Trabalho Final

iGo - Caminhos Inteligentes
Estruturas de Dados II

Antonio Emilio Pereira
Melyssa Mariana Gomes Silva

O que é?

O iGo é uma aplicação de geolocalização desenvolvido na linguagem C com o intuito de fornecer o menor caminho entre dois locais.

Como funciona?

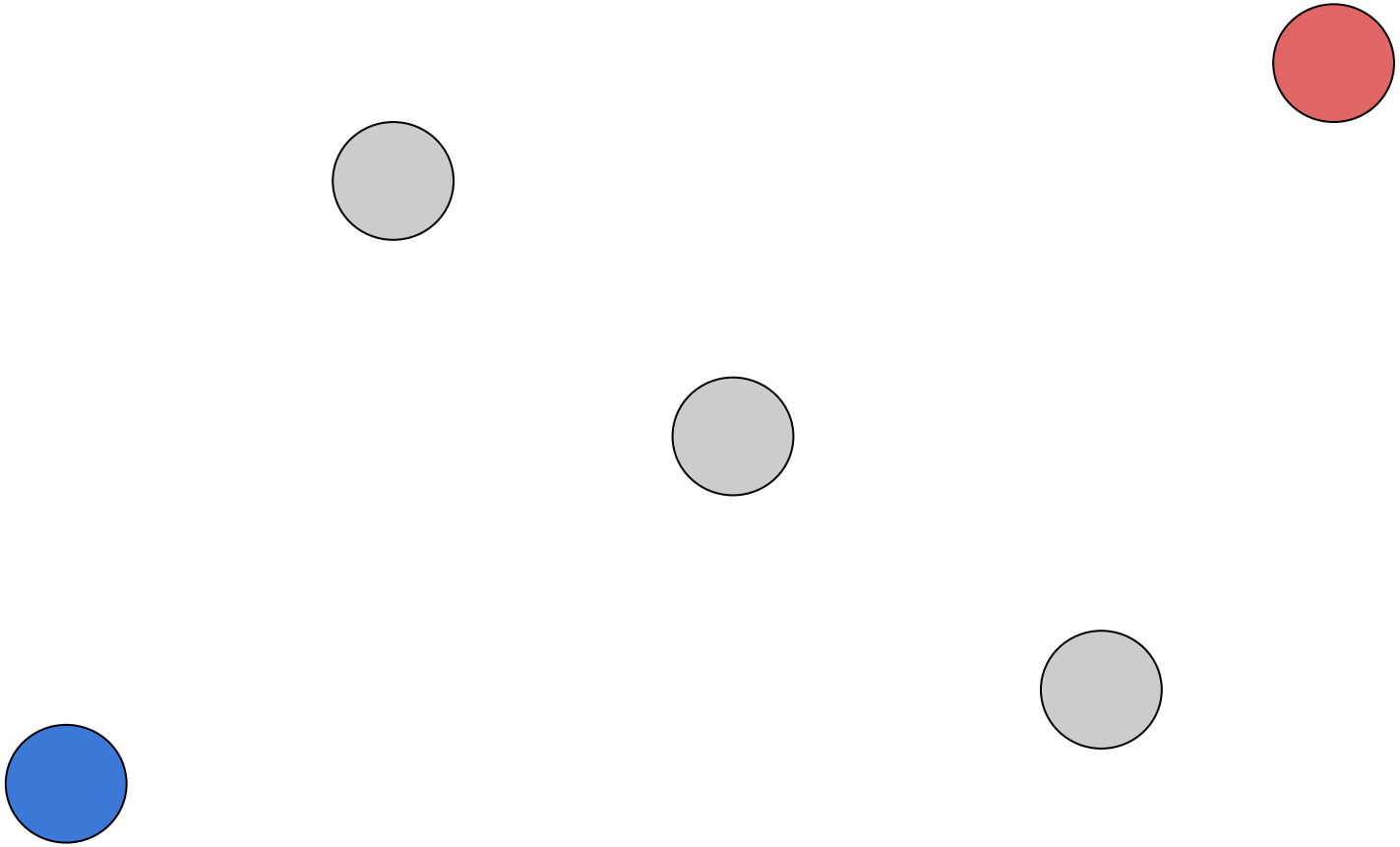
O iGo utiliza como principal algoritmo o conhecido algoritmo de Dijkstra. O algoritmo de Dijkstra, concebido pelo cientista da computação holandês Edsger Dijkstra em 1956 e publicado em 1959, soluciona o problema do caminho mais curto num grafo dirigido ou não dirigido com arestas de peso não negativo, no tempo computacional abaixo:

$$O(E + V \log(V))$$

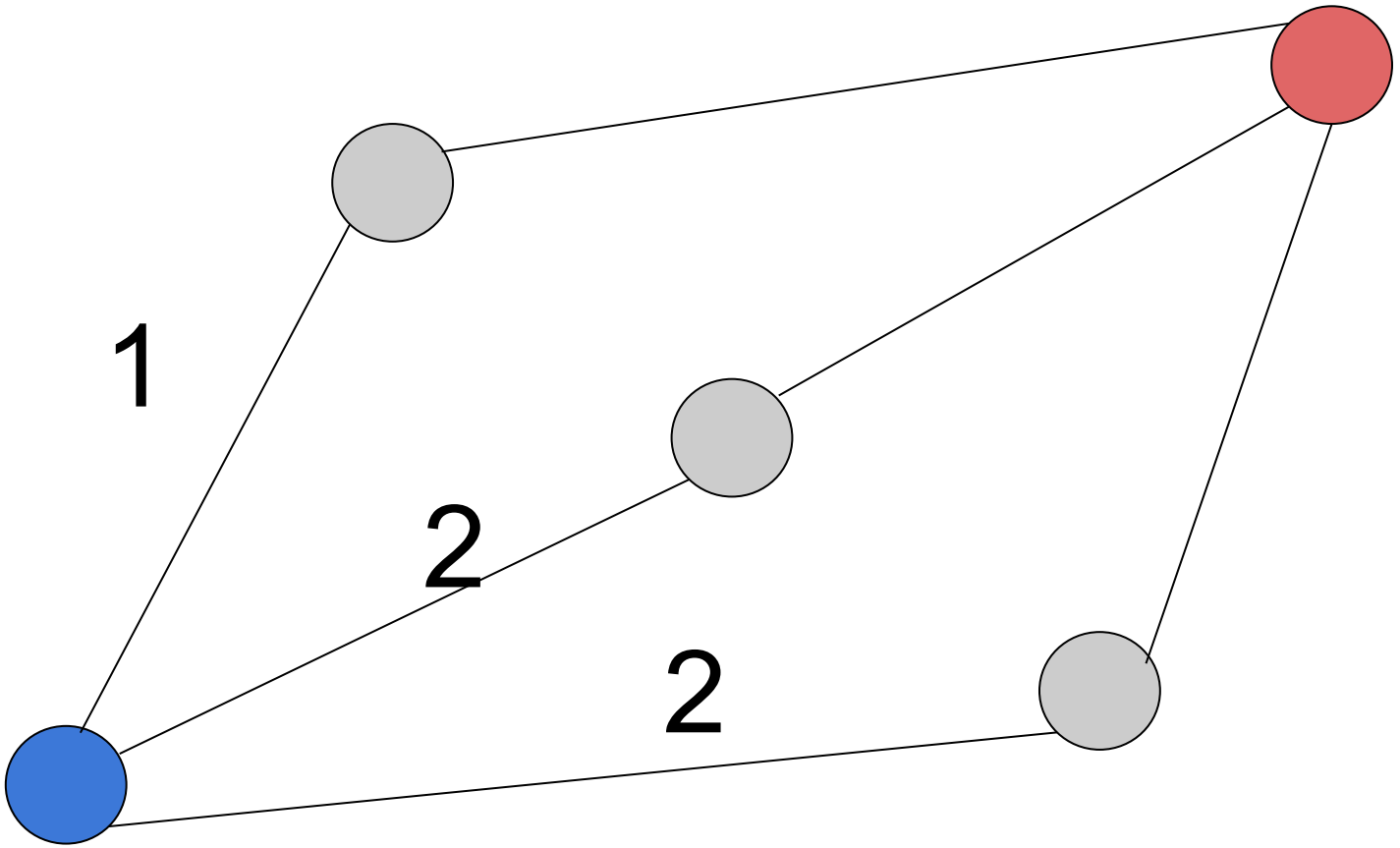
onde V é o número de vértices e E é o número de arestas.

Como funciona?

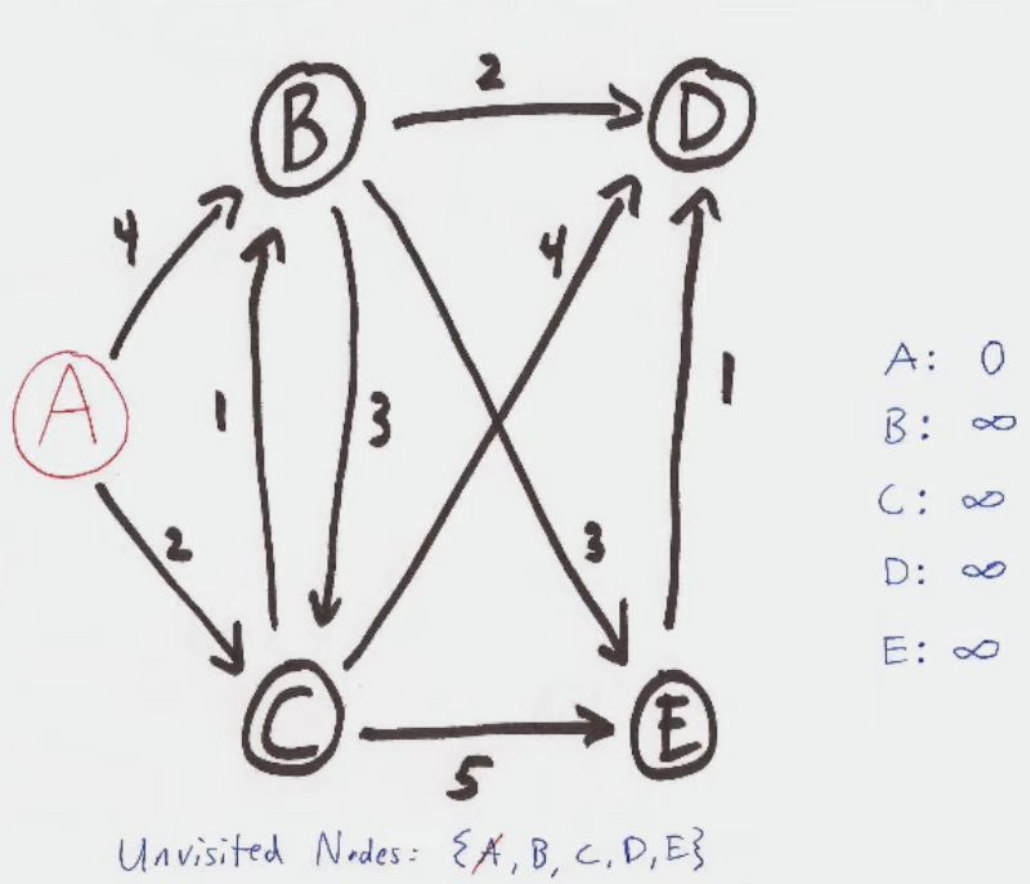
Como funciona?



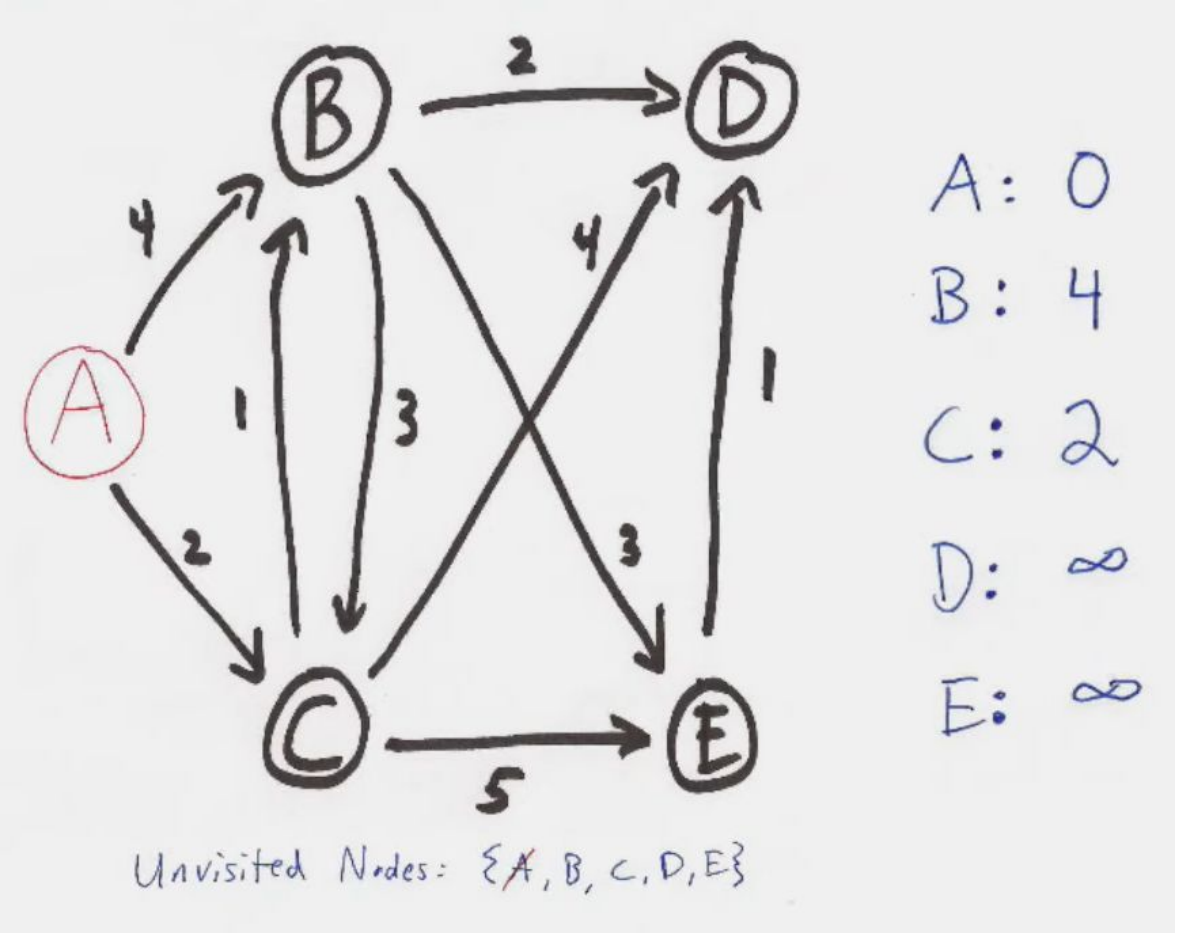
Como funciona?



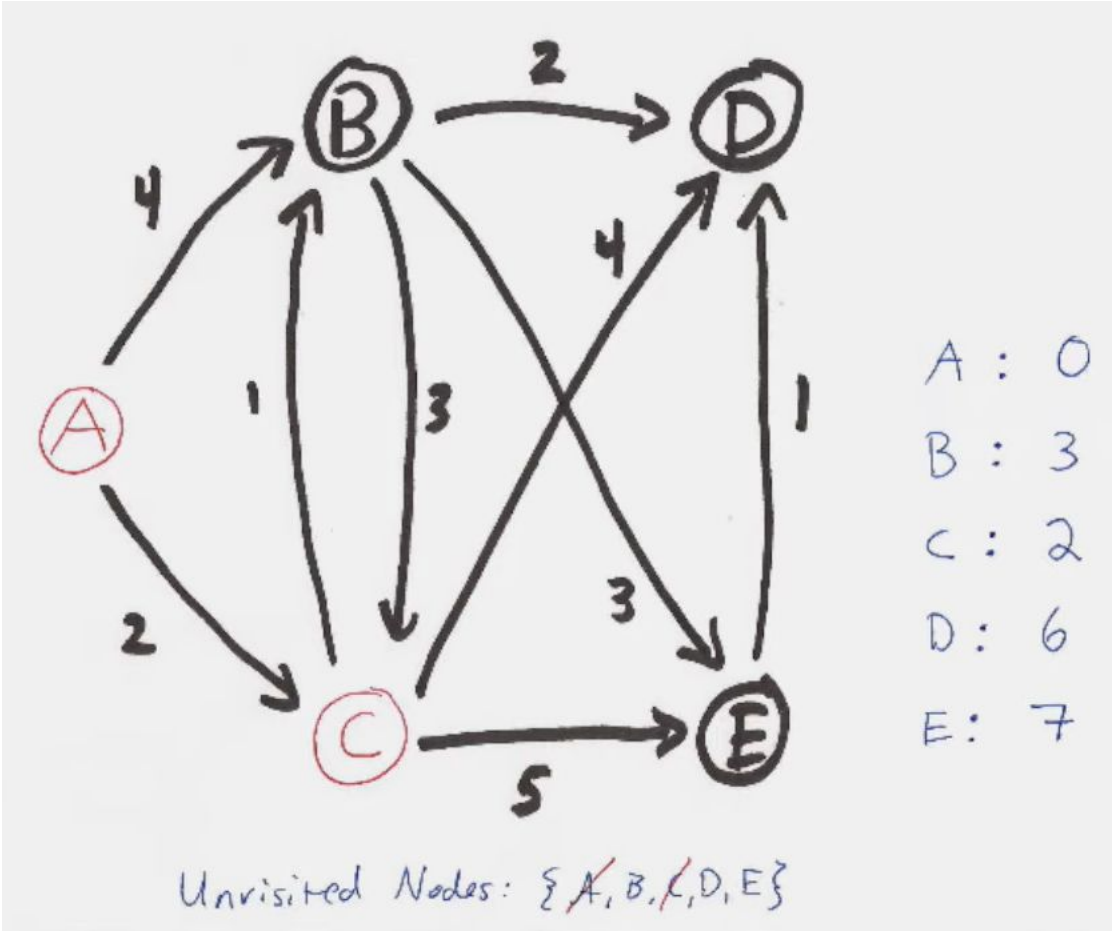
Como funciona?



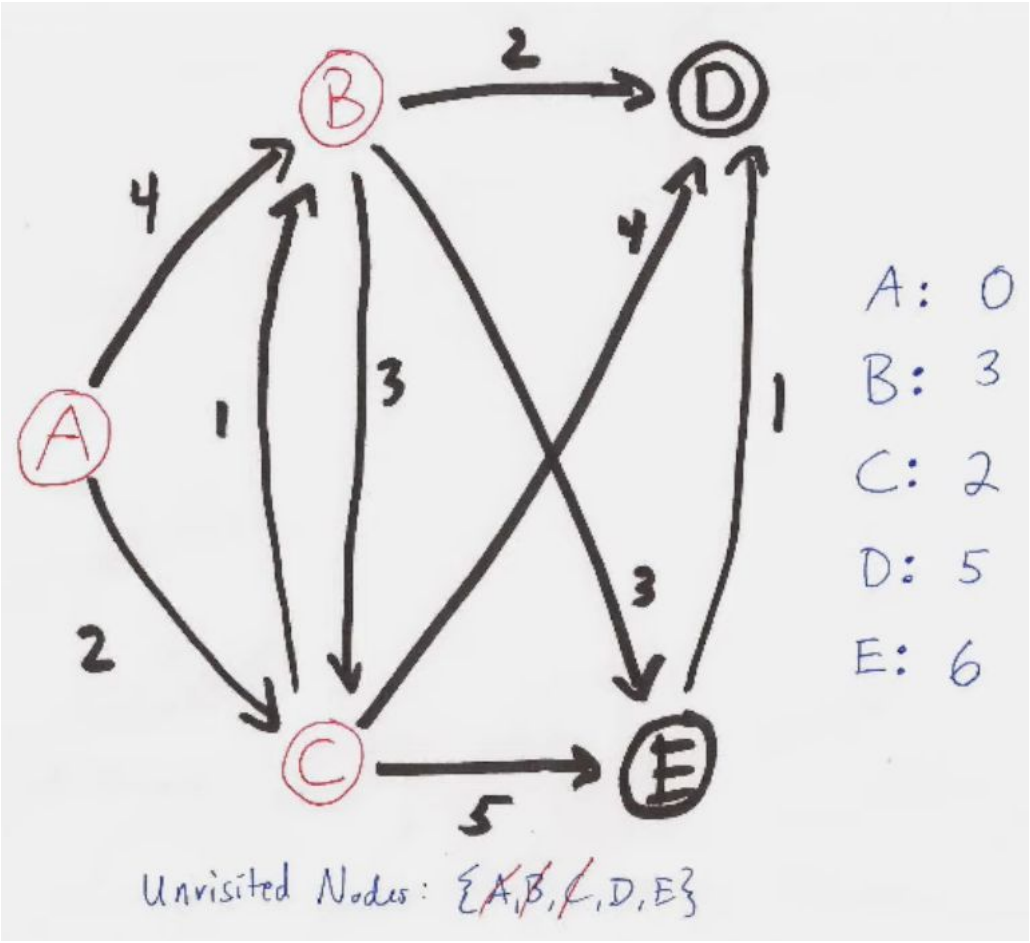
Como funciona?



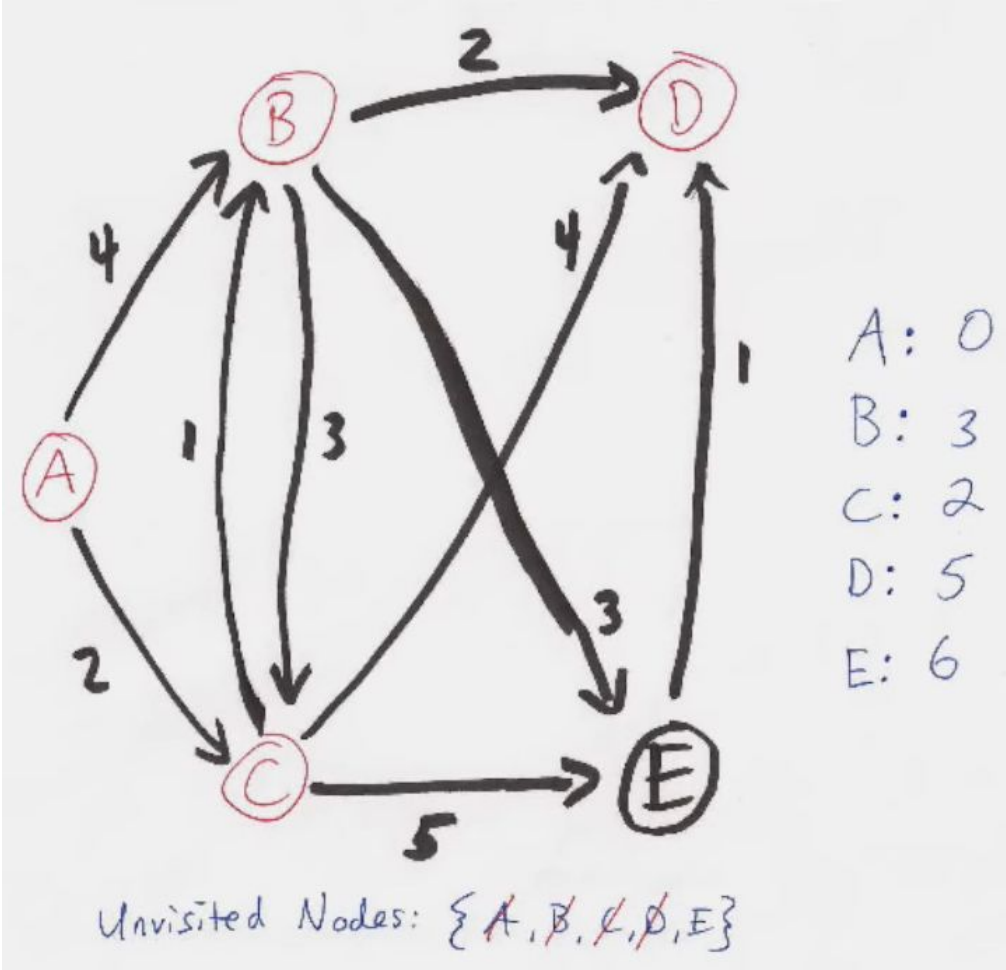
Como funciona?



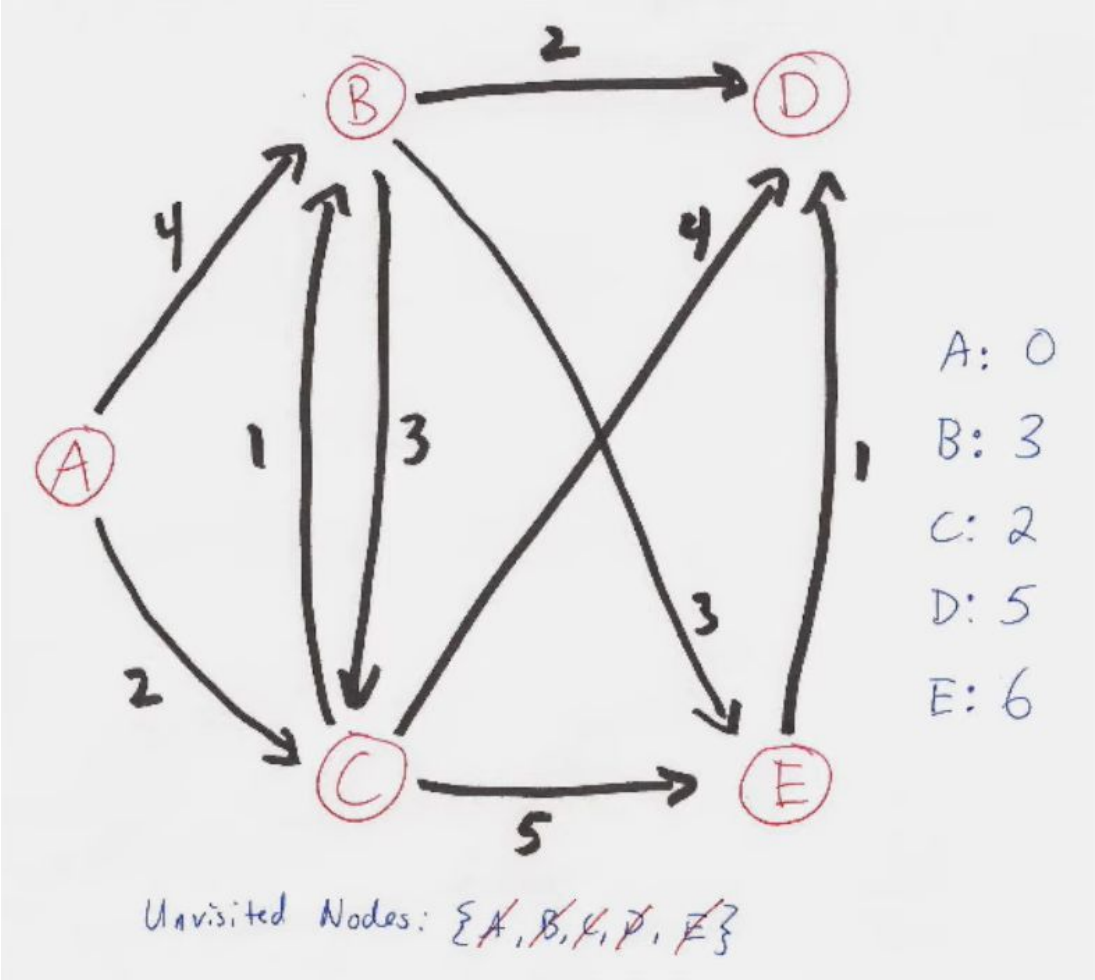
Como funciona?



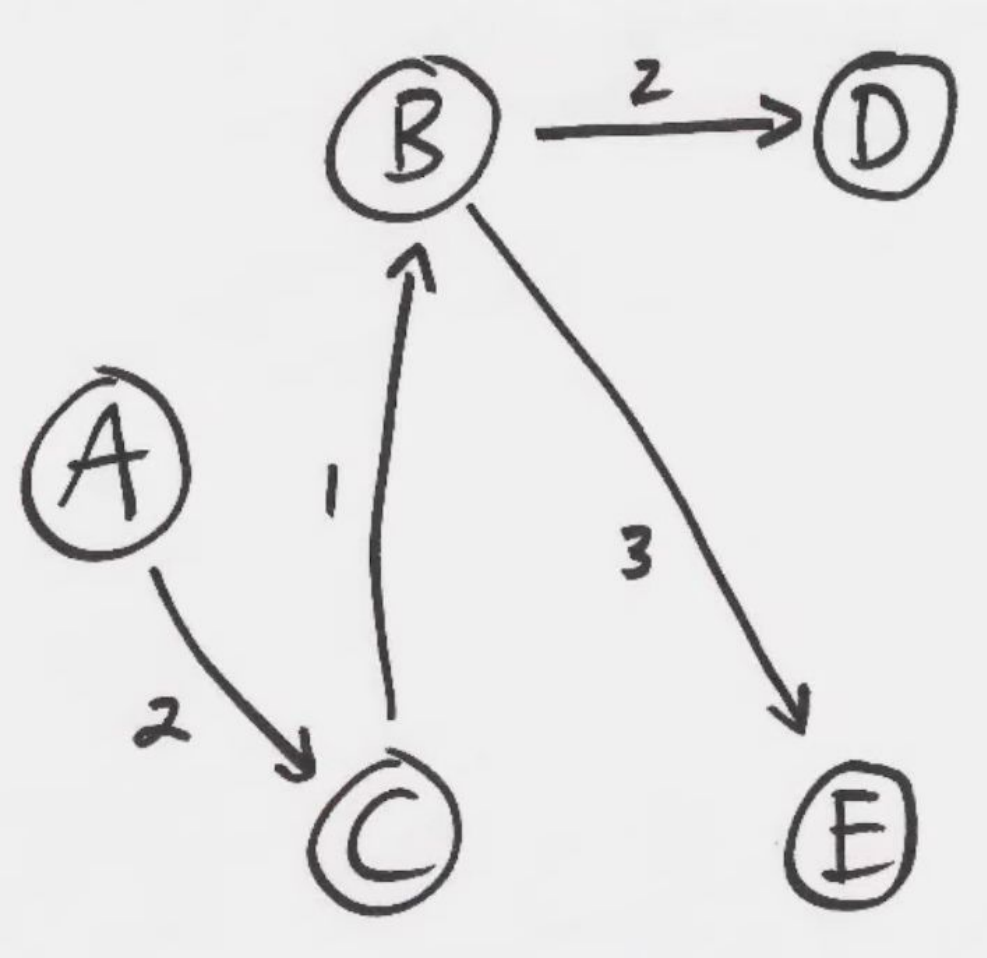
Como funciona?



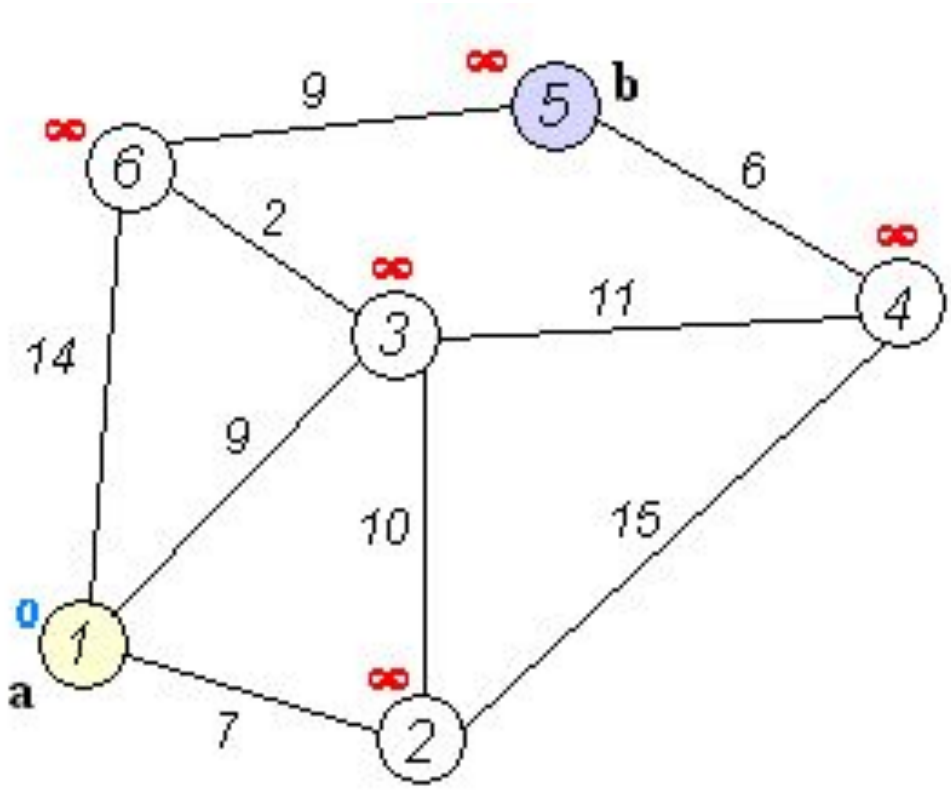
Como funciona?



Como funciona?



Como funciona?



Estruturas utilizadas

Grafos

nome	tipo	descrição
Nós	Lista encadeada de nós	Lista de nós que compoem o grafo
Contagem do nó	int	Contador atrelado aos nós
Tamanho do nó	int	Tamanho alocado para o nó

Estruturas utilizadas

Nós

nome	tipo	descrição
Vértices	Lista encadeada de vétices	Lista de vértices que compoem o nó
Contagem de vértices	int	Contador de vértices
Tamanho do vértice	int	Tamanho alocado para o vértice
Nome do nó	Ponteiro de char	Nome de identificação do nó

Estruturas utilizadas

Vértices

nome	tipo	descrição
Origem	Ponteiro de nós	Indicação do nó de origem
Destino	Ponteiro de nós	Indicação do nó de destino
Peso	int	Peso atrelado ao vértice

Estruturas utilizadas

Pilha

nome	tipo	descrição
Nós	Ponteiro de nós	Indicação do nó do topo
Tamanho	int	Indicação da quantidade máxima
Contador	int	Indicação da quantidade atual
Grafo	Ponteiro de grafo	Grafo ao qual se refere a pilha

Funcionalidades

- Selecionar mapa
- Mostrar mapa
- Ver tráfego atual do mapa
- Definir destino
- Obter melhor rota
- Rota aleatória
- Inserir destino
- Inserir conexão entre destinos
- Atualizar trânsito
- Visualizar log de rotas passadas
- Visualizar todos os destinos cadastrados
- Modo depuração para visualização de todas as rotas

Tecnologias

- Linguagem: C puro
- “Banco de Dados”: Arquivos txt
- Organização de estruturas: Grafos, pilhas…
- Ordenação de locais : Quicksort para ‘strings’
- Busca sequencial para geração de destinos aleatórios.

Telas


iGO
* Caminhos inteligentes. *
Digite um numero correspondente no menu:
1. Calcular rota especifica;
2. Verificar mapa e transito atual;
3. Visualizar log de rotas;
4. Modo de depuracao;
5. Apagar log de rotas;
6. Atualizar transito;
7. Incluir ligacao no mapa;
8. Incluir local no mapa;
9. Mostrar locais do mapa;
10. Calcular uma rota aleatoria;
11. Sair;

Telas

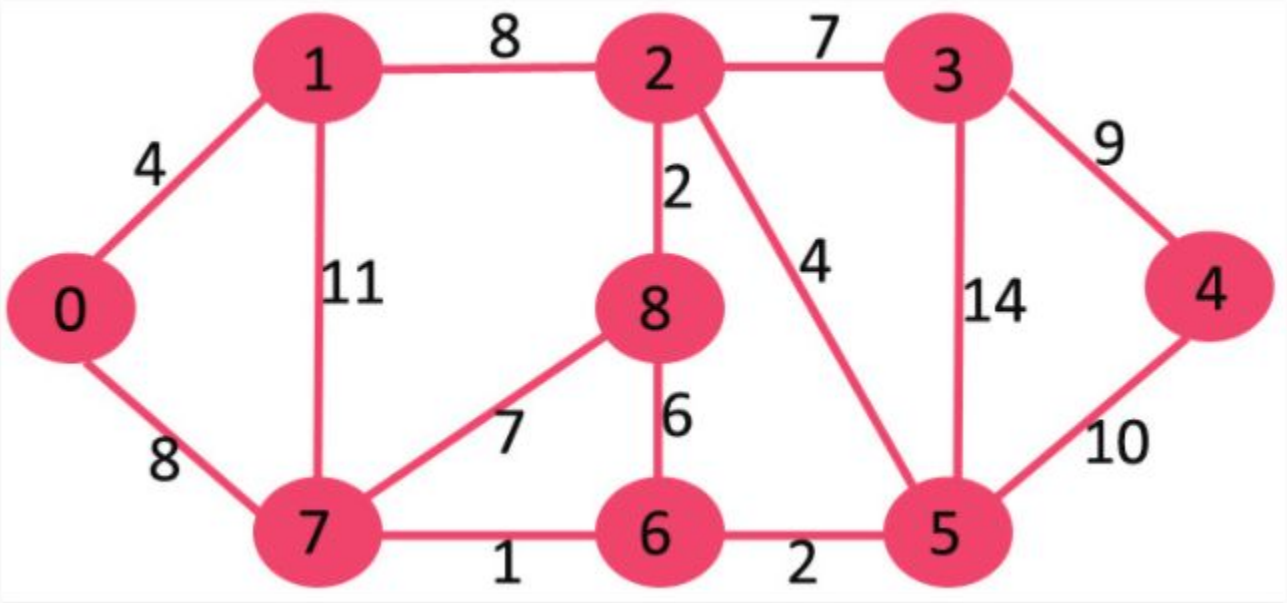
```
Onde voce esta?
Minha-Casa

Onde deseja ir?
Flamboyant
"Minha-Casa" -> "Rua-Buriti" [ETA: 4 min];
"Rua-Buriti" -> "Rua-Bela-Vista" [ETA: 5 min];
"Rua-Bela-Vista" -> "Av-Jamel-Cecilio" [ETA: 5 min];
"Av-Jamel-Cecilio" -> "Flamboyant" [ETA: 6 min];
ETA total: 20 min | Este trajeto geralmente e feito em 17 min | Condicao: Transito moderado-pesado.
```

Lógica de Funcionamento

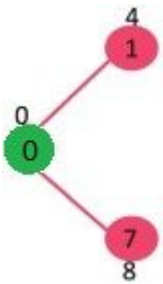
1. Cria “Min-Heap” com todos os nós.
2. Distância origem -> origem = 0; Restante = Infinito
3. Enquanto a “Min-Heap” não estiver vazia, executa:
    a. Extrai o vértice com menor valor, denominado u.
    b. Para cada vértice adjacente v de u, verifique se v está na “Min Heap”. Se v estiver na “Min Heap” e o valor da distância for maior que o peso de u-v mais o valor da distância de u, atualize o valor da distância de v.

Exemplo

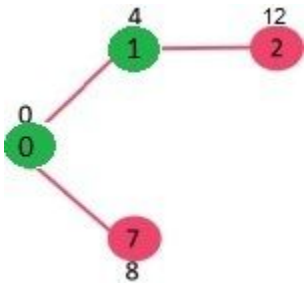


Lógica de Funcionamento

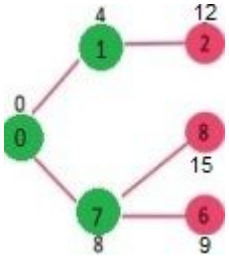
1. Cria uma “Min-Heap” com todos os nós.
2. Nó “0” extraído. Valor do peso de cada vértice para os nós adjacentes é atualizado, no caso, nós 1 e 7.
3. Nó “1” extraído. Distância para o nó 2 atualizada.
4. Nó “7” extraído. Distância para os nós 8 e 6 atualizada.



2



3



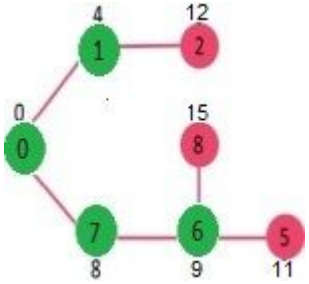
4

Lógica de Funcionamento

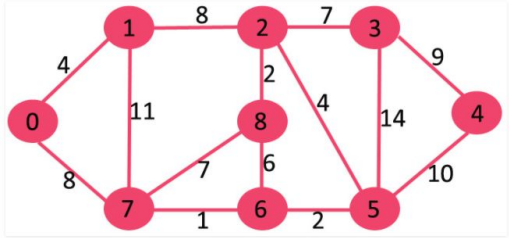
4. Nó “6” extraído. Distância para os nós 8 e 5 atualizadas.

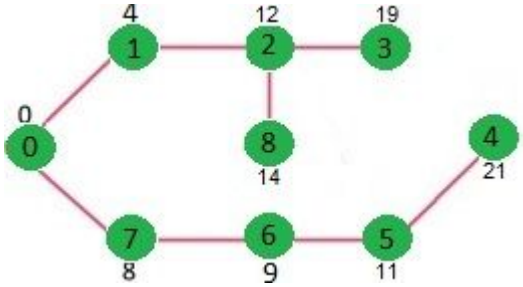
…….

Por fim, todos os nós terão sido removidos da “Min-Heap”, nos restando os menores caminhos para cada nó, partindo da origem.



4





Lógica de Funcionamento

5. Depois, montamos um array com os índices desejados para se chegar ao local e mostramos ao usuário!

Bonus