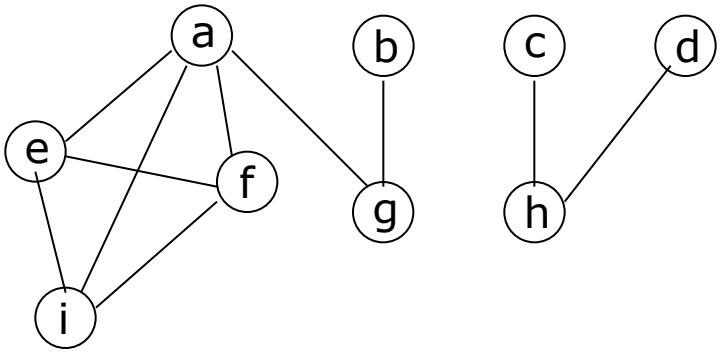
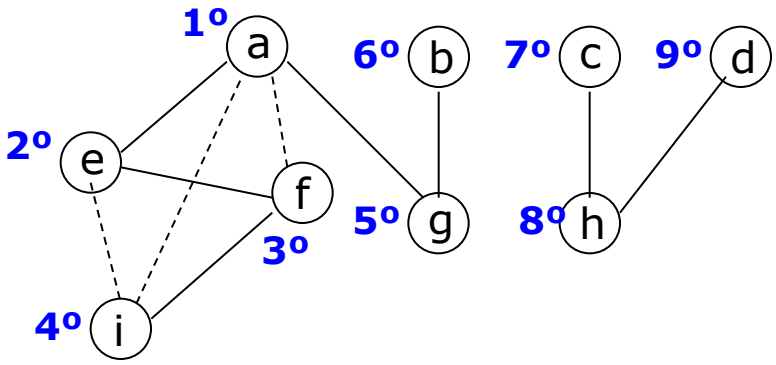
DFS - Profundidade





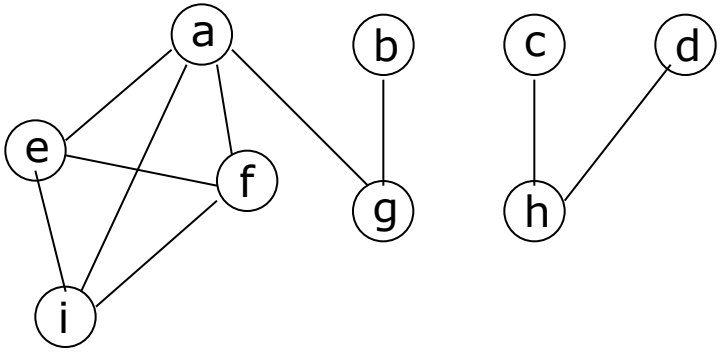
- Escolhe-se um vértice inicial;
- visita-se um primeiro vértice adjacente, marcando-o como visitado;
- coloca-se o vértice adjacente visitado numa pilha;
- o vértice visitado torna-se o novo vértice inicial;
- repete-se o processo até que o vértice procurado seja encontrado ou não haja mais vértices adjacentes. Se verdadeiro, desempilha-se o topo e procura-se o próximo adjacente, repetindo o algoritmo;
- o processo termina quando o vértice procurado for encontrado ou quando a pilha estiver vazia e todos os vértices tiverem sido visitados

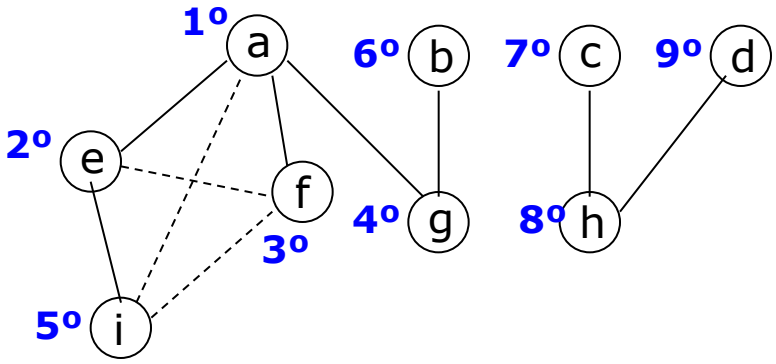
DFS - Profundidade

Pseudo-código:

```
1 procedure DFS (G,v):
2         create a stack S
3         S.push(v)
4         while S is not empty
5                 v ← S.pop()
6                 if v is not labeled as discovered:
7                         label v as discovered
8                         for all edges e in G.adjacentEdges(v) do
9                                 S.push(e)
10                        end loop
11                end if
12        end loop
13 end DFS
```

BFS – Amplitude (Largura)





- Escolhe-se um vértice para o início do caminhamento; visitam-se os vértices adjacentes, marcando-os como visitados;
- coloca-se cada um dos vértices adjacentes numa fila;
- após visitados os vértices adjacentes, o primeiro da fila torna-se o novo vértice inicial e reinicia o processo;
- termina quanto todos os vértices tiverem sido visitados ou o vértice procurado for encontrado.

BFS – Amplitude (Largura)

Pseudo-código:

```
1 procedure BFS(G,v) is
2         create a queue Q
3         Q.enqueue(v)
4         while Q is not empty loop
5                 t ← Q.dequeue()
6                 if v is not labeled as discovered:
7                         label v as discovered
8                         for all edges e in G.adjacentEdges(t) do
9                                 Q.enqueue(e)
10                        end loop
11                end if
12        end loop
13 end BFS
```