(1) Función algoritmo $S(n)$

```
cont ← 2^n
for j ← 1 to n do
    s ← cont
    while s >= 1 do
        s ← s/2
    end while
end for
returns s
```

n

n

while

$\boxed{S_0 = 2^n}$

$S_1 = \frac{S_0}{2} = \frac{2^n}{2^1}$

$S_2 = \frac{S_1}{2} = \frac{2^n}{2^{1+1}}$

$S_k = \frac{S_{k-1}}{2^k} = \frac{2^n}{2^k}$

$\frac{2^n}{2^k} = 1 \Rightarrow 2^n = 2^k \Rightarrow \boxed{n = k}$

$T(n) = n \cdot n = n^2$

$\therefore O(n^2)$

(2) $T(n) = 2t(n/2) + n \; ; \; t(1) = 1$

① $T(n) = 2t(n/2) + n$

② $T(n) = 2\,(2t(n/4) + n/2) + n \Rightarrow 4t(n/4) + 2n$

③ $t(n) = 4(2t(n/8) + n/4) + 2n = 8t(n/8) + 3$

$T(n) \Rightarrow \boxed{2^k t(n/2^k) + kn}$

$n/2^k = 1 \Rightarrow \boxed{n = 2^k} \Rightarrow \boxed{k = \log 2n}$

$k = \log 2(2^k) \Rightarrow \boxed{\log 2n = \log 2(2^k)}$

$\therefore\; t(n) = 2^{\log 2(n)} . T(1) + n\log_2 n$

$= n . 1 + n \log_2(n)$

$= n + n . \log_2(n)$

$\Rightarrow O(n) = O(n . \log(n))$