LeetCode, 21. Merge Two Sorted Lists

Initial list states are:

List 1 = ① → ② → ④

List 2 = ① → ③ → ④

Call 1 / Depth 1:

Initial state:

pList1 = ①

pList1 → next = ②

pList2 = ①

pList2 → next = ③

No swap → State stays the same

After the swap:

→ pList1 → next = (②, ①)

mergeTwoLists

→ pList1 = ①, pList1 → next = ①

Call 2 / Depth 2:

Initial state:

pList1 = ②

pList1 → next = ④

pList2 = ①

pList2 → next = ③

Swap →

After the swap:

pList1 = ①

pList1 → next = ③

pList2 = ②

pList2 → next = ④

→ pList1 → next = (③, ②)

mergeTwoLists

→ pList1 = ①, pList1 → next = ② → return pList1 = ①

Call 3 / Depth 3:

Initial state:

pList1 = ③

pList1 → next = ④

pList2 = ②

pList2 → next = ④

Swap →

After the swap:

pList1 = ②

pList1 → next = ④

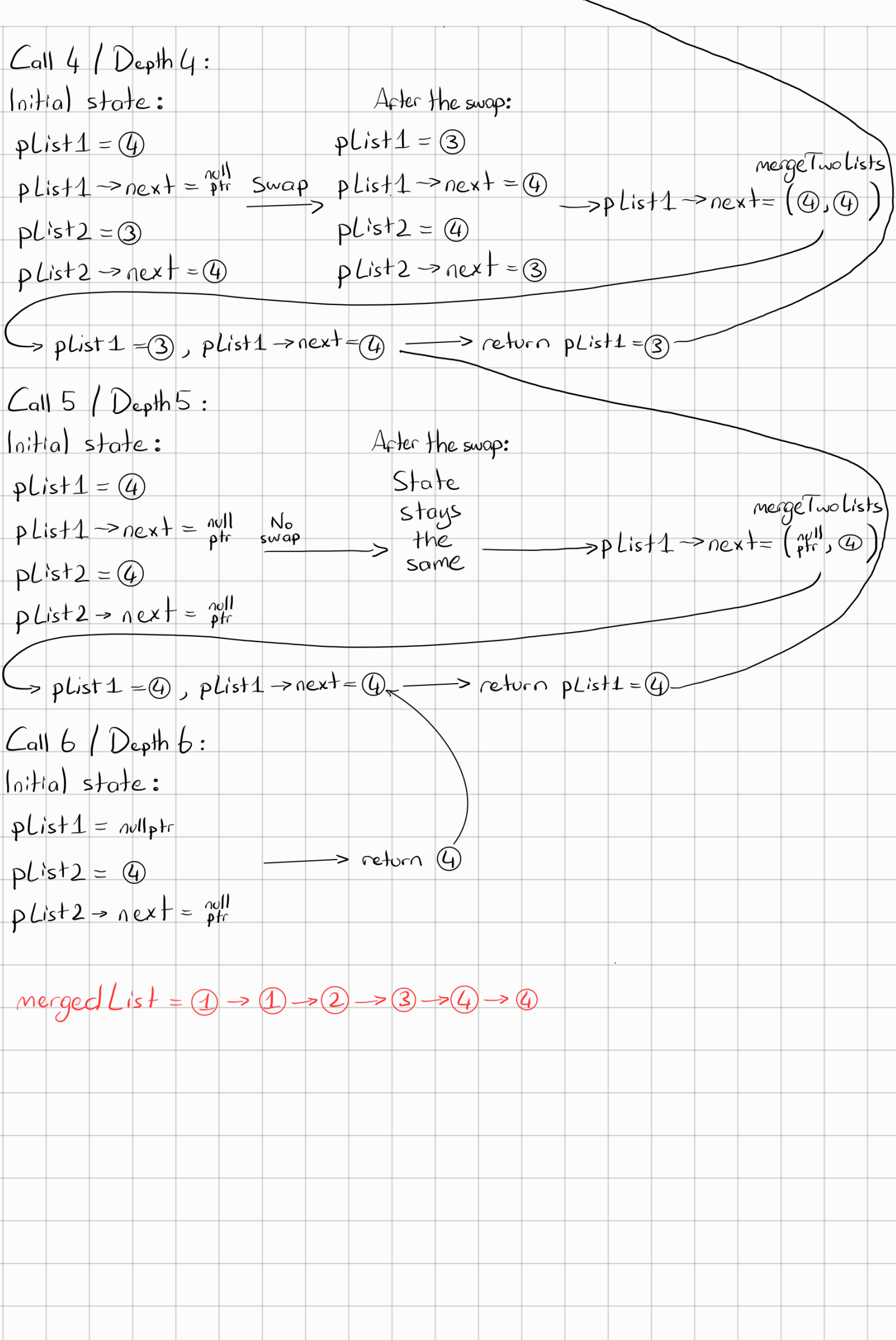
pList2 = ③

pList2 → next = ④

→ pList1 → next = (④, ③)

mergeTwoLists

→ pList1 = ②, pList1 → next = ③ → return pList1 = ②


Call 4 / Depth 4:
Initial state:
pList1 = ④
pList1->next = null ptr
pList2 = ③
pList2->next = ④
Swap
After the swap:
pList1 = ③
pList1->next = ④
pList2 = ④
pList2->next = ③
mergeTwoLists
->pList1->next = (④, ④)
pList1 = ③, pList1->next = ④
return pList1 = ③
Call 5 / Depth 5:
Initial state:
pList1 = ④
pList1->next = null ptr
pList2 = ④
pList2->next = null ptr
No swap
After the swap:
State stays the same
mergeTwoLists
->pList1->next = (null ptr, ④)
pList1 = ④, pList1->next = ④
return pList1 = ④
Call 6 / Depth 6:
Initial state:
pList1 = nullptr
pList2 = ④
pList2->next = null ptr
return ④
mergedList = ① -> ① -> ② -> ③ -> ④ -> ④