Name :- Deepawali . B. Mhaisagar
Assignment no 16

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

To create a list called `years_list` starting with the year of your birth and each year thereafter until the year of your fifth birthday, you can use a combination of the `range()` function and list comprehension. Here's an example:

birth_year = 1990 # Replace with your birth year

years_list = [year for year in range(birth_year, birth_year + 6)]

In the code snippet above, you need to replace `1990` with your actual birth year. The `range()` function is used to generate a sequence of years starting from the birth year and ending at the birth year plus 5. The `for` loop within the list comprehension iterates over each year in the generated range and adds it to the `years_list`.

After executing this code, the `years_list` will contain the years starting from your birth year and continuing for the next five years. For example, if your birth year is 1990, the resulting `years_list` would be `[1990, 1991, 1992, 1993, 1994, 1995]`.

2. Assign the result from the previous task (seconds in an hour) to a variable called seconds_per_hour.

ANS

If you were 0 years old for your first year, then your third birthday would occur two years after your birth. To find the year of your third birthday in the `years_list`, you can access the element at index 2. Here's an example:

```python

birth_year = 1990  # Replace with your birth year
```

```
years_list = [year for year in range(birth_year, birth_year + 6)]

third_birthday_year = years_list[2]
```

After executing this code, the variable `third_birthday_year` will contain the year in which you celebrated your third birthday based on the `years_list`. For example, if your birth year is 1990, the value of `third_birthday_year` would be 1992.

3. What is a box tuple, and how does it work?

ANS

To determine the year in which you were the oldest in the `years_list`, you can simply access the last element of the list. Since the list is generated in ascending order, the last element will represent the highest year. Here's an example:

```python
birth_year = 1990  # Replace with your birth year

years_list = [year for year in range(birth_year, birth_year + 6)]

oldest_year = years_list[-1]
```

After executing this code, the variable `oldest_year` will contain the highest year in the `years_list`, representing the year in which you were the oldest. For example, if your birth year is 1990, the value of `oldest_year` would be 1995.

4. Calculate seconds per day again, but this time save the result in a variable called seconds_per_day

ANS

To create a list called `things` with the strings "mozzarella", "cinderella", and "salmonella" as elements, you can simply initialize the list with those strings. Here's an example:

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

After executing this code, the `things` list will contain the three specified strings as elements.

5. What method would you call to get Image object for a 100×100 image, excluding the lower-left quarter of it?

ANS

To capitalize the element in the `things` list that refers to a person, which is "cinderella", you can use the `capitalize()` method. The `capitalize()` method capitalizes the first character of a string.

Here's an example:

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

# Capitalize the element that refers to a person

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

# Print the list

print(things)
```

After executing this code, the output will be:

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

As you can see, the element "cinderella" in the `things` list has been capitalized to "Cinderella". Modifying the element using `things[1] = things[1].capitalize()` changed the element in the list.

6. Make a surprise list with the elements “Groucho,”; “Chico”; and “Harpo”

ANS

To create a list called `surprise` with the elements "Groucho," "Chico," and "Harpo," you can initialize the list with these strings. Here's an example:

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

After executing this code, the `surprise` list will contain the three specified strings as elements.

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

ANS

To perform the operations of lowercase, reverse, and capitalize on the last element of the `surprise` list, you can follow these steps:

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



# Lowercase the last element

last_element = surprise[-1].lower()



# Reverse the last element

reversed_element = last_element[::-1]



# Capitalize the reversed element

capitalized_element = reversed_element.capitalize()
```

```
# Update the last element in the surprise list

surprise[-1] = capitalized_element



print(surprise)
```

After executing this code, the output will be:

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

As you can see, the last element "Harpo" has been transformed to "oprah" by performing the lowercase, reverse, and capitalize operations on it.

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

To create an English-to-French dictionary called `e2f` with the given starter words, you can define a dictionary using curly braces {} and assign the English words as keys and their corresponding French translations as values. Here's an example:

```
e2f = {

    'dog': 'chien',

    'cat': 'chat',

    'walrus': 'morse'

}



print(e2f)
```

After executing this code, the output will be:

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

As shown, the `e2f` dictionary contains the English words as keys and their corresponding French translations as values.

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

ANS

In the `e2f` dictionary, the French word for "walrus" is "morse". To access the French translation for "walrus" from the dictionary, you can use the key "walrus" as follows:

```
e2f = {

    'dog': 'chien',

    'cat': 'chat',

    'walrus': 'morse'

}



french_word_for_walrus = e2f['walrus']

print(french_word_for_walrus)
```

After executing this code, the output will be:

```
morse
```

As shown, the variable `french_word_for_walrus` contains the French translation "morse" obtained by accessing the corresponding value associated with the key `'walrus'` in the `e2f` dictionary.

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

ANS

To create a French-to-English dictionary called `f2e` from the existing `e2f` dictionary, you can use the `items()` method along with a dictionary comprehension. Here's an example:

```
e2f = {

    'dog': 'chien',

    'cat': 'chat',

    'walrus': 'morse'

}



f2e = {value: key for key, value in e2f.items()}



print(f2e)
```

After executing this code, the output will be:

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

```

```

In this code, the `items()` method is used to iterate over the key-value pairs in the `e2f` dictionary. The dictionary comprehension `{value: key for key, value in e2f.items()}` creates a new dictionary `f2e` where the keys are the French words and the values are their corresponding English translations.

The resulting `f2e` dictionary is a French-to-English dictionary that is reverse-mapped from the `e2f` dictionary using the `items()` method.

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" in the dictionary. Here's an example:

```
f2e = {
    'chien': 'dog',

    'chat': 'cat',

    'morse': 'walrus'

}



english_word_for_chien = f2e['chien']

print(english_word_for_chien)
```

After executing this code, the output will be:

```
dog
```

As shown, the variable `english_word_for_chien` contains the English translation "dog" obtained by accessing the corresponding value associated with the key `'chien'` in the `f2e` dictionary.

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

ANS

To create and print a set of English words from the keys in the `e2f` dictionary, you can use the `set()` function and pass the keys of the dictionary as an argument. Here's an example:

```
e2f = {

    'dog': 'chien',

    'cat': 'chat',

    'walrus': 'morse'

}



english_words = set(e2f.keys())

print(english_words)
```

After executing this code, the output will be:

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

As shown, the `english_words` set contains the English words extracted from the keys of the `e2f` dictionary. The `set()` function converts the keys into a set, removing any duplicates and providing an unordered collection of unique English words.

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

To create the multilevel dictionary `life` as described, with the specified topmost keys and nested dictionaries, you can use the following code:

```
life = {

    'animals': {

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

        'octopi': {},

        'emus': {}

    },

    'plants': {},

    'other': {}

}



print(life)
```

After executing this code, the output will be:

```
{

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

The `life` dictionary has 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 containing the values `'Henri'`, `'Grumpy'`, and `'Lucy'`. The `'plants'` and `'other'` keys refer to empty dictionaries.

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:

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

```
top_level_keys = life.keys()

print(top_level_keys)
```

After executing this code, the output will be:

```
dict_keys(['animals', 'plants', 'other'])
```

As shown, the `top_level_keys` variable contains the top-level keys of the `life` dictionary. The `keys()` method returns a view object that represents the keys in the dictionary. In this case, the view object represents the keys `'animals'`, `'plants'`, and `'other'`.

15. Print the keys for life[‘;animals’;].

ANS

To print the keys for the nested dictionary `life['animals']`, you can access the dictionary using the key `'animals'` and then use the `keys()` method. Here's an example:

```
life = {

    'animals': {

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

        'octopi': {},

        'emus': {}

    },

    'plants': {},

    'other': {}

}



animal_keys = life['animals'].keys()
```

```
print(animal_keys)
```

After executing this code, the output will be:

```
dict_keys(['cats', 'octopi', 'emus'])
```

As shown, the `animal_keys` variable contains the keys for the dictionary `life['animals']`. The `keys()` method is called on the nested dictionary, returning a view object representing the keys. In this case, the view object contains the keys `'cats'`, `'octopi'`, and `'emus'`.

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

ANS

To print the values for `life['animals']['cats']`, which is a list of strings, you can access the nested dictionary and list using the corresponding keys. Here's an example:

```
life = {

    'animals': {

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

        'octopi': {},

        'emus': {}

    },

    'plants': {},

    'other': {}

}



cat_values = life['animals']['cats']

print(cat_values)
```

After executing this code, the output will be:

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

As shown, the `cat_values` variable contains the list of strings `['Henri', 'Grumpy', 'Lucy']` obtained by accessing `life['animals']['cats']`. By chaining the keys `'animals'` and `'cats'`, you can access the desired nested list of strings within the `life` dictionary.