

Module	Description	Example	Script
core	continue, going on to next loop item	continue	g06/demo.py
core	dictionary, adding a new entry	co['po'] = 'CO'	g05/demo.py
core	dictionary, creating	co = {'name':'Colorado', 'capital':'Denver'}	g05/demo.py
core	dictionary, creating via comprehension	word_lengths = { w:len(w) for w in wordlist }	g06/demo.py
core	dictionary, iterating through key-value pairs	for w,l in word_lengths.items():	g06/demo.py
core	dictionary, looking up a value	name = ny['name']	g05/demo.py
core	dictionary, making a list of	list1 = [co,ny]	g05/demo.py
core	dictionary, obtaining a list of keys	names = super_dict.keys()	g05/demo.py
core	file, closing	fh.close()	g02/demo.py
core	file, opening for reading	fh = open('states.csv')	g05/demo.py
core	file, opening for writing	fh = open(filename,"w")	g02/demo.py
core	file, output using print	print("It was written during",year,file=fh)	g02/demo.py
core	file, output using write	fh.write("Where was this file was written?\n")	g02/demo.py
core	file, reading one line at a time	for line in fh:	g05/demo.py
core	for, looping through a list	for n in a_list:	g04/demo.py
core	function, calling	d1_ssq = sumsq(d1)	g07/demo.py
core	function, defining	def sumsq(values: list) -> float:	g07/demo.py
core	function, returning a result	return values	g07/demo.py
core	function, using type hinting	def readlist(filename: str) -> list:	g07/demo.py
core	if, starting a conditional block	if l == 5:	g06/demo.py
core	if, using an elif statement	elif s.isalpha():	g06/demo.py
core	if, using an else statement	else:	g06/demo.py
core	list, appending an element	a_list.append("four")	g03/demo.py
core	list, create via comprehension	cubes = [n**3 for n in a_list]	g04/demo.py
core	list, creating	a_list = ["zero", "one", "two", "three"]	g03/demo.py
core	list, determining length	n = len(b_list)	g03/demo.py
core	list, extending with another list	a_list.extend(a_more)	g03/demo.py
core	list, generating a sequence	b_list = range(1,6)	g04/demo.py
core	list, joining with spaces	a_string = " ".join(a_list)	g03/demo.py
core	list, selecting an element	print(a_list[0])	g03/demo.py
core	list, selecting elements 0 to 3	print(a_list[:4])	g03/demo.py

Module	Description	Example	Script
core	list, selecting elements 1 to 2	<code>print(a_list[1:3])</code>	g03/demo.py
core	list, selecting elements 1 to the end	<code>print(a_list[1:])</code>	g03/demo.py
core	list, selecting last 3 elements	<code>print(a_list[-3:])</code>	g03/demo.py
core	list, selecting the last element	<code>print(a_list[-1])</code>	g03/demo.py
core	list, sorting	<code>c_sort = sorted(b_list)</code>	g03/demo.py
core	list, summing	<code>total = sum(numbers)</code>	g06/demo.py
core	math, raising a number to a power	<code>a_cubes.append(n**3)</code>	g04/demo.py
core	math, rounding a number	<code>rounded = round(ratio,2)</code>	g05/demo.py
core	string, concatenating	<code>name = s1+" "+s2+" "+s3</code>	g02/demo.py
core	string, convert to lower case	<code>lower = [s.lower() for s in wordlist]</code>	g06/demo.py
core	string, convert to title case	<code>new_s = s.title()</code>	g06/demo.py
core	string, converting to an int	<code>value = int(s)</code>	g06/demo.py
core	string, creating	<code>filename = "demo.txt"</code>	g02/demo.py
core	string, finding starting index	<code>mm_start = long_string.find("mm")</code>	g06/demo.py
core	string, including a newline character	<code>fh.write(name+"!\n")</code>	g02/demo.py
core	string, is entirely numeric	<code>if s.isnumeric():</code>	g06/demo.py
core	string, matching a substring	<code>has_ñ = [s for s in lower if "ñ" in s]</code>	g06/demo.py
core	string, matching end	<code>a_end = [s for s in lower if s.endswith("a")]</code>	g06/demo.py
core	string, matching multiple starts	<code>ab_start = [s for s in lower if s.startswith(starters)]</code>	g06/demo.py
core	string, matching start	<code>a_start = [s for s in lower if s.startswith("a")]</code>	g06/demo.py
core	string, replacing a substring	<code>words = s.replace(","," ").split()</code>	g06/demo.py
core	string, splitting on a comma	<code>parts = line.split(',')</code>	g05/demo.py
core	string, splitting on whitespace	<code>b_list = b_string.split()</code>	g03/demo.py
core	string, stripping blank space	<code>clean = [item.strip() for item in parts]</code>	g05/demo.py
core	tuple, creating	<code>starters = ("a","b","0")</code>	g06/demo.py
json	importing the module	<code>import json</code>	g05/demo.py
json	using to print an object nicely	<code>print(json.dumps(list1,indent=4))</code>	g05/demo.py