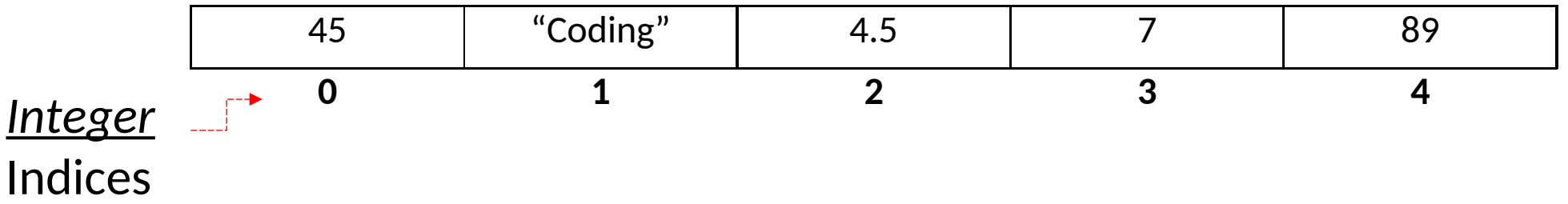
Programming with Python
Part 4: Dictionaries

Dictionaries

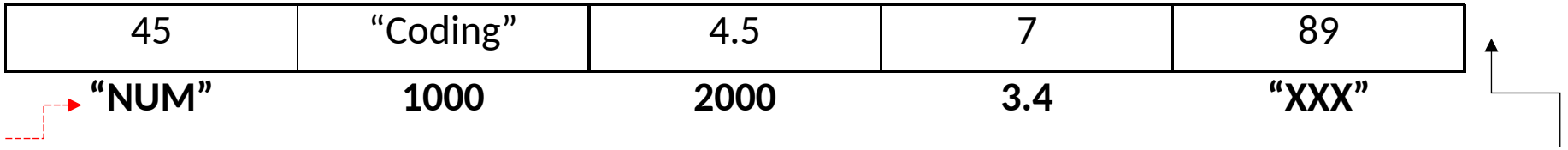
- Lists, tuples, and strings hold elements with *only integer* indices



- 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: A mapping Type

- Dictionaries store a mapping between a set of keys and a set of values
- Keys can be any immutable type.
- Values can be any type
- A single dictionary can store values of different types
- You can define, modify, view, lookup or delete the key-value pairs in the dictionary
- Python’s dictionaries are also known as hash tables and associative arrays

Creating & accessing dictionaries

```
>>> d = {‘user’:‘bozo’, ‘pswd’:1234}
>>> d[‘user’]
‘bozo’
>>> d[‘pswd’]
1234
```

Adding Elements to a Dictionary

```
>>> d = {'user':'bozo', 'pswd':1234}
>>> d['user'] = 'clown'
>>> d
{'user':'clown', 'pswd':1234}
```

- Keys must be unique
- Assigning to an existing key replaces its value

```
>>> d['id'] = 45
>>> d
{'user':'clown', 'id':45, 'pswd':1234}
```

- Dictionaries are unordered
- New entries can appear anywhere in output

Adding Elements to a Dictionary

- By indexing the dictionary via a key and assigning a corresponding value

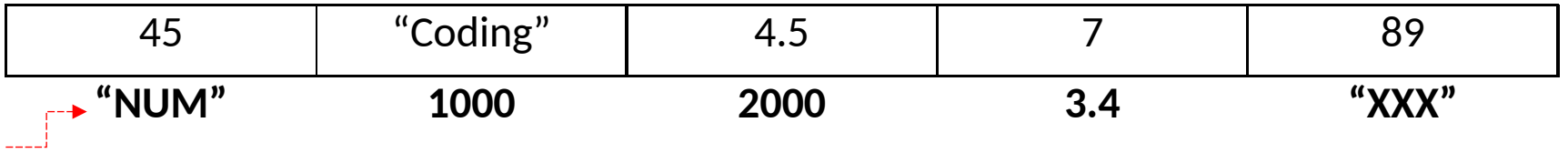
```
dic = {"first": 1, "second": 2, "third": 3}
print(dic)
dic["fourth"] = 4
print(dic)
```

Output:

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

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

Adding 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 from 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

- Exercise 1

- Convert two lists into a dictionary
- Below are the two lists. Write a Python program to convert them into a dictionary in a way that item from list1 is the key and item from list2 is the value

```
keys = ['Ten', 'Twenty', 'Thirty']

values = [10, 20, 30]
```

Hint:

Use the dictionary function **zip()**

Alternatively you can use a for loop with the dictionary function **update()**

Merge two Python dictionaries into one

- dict1 = {'Ten': 10, 'Twenty': 20, 'Thirty': 30}
- dict2 = {'Thirty': 30, 'Fourty': 40, 'Fifty': 50}

- Hint
  - Use the ** to unzip the dictionaries
  - Use update() function to add to the end of the dictionary