极客大学算法训练营
第四课
栈、队列、双端队列、优先队列

覃超

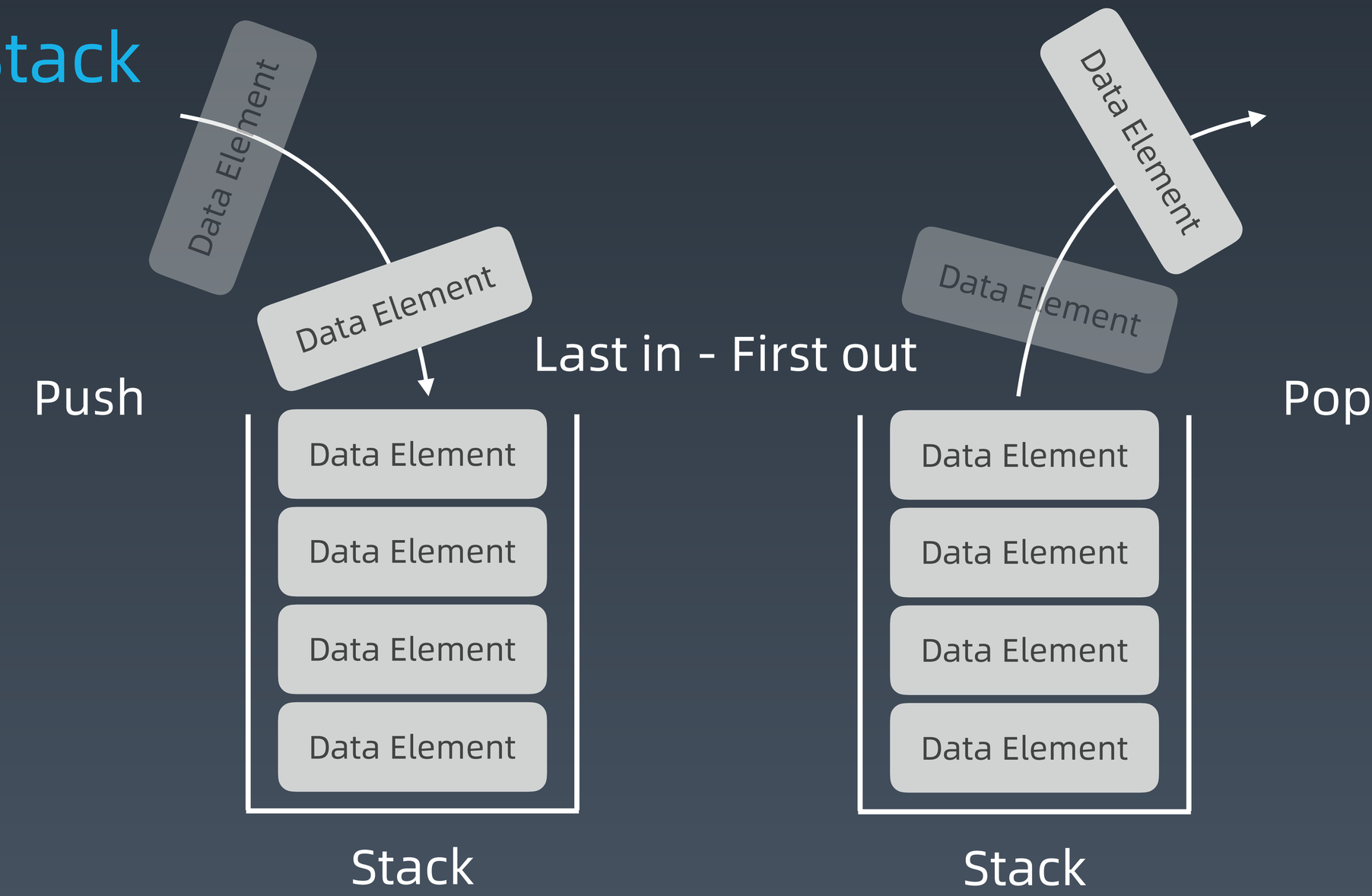
Sophon Tech 创始人，前 Facebook 工程师

目录

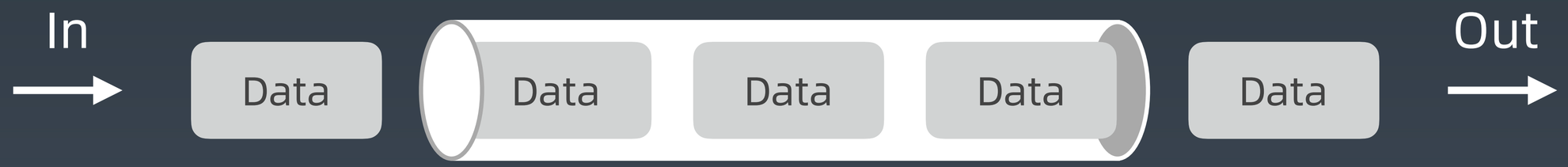
第一节

栈和队列的基本实现和特性

Stack


Data Element
Data Element
Push
Last in - First out
Data Element
Data Element
Data Element
Data Element
Data Element
Data Element
Data Element
Data Element
Data Element
Pop
Stack
Stack

Queue



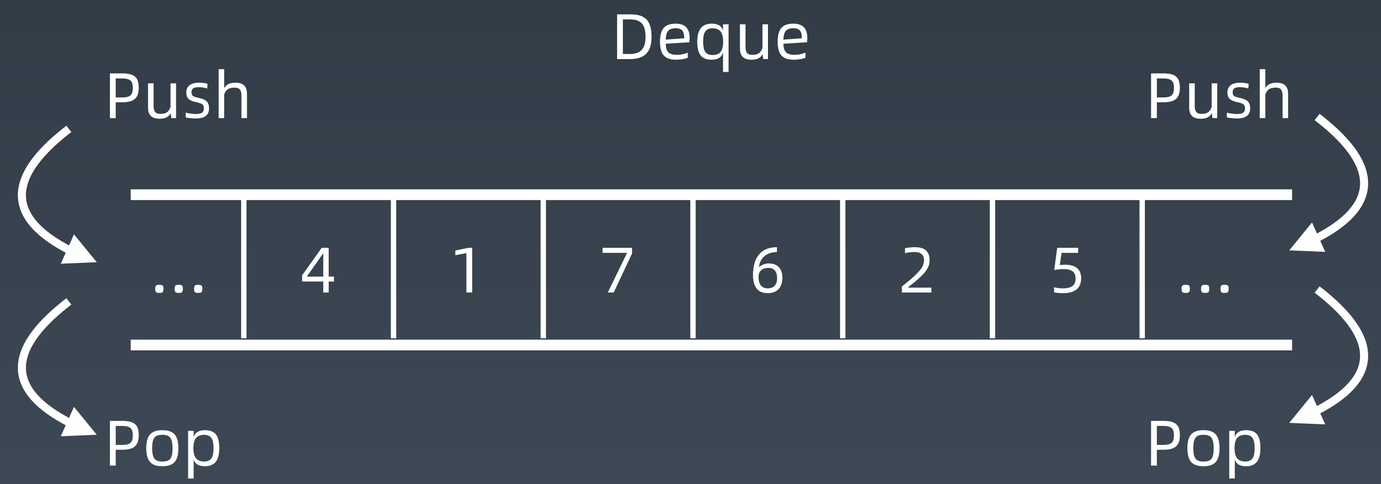
Last in - Last out

First in - First out

Stack & Queue 关键点

- Stack：先入后出；添加、删除皆为 O(1)
- Queue：先入先出；添加、删除皆为 O(1)

Deque: Double-End Queue



Deque

1. 简单理解：两端可以进出的 Queue
   Deque - double ended queue

2. 插入和删除都是 O(1) 操作

Stack、Queue、Deque 的工程实现

- Java、Python、C++ 等已有基础实现
- 如何查询接口信息？如何使用？（Demo）
- 示例代码

示例代码 - Stack

```
Stack<Integer> stack = new Stack<>();
stack.push(1);
stack.push(2);
stack.push(3);
stack.push(4);
System.out.println(stack);
System.out.println(stack.search(4));

stack.pop();
stack.pop();
Integer topElement = stack.peek();
System.out.println(topElement);
System.out.println(" 3的位置 " + stack.search(3));
```

示例代码 - Queue

```
Queue<String> queue = new LinkedList<String>();
queue.offer("one");
queue.offer("two");
queue.offer("three");
queue.offer("four");
System.out.println(queue);

String polledElement = queue.poll();
System.out.println(polledElement);
System.out.println(queue);

String peekedElement = queue.peek();
System.out.println(peekedElement);
System.out.println(queue);

while(queue.size() > 0) {
  System.out.println(queue.poll());
}
```

示例代码 - Deque

```
Deque<String> deque = new LinkedList<String>();

deque.push("a");
deque.push("b");
deque.push("c");
System.out.println(deque);

String str = deque.peek();
System.out.println(str);
System.out.println(deque);

while (deque.size() > 0) {
  System.out.println(deque.pop());
}
System.out.println(deque);
```

Priority Queue

1. 插入操作：O(1)

2. 取出操作：O(logN) - 按照元素的优先级取出

3. 底层具体实现的数据结构较为多样和复杂：heap、bst、treap

Java 的 PriorityQueue
https://docs.oracle.com/javase/10/docs/api/java/util/PriorityQueue.html

Stack 和 Queue 的实现

Java 源码分析:

Stack: http://developer.classpath.org/doc/java/util/Stack-source.html

Queue: http://fuseyism.com/classpath/doc/java/util/Queue-source.html

Priority Queue: 学员自己分析source code!

Python

```python
class Stack:
    def __init__(self):
        self.items = ['x', 'y']

    def push(self, item):
        self.items.append(item)

    def pop(self):
        self.items.pop()

    def length(self):
        return len(self.items)
```

```python
class Queue:

    def __init__(self):
        self.queue = []

    def enqueue(self, item):
        self.queue.append(item)

    def dequeue(self):
        if len(self.queue) < 1:
            return None
        return self.queue.pop(0)

    def size(self):
        return len(self.queue)
```

Python

1. heapq : https://docs.python.org/2/library/heapq.html

2. 高性能的 container 库：
   https://docs.python.org/2/library/collections.html

复杂度分析

Common Data Structure Operations

Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
Array	Θ(1)	Θ(n)	Θ(n)	Θ(n)	O(1)	O(n)	O(n)	O(n)	O(n)
Stack	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Queue	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Singly-Linked List	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Doubly-Linked List	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Skip List	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n log(n))
Hash Table	N/A	Θ(1)	Θ(1)	Θ(1)	N/A	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)
Cartesian Tree	N/A	Θ(log(n))	Θ(log(n))	Θ(log(n))	N/A	O(n)	O(n)	O(n)	O(n)
B-Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Red-Black Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Splay Tree	N/A	Θ(log(n))	Θ(log(n))	Θ(log(n))	N/A	O(log(n))	O(log(n))	O(log(n))	O(n)
AVL Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
KD Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)

图片来源：http://www.bigocheatsheet.com/

小结

1. Stack、Queue、Deque 的原理和操作复杂度

2. PriorityQueue 的特点和操作复杂度

3. 查询 Stack、Queue、Deque、PriorityQueue 的系统接口的方法

第二节

实战题目解析

预习题目

1. https://leetcode-cn.com/problems/valid-parentheses/
   - 最近相关性 —> 栈！

2. https://leetcode-cn.com/problems/min-stack/

实战题目

1. https://leetcode-cn.com/problems/largest-rectangle-in-histogram

2. https://leetcode-cn.com/problems/sliding-window-maximum

Homework

1. https://leetcode.com/problems/design-circular-deque

2. https://leetcode.com/problems/trapping-rain-water/

THANKS!