1. Create a list called years\_list, starting with the year of your birth, and each year thereafter until the year of your fifth birthday. For example, if you were born in 1980. the list would be years\_list = [1980, 1981, 1982, 1983, 1984, 1985].

Ans : the years\_list in Python:

birth\_year = 1990 years\_list = list(range(birth\_year, birth\_year + 6)) print(years\_list)

Output:

[1990, 1991, 1992, 1993, 1994, 1995]

In this example, I assumed your birth year is 1990. You can replace it with your actual birth year, and the code will generate the **years\_list** accordingly.

2. In which year in years\_list was your third birthday? Remember, you were 0 years of age for your first year.

Ans : Since you mentioned that I was 0 years old for my first year, my third birthday would occur in the year following my birth year. Here's an example of how you can find the year of my third birthday in the years\_list:

pythonCopy code

birth\_year = 1990 years\_list = list(range(birth\_year, birth\_year + 6)) third\_birthday\_year = birth\_year + 3 print(third\_birthday\_year)

Output:

yamlCopy code

1993

So, in this example, my third birthday would be in the year 1993.

3.In the years list, which year were you the oldest?

Ans : To determine the year when I was the oldest in the years\_list, we need to find the maximum value from the list. Here's an example of how you can do that in Python:

pythonCopy code

birth\_year = 1990 years\_list = list(range(birth\_year, birth\_year + 6)) oldest\_year = max(years\_list) print(oldest\_year)

Output:

yamlCopy code

1995

In this example, the oldest year in the **years\_list** is 1995, which would be the year when I turned 5 years old.

4. Make a list called things with these three strings as elements: "mozzarella", "cinderella", "salmonella".

Ans : Certainly! Here's an example of how you can create the things list with the given strings as elements:

pythonCopy code

things = ["mozzarella", "cinderella", "salmonella"] print(things)

Output:

cssCopy code

['mozzarella', 'cinderella', 'salmonella']

In this example, I created a list called **things** and assigned the three strings "mozzarella", "cinderella", and "salmonella" as its elements.

5. Capitalize the element in things that refers to a person and then print the list. Did it change the element in the list?

Ans : To capitalize the element in the things list that refers to a person ("cinderella"), you can use the capitalize() method in Python. Here's an example:

pythonCopy code

things = ["mozzarella", "cinderella", "salmonella"] things[1] = things[1].capitalize() print(things)

Output:

cssCopy code

['mozzarella', 'Cinderella', 'salmonella']

In this example, I accessed the second element in the **things** list (index 1) and applied the **capitalize()** method to capitalize the string "cinderella". The updated list is then printed.

Yes, it changed the element in the list. The original element "cinderella" was replaced with "Cinderella" after applying the **capitalize()** method.

6. Make a surprise list with the elements "Groucho," "Chico," and "Harpo."

Ans : Certainly! Here's an example of how you can create the surprise list with the given elements:

pythonCopy code

surprise = ["Groucho", "Chico", "Harpo"] print(surprise)

Output:

cssCopy code

['Groucho', 'Chico', 'Harpo']

In this example, I created a list called **surprise** and assigned the elements "Groucho," "Chico," and "Harpo" to it.

7. Lowercase the last element of the surprise list, reverse it, and then capitalize it.

Ans : Certainly! Here's an example of how you can lowercase the last element of the surprise list, reverse it, and then capitalize it:

surprise = ["Groucho", "Chico", "Harpo"] last\_element = surprise[-1].lower() reversed\_element = last\_element[::-1] capitalized\_element = reversed\_element.capitalize() surprise[-1] = capitalized\_element print(surprise)

Output:

['Groucho', 'Chico', 'oprah']

In this example, I accessed the last element of the **surprise** list using the index -1 and converted it to lowercase using the **lower()** method. Then, I reversed the lowercase element using slicing notation **[::-1]**. Finally, I capitalized the reversed element using the **capitalize()** method and assigned the modified element back to the last position of the **surprise** list.

The updated **surprise** list is then printed, with the last element changed from "Harpo" to "oprah".

8. Make an English-to-French dictionary called e2f and print it. Here are your starter words: dog is chien, cat is chat, and walrus is morse.

Ans : Sure! Here's an example of how you can create the English-to-French dictionary called e2f and print it with the given starter words:

pythonCopy code

e2f = { 'dog': 'chien', 'cat': 'chat', 'walrus': 'morse' } print(e2f)

Output:

arduinoCopy code

{'dog': 'chien', 'cat': 'chat', 'walrus': 'morse'}

In this example, I created a dictionary called **e2f** with the English words as keys and their corresponding French translations as values. The dictionary is then printed, showing the mapping from English to French for each word.

9. Write the French word for walrus in your three-word dictionary e2f.

Ans : The French word for "walrus" in the e2f dictionary is "morse". Here's how you can access and print the French word for "walrus":

pythonCopy code

e2f = { 'dog': 'chien', 'cat': 'chat', 'walrus': 'morse' } french\_word\_for\_walrus = e2f['walrus'] print(french\_word\_for\_walrus)

Output:

Copy code

morse

In this example, I accessed the value associated with the key **'walrus'** in the **e2f** dictionary using square bracket notation (**e2f['walrus']**). The variable **french\_word\_for\_walrus** stores the French word "morse", and it is printed.

10. Make a French-to-English dictionary called f2e from e2f. Use the items method.

Ans : Certainly! You can create a French-to-English dictionary called f2e from the existing e2f dictionary using the items() method. Here's an example:

pythonCopy code

e2f = { 'dog': 'chien', 'cat': 'chat', 'walrus': 'morse' } f2e = {value: key for key, value in e2f.items()} print(f2e)

Output:

arduinoCopy code

{'chien': 'dog', 'chat': 'cat', 'morse': 'walrus'}

In this example, the dictionary comprehension **{value: key for key, value in e2f.items()}** is used to create the **f2e** dictionary. It iterates over the key-value pairs of the **e2f** dictionary using the **items()** method, swapping the keys and values to create the French-to-English mapping. The resulting **f2e** dictionary is then printed, showing the mapping from French to English for each word.

11. Print the English version of the French word chien using f2e.

Ans : To print the English version of the French word "chien" using the f2e dictionary, you can access the value associated with the key "chien". Here's an example:

pythonCopy code

f2e = { 'chien': 'dog', 'chat': 'cat', 'morse': 'walrus' } english\_word\_for\_chien = f2e['chien'] print(english\_word\_for\_chien)

Output:

Copy code

dog

In this example, I accessed the value associated with the key **'chien'** in the **f2e** dictionary using square bracket notation (**f2e['chien']**). The variable **english\_word\_for\_chien** stores the English word "dog", and it is printed.

12. Make and print a set of English words from the keys in e2f.

Ans : Certainly! To create a set of English words from the keys in the e2f dictionary and print it, you can use the set() function. Here's an example:

pythonCopy code

e2f = { 'dog': 'chien', 'cat': 'chat', 'walrus': 'morse' } english\_words\_set = set(e2f.keys()) print(english\_words\_set)

Output:

arduinoCopy code

{'dog', 'cat', 'walrus'}

In this example, I used the **keys()** method to retrieve the keys from the **e2f** dictionary. Then, I passed the keys to the **set()** function to create a set containing the English words. Finally, the set is printed, displaying the unique English words present as keys in the **e2f** dictionary.

13. Make a multilevel dictionary called life. Use these strings for the topmost keys: 'animals', 'plants', and 'other'. Make the 'animals' key refer to another dictionary with the keys 'cats', 'octopi', and 'emus'. Make the 'cats' key refer to a list of strings with the values 'Henri', 'Grumpy', and 'Lucy'. Make all the other keys refer to empty dictionaries.

Ans : Certainly! Here's an example of how you can create a multilevel dictionary called life according to the given specifications:

pythonCopy code

life = { 'animals': { 'cats': ['Henri', 'Grumpy', 'Lucy'], 'octopi': {}, 'emus': {} }, 'plants': {}, 'other': {} } print(life)

Output:

arduinoCopy code

{ 'animals': { 'cats': ['Henri', 'Grumpy', 'Lucy'], 'octopi': {}, 'emus': {} }, 'plants': {}, 'other': {} }

In this example, I created the **life** dictionary with the topmost keys 'animals', 'plants', and 'other'. The 'animals' key refers to another dictionary with the keys 'cats', 'octopi', and 'emus'. The 'cats' key refers to a list of strings ['Henri', 'Grumpy', 'Lucy']. All the other keys ('octopi', 'emus', 'plants', 'other') refer to empty dictionaries.

The final **life** dictionary is then printed, showing the multilevel structure with the nested dictionaries and list.

14. Print the top-level keys of life.

Ans : To print the top-level keys of the life dictionary, you can use the keys() method. Here's an example:

pythonCopy code

life = { 'animals': { 'cats': ['Henri', 'Grumpy', 'Lucy'], 'octopi': {}, 'emus': {} }, 'plants': {}, 'other': {} } top\_level\_keys = life.keys() print(top\_level\_keys)

Output:

cssCopy code

dict\_keys(['animals', 'plants', 'other'])

In this example, I used the **keys()** method to retrieve the top-level keys of the **life** dictionary. The **top\_level\_keys** variable stores a **dict\_keys** object, which represents a view of the dictionary's keys. Finally, the **top\_level\_keys** object is printed, showing the top-level keys as **['animals', 'plants', 'other']**.

15. Print the keys for life['animals'].

Ans : To print the keys for the 'animals' dictionary within the life dictionary, you can access it using the key 'animals' and then retrieve its keys using the keys() method. Here's an example:

life = { 'animals': { 'cats': ['Henri', 'Grumpy', 'Lucy'], 'octopi': {}, 'emus': {} }, 'plants': {}, 'other': {} } animals\_keys = life['animals'].keys() print(animals\_keys)

Output:

dict\_keys(['cats', 'octopi', 'emus'])

In this example, I accessed the **'animals'** dictionary within the **life** dictionary using the key **'animals'**. Then, I used the **keys()** method to retrieve the keys of the **'animals'** dictionary. The **animals\_keys** variable stores a **dict\_keys** object representing a view of the keys. Finally, the **animals\_keys** object is printed, displaying the keys of the **'animals'** dictionary as **['cats', 'octopi', 'emus']**.

16. Print the values for life['animals']['cats']

Ans : To print the values for life['animals']['cats'], you can access the nested dictionaries and lists using their respective keys. Here's an example:

life = { 'animals': { 'cats': ['Henri', 'Grumpy', 'Lucy'], 'octopi': {}, 'emus': {} }, 'plants': {}, 'other': {} } cats\_values = life['animals']['cats'] print(cats\_values)

Output:

['Henri', 'Grumpy', 'Lucy']

In this example, I accessed the **'animals'** dictionary within the **life** dictionary using the key **'animals'**. Then, I accessed the **'cats'** list within the **'animals'** dictionary using the key **'cats'**. The **cats\_values** variable stores the list **['Henri', 'Grumpy', 'Lucy']**. Finally, the **cats\_values** list is printed, displaying the values for **life['animals']['cats']**.