List vs Set vs Dictionary vs Tuple

Lists	Sets	Dictionaries	Tuples
list = [10, 12, 15]	set = {1, 23, 34} print(set) -> {1, 23, 34} set = {1, 1} print(set) -> {1}	dict = {"ram": 26, "mary": 24}	words = ("spam", "eggs") Or words = "spam", "eggs"
Access: print(list[0])	Print(set[0]) Set elements can't be indexed	print(dict["ram"])	print(words[0])
Can contain duplicate elements	Can't contain duplicate elements. Faster compared to Lists	Can't contain duplicate keys, but can contain duplicate values.	Can contain duplicate elements. Faster compared to Lists
Ordered	Unordered	Unordered (OrderedDict, to have an ordered dictionary)	Ordered
list[0] = 100	set.add(7)	dict["ram"] = 27	words[0] = "care" => TypeError
Mutable - So, Unhashable	Mutable (!= FrozenSet) So, Unhashable	Mutable - So, Unhashable	Immutable – values can't be changed once assigned So, Hashable
List = []	Set = set()	Dict = {}	Tuple = ()
Slicing can be done print(list[1:2]) -> 12	Slicing: Not done	Slicing: Not done	Slicing can also be done on tuples
Usage: - Use lists if you have a collection of data that doesn't need random access - Use lists when you need a simple, iterable collection that is modified frequently - General purpose - Grow and shrink size as needed - Most widely used data structure	Usage: - Membership testing and the elimination duplicate entries - When you need uniqueness for the elements - Math Set ops (union, intersect, …)	Usage: - When you need a logical association b/w key:value pair - When you need fast lookup for your data based on a custom key - When your data is being constantly modified	Usage: - Use tuples when your data cannot change - A tuple is used in combination with a dictionary, for example a tuple might represent a key, because it is immutable. - Useful for fixed data