1. 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: Let's create the `zoo.py` file first with the `hours()` function, and then we'll use the interactive interpreter to import the `zoo` module and call its `hours()` function.

Step 1: Create the `zoo.py` file

Open a text editor and create a new file named `zoo.py`. Inside the file, add the following code:

# zoo.py

def hours():

print('Open 9-5 daily')

Save the file.

Step 2: Use the interactive interpreter

Now, open the Python interactive interpreter or any Python IDE you are using. In the interactive interpreter, type the following:

import zoo

zoo.hours()

When you run the above code, it will import the `zoo` module and call its `hours()` function, which will print the message "Open 9-5 daily" to the console.

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

ANS: To import the `zoo` module as `menagerie` and call its `hours()` function in the interactive interpreter, follow these steps:

Step 1: Launch the Python interactive interpreter or any Python IDE.

Step 2: Import the `zoo` module as `menagerie`:

import zoo as menagerie

Step 3: Call the `hours()` function from the `menagerie` module:

```python

menagerie.hours()

When you run the above code, it will import the `zoo` module under the alias `menagerie` and call its `hours()` function, printing the message "Open 9-5 daily" to the console.By importing the module with an alias, you can use the chosen alias (`menagerie` in this case) to access the module's functions and variables, making your code more readable and concise.

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

ANS:To explicitly import and call the `hours()` function from the `zoo` module in the Python interactive interpreter, follow these steps:

Step 1: Launch the Python interactive interpreter or any Python IDE.

Step 2: Explicitly import the `hours()` function from the `zoo` module:

from zoo import hours

Step 3: Call the `hours()` function directly:

hours()

When you run the above code, it will import only the `hours()` function from the `zoo` module, and then call the function, printing the message "Open 9-5 daily" to the console.

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

ANS: Sure! To import the `hours()` function from the `zoo` module as `info` and call it in the Python interactive interpreter, follow these steps:

Step 1: Launch the Python interactive interpreter or any Python IDE.

Step 2: Import the `hours()` function from the `zoo` module as `info`:

from zoo import hours as info

Step 3: Call the `info()` function directly:

info()

When you run the above code, it will import the `hours()` function from the `zoo` module with the alias `info`, and then call the function, printing the message "Open 9-5 daily" to the console.

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

ANS: Sure! Here's how you can create a plain dictionary with the key-value pairs 'a': 1, 'b': 2, and 'c': 3, and then print it out:

# Create the dictionary

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

# Print the dictionary

print(plain\_dict)

When you run this code, it will create the dictionary with the specified key-value pairs and then print it to the console:

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

The dictionary contains three key-value pairs: 'a': 1, 'b': 2, and 'c': 3. Each key is associated with its corresponding value.

6.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: To create an OrderedDict named `fancy` with the same key-value pairs 'a': 1, 'b': 2, and 'c': 3 and then print it, you can use the `collections` module from the Python standard library. The OrderedDict will preserve the order of insertion of elements.

Here's the code to achieve that:

from collections import OrderedDict

# Create the OrderedDict 'fancy'

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

# Print the OrderedDict 'fancy'

print(fancy)

When you run this code, it will create the OrderedDict with the specified key-value pairs and then print it to the console:

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

The OrderedDict retains the order of insertion, so it will print in the same order as it was created. In this case, the order will be 'a': 1, 'b': 2, and 'c': 3, just like in the `plain\_dict`.

7. 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: To create a defaultdict called `dict\_of\_lists` and pass it the argument `list`, then append the value 'something for a' to the list associated with the key 'a' in one assignment, you can use the `collections` module from the Python standard library. The defaultdict automatically creates missing keys with the specified default value (in this case, an empty list) when you try to access them.

Here's the code to achieve that:

from collections import defaultdict

# Create the defaultdict 'dict\_of\_lists' with default value as list

dict\_of\_lists = defaultdict(list)

# Append the value 'something for a' to the list associated with the key 'a'

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

# Print the value associated with the key 'a'

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

When you run this code, it will create the defaultdict with the default value as an empty list, and then append 'something for a' to the list associated with the key 'a', and finally, it will print the list associated with the key 'a':

['something for a']

As you can see, the value 'something for a' has been appended to the list associated with the key 'a' in the `dict\_of\_lists`. If you try to access a key that does not exist, it will automatically create a new key-value pair with the default value (an empty list in this case).