Problem 1

Algorithm A:

$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c)$

$8T\left(\frac{n}{4}\right) + O(n)$

$\mathrm{c} < \log_\mathrm{b}(\mathrm{a})$

$1 < \log_4(8)$

$T(n) = O(n^{\log_4(8)}) = O(n^{1.5})$

Algorithm B:

$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c)$

$T(n) = 2T(n-1) + O(1)$

$T(2) = 2 \cdot T(1)$

$\mathrm{T}(3) = 2 \cdot \mathrm{T}(2) = 2 \cdot 2 \cdot \mathrm{T}(1)$

$\mathrm{T}(4) = 2 \cdot \mathrm{T}(3) = 2 \cdot 2 \cdot 2 \cdot \mathrm{T}(1)$

$Hence, \qquad T(n) = O(2^n)$

Algorithm C:

$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c)$

$T(n) = 9T\left(\frac{n}{3}\right) + O(n^2)$

$\mathrm{c} = \log_\mathrm{b}(\mathrm{a})$

$2 = \log_3(9)$

$Hence, T(n) = O(n^2 \log(n))$

I would choose algorithm A because the power of n is lowest in the running time of algorithm A