W3-10020-Recursion

Number of participants: 36

1. Any questions so far?

1 respondent

I dont understand how the tower of hanoi works in both pseudocode and python code

2. What is a return value? Where is "return"?

9 respondents

- :O
- The return keyword is to exit a function and return a value.
- the answer
- the output of the function/ return at the end of the function definition
- It's the value that a function will output
- RETURN TO WHERE IS FUNCTION IS CALLED
- None
- output of a function
- output

3. A function can call itself

12 correct answers
out of 12 respondents

✓	True	100%	12 votes
	False	0%	0 votes

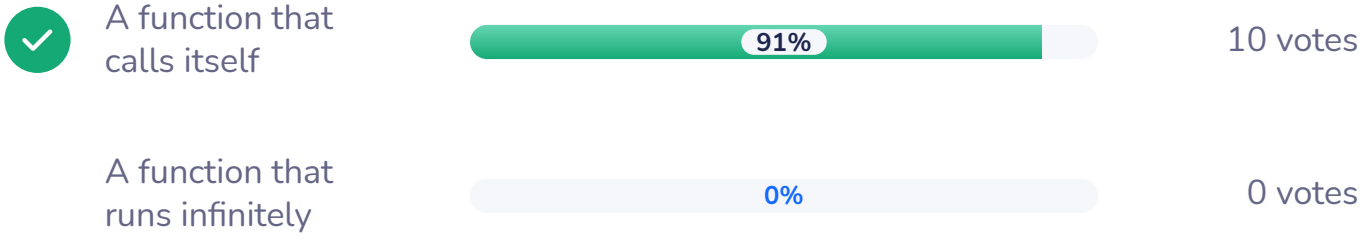
4. Let's say we have a function def f(arg): .... I can call this function using print(f)

14 correct answers
out of 15 respondents

	True	7%	1 vote
✓	False	93%	14 votes

5. What is recursion?

10 correct answers
out of 11 respondents



6. What happens if the base case is missing in a recursive function?

13 correct answers
out of 16 respondents

	The program will execute successfully	0%	0 votes
	The function will run only once	6%	1 vote
✓	The program may enter an infinite recursion and crash	81%	13 votes
	It will automatically terminate	13%	2 votes

7. How many times is f() called in the following code?

5 correct answers out of 18 respondents

	6	17%	3 votes
	7	28%	5 votes
	8	22%	4 votes
	10	6%	1 vote
✓	15	28%	5 votes

8. What is the time complexity of the naive recursive Fibonacci function?

0 correct answer
out of 1 respondent

```python
def nth_fibonacci(n):

    # Base case: if n is 0 or 1, return n
    if n <= 1:
        return n

    # Recursive case: sum of the two preceding Fibonacci numbers
    return nth_fibonacci(n - 1) + nth_fibonacci(n - 2)

n = 5
result = nth_fibonacci(n)
print(result)
```

	O(N)	0%	0 votes
	O(N log N)	0%	0 votes
✓	O(2^N)	0%	0 votes
	O(N^2)	100%	1 vote

9. What is the space complexity of the naive recursive Fibonacci function?

0 correct answer
out of 0 respondent

```python
def nth_fibonacci(n):

    # Base case: if n is 0 or 1, return n
    if n <= 1:
        return n

    # Recursive case: sum of the two preceding Fibonacci numbers
    return nth_fibonacci(n - 1) + nth_fibonacci(n - 2)

n = 5
result = nth_fibonacci(n)
print(result)
```

✓	O(N)	0%	0 votes
	O(N log N)	0%	0 votes
	O(2^N)	0%	0 votes
	O(N^2)	0%	0 votes

10. Which of the following problems is best solved using recursion?

0 correct answer
out of 0 respondent

✓	Heap Search	0%	0 votes
	Linear Search	0%	0 votes
	Insertion Sort	0%	0 votes
	Bubble Sort	0%	0 votes

11. Which step in Merge Sort is responsible for most of the computational cost?

0 correct answer
out of 0 respondent

	Dividing the array into two halves	0%	0 votes
✓	Merge two arrays	0%	0 votes
	Finding the midpoint	0%	0 votes
	Recursive function calls	0%	0 votes

12. How many recursive calls are made when sorting a list of 8 elements using merge sort (not counting the merge steps)?

0 correct answer
out of 0 respondent

	7	0%	0 votes
	8	0%	0 votes
✓	15	0%	0 votes
	16	0%	0 votes

13. What is the space complexity of Merge Sort? The one we implemented in class.

0 correct answer
out of 0 respondent

✓	O(N)	0%	0 votes
	O(N^2)	0%	0 votes
	O(N log N)	0%	0 votes
	O(1)	0%	0 votes

14. Merge Sort always divides the array into:

0 correct answer
out of 0 respondent

✓	Two equal parts	0%	0 votes
	One sorted half and one unsorted half	0%	0 votes
	Two unequal parts	0%	0 votes
	Depends on the array size	0%	0 votes