# DATA STRUCTURES

## LEGO BRICKS

Insight Builder

# PROBLEM SOLVED BY DS

## 1 UNDERSTAND DATA STRUCTURE

- Keeps the data so getting it back is easy
- Whole structure has different features, compared to the data it contains

## 2 WHAT IS THE PROBLEM?

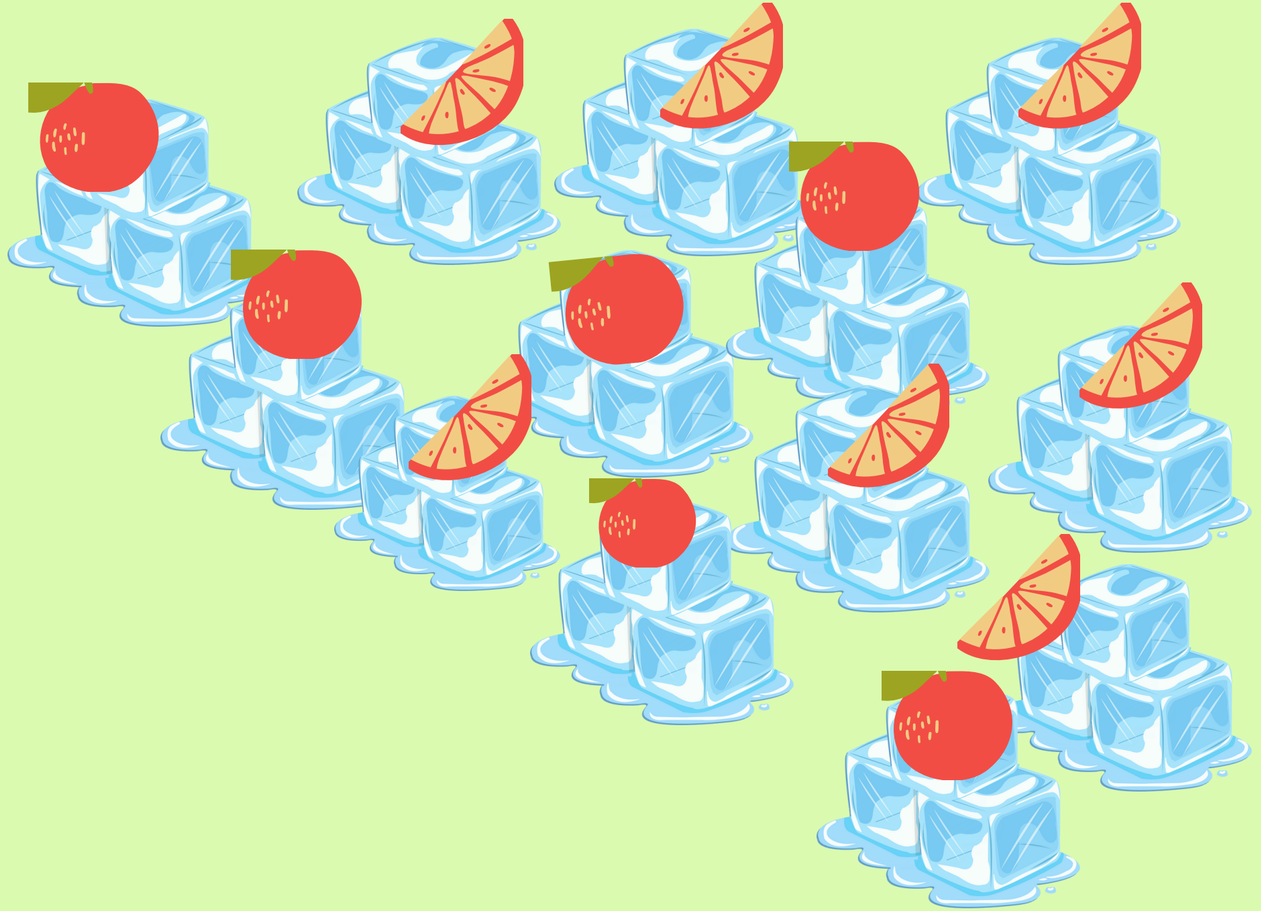
How to read and write Data that is huge in Volume, Variety and Velocity

## 3 HOW IT SOLVES THEM

- Building blocks of basic Templates fitted inside the templates for Data Structure

# CUBE

## LIST OF CUBES



## DICT OF CUBES & FRUITS

# EXAMPLE

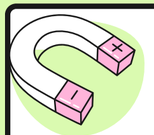
## DATA STRUCTURE

- Lists
- Dictionaries
- Sets

## VALUE

- [1,2,3,4,5,6,7,8,9,0]
- {'a':1,'b':2,'e':3,'k':57}
- {5,6,8,2,1,9}

## TEMPLATE



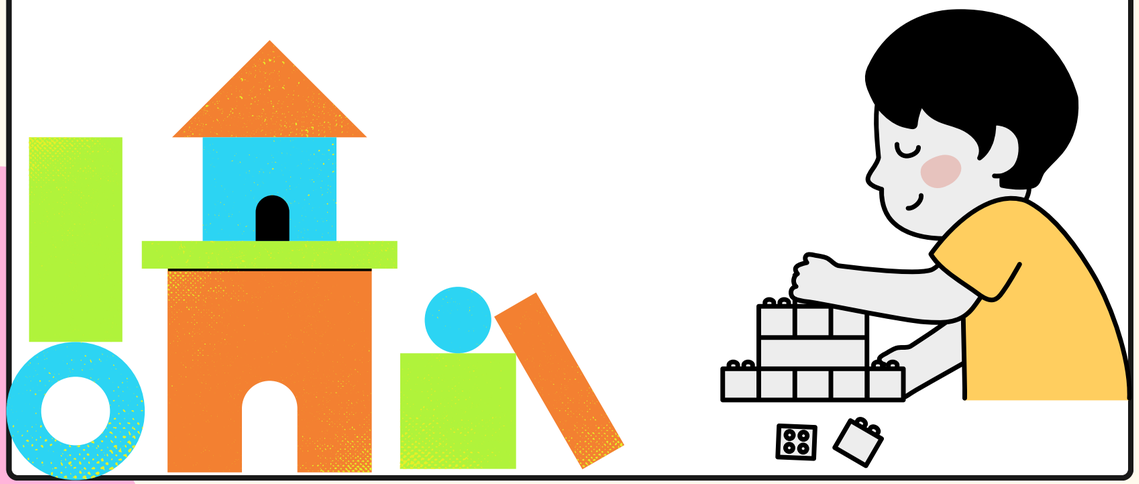
1. **List**
2. **Dict**
3. **Set**

## WHAT IT CAN DO?

- Loop, control and print data
- Direct access to Data
- No duplicates

# PSEUDO CODE

**Objective :** Write a script that creates List, Dict, Set, and then
List of Dicts,
Dict of lists,
Sets of Lists and print all



1. First scold the teacher ;)
2. Find out the commands
3. Create Lists, Dicts and Sets
4. Think how to create
    a. List of Dicts
    b. Dict of lists
    c. Sets of list
5. And Print them all, using print

# 1: FINDING COMMAND

## Objective 1: Explore list() and its methods

```
>>> fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
>>> fruits.count('apple')
2
>>> fruits.count('tangerine')
0
>>> fruits.index('banana')
3
>>> fruits.index('banana', 4)  # Find next banana starting at position 4
6
>>> fruits.reverse()
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'orange']
>>> fruits.append('grape')
>>> fruits
['banana', 'apple', 'kiwi', 'banana', 'pear', 'apple', 'orange', 'grape']
>>> fruits.sort()
>>> fruits
['apple', 'apple', 'banana', 'banana', 'grape', 'kiwi', 'orange', 'pear']
>>> fruits.pop()
'pear'
```

# 1: FINDING COMMAND

## Objective 2: Explore dict() and its methods

```
>>> tel = {'jack': 4098, 'sape': 4139}
>>> tel['guido'] = 4127
>>> tel
{'jack': 4098, 'sape': 4139, 'guido': 4127}
>>> tel['jack']
4098
>>> del tel['sape']
>>> tel['irv'] = 4127
>>> tel
{'jack': 4098, 'guido': 4127, 'irv': 4127}
>>> list(tel)
['jack', 'guido', 'irv']
>>> sorted(tel)
['guido', 'irv', 'jack']
>>> 'guido' in tel
True
>>> 'jack' not in tel
False
```

# 1: FINDING COMMAND

## Objective 3: Explore set() and its methods

```
>>> basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}
>>> print(basket)                      # show that duplicates have been removed
{'orange', 'banana', 'pear', 'apple'}
>>> 'orange' in basket                 # fast membership testing
True
>>> 'crabgrass' in basket
False

>>> # Demonstrate set operations on unique letters from two words
...
>>> a = set('abracadabra')
>>> b = set('alacazam')
>>> a                                  # unique letters in a
{'a', 'r', 'b', 'c', 'd'}
>>> a - b                              # letters in a but not in b
{'r', 'd', 'b'}
>>> a | b                              # letters in a or b or both
{'a', 'c', 'r', 'd', 'b', 'm', 'z', 'l'}
>>> a & b                              # letters in both a and b
{'a', 'c'}
>>> a ^ b                              # letters in a or b but not both
{'r', 'd', 'b', 'm', 'z', 'l'}
```

# METHODS COMMAND

Objective 4: Create the List,
Dict and Sets

- Always assign what ever you create to a variable.
- Think what in the real world can be represented with the data structure
- Try printing out and see whether you can understand the output

**Lets go to the Terminal**

## ds_file.py

```
#!/usr/bin/env python
#As shown in the video use the
#commands and create your script and
#execute
```

## COMMAND LINE

```
python ds_file.py
```

# WE HAVE DATA STRUCTURES, LETS DO FILES!!!!

Any Questions...