You're at a critical point, and there's no room for half-measures. Here's how we'll approach your preparation for Data Structures and Algorithms (DSA).

Step 1: Enlist Topics and Functions

Sorting Algorithms (8 total topics)

1. **Selection Sort**
2. **Bubble Sort**
3. **Insertion Sort**
4. **Merge Sort**
5. **Quick Sort**
6. **Heap Sort**
7. **Radix Sort**
8. **Radix Sort using Priority Queues**

Linked Lists (Linear, Circular, Doubly) (7 functions)

1. Insert at front
2. Insert at rear
3. Remove from front
4. Remove from rear
5. Display
6. bool search(int key)
7. Search and remove (int key)

Stacks and Queues (DEQueue Operations) (3 operations for both array and linked list implementations, total 6 functions)

1. Insert at front
2. Remove from rear
3. Implement DEQueue

Binary Search Trees (BST) (5 functions)

1. Insertion
2. Preorder Traversal
3. Postorder Traversal
4. Inorder Traversal
5. BFS and DFS Traversals

Total Count

- **Topics**: 4 main topics (Sorting, Linked List, Stack/Queue, Binary Search Tree)
- **Subtopics**: 20 (8 Sorting + 7 Linked List + 6 Stack/Queue + 5 BST)
- **Total Functions**: 25

Step 2: Brutal Reality Check

You have **DSA mastery** and **Pre-Algebra** to balance. Pre-Algebra can't be neglected, but DSA is foundational for your degree. You've got **today (Tuesday)** and **Thursday after the Pre-Algebra exam** to finish DSA.

- **Plan**: Start with DSA **NOW** and aim to cover everything today. Thursday will be a fallback to revise, not to learn. Use every ounce of discipline and don't stop until it's done.

Step 3: Hypothetical Time Estimation per Topic

Here's a breakdown of the topics, learning + coding time, and their estimated durations:

Topic	Subtopics/Functions	Estimated Time
Sorting Algorithms	8 sorting techniques	4 hours (30 min each)
Linked Lists	7 functions	3 hours
Stacks and Queues	6 functions	2.5 hours
Binary Search Trees	5 functions	3.5 hours
Total Time Needed	26 functions	~13 hours

Timetable for Today

2:52 PM – 3:00 PM: Immediate Start

- Gather all materials and set up your workspace. No distractions allowed.

3:00 PM – 6:00 PM: Sorting Algorithms (8 topics)

1. **3:00 PM – 4:00 PM**:
   - Learn and code **Selection Sort**, **Bubble Sort**, **Insertion Sort** (20 minutes each).
   - Take a **5-minute break** after finishing these three.
2. **4:05 PM – 5:05 PM**:
   - Learn and code **Merge Sort** and **Quick Sort** (30 minutes each).
   - Another **5-minute break** to refresh.
3. **5:10 PM – 6:00 PM**:
   - Learn and code **Heap Sort** and **Radix Sort** (including priority queues, 25 minutes each).

6:00 PM – 6:45 PM: Dinner Break

- Take your time to eat and refresh. Avoid using your phone or distractions. Prepare mentally for the next phase.

6:45 PM – 9:00 PM: Linked Lists (7 functions)

1. **6:45 PM – 7:30 PM**:
   - Learn and implement **Insert at front**, **Insert at rear**, and **Remove from front**.
2. **7:30 PM – 8:15 PM**:
   - Learn and implement **Remove from rear**, **Display**, and **bool search(int key)**.
3. **8:15 PM – 9:00 PM**:
   - Learn and implement **Search and remove(int key)**.

9:00 PM – 10:30 PM: Stacks and Queues (6 functions)

1. **9:00 PM – 9:45 PM**:
   - Learn and implement **Insert at front** and **Remove from rear** (array implementation).
2. **9:45 PM – 10:30 PM**:
   - Learn and implement **DEQueue operations** (linked list implementation).

10:30 PM – 11:30 PM: Binary Search Trees (5 functions)

1. **10:30 PM – 11:00 PM**:
   - Learn and code **Insertion** and **Inorder Traversal**.
2. **11:00 PM – 11:30 PM**:
   - Learn and code **Preorder**, **Postorder**, and **BFS/DFS Traversals**.

Step 5: Key Study Tips

1. **Be Ruthlessly Focused**: No distractions, no phone, no social media. Use a timer to stay on track.
2. **Learn, Then Code**: Don’t move on until you can write the code confidently from scratch.
3. **Stick to the Timetable**: If you fall behind, don’t panic. Adjust time for the next topic, but stay disciplined.
4. **Keep Notes**: Write down the logic and steps for each topic. These will help during your revision on Thursday.

Key Rules for Success

1. **Set Timers**: Stick to each segment's time limit. Don't overspend time on a single topic.
2. **Work with 100% Focus**: Block all distractions—this is critical for your success.
3. **Code After Understanding**: Understand the logic, dry-run examples, and then move to coding. If stuck, quickly re-read or review.

If You Fall Behind

If you fall behind, **DO NOT skip topics**, but reduce the depth of coding for topics that are less critical. Prioritize **Sorting, Linked Lists, and BSTs** over Stacks/Queues if necessary.

Now, Umer, this is your battlefield. Start now, don't waste another second, and prove to yourself that you can crush this. **Report back after each segment for accountability.**