Python Reference Sheet for Compsci 101, Exam 2, Spring 2025

DO NOT WRITE ANYTHING TO BE GRADED ON THESE REFERENCE SHEETS!!

Mathematical Operators		
Symbol	**Meaning**	**Example**
+	addition	4 + 5 = 9
-	subtraction	9 - 5 = 4
*	multiplication	3*5 = 15
/ and //	division	6/3 = 2.0 6/4 = 1.5 6//4 = 1
%	mod/remainder	5 % 3 = 2
**	exponentiation	3**2 = 9, 2**3 = 8
String Operators		
+	concatenation	"ab"+"cd"="abcd"
*	repeat	"xo"*3 = "xoxoxo"
Comparison Operators		
==	is equal to	3 == 3 is True
!=	is not equal to	3 != 3 is False
>=	is greater than or equal to	4 >= 3 is True
<=	is less than or equal to	4 <= 3 is False
>	is strictly greater than	4 > 3 is True
<	is strictly less than	3 < 3 is False
Boolean Operators		
x=5		
not	flips/negates the value of a bool	(not x == 5) is False
and	returns True only if both parts of it are True	(x > 3 and x < 7) is True (x > 3 and x > 7) is False
or	returns True if at least one part of it is True	(x < 3 or x > 7) is False (x < 3 or x < 7) is True
Type Conversion Functions		
int(x)	turn x into an integer value	int("123") == 123 int(5.8) == 5
	int can fail, e.g., int("abc") raises an error	
float(x)	turn x into an float value	float("2.46") == 2.46
	float can fail, e.g., float("abc") raises an error	
str(x)	turn x into a string value	str(432) == "432"
type(x)	the type of x	type(1) == int type(1.2) == float

String Index and Splicing

`s="colorful"`

		Example
s[x]	index a character	s[0] == 'c' s[-3] == 'f' s[5] == 'f'
s[x:y]	splice of string, substring from index x up to but not including index y	s[2:5] == 'lor' s[:5] == 'color' s[4:-1] == 'rfu' s[5:] == 'ful'

String Functions

`s="colorful"`

Name	Returns	Example
.find(str)	index of first occurrence	s.find("o") == 1
		s.find("e") == -1
.rfind(str)	index of last occurrence	s.rfind("o") == 3
		s.rfind("e") == -1
.index(str)	same as .find(str), error if str not in string	s.index("o") == 1
.count(str)	number of occurrences	s.count("o") == 2 s.count("r") == 1 s.count("e") == 0
.strip()	copy with leading/trailing whitespace removed	" big ".strip() == "big"
.split()	list of "words" in s	"big bad dog".split() == ["big","bad", "dog"]
.split(",")	list of "items " in s that are separated by a comma *In general can split on any string, not just a comma, e.g., s.split(":") will split on a colon and s.split("gat") will split on the string "gat".*	"this,old,man".split(",") == ["this", "old", "man"]
' '.join(lst)	concatenate elements of lst, a list of strings, separated by ' ' or any string	':'.join(['a','b','c']) == "a:b:c"
.startswith(str)	boolean if starts with string	s.startswith("color") == True s.startswith("cool") == False
.endswith(str)	boolean if ends with string	s.endswith("ful") == True s.endswith("color") == False
.upper()	uppercase of s	s.upper() == "COLORFUL"
.lower()	lowercase of s	"HELLO".lower() == "hello"
.isupper()	boolean is uppercase	'A'.isupper() == True 'a'.isupper() == False
.islower()	boolean is lowercase	'A'.islower() == False 'a'.islower() == True
.isalpha()	boolean is alphabetic character	'3'.isalpha() == False '?'.isalpha() == False 'z'.isalpha() == True
.capitalize()	capitalized s	s.capitalize() == "Colorful"

Miscellaneous Functions

help(x)	documentation for module x	
len(x)	length of sequence x, e.g., String or List	len("duke") == 4
list(str)	a list of the characters from string str	list("cards") == ['c','a','r','d','s']
sorted(x)	return list that is sorted version of sequence/iterable x, doesn't change x	sorted("cat") == ['a','c','t']
range(x)	a list of integers starting at 0 and going up to but not including x	range(5) == [0, 1, 2, 3, 4]
range(start, stop)	a list of integers starting at start and going up to but not including stop	range(3, 7) == [3, 4, 5, 6]
range(start, stop, inc)	a list of integers starting at start and going up to but not including stop with increment inc	range(3, 9, 2) == [3, 5, 7]
min(x, y, z)	minimum value of all arguments	min(3, 1, 2) == 1 min("z", "b", "a") == "a"
max(x, y, z)	maximum value of all arguments	max(3, 1, 2) == 3 max("z", "b", "a") == "z"
abs(x)	absolute value of the int or float x	abs(-33) == 33 abs(-33.5) == 33.5

List index, splicing and concatenation (+)

`lst =[3, 6, 8, 1, 7]`

		Example
lst[x]	index an element	lst[0] == 3 lst[-1] == 7
lst[x:y]	splice of list, sublist from index x up to but not including index y	lst[1:3] == [6, 8] lst[:4] == [3, 6, 8, 1] lst[3:] == [1,7]
+ operator	concatenate two lists	[3,4] + [1,3,2] == [3,4,1,3,2]

List Functions

`lst =[3, 6, 8, 1, 7]`

sum(lst)	returns sum of elements in list `lst`	sum([1,2,4]) == 7
max(lst)	returns maximal element in `lst`	max([5,3,1,7,2]) == 7
lst.append(...)	append an element to lst, changing lst	[1,2,3].append(8) == [1,2,3,8]
lst.insert(pos,elt)	append `elt` to `lst` at position `pos`, changing `lst`	[1,2,3].insert(1,8) == [1,8,2,3]
lst.extend(lst2)	append every element of lst2 to lst	[1,2,3].extend([8,9]) == [1,2,3,8,9]
lst.remove(elt)	remove first occurence of `elt` from `lst`	[1,2,3,2,3,2].remove(2) == [1,3,2,3,2]
lst.sort()	sorts the elements of `lst`	lst = [3,6,8,1,7] lst.sort() lst is now [1, 3, 6, 7, 8]
lst.index(elt)	return index of elt in `lst`, error if elt not in lst	[1,5,3,8].index(5) == 1
lst.count(elt)	return number of occurrences of elt in `lst`	[1,2,1,2,3].count(1) == 2
lst.pop()	remove and return last element in `lst`, so has side-effect of altering list and returns value.	lst = [3,6,8,1,7] x = lst.pop() x is 7, lst is [3,6,8,1]

lst.pop(index)	remove and return element at position index in `lst`, so has side-effect of altering list and returns value. Default index is last value.	lst = [3,6,8,1,7] x = lst.pop(1) x is 6, lst is [3,8,1,7]
Math Functions (import math)		
math.pi	3.1415926535897931	
math.sqrt(num)	returns square root of num as float	math.sqrt(9) == 3.0
File Functions		
open("filename")	opens a file, returns file object	f = open("foo.txt")
open("filename",mode)	specify mode of 'r', 'a', 'w', return file object	f = open("foo.txt", "a")
f.read()	returns the entire file as one string	s = f.read()
Random Functions (import random)		
random.choice(list_of_choices)	returns a random element from list_of_choices. Gives an error if list_of_choices has length 0.	
random.randint(start, end)	Returns a random integer between start and end. Unlike range() and list slicing, the largest value it can return is end, not end-1.	
random.random()	Returns a random float between 0 and 1.	
Set Functions		
set(lst)	returns a set of the elements from list `lst`	
s.add(item)	adds the item into the set, and returns nothing.	
s.update(lst)	adds the elements in the list `lst` into the set, and returns nothing.	
s.remove(item)	removes the item from the set, error if item not there.	
s.union(t)	returns new set representing s UNION t, i.e., all elements in either s OR t, t can be any iterable (e.g., a list)	
s.intersection(t)	returns new set representing s INTERSECT t, i.e., only elements in both s AND t, t can be any iterable (e.g., a list)	
s.difference(t)	returns new set representing s difference t, i.e., elements in s that are not in t	
s.symmetric_difference(t)	returns new set representing elements in s or t, but not in both	
s | t	returns/evaluates to union of s and t, both must be sets.	
s & t	returns/evaluates to intersection of s and t, both must be sets.	
s - t	returns/evaluates to set with all elements in s that are *not* in t	
s ^ t	returns/evaluates to set with all elements from s and t that are *not* in both s and t	
Dictionary Functions		
d[key]	returns the value associated with `key`, error if `key` not in dictionary `d`	
d.get(key)	returns value associated with `key`, returns None if `key` not in dictionary `d`	
d.get(key,default)	returns value associated with `key`, returns `default` if `key` not in `d`	
d.keys()	returns a list/view of the keys in dictionary	
d.values()	returns a list/view of values in dictionary	
d.items()	returns a list/view of tuples, (key,item) pairs from dictionary	
d.update(dict)	updates the dictionary with another dictionary `dict`	