R-4.5

Draw the recursion trace for the execution of function PuzzleSolve(3, S, U) (Code Fragment 4.14), where S is empty and U = { a, b, c, d }

U = {a, b, c, d}; S = " "; k = 3

for e in array: first iteration of {a, b, c, d}
- e = a
- k = 3
- add a to S
- S = a
- remove a from U
- U = {b, c, d}
- k != 1
- recursive call(3 - 1, a, {b, c, d})
    - for e in array: first iteration of {b, c, d}
        - e = b
        - k = 2
        - add b to S
        - S = ab
        - remove b from U
        - U = {c, d}
        - k != 1
        - recursive call (2 - 1, ab, {c, d})
            - for e in array: first iteration of {c, d}
                - e = c
                - k = 1
                - add c to S
                - S = abc
                - remove c from U
                - U = {d}
                - k == 1
                - **solve puzzle**
                    - return solution if solved, otherwise continue
                - remove c from S
                - S = ab
                - add c to U
                - U = {d, c}
                - complete for loop
            - for e in array: second and last iteration of {c, d}
                - e = d
                - k = 1
                - add d to S

            - S = abd
            - remove d from U
            - U = {c}
            - solve puzzle
                - return if solved, otherwise continue
            - remove d from S
            - S = ab
            - add d to U
            - U = {c, d}
            - complete for loop
        - break because iteration of {c, d} complete
- for e in array: second iteration of {b, c, d}
    - e = c
    - k = 2
    - add c to S
    - S = ac
    - remove c from U
    - U = {b, d}
    - k != 1
    - recursive call (2 - 1 , ac, {b, d}
        - for e in array: first iteration of {b, d}
            - e = b
            - k = 1
            - add b to S
            - S = acb
            - remove b from U
            - U = {d}
            - k == 1
            - **solve puzzle**
                - return solution if solved, otherwise continue
            - remove b from S
            - S = ac
            - add b to U
            - U = {d, b}
            - complete for loop
        - for e in array second and last iteration of {b, d}
            - e = d
            - follow same pattern ….
            - complete for loop
        - break because iteration of {b, d} complete
- for e in array: third and last iteration of {b, c, d}
    - e = d
    - k = 2
    - follow same pattern
    - recursive call (2 - 1, ad, {b, c}
        - same pattern
            - complete for loop

        - same pattern
            - complete for loop
        - break because iteration of {b, c} complete
    - break because iteration of {b, c, d} complete

for e in array: second iteration of {a, b, c, d}
- continue pattern

… continue pattern until parent for loop complete.