CS104 HW1

1.a. After 0 iterations, $i=2$
After 1 iteration, $i=4$
After 2 iterations, $i=16$
After 3 iterations, $i=256$

this loop runs until $i \geq n$
i grows from $2 \to 4 \to 16 \to 256$
$= 2^1 \to 2^2 \to 2^4 \to 2^8$
$= 2^{2^0} \to 2^{2^1} \to 2^{2^2} \to 2^{2^3}$

From the pattern, we see that after k iterations, $i = 2^{(2^k)}$
Loop runs until $i \geq n \to$ until $2^{(2^k)} \geq n$
$\to 2^k \geq \log_2 n \to$ loop runs until $k \geq \log_2(\log_2(n))$
In other words, loop runs $\log_2(\log_2(n))$ times
$\therefore$ Runtime is $\Theta(\log\log n)$

b. Outer loop runs n times
↳ For each iteration, inner loop runs if i is divisible by $\sqrt{n}$
i is divisible by $\sqrt{n}$ when $i = \sqrt{n}, 2\sqrt{n}, 3\sqrt{n}, \ldots, n$
↳ When inner loop runs, it runs i^3 times
In these cases, $i^3 = (\sqrt{n})^3, (2\sqrt{n})^3, (3\sqrt{n})^3, \ldots, n^3$
From this, the total num. of operations is given by $\sum_{k=1}^{\sqrt{n}} (k\sqrt{n})^3 = \sum_{k=1}^{\sqrt{n}} k^3 n^{\frac{3}{2}}$
$= n^{\frac{3}{2}} \sum_{k=1}^{\sqrt{n}} k^3 = n^{\frac{3}{2}} \left(\frac{\sqrt{n}(\sqrt{n}+1)}{2}\right)^2 = n^{\frac{3}{2}}\left(\frac{n^{\frac{4}{2}}}{4} + \frac{n^{\frac{3}{2}}}{2} + \frac{n}{4}\right) \to$ approx. $n^{\frac{7}{2}}$ operations
$\therefore$ Runtime is $\Theta(\sqrt{n^7})$

c. Outer loop runs n times, middle loop runs n times
↳ If $A[k] == i$, inner loop will run $\log_2 n$ times, as m doubles on each loop until $m \geq n$.
However, since $A[]$ is not changing, then $A[k]$ can only $== i$ at most n times, since $1 \leq k \leq n$
↳ Therefore, the inner loop will run at most n times
This gives us a worst-case number of operations of $n^2 + n\log_2 n$
↳ n^2 from the outer-middle loops and $n\log n$ from the inner loop
$n\log n$ term is dominated by n^2 term
$\therefore$ Runtime is $\Theta(n^2)$

d. Outer loop runs n times

↳ If $i == size$, an array with a new size of $\frac{3}{2}size$ is allocated and initialized → this process takes size num. of operations

The size of the array is $10 \to 15 \to 22 \to 33 \ldots$ until the size of the array is $\geq n$.

↳ Array size grows geometrically by $10\left(\frac{3}{2}\right)^k$, where k = num. of resizings

Array stops resizing when $10\left(\frac{3}{2}\right)^k \geq n \to \left(\frac{3}{2}\right)^k \geq \frac{n}{10} \to k \geq \log_{\frac{3}{2}}\left(\frac{n}{10}\right)$

$\to k \geq \log n \to$ there will be approx $\log n$ resizings

Again, each resizing takes $10\left(\frac{3}{2}\right)^k$ operations, and there are $\log n$ resizings

↳ Thus the total num. of operations involved in resizing is given by $\sum_{k=0}^{\log n} 10\left(\frac{3}{2}\right)^k$

$= \frac{10\left(1-\left(\frac{3}{2}\right)^{\log n}\right)}{1-\frac{3}{2}} = \frac{10(1-n)}{-\frac{1}{2}} = \frac{10-10n}{-\frac{1}{2}} = 20n - 20 \to$ simplify to $\Theta(n)$

Outer loop is $\Theta(n)$ and resizing takes $\Theta(n)$ operations $\to \Theta(n) + \Theta(n) = \Theta(n)$

∴ Runtime is $\Theta(n)$

2\. a. in1 = 1, 2, 3, 4   in2 = 5, 6

1 llrec(1, 5) → 1 = 1 → 5 → 2 → 6 → 3 → 4, returns 1

2 llrec(5, 2) → 5 = 5 → 2 → 6 → 3 → 4, returns 5

3 llrec(2, 6) → 2 = 2 → 6 → 3 → 4, returns 2

4 llrec(6, 3)   6 = 6 → 3 → 4, returns 6

5 llrec(3, nullptr) returns 3 (or 3 → 4 for full linked list)

∴ llrec(in1, in2) returns 1, 5, 2, 6, 3, 4

b. in1 = nullptr   in2 = 2

1 llrec(nullptr, 2) returns 2

∴ llrec(in1, in2) returns 2