Problem 3

Part a) This function is $O(\log(\log n))$. To show this, let's choose thresholds of n that change the number of times the while loop is run.

when $n = 2$, the loop runs 0 times

$n = 4$, it runs 1 times

$n = 16$, it runs 2 times

$n = 256$, it runs 3 times

⋮

Seeing how this is exponential, we can take the log to try to see a pattern.

Thus,

log n	loops
1	0
2	1
4	2
8	3

⋮

The resulting pattern is still exponential, so we can take a second log to show that

log (log n)	loops
0	0
1	1
2	2
3	3

⋮

Thus, because every time $\log(\log n)$ is incremented another loop occurs, and because the while loop is $O(1)$, this function is $\boxed{O(\log(\log n))}$.

b) The if statement happens $\sqrt{n}$ times because there are $n/\sqrt{n}$ values that i could take to have a value of zero with mod $\sqrt{n}$. Thus, the for loop happens $\sqrt{n}$ times with increasing values of i.

If we mathematically write it out, the time complexity is

$$O((\sqrt{n})^3 + (2\sqrt{n})^3 + (3\sqrt{n})^3 + (4\sqrt{n})^3 + \cdots + (\sqrt{n}\sqrt{n})^3)$$
$$= O(n^{3/2} + 8n^{3/2} + 27n^{3/2} + 64n^{3/2} + \cdots + n^3)$$
$$= O(n^{3/2}(1 + 8 + 27 + 64 + \cdots + n^{3/2}))$$
$$= O\left(n^{3/2} \times \frac{(n^{1/2})^2(n^{1/2}+1)^2}{4}\right)$$
$$= O\left(n^{3/2} \times \frac{n(n + 2\sqrt{n} + 1)}{4}\right)$$
$$= \boxed{O(n^{7/2})}$$

c) For the if statement, it will only run a max of n times, because in the worst case scenario there exists a true (i,k) pair that satisfies the if statement for every element of array A.

The for loop is $O(\log n)$ because each loop is $O(1)$, and every loop m is being doubled.

Thus, the final time complexity is $O(n \log n)$.

d) Let us count the time complexity of the for loop.
The first 9 times it loops, it is $O(1)$, because i has not reached size yet.
The 10th time it loops, it is $O(1) + O(10)$, because after constructing the new array, it copies the previous one over, and then squares the current element.
Similarly, from loops 11-14, it is $O(1)$, and on loop 15, it is $O(1) + O(15)$ due to the newly sized array.
The for loop only runs n times, so we can sum the n $O(1)$'s into $O(n)$.
Furthermore, the if statement only happens $\log_{3/2}(\frac{n}{10})$ times, because that is the number of times that i == size.

Thus, our time complexity looks like

$$O(n) + O(10) + O(10 \times \tfrac{3}{2}) + O(10 \times (\tfrac{3}{2})^2) + \cdots + O(10 \times (\tfrac{3}{2})^{\log_{3/2}(\frac{n}{10})})$$

$$= O(n) + O(10 \times (1 + \tfrac{3}{2} + (\tfrac{3}{2})^2 + (\tfrac{3}{2})^3 + \cdots + (\tfrac{3}{2})^{\log_{3/2}(\frac{n}{10})}))$$

$$= O(n) + O\left(10 \times \frac{(1 - (\frac{3}{2})^{\log_{3/2}(\frac{n}{10})})}{(1 - \frac{3}{2})}\right)$$

$$= O(n) + O\left(10 \times \frac{(1 - \frac{n}{10})}{-\frac{1}{2}}\right)$$

$$= O(n) + O\left(\frac{10 - n}{-1/2}\right)$$

$$= O(n) + O(2n - 20)$$

$$= O(3n - 20)$$

$$= \boxed{O(n)}$$