UNIVERSIDAD DE MÁLAGA

Análisis y Diseño de Algoritmos

Práctica 1

1.- Implementar un programa en Java que utilice la clase Temporizador proporcionada para tomar el tiempo empleado por los siguientes algoritmos en ejecutarse para diferentes tamaños de entrada.

a) Representar gráficamente en R los tiempos de ejecución de los siguientes algoritmos para distintos tamaños de entrada y decidir cuál es más complejo.
b) Visualizar para distintos tamaños de entrada (n) el cociente $T_1(n)/T_2(n)$, donde $T_1(n)$ es el tiempo de ejecución del algoritmo 1 y $T_2(n)$ el del algoritmo 2. En función de esta nueva representación, decidir cuál es más complejo.
c) Intentar descubrir el orden de crecimiento de los algoritmos comparándolos con las funciones de coste teóricas *log n* , *n*, n^2, 2^n y *n!*.

```
int algoritmo1(int n){
    //n >0
    if (n<=2) return 1;
    else return algoritmo1(n-1) + algoritmo1(n-2);
}
```

```
void algoritmo2(int [] a) {
     for(int i = 0; i < a.length; i ++) {
          int suma = 0;
          for (int j = 0; j < a.length; j++ ) {
               suma += a[j];
          }
          a[i] = suma;
     }
}
```

2.- Visualizar gráficamente las siguientes funciones y ordenarlas en orden creciente de complejidad:

a) $4 \log ((n+100)^{10})$
b) $10^{-6}n^5+9n^3$
c) $(\log n)^2$
d) 17
e) $(1/3)^n$
f) $(3/2)^n$
g) $\sqrt[3]{n}$
h) $\log(\log n)$
i) $n / \log n$
j) $\sqrt{n} \cdot \log(n)$
k) 5^n
l) $n+10^{10}$
m) $(n-2)!$