Need to complete all the topics and then see this for verification.

C : (use cases for all)

1. Compilation steps.

2. Linking and loading (elf).

3. Memory layout (RAM).

4.What is segmentation fault, why that will come ? How it is generated from OS point of view ?

5. Core dump analysis

6. GDB debugging (pointer address debuugging commands, back trace analysis).

7. Bitwise

8. storage classes

9. structrue/union (padding, packing, bit fields, Padding with CPU level execution under standing)

10. function pointers (function pointer increamenting analysis).

11. static,external and volatile varibales

12. Local static and global static.

13. Arrays and strings

14. implamentation of predefined functions (using both arrays and pointers):

i. strings lib funs

ii. memcopy, memmove

iii. Copy command

iv. malloc, calloc

15. Differences :

i. memcpy, memmove

ii. malloc, calloc

iii. Macro and inline

iv. static and dynamic libraries.

16. How the address of the dynamic library will be available to the executing procee. (linking of dynamic lib address to the process).

17. How to define typedef for function pointer.

18. Arrays of pointers.

19. pointer to array.

20. array of function pointer.

21. Null pointer, dangling pointer.

22. memory leak => with debugging technics (val grind and other)

23. Size of variable of other things without useing size of operater

24. typedef, macro, enum

25. Printf and scanf implementation and return values.

26. If (0), if (-1)

27. precedence of operators (, . + - ............)

28. Recursion.

29. Double pointers, memory allocation for doublke pointer.

30. Magic number pgrogram (sum of row elements= column elements and sum of one diagonal elements = sum of other diagonal elements).

31. Spiral way of printing of matrix.

32. Matrix multiplication.

33. Call by value and call by reference.

34. Prime number, febanocc series, amstrong number

35. Palindrome (number, string).

36. Factorial

37. Remove duplicates in number and strings (simple way )

38. Implementation of atoi and atof

39. File handling functions.

40. Stack frame.

DS : (Real time examples)

1. Time complexity, space complexity

2. Why and how the Big O notation came ?

3. self reference structure.

4. Single Linked list, double linlked list, circular linked list

5. stack and queue

6. Sorting alogorithms (with time and space complexity, when to use which algorithm based on the algorithm speciality)

7. Searching technics (linear, binary)

8. trees

9. Hashing. (we can use hashing use strings).

10. Colision avioding technics.

DS programs :

1. Add node at starting, ending, middle, assending order.

2. Delete node

3. Delete dupilcate node

4. Reverse list

5. middle node finding program using two pointers (slow pointer and fast pointer).

6. Circular dupliacte deletion with and without the number of nodes count.

7. Print the address the of the matched node address of the node while matching the two linked list.

8. Two or more linked lists merging. (with and without duplicates).

9. Find the 4th or 5th node from last in the linked list.

10. stack smashing and how to idenfy that.

Linux system programming :

1. Static and dynamic libraries.

2. Process and Threads management.

i. fork

ii. vfork

iii. Exec, execl family

iv. orphon, zomby process, init process.

v. exit, exit, wait, wait pid.

vi. socket programming

vii. Ps , ps -e, ps -ef and other commands implementation.

viii. System( ) api to execute the command from the program.

ix. Top command implementation.

x. CPU scheduling.

xi. process states

xii. Process priorities (nice, renice)

xiii. Context switching

xiv. Starvation time.

xv. CPU bounded time.

Xvi. Scheduler and dispature.

3. IPC

i. Pipe, named pipe (FIFO), message queue, shared memory.

ii. Memory barriers.

iii. Critical section.

4. Locking mechanisms

i. Symaphore,mutex, counting semaphore, spin lock.

ii. Dead lock, why that will come and how can we avoid.

iii. Semaphore and mutex difference.

iv. Ptherad compilation (link the library while building the pthread)

v. Pthread debugging using GDB.

5. File management

i. Static linking file and dynamic linking difference

ii. Soft link and hard link difference

iii. Types of file.

iv. mount command implementation.

v. directories and the content of each directory in the rootfs.

6. signals

i. Types of signals

ii. difference between signal and interrupt.

7. Network managements.

i. Netlink sockets (kernel spcae to user space signal).

ii. sockets.

iii. Socket address

iv. messagner application implementation between two computers connected via network.

8. Even odd printing from using two threads (and process).

9. How the system call will work. And own implementation of system call.

10. system call and library call difference.

11. malloc implementation (This will clear how the memory will be allocated for user space process).

Core Kernel :

1. Linux device model.

2. monolithic and micro kernel differences.

3. Semaphore, mutex, spin locks implementation. (synachronization technics).

4. Top half and bottom half (Interrupt techoincs).

5. VFS layer.

6. Char driver frame work

i. cdev related structures.

ii. cdev related function implementation (cdev\_init and cdev\_add)

7. IOCTL

8. sub system frame works (I2C, SPI, USB,....)

9. Timers in kernel (timers, jiffes, .........)

10. How the Memory will be allocated for user space and kernel space.

11. Overall booting process. (embedded)

12. device tree and device tree overlay.

13. Make file under standing.

14. tool chain and cross compiler.

15. Bootloader, Kernal image and root file system images sizes.

16. kernel image header analysis.

17. Kernel image building with attaching the address of the RAM to load at run time.

18. kernel command line arguments (boot and other arguments).

19. printk implementation.

20. dmesg implementation.

21. memmap and ioremap difference.

22. kmalloc implementation. Kzalloc, vmalloc.

23. there will be a memory limlt for continues physical memory via kmalloc, what is the limit.

24. kmalloc and vmalloc differences.

25. DMA

26. cache memory.

27. TLB (translation look aside buffer).

28. Page fault and how to handl that.

29. cache write throgh and write back.

30. dirty cache.

31. fragmentation (internal and external).

32. difference between segmentation and paging.

33. Link between segmentation and segmentation fault.

34. Bootloader and kernel starting addresses and they are defined, how and who are defined.

Drivers :

1. Basic driver.

2. Staic and dynamic drivers.

3. Major and minor number

4. probe and remove functions.

5. insmod and modprobe.

6. dynamic module compilation and linking.

7. symbol table.

8. device mode creation.

9. file operations structures

10. driver integration steps.

11. \_\_init and \_\_exit functions

12. module\_param

13. export symbols.

Android :

1. Android architecture.

2. Android booting process.

3. HAL, AIDL and HIDL.

4. Difference between linux and android kernel.

5. Each level of booting process (zygote, system servers...........)

6. EDK 2 and LK differences.

7. TI and QC chipset differences.

8. msm and sdm chipset differences.

9. HAL and HIDL starting Android versions names.

10. bootimage, system, vendor, user data and other images information.

11. Trible

12. SELinux

13. File systems used by android.

14. Encryption.

15. secure boot.

16.Android Manifest File,activities

Debugging :

1. GDB

2. KDB

3. KGDB.

4. printk implementation.

5. dmesg implementation.

6. dynamic debugging.

7. How dymainc debugging will work implementation.

8. JTAG.

9. Trace32.

BSP:

1. What is BSP

2. Bringup of Peripherals on development board

Puzzles :

1. Top 10 interview puzzles.

<http://www.crazyforcode.com/top-10-interview-puzzles/>

2. shekunthala puzzles

3. geeksforgeeks

<https://www.geeksforgeeks.org/puzzle-21-3-ants-and-triangle/>

Reference link :

<https://en.wikipedia.org/wiki/Translation_lookaside_buffer>

[https://www.softwaretestinghelp.com/android-interview-questions/amp/#aoh=15806431707778&amp\_ct=1580643759986&referrer=https%3A%2F%2Fwww.google.com&amp\_tf=From%20%251%24s](https://www.softwaretestinghelp.com/android-interview-questions/amp/" \l "aoh=15806431707778&amp_ct=1580643759986&referrer=https%3A%2F%2Fwww.google.com&amp_tf=From %251$s)