**ANSWERS**

**Q1. Create a zoo.py file first. Define the hours() function, which prints the string 'Open 9-5 daily'. Then, use the interactive interpreter to import the zoo module and call its hours() function.**

**Ans.** To create the zoo.py file, you can use any text editor to create a new file with the following contents:

def hours():

print('Open 9-5 daily')

Save the file as zoo.py.

Then, in the interactive interpreter, you can import the zoo module and call its hours() function using the following code:

import zoo

zoo.hours()

**Q2. In the interactive interpreter, import the zoo module as menagerie and call its hours() function.**

**Ans.** In the interactive interpreter, you can import the **zoo** module as **menagerie** and call its **hours()** function using the following code:

import zoo as menagerie

menagerie.hours()

**Q3. Using the interpreter, explicitly import and call the hours() function from zoo.**

**Ans.** Using the interpreter, you can explicitly import and call the **hours()** function from **zoo** using the following code:

from zoo import hours

hours()

**Q4. Import the hours() function as info and call it.**

**Ans.** You can import the **hours()** function as **info** and call it using the following code:

from zoo import hours as info

info()

**Q5. Create a plain dictionary with the key-value pairs 'a': 1, 'b': 2, and 'c': 3, and print it out.**

**Ans.** Here's the code to create a plain dictionary and print it:

my\_dict = {'a': 1, 'b': 2, 'c': 3}

print(my\_dict)

Output:

{'a': 1, 'b': 2, 'c': 3}

**Q6.Make an OrderedDict called fancy from the same pairs listed in 5 and print it. Did it print in the same order as plain?**

**Ans.** Here's the code to create an **OrderedDict** called **fancy** from the same key-value pairs and print it:

from collections import OrderedDict

fancy = OrderedDict([('a', 1), ('b', 2), ('c', 3)])

print(fancy)

Output:

OrderedDict([('a', 1), ('b', 2), ('c', 3)])

Yes, the OrderedDict will print the items in the same order as they were added, unlike a regular dictionary.

**Q7. Make a default dictionary called dict\_of\_lists and pass it the argument list. Make the list dict\_of\_lists['a'] and append the value 'something for a' to it in one assignment. Print dict\_of\_lists['a'].**

**Ans.** Here's the code to create a **defaultdict** called **dict\_of\_lists** and append a value to the list under the key 'a':

from collections import defaultdict

dict\_of\_lists = defaultdict(list)

dict\_of\_lists['a'].append('something for a')

print(dict\_of\_lists['a'])

Output:

['something for a']

The defaultdict automatically initializes the value for the key 'a' as an empty list, allowing us to directly append the value without explicitly creating the list first.