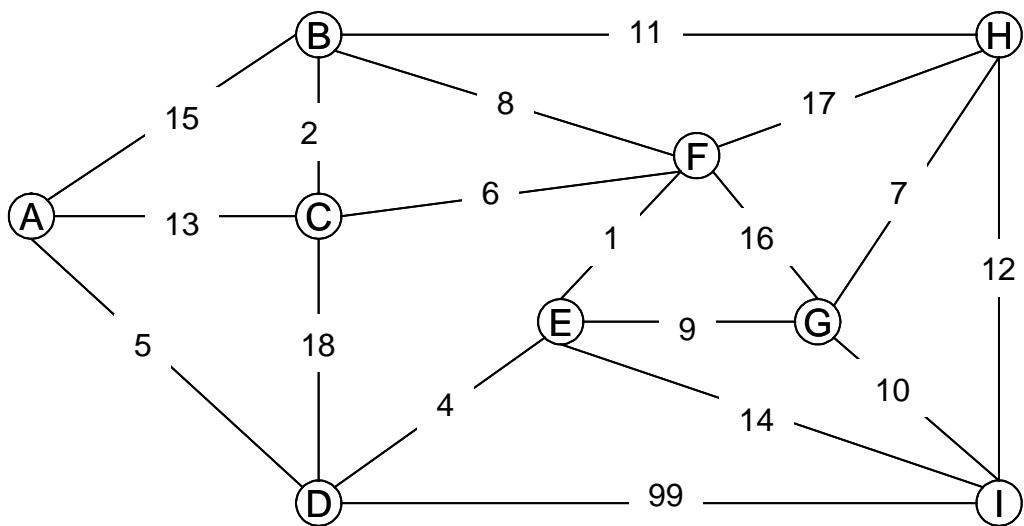


IDENTIFICAÇÃO

Nome: _____ ID: _____ 1/7/2010

Considere o grafo abaixo para responder as questões 1, 2, e 3.



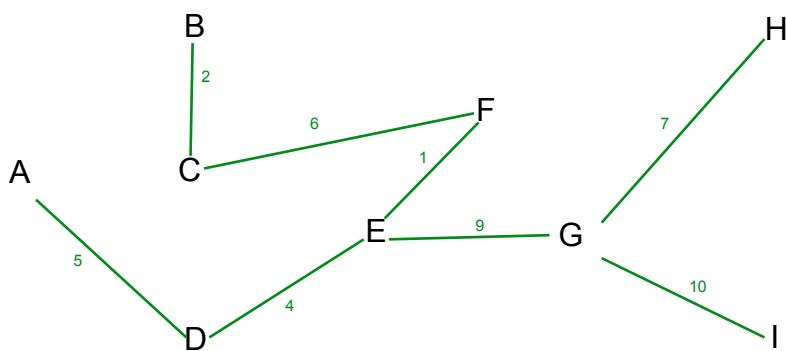
- 1) (valor 1.0) Descreva a seqüência de vértices visitada usando o caminhamento em profundidade e em amplitude, iniciando no vértice A. Considere que o grafo está representado por uma matriz de adjacência.

Profundidade: A, B, C, D, E, F, G, H, I

Amplitude: A, B, C, D, F, H, E, I, G

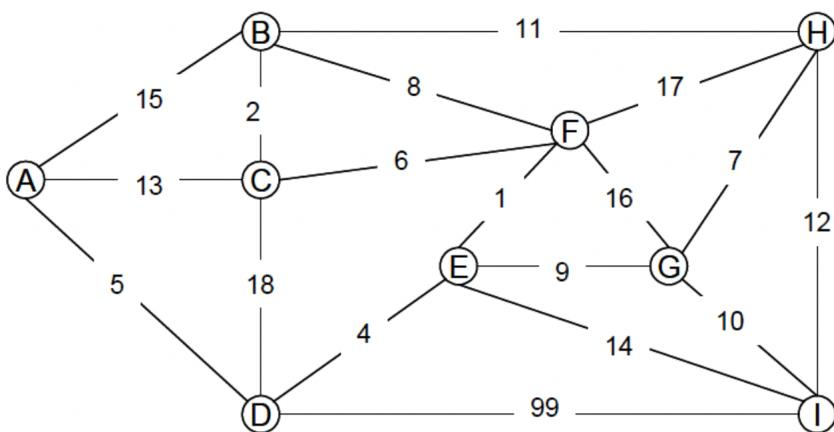
- 2) (valor 1.5) Mostre a árvore geradora mínima para o grafo. Qual o custo desta árvore?

Custo = 44



3) (valor 2.5) Mostre a execução **passo a passo** do algoritmo de Dijkstra (caminho mínimo) para o grafo, tomando como ponto de partida o vértice A. Sua solução deve mostrar:

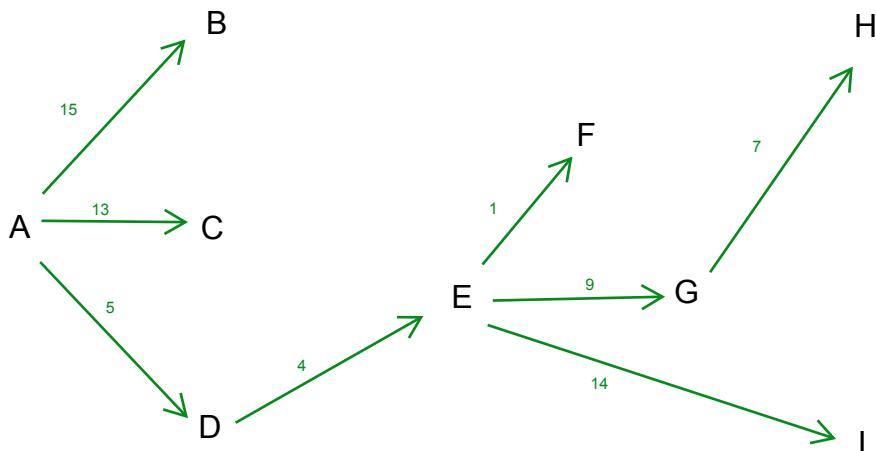
- A ordem de inclusão dos nodos na nuvem
- As distâncias mínimas a cada passo
- A árvore resultante



Nuvem	D[A]	D[B]	D[C]	D[D]	D[E]	D[F]	D[G]	D[H]	D[I]
\emptyset	0	∞							
A	0	15	13	5	∞	∞	∞	∞	∞
A,D	15	13	5	9	∞	∞	∞	∞	∞
A,D,E	15	13	5	9	10	18	∞	23	
A,D,E,F	15	13	5	9	10	18	27	23	
A,D,E,F,C	15	13	5	9	10	18	27	23	
A,D,E,F,C,B	15	13	5	9	10	18	26	23	
A,D,E,F,C,B,G	15	13	5	9	10	18	25	23	
A,D,E,F,C,B,G,I	15	13	5	9	10	18	25	23	
A,D,E,F,C,B,G,I,H	15	13	5	9	10	18	25	23	25

Boxed cell: 25

Predecessores									
A	B	C	D	E	F	G	H	I	
--	A	A	A						
--	A	A	A	D					
--	A	A	A	D	E	E			E
--	A	A	A	D	E	E	F		E
--	A	A	A	D	E	E	F	E	
--	A	A	A	D	E	E	B	E	
--	A	A	A	D	E	E	G	E	
--	A	A	A	D	E	E	G	E	



4) (valor 2.0) Crie uma função (**com COMENTÁRIOS**) que receba um grafo orientado representado por sua matriz de adjacência M e um vértice v. O resultado da função deve ser de acordo com a lista abaixo:

- 1 se o vértice for fonte
 - 1 se o vértice for sumidouro
 - 0 se o vértice não for nem fonte nem sumidouro

5) (valor 3.0) Reescreva o procedimento abaixo (percurso em profundidade) para que ele vá colorindo os vértices do grafo à medida que os percorre. Caso necessário, use funções auxiliares. O código deve ter comentários.

```

void DFS (int grafo[max+1][max+1], int v, int vis[max+1])
{
    int w;
    printf ("%d\n", v);
    vis[v] = 1;
    for(w = 1; w<=max ; w++)
        if ((grafo[v][w]== 1) && (vis[w]==1))
            DFS(grafo, w, vis);
}

```