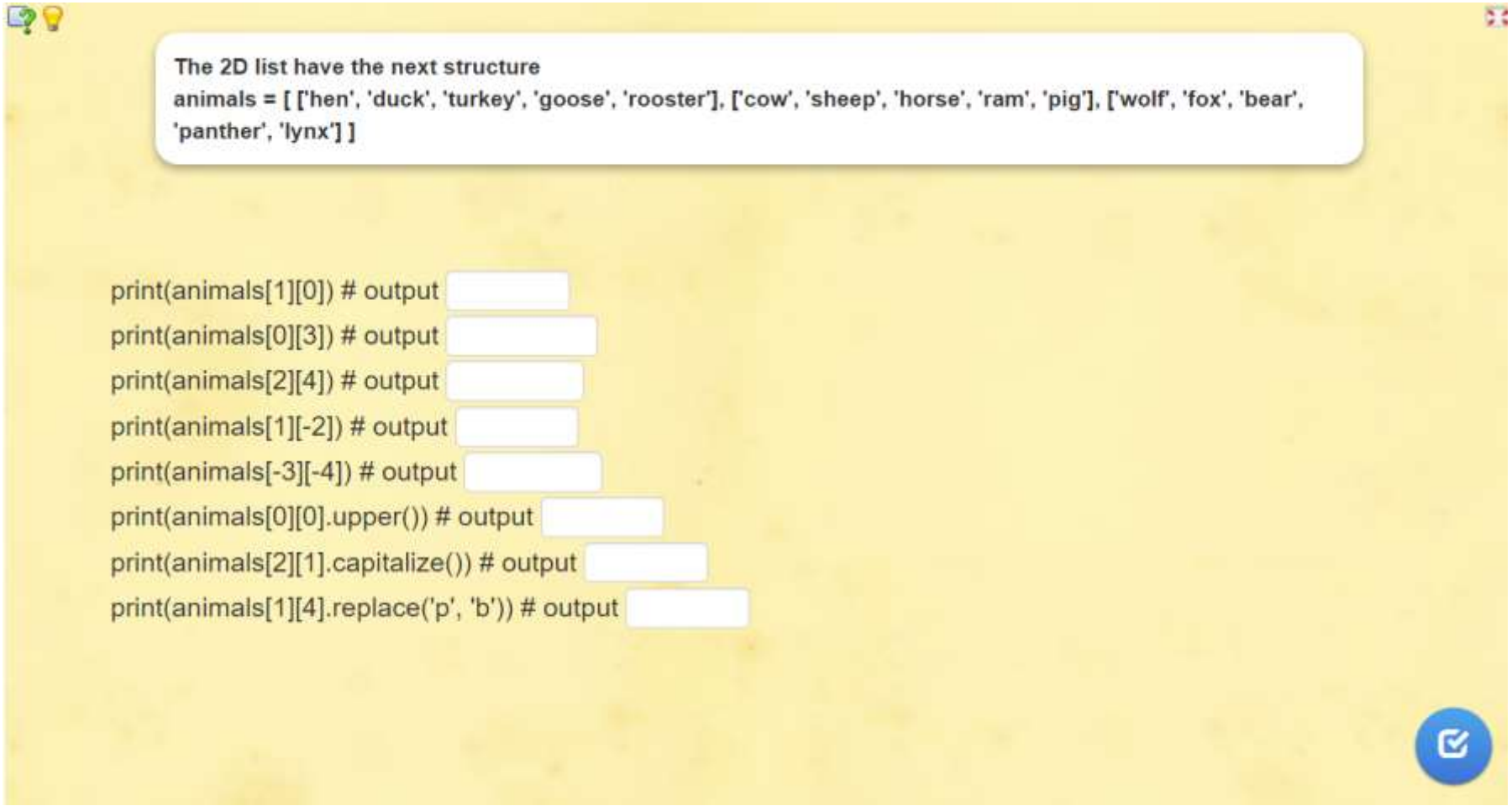
Revision: Nested Lists



Unit 11.3 A
Data structures
Dictionary

11.2.5.1 create a dictionary;

11.2.5.2 search for data in a dictionary for a given key;

11.2.3.6 determine the difference between different data structures;

11.4.3.2 solve applied problems of various subject areas.

Dictionary

We already know how to store data in indexed data structures such as lists.

But if we need to search for data in an explanatory dictionary or an English-Russian dictionary, then the way to store data in a list will not be very convenient.

In Python, the Dictionary data structure is used for this. Dictionaries are one of the core data structures in Python. A dictionary contains **key-value** pairs. A value can easily be accessed via its key.

Dictionary Syntax

Dictionaries are written with curly brackets, and have keys and values:

```
my_dict = {'key1' : 'value1', 'key2' : 'value2', ...}
```

You can access values in a dictionary by using the keys, for example:

```
print(my_dict['key1']) # Output: 'value1'
```

Dictionary Syntax

Two ways to create empty dictionaries:

```
d = dict()
d = {}
```

For example, the English-Russian dictionary of colors can be written as follows

```
colors = {'white': 'белый', 'red': 'красный', 'green':
'зеленый', 'blue': 'синий', 'black': 'черный'}
```

'white', 'red', 'green', 'blue', 'black' - **keys of the dictionary,**
'белый', 'красный', 'зеленый', 'синий', 'черный' - **their values.**

Dictionary

A dictionary is a mutable, iterated, and unordered data structure, that can be indexed by keys.

Dictionary keys can only be immutable data types. In Python, these are numbers, strings, and tuples.

Usually, the same type of data is used in one dictionary.

Access to the value of an element by key

```
colors = {'white': 'белый', 'red': 'красный', 'green':
'зеленый', 'blue': 'синий', 'black': 'черный’}
```

```
print(colors['red’])
```

```
красный
```

```
print(colors['blue’])
```

```
синий
```

```
print(colors['yellow'])
```

```
KeyError: 'yellow'
```

Changing an element

```
colors = {'green': 'зиленый', 'blue': 'синий', 'black':
'черный’}

colors['green'] = 'зеленый' #change value with key 'green’

print(colors) # {'green': 'зеленый', 'blue': 'синий',
'black': 'черный'}
```

Adding an element

```
colors = {'blue': 'синий', 'black': 'черный’}

colors['yellow'] = 'желтый' # add new element

print(colors) # {'blue': 'синий', 'black': 'черный',
'yellow': 'желтый'}
```

Delete element(del)

```
colors = {'white': 'белый', 'red': 'красный', 'green':
'зеленый', 'blue': 'синий', 'black': 'черный'}


del colors['white']
del colors['black']
print(colors)  # {'red': 'красный', 'green': 'зеленый',
'blue': 'синий'}
```

Delete element(pop)

```
colors = {'white': 'белый', 'red': 'красный', 'green':
'зеленый', 'blue': 'синий', 'black': 'черный’}

colors.pop('white') # delete element with key 'white’


deleted_element = colors.pop('black') # remember deleted
element with index 'black’

print(colors)  # {'red': 'красный', 'green': 'зеленый',
'blue': 'синий’}

print(deleted_element)  # output черный
```

Checking if an element exists in a dictionary

```
colors = {'white': 'белый', 'red': 'красный', 'green':
'зеленый', 'blue': 'синий', 'black': 'черный'}
if 'red' in colors:
    print('Dictionary has word: red’)
```

```
colors = {'white': 'белый', 'red': 'красный', 'green':
'зеленый', 'blue': 'синий', 'black': 'черный'}
if 'yellow' not in colors:
    print('Dictionary has not this word')
```

Practice Tasks(7 min)

Given a dictionary with capitals of countries:
capitals = {"Germany": "Berlin", "Canada": "Ottawa", "England": "London", "Japan": "Tokyo", "Kazakhstan": "Nur-Sultan", "Turkey": "Ankara", }

1. Print the capital of Canada
2. Print the capital of Germany
3. Add a country '**Italy**' with capital '**Rome**' to the original dictionary
4. Change the **'Nur-Sultan'** to **'Astana'**
5. Check if **'Japan'** is present in the dictionary:
6. Remove '**England**' with **del** method
7. Remove '**Turkey**' with **pop** method

Answer

1. print(capitals["Canada"])

2. print(capitals["Germany"])

3. capitals["Italy"] = "Rome"

4. capitals["Kazakhstan"] = "Astana"

5. if "Japan" in capitals:
   print("Japan is present in the dictionary.")
   else:
   print("Japan is not present in the dictionary.")

6. del capitals["England"]

7. capitals.pop("Turkey")

Methods of dictionary

Function/ Method	Example	Description

d.get(key)
d.get(key, other value)
	d = {'A': 65, 'B': 66, 'C': 67, 'D': 68}
	value = d.get('B')
	print(value) # output 66
	value = d.get('E', 'Not exist')
	print(value) # output Not exist
		In addition to the key, this method can also use the second parameter, in which you can display a message if the key is not in the dictionary. In this case, the program will not display an error.

d.keys()	d = {'A': 65, 'B': 66, 'C': 67, 'D': 68} letters = list(d.keys()) print(letters) # ['A', 'B', 'C', 'D']	Method keys() allows getting a list of all the keys of a dictionary
d.values()	d = {'A': 65, 'B': 66, 'C': 67, 'D': 68} letters = list(d.values()) print(letters) # [65, 66, 67, 68]	Method values() allow getting a list of all the values of a dictionary Can check the value in d.values()

Methods of dictionary

Function/ Method	Example	Description
d.items()	d = {'A': 65, 'B': 66, 'C': 67, 'D': 68} for key, value in d.items(): print(key, value) # Output: # A 65 # B 66 # C 67 # D 68	Method items() allow getting all pairs of keys and values of a dictionary
d1.update(d2)	d1 = {'A': 65, 'B': 66} d2 = {'C': 67, 'D': 68} d1.update(d2) print(d1) # {'A': 65, 'B': 66, 'C': 67, 'D': 68}	Merge two dictionaries into one

Define the output of the code

```
capitals = {
    "Germany": "Berlin",
    "Canada": "Ottawa",
    "England": "London",
    "Japan": "Tokyo",
    "Kazakhstan": "Nur-Sultan",
    "Turkey": "Ankara"}

print(capitals.get('England'))
print(list(capitals.keys()))
print(list(capitals.values()))

for key, value in capitals.items():
    print(key, value)
```

London

['Germany', 'Canada', 'England', 'Japan', 'Kazakhstan', 'Turkey’]

['Berlin', 'Ottawa', 'London', 'Tokyo', 'Nur-Sultan', 'Ankara’]

Germany Berlin
Canada Ottawa
England London
Japan Tokyo
Kazakhstan Nur-Sultan
Turkey Ankara

Define the output of the code

```
capitals = {
    "Germany": "Berlin",
    "Canada": "Ottawa",
    "England": "London",
    "Japan": "Tokyo",
    "Kazakhstan": "Nur-Sultan",
    "Turkey": "Ankara"}

capitals1 = {
    "Spain":"Madrid",
    "Italy": "Rome"
}
capitals.update({"Kazakhstan": 'Astana'})
print(capitals)

capitals.update(capitals1)
print(capitals)
```

{'Germany': 'Berlin', 'Canada': 'Ottawa', 'England': 'London', 'Japan': 'Tokyo', 'Kazakhstan': 'Astana', 'Turkey': 'Ankara’}

{'Germany': 'Berlin', 'Canada': 'Ottawa', 'England': 'London', 'Japan': 'Tokyo', 'Kazakhstan': 'Astana', 'Turkey': 'Ankara', 'Spain': 'Madrid', 'Italy': 'Rome'}

Loop Through a Dictionary

```
thisdict ={
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
```

Definition	Loop	Output
Print all key names in the dictionary, one by one	for x in thisdict: print(x)	
Print all ***values*** in the dictionary, one by one	for x in thisdict: print(thisdict[x])	
You can also use the values() method to return values of a dictionary	for x in thisdict.**values()**: print(x)	
You can use the keys() method to return the keys of a dictionary	for x in thisdict.**keys():** print(x)	
Loop through both *keys* and *values*, by using the items() method	for x, y in thisdict.**items():** print(x, y)	

You can input elements of a dictionary from the keyboard in Python by using the input() function to get user input. Here's an example:

```
dict = {}

# Get user input for the number of dictionary pairs
n = int(input("Enter the length of the dictionary:)"))

# Input dictionary pairs
for i in range(n):
    key = input("Enter a key: ")
    value = input("Enter a value : ")

    # Add the pair to the dictionary
    dict[key] = value

# Print the resulting dictionary
print("Dictionary:", dict)
```

Practice Tasks(Group work)

Task 1. Write a Python program to sum all the items in a dictionary.

Task 2. Write a Python program to multiply all the items in a dictionary.

Task 3. Write a Python program to sort a given dictionary by key

Task 4. Write a Python program to get the maximum and minimum values of a dictionary.

Individual tasks

Task 1

Write a Python script to print a dictionary where the keys are numbers between 1 and n (both included) and the values are square of keys

Sample Input:

```
3
```

Sample Output:

```
{1: 1, 2: 4, 3: 9}
```

Task 2

Given a long list of integers numbers. We know that some numbers appear more than once in this list. You need to find out exactly how many times each of the numbers occurs and write the data into a dictionary.

Sample Input:

```
4 5 6 5 4 9
```

Sample Output:

```
{4: 2, 5: 2, 6: 1, 9: 1}
```

Task 3

You are given a dictionary consisting of pairs of words. Each word is a synonym for its paired word. All words in the dictionary are different.

For the word from the dictionary written in the last line, determine its synonym.

Sample Input:

```
3
Hello Hi
Bye Goodbye
List Array
Goodbye
```

Sample Output:

```
Bye
```

Task 4

In the board game Scrabble, each letter has a certain value. In the case of the English alphabet, points are distributed as follows:

A, E, I, O, U, L, N, S, T, R – 1 point;
D, G – 2 points;
B, C, M, P – 3 points;
F, H, V, W, Y – 4 points;
K – 5 points;
J, X – 8 points;
Q, Z – 10 points.

Write a program that calculates the cost of a word entered by the user. We will assume that only one word is given as input, which contains only English words.

Sample Input:

```
apple
```

Sample Output:

```
9
```

Reflection

What have you learned during the lesson?

What remained unclear?