

✓ **Congratulations! You passed!**  
TO PASS 80% or higher

Keep Learning

GRADE  
100%

## Practice Quiz: Recursion

TOTAL POINTS 5

1. What is recursion used for?

1 / 1 point

- ☐ Recursion is used to create loops in languages where other loops are not available.
- ☐ We use recursion only to implement mathematical formulas in code.
- ☐ Recursion is used to iterate through sequences of files and directories.
- ☒ Recursion lets us tackle complex problems by reducing the problem to a simpler one.

✓ **Correct**

You nailed it! By reducing the problem to a smaller one each time a recursive function is called, we can tackle complex problems in simple steps.

2. Which of these activities are good use cases for recursive programs? Check all that apply.

1 / 1 point

- ☒ Going through a file system collecting information related to directories and files.

✓ **Correct**

Right on! Because directories can contain subdirectories that can contain more subdirectories, going through these contents is a good use case for a recursive program.

- ☐ Creating a user account.

- ☐ Installing or upgrading software on the computer.

- ☒ Managing permissions assigned to groups inside a company, when each group can contain both subgroups and users.

✓ **Correct**

You got it! As the groups can contain both groups and users, this is the kind of problem that is a great use case for a recursive solution.

- ☐ Checking if a computer is connected to the local network.

3. Fill in the blanks to make the `is_power_of` function return whether the number is a power of the given base. Note: base is assumed to be a positive number. Tip: for functions that return a boolean value, you can return the result of a comparison.

1 / 1 point

```
1 def is_power_of(number, base):
2     # Base case: when number is smaller than base.
3     if number < base:
4         # If number is equal to 1, it's a power (base**0).
5         return True if number == 1 else False
6
7     # Recursive case: keep dividing number by base.
8     return is_power_of(number/base, base)
9
10 print(is_power_of(8,2)) # Should be True
11 print(is_power_of(64,4)) # Should be True
12 print(is_power_of(70,10)) # Should be False
```

Run

Reset

True  
True  
False

✓ **Correct**

Nice job! You've made the code check for powers of numbers by reducing the problem to a smaller one.

4. The `count_users` function recursively counts the amount of users that belong to a group in the company system, by going through each of the members of a group and if one of them is a group, recursively calling the function and counting the members. But it has a bug! Can you spot the problem and fix it?

1 / 1 point

```
1 def count_users(group):
2     count = 0
```

```
3 | for member in get_members(group):
4 |     if is_group(member):
5 |         count += count_users(member)
6 |     else: count += 1
7 | return count
8
9 | print(count_users("sales")) # Should be 3
10 | print(count_users("engineering")) # Should be 8
11 | print(count_users("everyone")) # Should be 18
```

Run  
Reset

3  
8  
18

✓ Correct

Well done, you! You spotted the problem that was causing groups to be counted when we only wanted to count users!

5. Implement the `sum_positive_numbers` function, as a recursive function that returns the sum of all positive numbers between the number `n` received and 1. For example, when `n` is 3 it should return `1+2+3=6`, and when `n` is 5 it should return `1+2+3+4+5=15`.

1 / 1 point

```
1 | def sum_positive_numbers(n):
2 |     if n < 1:
3 |         return 0
4 |     return n + sum_positive_numbers(n - 1)
5
6 | print(sum_positive_numbers(3)) # Should be 6
7 | print(sum_positive_numbers(5)) # Should be 15
```

Run  
Reset

6  
15

✓ Correct

Here is your output:

6  
15

Great work! You've really nailed writing recursive functions!