ECMM455 Python Worksheet 16: Dictionaries

Prof Hywel Williams

September 12, 2019

1 Aims

- Practice using dictionaries

2 Further reading

Comprehensive information about dictionaries can be found in Chapter 11 of *Think Python*.

3 Dictionaries

Dictionaries are a distinctive and useful feature of Python. They are a collection of mappings from a key to a value. The dictionary is a named object that holds the collection of key-value pairs. Like a list, items in a dictionary can be accessed using square brackets. Unlike a list, the key/index can be almost any data type, commonly numbers or strings. The simple example below creates a dictionary called num_legs and stores in it a collection of key-value pairs where each key is a string denoting an animal and each value is a number representing how many legs that animal normally has.

```
num_legs = {}
num_legs['horse'] = 4
num_legs['snake'] = 0
num_legs['human'] = 2
num_legs['centipede'] = 'lots'
```

Note that the value associated with a key can be any data type. Once created, the values can be accessed (for reading or alteration) using square brackets:

```
>>> num_legs['human']
2
>>> num_legs['human'] = 1
>>> num_legs['human']
1
>>>
```

The contents of a dictionary can be displayed using the *print()* function. The format gives a comma-separated list of key:value pairs, enclosed in curly brackets:

```
>>> print(num_legs)
{’horse’: 4, ’snake’: 0, ’human’: 1, ’centipede’: ’lots’}
>>>
```

Lists of keys or values can be obtained using the *.keys()* and *.values()* methods:

```
>>> print(num_legs.keys())
dict_keys([’horse’, ’snake’, ’human’, ’centipede’])
>>> print(num_legs.values())
dict_values([4, 0, 1, ’lots’])
>>>
```

A for loop can be used to iterate through the items in a dictionary:

```
>>> for x in num_legs:
     ... print(x,num_legs[x])
horse 4
snake 0
human 1
centipede lots
>>>
```

Note that the counter *x* above is assigned to each key in turn.

4 Exercises

1. Write a Python program that creates and displays a dictionary in which the keys are the integers [1:*n*] for some *n* and each value is the square of its key. *Expected output for n=5 : {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}*

2. Write a Python program that creates and displays a dictionary in which the keys are the integers [1:*n*] for some *n* and each value is a list containing the square and the cube of its key. *Expected output for n=5: {1: [1,1], 2: [4,8], 3: [9,27], 4: [16,64], 5: [25,125]}*

3. Imagine you are a greengrocer whose inventory is recorded in the following two dictionaries: *stock = {”apple”: 5, “banana”: 3, “grape”: 430}* and *price = {”apple”: 1.50, “banana”: 2.10, “grape”: 0.25}*. Write a Python program that calculates the total value of your inventory if you were to sell it all.

4. Consider the greengrocer problem above. Could you represent the inventory as a single dictionary? (*Hint*: The value associated with a key can itself be a dictionary.)

5. Write a Python program to create a dictionary holding the frequencies of each letter in a given word. *Expected output for “mississippi”: {m: 1, i: 4, s: 4, p:2}*. (*Hint*: Use a for loop to traverse the string, store each letter as a key.)