Notes on Data Structures

1. What is a Data Structure?

A **data structure** organizes data for efficient use in computer programs.
Combines primitive data types (numbers, characters) into structured formats.
Enables operations like **sorting, searching, insertion, and deletion**.

Example:

```
daily_sales = [500, 800, 600, 1200, 950]  # Array data structure
```

2. Importance of Data Structures

Efficiency: Optimizes time & space complexity (measured using **Big O notation**).
Dynamic Programming: Stores sub-solutions for faster problem-solving.
DSA (Data Structures & Algorithms): Key for optimizing code performance.

3. Linear vs. Nonlinear Data Structures

Linear	Nonlinear
Sequential arrangement (e.g., arrays, linked lists).	Hierarchical/networked (e.g., trees, graphs).
Easy traversal (single pass).	Complex traversal (multiple paths).

4. Common Data Structures

Arrays

Stores similar data types in adjacent memory.
Use Case: Sorting, searching, storing data.
Example: `average_customer_score = [4, 3.5, 3.7, 4.1]`

Queues (FIFO)

First In, First Out (e.g., call center waitlist).
Use Case: Task scheduling, printer queues.

Stacks (LIFO)

Last In, First Out (e.g., undo operations).
Use Case: Browser history, backtracking.

Linked Lists

Nodes linked sequentially (easy insertion/deletion).
Use Case: Playlists, browser history.

Trees

Hierarchical (parent-child nodes).
Types: Binary Search Trees (BST), AVL Trees.
Use Case: File systems, organizational charts.

Graphs

Vertices (nodes) + Edges (connections).
Use Case: Social networks, GPS navigation.

Hash Tables

Uses **hash function** for quick key-value lookup.

Use Case: Databases, password storage.

5. Use Cases for Data Structures

Data Storage: Efficiently organize large datasets.
Indexing: Fast retrieval (e.g., hash tables).
Searching: Graphs for social media connections.
Scalability: Trees & hash tables for big data.

6. Key Takeaways

✓ **Arrays, Queues, Stacks** → Linear structures.

✓ **Trees, Graphs** → Nonlinear structures.

✓ **Hash Tables** → Fast key-value access.

✓ **DSA (Data Structures + Algorithms)** → Optimizes performance.

🚀 **Next Steps:** Learn **Big O notation** to analyze efficiency!

FAQ

Q: What's the difference between a stack and a queue?

Stack: LIFO (Last In, First Out).
Queue: FIFO (First In, First Out).

Q: Why use a hash table?

For **O(1) average-time lookups** (faster than arrays for large data).

Q: When to use a tree structure?

For **hierarchical data** (e.g., file systems, ML decision trees).

Source: IBM Think Article on Data Structures