LINUX INTERNALS :

a) Networking :

1. List out the difference between TCP/IP & UDP
2. System calls used in TCP server programming
3. Which is reliable and secure TCP and UDP?
4. What do you know about broadband connection and routers?
5. What is the OSI model and what are its benefits?

b) Threads :

1. What is Mutex? Explain with an example program.
2. What is a Semaphore? Explain with an example program.
3. What is deadlock? Write a program to demonstrate deadlock
4. What is a thread? Why do we need threads?

c) System Calls:

1. What are the various system calls that you know?
2. What is "kernel?" Explain the difference between privilege mode and user mode.
3. What is a “Monolithic” and “Micro” kernels?
4. How real-time systems (RTS) design is connected with kernel designs? Explain.
5. What is the connection between Real-time OS (RTOS) & Embedded OS (EOS)? Explain.
6. What are the type of interrupts? Explain the differences.
7. What is an exception in a system? How the OS handle that? Explain
8. What is a system call? How is it implemented in Linux?
9. Is it always necessary to implement system calls as soft interrupts?

MICROCONTROLLERS :

a) Communication protocols :

1. Explain about CAN protocol and types of frames in CAN.
2. Which are the receiver side error and transmitter side error?
3. Explain the UART and SPI in detail.
4. Explain the I2C in detail.
5. What is the frame? explain
6. How data is transferred in CAN
7. In I2C how data is transferred? I2Cis full duplex or half?
8. What are the things I need to configure in UART?

b) Embedded Systems :

1. What is ADC? How it converts analog to digital. How to write a code to do it.
2. Difference b/w microcontroller and microprocessor
3. How do you configure ADC?
4. How does a timer work?
5. Significance of different ADC's

c) Interrupts :

1. What is interrupt latency?
2. What is an interrupt? What are the types of interrupt?

d) PWM :

1. What is duty-cycle
2. What is PDM. What are the different methods to use PDM?

e) Basic Electronics :

1. What is MOSFET? explain it

2. What is ra ectifier? Explain it
3. Do you know about timer IC
4. What is OP-AMP?
5. What is gain
6. Tell the different configurations for OP-AMP
7. What things do I need to consider while designing a board?
8. how to design an SMPS.
9. . Is there any requirement for an inverter in SMPS?
10. How can I remove the noise?
11. Explain about clipper and clamper circuit
12. Explain about transistors. how can I use a transistor as a switch? (what will be the circuit)

DATA STRUCTURES :

1. Sum of the students marks using SLL
2. What is a linked list, tree?
3. What is hash table?
4. What is the difference between algorithms and data structure?
5. WAP is used to create and delete a node in an SLL.
6. What are the different types of linked lists**?**
7. WAP to delete a node somewhere in the middle without using any temp variable
8. List the difference between Arrays and linked lists
9. Insert & delete element in a single linked list(insert_at_first, insert_last, insert_after, insert_before, delete_first)
10. WAP to insert 5 elements in DLL, and SLL.
11. WAP to insert the data in between 2 nodes in DLL (insert_after, insert_before)

ADVANCED C :

a) Basic Refresher :

1. List out the different types of data types in C
2. Can the size of datatypes be changed?
3. char var=600; is the following statement correct?
4. List out the difference between while and do-while
5. What will be the output of the program? If there is any error, what is the error and what is the reason
6. Difference between logical and bit-wise operator
7. Explain about type conversion(implicit and explicit) Why does implicit type conversion happen? give an example.
8.
   ```
   #include<stdio.h>
   int main()
   {
       int i=32, j=0x20, k, l, m;
       k=i|j;
       l=i&j;
       m=k^l;
       printf("%d, %d, %d, %d, %d\n", i, j, k, l, m);
       return 0;
   }
   ```
9. What are qualifiers? explain different qualifiers in C.
10. Can I use const and volatile together?
11. Where the memory is allocated for the string. Can we change the string?

    ```
    #include<stdio.h>
    ```

```
int main()
{
    char *str;
    str = "%s";
    printf(str, "K\n");
    return 0;
}
```

12. What will be the output

```
#include<stdio.h>
int main()
{

    for(char a = 40; a < 300; a++)
    {
        printf("a = %d\n", a);
    }
    return 0;
}
```

13. How the negative numbers are stored?
14. const int a=10; static int b=20; a=50; b=60; what will be the output , Explain.
15. What are format specifiers?
16. What is a variable declaration, definition and initialization?
17. char a=555; print("%d",a); What will be printed? Why?
18. int x=0xaabb. how many bytes are allocated?
19. WAP takes two arrays and combines it into a single array

    Ex:- arr1{10,20,30}

    arr2{40,50,60}

    O/p:- { 10,20,30,40,50,60}

20. WAP to divide a num without using the / operator.

b) Functions and pointers :

1. What is a pointer to a constant? Give an example. How can I change the value in the pointer?
2. What is function declaration and function definition?
3. What is a null pointer?
4. What is the size of a null pointer?
5. What is recursion?
6. WAP to check the system is little and big-endian
7. What will be the output

   ```
   main(){
       int num=10;
       int *ptr=&num;
       *++ptr;
       print("%d",ptr);
      *ptr++;
      print("%d",ptr);
   }
   ```

8. int x=10; int *ptr=&x;

   *ptr=ptr +1; why this gives an error

   ptr=ptr+1 What happens here?

9.
   ```
   #include <stdio.h>
   void modify(int *ptr);
   void cptr(char *pt);
   int main()
   {
       int x=10;
   ```

```
    printf("value of x is %d\n",x);
    printf("%p",x);
    modify(&x);
    printf("value of x is %d\n",x);
    cptr(&x);
    printf("%p",x);
    return 0;
}
void modify(int *ptr)
{
    ptr=ptr+1;
}
void cptr(char *pt)
{
    pt=pt+1;
}
```

What is the output of the code?

10. What is int const *ptr give an example and explain.
11. List out the difference b/w call by value and call by reference
12. What is a wild pointer and a dangling pointer?
13. Write the syntax for the pointer to a constant value
14. List out the difference between a null pointer and void pointer
15. What is a static function?
16. What is a pointer? what are the different types of pointers?

c) Memory segments and storage classes :

1. What are the different types of storage classes in C?
2. What is DMA?
3. Difference between malloc() and calloc().
4. Explain the different memory segments in C

5. How heap memory is used?
6. what is the use of register? What is the difference between auto and register?
7. How extern variable be linked?
8. How to access the static variable outside the file?
9. Explain about global constant and local constant. Give an example.
10. What is the use of code segment?
11. What is the reason for the segmentation fault? (give an example)
12. What is the scope of an auto variable, static variable, and locally and globally? What will be the scope of that variable?
13. WAP to allocate memory of 8 bytes and stores some data in it.
14. Why can I not read a register variable?
15. Where memory is allocated for the const variable, if it is in stack we can overwrite it?
16. What value is stored in global and local static if it is not initialized?
17. List out the difference between malloc and calloc
18. Give a real-time application for volatile.
19. What is constant? where the memory is allocated?
20.

```
int num=20;
   int main()
   {
     static int num=10;
     print("%d", num);
     int x=50;
     {
       int x=60;
        print("%d", y);
    }
```

What will be the output? what happens if x is a register variable? what if it is a static variable?

d) Strings :

1. WAP to reverse string, find length
2. Implement strtok, strcmp

e) Advanced pointers and functions :

1. List out the difference between an array of pointers and pointer to an array.
2. Tell the application of an array of pointer
3. int var=0x12345678; Explain your approach to extract 34 and write the code for that.
4. What is a function pointer? write the syntax.
5. Take a double pointer and print both address and value.
6. Declare a 2d array statically and dynamically
7. List out the difference between static memory and dynamic array
8. Call a function using the function pointer
9. What is a multi-level pointer?
10. How to use multi-level pointer and what is the advantage of using it?
11. What is a callback function?
12. what is an array of pointers and a pointer to an array? Declare both. what happens if i increment them by 1?
13. WAP to swap two numbers using pointers and bitwise operator.

f) User-Defined Datatypes :

1. Create a structure other than student (elements of different data types) access the elements using pointers and print them. before running find the errors. If there is an error, what is the reason? how to correct them.
2. What is the use of typedef?
3. What is structure padding?
4. Tell the difference in memory allocation in structure and union.
5. List out the disadvantages of using a union.
6. Use of typedef. (Explain with code)
7. What is structure? Create an array of structures.
8. How to remove structure padding?
9. Why padding is done?
10. Create a structure with 2 elements id and name. what will be the size of the structure?
11. Create a structure array and store details of 5 employees and display it in a separate function
12. I have int char and double then what will be the size of the structure?
13. What is bit-field?
14. What is an enum? define an enum
15. What will be the default values stored in the enum?

g) Preprocessing :

1. What is #define and #ifdefine in macro. Give an example of a macro.
2. Write a macro to set a bit.
3. What is conditional compilation?
4. Explain about compilation stages.
5. Tell the command to get the intermediate files in compilation.
6. List out the difference between macro and function

7. List out the difference between structure and union
8. List out the difference between macro and inline function
9. Does context switching happen in inline function?
10. Create a macro to perform an addition operation
11. What is #ifdef and #endif. that is its usage (give a proper example)
12. How to use pragma?