

# Partition Linked List

input: 5

original list

3 8 5 10 2 1

expected output

3 2 1 8 5 10

everything that is lower than the input goes to the start of the list.

we can create 2 LL and then join it.

input: 5

temp

Small HEAD

3

8 5 10 2 1

BIG HEAD

8

5 10 2 1

8

5 10 2 1

8

5 10 2 1

8

5 10 2 1

8

5 10 2 1

8

5 10 2 1

8

5 10 2 1

to remove the 2 we have to point 10 to 1

and point 3 to 2 and 2 to null

do it with the "previous approach" where you always work on NEXT so you can do NEXT -> NEXT on the temp

SMALL TA

later point 2 to 1

and 10 to null

and to wrap point 1 to 8 the last SMALLy to first BIG

small list = temp < input

3 2 1

8 5 10

3 2 1 8 5 10

Goal:

Remove  
ALL SMALL

$x = 5$       SMALL:  $\leq x$

BIG HEAD: is the first temp bigger than  $x$

✓ denotes the SMALL TAIL

3 8 5 10 2 1

3 8 5 10 2 1  
BIG HEAD

3 8 5 10 2 1

3 8 5 10 2 1

3 8 5 10 2 1  
Next iteration don't change Temp  
points to 1 which is the next of next

3 8 5 10 2 1  
→ points to NULL because there's no next

3 8 5 10 2 1  
no more next so we stop

3 8 5 10 2 1

No Luck I gave up and watched an explanation.

my code and approach worked for like 90% of the cases but not all.

current  $x=5$   
↳ just iterate normally  
do not modify it.  
point current to small  
or big prev next

create two dummy HEAD  
↳ prev

SMALL prev  $\rightarrow 3 \rightarrow 2 \rightarrow 1$  set SMALL prev next to BIG DUMMY next  
BIG prev  $\rightarrow 8 \rightarrow 5 \rightarrow 10$  ⚠ to merge these new lists

{ at this moment 10 next points to 2. Set BIG prev to NULL to define it as our new TAIL

↳ the function returns SMALL DUMMY next which skips the  $\emptyset$

3 8 5 10 2 1  
3 8 5 10 2 1  
3 8 5 10 2 1  
3 8 5 10 2 1  
3 8 5 10 2 1  
3 8 5 10 2 1