problem #3: Kevin Chen

a. Prove $T(n) = \Theta(n^{\log_y(x)})$, when $z < \log_y(x)$, given that $T(n) = xT(n/y) + n^z$, we have the form $T(n) = aT(n/b) + f(n)$ with $a = x$, $b = y$, $f(n) = n^z$.
Now, applying master theorem, we compare $f(n)$ with $n^{\log_b(a)}$, which in this case is $n^{\log_y(x)}$. Since we have the condition $z < \log_y(x)$, it's implied that $f(n)$ grows slower than $n^{\log_y(x)}$, or $f(n) = O(n^{\log_y(x)-\epsilon})$ for $\epsilon > 0$. This aligns with case 1 of the Master Theorem, where $f(n)$ is asymptotically less than $n^{\log_b(a)}$, and recursive $aT(n/b)$ dominates the solution.
According to master theorem, $T(n)$ is dominated by the cost of the recursive calls, leading us to conclude to $T(n) = \Theta(n^{\log_b(a)})$, and substituting our values gives $T(n) = \Theta(n^{\log_y(x)})$, if $z < \log_y(x)$.
This is in agreement with the original answer which completes the proof.

b. If $z = \log_y(x)$, prove $T(n) = \Theta(n^{\log_y(x)} \cdot \log_y n)$
Given that $f(n) = n^z$, we can say that $f(n) = n^{\log_y(x)}$. Master theorem requires us to express this as $n^{\log_b(a)}$. And, $b = y$ since we're dividing by y with each recursive call. And x recursive calls are made, so $a = x$.
Thus, $z = \log_b(a)$.
Now, case 2 of master theorem applies when $f(n) = \Theta(n^{\log_b(a)} \log^k(n))$ for some $k \geq 0$. Then, $T(n) = \Theta(n^{\log_b(a)} \log^{k+1}(n))$. So since $f(n) = \Theta(n^{\log_y(x)})$, it follows that:

$$T(n) = \Theta(n^{\log_y(x)} \log^{0+1}(n))$$
$$T(n) = \Theta(n^{\log_y(x)} \log(n))$$

This matches the problems request, thus completing the proof.

c. Given recurrence relation $T(n) = xT(n/y) + n^z$, we have the following:
$a = x$, number of subproblems at each recursion level
$b = y$, factor which is divided by problem size
$f(n) = n^z$, non-recursive work at each step
Then, case 3 of master theorem states if $f(n) = \Omega(n^{\log_b(a)+\epsilon})$ for some $\epsilon > 0$, and if conditions $af(n/b)$ for some $k < 1$ and sufficiently large n holds true, then $T(n) = \Theta(f(n))$.
Since $z > \log_y(x)$, n^z grows faster than $n^{\log_y(x)}$, so $f(n) = n^z$ satisfies case 3 conditions.
To satisfy conditions of case 3, we know that $x \cdot f(\frac{n}{y}) = x \cdot (\frac{n}{y})^z \leq k \cdot n^z$ for some $k < 1$ and sufficiently large n. Since $z > \log_y(x)$, we substitute z with $\log_y(x)$ and get:

$$x \cdot y^{-z} = x \cdot y^{-\log_y(x)-\epsilon} = x \cdot \frac{1}{x} \cdot y^{-\epsilon} = y^{-\epsilon} < 1.$$

So, the regularity condition is satisfied since $y^{-\epsilon} < 1$ for $\epsilon > 0$. Thus all regularity conditions are satisfied and we can conclude $T(n) = \Theta(f(n)) = \Theta(n^z)$ by master theorem case 3. for $z > \log_y(x)$, $T(n) = \Theta(n^z)$.