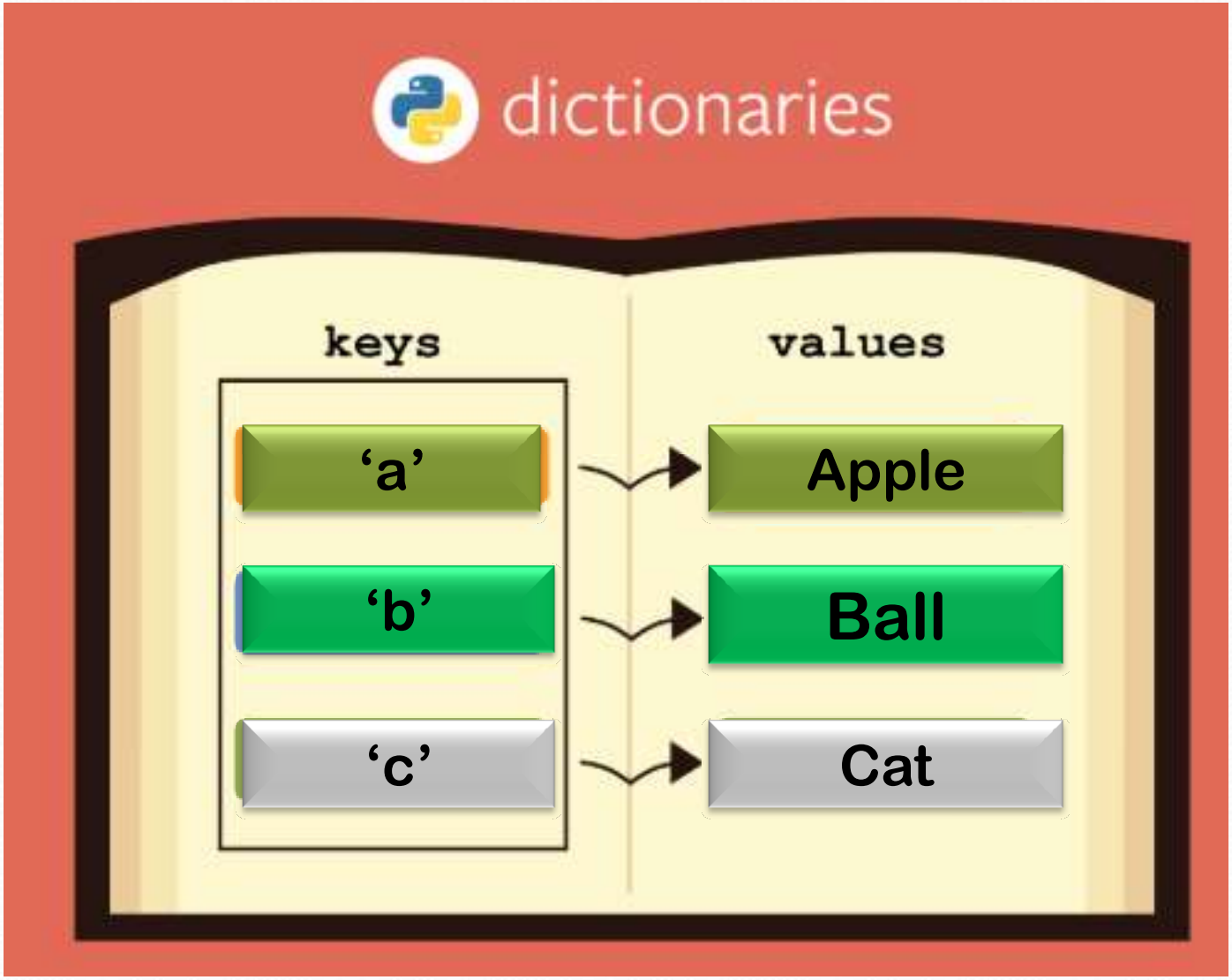
Python Collections (containers or Arrays)

There are four collection data types in the Python programming language, They are

- **List** is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- **Set** is a collection which is unordered and unindexed. No duplicate members.
- **Dictionary** is a collection which is unordered, changeable and indexed. No duplicate members.

When choosing a collection type, it is useful to understand the properties of that type. Choosing the right type for a particular data set could mean retention of meaning, and, it could mean an increase in efficiency or security.


dictionaries
keys
values
‘a’
Apple
‘b’
Ball
‘c’
Cat

Python dictionary

```
alpha={'a':'apple','b':'ball','c':'cat','d':'dog'}
```

Dictionary with mixed keys

```
d={1:'carrots','two':[1,2,3]}
```

```
a=[[1,2],[3,4]]
print(dict(a)) #{1: 2, 3: 4}
```

```
alpha={'a':'apple','b':'ball','c':'cat','d':'dog'}
print(alpha)
```

```
alpha={
        'a':'apple',
        'b':'ball',
        'c':'cat',
        'd':'dog'
    }
print(alpha)
```

```
print(alpha['a'])
print(alpha['b'])
print(alpha['c'])
```

Accessing/printing dictionary values

```
alpha={'a':'apple','b':'ball','c':'cat','d':'dog‘}

print(alpha) # print the whole dictionary

print(alpha.keys()) # print only the keys

print(alpha.values()) # print only values
```

```
# iterate over dictionary keys
for i in alpha.keys() :
    print(i)

# iterate over dictionary values
for i in alpha.values() :
    print(i)
```

Accessing/printing dictionary values

```
alpha={'a':'apple','b':'ball','c':'cat','d':'dog‘}

# iterate over dictionary elements (keys)
for i in alpha :
    print(i)

# iterate over dictionary  elements(values)
for i in alpha:
      print(alpha[i])

#print both keys and values
for i in alpha :
     print(i, "for", alpha[i])
```

Operations on dictionary

```
alpha={'a':'apple','b':'ball','c':'cat','d':'dog‘}

# check if key exist
print('a' in alpha)

# delete the key-value pair
del alpha['d']

print(alpha)
```

```
d={'a':'apple','b':'ball','c':'cat','d':'dog'}
x={'e':'eagle'}
#print(d+x)  #error
d.update(x)
print(d)
#{'a': 'apple', 'b': 'ball', 'c': 'cat', 'd': 'dog', 'e': 'eagle'}
```

```
d={'e':'elephant'}
x={'e':'eagle'}
print(d==x) #False
#based on both keys  and values
```

Functions on Dictionary

```
d={'a':'apple','b':'ball','c':'cat','d':'dog'}
print(len(d)) #4
print(max(d))#d
print(min(d)) #a
print(type(d)) #dict
print(sorted(d)) #on keys ['a', 'b', ‘c', 'd']
print(sorted(d.values()))
x=str(d)
print(x[0],x[1],x[2]) #{ ' a
print(sum(d))  # error
```

Methods of dictionary

Dict_object.method()

Dict.clear(): Removes all elements of dictionary dict

Dict.copy(): Returns a copy of dictionary dict

Dict.get(key): returns the value or none if key not in dictionary

Dict.items(): Returns a list of dict's (key, value) tuple pairs

Dict.keys(): Returns list of dictionary dict's keys

Dict.update(): Adds dictionary dict2's key-values pairs to dict

Dict.values(): Returns list of dictionary dict's values

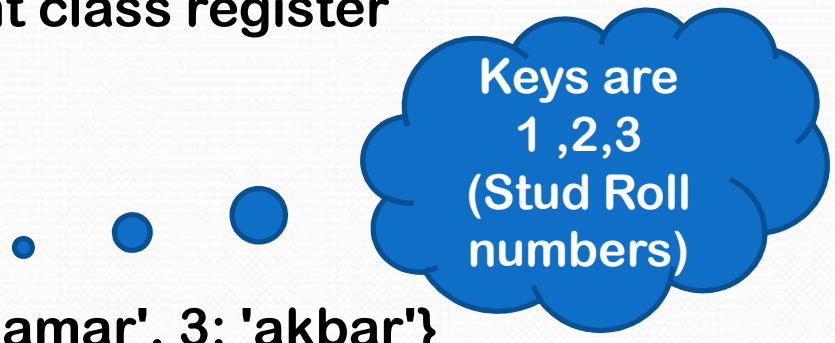
Methods of dictionary

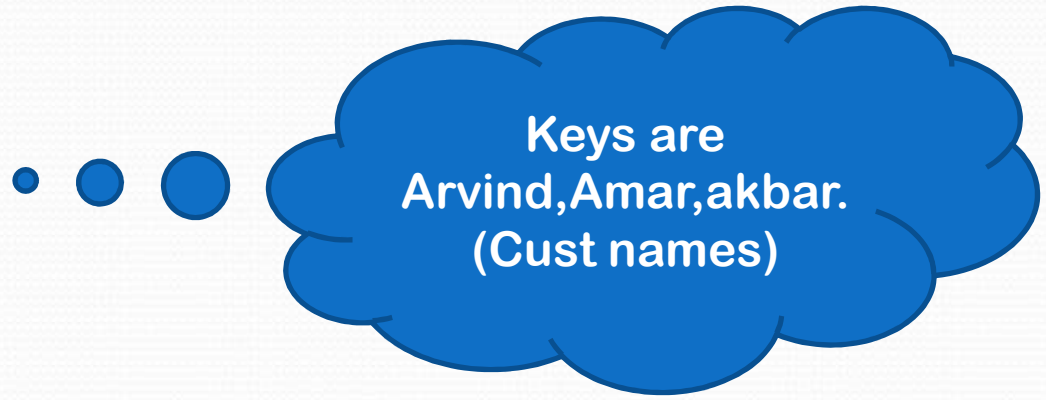
```
d={'a':'apple','b':'ball','c':'cat','d':'dog'}
print(d.keys()) #dict_keys(['a', 'b', 'c', 'd'])
print(d.items()) #dict_items([('a', 'apple'), ('b', 'ball'), ('c', 'cat'), ('d', 'dog')])
print(d.values()) #dict_values(['apple', 'ball', 'cat', 'dog'])
x=d.copy()
print(x)# {'a': 'apple', 'b': 'ball', 'c': 'cat', 'd': 'dog'}
d.update({'e':'eagle'})
print(d) #{'a': 'apple', 'b': 'ball', 'c': 'cat', 'd': 'dog', 'e': 'eagle'}
d.clear()
print(d) #{}
```

Creating and updating dictionary values

```
# Dictionary of a student class register
d={}
d[1]='arvind'
d[2]='amar'
d[3]='akbar'
print(d) #{1: 'arvind', 2: 'amar', 3: 'akbar'}

#dict of a bank with custname and balance
d={}
d['arvind']=4000
d['amar']=5000
d['akbar']=3000
print(d)
 #updating values of dict
d['amar']=d['amar']+2000
print(d)
```





Finding frequency of all words

```
d={‘happy':1,felt':1,‘saw':1}
w=‘happy'
#w=‘because'
if w in d:
        d[w]=d[w]+1
else:
       d[w]=1
print(d)
……................
```

s=" I felt happy because I saw the others were happy and because I knew I should feel happy, but I wasn’t really happy”

Finding frequency of all words

```
s = """betsy bottom bought some butter but the butter was bitter
    betsy bottom bought some better  butter to make the bitter
    butter  better"""“
```

```
d = {}
for w in s.split():
        if w in d:
            d[w] += 1
        else:
            d[w] =1
print(d)
```

```
import collections
w=s.split()
ctr=collections.Counter(w)
print(ctr)
```

```
d.update({w:1})
```

Composite/compound key

```
score = {}
#k1 = [ 'sunil', 'gavaskar' ]
#k2 = [ 'rohan', 'gavaskar' ]
# key cannot be a list; key should be immutable;
k1 = ( 'sunil', 'gavaskar' )
k2 = ( 'rohan', 'gavaskar' )
score = { k1  : 10000, k2 : 1000 }
print(score)
```

Index of the text book

Index

fluid

inco

nvolution filter

ass filter

see FVM

Airy wa

algorith

amortize

amplitu

angular

approxi

n-uniform Fourier

Bessel fu

zeroth order of the first kind, 30

Fourier trans-

form

Preparing index

```
s = "betsy bottom bought some butter but the butter
   was bitter betsy bottom bought some better  butter
   to make the bitter butter better”
d = {}
i=1
for w in s.split():
   if w in d:
         d[w].append(i)
   else:
        d[w] =[i]
   i=i+1
print(d)
```

```
for i in d:
    print(i,d[i])
```

```
x=sorted(d)
for i in x:
   print(i,d[i])
```

Nested structures

KA	PESIT	CSE
AP	GVR	CSE
KA	BIT	CSE
KA	PESIT	ECE
AP	ASN	ECE
KA	BIT	ISE
KA	BIT	ME
KA	BIT	IT
KA	AIT	CSE
KA	AIT	ISE
KA	AIT	ECE

KA
- PESIT
  - CSE
  - ECE
- BIT
  - CSE
  - ISE
  - ME

AP
- GVR
  - CSE
- ASN
  - ECE

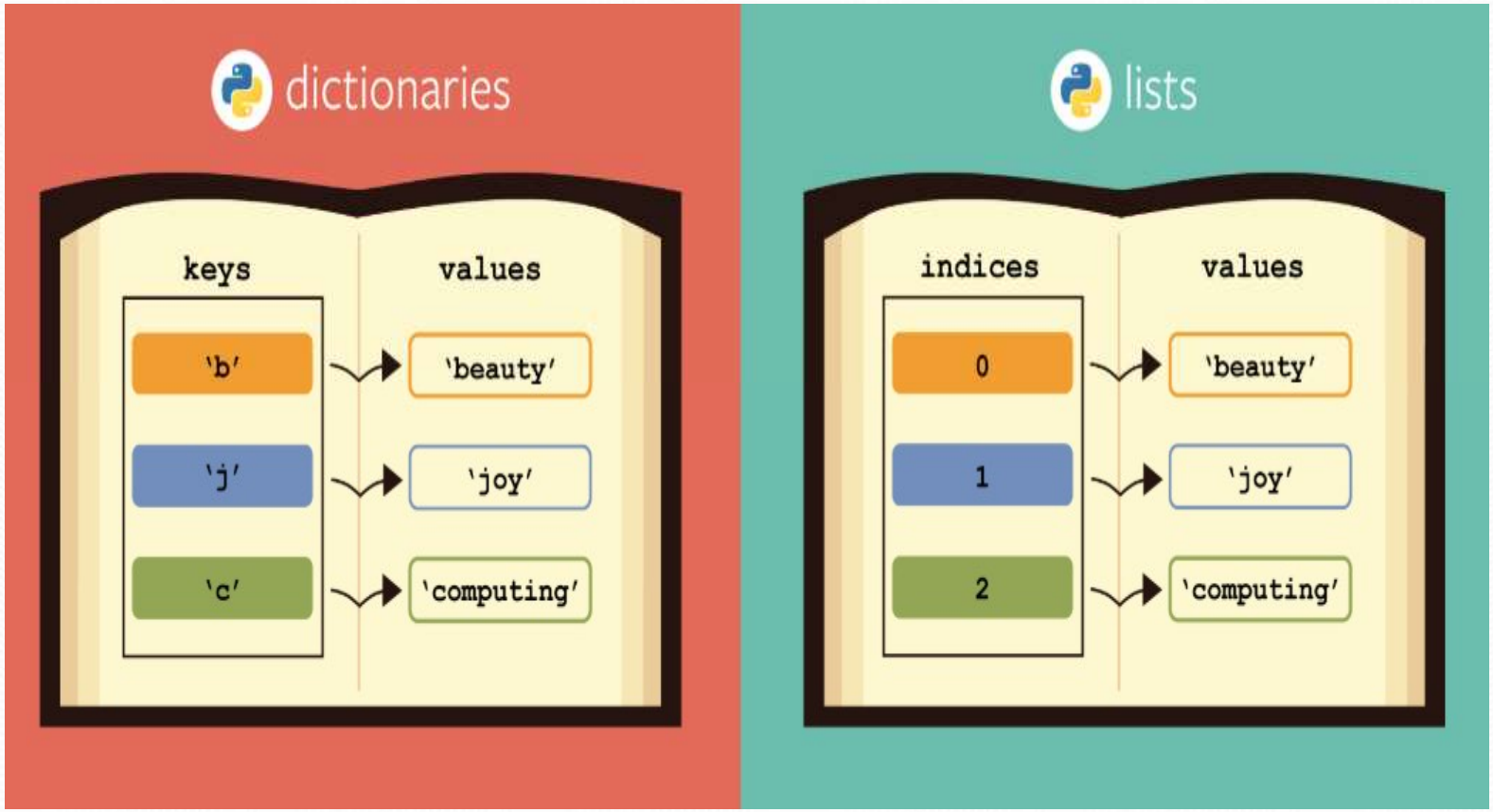
Nested structures

KA	PESIT	CSE
AP	GVR	CSE
KA	BIT	CSE
KA	PESIT	ECE
AP	ASN	ECE
KA	BIT	ISE
KA	BIT	ME
KA	BIT	IT
KA	AIT	CSE
KA	AIT	ISE
KA	AIT	ECE

KA
- PESIT
  - CSE
  - ECE
- BIT
  - CSE
  - ISE
  - ME

AP
- GVR
  - CSE
- ASN
  - ECE

soln: dict of dict of list


dictionaries
keys
values
'b'
'beauty'
'j'
'joy'
'c'
'computing'
lists
indices
values
0
'beauty'
1
'joy'
2
'computing'

Sorted function

```
# List
x = ['q', 'w', 'r', 'e', 't', 'y']
print sorted(x)

# Tuple
x = ('q', 'w', 'e', 'r', 't', 'y')
print sorted(x)

# String-sorted based on ASCII translations
x = "python"
print sorted(x)

# Dictionary
x = {'q':1, 'w':2, 'e':3, 'r':4, 't':5, 'y':6}
print sorted(x)

# Set
x = {'q', 'w', 'e', 'r', 't', 'y'}
print sorted(x)

# Frozen Set
x = frozenset(('q', 'w', 'e', 'r', 't', 'y'))   #immutable
print sorted(x)
```