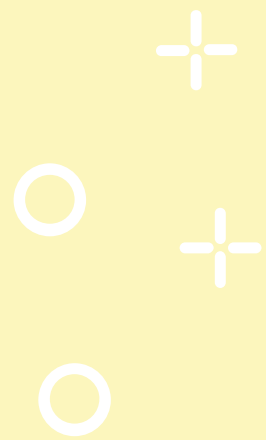


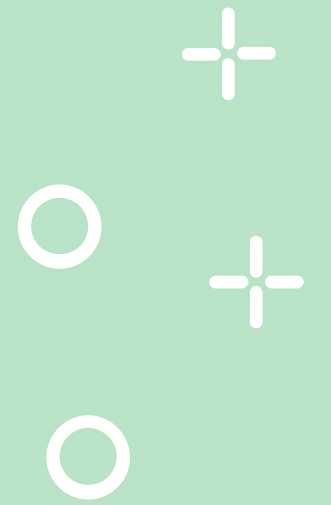
Algoritmo de Dijkstra



01

Revisão de Queue

Antes de adentrar ao tema, precisamos relembrar
de Queue



Queue

Uma queue ou fila, é uma estrutura de dados linear que segue o princípio FIFO (First In, First Out).

Sobre a implementação dela em Java, temos a biblioteca Java util que apresenta a classe Queue.

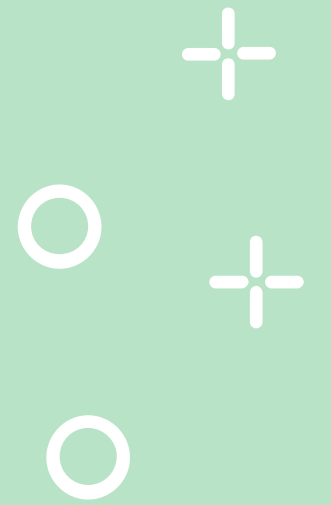
```
J Filajava > Filajava
1  import java.util.LinkedList;
2  import java.util.Queue;
3
4  public class Fila{
5      Run | Debug
6      public static void main(String[] args) {
7          Queue<Integer> fila = new LinkedList<>();
8          fila.add(e:1);
9          fila.add(e:2);
10         fila.add(e:3);
11         while(!fila.isEmpty()){
12             System.out.println(fila.poll());
13         }
14     }
```

```
1
2
3
```

02

Priority Queue

Agora que relembramos o que é Queue, vamos ver uma variação dela, a Priority Queue



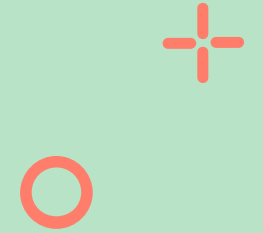
Priority Queue

Uma fila de prioridade, é um tipo de fila em que podemos determinar uma condição de prioridade. No Java essa prioridade é por padrão ascendente, mas podemos usar uma outra classe e formar um comparador personalizado.

```
J FilaDePrioridade.java > FilaDePrioridade > main(String[])
1  import java.util.PriorityQueue;
2  import java.util.Queue;
3
4  public class FilaDePrioridade {
    Run | Debug
5      public static void main(String[] args) {
6          //PriorityQueue<Integer> filaPrioridade = new PriorityQueue<>(new Comparador());
7          Queue<Integer> filaPrioridade = new PriorityQueue<>(new Comparador());
8          filaPrioridade.add(e:2);
9          filaPrioridade.add(e:3);
10         filaPrioridade.add(e:1);
11         while(!filaPrioridade.isEmpty()){
12             System.out.println(filaPrioridade.poll());
13         }
14     }
15 }
16
```

3
2
1

Priority Queue

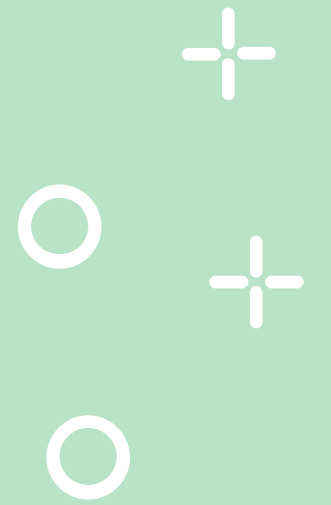


```
J Comparador.java > Comparador
1  import java.util.Comparator;
2
3  public class Comparador implements Comparator<Integer>{
4      @Override
5      public int compare(Integer o1, Integer o2) {
6          if(o1 < o2){
7              return 1;
8          }
9          return -1;
10     }
11 }
```

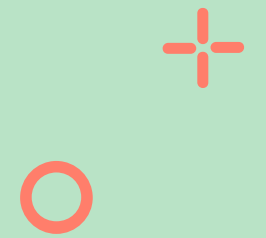
03

Introdução a Dijkstra

Vamos começar com um problema básico

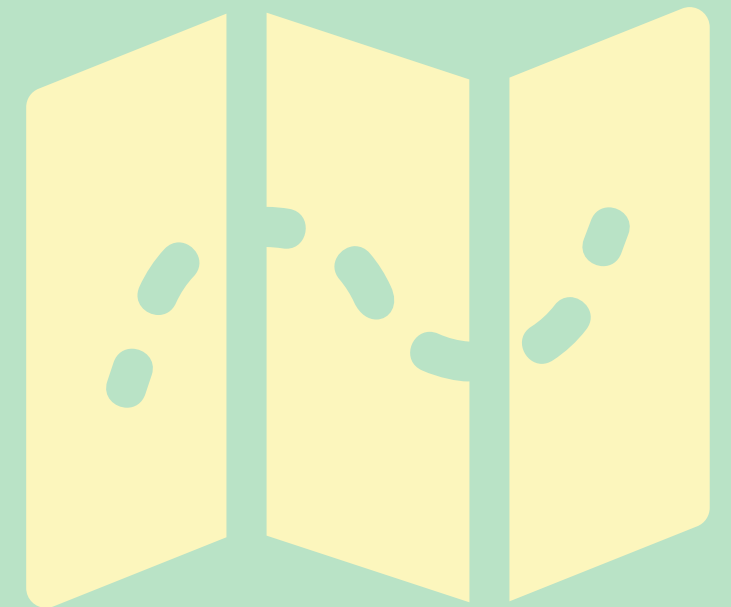



Introdução a Dijkstra



O algoritmo de Dijkstra que foi desenvolvido por Edsger Wybe Dijkstra em 1959. O objetivo desse algoritmo é encontrar o menor caminho a partir de um grafo valorado.

suponha que você precisa ir de uma cidade A até uma cidade B, porém seu tanque de gasolina está na metade, por isso você quer ver qual é o menor caminho possível para essa viagem.

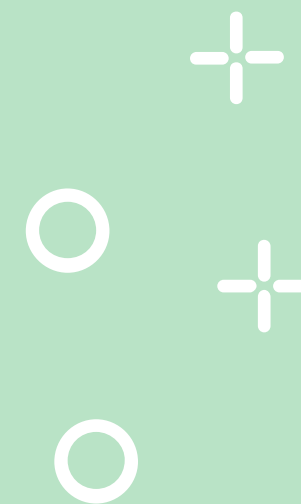




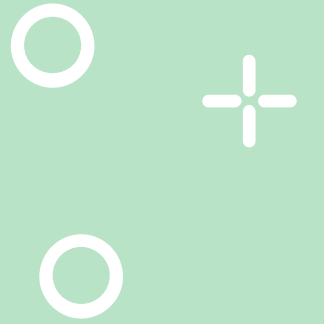
04

Dijkstra na prática

Questões com juízes online



Easy Dijkstra Problem



Caminho das Pontes



Equipe



Heloísa Fernanda
heloisa.00000845226@unicap.br



Isadora Xavier
isadora.00000844511@unicap.br



Gabriel Torres
gabriel.00000844543@unicap.br



Pedro Pepeu
pedro.00000844503@unicap.br

Obrigado!

