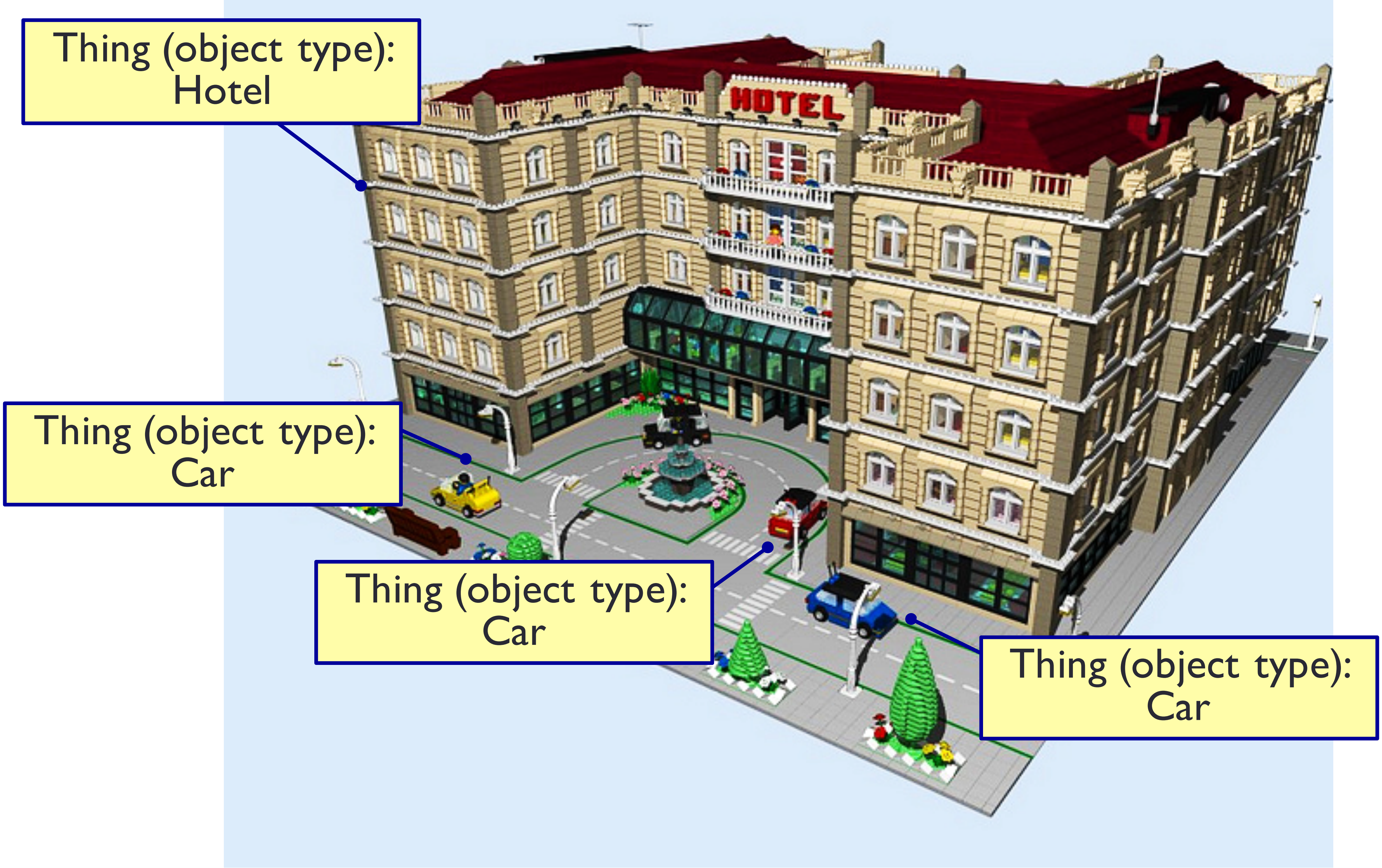
Dictionaries

CS 1111
Introduction to Programming

Spring 2019

How do Computer Programs Fit in with the World Around Them?



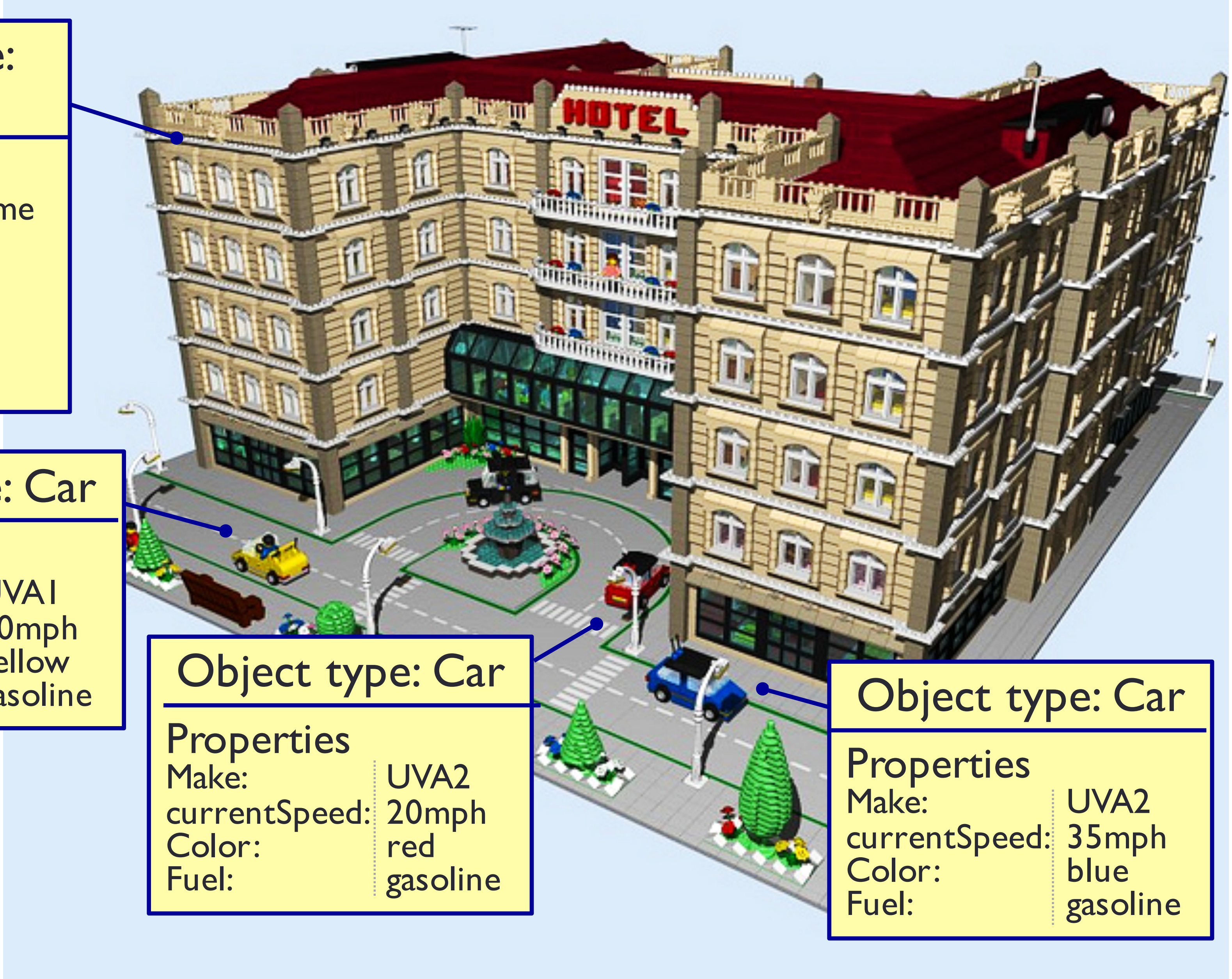
Thing (object type): Hotel

Thing (object type): Car

Thing (object type): Car

Thing (object type): Car

Objects and Properties



Object type: Hotel

Properties

Name:	Awesome
Rating:	5
Rooms:	70
Bookings:	56
Pool:	true
Gym:	true

Object type: Car

Properties

Make:	UVA1
currentSpeed:	30mph
Color:	yellow
Fuel:	gasoline

Object type: Car

Properties

Make:	UVA2
currentSpeed:	20mph
Color:	red
Fuel:	gasoline

Object type: Car

Properties

Make:	UVA2
currentSpeed:	35mph
Color:	blue
Fuel:	gasoline

Overview: Dictionaries

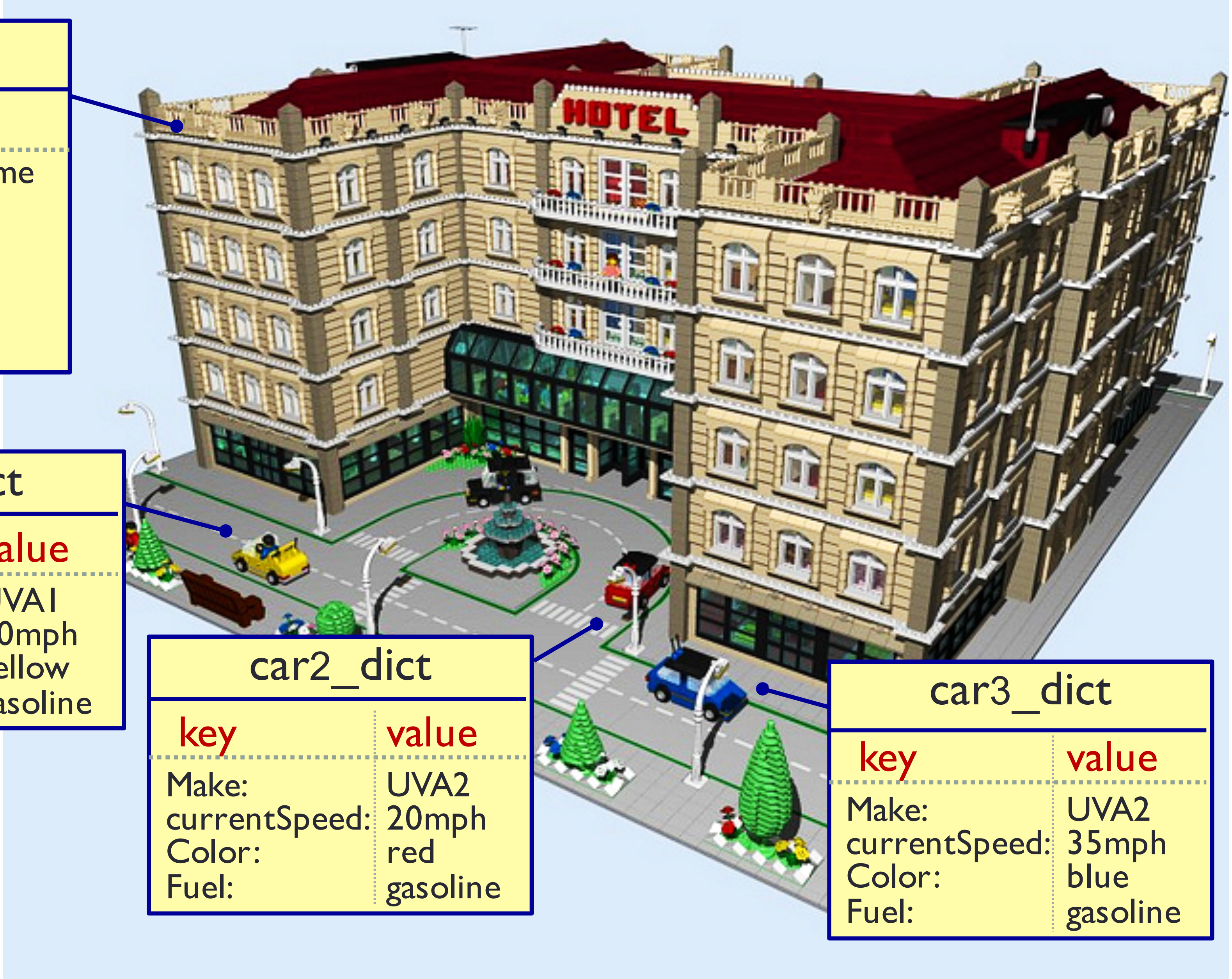
- Dictionary = unordered sequence of data
  - Python 3.6 remembers order of items in dictionary
- Mutable data type

- Each element in a dictionary consists of 2 parts: Key-value pair

- Key = index to locate a specific value
- Deterministic:
  - A particular key can only have one value

- Example
  - key = currentSpeed, value = 30mph
  - key = student ID, value = student name

Example: Dictionaries



hotel_dict

key	value
Name:	Awesome
Rating:	5
Rooms:	70
Bookings:	56
Pool:	true
Gym:	true

car1_dict

key	value
Make:	UVA1
currentSpeed:	30mph
Color:	yellow
Fuel:	gasoline

car2_dict

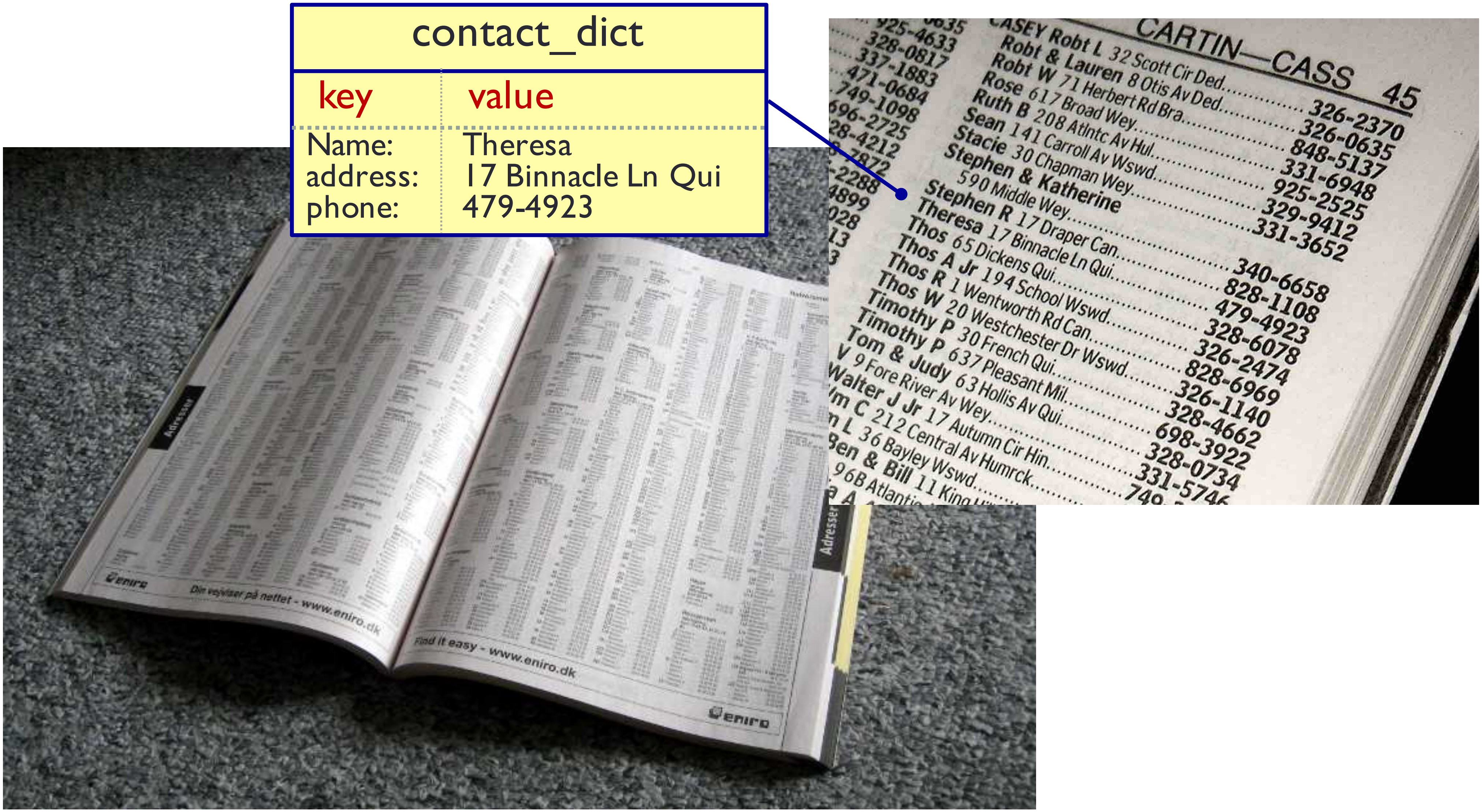
key	value
Make:	UVA2
currentSpeed:	20mph
Color:	red
Fuel:	gasoline

car3_dict

key	value
Make:	UVA2
currentSpeed:	35mph
Color:	blue
Fuel:	gasoline

Another Example

contact_dict	
key	value
Name:	Theresa
address:	17 Binnacle Ln Qui
phone:	479-4923



CARTIN—CASS 45

925-4633
328-0817
337-1883
471-0684
749-1098
696-2725
28-4212
-2288
4899
028

CASEY Robt L 32 Scott Cir Ded........ 326-2370
Robt & Lauren 8 Otis Av Ded........ 326-0635
Robt W 71 Herbert Rd Bra........ 848-5137
Rose 617 Broad Wey........ 331-6948
Ruth B 208 Atlntc Av Hul........ 925-2525
Sean 141 Carroll Av Wswd........ 329-9412
Stacie 30 Chapman Wey........ 331-3652
Stephen & Katherine
590 Middle Wey........ 340-6658
Stephen R 17 Draper Can........ 828-1108
Theresa 17 Binnacle Ln Qui........ 479-4923
Thos 65 Dickens Qui........ 328-6078
Thos A Jr 194 School Wswd........ 326-2474
Thos R 1 Wentworth Rd Can........ 828-6969
Thos W 20 Westchester Dr Wswd........ 326-1140
Timothy P 30 French Qui........ 328-4662
Timothy P 637 Pleasant Mil........ 698-3922
Tom & Judy 63 Hollis Av Qui........ 328-0734
V 9 Fore River Av Wey........ 331-5746
Walter J Jr 17 Autumn Cir Hin........ 749-
Wm C 212 Central Av Humrck........
Wm L 36 Bayley Wswd........
Ben & Bill 11 King
96B Atlantic

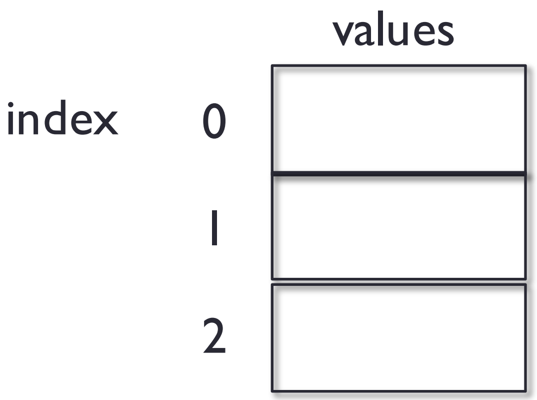
Adresser
eniro
Din vejviser på nettet - www.eniro.dk
Find it easy - www.eniro.dk
eniro
Adresser

[Images from https://en.wikipedia.org/wiki/Telephone_directory]

Lists vs. Dictionaries

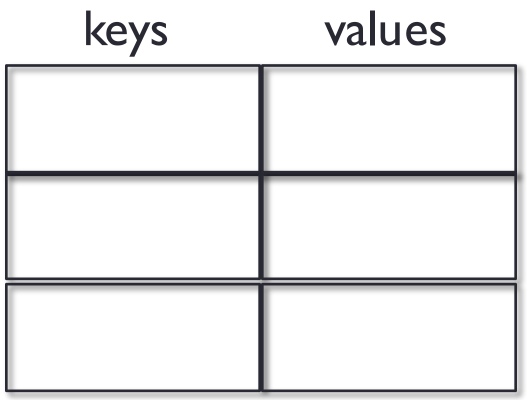
Lists

- Complex type
- Mutable
- Ordered sequence of data
- Index = 0, 1, 2, …



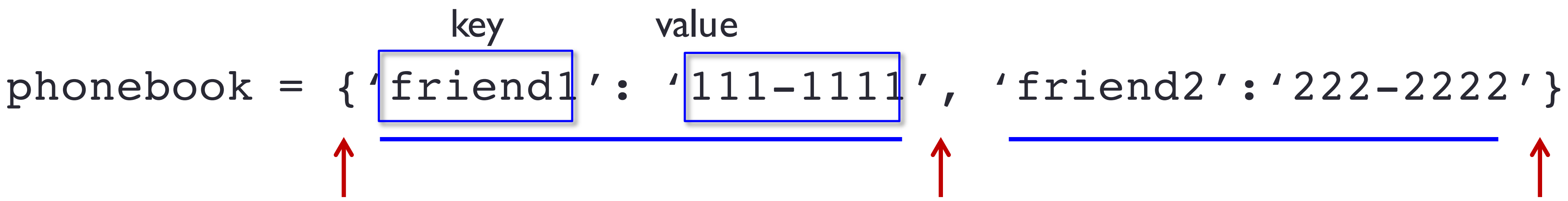
Dictionaries

- Complex type
- Mutable
- Unordered sequence of data (until Python 3.6),
- Index = *user-defined key*
- Unique key



Dictionaries

Create a dictionary



```
phonebook = {‘friend1’: ‘111-1111’, ‘friend2’:‘222-2222’}
```

```
phonebook2 = { }
```

Empty dictionary

keys	values
‘friend1’	‘111-1111’
‘friend2’	‘222-2222’

Exercise: Create Dictionary with {}

- Create a dictionary of a "friend" object.

- You will start by getting inputs from 5 friends (neighbors). Inputs contain
  - Name
  - Email address

- Use { } to create a "friends" dictionary with the information you gathered

- Print the dictionary content using

```
print(your-dictionary-name)
```

Reminder: Dictionary does not allowed duplicate key

Access items from a Dictionary

Retrieve a value from a dictionary

```
phonebook[‘friend1’]
```

Include quotations for string keys

```
Dictionary_name[key]
```

What would happen if we try to access a key that does not exist?

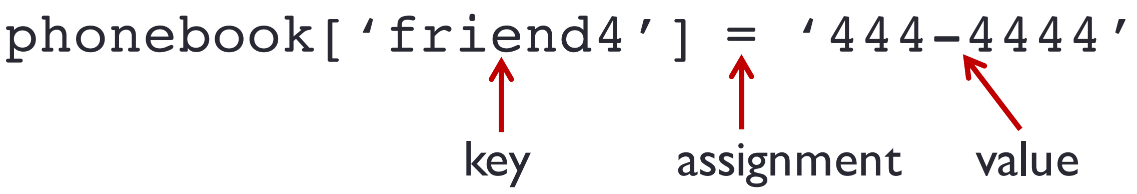
Exercise: Access Items with `[key]`

- Revisit your “friends” dictionary
- Access 2 friends and print their email addresses
- Try accessing a friend who is not in the dictionary and observe what happens
- Print the dictionary content using

```
print(your-dictionary-name)
```

Add Items to a Dictionary

```
phonebook = {‘friend1’: ‘111-1111’, ‘friend2’:‘222-2222’,
                ‘friend3’: ‘333-3333’}
```



```
phonebook[‘friend4’] = ‘444-4444’
```

```
phonebook[‘friend1’] = ‘555-5555’
```

keys	values
‘friend1’	‘555-5555’
‘friend2’	‘222-2222’
‘friend3’	‘333-3333’
‘friend4’	‘444-4444’

```
Dictionary_name[key] = value
```

- No duplicate keys in a dictionary
- When you assign a value to an existing key, the new value replaces the existing value

Exercise: Add Items with `[key]`

- Revisit your “friends” dictionary

- Add 2 more friends and their email addresses to the dictionary

- Try adding one more friend with the key already in the dictionary and observe what happens (… reassign the value)

- Print the dictionary content using

```
    print(your-dictionary-name)
```

Delete Items from Dictionaries

```
del phonebook[‘friend1’]
```

key

del deletes an element at a particular position

```
phone_number = phonebook.pop(‘friend1’)
```

key

pop() gets a value (and use it somewhere else), and deletes an element (a key/value pair)

What would happen if we try to delete an item with an index that doesn’t exist?

Exercise: Remove Item with `del` and `pop()`

- Revisit your "friends" dictionary
- Remove one friend from the dictionary, using `del`
- Print the dictionary content using

  `print(`*`your-dictionary-name`*`)`
- Try removing a friend whose name is not in the dictionary, using `del`, and observe what happens
- Remove one friend from the dictionary, using `pop()`
- Print the dictionary content using

  `print(`*`your-dictionary-name`*`)`
- Try removing a friend whose name is not in the dictionary, using `pop()`, and observe what happens

Length of Dictionaries

```
phonebook = {‘friend1’: ‘111-1111’,
             ‘friend2’: ‘222-2222’,
             ‘friend3’: ‘333-3333’}
num_items = len(phonebook)
```

`len()` is a function to return the length of a dictionary (i.e., the number of items in a dictionary)

Exercise: Get Size with `len(dict)`

- Revisit your “friends” dictionary
- Print the number of items of the dictionary
- Print the dictionary content using

```
print(your-dictionary-name)
```

Retrieve Values, Keys, or Items

```
# retrieve a value for a particular key
phonebook.get(”friend4”)

# access a non-existent key, set return value
phonebook.get(”friend99”, “friend99 does not exist”)

phonebook.items()   # retrieve all the keys and values
phonebook.keys()    # retrieve all the available keys
phonebook.values()  # retrieve all the values
```

`get(key, optional-msg)` gets a particular value based on key

`items()` gets all the keys and values

`keys()` gets all the keys

`values()` gets all the values

Exercise: Retrieve Value with `get()`

- Revisit your "friends" dictionary
- Print the dictionary content using

```
    print(your-dictionary-name)
```

- Retrieve an email address of one friend, using `get()`, and print it
- Try retrieving an email of a friend whose name is not in the dictionary, using `get()`, and observe what happens
- Try (again) retrieving an email of a friend whose name is not in the dictionary, using `get()`, set return value if the friend's name (key) is not found, and observe what happens

Exercise: Retrieve Items, Keys, Values

- Revisit your “friends” dictionary

- Print the dictionary content using

```
      print(your-dictionary-name)
```

- Retrieve all items from the dictionary using `items()`, and print them

- Retrieve all keys from the dictionary using `keys()`, and print them

- Retrieve all values from the dictionary using `values()`, and print them

Mix Data Types in Dictionaries

```
test_scores = {‘friend1’ : [88, 92, 100],
               ‘friend2’ : [95, 88, 81],
               ‘friend3’ : [70, 75, 78]}

print(test_scores)
print(‘friend2\’s scores: ’ + str(test_scores[‘friend2’]))
# why do we need str()?

friend3_scores = test_scores[’friend3’]
print(‘friend3\’s scores: ’ + str(friend3_scores))
```

Keys must be unique and immutable (primitive data type)

Values can be of any data types

Exercise: List in Dictionary

- You will now work with a dictionary that has mixed types of content.
- Gather some more information from friends. You will create a list of the information. Such as
  - List of email addresses, or
  - List of phone numbers, or
  - List of favorite cartoons (or movies), or
  - List of courses currently taken, or
  - List of anything you are interested to know about your friends
- Create a "favoritefriends" dictionary, using the friend's name as key and a list of the information you gather as value for that friend
- Print the dictionary
- Access 2 friends in the "favoritefriends" dictionary and print the corresponding values

in

```
phonebook = {‘friend1’: ‘111-1111’,
             ‘friend2’: ‘222-2222’,
             ‘friend3’: ‘333-3333’}

phonebook[‘upsorn’] = ‘444-4444’
print(‘upsorn’ in phonebook)
print(‘upsorn’ not in phonebook)
print(‘upsorn’ in phonebook.keys())
print(‘444-4444’ in phonebook.values())
```

`in` is a keyword and can be used to check if a particular item/key/value is in the dictionary/keys/values

Empty the Dictionaries

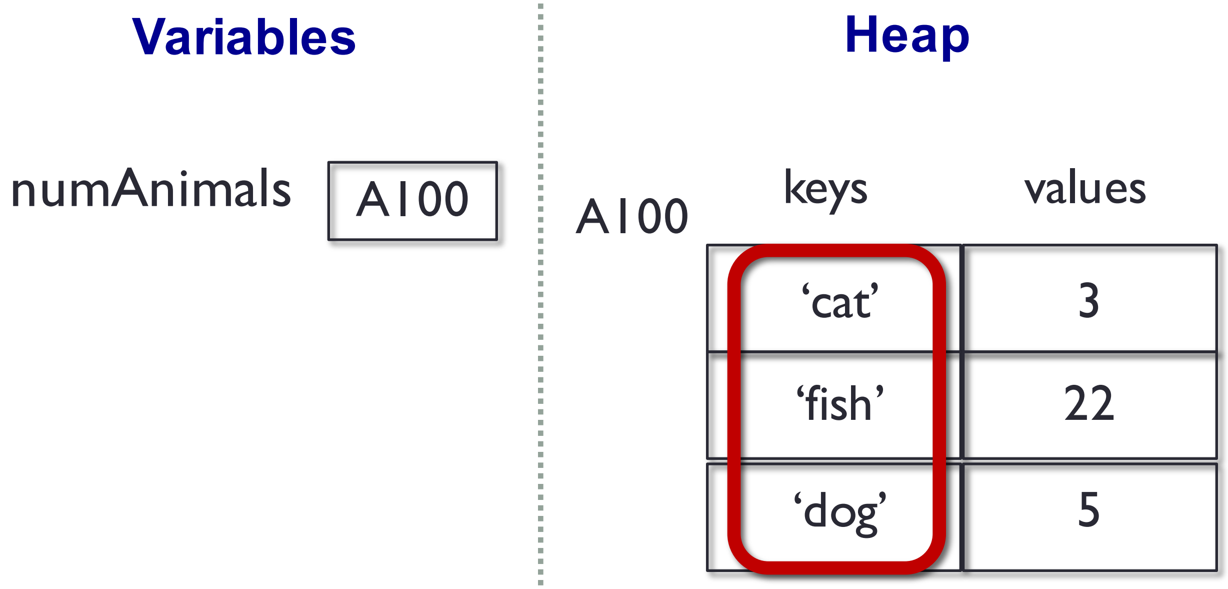
```
phonebook.clear()
```

`clear()` empties the dictionary

Tracing through Code with Dictionaries

Suppose we are using a dictionary to keep track of the number of animals in a small pet store

```
numAnimals = {}
numAnimals[‘cat’] = 3
numAnimals[‘fish’] = 22
numAnimals[‘dog’] = 5
```



Variables

numAnimals A100

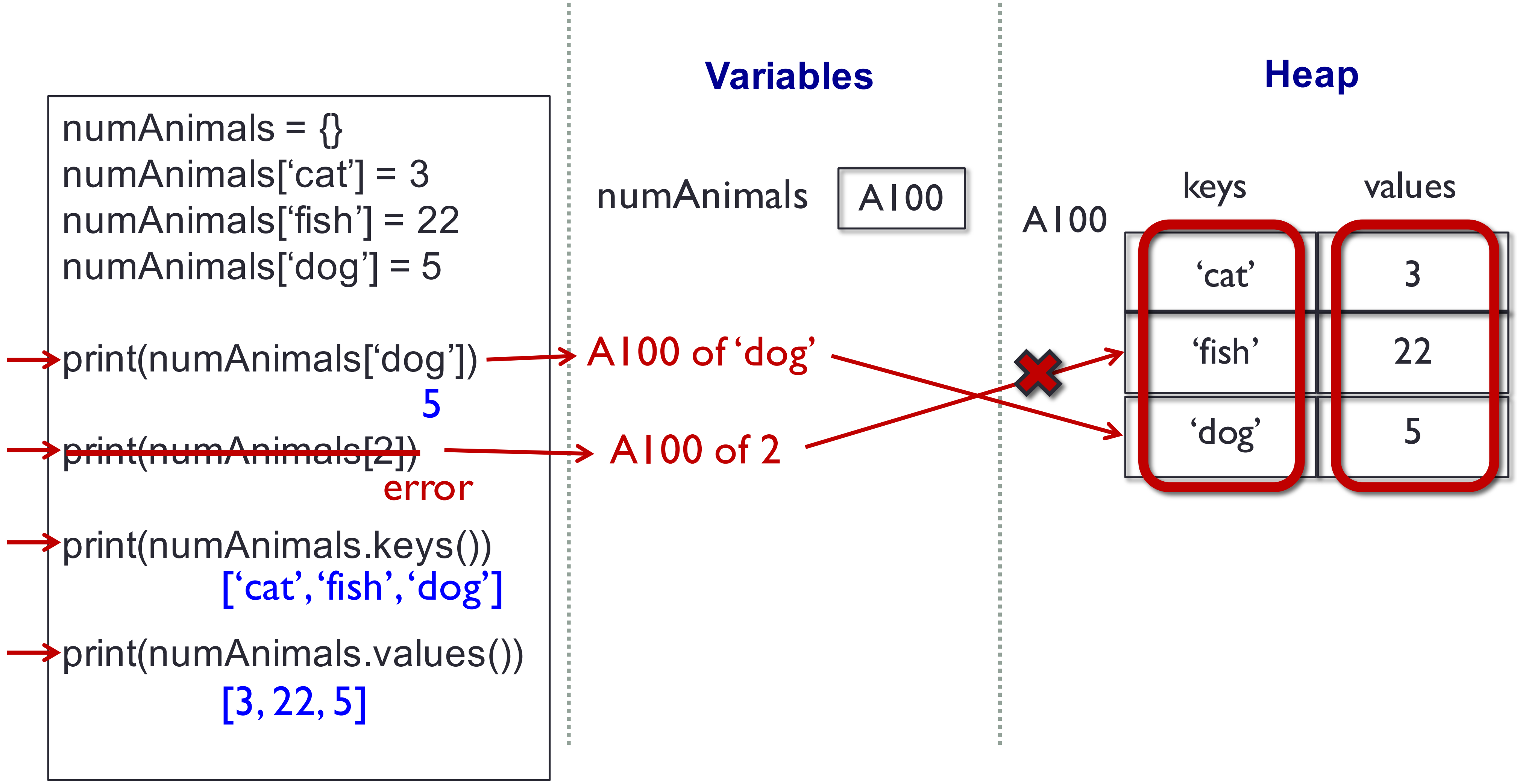
Heap

A100

keys	values
‘cat’	3
‘fish’	22
‘dog’	5

Tracing through Code with Dictionaries

Suppose we are using a dictionary to keep track of the number of animals in a small pet store



```
numAnimals = {}
numAnimals[‘cat’] = 3
numAnimals[‘fish’] = 22
numAnimals[‘dog’] = 5
```

print(numAnimals[‘dog’])
5

~~print(numAnimals[2])~~
error

print(numAnimals.keys())
[‘cat’, ‘fish’, ‘dog’]

print(numAnimals.values())
[3, 22, 5]

Variables

numAnimals A100

A100 of ‘dog’

A100 of 2

Heap

A100

keys	values
‘cat’	3
‘fish’	22
‘dog’	5

Tracing through Code with Dictionaries

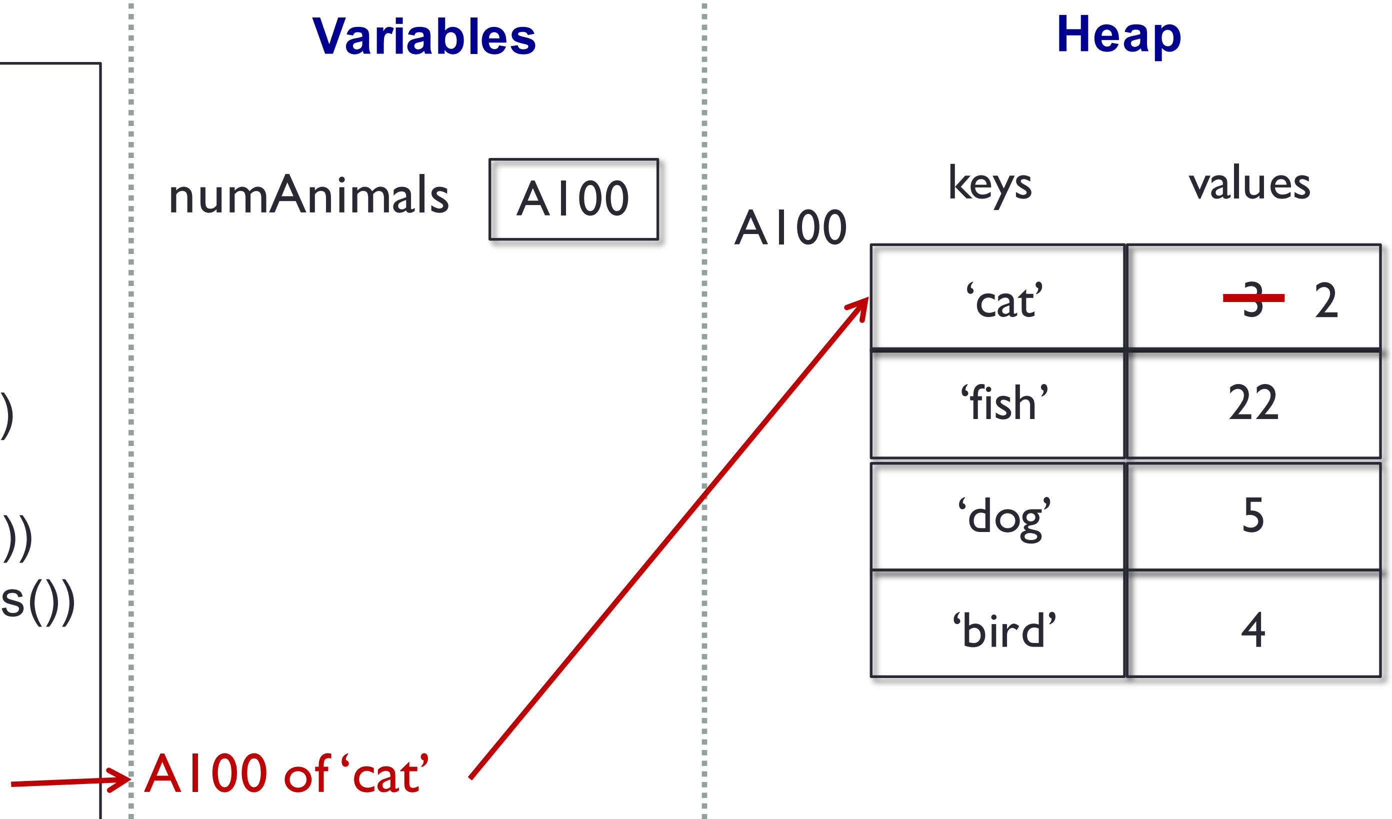
Suppose we are using a dictionary to keep track of the number of animals in a small pet store

```
numAnimals = {}
numAnimals['cat'] = 3
numAnimals['fish'] = 22
numAnimals['dog'] = 5

print(numAnimals['dog'])

print(numAnimals.keys())
print(numAnimals.values())

numAnimals['bird'] = 4
numAnimals['cat'] = 2
```



keys	values
'cat'	~~3~~ 2
'fish'	22
'dog'	5
'bird'	4

Dictionaries (wrap up)

```
dict = {}
dict[1] = ‘cat’
dict[‘dog’] = -8
dict[False] = ‘squirrel’
print(dict.keys())
print(dict.values())
print(dict)

if ‘dog’ in dict.keys():
    print(‘dog has a mapping!’)
if ‘cat’ in dict.keys():
    print(‘cat has a mapping!’)
dict[‘dog’] = 5
print(dict)
```

- declare a dictionary with curly braces
- add to a dict by specifying a key and assigning it a value
- **a key must be immutable (no lists)**
- the .keys() method returns all the keys (but we can’t rely on an order)
- the .values() method returns all the values (but we can’t rely on an order)
- assigning to a key that already has that value overwrites the old value

Exercise

- Create a dictionary of an “experience” object.
- You will start by getting inputs from users. Inputs contain
  - The name of the experience (e.g., "software engineer”)
  - The company of the experience (e.g., “IBM”)
  - The year of the experience (e.g., “1996”)
- Add the users’ inputs to an “experience” dictionary
  - The keys in the dictionary will be the year of the experience, while the values will be the name of the experience and the companies, stored as a list.
  - E.g., { ‘1996’ : [‘software engineer’, ‘IBM], ‘1993’ : [‘sale’, ‘Target’] }
- You should get at least 2 experience inputs from the users.
- Print each experience in a separate line
- You may assume that all experiences passed in as arguments never have two experiences with the same company and year.
- Try to add more actions: retrieve items, delete items, update items, …

Summary

- Must know (based on exam2 topic list, as of 03/17/2019)
  - `mapping.keys()`
  - `mapping.values()`
  - `mapping.items()`
  - `mapping.pop(key)`

(mapping refer to a variable of `dict` type)