Questions:

To the point questions:

1. what do you understand by term “memory leak”, explain with the help of sample code;

Text

Description automatically generated

**How to avoid Memory Leak?**

* Instead of managing memory manually, try to use smart pointers where applicable.
* use [std::string](https://www.geeksforgeeks.org/stdstring-class-in-c/) instead of char \*. The std::string class handles all memory management internally, and it’s fast and well-optimized.
* Never use a raw pointer unless it’s to interface with an older lib.
* The best way to avoid memory leaks in C++ is to have as few new/delete calls at the program level as possible – ideally NONE. Anything that requires dynamic memory should be buried inside an RAII object that releases the memory when it goes out of scope. RAII allocate memory in constructor and release it in destructor, so that memory is guaranteed to be deallocated when the variable leave the current scope.
* Allocate memory by new keyword and deallocate memory by delete keyword and write all code between them.

1. What are the key differences between malloc() and calloc()? Explain with the help of code and memory map.

Graphical user interface, text, application

Description automatically generated

1. What are the main differences between malloc() and “new”? How can you implement two dimensional arrays using new operator in C++?

Text

Description automatically generated

1. How realloc() can be used to resize allocated memory in heap. How programmers can get the same effect with “new” operator.

Graphical user interface, text, application, email

Description automatically generated

1. Why is the socket function listen(int count) so important for servers? Discuss the impact of variable count.

To cater the limited amount of clients and prevent server from DDOS attack

1. Why are joinable threads preferred over detachable threads? In what conditions detachable threads can be used?

Both can free resources. Joinable threads does not free their resources after the parent process finishes while detachable releases their resources. Joinable threads become a zombie after termination until some process calls joinable.

1. why is connection oriented communication preferred for file transfers?

Connection oriented communication such as TCP is preferred for file transfers because it ensures all packets of the file are received.

If any packet is missed, the protocol would know of it and retransmit ensuring successful transfer.

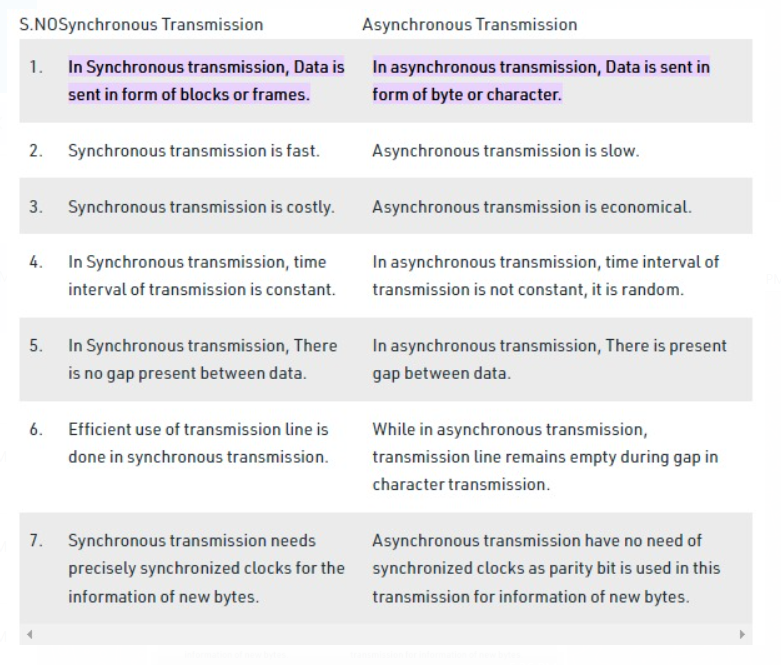
It becomes more important as the file size grows.

1. What are the key benefits of using connectionless protocols for live streaming?

Known

1. Differentiate asynchronous and synchronous socket functions with examples.

**Asynchronous sockets** use multiple threads from the system thread pool to process network connections. One thread is responsible for initiating the sending or receiving of data; other threads complete the connection to the network device and send or receive the data. Example: TCP



1. How socket functions accept() and connecT() are working in collaboration? Show their working with the help of a 3-way handshake.
2. How socket functions send()A and recv() are working in collaboration? How TCP timers, sequence and acknowledge numbers are used behind these functions? Why must programmers make sure their programs have received all sent data?

After your connection is set up, the OS manages the packets entering and leaving your system, the recv() call just reads the packet buffer, and the send() call just queues the packets.The sequence and acknowledgement numbers are part of the TCP header.Those two numbers help the computers to keep track of which data was successfully received, which data was lost, and which data was accidentally sent twice.The first computer sends a packet with data and a sequence number. The second computer acknowledges it by setting the ACK bit and increasing the acknowledgement number by the length of the received data.

OR

recv() is a blocking function so it is placed after the send() function in the code. Every time a message goes from one socket to another, it checks its message buffer for any message received. Before the message buffer can be checked, we would need to establish a connection by defining sockets for both the client and the server. These sockets after instantiation perform a 3-way TCP handshake (SYN, SYN/ACK, ACK). If the request times out for any of these, steps, it restarts its handshake.

1. what is the key difference between File and file descriptor?

You pass "naked" file descriptors to actual Unix calls, such as [read()](http://linux.die.net/man/2/read), [write()](http://linux.die.net/man/2/write) and so on.  
A FILE pointer is a C standard library-level construct, used to represent a file. The FILE wraps the file descriptor, and adds buffering and other features to make I/O easier. You pass FILE pointers to standard C functions such as [fread()](http://linux.die.net/man/3/fread) and [fwrite()](http://linux.die.net/man/3/fwrite).

2. In what cases following socket functions will fail and return some error and how that error can be handled? (8points 2 each)

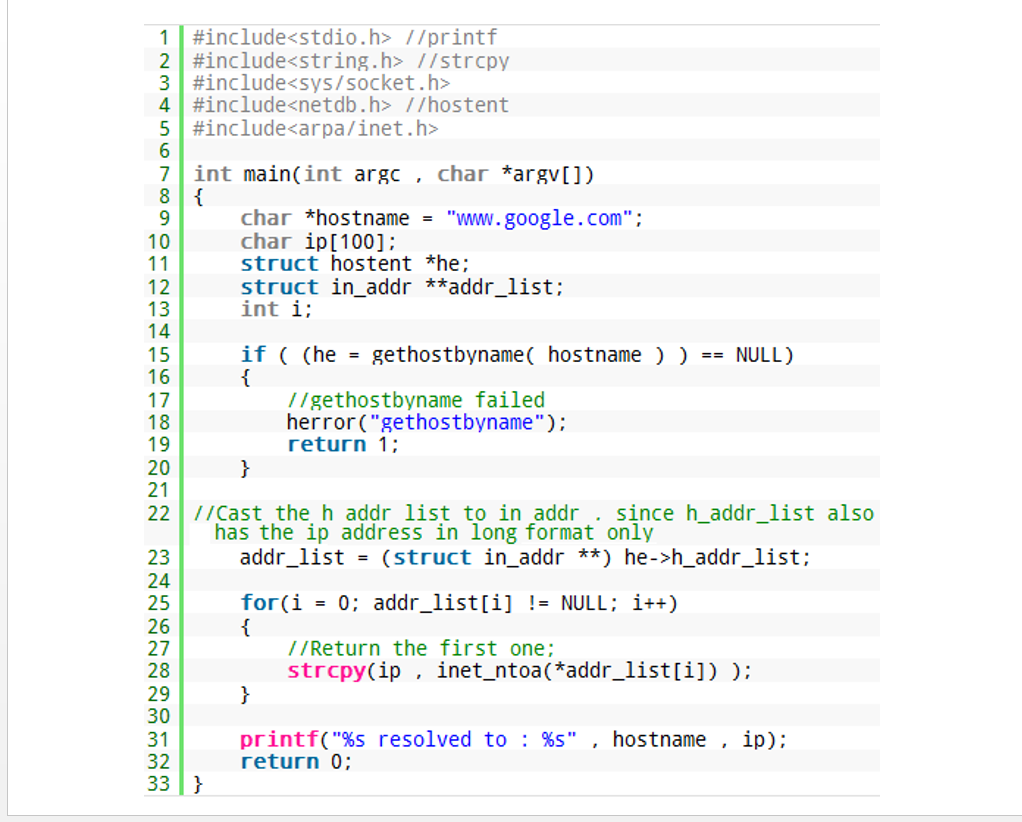
1. int bind(int sid, struct sockaddr \*addrptr, int len)

less than 0 = error

1. int connect(int sid, struct sockaddr \*addrptr, int len)
2. int accept(int sid, struct sockaddr \*addrptr, int len)
3. int sendto(int sid, const char \*buffer, int len, int flag struct sockaddr \*addrptr, int addrptr len)

Question3:

1. Write a program that can find and show all IP addresses against any domain name resolved through DNS.



1. Write a concurrent server that can handle multiple clients simultaneously. Server is just required to send “hello” message.

PART 1:

Question 1: T or F

1. The entire process will be block if any one of its kernel level thread call I/O instruction. [ T ]
2. Kernel I/O needs more memory to process files than buffered I/O. [ ]
3. Memory allocated through malloc() can be released through delete function. [ F ]
4. Kernel I/O function create() calls another kernel function open() to create a file. [ T ]
5. fseek() and lseek() both returns file position after successful completion. [ F ]
6. SOCK\_STREAM is used to represent TCP in socket programming [ T ]
7. vfork() makes a complete copy of the calling process’s address space and allows both the parent and child to execute independently. [ F ]
8. In POSIX threads, a thread by default starts as joinable thread. [T ]

Question 2: MCQs

1. “O\_LARGEFILE” is used to open field larger than \_\_\_\_\_GB in size.

a. 1Gb | b. 2Gb | c. 512MB | d 128MB

1. On a UNIX/Linux system everything is a file; if something is not a file it is a \_\_\_\_\_\_\_\_.

( a. device | b. Directory | c. Process | d. option a and b

1. The mask for permission S\_IRUSR | S\_IXGRP | S\_IRWXO will be\_\_\_\_\_. 0417 | b. 0777 c.| 0666 | d. 07144.
2. Which function is used to change allocated memory size in heap. (a. new | b. inc\_malloc() | c. realloc() d.calloc()) ANS = REALLOC
3. System call that is used to generate clone of the calling process in unix/linux. (a. system() | b. fork() | c. exec() | d. spawn() ANS = FORK
4. which partition is used in UNIX/LINUX for virtual memory. (a. root b. boot c. swap d. /)
5. Joinable thread becomes a zombie after finishing, and won’t disappear until invoke pthread \_\_\_\_\_. (a. join() b. detach() c. exit() d. end() ANS = JOIN
6. In socket programming which function is used to reserve PORT number. (a. accept b. bind c. socket d. listen) ANS = BIND
7. In socket programming which function is used by server to wait for a client request. a. accept b. bind c. socket d. listen ANS = LISTEN

Question 3: FIll in the blanks

1. pwrite() function is used for \_positional\_\_\_ writing.
2. In open() filing function, ) \_\_\_\_ flag is used to rewrite a file and discard all previous data.
3. File \_\_\_\_\_\_ remembers its position in a file and position can be changed without reading or writing in a file by using \_\_\_\_\_\_\_ function. \_\_\_\_\_\_\_\_ argument is used to reposition it from the beginning of the file.
4. \_\_\_\_\_\_\_\_\_\_ function is used to get detailed information (ls -l) of a file.
5. In standard I/O, \_IOF?Bf is used set \_\_\_\_\_\_\_
6. In fopen(), “\_\_\_\_\_ “ is used to open a file in binary append mode.
7. If a file descriptor is available then \_\_\_\_\_\_\_ function is used instead of fopen() to open a given file in buffered I/O.
8. \_\_\_\_\_ function is used to close all files before the ending of a program.
9. \_\_\_\_IP ADRESS \_\_\_\_, \_\_\_PORT\_\_\_\_ and \_\_\_PROTOCOL\_\_\_\_\_\_ are used to identify communication sessions at each socket end.
10. SOCK\_STREAM is used to represent \_\_\_\_CONNECTION ORIENTED TCP\_\_\_\_\_\_ transmission layer protocol.
11. \_\_\_\_\_\_BIND\_\_\_\_\_\_\_\_ socket function is used to request an operating system to reserve some network resources.
12. \_\_\_ACCEPT\_\_\_\_\_ and \_\_CONNECT\_\_\_\_ functions are involved behind TCP 3-way handshake.
13. \_\_DNS\_\_ client is called behind the gethostbyname() function.
14. SOCK\_ \_\_\_\_DGRAM \_\_\_\_\_ can be used for all kinds of transmission/communications.
15. \_\_LISTEN\_\_\_\_ function is used to limit number of concurrent socket connections
16. In the open() filing function, O \_\_\_\_\_\_\_\_\_ flag is to rewrite a file and discard all previous data.
17. File \_\_\_\_ remmebrs its position in a file and position can be changed without reading or writing in a file by using \_\_\_\_\_\_\_ function. \_\_\_\_\_\_ argument is used to reposition it from the beginning of the file.
18. \_\_\_\_\_\_ function is used to get detailed information (ls -l) of a file.
19. in sockets \_\_SEND\_\_\_ and \_\_RECIEVE\_\_\_ functions are involved behind full duplex data transfer.
20. SOCK\_\_STREEAM\_\_\_ can be used only for one to one direct communications.
21. In POSIX threads pthread \_\_DESTROY\_\_ is used to kill thread externally and pthread \_\_EXIT\_\_ is used to return value from a thread normally.
22. THe current version of POSIX thread is \_\_IEEE STD 1003.1 - 2017\_\_\_\_.

