```
GRAU_MAXIMO(G)
1   max = 0
2   para cada x ∈ V(G)
3          aux = GRAU(G,v)
4          se max < aux
5               aux = max
6   retorna max
```

Cada linha, sozinha, leva tempo constante para ser executada, exceto pela linha 3.

As linhas 1 e 6 são executadas uma única vez cada.

As linhas 2, 3 e 4 executam uma vez para cada vértice do grafo.

A linha 5 executa no máximo uma vez para cada vértice do grafo.

Vamos denotar o conjunto de vértices por $V(G) = \{v_1, v_2, \dots, v_n\}$ (temos então $n = v(G)$).

note que o laço poderia ser escrito:

```
para i = 1 até n
     aux = GRAU(G,v_i)
```

O tempo total é dado pela soma do tempo gasto em cada linha.

```
GRAU(G,v)
   cont = 0
   para cada x ∈ N(v)
        cont++
   retorna cont
```

Tempo:

Em lista = $\Theta(d_G(v))$

Em matriz = $\Theta(v(G))$

Obs: como $d_G(v) \leq v(G)$ para qualquer $v \in V(G)$, podemos também dizer $O(v(G))$ para listas. mas se o grau do vértice é 1 e temos 1000 vértices no grafo, dizer $O(v(G))$ está bem longe da realidade.

Assim, em matriz de adjacências o tempo é

$$\underbrace{\Theta(1)}_{\text{linha 1}} + \underbrace{\Theta(1)+\cdots+\Theta(1)}_{\substack{\text{linha 2} \\ \text{(n vezes, uma} \\ \text{p/ cada vértice)}}} + \underbrace{\Theta(n)+\cdots+\Theta(n)}_{\substack{\text{linha 3} \\ \text{(n vezes, uma} \\ \text{p/ cada vértice)}}} + \underbrace{\Theta(1)+\cdots+\Theta(1)}_{\substack{\text{linha 4} \\ \text{(n vezes, uma} \\ \text{p/ cada vértice)}}} + \underbrace{X}_{\text{linha 5}} + \underbrace{\Theta(1)}_{\text{linha 6}}$$

$$= \Theta(1) + \Theta(n) + \Theta(n^2) + \Theta(n) + X + \Theta(1)$$

$$= \Theta(n^2) + X$$

Como a linha 5 executa no máximo uma vez por vértice do grafo e leva tempo constante por execução, X é $O(n)$.

Logo, $\Theta(n^2)$ domina a expressão acima.

E em listas de adjacências o tempo é

$$\underbrace{\Theta(1)}_{\text{linha 1}} + \underbrace{\Theta(1) + \cdots + \Theta(1)}_{\substack{\text{linha 2} \\ \text{(n vezes, uma} \\ \text{p/ cada vértice)}}} + \underbrace{\Theta(d_G(v_1)) + \Theta(d_G(v_2)) + \cdots + \Theta(d_G(v_n))}_{\substack{\text{linha 3} \\ \text{(n vezes, uma p/ cada} \\ \text{vértice)}}} + \underbrace{\Theta(1) + \cdots + \Theta(1)}_{\substack{\text{linha 4} \\ \text{(n vezes, uma p/} \\ \text{cada vértice)}}} + \underbrace{X}_{\text{linha 5}} + \underbrace{\Theta(1)}_{\text{linha 6}}$$

$$= \Theta(1) + \Theta(n) + \Theta(e(G)) + \Theta(n) + X + \Theta(1)$$

$$= \Theta(n + e(G)) + X$$

Como X é $O(n)$, $\Theta(n + e(G))$ domina a expressão acima.

Como cada $d_G(v_i) \leq n$, poderíamos substituir $\Theta(d_G(v_i))$ acima por $O(n)$:

$$\underbrace{\Theta(1)}_{\text{linha 1}} + \underbrace{\Theta(1) + \cdots + \Theta(1)}_{\substack{\text{linha 2} \\ \text{(n vezes, uma} \\ \text{p/ cada vértice)}}} + \underbrace{O(n) + \cdots + O(n)}_{\substack{\text{linha 3} \\ \text{(n vezes, uma p/} \\ \text{cada vértice)}}} + \underbrace{\Theta(1) + \cdots + \Theta(1)}_{\substack{\text{linha 4} \\ \text{(n vezes, uma p/} \\ \text{cada vértice)}}} + \underbrace{X}_{\text{linha 5}} + \underbrace{\Theta(1)}_{\text{linha 6}}$$

$$= \Theta(1) + \Theta(n) + O(n^2) + \Theta(n) + X + \Theta(1)$$

$$= O(n^2) + X$$

e, novamente, $O(n^2)$ domina.

Mas se você tem um grafo com 10000 vértices e 10000 arestas, a análise que diz que o tempo é $\Theta(n + e(G))$ é bem melhor do que a que diz $O(n^2)$ (e veja que ambas valem!).