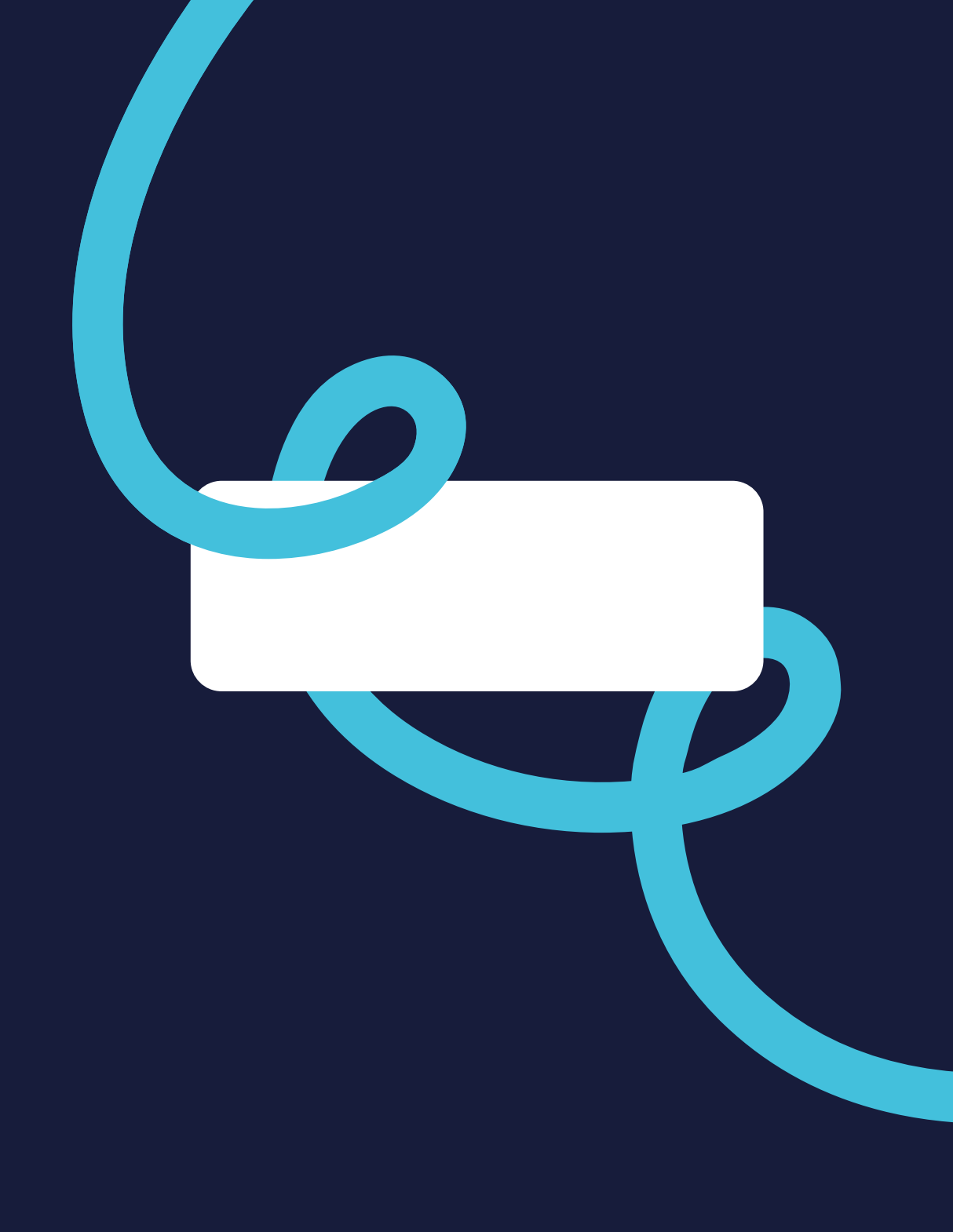


04/17/24

↔ Interview Prep

↳ DSA

↦ Here, we will go through a basic overview of all important data structures and algorithms:

1. **Arrays:** A collection of items stored at contiguous memory location.
   ↳ Can be one-dimensional or multi-dimensional.
   ↳ Common Operations: Access O(1)
   Insertion O(n)
   Deletion O(n)
   Search O(n)

2. **Linked lists:** A linear data structure where each element is a separate object called a node.
   ↳ Each node contains a data field and a reference to the next node in sequence.
   ↳ Common Operations: Access O(n)
   Insertion O(1)
   Deletion O(1)
   Search O(n)

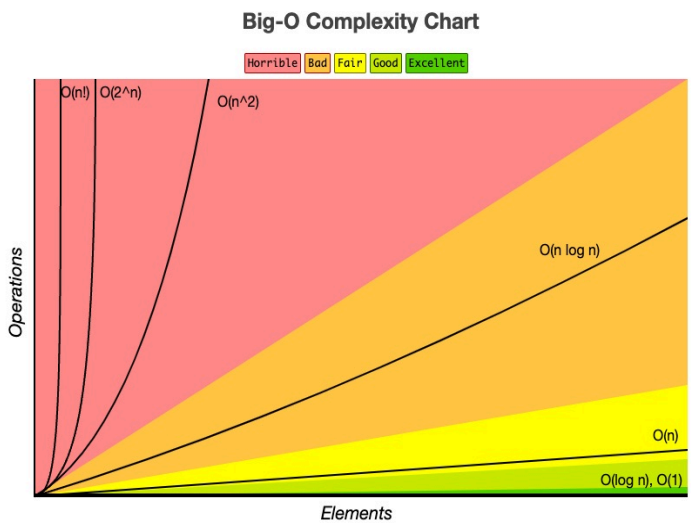
3. **Stacks:** A collection of elements that follows the Last In First Out (LIFO) principle.
   ↳ Common Operations: Push O(1)
   Pop O(1)
   Peek O(1)

4. Queues: A collection of elements that follows the First In First Out (FIFO) principle.
↳ Common Operations: Enqueue O(1)
Dequeue O(1)
front O(1)

5. Hash Tables: A data structure that implements an associative array abstract data type, a structure that can map keys to values.
Common Operations: Access O(1)
Insertion O(1)
Deletion O(1)
Search O(1)

6. Trees: ↳ Binary trees: a hierarchical data structure in which each node has at most two children, referred to as left child and right child.
↳ Binary Search Tree (BST): A tree where each node has at most two children, and the left child's value is less than the parent, and the right child's value is greater.
↳ AVL tree: A self balancing BST
↳ Red-Black tree: Another type of a self balancing BST
↳ Common operations: Insertion
Deletion
Search

7. Graphs: A set of nodes (or vertices) connected by edges.
↳ can be directed or undirected.
Common Operations: depends on the representation (adjacency matrix or adjacency list)



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)

Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Quicksort	Ω(n log(n))	θ(n log(n))	O(n^2)	O(log(n))
Mergesort	Ω(n log(n))	θ(n log(n))	O(n log(n))	O(n)
Timsort	Ω(n)	θ(n log(n))	O(n log(n))	O(n)
Heapsort	Ω(n log(n))	θ(n log(n))	O(n log(n))	O(1)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)	O(1)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)	O(1)
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)	O(1)
Tree Sort	Ω(n log(n))	θ(n log(n))	O(n^2)	O(n)
Shell Sort	Ω(n log(n))	θ(n(log(n))^2)	O(n(log(n))^2)	O(1)
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)	O(n)
Radix Sort	Ω(nk)	θ(nk)	O(nk)	O(n+k)
Counting Sort	Ω(n+k)	θ(n+k)	O(n+k)	O(k)
Cubesort	Ω(n)	θ(n log(n))	O(n log(n))	O(n)

Common Algorithms:

① Sorting Algorithms:

↳ Quick sort: Divide and Conquer algo.
↳ Avg case: $O(n \log n)$
Worst Case: $O(n^2)$

↳ Merge sort: Also divide and conquer algo.
↳ Stable and $O(n \log n)$

↳ Heap sort: Based on binary heap data structure
↳ $O(n \log n)$

② Search Algorithms:

↳ Binary Search: On sorted arrays
↳ $O(\log n)$

↳ Depth-first Search (DFS) and Breadth-first Search (BFS):

↓ travels in a way to explore all possible nodes in one direction first and then move to next direction
↳ uses Stack

↓ travels in every direction possible together.
↳ Uses Queue

3. Dynamic Programming:

↳ Memoization and Tabulazation: Techniques to solve problems by breaking them down into simpler subproblems and storing results.

→ Memoization is a technique that speeds up programs by storing the results of expensive function calls to pure functions. When the same input occurs again, the cached result is returned.

(4) Greedy Algorithms: Used for optimization problems where local optimal choices are made with the hope of finding a global optimum.

(5) Graph Algorithms:

↳ Dijkstra's Algorithms: Shortest path from a single source in weighted graph.

↳ Bellman-Ford Algorithm: Single source shortest path in graphs with negative weights

↳ Floyd-Warshall: All pairs shortest path.

↳ Kruskal's and Prim's Algorithms: Minimum Spanning tree

⇒ Essential Database Topics:

1. Relational Databases: Databases structured to recognize relations among stored items of information.
   ↳ Ex: MySQL, PostgreSQL, SQLite
   ↳ Key Concepts: Tables, rows, columns, primary keys, foreign keys, joins (inner, left, right, full)

2. Non-Relational Databases: Databases that do not use a table model like relational databases.
   ↳ Ex: MongoDB, Cassandra, Redis
   ↳ Key Concepts: Document-oriented, key-value stores, wide-column stores, graph databases.

3. SQL (Structured Query Language): The standard language for relational database management and data manipulation.
   ↳ Key Concepts: Queries, updates, schema creation, and database management commands.

4. ACID properties: Set of properties of database transactions intended to guarantee validity even in the event of errors, power failures, etc.
   ↳ Key Concepts: Atomicity, Consistency, Isolation, Durability

5. Transactions: A unit of work performed within a database management system against a database, and treated in a coherent and reliable way independent of other transactions.
↳ Key Concept: Begin transaction, commit, rollback

6. Indexing: Optimization techniques that speeds up the retrieval of operations by efficiently locating and accessing the data without having to search every row in a database table each time a database table is accessed.
↳ Key Concept: Primary index, Secondary index, composite index

7. Database Design: The process of producing a detailed data model of a database. This data model contains all the needed logical and physical design choices and physical storage parameters.
↳ Key Concepts: ER diagrams, normalization, database schemas.

Python and Databases:

① 'sqlite3': useful for smaller projects not requiring a full database server.

② 'pymysql/psycopg2': libraries to connect to MySQL or PostgreSQL databases.

③ 'sqlalchemy': SQL toolkit and Object-Relational Mapping (ORM) system for Python. Provides full power and flexibility of SQL.

from sqlalchemy import connect_engine

④ 'pymongo' : Tool for interacting with MongoDB from Python.

*) Machine learning Terminology and Topics:

1. Basic Concepts:

- Machine learning (ML): The study and construction of algorithms that can learn from and make predictions or decisions based on data.

  - Supervised learning: Training a model on known input and output data, enabling it to predict future outputs.

  - Unsupervised learning: Learning patterns from untagged data without any given outputs.

  - Reinforcement learning: Algorithms learn to perform actions in an environment so as to maximize some notions of cumulative reward.

2. Model Types:

- Regression: Predicts continuous values
  ↳ Ex: Predicting house prices.
- Classification: Predicts discrete values, categorizing data into predefined classes (e.g. spam or not spam).
- Clustering: Groups a set of objects in such a way

that objects in the same group are more similar to each other than to those in other groups.

3. Evaluation Metrics:

- Accuracy, Precision, Recall, F1 score: Metrics to evaluate classification models.
- Mean Squared Error (MSE), Mean Absolute Error (MAE): Metrics to evaluate regression models.

4. Overfitting and Underfitting:

- Overfitting: When a model learns the detail and noise in the training data to the extent that it negative impacts the performance of the model on new data.

- Underfitting: When a model is too simple, both performance on the training data and the ability to generalize to new data are low.

Overfit: model too complex, memorizes training data too well.
↳ high variance
↳ produce accurate result for training set but not for test set.

Underfit: model too simple, can't accurately find patterns and relationships.
↳ high bias
↳ produce inaccurate results for both train and test set

5. Algorithms:

① Linear Regression: A regression algorithm that models the relationship b/w a dependent variable and one or more independent variables by fitting a linear equation to observed data.
↳ The coefficients of the equation are derived by minimizing the sum of squared difference b/w the observed and predicted values.
↳ Usage: Commonly used in predicting analytics to forecast continuous outcomes, such as house prices, stock prices, or temperature forecasts.

② Logistic Regression: Classification algorithm used to estimate discrete values (binary outcomes like 0/1, yes/no) based on given sets of independent variables.
↳ It models the probability that a given input point belong to a certain class.
↳ Usage: Widely used for binary classification tasks such as spam detection, disease diagnosis, and churn prediction.

③ Decision trees: a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label.
↳ The path from root to leaf represent classification rules.

↳ Usage: Useful in both classification and regression tasks.
↳ Common application include customer segmentation, business decision analysis, and machine fault diagnosis.

④ Random Forests: An ensemble learning technique that builds multiple decision trees and merges them together to get a more accurate and stable prediction.
↳ Random forests correct for decision trees' habit of overfitting to their training set.
↳ Usage: Employed in both classification and regression tasks, popular in ensemble methods for its performance and ease of use in predictive modeling.

⑤ Support Vector Machines (SVM):

↳ Powerful and versatile classification technique that works by finding the hyperplane that best divides a dataset into classes.
↳ Support Vector Machines are particularly good at classifying complex but small- or medium-sized datasets.

SVM

↳ Usage: Commonly used in face detection, handwriting recognition, image classification, bioinformatics (e.g. for cancer classification), and other complex classification tasks.

(6) Neural Networks: Composed of layers of interconnected nodes or neurons, neural networks are a subset of ML algorithms modeled loosely after the human brain.

↳ They are designed to recognize patterns from complex data and learn tasks by considering examples, generally without task-specific programming.

↳ Usage: Extremely versatile, neural networks are used in image and speech recognition, medical diagnosis, financial forecasting, and beyond, especially where the relationship b/w input and output is nonlinear.

(7) K-means Clustering: A type of unsupervised learning, which is used when you have unlabeled data.

↳ The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K.

↳ The algo. works iteratively to assign each data point to one of K groups based on the features provided.

↳ Usage: Used in customer profiling, market segmentation, computer vision, and inventory categorization to derive insights from large unlabeled datasets.

6. Techniques:

- Feature Engineering: Process of using domain knowledge to select, modify, or create new features from raw data.
- Model selection: Choosing a model from a class of different models.
- Cross-validation: Technique for assessing how the results of a statistical analysis will generalize to an independent data set.

7. Neural Networks and Deep learning:

↳ Layers: Building blocks of neural network architectures, including input, hidden, and output layers.

↳ Activation func: Func like Sigmoid, ReLU, and Tanh that help a neural network learn complex patterns.

↳ CNN: Highly effective in image processing and recognition.

RNN: Useful for processing sequential data eg: speech or text.

Large Language Models (LLMs)

1. Foundations

- **Transformers: Architectural framework underlying most modern LLMs, known for self-attention mechanisms allowing global dependencies between input and output.**
- **Tokenization: Process of converting text into tokens which are smaller pieces like words or subwords.**

2. Training Techniques

- **Fine-Tuning: Adjusting a pre-trained model to a specific task by continuing the training process on a new dataset.**
- **Transfer Learning: Leveraging a model trained on one task to perform on another related task.**

3. Applications

- **Text Generation: Generating coherent and contextually relevant text based on a given prompt.**
- **Sentiment Analysis: Determining the sentiment expressed in a piece of text.**
- **Language Translation: Automatic translation of text from one language to another.**

4. Challenges and Ethical Considerations

- **Bias in Data: LLMs can inadvertently learn and perpetuate biases present in the training data.**
- **Model Explainability: Understanding and interpreting how decisions are made by LLMs is a challenge due to their complexity.**

5. Notable Models

- **GPT (Generative Pre-trained Transformer): Series of models designed for a variety of text-based tasks.**
- **BERT (Bidirectional Encoder Representations from Transformers): Designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers.**