*Python Slicing Basics Study Sheet Last Update 1 Aug ‘23*

What can slices be used on? List, tuple, string

Slicing Syntax: my\_list[x:y:z]

What are x, y, and z? start, stop, and step indices

What do x, y, and z default to if left blank?

start of the list, end of the list, and step size of 1

*Consider my\_list = [‘a’, ‘b’, ‘c’, ‘d’, ‘e’]*

What are the results of the following?

my\_list[1:4] [b, c, d] (includes first, skips last)

my\_list[3:] [d, e] (3th element to end)

my\_list[:2] [a, b] (stops before 2th element)

my\_list[:2] + my\_list[2:] [a, b, c, d, e] (the full list)

my\_list[-3:-1] [c, d]

my\_list[-3:] [c, d, e]

my\_list[:-3] [a, b]

n.b. negative indices still satisfy the nice property L[s:]+L[:s] = L

my\_list[::-1] [e, d, c, b, a] (iterates through backwards)

my\_list[::2] [a, c, e] (step size is 2)

*Consider my\_list = [‘a’, ‘b’, ‘c’, ‘d’, ‘e’]*

What are the results of the following?

my\_list[3::-1] [d, c, b, a] start at 3th element (and include it because it’s the start element) and go backwards

my\_list[:3:-1] [e] end at 3th element (and don’t include it because it’s the stop element) and go backwards

my\_list[-3::-1] [c, b, a] start from -3th element (and include it) and go backwards

my\_list[:-3:-1] [e, d] iterate backwards, stop at (and not including) -3th element

*write code to:*

take a slice from 2th to 4th index, inclusive, moving backwards:

my\_list[4:1:-1] remember, always include first index, always exclude last, so need to write 1 instead of 2 to avoid an off by one error; must write first and last indices in reverse order as we are moving backwards

take a slice from the second to first index to the second to last index, inclusive, moving backwards:

my\_list[-2:0:-1] similarly, start at second to last index, go to second to first index (so we use 0 because we don’t include the stop index), and write first and last indices in this order to align with direction of traversal