Welcome Back Everyone

Agenda for Today

1. Set
2. String
3. Dictionary
4. Functions

Shallow vs deep in set

Set methods

1. Copy () ⇒ does shallow copying

2. pop () ⇒ pop out random ~~element~~ as set is not having any sequence

3. Remove () # error if not present else removes.

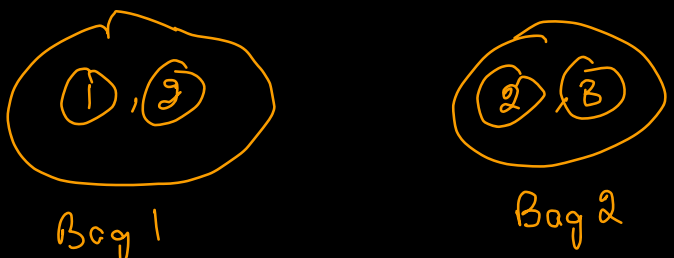
Remove	vs	Pop()
1. Just removes		1. Removes & return
2. Has an argument as the value which we wanna remove		2. No arguments

$s1 = \{1, 2, 3\}$

$s2 = \{3, 4, 5\}$

Union ⇒ Just give me the union as a set, doesn't change any of my set

update ⇒ It updates (make change) in the set on which it is called & equates it to the union



Bag 1 Bag 2

Bag(1). union(Bag 2) ⇒ 1, 2, 3

my Bag 1 and Bag 2 will not be changed

Bag 1. update (Bag 2) ⇒

Bag 1 will have items from Bag 2, ~~which~~ didn't have earlier

Strings

Collection of 0 or more characters

type ('a') # no char
type ('abc') # String

Single, double, triple Quotes

Indexing, Slicing

Concatenation

Membership

email list ⇒

'gmail' in ____

Country Code ⇒

'+91' in ____

Comparision in String

lexiographic comparision

'0' ⇒ 48

'A' ⇒ 65

'a' ⇒ 97

Identity Operator

is

is not

f string [string interpolation]

⇓

process of substituting values

%d , %s

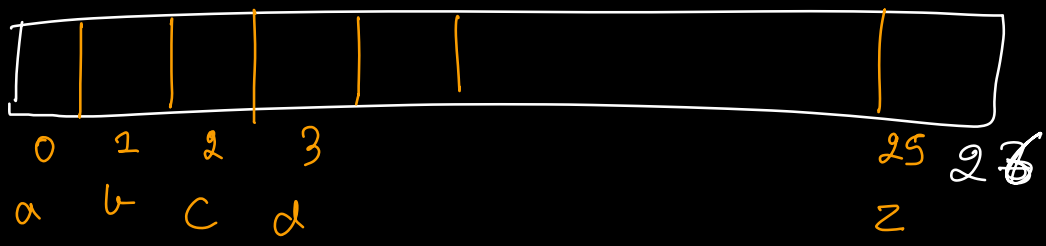
{ a } . format (a =)

f (f (" — { }

Dictionary Hashmap

"python is nice" a z '_'

count ~~alphabet~~ character



Now, what if you have to count words in a sentence

str = "I is I am python"

I → 2
is → 1
am → 1
python → 1

Till now we don't have any control on Key/Indexes.

l[0]

Dictionary are Key : value pair

Its a collection where I can define Key & corresponding value for my items I am storing.

l = [1, 2,]

0	1

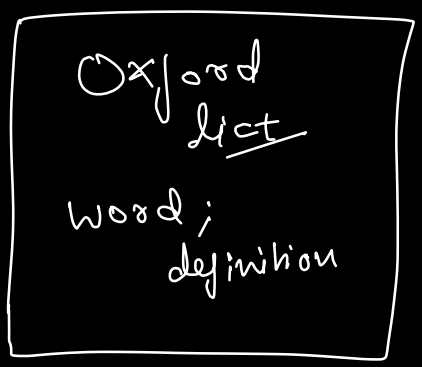
d = { Key1 : value1,
Key2 : value2,
:
}

dictionary items are changeable

don't allow duplicates

Key : value no duplicate Key

Items can be referred via there key name



set

python - a snake[1];
a programming language[2]

0 → a
1 → b

aaa8 → python

Looping in Dictionaries

⇒ By default Keys

- .keys
- .values
- .items