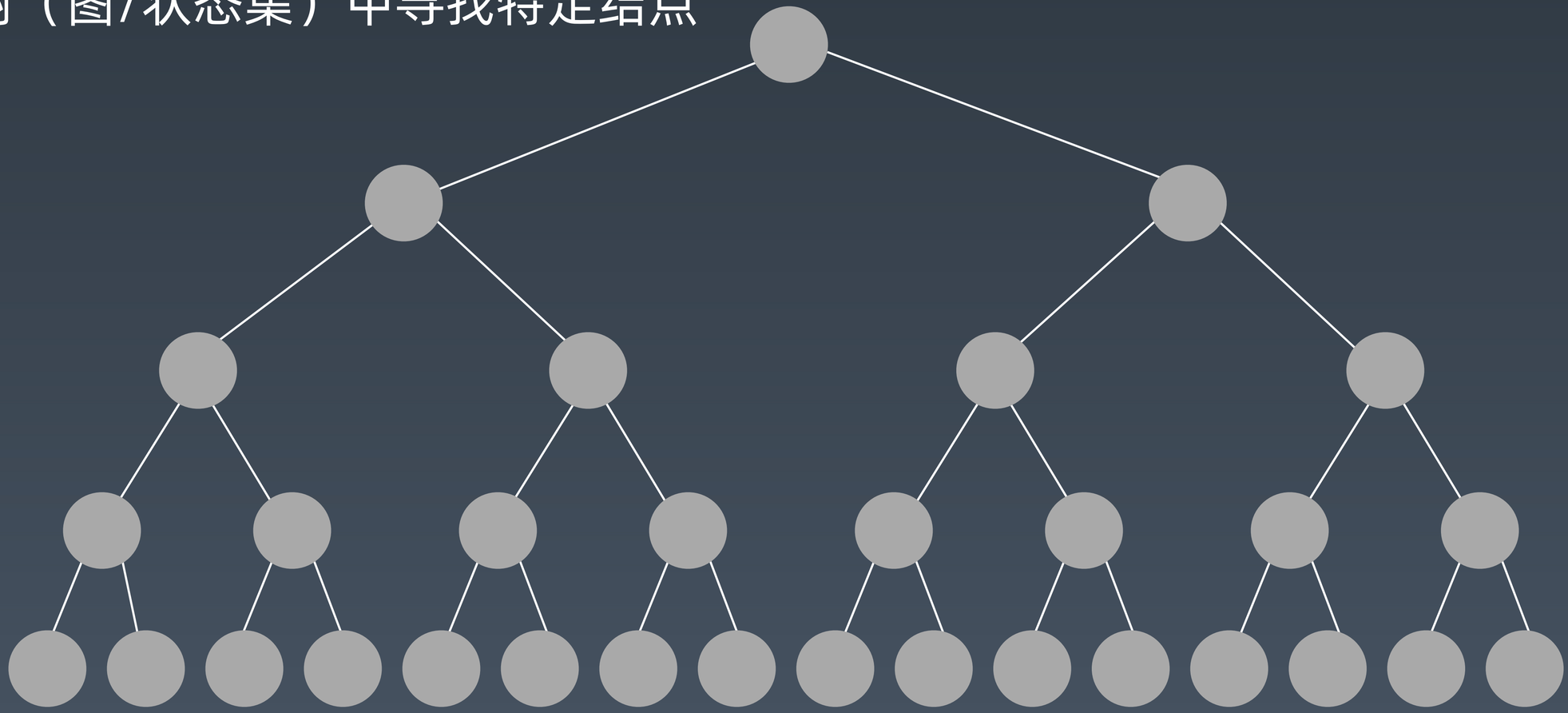
极客大学算法训练营
第九课
深度优先搜索和广度优先搜索

覃超

Sophon Tech 创始人，前 Facebook 工程师

遍历搜索

在树（图/状态集）中寻找特定结点



示例代码

Python

```python
class TreeNode:
  def __init__(self, val):
    self.val = val
    self.left, self.right = None, None
```

Java

```java
public class TreeNode {
  public int val;
  public TreeNode left, right;
  public TreeNode(int val) {
    this.val = val;
    this.left = null;
    this.right = null;
  }
}
```

C++

```cpp
struct TreeNode {
  int val;
  TreeNode *left;
  TreeNode *right;
  TreeNode(int x) : val(x), left(NULL), right(NULL) {}
}
```

搜索 - 遍历

- 每个节点都要访问一次
- 每个节点仅仅要访问一次
- 对于节点的访问顺序不限
  - 深度优先：depth first search
  - 广度优先：breadth first search

示例代码

```
def dfs(node):

    if node in visited:
        # already visited
        return

    visited.add(node)

    # process current node
    # ... # logic here
    dfs(node.left)
    dfs(node.right)
```

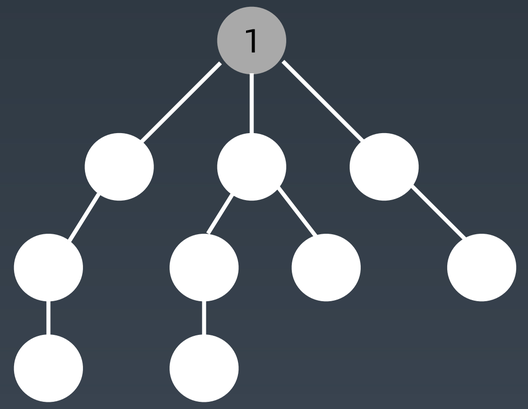
深度优先搜索

Depth-First-Search

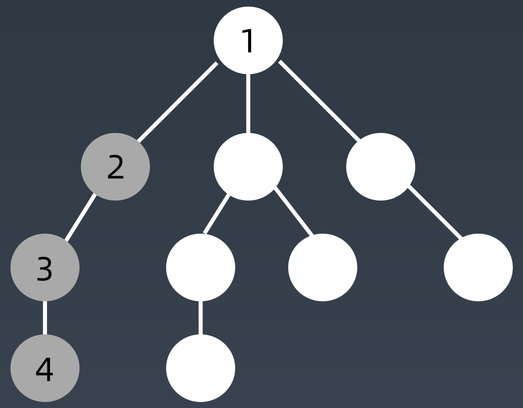
DFS 代码 - 递归写法

```
visited = set()

def dfs(node, visited):
  if node in visited: # terminator
     # already visited
     return

   visited.add(node)

   # process current node here.
   ...
   for next_node in node.children():
      if not next_node in visited:
         dfs(next node, visited)
```

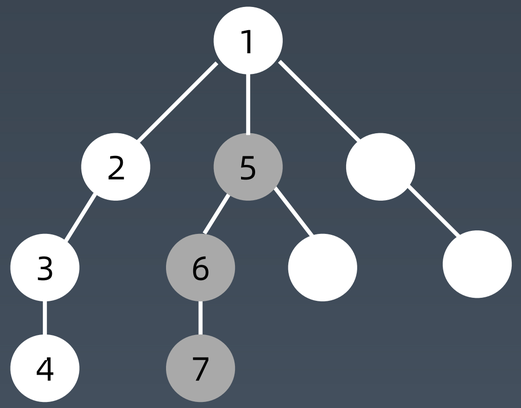
遍历顺序



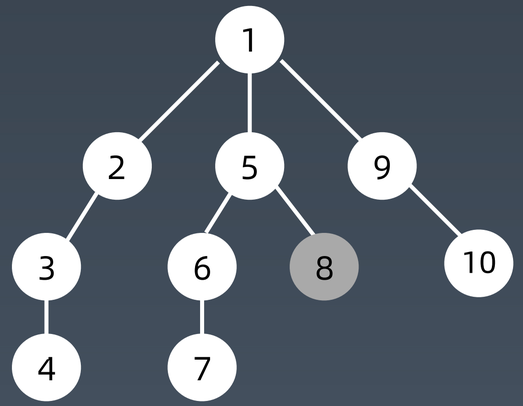
1



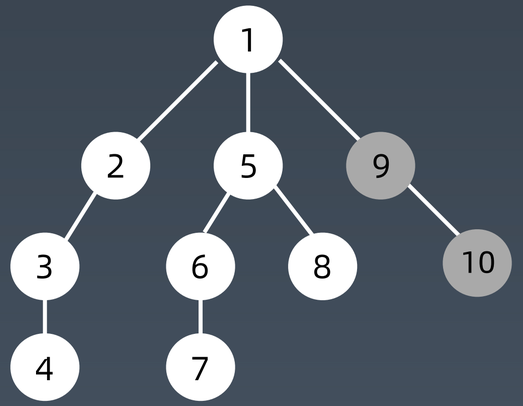
2



3

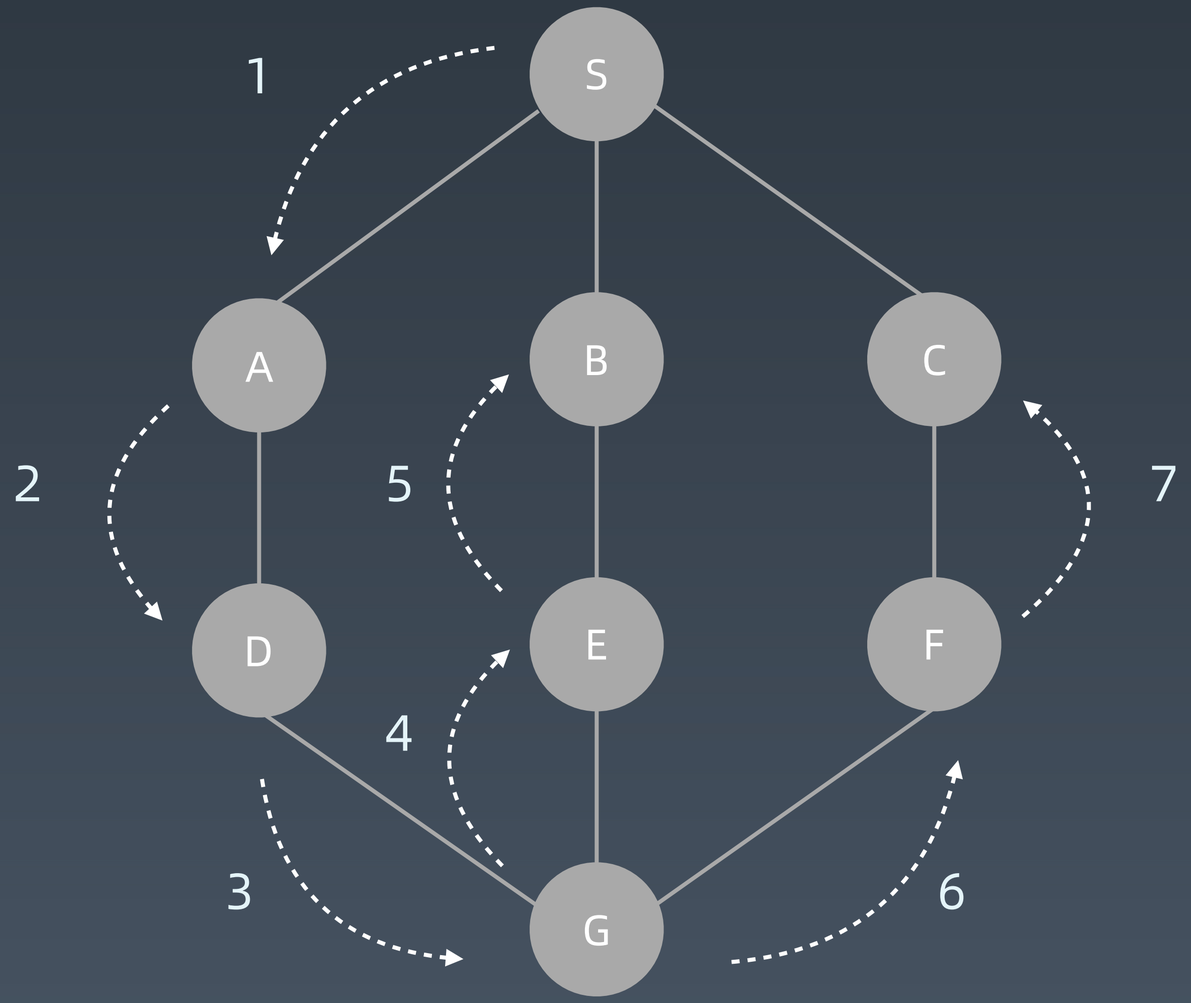


4



5

遍历顺序



DFS 代码 - 递归写法

```
visited = set()

def dfs(node, visited):
  if node in visited: # terminator
     # already visited
     return

   visited.add(node)

   # process current node here.
   ...
   for next_node in node.children():
      if not next_node in visited:
         dfs(next_node, visited)
```

DFS 代码 - 非递归写法

```
def DFS(self, tree):

    if tree.root is None:
        return []

    visited, stack = [], [tree.root]

    while stack:
        node = stack.pop()
        visited.add(node)

        process (node)
        nodes = generate_related_nodes(node)
        stack.push(nodes)

    # other processing work
    ...
```

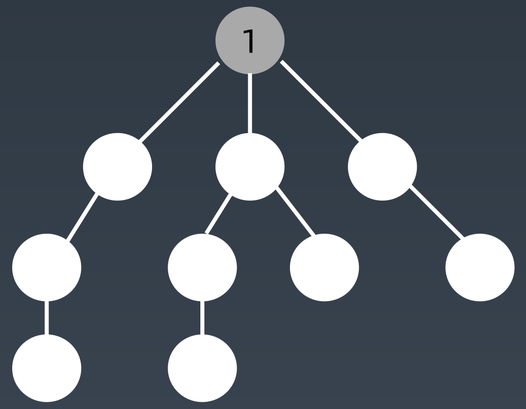
DFS 代码 - 递归写法

```
visited = set()

def dfs(node, visited):
    visited.add(node)

    # process current node here.
    ...
    for next_node in node.children():
        if not next_node in visited:
            dfs(next node, visited)
```

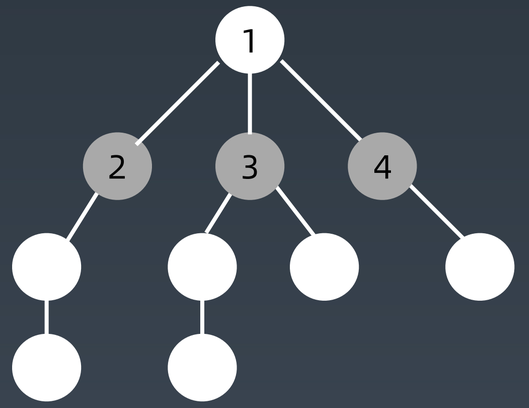
广度优先搜索

Breadth-First-Search

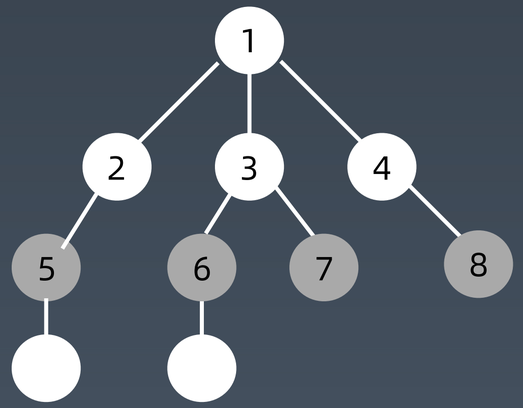
遍历顺序



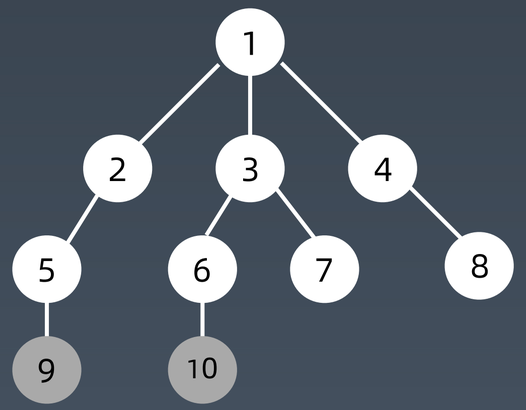
1



2

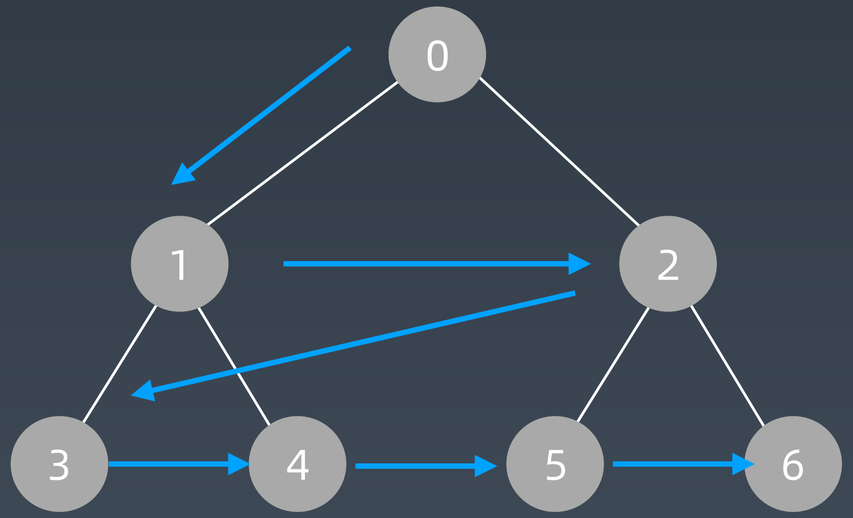


3

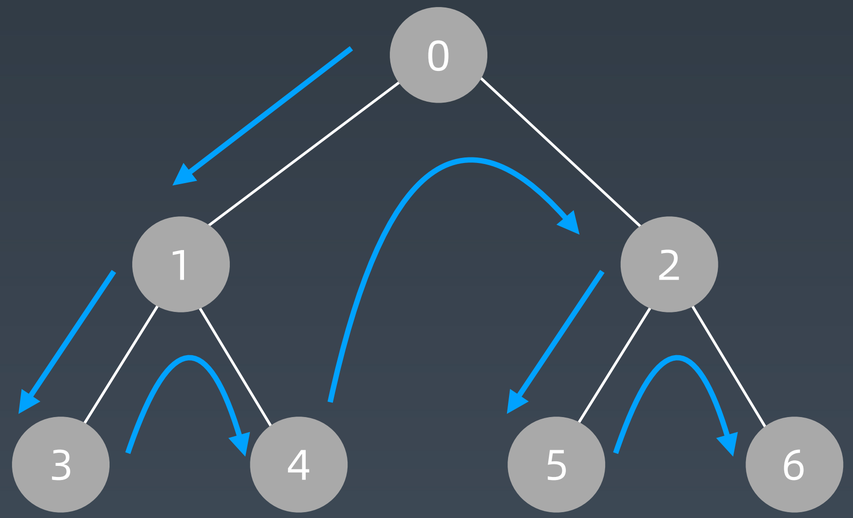


4

遍历顺序



BFS



DFS

BFS 代码

```
def BFS(graph, start, end):

    queue = []
    queue.append([start])
    visited.add(start)

    while queue:
        node = queue.pop()
        visited.add(node)

        process(node)
        nodes = generate_related_nodes(node)
        queue.push(nodes)

    # other processing work
    ...
```

DFS 代码 - 递归写法

```
visited = set()


def dfs(node, visited):

    visited.add(node)

    # process current node here.
    ...
    for next_node in node.children():
        if not next_node in visited:
            dfs(next node, visited)
```

BFS 代码

```
def BFS(graph, start, end):

    queue = []
    queue.append([start])
    visited.add(start)

    while queue:
        node = queue.pop()
        visited.add(node)

        process(node)
        nodes = generate_related_nodes(node)
        queue.push(nodes)
```

实战题目

1. https://leetcode-cn.com/problems/binary-tree-level-order-traversal/#/description

2. https://leetcode-cn.com/problems/minimum-genetic-mutation/#/description

3. https://leetcode-cn.com/problems/generate-parentheses/#/description

4. https://leetcode-cn.com/problems/find-largest-value-in-each-tree-row/#/description

Homework

1. https://leetcode-cn.com/problems/word-ladder/description/
2. https://leetcode-cn.com/problems/word-ladder-ii/description/
3. https://leetcode-cn.com/problems/number-of-islands/
4. https://leetcode-cn.com/problems/minesweeper/description/

THANKS!