1. Easy stacks and queues interview questions

You might be tempted to try to read all of the possible questions and memorize the solutions, but this is not feasible. Interviewers will always try to find new questions, or ones that are not available online. Instead, you should use these questions to practice the **fundamental concepts** of stacks and queues.

As you consider each question, try to replicate the conditions you'll encounter in your interview. Begin by writing your own solution without external resources in a fixed amount of time.

If you get stuck, go ahead and look at the solutions, but then try the next one alone again. Don't get stuck in a loop of reading as many solutions as possible! We've analysed dozens of questions and selected ones that are commonly asked and have clear and high quality answers.

Here are some of the easiest questions you might get asked in a coding interview. These questions are often asked during the "phone screen" stage, so you should be comfortable answering them without being able to write code or use a whiteboard.

1.1 Valid parenthesis

- Text guide (RedQuark)
- Video guide (NeetCode)
- Code example (LeetCode)

1.2 Min stack

- Text guide (After Academy)
- Video guide (NeetCode)
- Video guide (Nick White)
- Code example (LeetCode)

1.3 Implement stack using queues

- Text guide (LeetCode)
- Video guide (TECH DOSE)
- Code example (LeetCode)

1.4 Implement queue using stacks

- Text guide (LeetCode)
- Video guide (Back to Back SWE)
- Code example (LeetCode)

1.5 Next greater element I

- Text guide (Memogrocery)

- Video guide (Nick White)
- Video guide (TECH DOSE)
- Code example (LeetCode)

1.6 Backspace string compare

- Text guide (LeetCode)
- Video guide (Kevin Naughton Jr.)
- Code example (LeetCode)

1.7 Remove all adjacent duplicates in string

- Text guide (Techie Delight)
- Video guide (Nick White)
- Code example (LeetCode)

1.8 Build an array with stack operations

- Text guide (Medium/Fatboy Slim)
- Video guide (daose)
- Code example (LeetCode)

1.9 Final prices with a special discount in a shop

- Text guide (LeetCode)
- Video guide (Lead Coding by FRAZ)
- Code example (LeetCode)

1.10 Number of recent calls

- Text guide (LeetCode)
- Video guide (Nick White)
- Code example (LeetCode)

2. Medium stacks and queues interview questions

Here are some moderate-level questions that are often asked in a video call or onsite interview. You should be prepared to write code or sketch out the solutions on a whiteboard if asked.

2.1 Simplify path

- Text guide (LeetCode)
- Video guide (NeetCode)
- Code example (LeetCode)

2.2 Evaluate reverse polish notation

- Text guide (Dev.to/Rohith V)
- Video guide (Back to Back SWE)
- Code example (LeetCode)

2.3 Basic calculator II

- Text guide (Calvin Chan)
- Video guide (happygirlzt)
- Code example (LeetCode)

2.4 Remove duplicate letters

- Text guide (Medium/Jimmy Shen)
- Video guide (happygirlzt)
- Code example (LeetCode)

2.5 Flatten nested list iterator

- Text guide (Dev.to/Seanpgallivan)
- Video guide (Timothy H Chang)
- Code example (LeetCode)

2.6 Decode string

- Text guide (Medium/Signal Cat)
- Video guide (NeetCode)
- Code example (LeetCode)

2.7 Remove K digits

- Text guide (Dev.to/Aroup)
- Video guide (Nick White)
- Video guide (TECH DOSE)
- Code example (LeetCode)

2.8 Add two numbers II

- Text guide (Web Rewrite)
- Video guide (Code and Coffee)
- Code example (LeetCode)

2.9 Generate parentheses

- Text guide (Red Quark)

- Video guide (NeetCode)
- Code example (LeetCode)

2.10 132 pattern

- Text guide (Toronto edu)
- Video guide (Algopedia)
- Code example (LeetCode)

2.11 Next greater element II

- Text guide (GeeksForGeeks)
- Video guide (Nick White)
- Code example (LeetCode)

2.12 Score of parentheses

- Text guide (Dev.to/Seanpgallivan)
- Text guide (LeetCode)
- Video guide (Nick White)
- Code example (LeetCode)

2.13 Exclusive time of functions

- Text guide (Nerohoop)
- Video guide (happygirlzt)
- Code example (LeetCode)

2.14 Asteroid collision

- Text guide (Petamind)
- Text guide (LeetCode)
- Video guide (Kevin Naughton Jr.)
- Code example (LeetCode)

2.15 Daily temperatures

- Text guide (Medium/Greynut)
- Video guide (Alexander Le)
- Video guide (NeetCode)
- Code example (LeetCode)

2.16 Online stock span

- Text guide (Medium/Tanu)
- Video guide (Timothy H Chang)

- Video guide (TECH DOSE)
- Code example (LeetCode)

2.17 Sum of subarray minimums

- Text guide (Zhenyu0519)
- Video guide (Lead Coding by FRAZ)
- Code example (LeetCode)

2.18 Validate stack sequences

- Text guide (After Academy)
- Video guide (Michael Muinos)
- Code example (LeetCode)

2.19 Maximum width ramp

- Text guide (Medium/Yashasvi)
- Code example (LeetCode)

2.20 Next greater node in linked list

- Text guide (Medium/Jiang)
- Video guide (Nick White)
- Code example (LeetCode)

2.21 Smallest subsequence of distinct characters

- Text guide (Youmin)
- Video guide (Naresh Gupta)
- Code example (LeetCode)

2.22 Minimum cost tree from leaf values

- Text guide (Developpaper)
- Video guide (AceCoder Academy)
- Code example (LeetCode)

2.23 Reverse substrings between each pair of parentheses

- Text guide (GeeksForGeeks)
- Code example (LeetCode)

2.24 Remove all adjacent duplicates in string II

- Text guide (Dev.to/Seanpgallivan)

- Video guide (babybear4812)
- Code example (LeetCode)

2.25 Design circular queue

- Text guide (Dev.to/Seanpgallivan)
- Video guide (Timothy H Chang)
- Code example (LeetCode)

2.26 Reveal cards in increasing order

- Text guide (LeetCode)
- Code example (LeetCode)

2.27 Design front middle back queue

- Text guide (LeetCode)
- Video guide (Lead Coding by FRAZ)
- Code example (LeetCode)

3. Hard stacks and queues interview questions

Similar to the medium section, these more difficult questions may be asked in an onsite or video call interview. You will likely be given more time if you are expected to create a full solution.

3.1 Longest valid parentheses

- Text guide (Dev.to/Seanpgallivan)
- Video guide (Time Complexity Infinity)
- Code example (LeetCode)

3.2 Trapping rain water

- Text guide (After Academy)
- Text guide (LeetCode)
- Code example (LeetCode)

3.3 Largest rectangle in histogram

- Text guide (GeeksForGeeks)
- Video guide (TECH DOSE)
- Code example (LeetCode)

3.4 Maximal rectangle

- Text guide (Opengenus)

- Video guide (TECH DOSE)
- Code example (LeetCode)

3.5 Basic calculator

- Text guide (Yucoding)
- Video guide (Jessica Lin)
- Code example (LeetCode)

3.6 Maximum frequency stack

- Text guide (Dev.to/Seanpgallivan)
- Video guide (Algorithms Made Easy)
- Code example (LeetCode)

3.7 Odd even jump

- Text guide (Zirui)
- Code example (LeetCode)

3.8 Constrained subsequence sum

- Text guide (Medium/Sudip)
- Video guide (Happy Coding)
- Code example (LeetCode)

3.9 Sliding window maximum

- Text guide (After Academy)
- Video guide (NeetCode)
- Video guide (Jessica Lin)
- Code example (LeetCode)

3.10 Find the most competitive subsequence

- Text guide (Dev.to/Seanpgallivan)
- Video guide (Lead Coding by FRAZ)
- Code example (LeetCode)

3.11 Number of visible people in a queue

- Text guide (LeetCode)
- Video guide (Happy Coding)
- Code example (LeetCode)

3.12 Shortest subarray with sum at least K

- Text guide (Zirui)
- Video guide (TechLife With Shivank)
- Code example (LeetCode)

3.13 Stamping the sequence

- Text guide (LeetCode)
- Video guide (Algorithms Made Easy)
- Code example (LeetCode)

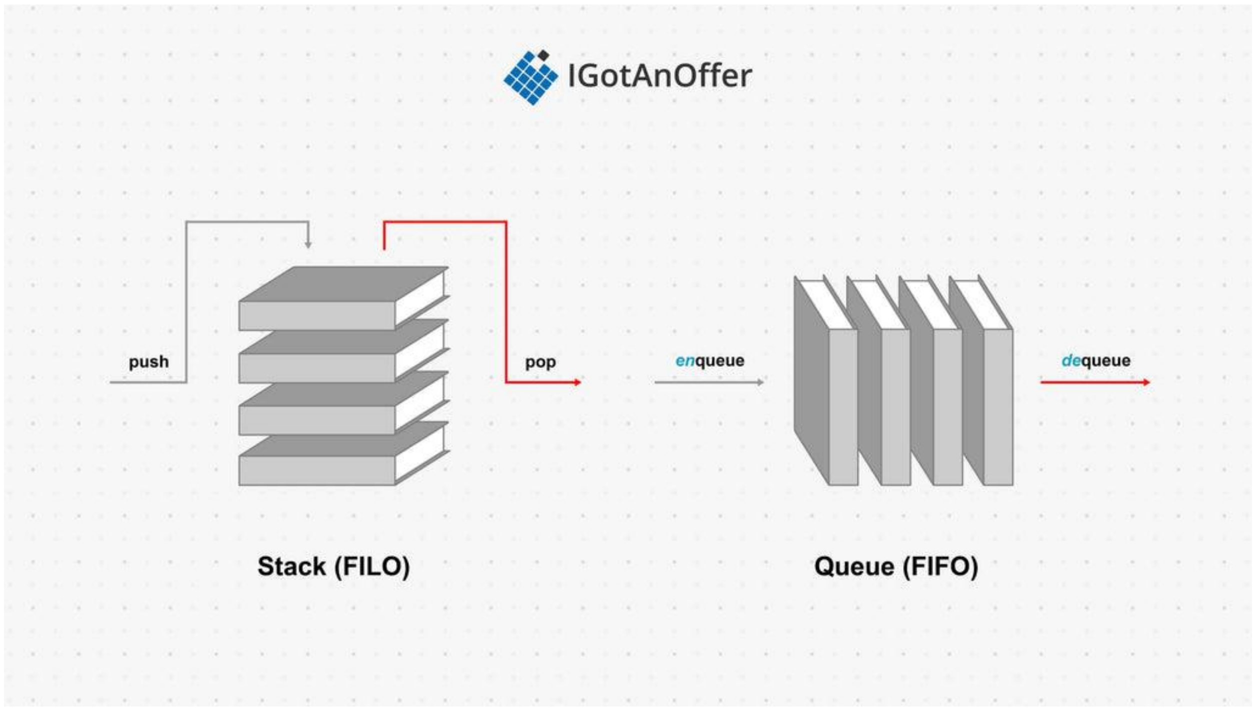
4. Stacks and queues basics

In order to crack the questions above and others like them, you’ll need to have a strong understanding of stacks and queues and how they work. Let’s get into it.

4.1 What are stacks and queues?

Stacks and queues are similar and complementary in many ways. Both are a sequenced collection of elements that are generally accessed one element at time, and are implemented using similar data structures.

Stacks are Last in First Out (LIFO) constructs. Queues are the opposite, being First In First Out (FIFO) constructs. These characteristics are what give these structures their name. A stack is analogous to a stack of physical objects, for example a stack of plates or chairs. The last plate or chair added to the stack is usually the first one removed when an item is needed. Similarly for a queue, the first person to enter a queue is usually the first person served.



There are two main operations on stacks and queues. These are adding a new element, and accessing the next logical element.

For a stack, these two operations are known as “push” and “pop.” “Push” adds a new element to the top of the stack. “Pop” returns the last element added to the stack, and removes it from the stack.

A queue’s two operations are known as “enqueue” and “dequeue”. “Enqueue” adds a new element to the end of the queue. “Dequeue” returns the oldest element in the queue, and removes it from the queue.

Stacks and queues often have a third operation defined called “peek”. This operation returns the next logical element, but does not remove it from the stack or queue.

Although stacks and queues have no fixed capacity in theory, in practice, resource bounds make it so that that they do. Adding an element to a stack when it has no more space is known as a “stack overflow” error. Trying to pop an element from an empty stack is called a “stack underflow” error. Similarly for queues, these errors are known as “queue overflow” and “queue underflow”.

4.1.1 Types of stacks and queues (Java, Python, C++)

In Java, a Stack class is available, which extends the Vector class. C++ has many containers in the STL that expose push and pop operations. It also has Stack and Queue templates which wrap underlying vectors or lists to provide stricter stack and queue interfaces. Python has a deque class, or double ended queue, which allows pushing and popping from the front or the back, allowing it to be used as a stack or a queue.

4.1.2 How stacks and queues store data

Stacks and queues are defined by their interface and functionality, not by their implementations. Stacks and queues are often implemented using arrays or, more commonly, linked lists.

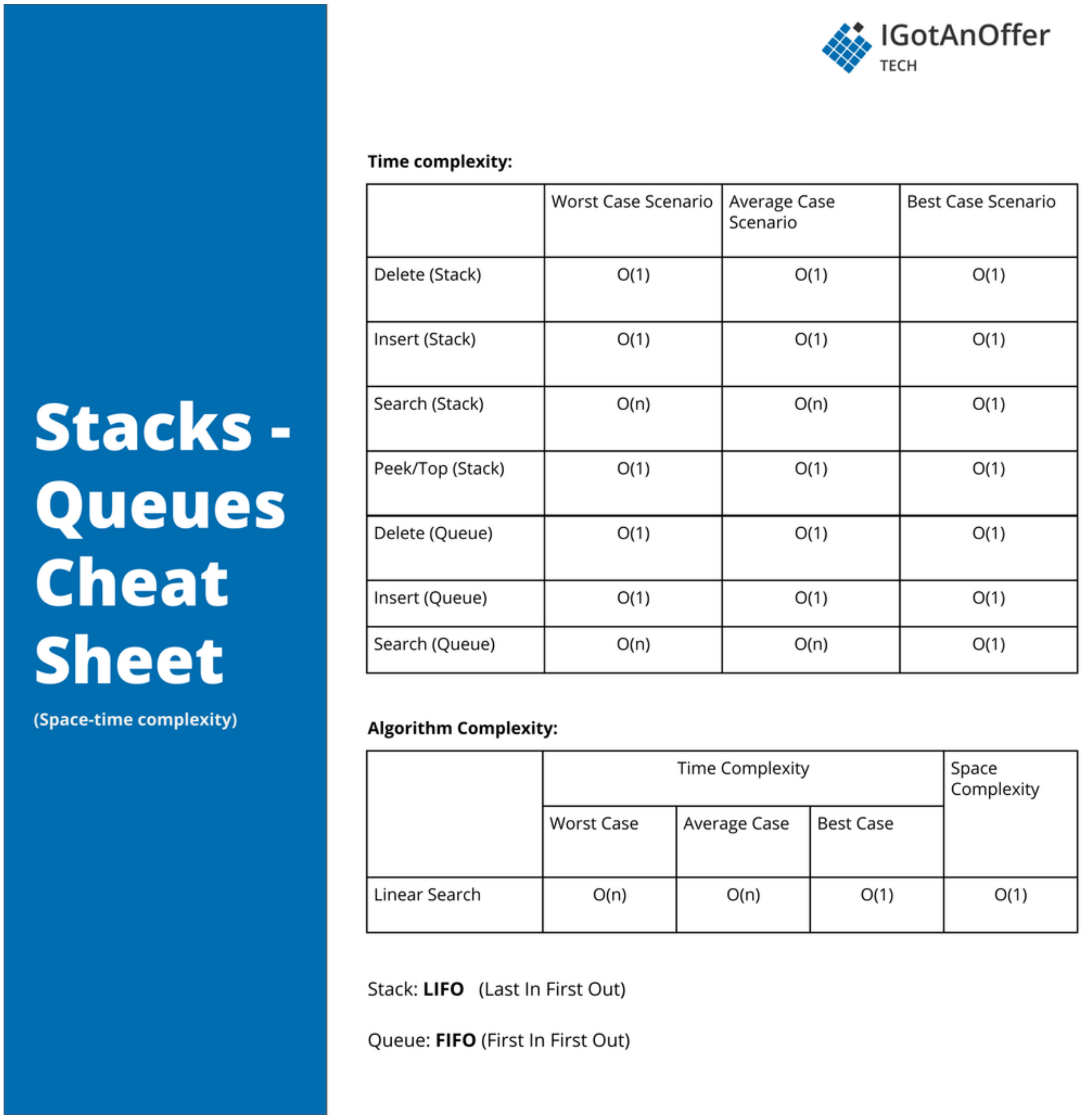
A straightforward implementation of a stack can be done with an array. Elements are pushed by adding them to the first unoccupied position of the array. The pop operation is accomplished by returning and removing the element at the last occupied position of the array. A variable is maintained indicating the last occupied position in the array, so that elements can be pushed and popped at the correct position.

Linked list implementations for stacks and queues can also be straightforward. For a stack, elements are pushed and popped from the tail of the linked list. For a queue, elements are enqueued at the tail, and dequeued from the head. For both implementations, maintaining a variable pointing to the head and the tail of the linked list makes the standard stack and queue operations possible in constant time.

4.1.3 How stacks and queues compare to other data structures

Stacks and queues are often mentioned with arrays and linked lists, as these are commonly used to implement them. Deques, or double ended queues, are also a variant that can be used as both a stack and a queue.

5. Stacks and queues cheat sheet



IGotAnOffer TECH

Stacks - Queues Cheat Sheet

(Space-time complexity)

Time complexity:

	Worst Case Scenario	Average Case Scenario	Best Case Scenario
Delete (Stack)	O(1)	O(1)	O(1)
Insert (Stack)	O(1)	O(1)	O(1)
Search (Stack)	O(n)	O(n)	O(1)
Peek/Top (Stack)	O(1)	O(1)	O(1)
Delete (Queue)	O(1)	O(1)	O(1)
Insert (Queue)	O(1)	O(1)	O(1)
Search (Queue)	O(n)	O(n)	O(1)

Algorithm Complexity:

	Time Complexity			Space Complexity
	Worst Case	Average Case	Best Case	
Linear Search	O(n)	O(n)	O(1)	O(1)

Stack: **LIFO** (Last In First Out)

Queue: **FIFO** (First In First Out)

You can download the cheat sheet here.