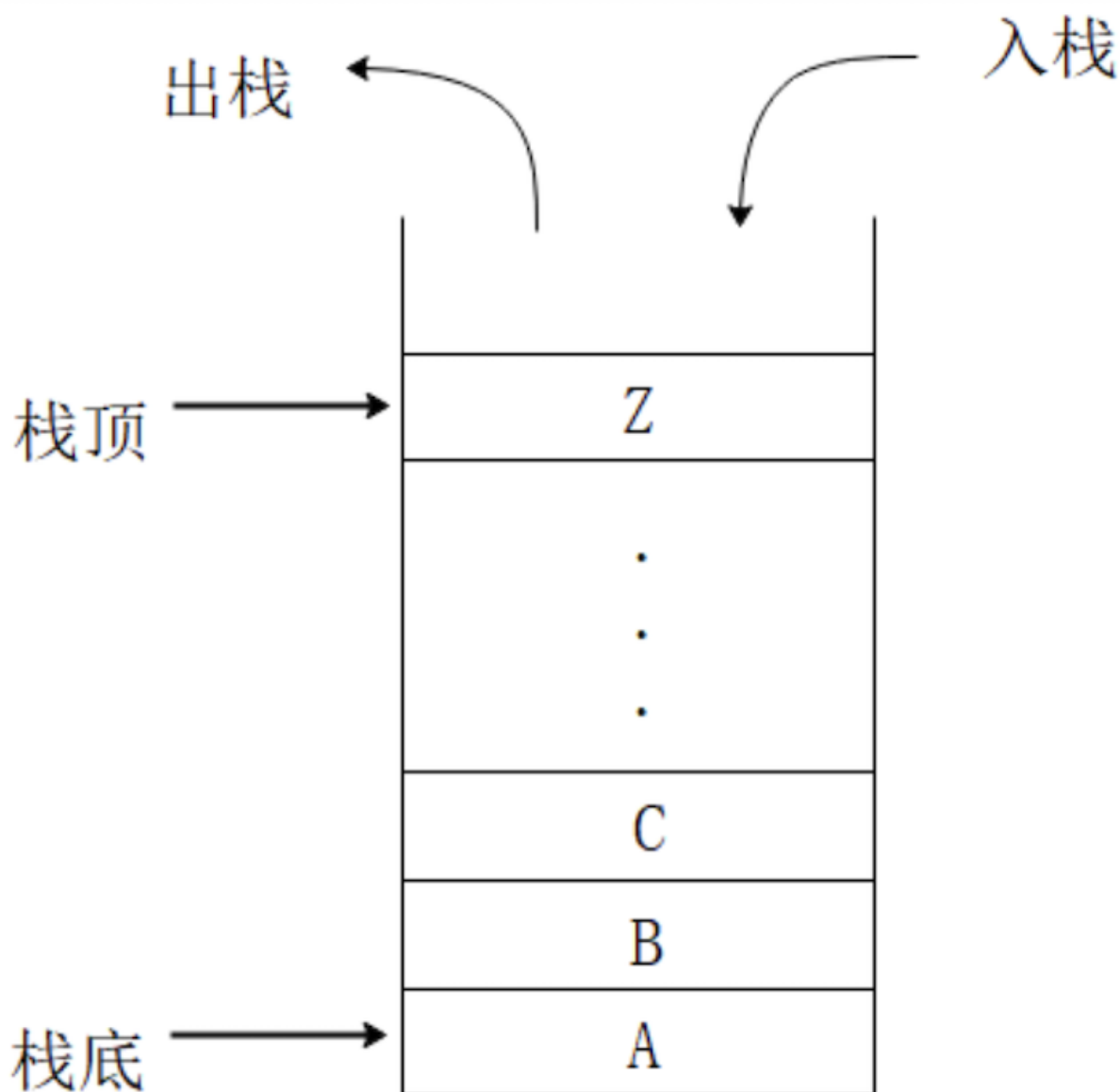


## 前端常见的数据结构

### 栈

栈（stack）是一种“先进后出”的数据结构。



实现方式：

```
// 数组实现
let arr = [];
arr.push(10); // 1, 入栈 (压栈)
arr.push(20); // 2, 入栈
arr.pop();   // 10, 出栈
arr.pop();   // 20, 出栈
```

# 队列

队列 (queue) 是一种“先进先出”的数据结构。

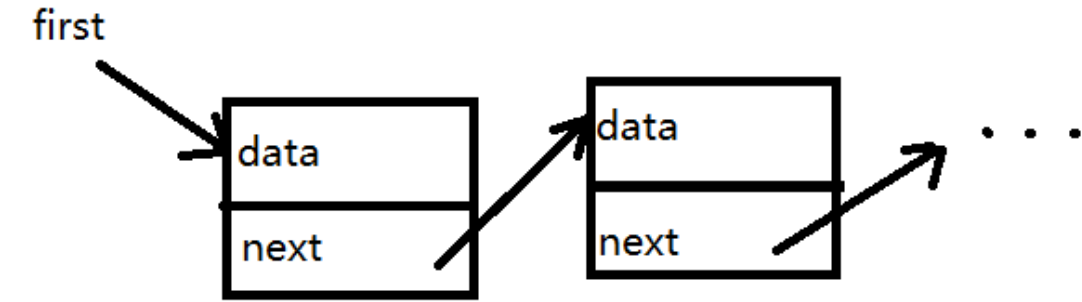


实现方式:

```
// 数组实现
let arr = [];
arr.push(10); // 1, 入队
arr.push(20); // 2, 入队
arr.shift();  // 10, 出队
arr.shift();  // 20, 出队
```

# 链表

链表 Linked list 不是连续的数据结构，而是一系列的节点组成，节点之间通过指针链接。

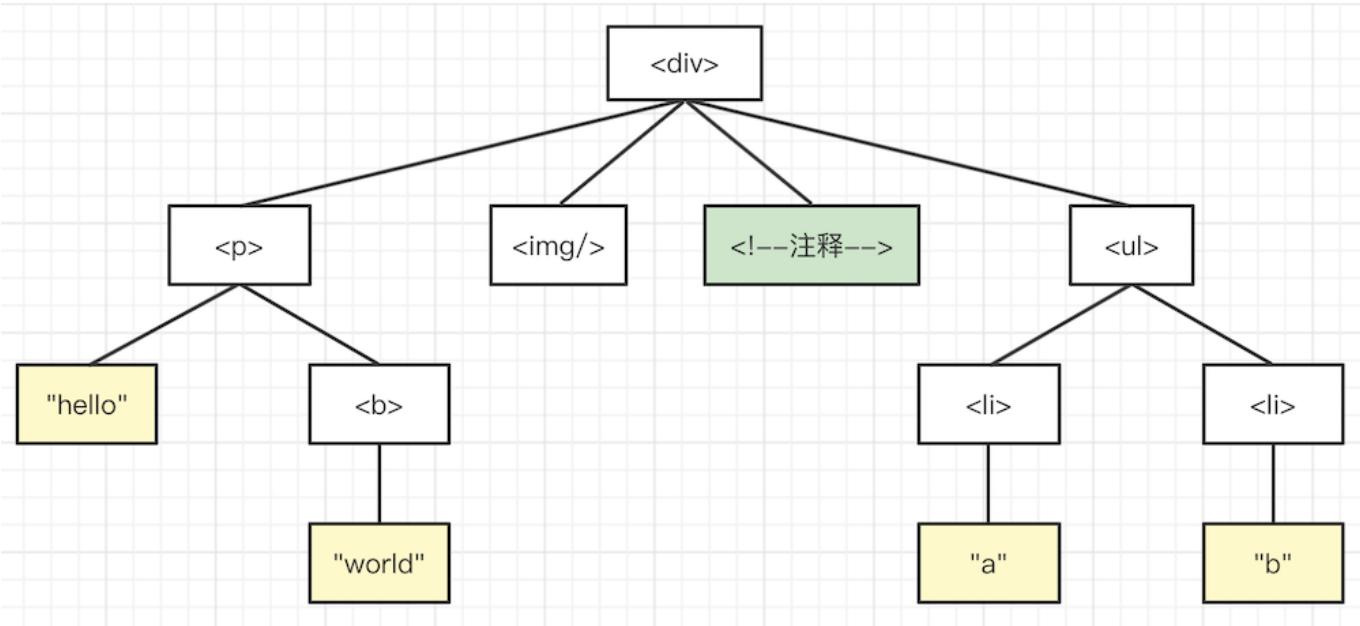


实现方式:

```
// 通过 interface 来定义链表结构
interface INodeList {
  data: any,
  next: INodeList | null;
}
```

# 树

树（tree）是一种有序的层级结构，每个节点下面可能有多个子节点。

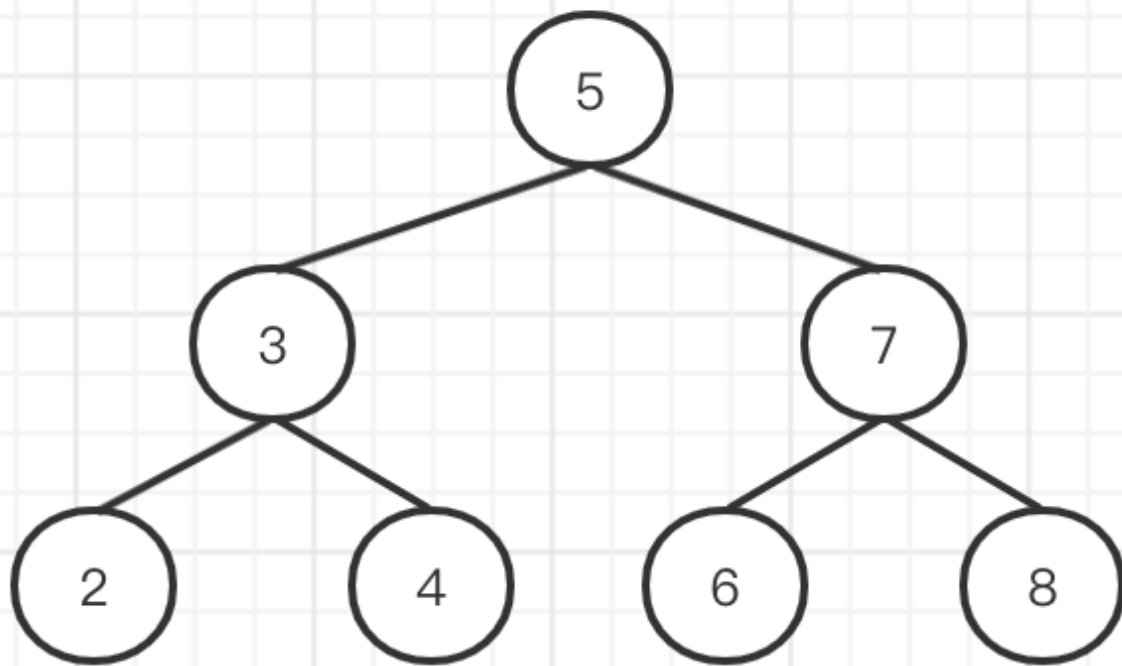


实现方式：

```
// interface 定义树结构
interface ITreeNode {
  data: any,
  children: ITreeNode[] | null
}
```

## 二叉树

二叉树（Binary Tree）是树的一种特例，它的每一个节点最多包含有两个子节点。分别为 left 和 right。



实现方式:

```
// interface 定义二叉树结构
interface IBinaryTreeNode {
  data: any;
  left: IBinaryTreeNode | null;
  right: IBinaryTreeNode | null;
}
```