DBMS:
important topics:

Why do we need DBMS	https://www.youtube.com/watch?v=T7AxM7Vqvaw
File management system vs DBMS	https://www.youtube.com/watch?v=ZtVw2iuFI2w
Data abstraction	https://www.youtube.com/watch?v=5fs1IdO6B5c
ER model, relational model	https://www.youtube.com/watch?v=VJUk-_CsqKw https://www.youtube.com/watch?v=UWNOchMUh8U
SQL and important keywords used	Already done
SQL queries	Already done
Joins and its types	Already done
Views	https://www.youtube.com/watch?v=8jU8SrAPn9c
Indexing	https://www.youtube.com/watch?v=fsG1XaZEa78 sanchit jain

Normalization basic definitions and usage Sanchit jain
Transactions (Lock based protocol for concurrency control)
ACID properties
https://www.youtube.com/watch?v=75T6muWuEFI&list=PLbKVlnc3OmnwG-2-EeoheUZxBHwJhRaZ-

SQL vs NoSQL https://www.youtube.com/watch?v=ZS_kXvOeQ5Y
Deadlock in DBMS
Starvation
https://youtu.be/Y80mas3B62M?list=RDCMUC-x_kRNcI1102UR5SmXOrpg
https://www.youtube.com/watch?v=loe4ERAI_3o

Questions on what database to be used
Scaling in databases
Cache and in-memory database

Also, practice standard interview questions from:

Most Standard topics : SQL queries, Normalization, transactions
[https://prepinsta.com/database-management-system-dbms/]Revision:
https://www.geeksforgeeks.org/last-minute-notes-dbms/

Interview questions:
https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions/
https://www.geeksforgeeks.org/commonly-asked-dbms-interview-questions-set-2/

DonneMartin book:
https://github.com/donnemartin/system-design-primer#database

By striver:

1. What is DBMS ? Mention advantages..
2. What is Database?
3. What is a database system?
4. What is RDBMS ? Properties..
5. Types of database languages
6. ACID properties (VVVVV IMP)
7. Difference between vertical and horizontal scaling
8. What is sharding
9. Keys in DBMS
10. Types of relationship
11. Data abstraction in DBMS, three levels of it
12. Indexing in DBMS
13. What is DDL (Data Definition Language)
14. What is DML (Data Manipulation Language)
15. What is normalization ? Types of them ..
16. What is denormalization ?
17. What is functional dependency ?
18. E-R Model ?
19. Conflict Serializability in DBMS ..
20. Explain Normal forms in DBMS
21. What is CCP ? (Concurrency Control Protocols)
22. Entity, Entity Type, Entity Set, Weak Entity Set..
23. What are SQL commands ? Types of them..
24. Nested Queries in SQL ?
25. What is JOIN .. Explain types of JOINs
26. Inner and Outer Join
27. Practice sql queries from leetcode
28. Diff between 2 tier and 3 tier architecture
29. Diff between TRUNCATE and DELETE command ..
30. Difference between Intension and Extension in a DataBase
31. Difference between share lock and exclusive lock, definition of lock.

DATABASE MANAGEMENT SYSTEM:

1. SQL Queries
2. Normalization (Meaning, Reason of normalizing tables, Different Normal Forms)
3. Lossless and Lossy Decomposition
4. Different types of keys in a table (Primary, Composite, Candidate, Super Key)
5. ER model (Meaning and Components)
6. File Structure (B-trees, Indexing)
7. Concurrency issues

OBJECT ORIENTED PROGRAMMING (C++):

1. Concepts of OOPS (Important)
2. Types of polymorphism
3. Virtual Functions - Run-time Polymorphism
4. Inheritance (Types, Virtual Class, Dreaded Diamond Problem)
5. Constructors and Destructors (Private Constructors and Destructors, Virtual Destructors)
6. Smart pointers
7. Singleton class
8. Friend function and friend class

OS:

Explain Operating System in layman terms
Types of OS - Batch OS, Multiprogramming OS, Multitasking OS, Time Sharing OS, Distributed OS, Real Time OS
RAM vs ROM + Types (Asked in DE Shaw)
Virtualization vs Containerization (OR Virtual machine vs Docker)
Program vs Process (Asked in DE Shaw)
Process vs Thread
User-level thread vs kernel level thread
Differences between multi-threading, multi-processing, multiprogramming, multi-tasking
Microservices based architecture
Process scheduling - Basic terminology like scheduling queue, different times in process like arrival time, Completion Time, Burst Time, Turn Around Time, Waiting Time (WT)
Different process scheduling algorithms - FCFS, SJF, SRTF etc.
Different criterias used - CPU utilization, throughput, response time etc.
Explain how does a process gets executed inside memory (Asked in DE Shaw)
Optimal number of threads for a process
Preemptive vs Non-preemptive scheduling
Some important terms associated with scheduling algorithms - Problem of Ageing, Starvation, Deadlock [should know basic definitions at least]
Synchronization - Semaphores, mutex vs counting semaphore, critical section problem + their three conditions (Mutual exclusion, Progress, Bounded waiting)
Deadlock : Basic definition, Necessary conditions, handling techniques
Memory Management [Very IMP]: Primary vs Secondary Memory, memory Allocation while running a process [IMP], Paging and segmentation (Basics should be clear)
Thrashing [IMP concept]
What is Cache and why is it used?
Memory Partitioning
What is Virtual memory and why?
LRU Cache implementation

Also, practice standard interview questions from:

Interview questions:
https://www.geeksforgeeks.org/commonly-asked-operating-systems-interview-questions-set-1/
https://www.interviewbit.com/operating-system-interview-questions/

Other questions for practice:
https://www.javatpoint.com/operating-system-interview-questions

Revision:
https://www.geeksforgeeks.org/last-minute-notes-operating-systems/

Book (if enough time):
Operating System Principles by Galvin: https://amzn.to/2UuxwEJ

By striver:

1. What is the main purpose of an operating system? Discuss different types?
2. What is a socket, kernel and monolithic kernel ?
3. Difference between process and program and thread? Different types of process.
4. Define virtual memory, thrashing, threads.
5. What is RAID ? Different types.
6. What is a deadlock ? Different conditions to achieve a deadlock.
7. What is fragmentation? Types of fragmentation.
8. What is spooling ?
9. What is semaphore and mutex (Differences might be asked)? Define Binary semaphore.
10. Belady's Anomaly
11. Starving and Aging in OS
12. Why does trashing occur?
13. What is paging and why do we need it?
14. Demand Paging, Segmentation
15. Real Time Operating System, types of RTOS.
16. Difference between main memory and secondary memory.
17. Dynamic Binding
18. FCFS Scheduling
19. SJF Scheduling
20. SRTF Scheduling
21. LRTF Scheduling
22. Priority Scheduling
23. Round Robin Scheduling
24. Producer Consumer Problem
25. Banker's Algorithm
26. Explain Cache
27. Diff between direct mapping and associative mapping
28. Diff between multitasking and multiprocessing

By apni kaksha:
https://drive.google.com/file/d/1CljO4lsVcxLXj59X0OMBB5WNhG0fzVOw/view

By nishant:

OPERATING SYSTEMS:

1. Operating Systems and its types.
2. Process Management (Attributes, States of Process)
3. CPU Scehduling Algorithms (FCFS, SJF, SRTF, Round Robin, Priority Scheduling)
4. Process Synchronisation (Necessary Conditions, Bakery Algorithm, Producer-Consumer Problem, Dining Philosopher Problem, Read-Write Problem)
5. Mutex and Semaphores (Important)
6. Threads (Important)
7. Deadlocks (Necessary Conditions, Banker's Algorithm, Deadlock Prevention, Avoidance, Recovery, Correction)
8. Memory Management (Multi-partition, External and Internal Fragmentation, Paging, Segmentation)
9. Virtual Memory (Demand Paging, Page replacement algorithms, Thrashing)
10. File allocation (Coninuous, Linked and Index File allocation)
11. Disk Scheduling Algorithms (FIFO, SCAN, C-SCAN, LOOK, C-LOOK)

Compute Networks:

1. Define network
2. What do you mean by network topology, and explain types of them
3. Define bandwidth, node and link ?
4. Explain TCP model ..
5. Layers of OSI model
6. Significance of Data Link Layer
7. Define gateway, difference between gateway and router ..
8. What does ping command do ?
9. What is DNS, DNS forwarder, NIC, ?
10. What is a MAC address ?
11. What is IP address, private IP address, public IP address, APIPA ?
12. Difference between IPv4 and IPv6
13. What is subnet ?
14. Firewalls
15. Different type of delays
16. 3 way handshaking
17. Server-side load balancer
18. RSA Algorithm
19. What is HTTP and HTTPS protocol ?
20. What is SMTP protocol ?
21. TCP and UDP protocol, prepare differences
22. What happens when you enter “google.com” (very very famous question)
23. Hub vs Switch
24. VPN, advantages and disadvantages of it
25. LAN

COMPUTER NETWORKS:

1. OSI Model (Functions of different layers)
2. TCP/IP Protocol Suite
3. Data Link Layer (Error detection techniques, Framing)
4. Network Layer (Routing protocols, IPv4 and IPv6 - Supernetting and Subnetting)
5. Transport Layer (3 way Handshake, TCP packet components, UDP packet components, Advantages of UDP over TCP, Applications of UDP)

CR sheet by striver 3 months:

MUST-DO ALGORITHMS for CODING ROUNDS

1.Binary Search
   a. https://codeforces.com/problemset/problem/1354/B (Easy)
   b. https://www.interviewbit.com/problems/allocate-books/ (Medium)
   c. https://codeforces.com/problemset/problem/1359/C
      (Hard -> no need to do if very less time is left)
2. Prefix Sum
   a. https://cses.fi/problemset/task/1646 (easy)
   b. https://www.hackerrank.com/contests/ab-yeh-kar-ke-dikhao/challenges/kj-and-street-lights/problem (Medium -> Scanline Algo)
   c. https://www.codechef.com/CENS2020/problems/CENS20A (Hard)

3. Primes/Divisors
   a. https://www.codechef.com/problems/CNTPRIME (Easy)(Sieve)
   b. https://www.spoj.com/problems/PRIME1/ (Medium) (Segmented Sieve)
   c. https://cses.fi/problemset/task/2182 (hard -> can be left)

4. Divide and Conquer
   a. https://www.spoj.com/problems/INVCNT/ (Easy)
   b. https://cses.fi/problemset/task/1628 (Medium)
   c. https://lightoj.com/problem/funny-knapsack (Hard -> can be left)

5. String Algorithms
   a. https://cses.fi/problemset/task/1753 (Easy) (KMP, Z, Rabin-Karp) (Solve using all 3 algos)
   b. https://cses.fi/problemset/task/1111 (Medium)
   c. https://codeforces.com/problemset/problem/271/D (Medium/Hard)

6. Tree Algorithms
   a. https://cses.fi/problemset/task/1674 (Easy)
   b. https://cses.fi/problemset/task/1131 (Medium)
   c. https://cses.fi/problemset/task/1135 (Hard, covers LCA using Binary Lifting)

7. Graph Algorithms
   a. BFS Questions super duper important (https://cses.fi/problemset/task/1192)
      (Also do problems like https://cses.fi/problemset/task/1193)
   b. https://cses.fi/problemset/task/1671 (Dijsktra)
   c. https://www.spoj.com/problems/EC_P/ (Bridges)
   d. https://www.spoj.com/problems/SUBMERGE/ (Articulation Point)
   e. Rest do all Graph problems from Striver's Graph series (https://www.youtube.com/watch?v=YTtpfjGlH2M&list=PLgUwDviBIf0rGEWe64KWas0Nryn7SCRWw)

8. Disjoint Set
    a. https://www.hackerearth.com/practice/data-structures/disjoint-data-strutures/basics-of-disjoint-data-structures/practice-problems/algorithm/disjoint-set-union/
    https://www.youtube.com/watch?v=3gbO7FDYNFQ&t=11s

    b. https://codeforces.com/contest/25/problem/D (Medium)
    c. https://www.spoj.com/problems/CLFLARR/ (Hard -> offline solution)

9. Segment Trees
    a. https://cses.fi/problemset/task/1647 (Simple range query)
    (https://www.youtube.com/watch?v=-dUiRtJ8ot0)

    a. https://cses.fi/problemset/task/1649 (Range query with point update)
    (https://www.youtube.com/watch?v=-dUiRtJ8ot0)

    b. https://cses.fi/problemset/task/1735 (hard-> can be left ..)
    (https://www.youtube.com/watch?v=rwXVCELcrqU)

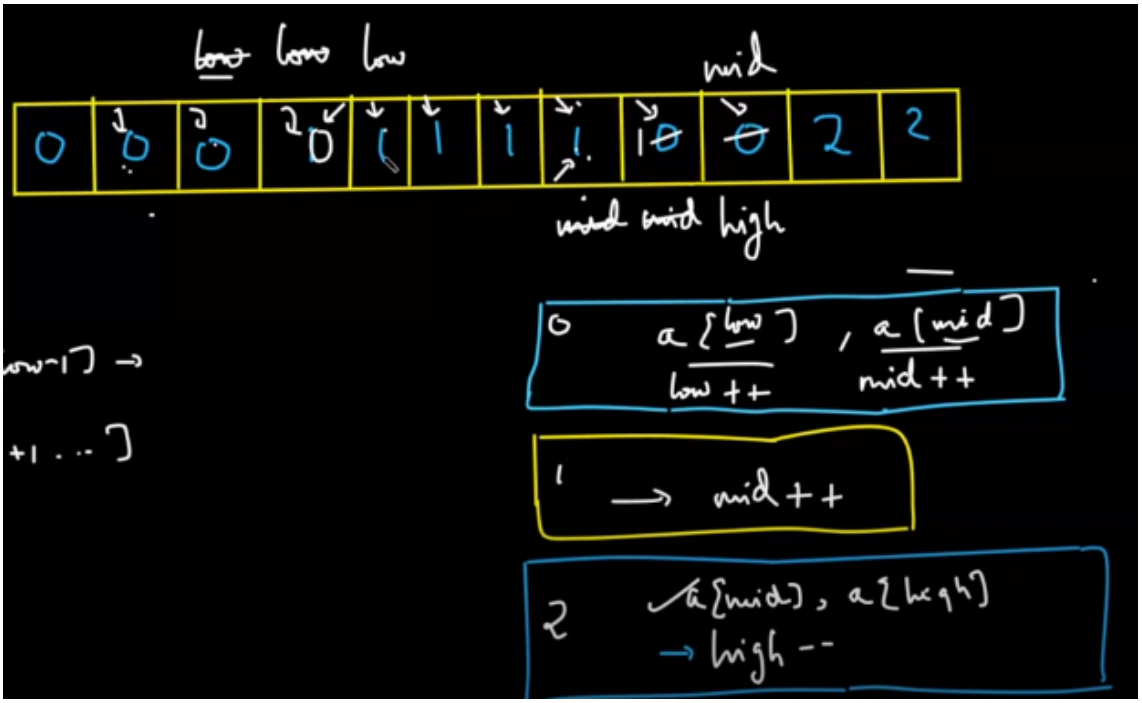
10. Dynamic Programming
    a. Generally the problems are variations of standard DP problems in geeksforgeeks. Do the problems named as “DP-3” to DP-28” on GFG, will automatically be covered if you doing SDE sheet)
    b. Digit DP (hard -> might appear if you are giving rounds in Hackerearth, else will not..)
    https://cses.fi/problemset/task/2220

SDE Sheet by striver 30 days prep:

Only start doing these problems if you feel you are comfortable with solving basic problems of DSA. Once you are, you can start preparing for these problems, because these problems are solely interview based.

1. **Sort an array of 0's 1's 2's without using extra space or sorting algo**
   https://www.youtube.com/watch?v=oaVa-9wmpns&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=2 (Problem link in description) leetcode

   Imp:
   Brute force: simple sort $O(N^2)$
   Optimize: two pass O(2N)
   Optimal: Dutch Flag Algorithm(variation):

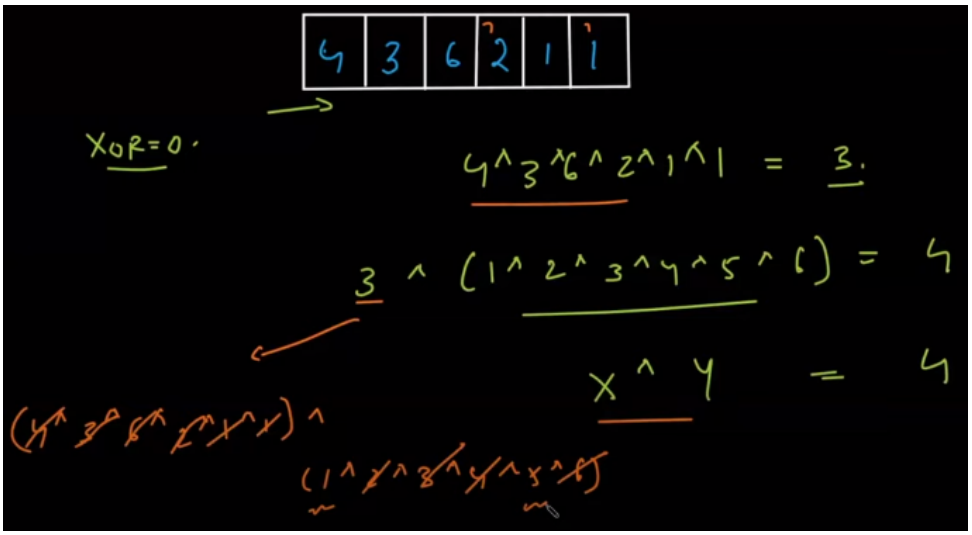


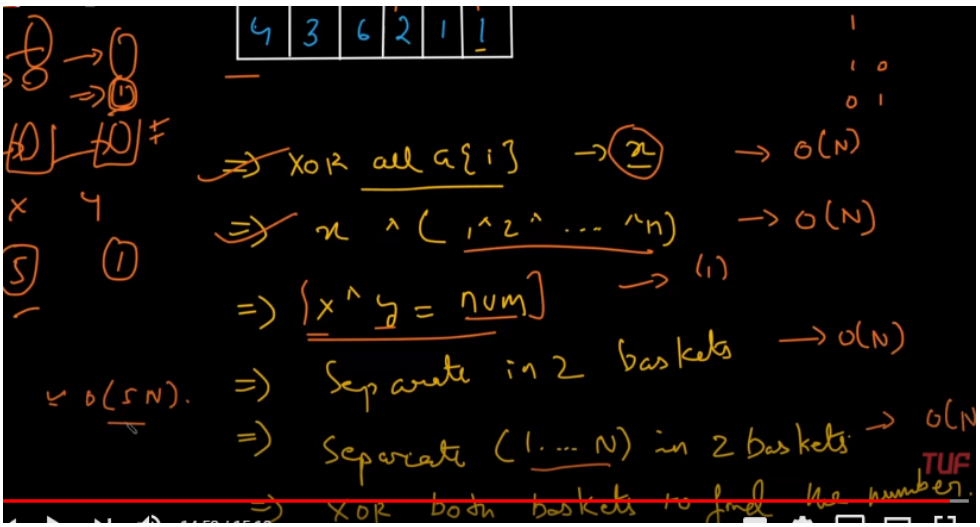
   Code:

```
low=mid=0
    high=len(nums)-1

    while mid<=high:
      if nums[mid]==0:
        nums[mid],nums[low]=nums[low],nums[mid]
        mid+=1
        low+=1
      elif nums[mid]==1:
        mid+=1
      elif nums[mid]==2:
        nums[mid],nums[high]=nums[high],nums[mid]
        high-=1
    return nums
```

2. **Repeat and Missing Number**
   https://www.youtube.com/watch?v=5nMGY4VUoRY&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=3 (Problem link in description)

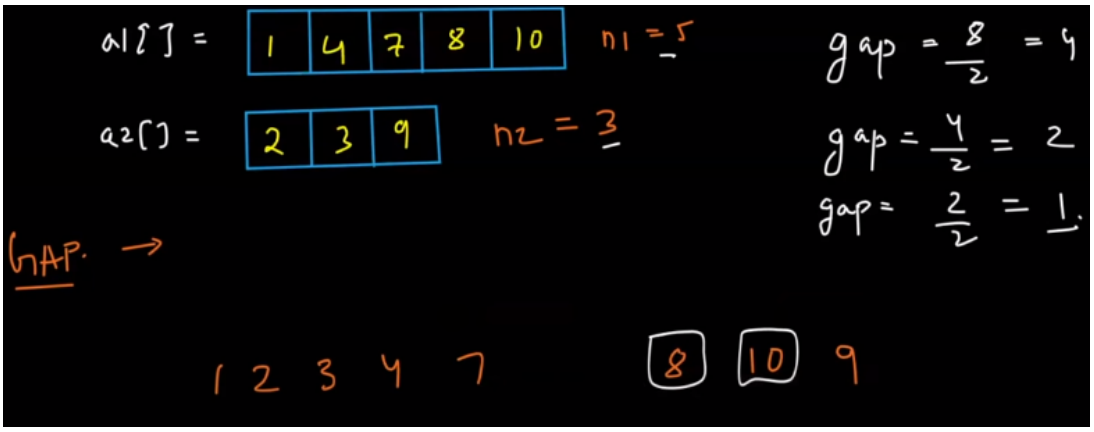
   Brute force: O(N^2)
   Optimize: using Hashing with two pass O(2N) and space is O(N)
   Optimal: using square $X^2 - Y^2 = S^2$ but not used for all variants as squaring may require a long number to store.
   More Optimal: using XOR as X ^ Y = XOR of all numbers separating in two different BUCKETS X and Y.





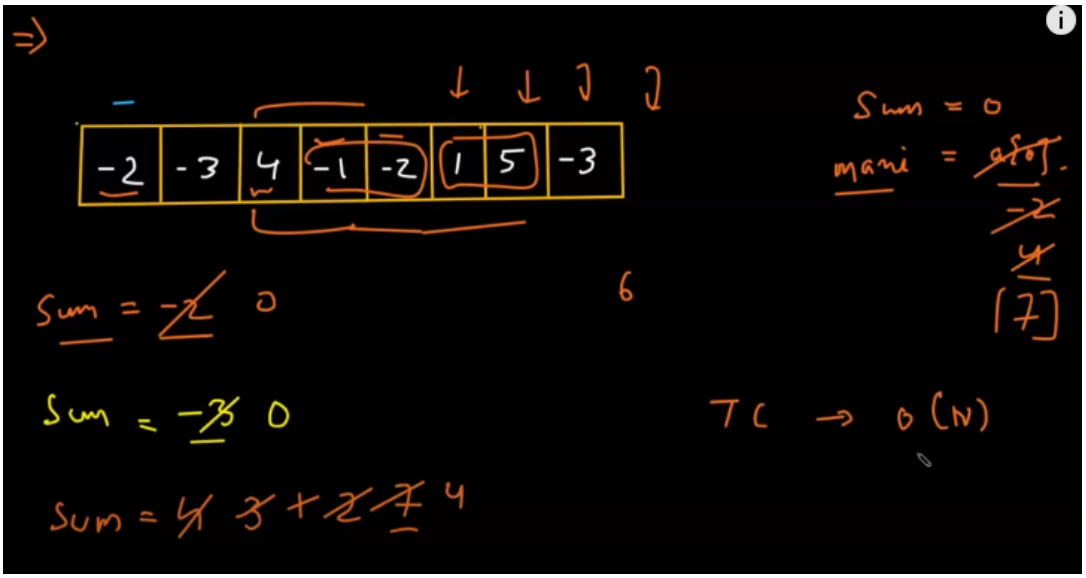
   Code:

3. **Merge two sorted Arrays without extra space**
   https://www.youtube.com/watch?v=hVl2b3bLzBw&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=4 (Problem link in description)
   Brute force: using Third Array which combine both given array and then replaces it with the first array with time complexity as O(N)+O(nlogn)+O(N) and space as O(N)
   Better: perform a variation of insertion sort by swapping minimum element from second array with time as O(n*m) and space as only O(1)
   Optimal: Using GAP algorithm aka SHELL sort with time as O(logN) as we are dividing gap by 2 every time + O(N) for iteration = O(NlogN)



   Code:

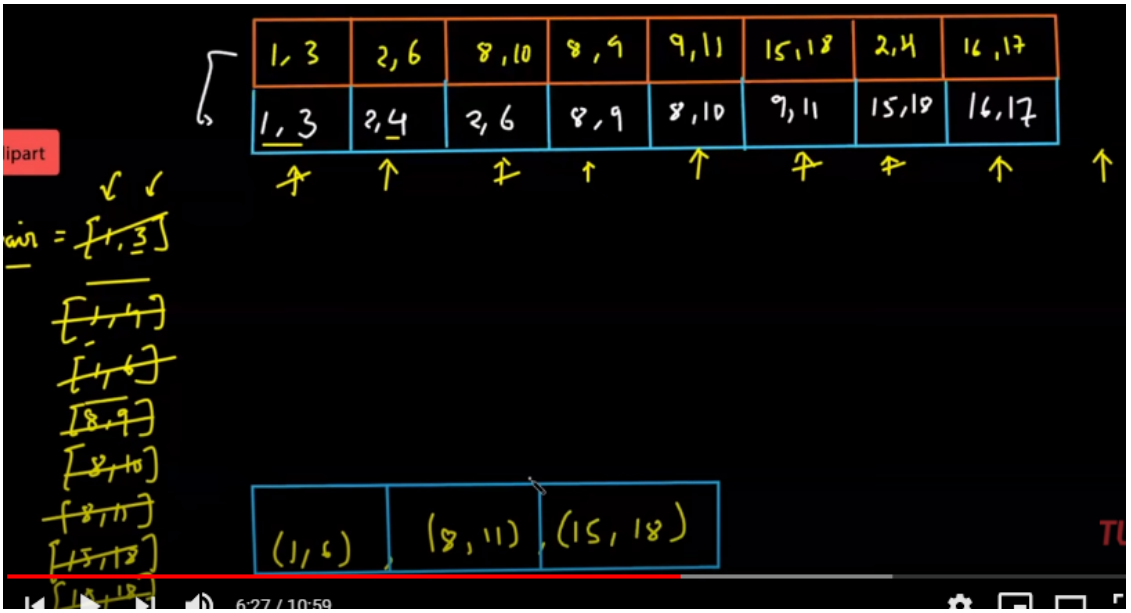
4. **Kadane's Algorithm**
   https://www.youtube.com/watch?v=w_KEocd__20&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=5
   Brute force: using three loops i j k and iterating from i to j to find maximum subarray with time as $O(N^3)$
   Better: using two loops as adding the sum of j element in the subarray will eliminate the need the 3rd loop with time as $O(N^2)$ sum+=arr[j]
   Optimal: more linear approach using Kadane algorithm in one pass with time as O(N) and space as O(1)



Code:

```
sum=0
maxi=nums[0]

for i in nums:
   #calculating sum
   sum+=i

   #storing maximum between sum and maxi
   maxi=max(maxi,sum)

   #if sum is <0 then no use and reset it to 0
   if sum<0:
      sum=0

#maximum will be answer
return maxi
```

5. **Merge Overlapping Subintervals**
   https://www.youtube.com/watch?v=2JzRBPFYbKE&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=6
   Ask if given is sorted
   Brute force: sort and iterate linearly using one more data structure like list to store answer with time as O(NlogN) + O(N^2) since we are comparing every interval with answer and merging.
   Optimal: time as O(NlogN) + O(N) and space as O(N) for answer.



   Code:

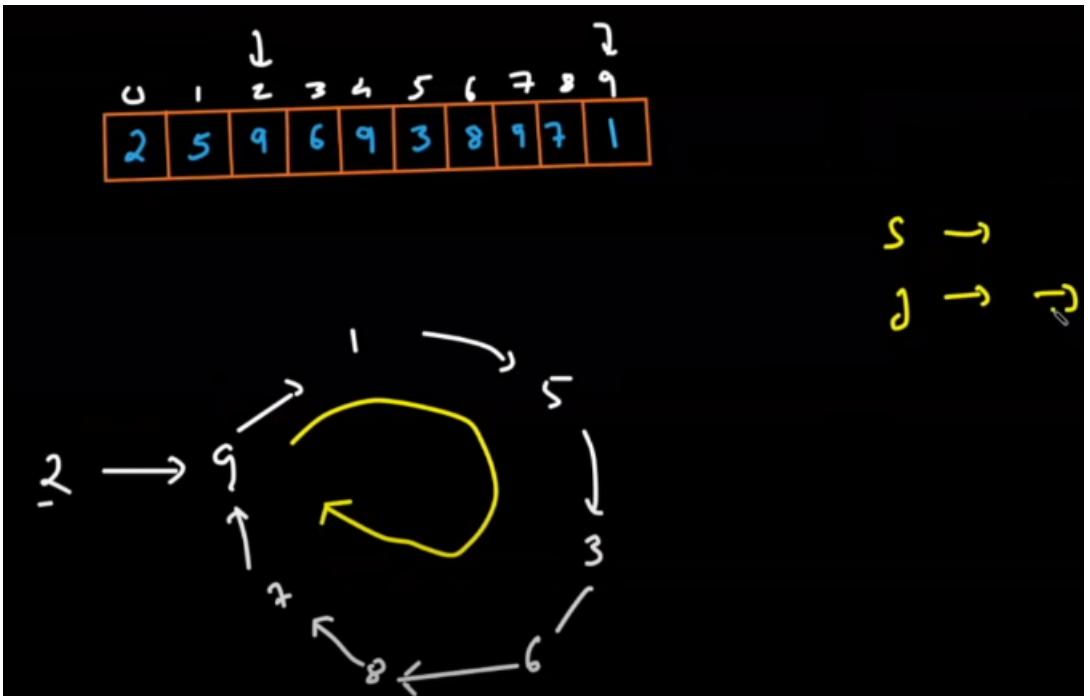
```
ans=[]

#edge case
if len(intervals)==0:
    return ans

intervals.sort()

#[[1,3],[2,6],[15,18],[8,10]]

#[1,3]
temp = intervals[0]

#storing first interval
ans.append(temp)

for item in intervals:

    # 2 < 3
    if item[0]<=temp[1]:
        #do [1,3]->[1,6]
        temp[1]=max(item[1],temp[1])
```

```
        else:
            # 6 !< 8
            # [8,10] -> new temp
            ans.append(item)
            temp=item

    return ans
```

6. **Find the duplicate in an array of N+1 integers.**
   (Ignore the video quality, as this was the first video which i recorded)
   https://www.youtube.com/watch?v=32Ll35mhWg0&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=1
   Brute force: sort array and check for similar elements arr[i]==arr[i+1] with time as O(NlogN) for sorting and space as O(1)
   Better: using frequency array of size N and incrementing value of its index when a element is found element==index with time as O(N) and space O(N)
   OR using hashset code:

```
hs = set()
    for i in nums:
        if i in hs:
            return i
        hs.add(i)
```

   Optimal: using two pointer Tortoise approach to which will detect cycle with time as O(N) and space as O(1)

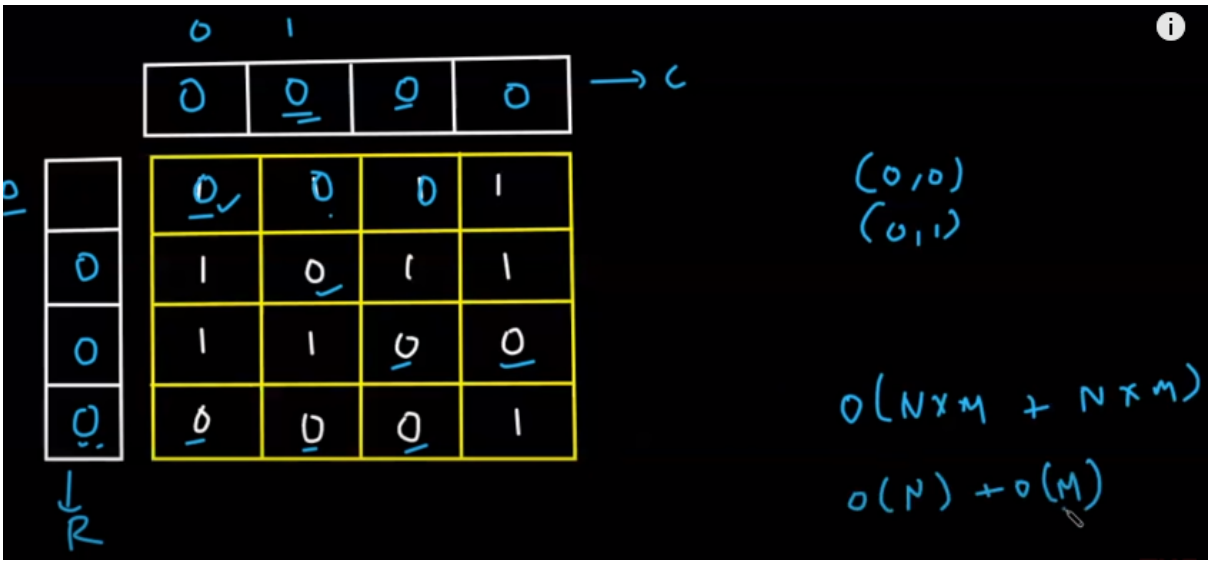


Code:

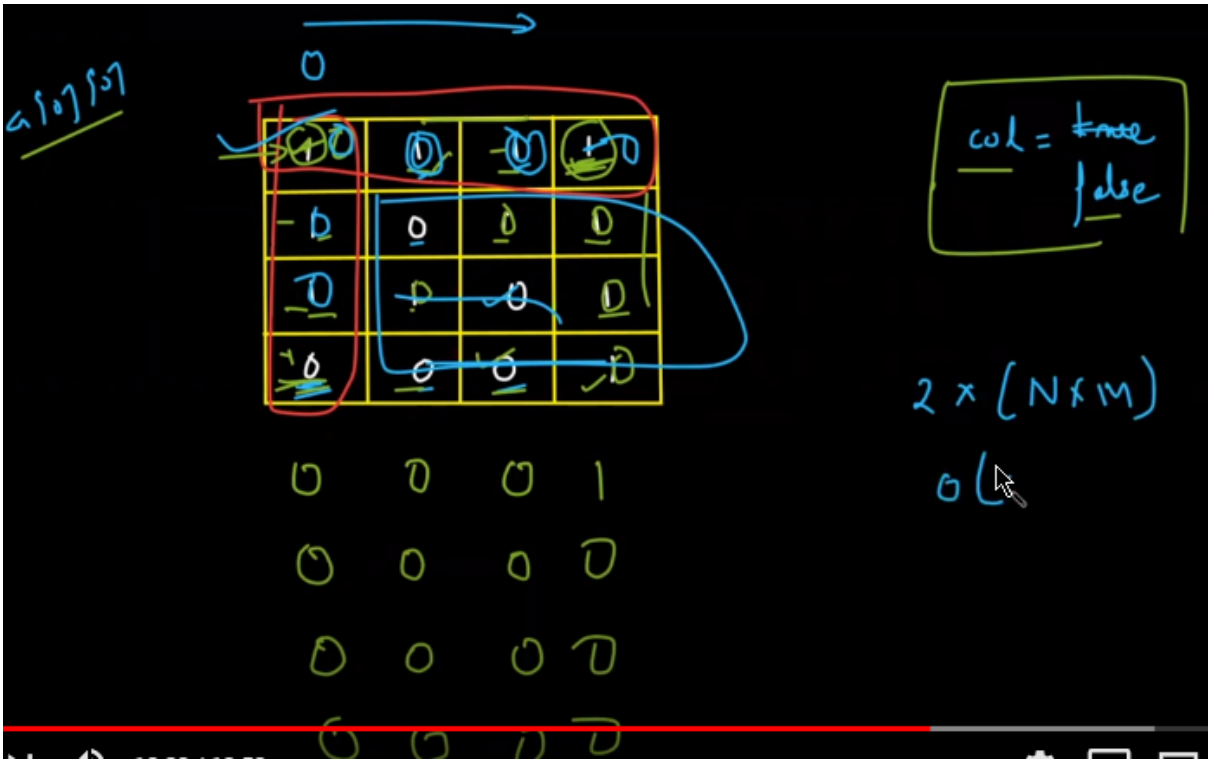
```
slow=fast=nums[0]

    while True:
        slow=nums[slow]
        fast=nums[nums[fast]]
        if slow==fast:
            break
    fast = nums[0]

    while fast!=slow:
        slow=nums[slow]
        fast=nums[fast]
    return fast
```

Day2: (Arrays)

1. **Set Matrix Zeros**
   (https://www.youtube.com/watch?v=M65xBewcqcI&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=7)
   Ask the range of number
   Brute force: if it are positive only >0 then traverse the entire matrix and start marking the column and row [i][j] as -1 with time as O(m*n)+O(m+n) and space as O(N)
   Better: using 2 dummy array and marking 0 in the respective row and column with time as O(N*M + N*M) since traversing 2 times in matrix and space as O(N)+O(M)



Optimal: optime the better solution with time as 2*(N*M) and space as O(1) as no need of dummy array.



2. **Pascal Triangle**
   https://www.youtube.com/watch?v=6FLvhQjZqvM&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=8
   Brute force:
   Better:
   Optimal:
   Code:

```
if n == 0:
    return []
elif n == 1:
    return [[1]]
ans=[[1]]

for i in range(1,n):
    row=[1]
    for j in range(1,i):
        row.append(ans[i-1][j-1]+ans[i-1][j])
    row.append(1)
    ans.append(row)
return ans
```

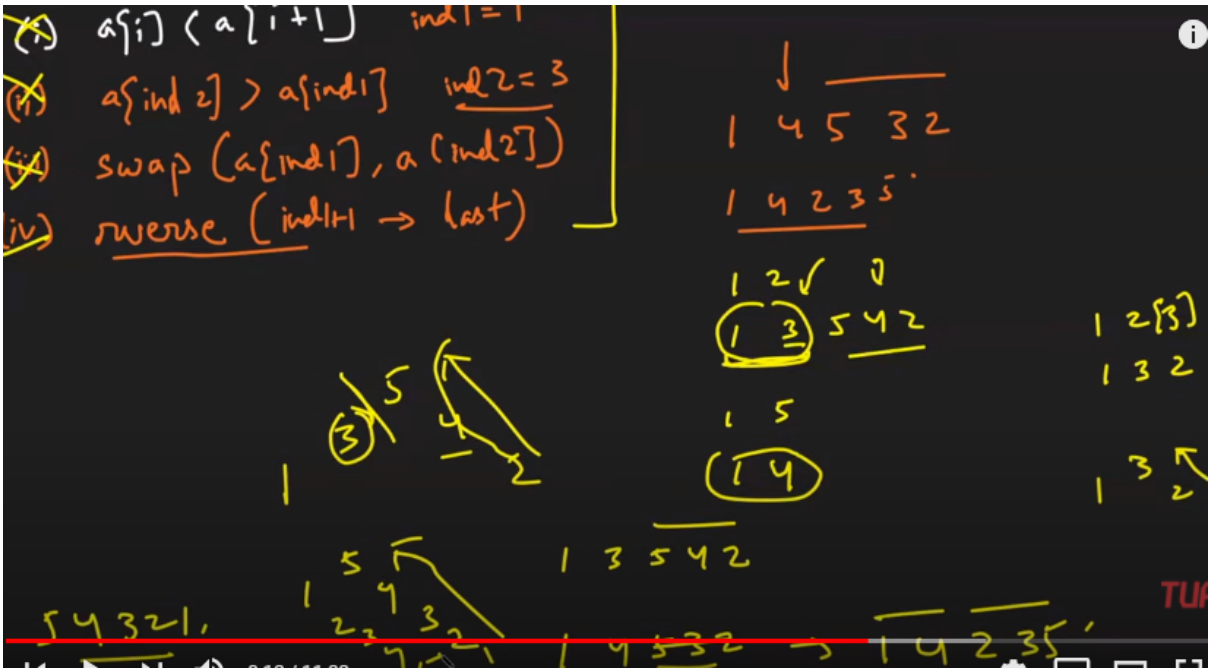
3. **Next Permutation**

https://www.youtube.com/watch?v=LuLCLgMElus&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=9

Brute force: first generate all the possible permutations and then find its next element with time as O(N!*N) since there are n! possible answer for n

Optimal: using following algorithm which state traverse the number or given string from behind and find the index where a[i] < a[i+1] # 1 3 5 4 2 -> 1 4 5 3 2 and then swap this two numbers and then reverse the remaining right side

If we don't find such an index then we skip 3 steps and directly reverse the right side which is the next permutation with O(3N) aka O(N) and space as O(1)



Code:

```
if len(nums)<=1:
        return nums
    n = len(nums)
    i = n - 1
    while i > 0 and nums[i-1] >= nums[i]:
        i -= 1
     # use binary search because it's descending order
    if i > 0:
        left, right = i, n - 1
        while left <= right:
            mid = left + (right - left) // 2
            if nums[i-1] < nums[mid]:
                left = mid + 1
            else:
                right = mid - 1
        nums[i-1], nums[left-1] = nums[left-1], nums[i-1]

    # reverse
    left, right = i, n - 1
```

```
while left < right:
    nums[left], nums[right] = nums[right], nums[left]
    left += 1
    right -= 1
return nums
```

4. **Inversion of Array** (Using Merge Sort)
   COUNT INVERSIONS in an ARRAY | Leetcode | C++ | Java | Brute-Optimal
   Brute force: using two loops to traverse and find i<j and a[i]>a[j] with time as $O(N^2)$ and space as O(N)
   Optimal:

5. **Stock Buy and Sell**
   Best Time to BUY and SELL STOCK | Leetcode | C++ | Java | Brute-Optimal
   Brute force:
   Better:
   Optimal:

6. **Rotate Matrix**
   ROTATE IMAGE | Leetcode | C++ | Java | Brute-Optimal
   Brute force:
   Better:
   Optimal:

Day3: (Arrays/maths)

1. **Search in a 2D matrix**
   Search in 2D-MATRIX | Leetcode | GFG | C++ | Java | Brute-Better-Better-Opti…

2. **Pow(X,n)**
   POW(x,n) | Binary Exponentiation | Leetcode

3. **Majority Element (>N/2 times)**
   https://www.youtube.com/watch?v=AoX3BPWNnoE&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=15

4. **Majority Element (>N/3 times)**
   https://www.youtube.com/watch?v=yDbkQd9t2ig&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=16

5. **Grid Unique Paths**
   https://www.youtube.com/watch?v=t_f0nwwdg5o&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=17

6. **Reverse Pairs (Leetcode)**

https://www.youtube.com/watch?v=S6rsAlj_iB4&list=PLgUwDviBIf0rPG3Ictpu74YWBQ1CaBkm2&index=18

7. **Go through Puzzles from GFG** (Search on own)

Day4: (Hashing)

1. **2 Sum problem**
   https://www.youtube.com/watch?v=dRUpbt8vHpo&list=PLgUwDviBIf0rVwua0kKYIsS_ik_1IyVK_&index=1

2. **4 Sum problem**
   https://www.youtube.com/watch?v=4ggF3tXIAp0&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=20

3. **Longest Consecutive Sequence**
   https://www.youtube.com/watch?v=qgizvmgeyUM&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=21

4. **Largest Subarray with 0 sum**
   https://www.youtube.com/watch?v=xmguZ6GbatA&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=22

5. **Count number of subarrays with given XOR**(this clears a lot of problems)
   https://www.youtube.com/watch?v=lO9R5CaGRPY&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=23

6. **Longest substring without repeat**
   https://www.youtube.com/watch?v=qtVh-XEpsJo&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=25

Day5: (LinkedList)

1. **Reverse a LinkedList**
   https://www.youtube.com/watch?v=iRtLEoL-r-g&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=26

2. **Find middle of LinkedList**
   https://www.youtube.com/watch?v=sGdwSH8RK-o&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=27

3. **Merge two sorted Linked List**
   https://www.youtube.com/watch?v=Xb4slcp1U38&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=28

4. **Remove N-th node from back of LinkedList**
   https://www.youtube.com/watch?v=Lhu3MsXZy-Q&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=29

5. **Delete a given Node when a node is given. (0(1) solution)**

https://www.youtube.com/watch?v=icnp4FJdZ_c&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=30

6. **Add two numbers as LinkedList**
   https://www.youtube.com/watch?v=LBVsXSMOIk4&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=31

Day6:

1. **Find intersection point of Y LinkedList**
   https://www.youtube.com/watch?v=u4FWXfgS8jw&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=32

2. **Detect a cycle in Linked List**
   https://www.youtube.com/watch?v=354J83hX7RI&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=33

3. **Reverse a LinkedList in groups of size k**.
   https://www.youtube.com/watch?v=Of0HPkk3JgI&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=33

4. **Check if a LinkedList is palindrome or not.**
   https://www.youtube.com/watch?v=-DtNInqFUXs&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=35

5. **Find the starting point of the Loop of LinkedList**
   https://www.youtube.com/watch?v=QfbOhn0WZ88&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=36

6. **Flattening of a LinkedList**
   https://www.youtube.com/watch?v=ysytSSXpAI0&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=37

7. **Rotate a LinkedList**
   https://www.youtube.com/watch?v=9VPm6nEbVPA&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=38

Day7: (2-pointer)

1. **Clone a Linked List with random and next pointer**
   https://www.youtube.com/watch?v=VNf6VynfpdM&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=39

2. **3 sum**
   https://www.youtube.com/watch?v=onLoX6Nhvmg&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=40

3. **Trapping rainwater**

https://www.youtube.com/watch?v=m18Hntz4go8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=41

4. **Remove Duplicate from Sorted array**
   https://www.youtube.com/watch?v=Fm_p9lJ4Z_8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=42

5. **Max consecutive ones**
   https://www.youtube.com/watch?v=Mo33MjjMlyA&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=43

Day8: (Greedy)

1. **N meeting in one room**
   https://www.youtube.com/watch?v=II6ziNnub1Q&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=44

2. **Minimum number of platforms required for a railway**
   https://www.youtube.com/watch?v=dxVcMDI7vyI&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=45

3. **Job sequencing Problem**
   https://www.youtube.com/watch?v=LjPx4wQaRIs&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=46

4. **Fractional Knapsack Problem**
   https://www.youtube.com/watch?v=F_DDzYnxO14&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=48

5. **Greedy algorithm to find minimum number of coins**
   https://www.youtube.com/watch?v=mVg9CfJvayM&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=47

6. **Activity Selection (it is same as N meeting in one room)**
   https://www.youtube.com/watch?v=II6ziNnub1Q&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=44

Day9 (Recursion):

1. **Subset Sums**
   https://www.youtube.com/watch?v=rYkfBRtMJr8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=52

2. **Subset-II**
   https://www.youtube.com/watch?v=RIn3gOkbhQE&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=53

3. **Combination sum-1**
   https://www.youtube.com/watch?v=OyZFFqQtu98&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=49

4. **Combination sum-2**
   https://www.youtube.com/watch?v=G1fRTGRxXU8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=50

5. **Palindrome Partitioning**
   https://www.youtube.com/watch?v=WBgsABoCIE0&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=51

6. **K-th permutation Sequence**
   https://www.youtube.com/watch?v=wT7gcXLYoao&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=55

Day10: (Recursion and Backtracking)

1. **Print all Permutations of a string/array**
   https://www.youtube.com/watch?v=f2ic2Rsc9pU&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=52

2. **N queens Problem**
   https://www.youtube.com/watch?v=i05Ju7AftcM&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=57

3. **Sudoku Solver**
   https://www.youtube.com/watch?v=FWAlf_EVUKE&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=58

4. **M coloring Problem**
   https://www.youtube.com/watch?v=wuVwUK25Rfc&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=59

5. **Rat in a Maze**
   https://www.youtube.com/watch?v=bLGZhJlt4y0&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=60

6. **Word Break (print all ways)** (Will be covered later in DP series)

Day11: **(Binary Search)**

1. **N-th root of an integer (use binary search) (square root, cube root, ..)**
   https://www.youtube.com/watch?v=WjpswYrS2nY&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=61

2. **Matrix Median**
   https://www.youtube.com/watch?v=63fPPOdIr2c&list=PLgUwDviBIf0p4ozDR_kJJkO

Nnb1wdx2Ma&index=62

3. **Find the element that appears once in sorted array, and rest element appears twice (Binary search)**
   https://www.youtube.com/watch?v=PzszoiY5XMQ&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=63

4. **Search element in a sorted and rotated array/ find pivot where it is rotated**
   https://www.youtube.com/watch?v=r3pMQ8-Ad5s&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=64

5. **Median of 2 sorted arrays**
   https://www.youtube.com/watch?v=NTop3VTjmxk&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=65

6. **K-th element of two sorted arrays**
   https://www.youtube.com/watch?v=nv7F4PiLUzo&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=66

7. **Allocate Minimum Number of Pages**
   https://www.youtube.com/watch?v=gYmWHvRHu-s&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=69

8. **Aggressive Cows**
   https://www.youtube.com/watch?v=wSOfYesTBRk&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=70

Day12: (Bits) (Optional, very rare topic in interviews, but if you have time left, someone might ask)

1. Check if a number if a power of 2 or not in O(1)
2. Count total set bits
3. Divide Integers without / operator
4. **Power Set (this is very important)**
   https://www.youtube.com/watch?v=b7AYbpM5YrE&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=67
5. Find MSB in o(1)
6. Find square of a number without using multiplication or division operators.

Day13: (Stack and Queue)

1. **Implement Stack Using Arrays**
   https://www.youtube.com/watch?v=GYptUgnIM_I&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=68

2. **Implement Queue Using Arrays**
   https://www.youtube.com/watch?v=M6GnoUDpqEE&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=72

3. **Implement Stack using Queue (using single queue)**
   https://www.youtube.com/watch?v=jDZQKzEtbYQ&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=74

4. **Implement Queue using Stack (0(1) amortised method)**
   https://www.youtube.com/watch?v=3Et9MrMc02A&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=75

5. **Check for balanced parentheses**
   https://www.youtube.com/watch?v=wkDfsKijrZ8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=74

6. **Next Greater Element**
   https://www.youtube.com/watch?v=Du881K7Jtk8&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=75

Day14:

1. **Next Smaller Element**
   Similar to previous question next greater element, just do pop the greater elements out ..

2. **LRU cache (vvvv. imp)**
   https://www.youtube.com/watch?v=xDEuM5qa0zg&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=77

   Clean code:
   https://www.youtube.com/watch?v=Xc4sICC8m4M&list=PLgUwDviBIf0p4ozDR_kJJkONnb1wdx2Ma&index=78

3. **LFU Cache** (Supe

4. Largest rectangle in histogram
5. Sliding Window maximum
6. Implement Min Stack
7. Rotten Orange (Using BFS)

Day15: (String)

1. Reverse Words in a String
2. Longest Palindrome in a string
3. Roman Number to Integer and vice versa
4. Implement ATOI/STRSTR
5. Longest Common Prefix
6. Rabin Karp

Day16: (String)

1. Prefix Function/Z-Function
2. KMP algo

3. Minimum characters needed to be inserted in the beginning to make it palindromic.
4. Check for Anagrams
5. Count and Say
6. Compare version numbers

Day17: (Binary Tree)

1. Inorder Traversal (with recursion and without recursion)
2. Preorder Traversal (with recursion and without recursion)
3. Postorder Traversal (with recursion and without recursion)
4. LeftView Of Binary Tree
5. Bottom View of Binary Tree
6. Top View of Binary Tree

Day18: (Binary Tree)

1. Level order Traversal / Level order traversal in spiral form
2. Height of a Binary Tree
3. Diameter of Binary Tree
4. Check if Binary tree is height balanced or not
5. LCA in Binary Tree
6. Check if two trees are identical or not

Day 19: (Binary Tree)

1. Maximum path sum
2. Construct Binary Tree from inorder and preorder
3. Construct Binary Tree from Inorder and Postorder
4. Symmetric Binary Tree
5. Flatten Binary Tree to LinkedList
6. Check if Binary Tree is mirror of itself or not

Day 20: (Binary Search Tree)

1. Populate Next Right pointers of Tree
2. Search given Key in BST
3. Construct BST from given keys.
4. Check is a BT is BST or not
5. Find LCA of two nodes in BST
6. Find the inorder predecessor/successor of a given Key in BST.

Day21: (BinarySearchTree)

1. Floor and Ceil in a BST
2. Find K-th smallest and K-th largest element in BST (2 different Questions)
3. Find a pair with a given sum in BST
4. BST iterator
5. Size of the largest BST in a Binary Tree
6. Serialize and deserialize Binary Tree

Day22: (Mixed Questions)

1. Binary Tree to Double Linked List
2. Find median in a stream of running integers.

3. K-th largest element in a stream.
4. Distinct numbers in Window.
5. K-th largest element in an unsorted array.
6. Flood-fill Algorithm

Day23: (Graph)
https://www.youtube.com/watch?v=YTtpfjGlH2M&list=PLgUwDviBIf0rGEWe64KWas0Nryn7SCRWw

1. Clone a graph (Not that easy as it looks)
2. DFS
3. BFS
4. Detect A cycle in Undirected Graph/Directed Graph
5. Topo Sort
6. Number of islands (Do in Grid and Graph both)
7. Bipartite Check

Day24: (Graph)
https://www.youtube.com/watch?v=YTtpfjGlH2M&list=PLgUwDviBIf0rGEWe64KWas0Nryn7SCRWw

1. SCC(using KosaRaju's algo)
2. Djisktra's Algorithm
3. Bellman Ford Algo
4. Floyd Warshall Algorithm
5. MST using Prim's Algo
6. MST using Kruskal's Algo

Day25: (Dynamic Programming)

1. Max Product Subarray
2. Longest Increasing Subsequence
3. Longest Common Subsequence
4. 0-1 Knapsack
5. Edit Distance
6. Maximum sum increasing subsequence
7. Matrix Chain Multiplication

Day26: (DP)

1. Maximum sum path in matrix, (count paths, and similar type do, also backtrack to find the maximum path)
2. Coin change
3. Subset Sum
4. Rod Cutting
5. Egg Dropping
6. Word Break
7. Palindrome Partitioning (MCM Variation)
8. Maximum profit in Job scheduling

Day27:

1. Revise OS notes that you would have made during your sem
2. If not made notes, spend 2 or 3 days and make notes from Knowledge Gate.

Day28:

1. Revise DBMS notes that you would have made during your semesters.
2. If not made notes, spend 2 or 3 days and make notes from Knowledge Gate.

Day29:

1. Revise CN notes, that you would have made during your sem.
2. If not made notes, spend 2 or 3 days and make notes from Knowledge Gate.

Day30:

1. Make a note of how will your represent your projects, and prepare all questions related to tech which you have used in your projects. Prepare a note which you can say for 3-10 minutes when he asks you that say something about the project.

Hurrah!! You are ready for your placement after a month of hard-work without a cheat day.

Notes for projects:

Tech Stack info:

Bed Tracking App

-
-

Real Time Chat App

-
-

Audio Streaming App

-
-