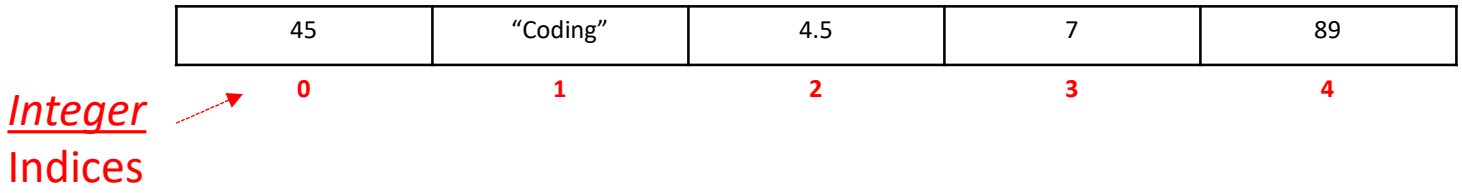
Dictionaries

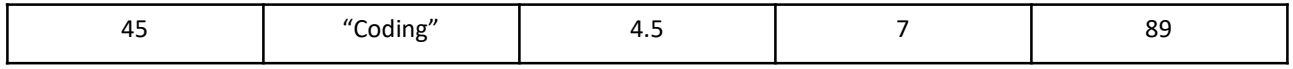
- Lists, tuples, and strings hold elements with *only integer* indices



- In essence, each element has an *index* (or a *key*) which can *only* be an integer, and a *value* which can be of any type (e.g., in the above list/tuple, the first element has key 0 and value 45)
  - *What if we want to store elements with non-integer indices (or keys)?*

Dictionaries

- In Python, you can use a dictionary to store elements with *keys of any hashable types* (e.g., integers, floats, Booleans, strings, and tuples; but not lists and dictionaries themselves) and *values of any types*



keys of different types

Values of different types

- The above dictionary can be defined in Python as follows:

```
dic = {"NUM":45, 1000:"coding", 2000:4.5, 3.4:7, "XXX":89}
```

key *value*

Each element is a *key*:*value* pair, and elements are separated by commas

Dictionaries

- To summarize, dictionaries:
  - Can contain any and different types of elements (i.e., hashable keys & values)
  - Can contain only *unique* keys but duplicate values

```
dic2 = {"a":1, "a":2, "b":2}
print(dic2)
```

Output: {'a': 2, 'b': 2}

The element “a”:2 will override the element “a”:1 because only ONE element can have key “a”

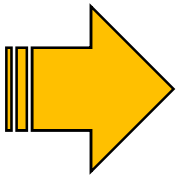
  - Can be indexed *but only* through keys (i.e., dic2[“a”] will return 1 but dic2[0] will return an error since there is no element with key 0 in dic2)

Dictionaries

- To summarize, dictionaries:
  - CANNOT be concatenated
  - Can be nested (e.g., d = {"first":{1:1}, "second":{2:"a"}}
  - Can be passed to a function and will result in a *pass-by-reference* and not *pass-by-value* behavior since they are *mutable* (similar to lists)

```
def func1(d):
    d["first"] = [1, 2, 3]

dic = {"first":{1:1},
"second":{2:"a"}}
print(dic)
func1(dic)
print(dic)
```



Output:

{'first': {1: 1}, 'second': {2: 'a'}}
{'first': [1, 2, 3], 'second': {2: 'a'}}

Dictionaries

- To summarize, dictionaries:
  - Can be iterated or looped over

```
dic = {"first": 1, "second": 2, "third": 3}
for i in dic:
    print(i)
```

Output:

first
second
third

ONLY the keys will be returned.

How to get the values?

Dictionaries

- To summarize, dictionaries:
  - Can be iterated or looped over

```
dic = {"first": 1, "second": 2, "third": 3}
for i in dic:
    print(dic[i])
```

Output:

1
2
3

Values can be accessed via indexing!

Adding Elements to a Dictionary

- How to add elements to a dictionary?
  - By indexing the dictionary via a key and assigning a corresponding value

```
dic = {"first": 1, "second": 2, "third": 3}
print(dic)
dic["fourth"] = 4
print(dic)
```

Output:

```
{'first': 1, 'second': 2, 'third': 3}
{'first': 1, 'second': 2, 'third': 3, 'fourth': 4}
```

Adding Elements to a Dictionary

- How to add elements to a dictionary?
  - By indexing the dictionary via a key and assigning a corresponding value

```
dic = {"first": 1, "second": 2, "third": 3}
print(dic)
dic[”second"] = 4
print(dic)
```

If the key already exists, the value will be overridden

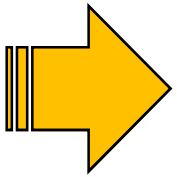
Output:

{'first': 1, 'second': 2, 'third': 3}
{'first': 1, 'second’: 4, 'third': 3}

Deleting Elements to a Dictionary

- How to delete elements in a dictionary?
  - By using **del**

```
dic = {"first": 1, "second": 2, "third": 3}
print(dic)
dic["fourth"] = 4
print(dic)
del dic["first"]
print(dic)
```



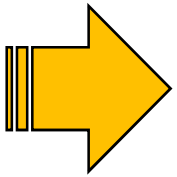
Output:

```
{'first': 1, 'second': 2, 'third': 3}
{'first': 1, 'second': 2, 'third': 3, 'fourth': 4}
{'second': 2, 'third': 3, 'fourth': 4}
```

Deleting Elements to a Dictionary

- How to delete elements in a dictionary?
  - Or by using the function **pop(key)**

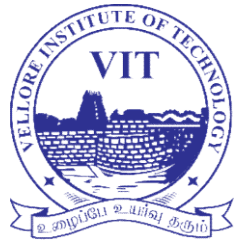
```
dic = {"first": 1, "second": 2, "third":
3}
print(dic)
dic["fourth"] = 4
print(dic)
dic.pop("first")
print(dic)
```



Output:

{'first': 1, 'second': 2, 'third': 3}
{'first': 1, 'second': 2, 'third': 3, 'fourth': 4}
{'second': 2, 'third': 3, 'fourth': 4}

Dictionary Functions



- Many other functions can also be used with dictionaries

Function	Description
dic.clear()	Removes all the elements from dictionary dic
dic.copy()	Returns a copy of dictionary dic
dic.items()	Returns a list containing a tuple for each key-value pair in dictionary dic
dic.get(k)	Returns the value of the specified key k from dictionary dic
dic.keys()	Returns a list containing all the keys of dictionary dic
dic.pop(k)	Removes the element with the specified key k from dictionary dic

Dictionary Functions

- Many other functions can also be used with dictionaries

Function	Description
dic.popitem()	Removes the last inserted key-value pair in dictionary dic
dic.values()	Returns a list of all the values in dictionary dic