly.add_edge(<i>dx</i>, <i>dy</i>)</code>	Adds a vertex shifted by <i>dx</i> and <i>dy</i> from the preceeding vertex.
<code>poly.add_polar_edge(<i>r</i>, <i>theta</i>)</code>	Adds a vertex shifted by <i>r</i> units in direction <i>theta</i> .
<code>poly.get_bounds()</code>	Returns a GRect object of the bounding rectangle of the polygon.

Methods in the GImage class

Creating GImages

<code>GImage(filename)</code>	Creates a new GImage by reading the image data from the specified file.
<code>GImage(array)</code>	Creates a new GImage from the provided pixel array.

Method to read pixels

<code>image.get_pixel_array()</code>	Returns the pixel array for this image.
--------------------------------------	---

Static methods relating to GImages

<code>GImage.get_red(pixel)</code>	Returns the red component of the pixel as an integer between 0 and 255.
<code>GImage.get_green(pixel)</code>	Returns the green component of the pixel as an integer between 0 and 255.
<code>GImage.get_blue(pixel)</code>	Returns the blue component of the pixel as an integer between 0 and 255.
<code>GImage.create_rgb_pixel(r, g, b)</code>	Creates a pixel value from the specified <i>r</i> , <i>g</i> , and <i>b</i> components, each of which is between 0 and 255.
<code>GImage.create_rgb_pixel(a, r, g, b)</code>	Creates a pixel value from the specified <i>r</i> , <i>g</i> , and <i>b</i> components and with opacity <i>a</i> , each of which is between 0 and 255.

5 String Methods

Common methods in Python's string class

Finding patterns

<code>str.find(pattern)</code>	Searches the string <i>str</i> for the string <i>pattern</i> , starting at the beginning of <i>str</i> . Returns the first index at which the pattern appears, or <code>-1</code> if not found.
<code>str.find(pattern, k)</code>	Same as above, but starts the search at index <i>k</i> .
<code>str.rfind(pattern)</code>	Searches backward in <i>str</i> for the last instance of <i>pattern</i> , starting at the end of <i>str</i> . Returns the last index at which the pattern appears, or <code>-1</code> if not found.
<code>str.rfind(pattern, k)</code>	Same as above, but starts at index <i>k</i> .
<code>str.startswith(prefix)</code>	Returns <code>True</code> if <i>str</i> starts with the characters in <i>prefix</i> .
<code>str.endswith(suffix)</code>	Returns <code>True</code> if <i>str</i> ends with the characters in <i>suffix</i> .

Creating transformed strings

<code>str.lower()</code>	Returns a copy of <i>str</i> with all letters converted to lowercase.
<code>str.upper()</code>	Returns a copy of <i>str</i> with all letters converted to uppercase.
<code>str.capitalize()</code>	Returns a lowercase copy of <i>str</i> but with the first letter capitalized.
<code>str.lstrip()</code>	Returns a copy of <i>str</i> after removing any whitespace characters from the left side.
<code>str.rstrip()</code>	Returns a copy of <i>str</i> after removing any whitespace characters from the right side.
<code>str.strip()</code>	Returns a copy of <i>str</i> after removing any whitespace characters from both sides.
<code>str.replace(old, new)</code>	Returns a copy of <i>str</i> after replacing all instances of the string <i>old</i> with the string <i>new</i> .

Testing for character properties

<code>str.isalpha()</code>	Returns <code>True</code> if <i>str</i> is nonempty and contains only letters.
<code>str.isdigit()</code>	Returns <code>True</code> if <i>str</i> is nonempty and contains only numeric digits.
<code>str.isalnum()</code>	Returns <code>True</code> if <i>str</i> is nonempty and contains only letters or digits.
<code>str.islower()</code>	Returns <code>True</code> if <i>str</i> has at least one letter and all letters are lowercase.
<code>str.isupper()</code>	Returns <code>True</code> if <i>str</i> has at least one letter and all letters are uppercase.
<code>str.isspace()</code>	Returns <code>True</code> if <i>str</i> is nonempty and contains only whitespace characters (eg. space, tab, or newline).

Formatting strings

<code>str.center(width)</code>	Returns a copy of <i>str</i> centered in a field of the specified <i>width</i> .
<code>str.ljust(width)</code>	Returns a copy of <i>str</i> flush to the left in a field of the specified <i>width</i> .
<code>str.rjust(width)</code>	Returns a copy of <i>str</i> flush to the right in a field of the specified <i>width</i> .

Splitting and joining strings

<code>str.split(pattern)</code>	Splits the <i>str</i> into a list of strings by dividing it at <i>pattern</i> .
<code>str.splitlines()</code>	Splits a multiline string into a list of the individual lines.
<code>sep.join(list)</code>	Joins the elements of <code>list</code> into a string using <i>sep</i> to separate the elements.

Format Spec components

Symbol that you want to fill any empty space. By default it is the space character.

<, >, or ^ symbols controlling how the string is justified in the available width.

Grouping symbol (,) if desired. Defaults to none.

Controls desired number of digits after decimal. Has the form .*number*

[fill] [align] [sign] [width] [group] [precision] [type]

Digit representing the *minimum* number of characters the representation will occupy.

Including a + symbol will force always showing the sign of a number. (whether positive or negative)

Controls how the substitute value is represented:

- d: base-10 integers
- f: floating-point decimals
- e: scientific notation
- s: strings
- b: binary
- x: hexadecimal

6 List Methods

Common methods in Python's list class

Methods that leave the original list unchanged

<code>list.index(value)</code>	Returns the first index matching <i>value</i> . This method raises a <code>ValueError</code> exception if no match is found.
<code>list.index(value, start)</code>	Starting from <i>start</i> , returns the first index matching <i>value</i> . This method raises a <code>ValueError</code> exception if no match is found.
<code>list.count(value)</code>	Returns the number of times that <i>value</i> appears in the list.
<code>list.copy()</code>	Returns a <i>shallow</i> copy of the original list.

Methods that add or remove elements

<code>list.append(value)</code>	Adds <i>value</i> to the end of the list.
<code>list.extend(list₂)</code>	Adds the elements in <i>list₂</i> to the end of the list.
<code>list.insert(index, value)</code>	Inserts <i>value</i> before the specified <i>index</i> position.
<code>list.remove(value)</code>	Removes the first instance of <i>value</i> from the list, raising a <code>ValueError</code> exception if it does not appear.
<code>list.pop()</code>	Removes and returns the last element of the list.
<code>list.pop(index)</code>	Removes and returns the element at the specified <i>index</i> position.
<code>list.clear()</code>	Removes all elements from a list.

Methods that reorder the elements of a list

<code>list.reverse()</code>	Reverses the order of the elements in the list.
<code>list.sort()</code>	Sort the elements of the list in ascending order.
<code>list.sort(key)</code>	Sort the elements of the list in ascending order according to a particular <i>key</i> .
<code>list.sort(key, reverse=True)</code>	Sort the elements of the list in descending order according to a particular <i>key</i> .

String methods that involve lists

<code>str.split()</code>	Splits the string <i>str</i> into a list of substrings by dividing it at each occurrence of a whitespace character.
<code>str.split(separator)</code>	Splits the string <i>str</i> into a list of substrings by dividing it at each instance of the string <i>separator</i> .
<code>str.splitlines()</code>	Splits <i>str</i> into a list of substrings by dividing it at each line break.
<code>sep.join(list)</code>	Joins the elements of <i>list</i> into a string using <i>sep</i> to separate the elements.

7 Dictionary Methods

Common methods in Python's dict class

<code>len(dict)</code>	Returns the number of key-value pairs in the dictionary.
<code>dict.clear()</code>	Removes all key-value pairs from the dictionary, leaving it empty.
<code>dict.copy()</code>	Creates and returns a shallow copy of this dictionary.
<code>dict.get(key, value)</code>	Returns the value associated with <i>key</i> in the dictionary, or the specified <i>value</i> if <i>key</i> is not found. The default value for <i>value</i> is <code>None</code> .
<code>dict.setdefault(key, value)</code>	If <i>key</i> is not in the dictionary, <code>setdefault</code> creates a new entry for the key-value pair. The method then returns the value, which is either its previous value or the newly created one.
<code>dict.pop(key)</code>	Removes and returns the key-value pair corresponding to <i>key</i> . If the key is not found, then a <code>KeyError</code> exception is raised.
<code>dict.items()</code>	Returns an iterable object that returns successive tuples consisting of a key-value pair.

8 Set Methods

Common operators and methods in Python's set class

<code>set()</code>	Creates an empty set. The expression <code>{ }</code> creates an empty dictionary.
<code>set(iterable)</code>	Creates a set from the elements of the iterable object. (Which could be a list, string, etc. Anything you could loop over.)
<code>len(set)</code>	Returns the number of elements in the set.
<code>set1.union(set2)</code> <code>set1 set2</code>	Returns the union of <i>set1</i> and <i>set2</i> .
<code>set1.intersection(set2)</code> <code>set1 & set2</code>	Returns the intersection of <i>set1</i> and <i>set2</i> .
<code>set1.difference(set2)</code> <code>set1 - set2</code>	Returns the difference of <i>set1</i> and <i>set2</i> .
<code>set1.symmetric_difference(set2)</code> <code>set1 ^ set2</code>	Returns the symmetric difference of <i>set1</i> and <i>set2</i> .
<code>set1 == set2</code>	Returns <code>True</code> if <i>set1</i> and <i>set2</i> contain the same elements.
<code>set1 <= set2</code>	Returns <code>True</code> if <i>set1</i> is a subset of <i>set2</i> .
<code>set1 < set2</code>	Returns <code>True</code> if <i>set1</i> is a proper subset of <i>set2</i> .
<code>element in set</code>	Returns <code>True</code> if <i>element</i> is in the <i>set</i> .
<code>set1.isdisjoint(set2)</code>	Returns <code>True</code> if <i>set1</i> and <i>set2</i> contain no elements in common.
<code>set.clear()</code>	Removes all elements from the set, leaving it empty.
<code>set.copy()</code>	Creates and returns a shallow copy of the set.
<code>set.add(element)</code>	Adds the specified element to the set.
<code>set.remove(element)</code>	Removes the element from the set, raising a <code>ValueError</code> if it is missing.
<code>set.discard(element)</code>	Removes the element from the set, doing nothing if it is missing.