William Clark

a.) $i = 2$ at the start, loop updates i as $i = i^2$ each loop

exponential growth $\rightarrow$ 2, 4, 16 ....

i after j iterations is $i_j = 2^{2^j}$

loop stops when $2^{2^j} \geq n$

Take log twice

$$2^j \geq \log_2(n)$$

$$j \geq \log_2(\log_2(n))$$

Each iteration is $O(1)$ so total runtime is $\Theta(\log \log n)$

b. First loop iterates $i = 1$ to $i = n$ with i increasing by 1, so n times

If statement checks if i is divisible by $\sqrt{n}$, so there are $\sqrt{n}$ of such values of i

The inner loop runs i^3 times per i

$$\sum_{j=1}^{\sqrt{n}} (j\sqrt{n})^3 \quad \leftarrow \text{Summing through innerloop}$$

$$\sum_{j=1}^{\sqrt{n}} j^3 n^{3/2}$$

substitute $\sqrt{n}$ for k

$$n^{\frac{3}{2}} \sum_{j=1}^{k} j^3 = \Theta(k^4) \quad \leftarrow \text{Sum algorithm for cubes}$$

plug $\sqrt{n}$ back in

$$n^{\frac{3}{2}} \sum_{j=1}^{\sqrt{n}} j^3 = \Theta(n^2)$$

$$n^{\frac{3}{2}} \cdot \Theta(n^2) = n^{3.5} = \underline{\Theta(n^3)}$$

c.) First two loops are standard incrementing nested for loops so run at O(

The inner for loop is logarithmic as it doubles the value of m each step → 1, 2, 4, 8 .... n. This loop has O(log n) complexity
However, ~~this last~~ the inner loop is restricted by an if statement,
For calculating time complexity, we have to assume worse case Scenario which is that A[k] == i at most once for each i and k pair (n^2), meaning total complexity is the multiple of the two outer loops and the inner, or $\Theta(n^2 \log n)$

d.) Outer loop iterates n times,

Resize occurs when i == size

Siz values go $10 \to \frac{3}{2} \cdot 10, \left(\frac{3}{2}\right)^2 \cdot 10 \dots$

After x resizes, say size ≥ n

$$10\left(\frac{3}{2}\right)^x \geq n$$

$x = \log_{3/2}\left(\frac{n}{10}\right) = O(\log n)$ so resizing happens $O(\log n)$ times

To find the size of the copies we can estimate it as O(n)
~~we can use the formula to approximate the sum of the geometric series above.~~
Since that is roughly the last term and a geometric sequence with a large T value will be carried by the last term.

So even though there are log n resizes, the cost is only O(n), making O(n) the time complexity

William Clark

Q2a)

in1 = 1,2,3,4 in2 = 5,6

llrec(1,5)

return 1 ↳ llrec(5,2)

return 5 ↳ llrec(2,6)

return 2 ↳ llrec(6,3)

return 6 ⟶ llrec(3,4)

return 3 ↳ llrec(4, nullptr)

Base case: return 4

Final List: 1→5→2→6→3→4→nullptr

b) in1 = null ptr, in2 = 2

First if is triggered, it returns in2 which is

Final List: 2→nullptr