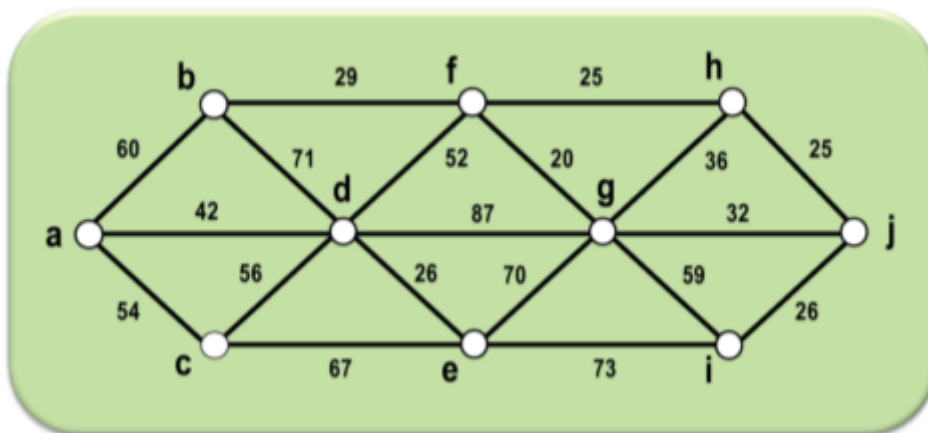


Grafos

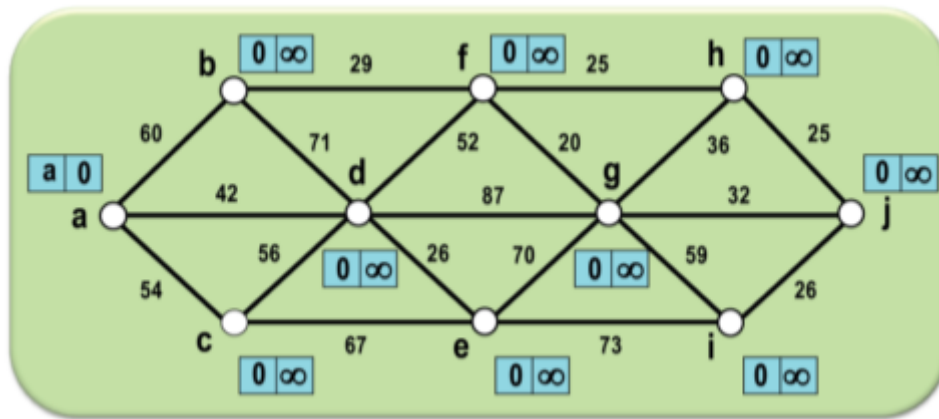
- 1) O algoritmo de Dijkstra nos permite encontrar o caminho mais curto entre quaisquer dois vértices de um gráfico, devemos estimar a distância de cada vértice do vértice inicial. Então visitamos cada nó e seus vizinhos para encontrar o caminho mais curto para esses vizinhos.

Processo geral:

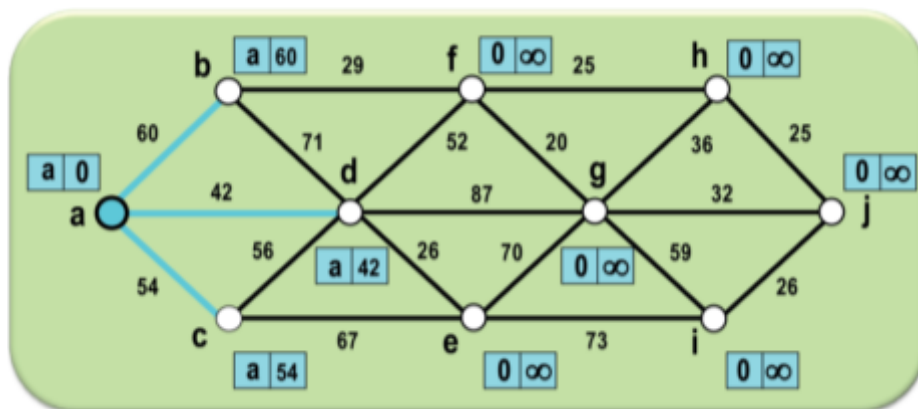
1. Escolher um vértice inicial e atribuir valores infinitos a todos os outros vértices;
2. Ir em cada vértice e sempre atualize seu comprimento do caminho caso ele seja menor que a adjacência do vértice seguinte;
3. Após cada iteração, escolhemos o vértice não visitado com o menor comprimento do caminho;
4. Repetir até que todos os vértices tenham sido visitados.



Grafo inicial com as distâncias em cada vértices vazias



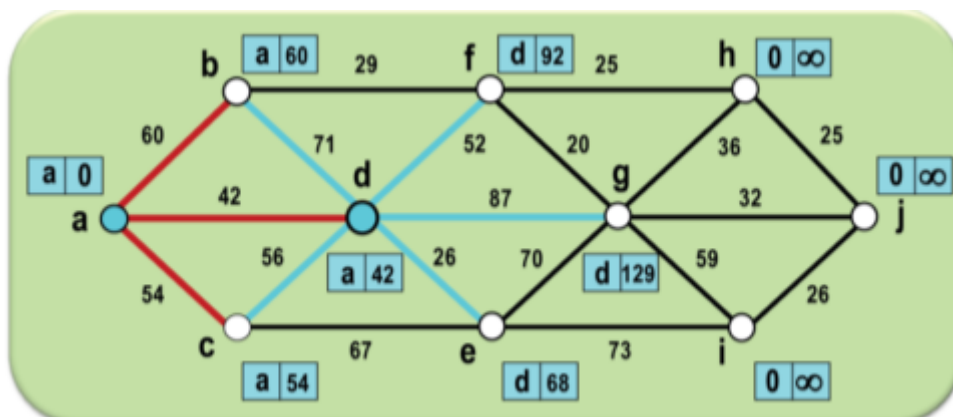
Preenchimento do grafo colocando todos os vértices (exceto o primeiro) com distância infinita. Nesse momento, todos os vértices estão abertos.



É escolhido o vértice cuja estimativa de distância seja menor que o infinito. Nesse caso, o vértice "a" é escolhido. o vértice a é fechado.

É verificado em seguida se tem vértice aberto na adjacência do vértice "a": Nesse caso, o vértice "b" é verificado (por estar na ordem de adição). Nesse momento a distância da adjacência "a" = 0 + a distância da aresta = 60 < que a adjacência de b que é infinita. Então a mesma é atualizada, passando a ter 60.

Como os vértices "d" e "c" ainda estão abertos, o mesmo processo ocorre até todas as adjacências do vértice "a" estiverem fechadas, resultando no exemplo abaixo:



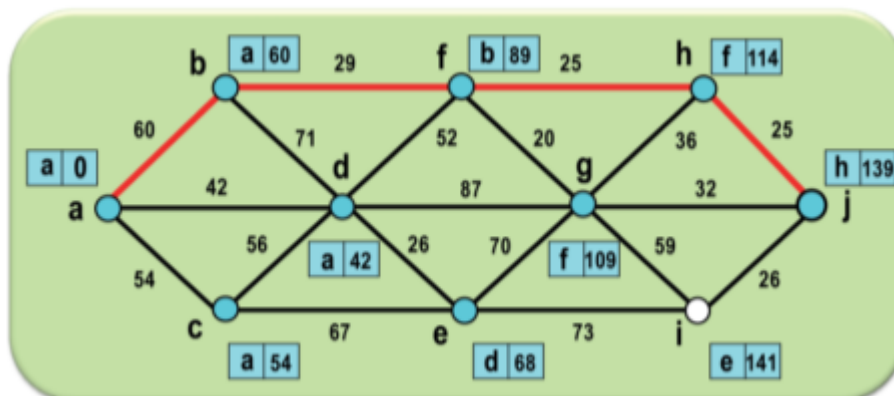
Continuando com o próximo vértice temos: É escolhido o vértice cuja estimativa de distância seja menor que o infinito: Nesse caso, o vértice “d” é escolhido. o vértice “d” é fechado.

Verificado em seguida se tem vértice aberto relacionado ao vértice “d”: nesse caso, o vértice “e” é verificado (por estar na ordem de adição).

Nesse momento a distância da adjacência “d” = 42 + a distância da aresta = 26 < que a adjacência de “e” que é infinita no momento. Então a mesma é atualizada, passando a ter 68.

Como os vértices “f” e “g” ainda estão abertos, o mesmo processo ocorre até todas as adjacências do vértice “d” estiverem fechadas:

Todo processo descrito acima é repetido até não sobrar nenhum vértice que esteja aberto.



Como todas as distâncias foram inseridas, é possível determinar qual a menor distância entre 2 vértices

Processo

Enquanto houver vértice aberto:

 escolher um vértice aberto cuja a estimativa de distância seja menor entre os demais vértices abertos.

 fechar esse vértice.

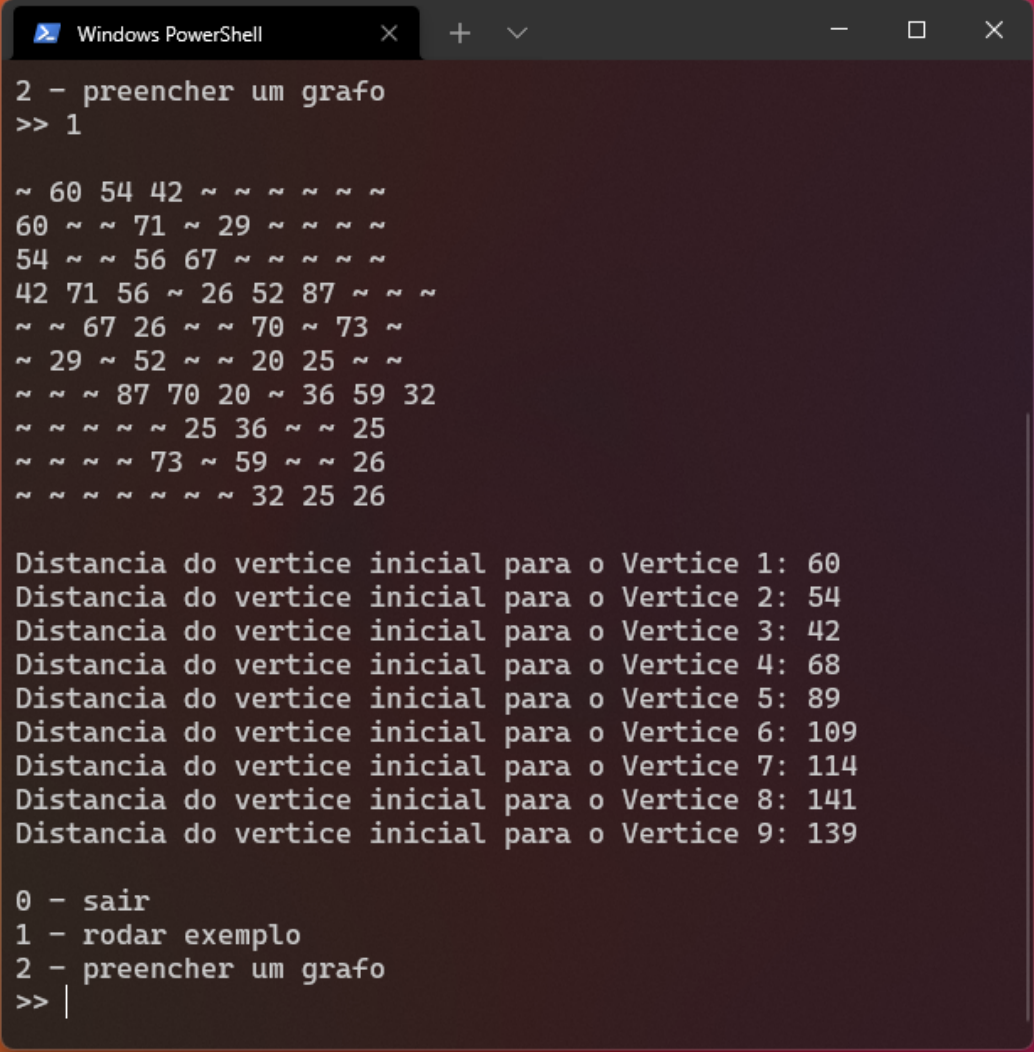
 para todo vértice v aberto na adjacência de u:

 somar a distância do vértice u ao peso da aresta (u, v).

 Caso a soma seja menor que a estimativa de distância de v:

 atualizar a distância do vértice adjacente.

Teste de execução



```
Windows PowerShell
2 - preencher um grafo
>> 1

~ 60 54 42 ~ ~ ~ ~ ~
60 ~ ~ 71 ~ 29 ~ ~ ~ ~
54 ~ ~ 56 67 ~ ~ ~ ~ ~
42 71 56 ~ 26 52 87 ~ ~ ~
~ ~ 67 26 ~ ~ 70 ~ 73 ~
~ 29 ~ 52 ~ ~ 20 25 ~ ~
~ ~ ~ 87 70 20 ~ 36 59 32
~ ~ ~ ~ ~ 25 36 ~ ~ 25
~ ~ ~ ~ 73 ~ 59 ~ ~ 26
~ ~ ~ ~ ~ ~ ~ 32 25 26

Distancia do vertice inicial para o Vertice 1: 60
Distancia do vertice inicial para o Vertice 2: 54
Distancia do vertice inicial para o Vertice 3: 42
Distancia do vertice inicial para o Vertice 4: 68
Distancia do vertice inicial para o Vertice 5: 89
Distancia do vertice inicial para o Vertice 6: 109
Distancia do vertice inicial para o Vertice 7: 114
Distancia do vertice inicial para o Vertice 8: 141
Distancia do vertice inicial para o Vertice 9: 139

0 - sair
1 - rodar exemplo
2 - preencher um grafo
>> |
```

Exemplo de execução do gráfico de exemplo acima (demonstração apenas com a distância das adjacências)