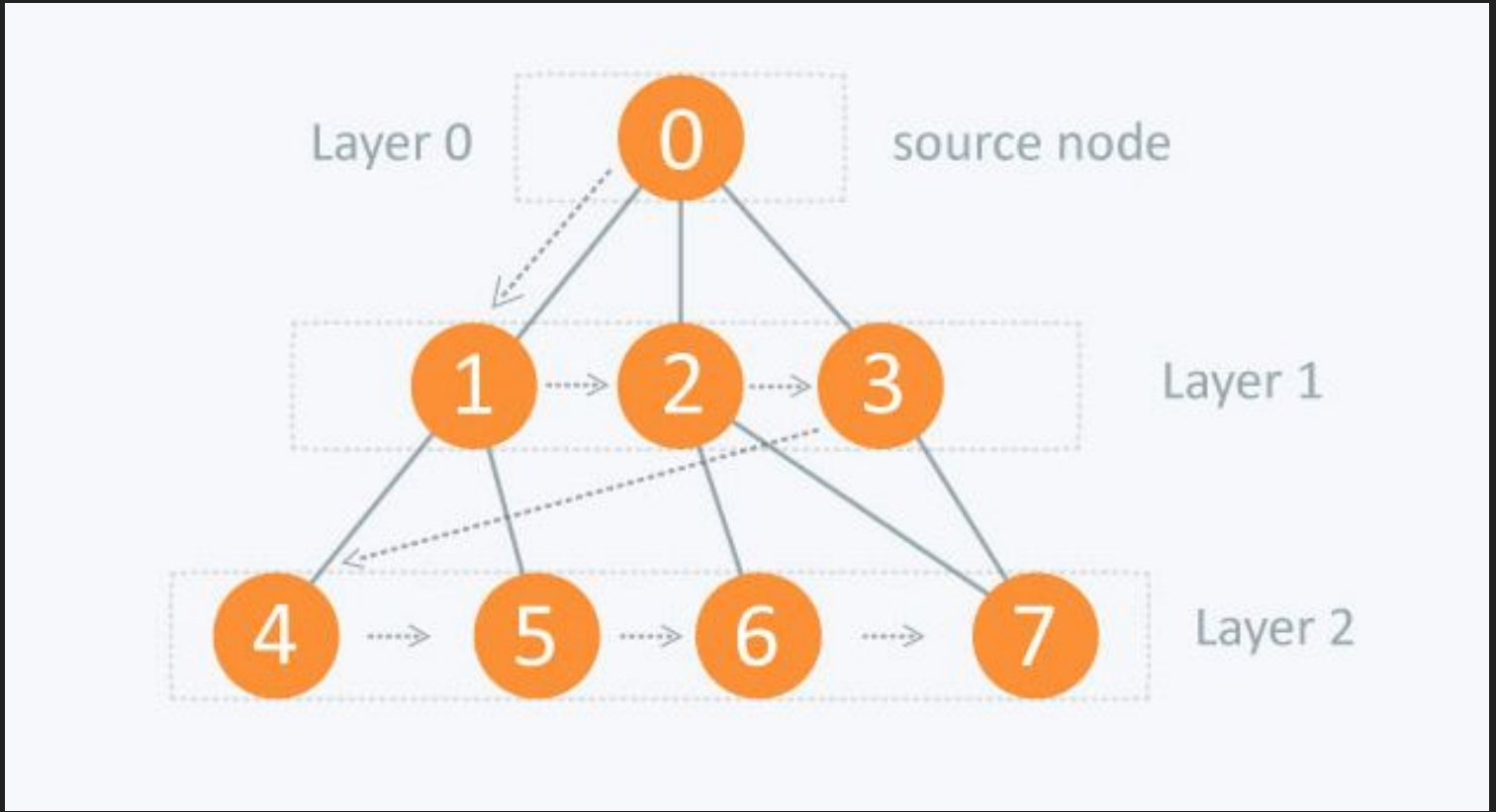
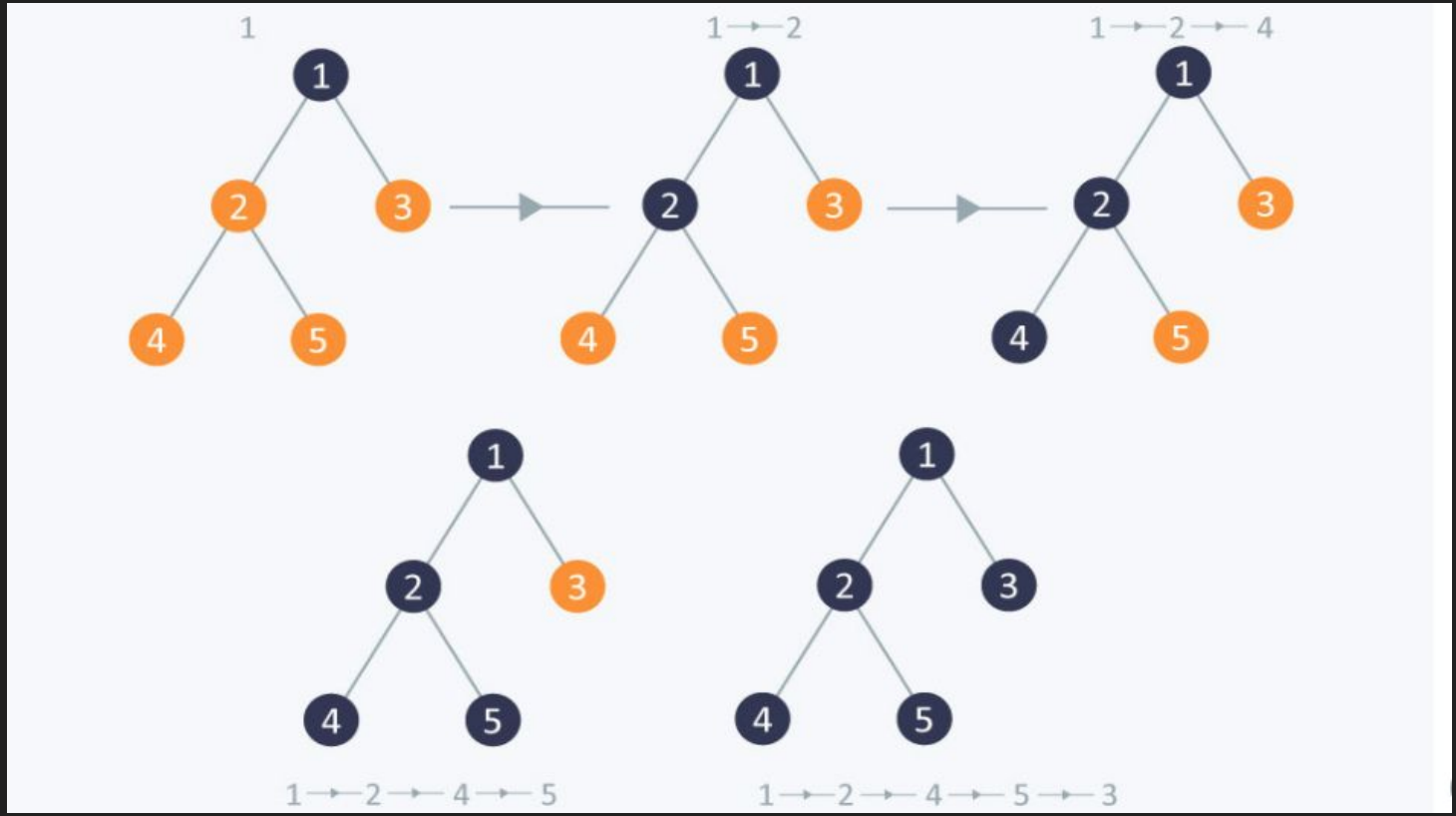
Resolvendo 8-puzzle usando BFS e DFS

Algoritmo BFS



```java
public boolean search() {
  Utils info = new Utils(); // Classe que contem algumas estruturas de dados hashmap, fila, pilha
  TabuleiroNo node = initialNode;
  info.queue.add(node);  //No inicial é adicionado na fila

  while(!(info.queue.isEmpty())) {   // Loop continua enquanto a fila não estiver vazia
    node = info.queue.remove();   // Remove o no do inicio da fila
    info.visited.put(node.hashCode(), node);   //Coloca o nó no hashMap como visitado

    if (node.isGaol()) {  //Se o objetivo for encontrado, um caminho é criado e um caminho é impresso
      PathActions p = new PathActions(initialNode, node);
      p.printPath();
      TabuleiroNo.setExisteSolucao(true);
      return true;
    }

    Successor s = new Successor(); //Função sucessora que fornece os filhos do nó
    List<TabuleiroNo> list = s.successor(node);

    for(TabuleiroNo temp: list) {
      boolean vis = info.visited.containsKey(temp.hashCode());
      if(!vis)info.queue.add(temp);
    }
  }

  return false;
}
```

Algoritmo DFS



```java
public boolean search() {
  Utils info = new Utils(); // Classe que contem algumas estruturas de dados hashmap, fila, pilha
  TabuleiroNo node = initialNode;
  info.stack.push(node);  // Uma pilha é usada para simular uma recursao

  while(!(info.stack.isEmpty())) { // Loop continua enquanto a pilha não estiver vazia
    node = info.stack.pop(); // Remove o no do inicio da pilha
    info.visited.put(node.hashCode(), node);// Coloca o nó no hashMap como visitado
    // Se o objetivo for encontrado, um caminho é criado e um caminho é impresso
    if (node.isGaol()) {
      PathActions p = new PathActions(initialNode, node);
      p.printPath();
      TabuleiroNo.setExisteSolucao(true);
      return true;
    }

    Successor s = new Successor(); // Função sucessora que fornece os filhos do nó
    List<TabuleiroNo> list = s.successor(node);

    for(TabuleiroNo temp: list) {
      boolean vis = info.visited.containsKey(temp.hashCode());
      if(!vis)info.stack.push(temp);
    }
  }
  return false;
}
```

Matéria: Programação lógica e inteligencia artificial

Professor: Marcelo Rodrigues de Sousa

Dupla: Bruno Boreli 11921ECP005 e Guilherme Gabriel 11921ECP001