Sure, here are concise and interview-focused answers to each of the questions:

1. Define basic principle of OOPs:

- Encapsulation: Bundling data and methods that operate on the data within a single unit, a class.

- Abstraction: Hiding complex implementation details and showing only the necessary features.

- Inheritance: Creating new classes from existing ones to promote code reuse.

- Polymorphism: Allowing entities to be represented in multiple forms, enabling methods to do different things based on the object it is acting upon.

2. Why Java is platform-independent:

- Java is platform-independent due to the Java Virtual Machine (JVM). Java programs are compiled into bytecode, which the JVM interprets and executes on any platform that has a compatible JVM.

3. Why we not install JVM:

- We actually install the JVM when we install the Java Runtime Environment (JRE) or the Java Development Kit (JDK). JVM is bundled within these installations.

4. How does Java enable high performance?:

- Java enables high performance through Just-In-Time (JIT) compilation, which compiles bytecode to native machine code at runtime, and other JVM optimizations like HotSpot.

5. What are the Java IDEs?:

- Popular Java IDEs include IntelliJ IDEA, Eclipse, and NetBeans.

6. What do you mean by Constructor?:

- A constructor is a special method in a class that initializes new objects of that class. It has the same name as the class and does not have a return type.

7. What is meant by Local variable and Instance variable?:

- Local variable: A variable declared inside a method, constructor, or block, and is accessible only within that scope.

- Instance variable: A variable declared inside a class but outside any method, and is associated with an instance of the class.

8. What is a Class?:

- A class is a blueprint or template for creating objects, encapsulating data for the object and methods to manipulate that data.

9. What is Polymorphism?:

- Polymorphism allows objects of different classes to be treated as objects of a common superclass. It is achieved through method overriding and method overloading.

10. Difference between overloading and overriding:

- Overloading: Multiple methods in the same class with the same name but different parameters.

- Overriding: A subclass provides a specific implementation of a method already defined in its superclass.

11. What is meant by Interface?:

- An interface is a reference type in Java, similar to a class, that can contain only constants, method signatures, default methods, static methods, and nested types. Interfaces cannot contain instance fields or constructors.

12. What do you mean by public static void main(String args[])?:

- public: Accessible from anywhere.

- static: Can be called without creating an instance of the class.

- void: Does not return any value.

- main: The entry point of any Java application.

- String args[]: An array of strings that stores command-line arguments.

13. What is meant by Abstract class?:

- An abstract class cannot be instantiated and is meant to be subclassed. It can contain abstract methods (without implementation) and concrete methods (with implementation).

14. What do you mean by constructor overloading?:

- Constructor overloading occurs when a class has more than one constructor with different parameters, allowing different ways to initialize an object.

15. Define type of inheritance:

- Single inheritance: A class inherits from one superclass.

- Multilevel inheritance: A class inherits from a class, which in turn inherits from another class.

- Hierarchical inheritance: Multiple classes inherit from one superclass.

- Multiple inheritance: Not directly supported in Java (achieved through interfaces).

- Hybrid inheritance: Combination of two or more types of inheritance (achieved through interfaces).

16. Diff between final, finally, finalize:

- final: Used to declare constants, prevent inheritance, and prevent method overriding.

- finally: A block used for code that must execute whether or not an exception is thrown.

- finalize: A method called by the garbage collector before an object is destroyed.

17. Diff between interface and abstract class:

- Interface: Can only have abstract methods (until Java 8) and static/default methods (from Java 8). Supports multiple inheritance. No instance fields.

- Abstract class: Can have both abstract and concrete methods. Can have instance fields and constructors. Does not support multiple inheritance directly.

18. Difference between Array and ArrayList:

- Array: Fixed size, can store primitive types and objects.

- ArrayList: Resizable array implementation of the List interface, can only store objects, provides dynamic resizing.

19. Difference between String, StringBuilder, and StringBuffer:

- String: Immutable sequence of characters.

- StringBuilder: Mutable sequence of characters, not synchronized, faster.

- StringBuffer: Mutable sequence of characters, synchronized, slower than StringBuilder.

20. Define mutable and immutable string:

- Mutable string: Can be changed after it is created (e.g., StringBuilder, StringBuffer).

- Immutable string: Cannot be changed after it is created (e.g., String).

21. Explain about Public and Private access specifiers:

- Public: The member is accessible from any other class.

- Private: The member is accessible only within its own class.

22. Difference between Default and Protected access specifiers:

- Default: The member is accessible only within classes in the same package.

- Protected: The member is accessible within its own package and by subclasses in other packages.

23. Difference between HashMap and Hashtable:

- HashMap: Not synchronized, allows one null key and multiple null values, faster.

- Hashtable: Synchronized, does not allow null keys or values, slower.

24. Difference between HashSet and TreeSet:

- HashSet: Does not guarantee any order, based on hash table.

- TreeSet: Guarantees sorted order, based on a red-black tree.

25. What is mean by Collections in Java?:

- Collections in Java are a framework that provides an architecture to store and manipulate a group of objects. It includes interfaces, implementations, and algorithms for data manipulation.

26. What are all the Classes and Interfaces that are available in the collections?:

- Interfaces: Collection, List, Set, Queue, Deque, Map, SortedSet, SortedMap, etc.

- Classes: ArrayList, LinkedList, HashSet, TreeSet, PriorityQueue, ArrayDeque, HashMap, TreeMap, etc.

27. Explain about the different lists available in the collection:

- ArrayList: Resizable array, allows fast random access.

- LinkedList: Doubly-linked list, allows fast insertion and deletion.

- Vector: Similar to ArrayList but synchronized.

- Stack: Subclass of Vector implementing a LIFO stack.

Sure, here are concise and interview-focused answers to the questions related to SQL and databases:

28. What is JOIN? What is Identity?:

- JOIN: A SQL operation used to combine rows from two or more tables based on a related column. Types include INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN.

- Identity: A property in SQL used to automatically generate unique values for a column, typically used for primary keys. Syntax: `IDENTITY(seed, increment)`.

29. What is a view in SQL? How to create one?:

- View: A virtual table based on the result-set of an SQL query. It can encapsulate complex queries.

- Create View:

sql

CREATE VIEW view\_name AS

SELECT column1, column2, ...

FROM table\_name

WHERE condition;

30. What are the uses of view?:

- Simplifies complex queries.

- Provides a layer of security by restricting access to specific data.

- Enhances maintainability and reusability.

- Presents data in a specific format without changing the underlying tables.

31. What is a Trigger?:

- A trigger is a set of SQL statements that automatically execute in response to certain events on a particular table or view, such as INSERT, UPDATE, or DELETE.

32. What is a stored procedure?:

- A stored procedure is a precompiled set of SQL statements stored in the database that can be executed as a single unit, often used to encapsulate repetitive or complex operations.

33. What is the difference between Trigger and Stored Procedure?:

- Trigger: Automatically executes in response to specific events (INSERT, UPDATE, DELETE). Cannot be called directly.

- Stored Procedure: Explicitly called by the user or application. Can accept parameters and return values.

34. What is a transaction? What are ACID properties?:

- Transaction: A sequence of one or more SQL operations treated as a single logical unit of work.

- ACID properties: Ensure reliable transaction processing:

- Atomicity: All operations in a transaction are completed; otherwise, the transaction is aborted.

- Consistency: Transactions transform the database from one valid state to another valid state.

- Isolation: Transactions do not interfere with each other.

- Durability: Once a transaction is committed, changes are permanent.

35. What are indexes?:

- Indexes are database objects that improve the speed of data retrieval operations on a table by providing quick access to rows.

36. What are clustered and non-clustered indexes?:

- Clustered Index: Sorts and stores the data rows in the table based on the index key. Each table can have only one clustered index.

- Non-Clustered Index: Contains a pointer to the data rows stored in the clustered index or the heap. A table can have multiple non-clustered indexes.

37. Query to find 2nd highest salary of an employee:

sql

SELECT MAX(salary) AS second\_highest\_salary

FROM employees

WHERE salary < (SELECT MAX(salary) FROM employees);

38. Why we cannot use WHERE clause with aggregate functions like HAVING?:

- WHERE clause: Filters rows before any groupings are made.

- HAVING clause: Filters groups after the aggregation has been performed. Aggregate functions require grouping, which is processed after the WHERE clause.

39. Difference between primary key and unique key and why one should use unique key if it allows only one null:

- Primary Key: Uniquely identifies each record in a table. Cannot be null. Only one primary key per table.

- Unique Key: Ensures all values in a column or a group of columns are unique. Can be null (only one null value per unique constraint).

- Use of Unique Key: Provides a way to enforce uniqueness on columns that are not primary keys, ensuring data integrity without being a primary key.

Sure, here are concise and interview-focused answers to the additional questions:

40. What’s the difference between materialized and dynamic view?:

- Materialized View: Stores the result of a query physically and can be refreshed periodically.

- Dynamic View: Does not store data physically; it fetches data dynamically from the underlying tables each time it is queried.

41. How to check if a given Binary Tree is BST or not?:

- Perform an in-order traversal of the tree and check if the values are in ascending order.

- Alternatively, use a recursive function to ensure each node's value is within the allowed range (left subtree values < node value < right subtree values).

42. What are the areas of applications of Data Structures?:

- Data structures are used in various areas including databases, operating systems, networking, AI, and more. Specific applications include:

- Arrays: Used in databases, and for data manipulation.

- Linked Lists: Used in dynamic memory allocation, file systems.

- Stacks: Used in expression evaluation, syntax parsing.

- Queues: Used in scheduling, buffering.

- Trees: Used in databases (B-trees), networking (routing tables).

- Graphs: Used in network analysis, social networks.

43. What is the difference between file structure and storage structure?:

- File Structure: Refers to the way data is organized and stored in files on a disk (logical storage).

- Storage Structure: Refers to the way data is physically stored in memory or storage devices (physical storage).

44. Difference between file structure and storage structure:

- See answer 43 above.

45. List the data structures used in RDBMS, Network Data Model, and Hierarchical Data Model:

- RDBMS: Uses tables (relations), indexes (B-trees), and linked lists.

- Network Data Model: Uses graphs (sets, records, and pointers).

- Hierarchical Data Model: Uses trees (parent-child relationships).

46. Which data structure is used to perform recursion?:

- Stack: Recursion uses the call stack to manage function calls and local variables.

47. List the area of applications where stack data structure can be used:

- Function call management (recursion)

- Expression evaluation and syntax parsing

- Undo mechanisms in text editors

- Backtracking algorithms (e.g., maze solving)

- Depth-first search in graph algorithms

48. Write the stack overflow condition:

- Stack Overflow occurs when attempting to push an element onto a stack that is already full.

49. What is the difference between PUSH and POP?:

- PUSH: Adds an element to the top of the stack.

- POP: Removes the top element from the stack.

50. Write the steps involved in the insertion and deletion of an element in the stack:

- Insertion (PUSH):

1. Check if the stack is full.

2. If not full, increment the top pointer.

3. Add the element at the top position.

- Deletion (POP):

1. Check if the stack is empty.

2. If not empty, remove the element at the top.

3. Decrement the top pointer.

51. Which notations are used in Evaluation of Arithmetic Expressions using prefix and postfix forms?:

- Prefix Notation (Polish Notation): Operator precedes the operands (e.g., +AB).

- Postfix Notation (Reverse Polish Notation): Operator follows the operands (e.g., AB+).

52. What are advantages of DBMS over traditional file-based systems?:

- Improved data sharing

- Enhanced data security

- Better data integration

- Minimized data inconsistency

- Improved data access and decision making

- Enhanced data integrity

- Support for concurrent access

53. What are super, primary, candidate and foreign keys?:

- Super Key: A set of one or more columns that uniquely identifies a record in a table.

- Primary Key: A specific super key chosen to uniquely identify records in a table. Cannot be null.

- Candidate Key: A minimal super key; no subset of it can uniquely identify a record.

- Foreign Key: A column or set of columns in one table that uniquely identifies a row of another table, establishing a relationship between tables.

54. What is the difference between primary key and unique constraints?:

- Primary Key: Uniquely identifies each record. Cannot be null. Only one primary key per table.

- Unique Constraint: Ensures all values in a column or a set of columns are unique. Can be null. Multiple unique constraints can be defined in a table.

55. What is database normalization?:

- Database normalization is the process of organizing the attributes and tables of a database to minimize redundancy and dependency. It involves dividing large tables into smaller ones and defining relationships between them.

56. What are the differences between DDL, DML and DCL in SQL?:

- DDL (Data Definition Language): Defines database structure (e.g., CREATE, ALTER, DROP).

- DML (Data Manipulation Language): Manages data within schema objects (e.g., SELECT, INSERT, UPDATE, DELETE).

- DCL (Data Control Language): Controls access to data (e.g., GRANT, REVOKE).

57. What is the difference between HAVING and WHERE clause?:

- WHERE Clause: Filters rows before any groupings are made. Used with SELECT, UPDATE, DELETE statements.

- HAVING Clause: Filters groups after the aggregation has been performed. Used with GROUP BY clause.

Here are concise, interview-focused answers to the additional questions:

98. What are local static variables? What is their use?

- Local Static Variables: Variables declared within a function with the `static` keyword. They retain their value between function calls.

- Use: Maintain state information across multiple function invocations without using global variables.

99. What is the difference between `i++` and `++i`?

- `i++` (Post-increment): Increments `i` after its current value is used.

- `++i` (Pre-increment): Increments `i` before its current value is used.

100. Difference between `++\*p`, `\*p++` and `\*++p`?

- `++\*p`: Increments the value pointed to by `p`.

- `\*p++`: Dereferences `p`, then increments the pointer.

- `\*++p`: Increments the pointer, then dereferences the new pointer.

101. What is an array?

- Array: A collection of elements of the same type stored in contiguous memory locations.

102. How to reference all the elements in a one-dimension array?

- Use a loop (e.g., `for`, `while`) to iterate over the array indices.

103. What is a multidimensional array?

- Multidimensional Array: An array of arrays, such as a 2D array (matrix), where elements are accessed using multiple indices.

104. How are the elements of a 2D array stored in memory?

- Row-major order: Elements of each row are stored in contiguous memory locations.

- Column-major order: Elements of each column are stored in contiguous memory locations.

105. Calculate the address of a random element present in a 2D array, given base address as BA.

- For a row-major 2D array with base address `BA`, element at row `i` and column `j`:

`Address = BA + [(i \* number\_of\_columns) + j] \* element\_size`

106. Are linked lists considered linear or non-linear data structures?

- Linear Data Structures: Because elements are arranged sequentially.

107. What are the advantages of Linked List over an array?

- Advantages:

- Dynamic size

- Ease of insertion/deletion (no need to shift elements)

108. Write the syntax in C to create a node in the singly linked list.

c

struct Node {

int data;

struct Node\* next;

};

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->data = value;

newNode->next = NULL;

109. If you are using C language to implement the heterogeneous linked list, what pointer type should be used?

- Void pointer (`void\*`) to store different data types.

110. What is a doubly linked list?

- Doubly Linked List: A linked list where each node contains a data part and two pointers, `next` and `prev`, pointing to the next and previous nodes respectively.

111. Write the C program to insert a node in a circular singly list at the beginning.

c

struct Node {

int data;

struct Node\* next;

};

void insertAtBeginning(struct Node head, int value) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->data = value;

if (\*head == NULL) {

\*head = newNode;

newNode->next = \*head;

} else {

struct Node\* temp = \*head;

while (temp->next != \*head)

temp = temp->next;

newNode->next = \*head;

temp->next = newNode;

\*head = newNode;

}

}

112. List some applications of queue data structure.

- Applications:

- Job scheduling

- Print spooling

- Handling of interrupts in real-time systems

- Breadth-first search (BFS) in graph algorithms

113. What are the drawbacks of array implementation of Queue?

- Drawbacks:

- Fixed size (cannot expand dynamically)

- Inefficient utilization of space if elements are dequeued (leading to wasted space)

114. What are the scenarios in which an element can be inserted into the circular queue?

- Scenario: When the queue is not full (i.e., the next position of the rear pointer is not the front pointer).

115. What is a dequeue?

- Deque (Double-ended Queue): A data structure where elements can be inserted or removed from both ends (front and rear).

116. What Are The Different Types Of Control Structures In Programming?

- Control Structures:

- Sequential: Default mode, statements executed sequentially.

- Selection: `if`, `else`, `switch` statements.

- Iteration: `for`, `while`, `do-while` loops.

- Jump: `break`, `continue`, `return`, `goto`.

117. What Is `||` Operator And How Does It Function In A Java Program?

- `||` Operator: Logical OR operator. It returns true if at least one of the operands is true.

118. Can The “if” Function Be Used In Comparing Strings?

- In Java, the `if` statement can be used to compare strings using the `.equals()` method, not `==`.

119. What Will Be The Outcome Of The Following Conditional Statement If The Value Of Variable `S` Is 10?

- The specific conditional statement is missing, but generally, the outcome will depend on the condition being tested (e.g., `if (S > 5)` would be true if `S` is 10).

120. Describe The Order Of Precedence With Regards To Operators In C?

- Order of Precedence (highest to lowest):

- Parentheses `()`

- Unary operators `!`, `++`, `--`

- Multiplicative `\*`, `/`, `%`

- Additive `+`, `-`

- Relational `<`, `<=`, `>`, `>=`

- Equality `==`, `!=`

- Logical AND `&&`

- Logical OR `||`

- Assignment `=`, `+=`, `-=`, etc.

121. What Is Wrong With This Statement? `My Name = “robin”;`

- Issue: Missing semicolon (`;`) at the end.

- Correction: `MyName = "robin";` (assuming `MyName` is a properly declared string variable).

122. How Do You Determine The Length Of A String Value That Was Stored In A Variable?

- In C: Use `strlen()` function.

- In Java: Use `.length()` method.

123. Is It Possible To Initialize A Variable At The Time It Was Declared?

- Yes: Example in C: `int x = 10;` Example in Java: `int x = 10;`

124. Why Is C Language Being Considered A Middle Level Language?

- Middle Level: Combines elements of high-level languages (like easy syntax) and low-level languages (like pointer manipulation).

125. What Is The Different File Extensions Involved When Programming In C?

- File Extensions:

- Source file: `.c`

- Header file: `.h`

- Object file: `.o` or `.obj`

- Executable file: `.exe`

126. What Are Reserved Words?

- Reserved Words: Keywords that are part of the language syntax and cannot be used as identifiers (e.g., `int`, `return`, `if`).

127. What is Linked List?

- Linked List: A data structure consisting of nodes where each node contains data and a reference to the next node.

128. What Is FIFO?

- FIFO (First In, First Out): A principle where the first element added is the first one to be removed (used in queues).

129. What is a Data Structure?

- Data Structure: A way of organizing and storing data to enable efficient access and modification.

130. What are linear and non-linear data structures?

- Linear Data Structures: Elements are arranged sequentially (e.g., arrays, linked lists).

- Non-Linear Data Structures: Elements are not in sequence (e.g., trees, graphs).

131. What are the various operations that can be performed on different Data Structures?

- Operations:

- Arrays/Linked Lists: Insertion, deletion, traversal, search, update.

- Stacks: Push, pop, peek.

- Queues: Enqueue, dequeue, front, rear.

- Trees/Graphs: Insertion, deletion, traversal (DFS, BFS), search.

132. How is an Array different from Linked List?

- Array: Fixed size, contiguous memory, random access.

- Linked List: Dynamic size, non-contiguous memory, sequential access.

133. What is Stack and where it

can be used?

- Stack: A linear data structure following LIFO (Last In, First Out) principle.

- Uses: Function call management, expression evaluation, syntax parsing, undo mechanisms.

134. What is a Queue, how it is different from stack and how is it implemented?

- Queue: A linear data structure following FIFO (First In, First Out) principle.

- Difference: Queue is FIFO, Stack is LIFO.

- Implementation: Using arrays or linked lists.

135. What is a Linked List and What are its types?

- Linked List: A sequence of nodes where each node points to the next node.

- Types:

- Singly linked list

- Doubly linked list

- Circular linked list

136. Which data structures are used for BFS and DFS of a graph?

- BFS (Breadth-First Search): Queue.

- DFS (Depth-First Search): Stack or recursion (call stack).

137. How to implement a stack using queue?

- Use two queues. Push operation involves enqueuing to one queue, while pop operation involves dequeuing all elements except the last to another queue and then dequeuing the last element.

138. How to implement a queue using stack?

- Use two stacks. Enqueue by pushing onto the first stack. Dequeue by popping all elements from the first stack to the second stack, then popping from the second stack.

139. Which Data Structure Should be used for implementing LRU cache?

- LRU Cache: Combination of a doubly linked list and a hash map.

140. How to check if a given Binary Tree is BST or not?

- Perform in-order traversal and check if the sequence is sorted in ascending order.

141. What are the areas of applications of Data Structure?

- Databases, operating systems, networking, AI, simulations, compilers.

142. What is the difference between file structure and storage structure?

- File Structure: Logical organization of data in a file.

- Storage Structure: Physical organization of data in memory or storage.

143. Difference between file structure and storage structure:

- See answer 142 above.

144. Tell the list of data structures which are used in RDBMS, Network Data Model, and Hierarchical Data Model.

- RDBMS: Tables, indexes (B-trees), hash tables.

- Network Data Model: Graphs.

- Hierarchical Data Model: Trees.

145. Which data structure is used to perform recursion?

- Stack (call stack).

146. List the area of applications where stack data structure can be used?

- Function call management, expression evaluation, syntax parsing, undo mechanisms, backtracking.

147. Write the stack overflow condition.

- Stack overflow occurs when attempting to push onto a stack that is already full (exceeds its limit).

148. What is the difference between PUSH and POP?

- PUSH: Adds an element to the top of the stack.

- POP: Removes the top element from the stack.

149. Write the steps involved in the insertion and deletion of an element in the stack.

- Insertion (PUSH):

1. Check if the stack is full.

2. Increment the top pointer.

3. Add the element at the top position.

- Deletion (POP):

1. Check if the stack is empty.

2. Remove the element at the top.

3. Decrement the top pointer.

150. Which notations are used in Evaluation of Arithmetic Expressions using prefix and postfix forms?

- Prefix Notation (Polish Notation): Operator precedes the operands (e.g., `+AB`).

- Postfix Notation (Reverse Polish Notation): Operator follows the operands (e.g., `AB+`).

Here are concise, interview-focused answers to the additional questions:

165. What is the minimum number of queues that can be used to implement a priority queue?

- Minimum Number: Two queues can be used—one for storing the elements and the other for priority.

166. Define the tree data structure.

- Tree Data Structure: A hierarchical structure consisting of nodes, where each node has a value and references to its children nodes. The topmost node is the root, and nodes without children are leaves.

167. List the types of tree.

- Types of Trees:

- Binary Tree

- Binary Search Tree (BST)

- AVL Tree

- B Tree

- B+ Tree

- Red-Black Tree

- Splay Tree

- Trie

- Segment Tree

168. What are Binary trees?

- Binary Trees: A type of tree where each node has at most two children, referred to as the left child and the right child.

169. Write the C code to perform in-order traversal on a binary tree.

c

void inOrder(struct Node\* node) {

if (node == NULL)

return;

inOrder(node->left);

printf("%d ", node->data);

inOrder(node->right);

}

170. What is the maximum number of nodes in a binary tree of height k?

- Maximum Nodes: \( 2^{k+1} - 1 \)

171. Which data structure suits the most in the tree construction?

- Dynamic Data Structures: Such as linked lists or dynamic arrays, since tree sizes can vary.

172. Write the recursive C function to count the number of nodes present in a binary tree.

c

int countNodes(struct Node\* node) {

if (node == NULL)

return 0;

return 1 + countNodes(node->left) + countNodes(node->right);

}

173. Write a recursive C function to calculate the height of a binary tree.

c

int height(struct Node\* node) {

if (node == NULL)

return 0;

int leftHeight = height(node->left);

int rightHeight = height(node->right);

return (leftHeight > rightHeight ? leftHeight : rightHeight) + 1;

}

174. How can AVL Tree be useful in all the operations as compared to Binary search tree?

- AVL Tree: It maintains balance (height difference between left and right subtrees is at most 1), ensuring O(log n) time complexity for insertion, deletion, and search operations, unlike an unbalanced BST which can degrade to O(n).

175. State the properties of B Tree.

- Properties of B Tree:

- All leaves are at the same level.

- Each node can have a maximum of m children (where m is the order of the tree).

- A non-leaf node with k children contains \( k-1 \) keys.

- Keys in nodes are in sorted order.

176. What are the differences between B tree and B+ tree?

- B Tree:

- Keys and data can be stored in internal and leaf nodes.

- More efficient for in-memory searches.

- B+ Tree:

- Data is stored only in leaf nodes; internal nodes store keys.

- Leaf nodes are linked, allowing efficient range queries.

177. List some applications of Tree-data structure?

- Applications:

- Hierarchical data representation (e.g., file systems)

- Databases (e.g., B-Trees in indexing)

- Network routing algorithms

- Syntax trees in compilers

- Decision trees in AI and machine learning

178. Differentiate among cycle, path, and circuit?

- Path: A sequence of edges connecting a sequence of vertices.

- Cycle: A path that starts and ends at the same vertex with no repeated edges or vertices except the start/end vertex.

- Circuit: Similar to a cycle, but in directed graphs, it is called a circuit.

179. Mention the data structures which are used in graph implementation.

- Graph Implementations:

- Adjacency matrix

- Adjacency list

- Edge list

180. Which data structures are used in BFS and DFS algorithm?

- BFS: Queue.

- DFS: Stack (or recursion, which uses the call stack).

181. What are the applications of Graph data structure?

- Applications:

- Social networks

- Network routing

- Web page linking (search engines)

- Dependency resolution

- Pathfinding algorithms (e.g., GPS systems)

182. In what scenario, Binary Search can be used?

- Scenario: When the array or list is sorted.

183. What are the advantages of Binary search over linear search?

- Advantages:

- Binary Search: O(log n) time complexity, much faster for large datasets compared to linear search’s O(n).

184. What are the advantages of Selection Sort?

- Advantages:

- Simple and easy to implement.

- Performs well on small lists.

- Requires no additional memory.

185. List Some Applications of Multilinked Structures?

- Applications:

- Graph representations (adjacency list)

- Sparse matrix representations

- Network routing

186. What is the difference between NULL and VOID?

- NULL: A value representing a null pointer, indicating that the pointer points to nothing.

- VOID: A type indicating no value or no type, used in function declarations and pointers.

187. What are advantages of DBMS over traditional file-based systems?

- Advantages:

- Data integrity and security

- Reduces data redundancy

- Efficient data access

- Data consistency

- Transaction management

188. What are super, primary, candidate and foreign keys?

- Super Key: A set of attributes that uniquely identifies a row in a table.

- Primary Key: A specific candidate key chosen to uniquely identify rows.

- Candidate Key: Minimal super keys; no subset can uniquely identify rows.

- Foreign Key: An attribute that creates a relationship between two tables.

189. What is the difference between primary key and unique constraints?

- Primary Key: Uniquely identifies each record in a table and cannot be NULL.

- Unique Constraint: Ensures all values in a column are unique but can have one NULL value.

190. What is database normalization?

- Normalization: The process of organizing data to minimize redundancy and improve data integrity by dividing a database into tables and defining relationships.

191. What are the differences between DDL, DML and DCL in SQL?

- DDL (Data Definition Language): Defines the structure of the database (e.g., CREATE, ALTER, DROP).

- DML (Data Manipulation Language): Manipulates data (e.g., SELECT, INSERT, UPDATE, DELETE).

- DCL (Data Control Language): Controls access to the data (e.g., GRANT, REVOKE).

192. What is the difference between HAVING and WHERE clause?

- WHERE Clause: Filters rows before any groupings are made.

- HAVING Clause: Filters groups after the `GROUP BY` clause is applied.

Here are concise, interview-focused answers to the additional questions:

207. What’s the difference between materialized and dynamic view?

- Materialized View: Stores a copy of the data derived from a query, which can be periodically refreshed. Improves performance for complex queries by storing precomputed results.

- Dynamic View: Does not store data; it is a virtual table created dynamically upon query execution. Always reflects the current data in the underlying tables.

208. What is embedded and dynamic SQL?

- Embedded SQL: SQL statements are embedded within a host language (e.g., C, Java) and are parsed during compilation.

- Dynamic SQL: SQL statements are constructed and executed at runtime, allowing for more flexible and dynamic query generation.

209. What is the difference between CHAR and VARCHAR?

- CHAR: Fixed-length character data type. The defined length is always used, padding with spaces if necessary.

- VARCHAR: Variable-length character data type. Only uses the required space plus one byte for length storage.

210. Define Normal Forms

- Normal Forms: Levels of database normalization designed to reduce redundancy and dependency:

- 1NF (First Normal Form): No repeating groups; each column contains atomic values.

- 2NF (Second Normal Form): 1NF + no partial dependency; every non-key attribute is fully functional dependent on the primary key.

- 3NF (Third Normal Form): 2NF + no transitive dependency; non-key attributes are only dependent on the primary key.

- Higher normal forms include BCNF, 4NF, and 5NF, addressing more specific anomalies.

211. What is cardinality?

- Cardinality: Refers to the uniqueness of data values in a column or the number of relationships between entities in a database. It indicates the number of distinct values in a dataset or the number of instances in a relationship.

212. View Serializable and View Equivalence

- Serializable: A schedule is serializable if it produces the same results as some serial execution of the transactions.

- View Equivalence: Two schedules are view-equivalent if they generate the same results for all the transactions and have the same view of the database (same final state and read/write operations).

213. Cascadeless Recoverable Schedules

- Cascadeless Schedule: A schedule where transactions commit only after ensuring that all transactions from which they have read data have also committed, preventing cascading rollbacks.

- Recoverable Schedule: A schedule where, if a transaction reads a data item from another transaction, the second transaction commits before the first transaction.

214. Difference between IO and UTIL package in Java

- IO Package (`java.io`): Contains classes for system input and output through data streams, serialization, and the file system (e.g., `InputStream`, `OutputStream`, `File`).

- UTIL Package (`java.util`): Contains utility classes, including collections framework, date and time facilities, random number generation, and other utility functions (e.g., `ArrayList`, `HashMap`, `Date`).

215. Write a program in C to count the number of words in a given sentence until EOF is scanned.

c

#include <stdio.h>

#include <ctype.h>

int main() {

int c, in\_word = 0, word\_count = 0;

while ((c = getchar()) != EOF) {

if (isspace(c)) {

if (in\_word) {

in\_word = 0;

word\_count++;

}

} else {

in\_word = 1;

}

}

if (in\_word) {

word\_count++;

}

printf("Word count: %d\n", word\_count);

return 0;

}

216. What is JAVA?

- JAVA: A high-level, class-based, object-oriented programming language designed to be platform-independent through the use of the Java Virtual Machine (JVM). It is widely used for building enterprise-scale applications, mobile applications, and web applications.

217. What are the features in JAVA?

- Features:

- Platform Independent: Write once, run anywhere (WORA) due to JVM.

- Object-Oriented: Supports concepts like inheritance, polymorphism, encapsulation, and abstraction.

- Simple: Syntax is easy to learn and similar to C++ without complex features.

- Secure: Provides a secure environment with its runtime and bytecode verification.

- Multithreaded: Built-in support for multithreaded programming.

- Robust: Strong memory management, exception handling, and garbage collection.

- High Performance: JIT compiler improves execution speed.

- Distributed: Supports distributed computing with EJB and RMI.

- Dynamic: Can dynamically load classes at runtime.

Here are the answers to the additional interview questions:

376. Define the tree data structure.

- A tree is a hierarchical data structure consisting of nodes, with a single node designated as the root. Each node has zero or more child nodes, and no node has more than one parent. Trees are used to represent hierarchical relationships and can be traversed using various algorithms.

377. List the types of tree.

- Types of trees include:

- Binary Tree

- Binary Search Tree (BST)

- AVL Tree

- Red-Black Tree

- B-Tree

- B+ Tree

- Trie

- N-ary Tree

- Splay Tree

378. What are Binary trees?

- A Binary Tree is a tree data structure in which each node has at most two children, referred to as the left child and the right child.

379. Write the C code to perform in-order traversal on a binary tree.

c

#include <stdio.h>

#include <stdlib.h>

struct Node {

int data;

struct Node\* left;

struct Node\* right;

};

void inOrderTraversal(struct Node\* node) {

if (node == NULL) return;

inOrderTraversal(node->left);

printf("%d ", node->data);

inOrderTraversal(node->right);

}

struct Node\* createNode(int data) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->data = data;

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

int main() {

struct Node\* root = createNode(1);

root->left = createNode(2);

root->right = createNode(3);

root->left->left = createNode(4);

root->left->right = createNode(5);

printf("In-order traversal: ");

inOrderTraversal(root);

printf("\n");

return 0;

}

380. What is the maximum number of nodes in a binary tree of height k?

- The maximum number of nodes in a binary tree of height \( k \) is \( 2^{k+1} - 1 \).

381. Which data structure suits the most in the tree construction?

- Linked lists or dynamic node structures suit the most for tree construction because they allow for efficient insertion and deletion of nodes.

382. Which data structure suits the most in the tree construction?

- Linked lists or dynamic node structures (repeated question).

383. Write the recursive C function to count the number of nodes present in a binary tree.

c

int countNodes(struct Node\* node) {

if (node == NULL) return 0;

return 1 + countNodes(node->left) + countNodes(node->right);

}

384. Write a recursive C function to calculate the height of a binary tree.

c

int height(struct Node\* node) {

if (node == NULL) return 0;

int leftHeight = height(node->left);

int rightHeight = height(node->right);

return (leftHeight > rightHeight ? leftHeight : rightHeight) + 1;

}

385. How can AVL Tree be useful in all the operations as compared to Binary search tree?

- AVL Trees maintain balance by ensuring the heights of subtrees differ by at most one, providing O(log n) time complexity for insertion, deletion, and search operations, which prevents degradation to O(n) as in unbalanced binary search trees.

386. State the properties of B Tree.

- Properties of B-Trees:

- All leaves are at the same level.

- Each node has a minimum of ceil(m/2) and a maximum of m children (where m is the order of the tree).

- Each node contains a number of keys such that ceil(m/2) - 1 ≤ keys ≤ m - 1.

- The keys are kept in sorted order.

387. What are the differences between B tree and B+ tree?

- B-Tree: Internal nodes can store keys and values. Leaf nodes contain keys and values.

- B+ Tree: Only leaf nodes store keys and values. Internal nodes store only keys to guide searches. Leaf nodes are linked to provide a sequential linked list for faster range queries.

388. List some applications of Tree-data structure.

- Applications include:

- Database indexing (B-Trees, B+ Trees)

- File system directories

- Network routing algorithms (Tries)

- Hierarchical data representation (XML/HTML documents)

- Priority queues (Binary Heaps)

389. Differentiate among cycle, path, and circuit?

- Path: A sequence of vertices where each adjacent pair is connected by an edge, with no vertex repeated.

- Cycle: A path that starts and ends at the same vertex with no other vertex repeated.

- Circuit: A cycle but in the context of directed graphs.

390. Mention the data structures which are used in graph implementation.

- Data structures include:

- Adjacency Matrix

- Adjacency List

- Edge List

391. Which data structures are used in BFS and DFS algorithm?

- BFS (Breadth-First Search): Queue

- DFS (Depth-First Search): Stack (or recursion, which uses the call stack)

392. What are the applications of Graph data structure?

- Applications include:

- Social networks

- Web page linking

- Network routing

- Dependency graphs

- Pathfinding algorithms (e.g., in maps)

393. In what scenario, Binary Search can be used?

- Binary search is used when the list is sorted, and quick lookup, insert, and delete operations are required.

394. What are the advantages of Binary search over linear search?

- Advantages include:

- O(log n) time complexity compared to O(n) for linear search.

- More efficient for large datasets due to the divide-and-conquer approach.

395. What are the advantages of Selection Sort?

- Advantages include:

- Simple to implement

- Performs well on small datasets

- Doesn't require additional memory for sorting (in-place)

396. List Some Applications of Multilinked Structures?

- Applications include:

- Sparse matrix representations

- Network routing tables

- Social network graph representation

- Graph adjacency lists

397. What is the difference between NULL and VOID?

- NULL: A value that represents a null pointer, indicating no valid data/address.

- VOID: A type indicating no value or an empty return type in functions.

398. What are advantages of DBMS over traditional file-based systems?

- Advantages include:

- Data abstraction and independence

- Reduced data redundancy and inconsistency

- Improved data integrity and security

- Concurrent access and crash recovery

- Efficient data access and management

399. What are super, primary, candidate and foreign keys?

- Super Key: A set of one or more columns that can uniquely identify a row in a table.

- Primary Key: A unique super key chosen to uniquely identify rows.

- Candidate Key: A minimal super key; any candidate key can be a primary key.

- Foreign Key: A field in one table that uniquely identifies a row of another table, creating a link between the two tables.

400. What is the difference between primary key and unique constraints?

- Primary Key: Uniquely identifies each row in a table and cannot be NULL.

- Unique Constraint: Ensures all values in a column or a set of columns are unique, but can have one NULL value.

401. What is database normalization?

- Database Normalization: The process of organizing a database to reduce redundancy and improve data integrity. It involves dividing tables into smaller, related tables and defining relationships between them to achieve higher normal forms like 1NF, 2NF, 3NF, etc.

### View Serializable and View Equivalence (In Short)

#### View Serializability

- Definition: A schedule is view-serializable if it can be transformed into a serial schedule (where transactions are executed one after the other) by reordering its operations without changing the outcome.

- Conditions:

1. Initial Read: Transactions must read the initial value of data items in the same order as in the serial schedule.

2. Update Read: If a transaction reads a value written by another transaction, the same read must occur in both schedules.

3. Final Write: The transaction that performs the final write on a data item must be the same in both schedules.

- Purpose: Ensures concurrent transactions produce the same results as if they were executed serially, maintaining database consistency.

#### View Equivalence

- Definition: Two schedules are view-equivalent if:

1. Initial Read: Transactions read the initial value of data items in the same order.

2. Update Read: Transactions read values written by the same transactions.

3. Final Write: The final write operations on data items are performed by the same transactions.

- Purpose: Ensures that two different schedules produce the same final state of the database and the same sequence of read/write operations.

### Key Points

- View Serializability: A property of a schedule ensuring it is equivalent to some serial execution of transactions.

- View Equivalence: A property comparing two schedules to determine if they result in the same final state and read/write sequences.

- Use in Databases: Essential for maintaining consistency and integrity when multiple transactions are executed concurrently.