1. cont ← 2n
For j ← 1 to n do
  s ← cont
  while s ≥ 1 do
    s ← s/2
  end while
end For
return s

$\left.\begin{array}{l} \text{For } j \leftarrow 1 \text{ to } n \text{ do} \\ \quad \left.\begin{array}{l}\text{while } s \geq 1 \text{ do} \\ s \leftarrow s/2 \\ \text{end while}\end{array}\right\} n \end{array}\right\} n$

$S_0 = 2^n$

$S_1 = \frac{S_0}{2} = \frac{2^n}{2^2}$

$S_2 = \frac{S_1}{2} = \frac{2^n}{2^3}$

$\therefore \frac{2^n}{2^k} \leq 1 \rightarrow 2^n \leq 2^k = n = k$

$O(f(n)) = O(n^2)$

$\therefore T(n) = 3n^2 + 2$

Escaneado con CamScanner

2.

Iteración 1: $T(n) = 2T(n/2) + n$

Iteración 2: $T(n/2) = 2T(n/4) + n/2$

Sustituyendo $T(n/2)$: $\Rightarrow T(n) = 2(2T(n/4) + n/2) + n = 4T(n/4) + 2n$

Iteración 3: $T(n/4) = 2T(n/8) + n/4$

Sustituyendo $T(n/4)$: $\Rightarrow T(n) = 4(2T(n/8) + n/4) + 2n = 5T(n/8) + n + 2n + 2n = 8T(n/8) + 4n$

$\therefore T(n) = 2^k T(n/2^k) + kn$

Valor de k: $n/2^k = 1 \rightarrow k = \log_2 n$

$T(n) = 2^{\log_2 n} T(1) + n\log_2 n$

Si $T(1) = 1 \rightarrow T(n) = n + n\log_2 n$

$\therefore$ Complejidad $O(n\log n)$