Great! Here are short and easy-to-understand answers for **Assignments 1 to 6** from your Distributed Systems practical question bank. The list of difficult term explanations follows after all the answers.

**Assignment 1: RMI (Remote Method Invocation)**

1. **What is Remote Method Invocation (RMI)?**  
   RMI allows a Java program to call methods on an object located on another machine.
2. **Explain the concept of stub and skeleton in RMI.**  
   Stub acts as the client-side proxy, while skeleton is the server-side handler for the remote object.
3. **Write steps to implement RMI-based application.**  
   Define interface, implement it, create stub/skeleton using rmic, start registry, and run server/client.
4. **Write a program to create a simple RMI server and client.**  
   Create interface, server class binding object to registry, and client class looking up and invoking remote method.
5. **What are the advantages of using RMI in distributed applications?**  
   RMI simplifies remote communication and hides networking details from the programmer.
6. **Explain how to start the rmiregistry and why it is necessary.**  
   Run rmiregistry command; it allows remote object lookup by clients.
7. **Differentiate between local method calls and remote method calls.**  
   Local calls occur within the same JVM, remote calls happen over a network.
8. **How do you handle RemoteExceptions in RMI?**  
   Use try-catch blocks to handle RemoteException which may occur due to network issues.
9. **What role does the Naming service play in RMI?**  
   It registers and looks up remote objects using names.
10. **Explain the lifecycle of a remote method call in RMI.**  
    Client calls stub → stub contacts remote object → method executes → result returns to client.

**Assignment 2: CORBA (Common Object Request Broker Architecture)**

1. **What is CORBA?**  
   CORBA is a standard for enabling communication between programs written in different languages over a network.
2. **Explain the concept of IDL in CORBA.**  
   IDL is used to define interfaces that can be used across different programming languages.
3. **Write the steps involved in creating a CORBA-based application in Java.**  
   Define IDL, compile it, implement server and client, start ORB and Naming Service, run client.
4. **What is the role of the ORB in CORBA?**  
   ORB handles communication and data exchange between client and server objects.
5. **Differentiate between CORBA and RMI.**  
   CORBA supports multiple languages, RMI supports only Java.
6. **Write a program to create a simple CORBA server and client for a Calculator application.**  
   Define calculator IDL, implement server class with operations, write client to access methods.
7. **What is IIOP in CORBA?**  
   IIOP is the protocol used for communication between CORBA objects over the internet.
8. **What is Portable Object Adapter (POA) in CORBA?**  
   POA is a component that helps manage server-side object implementations.
9. **Explain the role of the Naming Service in CORBA.**  
   It allows clients to find server objects using names.
10. **How does Java support CORBA integration?**  
    Java provides built-in packages like org.omg for CORBA support.

**Assignment 3: MPI or OpenMPI (Message Passing Interface)**

1. **What is Message Passing Interface (MPI)?**  
   MPI is a standard used to allow processes to communicate in parallel computing environments.
2. **Write the steps to install and configure MPJ Express.**  
   Download MPJ Express, set environment variables, compile and run programs using mpjrun.
3. **Write a Java program using MPI to distribute array elements among processors.**  
   Divide array and use MPI\_Send and MPI\_Recv to pass segments to each processor.
4. **How does MPI help in achieving parallelism?**  
   It allows multiple processes to run and communicate independently to solve parts of a task.
5. **Differentiate between MPI and OpenMP.**  
   MPI is for distributed systems, OpenMP is for shared memory systems.
6. **What is the significance of MPI\_COMM\_WORLD communicator?**  
   It includes all processes in an MPI program and helps them communicate.
7. **Explain the use of MPI\_Send and MPI\_Recv functions.**  
   MPI\_Send sends data to another process, MPI\_Recv receives it.
8. **What is the difference between Scatter and Gather in MPI?**  
   Scatter sends data from one process to many; Gather collects data from many to one.
9. **How are the intermediate sums calculated in distributed array sum using MPI?**  
   Each process sums its part and sends it to the root which adds them up.
10. **Explain SPMD model with respect to MPJ Express.**  
    All processes run the same code but work on different parts of the data.

**Assignment 4: Berkeley Algorithm (Clock Synchronization)**

1. **What is Berkeley’s algorithm?**  
   A method to synchronize clocks of all nodes by averaging time values.
2. **Write steps involved in Berkeley clock synchronization.**  
   Master polls time, calculates average offset, and updates slaves.
3. **How is the master node selected in Berkeley’s algorithm?**  
   It is pre-defined or chosen based on system design.
4. **Explain the importance of clock synchronization in distributed systems.**  
   It ensures coordinated operations and correct event ordering.
5. **Differentiate between Cristian’s algorithm and Berkeley’s algorithm.**  
   Cristian’s uses a time server; Berkeley’s uses average of all clocks.
6. **How is the average clock offset calculated in Berkeley’s algorithm?**  
   Subtracts each clock’s time from average and adjusts accordingly.
7. **What are the features of Berkeley’s algorithm?**  
   Uses averaging, handles drift, and doesn’t rely on a time server.
8. **How does Berkeley’s algorithm handle faulty nodes?**  
   It excludes nodes with large time differences from calculation.
9. **What is the role of master and slave nodes in Berkeley’s algorithm?**  
   Master coordinates time collection and distribution; slaves adjust clocks.
10. **Illustrate Berkeley’s algorithm with an example.**  
    Master polls time: A=10, B=12, C=14 → avg=12 → all set to 12.

**Assignment 5: Token Ring Based Mutual Exclusion Algorithm**

1. **What is the token ring algorithm?**  
   A mutual exclusion method where a token is passed in a ring; only the holder can enter the critical section.
2. **Write a program to implement token ring mutual exclusion.**  
   Create processes in a ring and pass token to allow critical section access.
3. **How is the token passed in a token ring algorithm?**  
   Each process sends the token to the next in a fixed circular order.
4. **What happens if the token is lost in a token ring system?**  
   A new token must be generated to restore coordination.
5. **How does token ring ensure mutual exclusion?**  
   Only one token exists, so only one process can enter the critical section at a time.
6. **Explain the advantages of token ring algorithm.**  
   Simple, fair, and prevents starvation.
7. **What are the drawbacks of the token ring method?**  
   Token loss or process failure can halt the system.
8. **Describe the message complexity in token ring algorithm.**  
   Requires one message (token) per entry to the critical section.

**Assignment 6: Bully and Ring Algorithm (Leader Election)**

1. **What is the Bully algorithm for leader election?**  
   Highest-ID process becomes leader by forcing lower-ID processes to back down.
2. **What is the Ring algorithm for leader election?**  
   Nodes pass messages in a ring; the highest-ID node becomes leader.
3. **Write a program to implement Bully algorithm.**  
   Each process checks if others are alive and higher ID; initiates election if needed.
4. **Write a program to implement Ring algorithm.**  
   Each process sends its ID around the ring; the highest-ID one is chosen.
5. **How does a node initiate an election in Bully algorithm?**  
   It sends messages to higher-ID processes; if no reply, it becomes leader.
6. **What are the assumptions made in Bully algorithm?**  
   All nodes know each other’s IDs and the system is synchronous.
7. **Compare Bully and Ring election algorithms.**  
   Bully is faster but needs more messages; Ring is simpler and uses fewer resources.
8. **How does the Ring algorithm ensure fairness during election?**  
   Every node gets a chance to become leader by passing messages equally.

**Assignment 7: Web Services in Distributed Systems**

1. **What are Web Services in Distributed Systems?  
   Web services allow software applications to talk to each other over the internet using standard protocols.**
2. **Differentiate between SOAP and RESTful Web Services.  
   SOAP uses XML and is strict; REST is lightweight and uses URLs with multiple formats like JSON or XML.**
3. **Write the steps to create a RESTful web service in Java.  
   Use a framework like Jersey, create resource classes with annotations, deploy on a server like Tomcat.**
4. **Write a Java program to implement a simple RESTful web service for a Library system.  
   Define book resource class with methods for add, view, delete using @GET, @POST, @DELETE.**
5. **How is HTTP used in RESTful web services?  
   HTTP methods like GET, POST, PUT, and DELETE perform operations on web resources.**
6. **What are the advantages of using REST over SOAP?  
   REST is faster, easier to implement, and better for web-based applications.**
7. **Explain the role of WSDL in SOAP-based web services.  
   WSDL describes the web service's operations, input/output, and location.**
8. **How does JSON help in RESTful web services?  
   JSON is a lightweight data format used to send/receive data in REST APIs.**
9. **What is JAX-RS in Java?  
   JAX-RS is a Java API used to build RESTful web services using annotations.**
10. **How does a client interact with a RESTful web service?  
    Client sends HTTP requests (e.g., GET/POST) to the service's URL and receives data.**

**Extra Questions**

1. **What is a Distributed System?  
   A system where multiple computers work together and appear as a single system to the user.**
2. **List advantages of Distributed Systems.  
   Resource sharing, scalability, fault tolerance, and better performance.**
3. **Differentiate between tightly coupled and loosely coupled systems.  
   Tightly coupled systems share memory; loosely coupled systems use message passing.**
4. **What is Scalability in Distributed Systems?  
   The system's ability to handle growth in users or workload without performance drop.**
5. **What is the role of middleware in Distributed Systems?  
   Middleware helps applications communicate and manage resources across distributed systems.**
6. **What are the goals of Distributed Systems?  
   Transparency, scalability, reliability, and resource sharing.**
7. **What are the challenges in Distributed Systems?  
   Synchronization, fault tolerance, network latency, and security.**
8. **Define Transparency in Distributed Systems.  
   Hiding the complexity of distribution from users (e.g., location or access transparency).**
9. **What is fault tolerance?  
   The system’s ability to keep working even if some components fail.**

**Additional Term Explanations**

* **SOAP (Simple Object Access Protocol): A protocol for accessing web services using XML.**
* **REST (Representational State Transfer): A web service style using HTTP methods.**
* **JAX-RS: Java API for RESTful web services.**
* **WSDL (Web Services Description Language): Describes SOAP web services.**
* **JSON (JavaScript Object Notation): Lightweight data format for communication.**
* **Jersey: A framework for developing RESTful web services in Java.**
* **Tomcat: A web server to run Java-based web applications.**
* **Middleware: Software that connects different applications or services.**
* **Transparency (in distributed systems): Making the system appear as a single unified entity to users.**

**List of Difficult Term Explanations**

* **RMI (Remote Method Invocation):** Allows Java objects to communicate over a network.
* **Stub/Skeleton:** Proxy objects for communication in RMI.
* **rmiregistry:** Registry to locate RMI objects.
* **ORB (Object Request Broker):** Middleware that handles communication in CORBA.
* **IDL (Interface Definition Language):** Language-neutral interface definition in CORBA.
* **IIOP:** Internet protocol used in CORBA.
* **MPI:** Communication standard for parallel processes.
* **MPI\_COMM\_WORLD:** Default communicator in MPI.
* **SPMD:** A parallel model where all processors run the same code.
* **Clock Synchronization:** Aligning clocks of different nodes in a system.
* **Token:** A permission message used in mutual exclusion.
* **Critical Section:** A part of code where shared resource access happens.
* **Bully Algorithm:** Leader election where the highest-ID process wins.
* **Ring Algorithm:** Leader election using circular message passing.

**Extra Viva-Worthy Questions**

1. **What is RPC (Remote Procedure Call)?**  
   It lets a program call a function on another computer as if it were local.
2. **What is RMI (Remote Method Invocation) in Java?**  
   It allows Java programs to invoke methods on remote Java objects.
3. **What is a Stub in RMI?**  
   A stub is a local object that acts as a proxy for the remote object.
4. **What is Latency in Distributed Systems?**  
   The time delay between a request and response over the network.
5. **What is Load Balancing?**  
   Distributing work evenly across servers to avoid overload.
6. **What is Consistency in Distributed Systems?**  
   Ensuring all nodes see the same data at the same time.
7. **What is a Proxy Server?**  
   A server that acts as an intermediary for client requests.
8. **Explain the CAP Theorem.**  
   A distributed system can only guarantee two out of Consistency, Availability, and Partition tolerance at a time.
9. **What is Data Replication?**  
   Storing the same data on multiple machines for reliability and speed.
10. **What is a Distributed File System?**  
    A file system that lets users access files stored on multiple computers as if they were local.