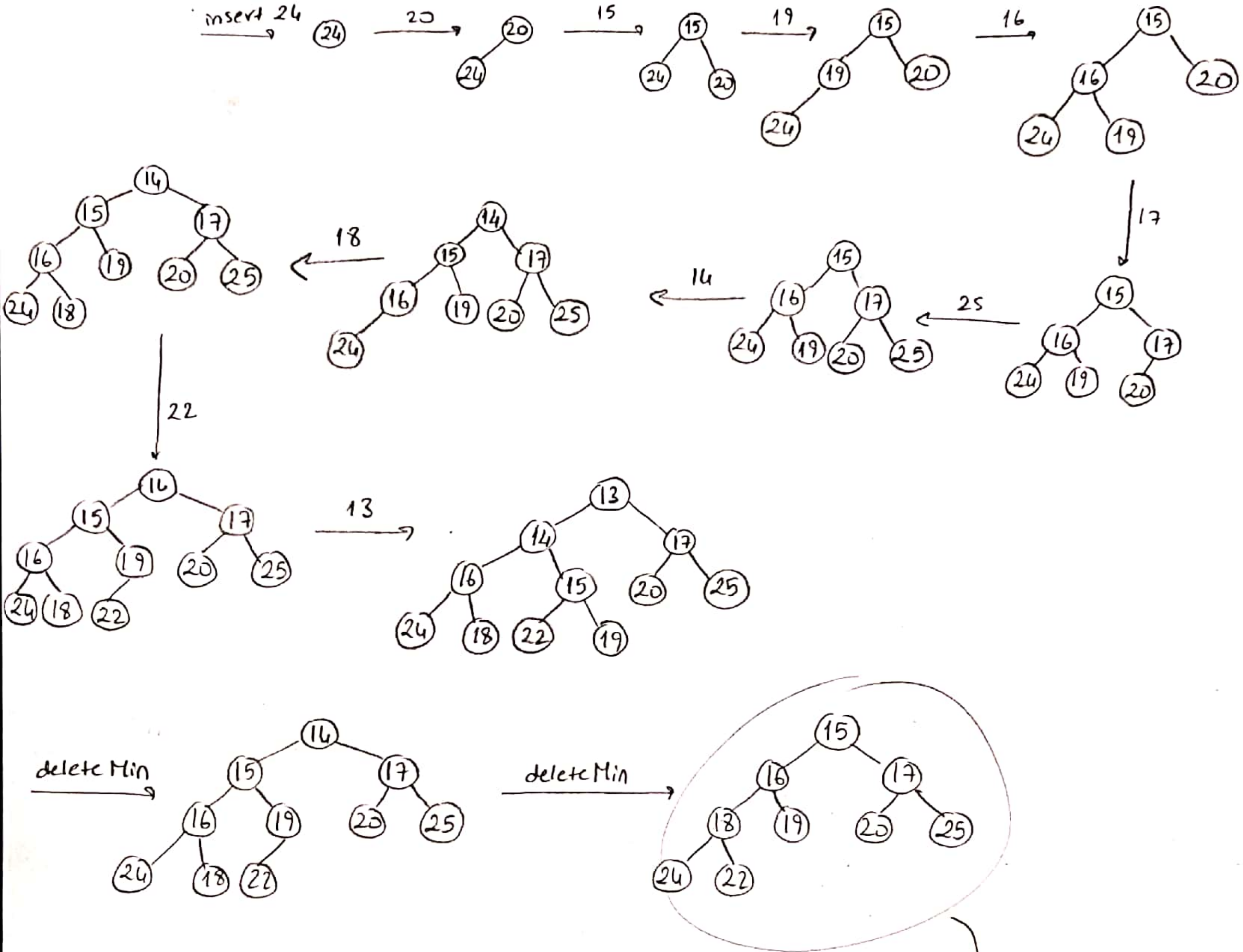
Ayda Yurtoğlu

21903153 Section 1

Q1) a) binary min heap



b) Consides the binary min heap we acquired in part (a). (the very last result)

preorder traversal: 15, 16, 18, 24, 22, 19, 17, 20, 25

↳ not sorted (for example 24>22 but this list put 24 before 22)

inorder traversal: 24, 18, 22, 16, 19, 15, 20, 17, 25

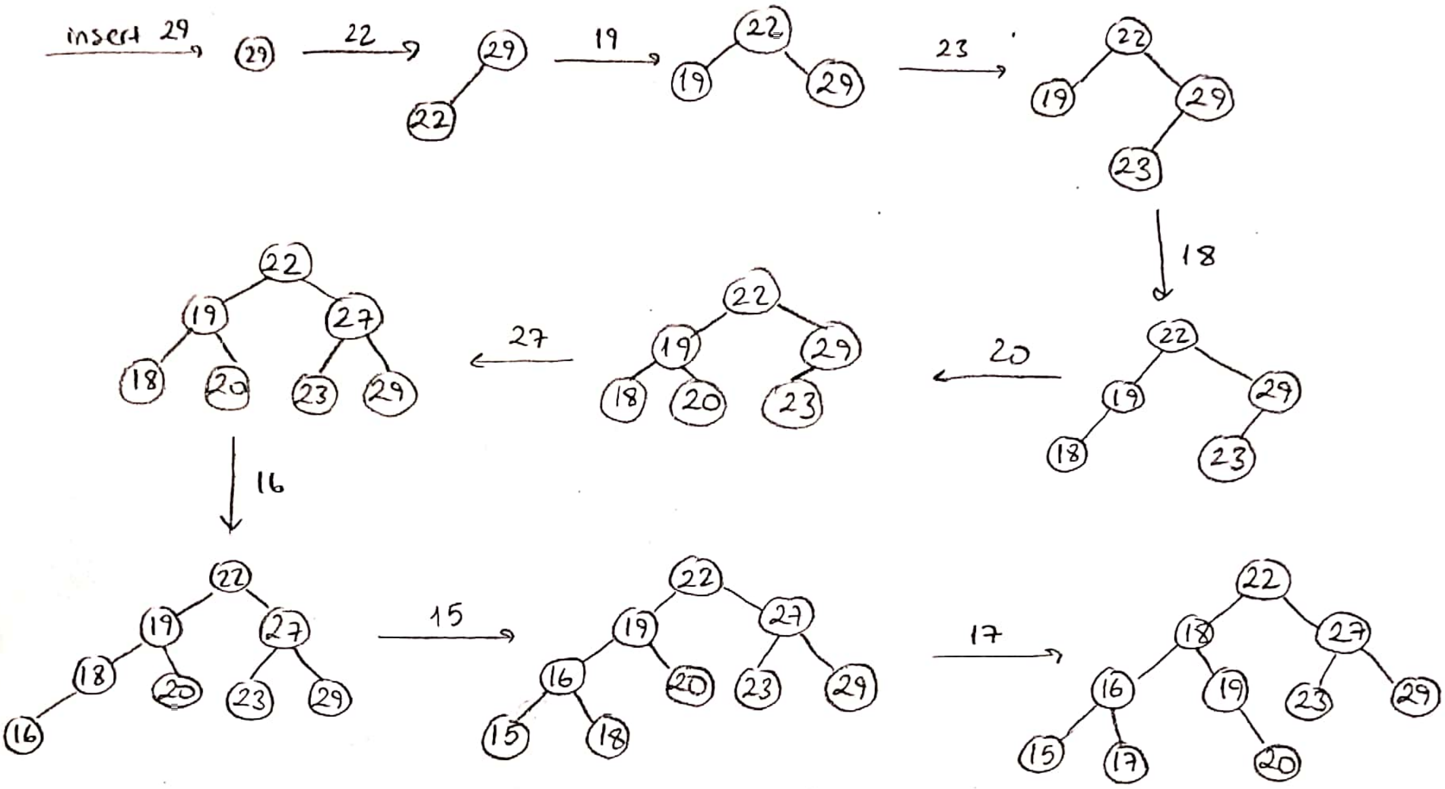
↳ not sorted (take this part for example)

postorder traversal: 24, 22, 18, 19, 16, 20, 25, 17, 15

↳ not sorted (this part for example)

So, we can't obtained the sorted array with any of these traversals.

c)


insert 29
29
22
29
22
19
22
19
29
23
22
19
29
23
18
22
19
29
18
23
20
22
19
29
18
20
23
27
22
19
27
18
20
23
29
16
22
19
27
18
20
23
29
16
15
22
19
27
16
20
23
29
15
18
17
22
18
27
16
19
23
29
15
17
20

Ayda Yurtoğlu
21903153
Section 1

Question 3 :

Executing the simulation by starting from 1 printer until we find the minimum number of printers is not efficient, because we might need one million printers and in this case, we'd have to call the function for one million times. Average waiting time with 1 printer will the maximum result we can obtain for different numbers of printers. So, beginning by 1 is not a good idea. We need to find some other number closest to the minimum number of printers we require. We can maybe calculate this number by taking the root of number of request and begin our function with this number. Depending on our result we can increase or decrease the number of printers. For one million, we will calculate with one thousand printers which will be a lot faster than with one printer.

Another option might be a similar approach to binary search. We can take the half of the number of requests and try that number as our number of printers. If average waiting time is greater than the wanted maximum average waiting time, we can take the ¾ of that number and try again. If it's not, to find the exact minimum we can decrease the number of printers until our result doesn't exceed the maximum average waiting time.