Part-1

Which the best basic way of computer architecture?

There are two fundamental approaches to computer architecture:

CISC (Complex Instruction Set Computer)

- CISC architectures feature a wide variety of complex and multi-step instructions
- Suited for tasks that require complex operations and memory access.
- Complex instructions can lead to longer execution times.
- More complex hardware design can lead to higher power consumption and heat generation.

RISC (Reduced Instruction Set Computer)

- RISC architectures focus on a smaller set of simple and fast instructions.

- The goal of RISC architectures is to optimize for instruction execution speed and efficiency.
- streamlined instruction pipeline, leading to faster execution times.
- Reduced hardware complexity, resulting in lower power consumption and heat generation.

I think that RISC architectures is better because of the speed and ease of implementation, which results in less energy consumption and less hardware complexity.

Part-2

Number Languages support auto garbage collection

- Java
- Scala
- Python
- C#
- Ruby

Part-3

What is BIOS?

output system) is the program a computer's microprocessor uses to start the computer system after it is powered on.

Why do we use it?

It is managed data flow between the computer's operating system (OS) and attached devices, such as the hard disk, video adapter, keyboard, mouse and printer.

Part-4

Linux Vs Unix with example

What is Linux?

Linux is an open-source operating system. This OS is supported on several computer platforms and includes multiple software features that handle computer resources, and allow you to do tasks.

What is Unix?

Unix is a powerful and multitasking operating system that behaves like a bridge between the user and the computer. It allows the user to perform specific functions.

	Linux	Unix
OS family	It belongs to the Unix-like family.	It belongs to the Unix family.
Available in	It is available in multiple languages.	It is available in English.
Written in	C and other programming languages.	C and assembly language.
File system support	It supports more file systems than Unix.	It also supports less than Linux.
Usage	It is used in several systems like desktop, smartphones, mainframes and servers.	Unix is majorly used on workstations and servers.
Examples	Some examples of Linux are: Fedora, Debian, Red Hat, Ubuntu, Android, etc.	Some examples of unix are IBM AIX, Darwin, Solaris, HP-UX, macOS X, etc.
Price	Linux is free and its corporate support is available at a price.	Unix is not totally free. There are some Unix versions that are free, other than that UNIX is expensive.

Part-5

What is fragmentation?

Fragmentation refers to the phenomenon where storage space or memory becomes divided or scattered into smaller pieces, making it inefficient to use them effectively.

Types of Fragmentation:

Disk Fragmentation:

- External Fragmentation: Free space becomes fragmented into non-contiguous segments, making it difficult to allocate large files efficiently.

- Internal Fragmentation: Within individual files, allocated space is larger than needed, resulting in wasted space.

Memory Fragmentation:

- External Memory Fragmentation: Scattered free memory blocks make it challenging to allocate contiguous memory blocks.

- Internal Memory Fragmentation: Allocated memory blocks are larger than required, leading to unused memory space.

Part-6

Compare among all scheduling algorithms [Round robin - Priority - First come first serve]

	Round robin	Priority	First come first serve
Definition	is the preemptive process scheduling algorithm.	is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems.	Jobs are executed on first come, first serve basis.
Fairness	Fair due to time-slice distribution.	Can be unfair if lower-priority	Fair in terms of order of arrival, but not in

		processes starve.	execution time
Preemption	Preemptive by nature due to time slices.	Preemptive, as higher priority tasks can preempt lower priority tasks.	Non-preemptive, as a process runs until it completes.
Response Time	Fair response time due to time-slice distribution.	High-priority tasks have low response time.	Depends on the order of arrival; short jobs may experience longer waiting times.

Part-7

Parallel processing Vs Threads

Parallel processing and threads are both concepts related to improving the performance and efficiency of computer systems, particularly in the context of multitasking and multitasking environments.

	Parallel processing	Threads
Relationship	Focuses on executing multiple tasks or processes concurrently, potentially on different processors or computers.	Focuses on concurrent execution of multiple tasks within the same process, utilizing multiple threads of execution.

Data Sharing	Typically requires explicit communication mechanisms for data sharing between different processes.	Threads within the same process share the same memory space, making data sharing more straightforward.
Complexity	Can be more complex due to the need for communication and synchronization between different processes.	Generally simpler to implement since threads share memory and resources within the same process.

Part-8

Languages support multithreading

- Java
- Python
- C++
- C#
- Ruby

Part-9

Clean Code principles

- Meaningful Names
- Functions and Methods
- Comments
- Formatting and Indentation
- Avoid Code Duplication
- Single Responsibility Principle (SRP)
- Open/Closed Principle (OCP)
- Liskov Substitution Principle (LSP)
- Interface Segregation Principle (ISP)
- Dependency Inversion Principle (DIP)
- Testing