Taller 5.
Cost computacional dels algorismes

4. Relaciona a través de la mesura O(f(n)), les funcions: g(n)=n+k i f(n)=n, on k és una constant positiva.

Solución:
Si escogemos que c = k, entonces para $n_0 \geq 1$,
$g(n) = n + k \leq k * n = k * f(n)$
Por lo que, $g(n) \in O(f(n))$

8. Calcula pas a pas i de manera clara l'eficiència computacional del següent algorisme en termes de les mesures asimptòtiques O i Θ.

```
i:=n;
while i>1 do
        i:= i div 2
end while
```

13. Demostra que $f(n) \in O(g(n))$ si $g(n) \in \Omega(f(n))$.

14. Considera les següents funcions de n:
$f1(n) = n^2$
$f2(n) = n^2 + 1000n$

$$f_3(n) = \begin{cases} n, \text{ si n és imparell} \\ n^3, \text{ si n és parell} \end{cases}$$

Per a cada possible parell de valors de i i j, indica si fi(n) és O(fj(n)) o no, i si fi(n) és Ω(fj(n)) o no.

16. Siguin $T1(n)=5n^3$ i $T2(n)=100n^2$ els temps d'execució de dos programes. Creus que el programa de temps T2 és sempre més ràpid que el de temps T1 independentment del tamany de les entrades, n? Raona i justifica clara i formalment la resposta.

17. Calcula, mitjançant la notació O, el temps d'execució del següent algorisme:

```
procediment prova(n: integer);
var i, j, k: integer;
begin
        for i:=1 to n-1 do
                for j:=i+1 to n do
                        for k:=1 to j do
                        {alguna sentència que requereixi un temps O(1)}
        end;
```

Solución:

$\text{for}_3 = \sum_{k=1}^{j} 1 = j$

$\text{for}_2 = \sum_{j=i+1}^{n} j = \sum_{j=1}^{n} j - \sum_{j=1}^{i} j = ½ *n(n+1) - ½ *i(i+1)$

$\text{for}_1 = \sum_{i=1}^{n-1} ½\, n * (n + 1) - ½\, i * (i + 1) = \sum_{i=1}^{n-1} ½\, n * (n + 1) - \sum_{i=1}^{n-1} ½\, i * (i + 1) =$

$= ⅓\, n^3 - ⅓\, n \in O(n^3)$

$\sum_{i=1}^{n-1} ½\, i * (i + 1) = \sum_{i=1}^{n-1} ½(i^2+i) = ½ \sum_{i=1}^{n-1} i^2 + ½ \sum_{i=1}^{n-1} i = ½ \left(\sum_{i=1}^{n-1} i^2 + \sum_{i=1}^{n-1} i \right)$

//Fórmulas

$\sum_{j=1}^{n} 1 = n$

$\sum_{j=1}^{n} j = \frac{n*(n+1)}{2}$

$\sum_{j=1}^{n} j^2 = ½\, n*(n+1)(2n+1)$

18. Calcula, mitjançant la notació O, el temps d'execució del següent algorisme:

```
procediment prod_mat(n: integer);
var i, j, k: integer;
begin
	for i:=1 to n do
		for j:=1 to n do begin
			C[i,j]:=0;
			for k:=1 to n do
				C[i,j]:= C[i,j] + A[i,k]*B[k,j]
		end
	end;
```

Solución:

$\text{for}_3 = O(1) * O(n) = O(n*1) = O(n)$

$\text{for}_2 = O(n) * (O(1)+O(n)) = O(n) * (\max(1,n)) = O(n) * O(n) = O(n*n) = O(n^2)$

$\text{for}_1 = O(n) * O(n^2) = O(n*n^2) = O(n^3)$