1. \*\*What will be the output of the following code?\*\*

```python

nested\_list = [1, [2, 3, [4, [5, 6]]]]

nested\_list[1][2][1].append(7)

print(nested\_list)

```

- A) `[1, [2, 3, [4, [5, 6]]]]`

- B) `[1, [2, 3, [4, [5, 6, 7]]]]`

- C) `[1, [2, 3, [4, 5, 6, 7]]]`

- D) `Error: IndexError`

**Answer:** B) [1, [2, 3, [4, [5, 6, 7]]]]

**Explanation:** The append operation modifies the nested list inside the list. nested\_list[1][2][1] accesses [5, 6], and appending 7 makes it [5, 6, 7].

2. \*\*What will be the output of this code involving nested dictionaries and list indexing?\*\*

```python

nested\_dict = {'A': {'B': [{'C': 1}, {'D': 2}]}}

print(nested\_dict['A']['B'][1]['D'])

```

- A) `1`

- B) `2`

- C) `Error: KeyError`

- D) `Error: IndexError`

**Answer:** B) 2

**Explanation:** nested\_dict['A']['B'] gives the list [{‘C’: 1}, {'D': 2}], and nested\_dict['A']['B'][1]['D'] accesses the dictionary {'D': 2} at index 1 and retrieves 2.

3. \*\*Consider the following code snippet. What is the most efficient way to access the value `8`?\*\*

```python

complex\_dict = {'x': [1, {'y': [2, 3, {'z': 4}]}, 5], 'w': [6, {'q': 8}]}

```

- A) `complex\_dict['x'][1]['y'][2]['z']`

- B) `complex\_dict['w'][1]['q']`

- C) `complex\_dict['w'][0]['q']`

- D) `complex\_dict['x'][2]['q']`

 **Answer:** B) complex\_dict['w'][1]['q']

 **Explanation:** complex\_dict['w'] gives [6, {'q': 8}]. The dictionary {'q': 8} is at index 1, and accessing ['q'] gives 8.

4. \*\*What will be the result of the following operation?\*\*

```python

my\_tuple = (1, [2, 3], (4, 5))

my\_tuple[1].append(6)

print(my\_tuple)

```

- A) `(1, [2, 3], (4, 5))`

- B) `(1, [2, 3, 6], (4, 5))`

- C) `Error: Tuples are immutable`

- D) `Error: Append operation not supported`

 **Answer:** B) (1, [2, 3, 6], (4, 5))

 **Explanation:** While tuples are immutable, the list inside the tuple ([2, 3]) can be modified. The append operation adds 6 to the list, so the tuple becomes (1, [2, 3, 6], (4, 5)).

5. \*\*What is the correct way to replace the value `'A'` with `'Z'` in the following nested dictionary?\*\*

```python

nested\_dict = {'level1': {'level2': {'level3': 'A'}}}

```

- A) `nested\_dict['level1']['level2']['level3'] = 'Z'`

- B) `nested\_dict['level1']['level3']['level2'] = 'Z'`

- C) `nested\_dict.replace('A', 'Z')`

- D) `nested\_dict['level1']['level2'] = 'Z'`

 **Answer:** A) nested\_dict['level1']['level2']['level3'] = 'Z'

 **Explanation:** The value 'A' is nested within nested\_dict['level1']['level2']['level3'], and you can directly assign 'Z' to this key.

6. \*\*Given the following nested tuple, what is the output of accessing `nested\_tuple[1][1][1][1]`?\*\*

```python

nested\_tuple = (1, (2, [3, (4, [5, (6, 7)])]))

```

- A) `7`

- B) `6`

- C) `4`

- D) `IndexError`

 **Answer:** A) 7

 **Explanation:** nested\_tuple[1][1][1][1] accesses the innermost list and tuple, eventually reaching the value 7.

7. \*\*In the following complex data structure, how would you access the second element of the list inside the dictionary?\*\*

```python

data\_structure = {'a': (1, 2), 'b': [3, {'c': [4, 5]}, 6]}

```

- A) `data\_structure['b'][1]['c'][1]`

- B) `data\_structure['b'][0]`

- C) `data\_structure['b'][1][1]`

- D) `data\_structure['a'][1][1]`

 **Answer:** A) data\_structure['b'][1]['c'][1]

 **Explanation:** data\_structure['b'] accesses [3, {'c': [4, 5]}, 6]. data\_structure['b'][1] gives {'c': [4, 5]}, and data\_structure['b'][1]['c'][1] accesses 5.

8. \*\*What will be the output of the following code involving a nested list and dictionary?\*\*

```python

nested\_list = [1, {'a': [2, 3]}, [4, 5]]

nested\_list[1]['a'].append(6)

print(nested\_list)

```

- A) `[1, {'a': [2, 3]}, [4, 5]]`

- B) `[1, {'a': [2, 3, 6]}, [4, 5]]`

- C) `Error: Tuples are immutable`

- D) `Error: KeyError`

 **Answer:** B) [1, {'a': [2, 3, 6]}, [4, 5]]

 **Explanation:** The append operation modifies the list inside the dictionary at nested\_list[1]['a'], making it [2, 3, 6].

9. \*\*What will be the output of the following code involving list and tuple slicing?\*\*

```python

my\_tuple = (1, [2, 3, (4, 5)], 6)

result = my\_tuple[1][2][0:2]

print(result)

```

- A) `[4]`

- B) `(4, 5)`

- C) `Error: Tuples are immutable`

- D) `Error: IndexError`

 **Answer:** B) (4, 5)

 **Explanation:** my\_tuple[1][2] accesses the tuple (4, 5), and slicing 0:2 returns the whole tuple (4, 5) since both elements are within the slice range.

10. \*\*How do you correctly update the value of `'z'` to `99` in the following dictionary?\*\*

```python

nested\_dict = {'x': {'y': {'z': 5}}, 'w': {'u': 7}}

```

- A) `nested\_dict['x']['y']['z'] = 99`

- B) `nested\_dict['w']['y']['z'] = 99`

- C) `nested\_dict['y']['z'] = 99`

- D) `nested\_dict.update({'z': 99})`

 **Answer:** A) nested\_dict['x']['y']['z'] = 99

 **Explanation:** The value for 'z' is nested under nested\_dict['x']['y']['z'], so you can directly assign 99 to this key.

11. \*\*What will be the output of the following code?\*\*

```python

nested\_list = [1, [2, [3, [4, [5, 6]]]]]

value = nested\_list[1][1][1][1][0]

print(value)

```

- A) `4`

- B) `5`

- C) `6`

- D) `Error`

\*\*Explanation:\*\* This accesses a deeply nested list. `nested\_list[1]` gives `[2, [3, [4, [5, 6]]]]`, and the indexing continues correctly down to `5`.

---

12. \*\*What will be the output of this code?\*\*

```python

nested\_dict = {'A': {'B': [{'C': [1, 2]}, {'D': [3, 4]}]}}

result = nested\_dict['A']['B'][1]['D'][0]

print(result)

```

- A) `1`

- B) `2`

- C) `3`

- D) `4`

\*\*Explanation:\*\* The value for `D` is `[3, 4]`, and accessing `[0]` returns `3`.

---

13. \*\*Given the tuple `nested\_tuple = (1, [2, 3, (4, 5)], 6)`, how can you modify the value 3 to 33?\*\*

- A) `nested\_tuple[1][1] = 33`

- B) `nested\_tuple[2] = 33`

- C) `nested\_tuple[1][2][1] = 33`

- D) Cannot modify the tuple

\*\*Explanation:\*\* Tuples themselves are immutable, but elements inside a tuple that are mutable (like a list) can be changed.

---

14. \*\*What is the correct way to access the last element (`7`) in the following list?\*\*

```python

lst = [1, [2, [3, [4, [5, [6, 7]]]]]]

```

- A) `lst[5][6]`

- B) `lst[1][1][1][1][1][1]`

- C) `lst[-1][-1]`

- D) `lst[1][1][-1][0][-1]`

\*\*Explanation:\*\* The path is through multiple nested lists: `[1][1][1][1][1][1]`.

---

15. \*\*What will be the output of the following code?\*\*

```python

nested\_list = [1, 2, [3, 4], 5]

nested\_list[2].insert(1, [6, 7])

print(nested\_list)

```

- A) `[1, 2, [3, [6, 7], 4], 5]`

- B) `[1, 2, [3, 4, [6, 7]], 5]`

- C) `Error: IndexError`

- D) `[1, 2, 3, 4, 6, 7, 5]`

\*\*Explanation:\*\* `insert(1, [6, 7])` inserts `[6, 7]` at index `1` within the sublist.

---

16. \*\*What is the correct way to delete the key `'z'` from the following dictionary?\*\*

```python

nested\_dict = {'x': {'y': {'z': 9}}, 'w': [5, 6]}

```

- A) `del nested\_dict['x']['y']['z']`

- B) `nested\_dict.pop('z')`

- C) `nested\_dict.pop('y')['z']`

- D) `nested\_dict['z'] = None`

\*\*Explanation:\*\* The correct method is to use `del` on the specific key inside the nested dictionary.

---

17. \*\*What will be the output of this tuple manipulation?\*\*

```python

my\_tuple = (1, [2, 3], (4, 5))

my\_tuple[1][1] = 33

print(my\_tuple)

```

- A) `(1, [2, 33], (4, 5))`

- B) `(1, [2, 3], (4, 5))`

- C) `Error: Tuples are immutable`

- D) `(1, [2, [33]], (4, 5))`

\*\*Explanation:\*\* Tuples are immutable, but lists inside a tuple can be modified, so the result will be `(1, [2, 33], (4, 5))`.

---

18. \*\*Given the dictionary, how do you change the value `'hello'` to `'world'`?\*\*

```python

d = {'a': [1, 2, {'b': 'hello'}]}

```

- A) `d['a'][2]['b'] = 'world'`

- B) `d['a']['b'] = 'world'`

- C) `d['b'] = 'world'`

- D) `d['a'][1] = 'world'`

\*\*Explanation:\*\* You can access `'b'` through `d['a'][2]['b']` and change the value to `'world'`.

---

19. \*\*What will be the output of this nested dictionary operation?\*\*

```python

d = {'a': {'b': {'c': [1, 2, 3]}}}

d['a']['b']['c'].append(4)

print(d)

```

- A) `{'a': {'b': {'c': [1, 2, 3, 4]}}}`

- B) `{'a': {'b': {'c': [1, 2, 3]}}}`

- C) `Error: Cannot append`

- D) `Error: KeyError`

\*\*Explanation:\*\* The append modifies the list inside the nested dictionary, resulting in `[1, 2, 3, 4]`.

---

20. \*\*Which of the following methods correctly adds an element to a set inside a dictionary?\*\*

```python

d = {'a': [1, 2], 'b': {3, 4}}

```

- A) `d['b'].add(5)`

- B) `d['b'].append(5)`

- C) `d['b'].insert(5)`

- D) `d['b'] = d['b'] + {5}`

\*\*Explanation:\*\* The `add()` method is used to add elements to a set.

---

21. \*\*What will be the result of this code?\*\*

```python

my\_tuple = (1, [2, 3, 4], 5)

my\_tuple[1].remove(3)

print(my\_tuple)

```

- A) `(1, [2, 4], 5)`

- B) `(1, [2, 3, 4], 5)`

- C) `Error: Tuples are immutable`

- D) `(1, [3, 2, 4], 5)`

\*\*Explanation:\*\* Lists inside a tuple can be modified, so `remove(3)` works, leaving `(1, [2, 4], 5)`.

---

22. \*\*What will be the output of the following nested list slicing?\*\*

```python

nested\_list = [1, 2, [3, 4, [5, 6]]]

result = nested\_list[2][2][0:1]

print(result)

```

- A) `[5]`

- B) `[5, 6]`

- C) `(5,)`

- D) `Error: Slicing does not work on nested lists`

\*\*Explanation:\*\* Slicing `[0:1]` returns a list containing the first element `[5]`.

---

23. \*\*What will be the output of the following code?\*\*

```python

my\_dict = {'a': {'b': {'c': 5}}}

print('d' in my\_dict['a']['b'])

```

- A) `True`

- B) `False`

- C) `Error: KeyError`

- D) `Error: TypeError`

\*\*Explanation:\*\* The key `'d'` does not exist in `my\_dict['a']['b']`, so the result is `False`.

---

24. \*\*Which of the following accesses the last element in the nested list?\*\*

```python

lst = [1, [2, [3, [4, [5, 6]]]]]

```

- A) `lst[-1][-1][-1][-1][-1]`

- B) `lst[1][1][1][1][1][1]`

- C) `lst[1][1][1][1][1][0]`

- D) `lst[1][1][1][0][1]`

\*\*Explanation:\*\* The deepest element, `6`, is at `[1][1][1][

1][1][1]`.

---

25. \*\*What will be the output of the following code involving nested tuples?\*\*

```python

t = (1, (2, 3, (4, 5)), 6)

result = t[1][2][1]

print(result)

```

- A) `4`

- B) `5`

- C) `6`

- D) `Error: IndexError`

\*\*Explanation:\*\* The element at `t[1][2]` is `(4, 5)`, and `[1]` gives `5`.

---

26. \*\*What will be the output of the following code?\*\*

```python

d = {'a': [1, 2, {'b': 3, 'c': 4}]}

result = d['a'][2].pop('b')

print(result)

print(d)

```

- A) `3`, `{'a': [1, 2, {'c': 4}]}`

- B) `{'b': 3}`, `{'a': [1, 2]}`

- C) `4`, `{'a': [1, 2, {'b': 3}]}`

- D) `Error: Pop does not work`

\*\*Explanation:\*\* The `pop('b')` removes and returns the value `3`.

---

27. \*\*What will be the output of this code involving tuple unpacking?\*\*

```python

t = (1, [2, 3], (4, 5))

a, b, c = t

b.append(6)

print(t)

```

- A) `(1, [2, 6], (4, 5))`

- B) `(1, [2, 3, 6], (4, 5))`

- C) `(1, [6], (4, 5))`

- D) `Error: Tuples cannot be modified`

\*\*Explanation:\*\* Since `b` is a list, appending `6` modifies the tuple to `(1, [2, 3, 6], (4, 5))`.

---

28. \*\*Which of the following correctly modifies the value `'d'` to `'e'` in a nested dictionary?\*\*

```python

d = {'a': {'b': {'c': 'd'}}}

```

- A) `d['a']['b']['c'] = 'e'`

- B) `d['a']['c'] = 'e'`

- C) `d['b']['c'] = 'e'`

- D) `d['a']['b'] = 'e'`

\*\*Explanation:\*\* The correct path to `'d'` is `['a']['b']['c']`.

---

29. \*\*What will be the result of the following code?\*\*

```python

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

del d['d']

```

- A) `Error: KeyError`

- B) `{'a': 1, 'b': 2, 'c': 3}`

- C) `{'a': 1, 'b': 2}`

- D) `Error: TypeError`

\*\*Explanation:\*\* Attempting to delete a non-existent key `'d'` raises a `KeyError`.

---

30. \*\*Which of the following correctly merges two dictionaries in Python 3.9+?\*\*

```python

d1 = {'a': 1}

d2 = {'b': 2}

```

- A) `d3 = {\*\*d1, \*\*d2}`

- B) `d3 = d1.update(d2)`

- C) `d3 = d1 + d2`

- D) `d3 = d1 \* d2`

\*\*Explanation:\*\* The `\*\*` operator merges two dictionaries by unpacking their key-value pairs.