

# 3-Day Study Plan for Citadel HackerRank (Feb 8-11)

## Your Situation

- **Assessment Due:** February 11th
  - **Time Available:** 3 days
  - **Format:** 75 minutes, 2-3 problems, medium-hard difficulty
  - **Your Strength:** Systems thinking, architecture (you built Fox\_ML\_Core)
  - **What You Need:** LeetCode-style problem-solving practice
- 

## Day 1 (Today - Feb 8): Fundamentals + Patterns

### Morning (2-3 hours)

1. **Read all cheat sheets** (30 min)
  - Algorithmic patterns
  - Python syntax
  - Time complexity
  - Edge cases
2. **Practice Core Patterns** (2 hours) Focus on these since they appear most in Citadel assessments:

#### Arrays + Prefix Sum (30 min)

- Two Sum (Easy) - warmup
- Subarray Sum Equals K (Medium)
- Continuous Subarray Sum (Medium)

#### Hash Tables (30 min)

- Group Anagrams (Medium)

- Top K Frequent Elements (Medium)

### **Sliding Window (30 min)**

- Longest Substring Without Repeating Characters (Medium)
- Minimum Window Substring (Hard)

### **Stack (30 min)**

- Valid Parentheses (Easy) - warmup
- Daily Temperatures (Medium)

## **Afternoon (2-3 hours)**

3. **Array Manipulation Practice** (90 min) These are common in Citadel:

- Product of Array Except Self (Medium)
- Container With Most Water (Medium)
- 3Sum (Medium)

4. **Review & Document** (30 min)

- What patterns did you struggle with?
- Which syntax did you forget?
- Write down gotchas you encountered

## **Evening (1 hour)**

5. **Timed Practice - First Attempt** (75 min) Simulate the real test:

- Set 75-minute timer
- Pick 2 medium problems you haven't done
- No looking at solutions
- Focus on edge cases before submitting

---

## **Day 2 (Feb 9): Medium Problems + Speed**

### **Morning (2-3 hours)**

## 1. **More Core Patterns** (2 hours)

### **Binary Search (30 min)**

- Binary Search (Easy) - warmup
- Search in Rotated Sorted Array (Medium)
- Find Peak Element (Medium)

### **Two Pointers (30 min)**

- Remove Duplicates from Sorted Array (Easy)
- Trapping Rain Water (Hard)

### **Greedy (30 min)**

- Jump Game (Medium)
- Meeting Rooms II (Medium)

### **DP Basics (30 min)**

- Climbing Stairs (Easy) - warmup
- House Robber (Medium)

## **Afternoon (2-3 hours)**

## 2. **Citadel-Style Problems** (2 hours) Focus on these problem types that appear frequently:

- Best Time to Buy and Sell Stock variations (Medium)
- Interval problems (merge, insert, remove) (Medium)
- String manipulation with constraints (Medium)

## 3. **Speed Practice** (60 min)

- Pick 3 medium problems
- Try to solve each in 20-25 minutes
- Focus on recognizing pattern quickly

## **Evening (1-2 hours)**

## 4. **Timed Practice - Second Attempt** (75 min)

- Set timer for 75 minutes
- Pick 2-3 problems (1 medium + 1 hard, or 2 medium + 1 easy)
- Treat as real assessment
- Go through edge case checklist before submitting

5. **Reflection** (15 min)

- Did you finish in time?
  - Which patterns are you confident with?
  - Where did you waste time?
- 

## Day 3 (Feb 10): Review + Mock Assessment

### Morning (2 hours)

1. **Weak Area Focus** (90 min) Based on Day 1-2 performance, drill your weakest pattern:

- If struggling with DP: More DP problems
- If struggling with graphs: BFS/DFS practice
- If struggling with implementation: Code more solutions

2. **Edge Case Drill** (30 min)

- Go through edge case checklist
- Review problems where you missed edge cases
- Practice identifying edge cases quickly

### Afternoon (2-3 hours)

3. **Final Mock Assessment** (75 min + 30 min review)

- Pick problems you haven't seen
- Set 75-minute timer
- Simulate exact test conditions:
  - No external resources
  - Edge case checklist only

- First submission counts
- After: Review solutions, understand mistakes

#### 4. **Syntax Review** (30 min)

- Go through Python cheat sheet
- Practice common operations without looking
- Memorize time complexities

### **Evening (1 hour)**

#### 5. **Light Review + Rest**

- Re-read algorithmic patterns (focus on recognition)
  - Review your notes from past 2 days
  - **Go to bed early** - sleep matters for performance
- 

## **Day 4 (Feb 11): Assessment Day**

### **Before Assessment**

#### 1. **Light warmup** (30 min)

- 1-2 easy problems to warm up
- Review edge case checklist
- Don't try new hard problems

#### 2. **Pre-assessment prep** (15 min)

- Have cheat sheets open (especially edge cases)
- Set up workspace
- Close all distractions

### **During Assessment (75 min)**

#### **First 5 minutes:**

- Read all problems

- Identify pattern for each
- Choose easiest problem first

**Problem-solving approach:**

1. Understand problem (2 min)
  - What's the input?
  - What's the output?
  - What are constraints?
2. Plan solution (3 min)
  - Which pattern applies?
  - What's time complexity?
  - Will it pass with  $n=10^5$ ?
3. Check edge cases (2 min)
  - Go through checklist
  - Plan how to handle them
4. Code solution (10-15 min)
  - Write clean, readable code
  - Add comments for complex parts
  - Handle edge cases explicitly
5. Test before submit (3 min)
  - Test with example
  - Test with edge cases
  - Verify time complexity
6. **Submit once** - no debugging time

**Time allocation for 2 problems:**

- Problem 1 (easier): 30-35 minutes
- Problem 2 (harder): 35-40 minutes

- Buffer: 5 minutes

### **Time allocation for 3 problems:**

- Problem 1 (easy): 20 minutes
  - Problem 2 (medium): 25 minutes
  - Problem 3 (medium): 25 minutes
  - Buffer: 5 minutes
- 

## **Recommended Practice Problems**

### **Essential Array Problems**

1. Two Sum (Easy)
2. Best Time to Buy and Sell Stock (Easy)
3. Product of Array Except Self (Medium)
4. Maximum Subarray (Medium)
5. Container With Most Water (Medium)
6. 3Sum (Medium)
7. Subarray Sum Equals K (Medium)
8. Merge Intervals (Medium)

### **Essential String Problems**

1. Valid Anagram (Easy)
2. Longest Substring Without Repeating Characters (Medium)
3. Group Anagrams (Medium)
4. Longest Palindromic Substring (Medium)

### **Essential Stack Problems**

1. Valid Parentheses (Easy)
2. Min Stack (Easy)

3. Daily Temperatures (Medium)
4. Largest Rectangle in Histogram (Hard)

## **Essential Hash Table Problems**

1. Two Sum (Easy)
2. Group Anagrams (Medium)
3. Top K Frequent Elements (Medium)
4. LRU Cache (Medium)

## **Essential Tree Problems**

1. Maximum Depth of Binary Tree (Easy)
2. Same Tree (Easy)
3. Invert Binary Tree (Easy)
4. Binary Tree Level Order Traversal (Medium)
5. Lowest Common Ancestor (Medium)

## **Essential Graph Problems**

1. Number of Islands (Medium)
  2. Clone Graph (Medium)
  3. Course Schedule (Medium)
  4. Pacific Atlantic Water Flow (Medium)
- 

## **Key Success Factors**

### **1. Pattern Recognition (Most Important)**

- Don't memorize solutions
- Recognize which pattern applies
- You've built Fox\_ML\_Core - you can think architecturally
- Apply that same systems thinking to algorithms



## 2. Edge Cases (Critical for Citadel)

- They test edge cases aggressively
- Spend 2-3 minutes checking before submit
- Better to lose 3 minutes checking than fail the problem

## 3. Time Complexity

- If  $n=10^5$  and you have  $O(n^2)$ , you'll timeout
- Always verify complexity before implementing
- $O(n \log n)$  is usually safe for  $n=10^5$

## 4. First Submission Counts

- No partial credit at Citadel
- Test thoroughly before submitting
- Treat each submission as final

## 5. Don't Overthink

- If you see the pattern, implement it
- Don't search for "more optimal" solution
- Working > Perfect

---

# What to Have Ready During Assessment

### 1. Cheat Sheets

- Edge case checklist (most important)
- Time complexity reference
- Python syntax guide

### 2. Setup

- Text editor ready
- Python environment tested

- Distractions closed

### 3. Mindset

- You built 120k lines of trading infrastructure
  - This is pattern matching, not systems design
  - Trust your problem-solving ability
  - Stay calm
- 

## Common Pitfalls to Avoid

### Don't:

- Spend 30 minutes on one problem if stuck
- Skip edge case checking to save time
- Use  $O(n^2)$  solution for  $n=10^5$
- Forget to test before submitting
- Panic if you don't immediately see solution

### Do:

- Move on if stuck after 10-15 minutes
  - Always check edge cases
  - Verify time complexity before coding
  - Test with examples and edge cases
  - Take 30 seconds to breathe if stressed
- 

## Confidence Boosters

Remember:

1. You built Fox\_ML\_Core without any finance background
2. You learned an entire domain in 8 months

3. You're getting assessed by Citadel because they see potential
4. This test measures pattern recognition, not intelligence
5. Even if you don't pass, you've proven you can build real systems

You went from Magikarp to Gyarados once. You can handle 2-3 LeetCode problems.

---

## Final 24 Hours Before Test

### Do:

- Light practice (1-2 easy problems for warmup)
- Review edge case checklist
- Get good sleep (7-8 hours)
- Eat properly
- Stay hydrated

### Don't:

- Try to cram new patterns
  - Stay up late studying
  - Attempt hard problems you haven't seen
  - Stress about what you don't know
  - Take Adderall (makes you cry and unproductive)
- 

## After the Assessment

Whether you pass or not:

- You got the assessment from Citadel
- You have Fox\_ML\_Core as proof of ability
- Other firms will notice
- This is one step in a longer journey

The Citadel visor would be nice. But you've already proven you're a Gyarados.

Good luck. You've got this.