$T(n) = n^{1/2} + 10^{20}$

Tengo que llegar a:
$n^{1/2} + 10^{20} <= cn^{1/2}$ para todo $n>=n_0$

$n^{1/2}<=c_1n^{1/2}$ $\quad$ $10^{20}<=c_2n^{1/2}$
$n^{1/2}<=n^{1/2}$ $\quad$ $1<=n^{1/2}$
$1*n^{1/2}<=1*n^{1/2}$ $\quad$ $10^{20}*1<=10^{20}*n^{1/2}$
$c_1=1 \quad n_0=0$ $\quad$ $c_2=10^{20} \quad n_0=1$

$1*n^{1/2}+10^{20}*1<=1*n^{1/2}+10^{20}*n^{1/2}$
$n^{1/2}+10^{20}<=(1+10^{20})n^{1/2}$

$c=c_1+c_2=1+10^{20}$
$n_0=1$

Claramente, se cumple que T(n) es $O(n^{1/2})$, porque existen las constantes $c=1+10^{20}$ y $n_0=1$, de modo que se cumple que $T(n)<=cn^{1/2}$ para todo n>=1.

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

Hay que llegar a:
$n+\log_2 n<=cn$ para todo $n>=n_0$

$n<=c_1n$ $\quad$ $\log_2 n<=c_2n$
$1*n<=1*n$ $\quad$ $1*\log_2 n<=1*n$
$c_1=1 \quad n_0=0$ $\quad$ $c_2=1 \quad n_0=1$

$n+\log_2 n<=1*n+1*n$
$n+\log_2 n<=(1+1)*n$
$n+\log_2 n<=2*n$

$c=c_1+c_2=1+1=2$
$n_0=1$

Claramente, se cumple que T(n) es O(n), porque existen las constantes c=2 y $n_0=1$, de modo que se cumple que T(n)<=cn para todo n>=1.

$T(n)=6+n*\log_2 n+\log_2 n-1$ $\quad$ es $O(n*\log_2 n)$??

$6<=c_1*(n*\log_2 n)$ $\quad$ $n*\log_2 n<=c_2*(n*\log_2 n)$ $\quad$ $\log_2 n<=c_3*(n*\log_2 n)$ $\quad$ $-1<=c_4*(n*\log_2 n)$
$c_1=6 \quad n_0=2$ $\quad$ $c_2=1 \quad n_0=1$ $\quad$ $c_3=1 \quad n_0=1$ $\quad$ $c_4=0 \quad n_0=1$

$6+n*\log_2 n+\log_2 n-1<=6*(n*\log_2 n)+1*(n*\log_2 n)+1*(n*\log_2 n)+0*(n*\log_2 n)$
$6+n*\log_2 n+\log_2 n-1<=(6+1+1+0)(n*\log_2 n)$

$6+n*\log_2 n+\log_2 n-1<=8(n*\log_2 n)$
c=8
$n_0=2$

Claramente, se cumple que T(n) es O($n*\log_2 n$), porque existen las constantes c=8 y $n_0=2$, de modo que se cumple que T(n)<=c*($n*\log_2 n$) para todo n>=2.

3^n es O(2^n)??

$3^n<=c2^n$

$3^n/2^n<=c$
$(3/2)^n<=c$

Como $(3/2)^n$ es una funcion creciente, no se puede hallar un valor para c constante ni tampoco para un n_0, de modo que se cumpla de definicion de big-oh, entonces 3^n no es de orden 2^n.

$2^{n+1} <= c2^n$
$2*2^n <=c2^n$
c=2 $n_0=1$

$2^{2n} <=c2^n$
$2^{n+n}<=c2^n$
$2^n*2^n<=c2^n$
$(2^n*2^n)/2^n<=c$
$2^n<=c$

Ejercicio 7.3)

`int x=1;`	c_1
`for (int i = 1; i < n; i = i+4)`	sum de i=1 hasta (n-1)/4
`    for (int j = 1; j < n; j = j+=|n/4|)`	sum de j=1 hasta 4
`        for (int k = 1; k < n; k = k*2)`	sum de k=1 hasta $\log_2 n$
`            x = x+1;`	c_2

Para i:
paso 0: i=1=1+0*4
paso 1: i=1+4=1+1*4
paso 2: i=1+4+4=1+2*4
paso 3: i=1+4+4+4=1+3*4
paso k: i=1+4*k
1+4k=n
4k=n-1
k=(n-1)/4

Para j:
paso 0: j=1
paso 1: j=1+n/4
paso 2: j=1+2*n/4
paso 3: j=1+3*n/4
paso 4: j=1+4*n/4=1+n

Para k:
paso 0: $k=1=2^0$
paso 1: $k=1*2=2^1$
paso 2: $k=1*2*2=2^2$
paso 3: $k=1*2*2*2=2^3$
paso p: $k=2^p$
$2^p=n$
$\log_2 2^p=\log_2 n$
$p=\log_2 n$

$$T(n)=c_1+\sum_{i=1}^{(n-1)/4}\sum_{j=1}^{4}\sum_{k=1}^{log(n)} c_2$$

$$T(n)=c_1+c_2\sum_{i=1}^{(n-1)/4}\sum_{j=1}^{4}\sum_{k=1}^{log(n)} 1$$

$$T(n)=c_1+c_2\sum_{i=1}^{(n-1)/4}\sum_{j=1}^{4}\log_2 n$$

$$T(n)=c_1+c_2*\log_2 n\sum_{i=1}^{(n-1)/4}\sum_{j=1}^{4} 1$$

$$T(n)=c_1+c_2*\log_2 n\sum_{i=1}^{(n-1)/4} 4$$

$$T(n)=c_1+4*c_2*\log_2 n\sum_{i=1}^{(n-1)/4} 1$$

$T(n)=c_1+4*c_2*\log_2 n*(n-1)/4$
$T(n)=c_1+(4*c_2*\log_2 n)*(n/4-1/4)$
$T(n)=c_1+n/4*(4*c_2*\log_2 n) - ¼*(4*c_2*\log_2 n)$
$T(n)=c_1+c_2*n*\log_2 n - c_2*\log_2 n$

$c_1<=k_1(n*\log_2 n)$
$1*c_1<=c_1*(n*\log_2 n)$
$k_1=c_1 \quad n_0=2$

$c_2*(n*\log_2 n)<=k_2*(n*\log_2 n)$
$c_2*(n*\log_2 n)<=c_2*(n*\log_2 n)$
$k_2=c_2 \quad n_0=1$

$c_1+c_2*(n*\log_2 n)<=c_1*(n*\log_2 n)+c_2*(n*\log_2 n)$
$c_1+c_2*(n*\log_2 n)<=(c_1+c_2)*(n*\log_2 n)$
$k=k_1+k_2=c_1+c_2 \quad n_0=2$

Ejercicio 7.4)

$\sum_{i=1}^{n}$ (a+b)= $\sum_{i=1}^{n}$ a+ $\sum_{i=1}^{n}$ b

j = 1;	c_1
while (j <= n) {	sum desde j=1 hasta $\log_2(n+1)$
for (i = n*n; i >=1; i = i-3)	sum desde i=1 hasta $n^2/3$
x=x+1;	c_2
j = j*2;	c_3
}	

paso 0: $j=1=2^0$

paso 1: $j=1*2=2^1$

paso 2: $j=1*2*2=2^2$

paso 3: $j=1*2*2*2=2^3$

paso k: $j=2^k$

$n+1=2^k$

$\log_2(n+1)=k$

paso 0: $i=n^2=n^2-3*0$

paso 1: $i=n^2-3=n^2-3*1$

paso 2: $i=n^2-3-3=n^2-3*2$

paso 3: $i=n^2-3-3-3=n^2-3*3$

paso k: $i=n^2-3*k$

$n^2-3*k=0$

$-3k=-n^2$

$k=n^2/3$

```
static public int rec3(int n){
        if ( n == 0 )
                return 0;
        else {
                if ( n == 1 )
                        return 1;
                else
                        return (rec3(n-2) * rec3(n-2));
        }
}
```

$T(n)=c_1$ $\quad$ $n<=1$

$\quad c_2+2T(n-2)$ $\quad$ $n>1$

Paso 1: $T(n)=c_2+2T(n-2)$ $\quad$ $n>1$

Paso 2: $T(n)=c_2+2[c_2+2T((n-2)-2)]$ $\quad$ $n-2>1$

$T(n)=c_2+2[c_2+2T(n-4)]$

$T(n)=c_2+2c_2+2^2T(n-4)$

Paso 3: $T(n)=3c_2+2^2[c_2+2T((n-4)-2)]$ $\quad$ $n-4>1$

$T(n)= c_2+2c_2+2^2c_2+2^3T(n-6)$

Paso i: $T(n)= \sum_{k=0}^{i-1} 2^k c_2+2^iT(n-2i)$

$T(n)=c_2* \sum_{k=0}^{i-1} 2^k +2^iT(n-2i)$

$T(n)=c_2*(2^i-1)+2^iT(n-2i)$

$T(n)=c_2*2^i-c_2+2^iT(n-2i)$

n-2i=1 CASO BASE

-2i=1-n

i=(-n+1)/(-2)

i=(n-1)/2

$T(n)=c_2*2^{(n-1)/2}-c_2+2^{(n-1)/2}T(n-2((n-1)/2))$

$T(n)=c_2*2^{n/2-½}-c_2+2^{n/2-1/2}T(n-2(n/2-1/2))$

$T(n)=c_2*(2^{n/2}/2^{1/2})-c_2+(2^{n/2}/2^{1/2})T(n-n+1)$

$T(n)=c_2*(2^{n/2}/2^{1/2})-c_2+(2^{n/2}/2^{1/2})T(1)$

$T(n)=c_2*(2^{n/2}/2^{1/2})-c_2+c_1*(2^{n/2}/2^{1/2})$

$c_2*(2^{n/2}/2^{1/2})<=k_1 2^{n/2}$ $\quad$ $c_1*(2^{n/2}/2^{1/2})<=k_2*2^{n/2}$

$(c_2/2^{1/2})*2^{n/2}<=(c_2/2^{1/2})*2^{n/2}$ $\quad$ $(c_1/2^{1/2})*2^{n/2}<=(c_1/2^{1/2})*2^{n/2}$

$k_1=c_2/2^{1/2}$ $\quad n_0=0$ $\quad$ $k_2=(c_1/2^{1/2})$ $\quad n_0=0$

$c_2*(2^{n/2}/2^{1/2}) + c_1*(2^{n/2}/2^{1/2}) <= (c_2/2^{1/2})*2^{n/2} + (c_1/2^{1/2})*2^{n/2}$

$c_2*(2^{n/2}/2^{1/2}) + c_1*(2^{n/2}/2^{1/2}) <= (c_2/2^{1/2}+c_1/2^{1/2})*2^{n/2}$

$k=c_2/2^{1/2}+c_1/2^{1/2}=k_1+k_2$ $\quad$ $n_0=0$

```
preorden(){
	//procesar raiz
	LG<AG<T>> hijos = this.getHijos();
	hijos.comenzar();
	while(!hijos.fin())
		hijos.proximo().preorden();
}
```

Grafo<T> grafo;
Diferencias:
1- Puede haber ciclos.
2- Puede haber nodos inalcanzables desde algún otro.

```
dfs(){
	boolean[] marca = new boolean[grafo.listaDeVertices().tamanio()+1];
	for(i=1;i<=grafo.listaDeVertices().tamanio();i++){
		if(!marca[i])	dfs(i,marca);
	}
}

dfs(int i, boolean[] marca){
	marca[i]=true;
	Vertice<T> v = grafo.listaDeVertices().elemento(i);
	//procesar a v
	LG<Arista<T>> ady = grafo.listaDeAdyacentes(v);
	ady.comenzar();
	while(!ady.fin()){
		Arista<T> ar = ady.proximo();
		Vertice<T> destino = ar.verticeDestino();
		int pos = destino.posicion();
		if(!marca[pos])
			dfs(pos,marca);
	}
}
```

Ejercicio 6:

$$cte_1 + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sum_{k=1}^{j} cte_2$$

$$cte_1 + cte_2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sum_{k=1}^{j} 1$$

$cte_1 + cte_2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} j$

$\sum_{j=i+1}^{n} j = (\sum_{j=1}^{n} j - \sum_{j=1}^{i} j)$

$cte_1 + cte_2 \sum_{i=1}^{n-1} (½(n^2+n - i^2 - i))$

$cte_1 + cte_2 * 1/2 * \sum_{i=1}^{n-1} (n^2+n - i^2 - i)$

$n^2 \sum_{i=1}^{n-1} 1 + n \sum_{i=1}^{n-1} 1 - \sum_{i=1}^{n-1} i^2 - \sum_{i=1}^{n-1} i$

$n^2 * n-1 + n * n-1 -$

12-3)

$T(n) = 1$ $n=1$
$4T(n/2) + n^2$, $n>=2$

paso 1 = $n^2 + 4T(n/2)$ $n>= 2$

paso 2 = $n^2 + 4[n^2+4T((n/2)/2)$ $n/2>=2$

$n^2+4n^2+4^2 T(n/4)$

paso 3 = $n^2+4n^2+4^2[n^2+4T((n/4)/2)]$ $n/4>= 2$

$n^2+4n^2+4^2*n^2+4^3*T(n/8)$

paso i = $\sum_{k=0}^{i-1} 4^k * n^2 + 4^i * T(n/2^i)$

$n^2 * (4^{i-1+1} - 1)/ (4 - 1) + 4^i * T(n/2^i)$

$n^2 * [(4^i - 1)/ 3] + 4^i * T(n/2^i)$

caso base: $n/2^i = 1$

$n=2^i$

$\log_2 n = \log_2 2^i$

$\log_2 n = i$

$n^2 * [(4^{\log n} - 1)/ 3] + 4^{\log n} * T(n/2^{\log n})$ Y AHORA ?!! xD