Question 1: Here are the adjacency lists for an undirected graph:

- Node A: { C, D }
- Node B: { F }
- Node C: { A, E }
- Node D: { A }
- Node E: { C }
- Node F: { B }

I call **DFS(A)**. Suppose the DFS's action is to print the character of the node before it visits its children, and after it visits its children. To be precise, each node that gets visited will have its character printed exactly twice. What are the characters printed, in the order in which they are printed (just enter a single string with no spaces)?

Question 2: How many connected components are in this graph?

Answer to DFS Clicker Question

Question 1: Let's simply walk through it. Here is the order of the DFS calls, indented by recursion level:

```
A             - Print A, recursively call on C and D
| C           - Print C, recursively call on A and E
| | A         - Already visited
| | E         - Print E, recursively call on C
| | | C       - Already visited
| | E         - E returns -- print E
| C           - C returns -- print C
| D           - Print D, recursively call on A
| | A         - Already visited
| D           - Print D
A             - Print A
```

So the answer is "ACEECDDA".

Question 2: So A, C, D, and E are in one component. It's pretty clear that B and F are in another component. The answer is two.