Lists:

1. What is the output of the following code?

python

Copy

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

print(lst[::2])

a) [1, 3, 5] b) [2, 4] c) [5, 4, 3, 2, 1] d) [1, 2, 3]

1. Which method would you use to add an element to the end of a list? a) list.append() b) list.extend() c) list.insert() d) list.add()
2. What is the time complexity of searching for an element in an unsorted list? a) O(1) b) O(n) c) O(log n) d) O(n^2)
3. What will be the output of the following code?

python

Copy

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

lst[1:4] = [6, 7]

print(lst)

a) [1, 6, 7, 5] b) [1, 6, 7, 4, 5] c) [1, 2, 3, 4, 5, 6, 7] d) [1, [6, 7], 5]

1. Which of the following will create a new list with elements in reverse order without modifying the original list? a) list.reverse() b) reversed(list) c) list[::-1] d) Both b and c
2. What is the output of the following code?

python

Copy

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

print(lst.pop(1))

print(lst)

a) 1, [2, 3, 4, 5] b) 2, [1, 3, 4, 5] c) 5, [1, 2, 3, 4] d) IndexError

1. Which method would you use to remove all occurrences of a specific element from a list? a) list.remove() b) list.pop() c) del list[element] d) None of the above
2. What will be the output of the following code?

python

Copy

lst1 = [1, 2, 3]

lst2 = lst1

lst2.append(4)

print(lst1)

a) [1, 2, 3] b) [1, 2, 3, 4] c) [4, 1, 2, 3] d) [1, 2, 3, [4]]

1. Which of the following will flatten a list of lists into a single list? a) [item for sublist in list\_of\_lists for item in sublist] b) sum(list\_of\_lists, []) c) reduce(lambda x, y: x + y, list\_of\_lists) d) All of the above
2. What is the output of the following code?

python

Copy

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

print(lst[3:1:-1])

a) [4, 3] b) [3, 2] c) [5, 4] d) []

Lists:

1. Answer: a) [1, 3, 5] Explanation: lst[::2] returns every second element from the start of the list.
2. Answer: a) list.append() Explanation: append() adds a single element to the end of a list.
3. Answer: b) O(n) Explanation: Searching an unsorted list requires checking each element in the worst case.
4. Answer: a) [1, 6, 7, 5] Explanation: lst[1:4] = [6, 7] replaces elements at indices 1, 2, and 3 with 6 and 7.
5. Answer: d) Both b and c Explanation: Both reversed(list) and list[::-1] create a reversed version without modifying the original.
6. Answer: b) 2, [1, 3, 4, 5] Explanation: lst.pop(1) removes and returns the element at index 1 (which is 2).
7. Answer: d) None of the above Explanation: To remove all occurrences, you'd typically use a list comprehension or filter.
8. Answer: b) [1, 2, 3, 4] Explanation: lst2 is a reference to lst1, so modifying lst2 also modifies lst1.
9. Answer: d) All of the above Explanation: All these methods can be used to flatten a list of lists.
10. Answer: a) [4, 3] Explanation: lst[3:1:-1] starts at index 3, moves towards index 1 (exclusive) in reverse order.