Quick Review Guide

This is supposed to make a document which could be viewed before going for interview for fresh graduates. A quick review is not covering all aspects, but mostly covers a quick guide that you must know these things before going for interview. This document consists of some topics that you need to be reviewed, the last night before interview. Ah! Let's move towards things to make it easy ;)

➢ Important Subjects

1. Object Oriented Programming (OOP or OOPS)
2. Data Structures and Algorithms (DSA)
3. Databases
4. Analysis of Algorithms
5. Software Engineering (basics)
6. Object Oriented Analysis and Designs (OOAD)
7. Operating Systems

➢ Topics Subject-wise

1. Object Oriented Programming (OOP or OOPS)

Number	Topics
1.	Classes & Objects
2.	4 Component of Object Oriented Programming ➢ Inheritance ➢ Polymorphism ➢ Abstraction ➢ Encapsulation
3.	Overloading and Overriding
4.	Runtime VS Compile time Polymorphism
5.	Abstract Classes VS Interfaces (their differences and when they will use)
6.	Call by value & Call by reference
7.	Memory structure of objects
8.	Pointer VS Reference
9.	Structure VS Classes

10.	Why OOP is introduced? Actually we can done this with structural programming!
11.	Calling Sequence of Constructors and Destructors in inheritance
12.	Friend function and class
13.	Abstraction VS Encapsulation (You should have clear concepts of this)
14.	Virtual function VS Pure Virtual functions
15.	Diamond Problem
16.	Access Specifiers (public, private, protected, <<default>>)
17.	Multiple VS Multilevel Inheritance
18.	Aggregation VS Composition (Their implementation corresponding to coding)
19.	Association?
20.	Operator Overloading
21.	Finally, finalize and final keywords
22.	super vs this keywords
23.	Static (variable, method and class)
24.	Early binding VS Late Binding
25.	Shallow copy and Deep copy
26.	Array VS Linked list (when array will use and when linked list)
27.	Static binding and dynamic binding

2. Data Structures and Algorithms (DSA)

Number	Topic
1.	Insertion, deletion, update and creating costs in array and linked lists
2.	Linked list and its types ➢ Singly Linked list

	➢ Doubly Linked list ➢ Circular Linked list (All operations on linked list must be prepared e.g. - Nth last element finding in linked list - Find intersection point in two linked lists - Find loops in linked list etc. All these must be solved within O(n) and no extra data structure used)
3.	Queue VS Stack
4.	Greedy, Divide and Conquer Algorithms
5.	Dynamic Programming
6.	Postfix and Prefix expression conversion (using stack) - Infix to postfix (vice versa) - Infix to prefix (vice versa)
7.	Searching Techniques & their complexities ➢ Linear Search ➢ Binary Search
8.	Hashing (Hash Table)
9.	Sorting Algorithms & their complexities ➢ Bubble Sort ➢ Insertion Sort ➢ Selection Sort ➢ Merge Sort ➢ Shell Sort ➢ Quick Sort (which one is better for which scenario?)
10.	Graphs & its traversal ➢ Depth First Traversal ➢ Breadth First Traversal
11.	Some Searches - Best First Search - Depth First Search - Depth First Search with Iterative Deepning

12.	Tree Data Structures (root, leaf, parent, child nodes) ➢ Binary Search Tree (BST) (Technique linked with indexing in Database) ➢ AVL Tree ➢ Spanning Tree (why this is used for?) ➢ Red Black Tree ➢ Heap (MinHeap, MaxHeap, Heapify) Many more examples of trees are there!
13.	Recursion and its everyday example
14.	Infix, postfix and prefix conversions (using trees)
15.	Tree Traversing - In – Order (Left, Node, Right) (LNR) node in center - Pre – Order (Node, Left, Right) (NLR) node in left - Post – Order (Left, Right, Node) (LRN) node at right

3. Databases

Number	Question
1.	DBMS?
2.	SQL and MySQL?
3.	Data Models and Data Schemas
4.	ER Model (Complete concepts)
5.	Relational DB
6.	Database Normalizations (1NF, 2NF, 3NF, BCNF, 4NF, 5NF)
7.	Joins and its types - Self Joins - Outer joins (left, right, full) - Equi-joins - Natural or Cartesian Joins
8.	Sub queries

9.	Indexing (in what scenario we index data, why we index data, what is benefits of indexing, must know about syntax of indexing in SQL) & its types
10.	Transactions? And its properties (ACID) - Atomicity - Consistency - Durability - Isolation
11.	Stored Procedures (What are they? Purpose? When created and when not?)
12.	Triggers (types of triggers, why we use them?)
13.	Commit, rollback and save points
14.	Cluster and Non-Cluster Database
15.	De – Normalization of Database
16.	Which Databases are not relational? Examples
17.	DML, DDL, DCL
18.	Temporary Tables
19.	NULL (questions like when null multiplied with a number what would be the result etc.)
20.	Having and Where Clause difference
21.	NVL vs NVL2
22.	SQL Injection
23.	Connection Pooling
24.	IN, ALL keywords
25.	Constraints
26.	Foreign key and data deletion from foreign key table
27.	Roles in Database
28.	Case Statement Queries

4. Analysis of Algorithms

Number	Topics
1.	What is algorithm?
2.	Asymptotic Notations (Ο, Ω, Θ)
3.	Brute force algorithm
4.	Greedy Algorithms and Examples
5.	Minimum Spanning Trees finding (Kruskal and Prims method)
6.	Shortest Paths (Dijkstra, Bellman-ford, Floyad-Warshal Algorithm)
7.	Decision Problem? NP, NPC (NP Complete), NPH (NP Hard)
8.	Sorting Algorithms complexities - Bubble - Quick - Merge - Counting - Bucket/Bin - Radix - Shell
9.	In-place and not in-place algorithms
10.	Stable and Not Stable Algorithms

5. Software Engineering

Number	Topics
1.	Processes and Software Processes?
2.	4 P's
3.	Agile and Scrum
4.	Actors (Primary, Secondary, Off-Stage)

5.	Class Diagrams, Domain Models
6.	Use cases VS Use case Diagram VS Use case Description
7.	SDLC
8.	Requirement Engineering and its phases
9.	DFD VS Flow Chart
10.	CRUD operations
11.	Sequence VS Collaboration Diagrams
12.	N – Tier Applications
13.	Testing Techniques - Unit - Stress - Load - Smoke - Sanity - Regression - Integration - Black box - White box
14.	Debugging?
15.	Validation VS Verification

6. Object Oriented Analysis and Design (OOAD)

Number	Topics
1.	UML Diagrams
2.	UML – Structural Diagrams UML – Behavioral Diagrams
3.	Testing and Quality Assurances
4.	Software Design Patterns

	- Factory - Abstract Factory* - Singleton* - Builder* (Question was asked like you have laptop components now build laptop in OOP approach) - Prototype - Adapter* - Bridge - Filter - Composite - Decorator - Façade* - Flyweight - Proxy* - Command - Interpreter - Iterator* - Mediator* - Memento - Observer* - State - Null Object - Strategy* - Template - Visitor - MVC* *Important ones
5.	Grasp Design Patterns
6.	Gang of Four (GOF)
7.	Software Development Techniques - Agile Software Development - Rapid Application Development - Dynamic Systems Development Model - Crystal Methods - Joint Application Development - Lean Development etc.
8.	Methodologies of SDLC - Linear (Waterfall) - Iterative Model

	- Spiral Model - Agile Model - V – Shaped Model - Big Bang Model

7. Operating Systems

Number	Topic
1.	Multi processing VS Multi programming VS Multi Threading
2.	Interrupts, Traps, Signals
3.	Deadlock and types to resolve deadlocks
4.	System call VS Library calls
5.	Process Control Block (PCB) and its parts
6.	Schedulers (long term, short term, medium term)
7.	Pipe VS Fifo
8.	Do you know about some shell commands
9.	Threads VS Processes
10.	User level VS Kernel Level Threads
11.	Scheduling Criteria? - Shortest job First - Longest job First - Round Robbin etc.
12.	Synchronization and Critical Sections
13.	Critical Section Problems
14.	Starvation
15.	Address Binding
16.	Paging, Segmentation concepts (should be clear)

17.	Hit ration VS Miss Ration
18.	Overlays
19.	Swapping
20.	Semaphore VS Mutex
21.	Preemptive and non – preemptive?
22.	Race Condition?
23.	What process shares and what thread shares?
24.	Any real life example of deadlock?

So these are the subjects and related topics need to be revised before going to interview. Repeating that these are not all topics but just I remember for revision. Questions will be from these mostly and can be changed a little bit. Happy Learning ☺