Data Structure	Definition	Data Type	Is Mutable? (alterable once declared)	Declaration	Example	Common Functions	Use
Lists	A list is a mutable, ordered sequence of items	Integers, Strings, Objects and combination of all these	Yes	**[ ]**	List = [1, 2, 4, 'Harry']	1. append () – Adds an element to the end of the list 2. clear () – Removes all elements from the list 3. copy () – Returns a shallow copy of the list 4. count () – Returns the total number of items passed as an argument 5. extend () – Adds all elements of a list to some other list 6. index () – Returns the index of an element (Note: If the same element appears multiple times in the 7. list then the index of the very first match is returned) 8. insert () – Inserts an element to the list at the defined index 9. pop () – Eliminates and returns an element from the list 10. remove () – Eliminates an element from the list 11. reverse () – Reverses the order of all elements of the list 12. sort () – Sort all elements of a list in the ascending order	Use lists when you need a collection of similar/non similar items. Example: similar items = ['Bread','Butter','Jam','Milk']
Tuples	Tuples are just like lists and are usually faster than lists due to smaller size	Integers, Strings, Objects, more Lists and combination of all these	No	**()**	Tuple = (1, "A", 2+3,"Word")	1. cmp () – Compares elements of two tuples 2. len () – Gives out the total length of some tuple 3. max () – Returns the biggest value from a tuple 4. min () – Returns the smallest value from a tuple 5. tuple () – Converts some list into a tuple	Use a tuple when you know what information goes in it. Example: When you want to store a user's credentials for your website. User = ('Name','visitor','pass123')
Dictionaries	A dictionary is a key:value pair, similar to an associative array found in other programming languages. A dictionary is like an address-book. Each value will be associated a unique key. Dictionary keys are case sensitive. Tuple can also be a dictionary key. Neither a list nor another dictionary can serve as a dictionary key.	Integers, Strings, Floating points etc.	No	**{}**	Dict = {1: 'Hello', 'text': 'World', 3: 'Hi'}	1. copy()-They copy() method returns a shallow copy of the dictionary.2. clear()-The clear() method removes all items from the dictionary.3. pop()-Removes and returns an element from a dictionary having the given key.4. popitem()-Removes the arbitrary key-value pair from the dictionary and returns it as tuple.5. get()-It is a conventional method to access a value for a key.6. dictionary_name.values()- returns a list of all the values available in a given dictionary.7. str()-Produces a printable string representation of a dictionary.8. update()-Adds dictionary dict2's key-values pairs to dict9. setdefault()-Set dict[key]=default if key is not already in dict10. keys()-Returns list of dictionary dict's keys11. items()-Returns a list of dict's (key, value) tuple pairs12. has_key()-Returns true if key in dictionary dict, false otherwise	Use when you need a lot of organization because the data is very dependent on each other. i.e. When you need a logical association between a key:value pair. Example: Telephone directory/book.
Sets	A set is an unordered / unindexed collection of simple objects in Python. Does not take duplicate Values		Yes	**{}**	Set_1 = {"book1", "book2", "book3"}	1. add () – Adds an item to the set (Note: As sets don't have repeating values, the item that is to be 2. added to a set must not be already a member of the set.) 3. clear () – Removes all items of the set 4. difference () – Returns a set with all elements of the invoking set but not of the second set 5. intersect () – Returns an intersection of two sets 6. union () – Returns a union of two sets	Useful when you need to ensure that an item only appears once in the set. Example: You have a list of numbers and you needed the entire list but wanted a second list which does not include the duplicates from the first list