For each of the following recurrence equations which describe the running time $T(n)$ of a recursive algorithm, use the master method to express the asymptotic complexity (assuming that $T(n) = c$ for $n < d$, for constants $c > 0$ and $d \geq 1$).

Example 1. $T(n) = 9T(n/3) + n^2$
Then, $f(n) = n^2$ and $n^{\log_b(a)} = n^{\log_3(9)} = n^2$.
By case 2 (with $k = 0$) of the master theorem, $T(n) = \Theta(n^2 \log n)$.

Example 2. $T(n) = 4T(n/2) + n$
Then, $f(n) = n$ and $n^{\log_b(a)} = n^{\log_2(4)} = n^2$.
By case 1 of the master theorem, $T(n) = \Theta(n^2)$.

Example 3. $T(n) = 8T(n/8) + n^2 \log n$
Then, $f(n) = n^2 \log n$ and $n^{\log_b(a)} = n^{\log_8(8)} = n$.
By case 3 of the master theorem, $T(n) = \Theta(n^2 \log n)$.

Example 4. $T(n) = 4T(n/4) + n$
Then, $f(n) = n$ and $n^{\log_b(a)} = n^{\log_4(4)} = n$.
By case 2 of the master theorem (with $k = 0$), $T(n) = \Theta(n \log n)$.

Example 5. $T(n) = 4T(n/2) + n^2 \log^3 n$
Then, $f(n) = n^2 \log^3 n$ and $n^{\log_b(a)} = n^{\log_2(4)} = n^2$.
By case 2 (with $k = 3$) of the master theorem, $T(n) = \Theta(n^2 \log^4 n)$.

Example 6. $T(n) = 4T(n/2) + \log n$
Then, $f(n) = \log n$ and $n^{\log_b(a)} = n^{\log_2(4)} = n^2$.
By case 1 of the master theorem, $T(n) = \Theta(n^2)$.

Example 7. $T(n) = 4T(n/4) + n^2$
Then, $f(n) = n^2$ and $n^{\log_b(a)} = n^{\log_4(4)} = n$.
By case 3 of the master theorem, $T(n) = \Theta(n^2)$.

Example 8. $T(n) = 7T(n/3) + \log n$
Then, $f(n) = \log n$ and $n^{\log_b(a)} = n^{\log_3(7)} = n^{1.7712437}$.
By case 1 of the master theorem, $T(n) = \Theta(n^{\log_3(7)})$.