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].**

As an AI language model, I don't have a birth year, but I can certainly help you create the requested list. Let's assume your birth year is 2018, and you want the list of years until your fifth birthday (up to 2023). Here's the code to create the list:

```python

# Assuming your birth year is 2018

birth\_year = 2018

# Create a list of years starting from your birth year and up to your fifth birthday

years\_list = [year for year in range(birth\_year, birth\_year + 5)]

# Print the list

print(years\_list)

```

The output will be:

```

[2018, 2019, 2020, 2021, 2022]

```

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

If we consider that you were 0 years old for your first year, then your third birthday would have occurred three years after your birth. To find the year of your third birthday in the `years\_list`, you can access the element at index 3 (since lists are zero-indexed in Python).

Assuming the `years\_list` is the same as the one created in the previous answer:

```python

# Assuming your birth year is 2018

birth\_year = 2018

# Create a list of years starting from your birth year and up to your fifth birthday

years\_list = [year for year in range(birth\_year, birth\_year + 5)]

# Find the year of your third birthday

year\_of\_third\_birthday = years\_list[3]

# Print the result

print(year\_of\_third\_birthday)

```

The output will be:

```

2021

```

So, your third birthday would have been in the year 2021.

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

To find the year in the `years\_list` when you were the oldest, you need to select the maximum value from the list. The maximum value in the `years\_list` will represent the year when you were the oldest.

Assuming the `years\_list` is the same as before:

```python

# Assuming your birth year is 2018

birth\_year = 2018

# Create a list of years starting from your birth year and up to your fifth birthday

years\_list = [year for year in range(birth\_year, birth\_year + 5)]

# Find the year when you were the oldest

oldest\_year = max(years\_list)

# Print the result

print(oldest\_year)

```

The output will be:

```

2022

```

So, you were the oldest in the year 2022.

1. **Make a list called things with these three strings as elements: “mozzarella”, “cinderella”, “salmonella”.**

Sure, here's a list called `things` with the three strings as elements:

```python

things = ["mozzarella", "cinderella", "salmonella"]

print(things)

```

The output will be:

```

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

```

Now, you have a list named `things` with the elements "mozzarella", "cinderella", and "salmonella".

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

**element in the list?**

To capitalize the element in the `things` list that refers to a person (which is "cinderella" in this case), we can use the `capitalize()` method. The `capitalize()` method returns a copy of the string with its first character capitalized and the rest of the characters in lowercase.

Here's how you can do it and print the list:

```python

things = ["mozzarella", "cinderella", "salmonella"]

# Capitalize the element that refers to a person

things[1] = things[1].capitalize()

# Print the updated list

print(things)

```

The output will be:

```

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

```

As you can see, the element "cinderella" has been changed to "Cinderella" in the `things` list after capitalizing it. The `capitalize()` method modified the element in the list.

1. **Make a surprise list with the elements “Groucho” ,“Chico” and “Harpo.”**

Sure, here's a list called `surprise` with the elements "Groucho," "Chico," and "Harpo":

```python

surprise = ["Groucho", "Chico", "Harpo"]

print(surprise)

```

The output will be:

```

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

```

Now, you have a list named `surprise` with the elements "Groucho," "Chico," and "Harpo."

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

Sure! Here's how you can lowercase the last element of the `surprise` list, reverse it, and then capitalize it:

```python

surprise = ["Groucho", "Chico", "Harpo"]

# Lowercase the last element

last\_element\_lowercased = surprise[-1].lower()

# Reverse the last element

reversed\_last\_element = last\_element\_lowercased[::-1]

# Capitalize the reversed last element

capitalized\_reversed\_last\_element = reversed\_last\_element.capitalize()

# Replace the last element in the surprise list with the modified one

surprise[-1] = capitalized\_reversed\_last\_element

# Print the updated surprise list

print(surprise)

```

The output will be:

```

['Groucho', 'Chico', 'Oprah']

```

Now, the last element "Harpo" has been modified to "Oprah" in the `surprise` list after the specified operations.

1. **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.**

Sure, I'll create an English-to-French dictionary called `e2f` and print its contents:

```python

# Create the English-to-French dictionary

e2f = {

'dog': 'chien',

'cat': 'chat',

'walrus': 'morse'

}

# Print the e2f dictionary

print(e2f)

```

The output will be:

```

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

```

Now, you have the `e2f` dictionary with the English words as keys and their corresponding French translations as values.

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

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

```python

# Create the English-to-French dictionary

e2f = {

'dog': 'chien',

'cat': 'chat',

'walrus': 'morse'

}

# Access the French word for "walrus"

french\_word\_for\_walrus = e2f['walrus']

# Print the result

print(french\_word\_for\_walrus)

```

The output will be:

```

morse

```

So, in the `e2f` dictionary, "walrus" maps to "morse" in French.

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

You can create a French-to-English dictionary called `f2e` from the existing `e2f` dictionary using the `items()` method. The `items()` method returns a view object that displays a list of dictionary's key-value pairs. To create `f2e`, we will swap the keys and values from `e2f`. Here's how you can do it:

```python

# Existing English-to-French dictionary e2f

e2f = {

'dog': 'chien',

'cat': 'chat',

'walrus': 'morse'

}

# Create the French-to-English dictionary f2e using items() method

f2e = {french\_word: english\_word for english\_word, french\_word in e2f.items()}

# Print the f2e dictionary

print(f2e)

```

The output will be:

```

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

```

Now, you have the `f2e` dictionary with French words as keys and their corresponding English translations as values, derived from the `e2f` dictionary.

1. **Print the English version of the French word chien using f2e.**

To print the English version of the French word "chien" using the `f2e` dictionary, you can simply access the value corresponding to the key "chien" in the `f2e` dictionary. Here's how to do it:

```python

# Existing French-to-English dictionary f2e

f2e = {

'chien': 'dog',

'chat': 'cat',

'morse': 'walrus'

}

# Print the English version of the French word "chien"

english\_word\_for\_chien = f2e['chien']

print(english\_word\_for\_chien)

```

The output will be:

```

dog

```

So, the English version of the French word "chien" is "dog" according to the `f2e` dictionary.

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

To create a set of English words from the keys in the `e2f` dictionary and then print it, you can use the `set()` function. The `set()` function creates a new set containing the unique elements from the given iterable (in this case, the keys of the `e2f` dictionary). Here's how you can do it:

```python

# Existing English-to-French dictionary e2f

e2f = {

'dog': 'chien',

'cat': 'chat',

'walrus': 'morse'

}

# Create a set of English words from the keys in e2f

english\_words\_set = set(e2f.keys())

# Print the set of English words

print(english\_words\_set)

```

The output will be:

```

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

```

Now, you have a set containing the English words from the keys in the `e2f` dictionary: {'dog', 'cat', 'walrus'}.

1. **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.**

Sure, here's the multilevel dictionary called `life` as described:

```python

# Create the multilevel dictionary life

life = {

'animals': {

'cats': ['Henri', 'Grumpy', 'Lucy'],

'octopi': {},

'emus': {}

},

'plants': {},

'other': {}

}

# Print the life dictionary

print(life)

```

The output will be:

```

{

'animals': {

'cats': ['Henri', 'Grumpy', 'Lucy'],

'octopi': {},

'emus': {}

},

'plants': {},

'other': {}

}

```

Now, you have the `life` dictionary structured as follows:

- The 'animals' key refers to another dictionary containing keys 'cats', 'octopi', and 'emus'.

- The 'cats' key refers to a list of strings ['Henri', 'Grumpy', 'Lucy'].

- The 'octopi' and 'emus' keys refer to empty dictionaries.

- The 'plants' and 'other' keys refer to empty dictionaries as well.

1. **Print the top-level keys of life.**

To print the top-level keys of the `life` dictionary, you can use the `keys()` method. The `keys()` method returns a view object that displays a list of all the keys in the dictionary. Here's how you can do it:

```python

# The multilevel dictionary life

life = {

'animals': {

'cats': ['Henri', 'Grumpy', 'Lucy'],

'octopi': {},

'emus': {}

},

'plants': {},

'other': {}

}

# Print the top-level keys of the life dictionary

top\_level\_keys = life.keys()

print(top\_level\_keys)

```

The output will be a view object of the top-level keys:

```

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

```

The `dict\_keys` object is similar to a list of the keys, and you can convert it to a list if you want to manipulate it further:

```python

top\_level\_keys\_list = list(top\_level\_keys)

print(top\_level\_keys\_list)

```

The output will be a list of the top-level keys:

```

['animals', 'plants', 'other']

```

Now, you have the top-level keys of the `life` dictionary stored in the `top\_level\_keys\_list`.

1. **Print the keys for life[‘animals’].**

To print the keys for the `'animals'` dictionary within the `life` dictionary, you can simply access it using the key `'animals'` and then use the `keys()` method. Here's how you can do it:

```python

# The multilevel dictionary life

life = {

'animals': {

'cats': ['Henri', 'Grumpy', 'Lucy'],

'octopi': {},

'emus': {}

},

'plants': {},

'other': {}

}

# Print the keys for the 'animals' dictionary

animals\_keys = life['animals'].keys()

print(animals\_keys)

```

The output will be a view object of the keys for the 'animals' dictionary:

```

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

```

If you want to convert the `dict\_keys` object into a list, you can do the following:

```python

animals\_keys\_list = list(animals\_keys)

print(animals\_keys\_list)

```

The output will be a list of the keys for the 'animals' dictionary:

```

['cats', 'octopi', 'emus']

```

Now, you have the keys for the 'animals' dictionary within the `life` dictionary.

1. **Print the values for life[‘animals’][‘cats’]**

To print the values for the `'cats'` key within the `'animals'` dictionary of the `life` dictionary, you can directly access it using the appropriate keys. Here's how you can do it:

```python

# The multilevel dictionary life

life = {

'animals': {

'cats': ['Henri', 'Grumpy', 'Lucy'],

'octopi': {},

'emus': {}

},

'plants': {},

'other': {}

}

# Print the values for the 'cats' key within the 'animals' dictionary

cats\_values = life['animals']['cats']

print(cats\_values)

```

The output will be the list of values for the 'cats' key:

```

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

```

Now, you have the values for the 'cats' key within the `'animals'` dictionary of the `life` dictionary.