



















Guía examen. Preguntas y respuestas

















1. **Un sistema dónde el código y los datos residen en una sola computadora** → ☒ Sistema centralizado
2. **Si el nodo A inicia el algoritmo de Berkeley ¿Qué hora tendrán las computadoras al final?** → ☒ 3:20, B: 3:20, C: 3:20
3. **Sincronizar los relojes de dos o más servidores significa** → ☒ Los servidores se ponen de acuerdo en una misma hora
4. **En el algoritmo del abusón** → ☒ Un nodo envía un mensaje de elección a los nodos con mayor número de nodo
5. **En el algoritmo de elección en anillo** → ☒ Los nodos están conectados en anillo ordenados de menor a mayor
6. **Si la variable t contiene una instancia de la clase Thread ¿Qué hace el método t.start?** → ☒ Inicia la ejecución del thread t
7. **El nodo 1 envía un mensaje al tiempo 70 y el nodo 2 recibe el mensaje al tiempo 60, de acuerdo al algoritmo de Lamport** → ☒ El nodo 2 cambia su tiempo a 71
8. **En el algoritmo centralizado de exclusión mutua, si todos los nodos quieren adquirir el bloqueo, entonces** → ☒ Un nodo adquiere el bloqueo, los otros se encolan en el coordinador
9. **En el algoritmo del abusón ¿Cuándo se inicia un proceso de elección?** → ☒ Cuando algún nodo se da cuenta que el coordinador no responde
10. **Posibilidad de ejecutar el programa en diferentes plataformas sin la necesidad de hacer cambios al programa** → ☒ Portabilidad
11. **Si un procesador bloquea un recurso de manera exclusiva** → ☒ No se puede bloquear el recurso de manera exclusiva o compartida
12. **Si A happens-before B y B happens-before C, entonces** → ☒ A happens-before C




13. **Un sistema puede escalar en** → ☒ Tamaño, geografía y administración
14. **Los nodos del 0 al 9 están conectados en anillo, el nodo 1 inicia proceso de elección ¿Qué nodo se erige coordinador?** → ☒ El nodo 9
15. **En el algoritmo de exclusión mutua de token-ring ¿Cuándo puede adquirir el bloqueo un nodo?** → ☒ Cuando el nodo recibe el token
16. **En el algoritmo de elección en anillo ¿Cuándo se inicia un proceso de elección?** → ☒ Cuando algún nodo se da cuenta que el coordinador no responde
17. **Si un procesador bloquea un recurso de manera compartida** → ☒ Se puede bloquear el recurso de manera compartida pero no exclusiva
18. **En una malla, capa compuesta por las aplicaciones que ejecutan dentro de la organización virtual** → ☒ Capa de aplicaciones
19. **En el algoritmo distribuido de exclusión mutua ¿qué nodo "gana" el bloqueo?** → ☒ El que envió el mensaje de petición en un tiempo lógico menor
20. **La distribución de los datos** → ☒ Aumenta la confiabilidad y mejora el rendimiento
21. **¿Qué se hace para sincronizar los segundos UTC con los segundos TAI?** → ☒ Atrasar el tiempo UTC un segundo al año con respecto al tiempo TAI
22. **Si dos computadoras no están conectadas** → ☒ No requieren sincronizar sus tiempos
23. **Son objetivos de los sistemas distribuidos** → ☒ Facilidad de acceso a recursos, transparencia, apertura, escalabilidad
24. **Capacidad del sistema distribuido de ocultar una falla** → ☒ Transparencia ante fallas
25. **¿Qué afirmación es correcta sobre el método read de la clase DataInputStream?** → ☒ El método read podría leer solo una fracción del mensaje enviado
26. **Si el evento A es el envío de un mensaje y el evento B la recepción del mensaje** → ☒ A happens-before B
27. **Capa de software distribuido que actúa como "puente" entre las aplicaciones y el sistema operativo** → ☒ Middleware

28. ¿Qué hace el método `accept` de la clase `ServerSocket`? → ☒ Espera una conexión del cliente, regresa un objeto de tipo `Socket`
29. En este algoritmo los clientes consultan un servidor de tiempo → ☒
`Network Time Protocol`
30. Cuando dos o más threads acceden a una misma variable, y al menos uno de los threads modifica la variable, es necesario → ☒ Sincronizar el acceso de los threads a la variable
31. ¿Con qué comando se instala NTP en Ubuntu sin utilizar el usuario root? → ☒ `sudo apt-get install ntp`
32. ¿Por qué un programa que utiliza sincronización es más lento? → ☒
Porque ciertas partes del programa se ejecuten en serie
33. El CPU requiere escribir una variable que se encuentra en la memoria RAM ¿qué pasa si la variable no existe en la cache? → ☒ Se copia la línea de cache donde está la variable, de la RAM a la cache
34. Ordenados de mayor a menor velocidad → ☒ Registros, Cache, RAM, Disco duro
35. Si la variable entrada contiene un objeto de tipo `DataInputStream` ¿Qué hace el método `entrada.readInt`? → ☒ Lee un entero de 32 bits
36. A la fracción $1/86400$ de día se le llama → ☒ Segundo solar
37. Qué hace la instrucción: `Socket conexion = new Socket("midominio.com",10000);` → ☒ Conecta el cliente con el servidor `midominio.com` usando el puerto 10000
38. La capacidad de un sistema distribuido de presentarse ante los usuarios y aplicaciones como una sola computadora → ☒ Transparencia
39. Ordenados de mayor a menor capacidad → ☒ Disco duro, RAM, Cache, Registros
40. El recíproco de la frecuencia natural de resonancia del Cesio 133 → ☒
Segundo atómico
41. En un sistema distribuido se puede distribuir → ☒ El procesamiento y los datos
42. Si la variable salida contiene un objeto de tipo `DataOutputStream` ¿Qué hace el método `salida.write`? → ☒ Escribe un arreglo de bytes

43. **Suponga que tiene un programa Cliente y un programa Servidor, entonces** → ☒ El cliente siempre se conecta al servidor
44. **Capacidad del sistema distribuido de ocultar la existencia de recursos replicados** → ☒ Transparencia de replicación
45. **Promedio mundial de los segundos atómicos transcurridos desde el 1958-01-01** → ☒ Tiempo Atómico Internacional
46. **Es ejemplo de sistema centralizado** → ☒ Computadora stand-alone
47. **En una computadora ¿Dónde se encuentran los registros?** → ☒ En el CPU
48. **Conjunto de computadoras homogéneas con el mismo sistema operativo conectadas mediante una red local de alta velocidad** → ☒ Cluster
49. **Cuando un thread espera la terminación de uno o más threads para continuar su ejecución, se dice que se implementa** → ☒ Una barrera
50. **"Write once, run everywhere" se refiere a** → ☒ La portabilidad de Java
51. **¿Qué es la autorización?** → ☒ Esquema de permisos para el acceso a los recursos
52. **Sea Pi un procesador. A la función Ci(A), la cual asigna un número al evento A, se le llama** → ☒ Reloj lógico
53. **¿En que caso la cache es contraproducente?** → ☒ Cuando se acceden datos que se encuentran separados en la memoria
54. **Si A happens-before B entonces** → ☒ no(B happens-before A)
55. **Conjunto de computadoras generalmente heterogéneas agrupadas en organizaciones virtuales** → ☒ Malla
56. **Las ventajas de un sistema distribuido son, entre otras** → ☒ Usuarios, procesamiento, almacenamiento, ancho de banda casi ilimitados
57. **Capacidad de los sistemas de crecer mediante la incorporación de componentes fáciles de reemplazar y adaptar** → ☒ Extensibilidad
58. **Capacidad de cambiar la ubicación de un recurso mientras está en uso, sin afectar al usuario que accede el recurso** → ☒ Transparencia de re-ubicación
59. **¿Qué estándar de tiempo utilizan los proveedores de nube?** → ☒ El tiempo UTC

- 60. **Son ejemplos de sistemas distribuidos** →  World Wide Web, cómputo en la nube, SETI, TOP500
- 61. **Si un sistema es tolerante a fallas entonces es un sistema** →  Fiable
- 62. **Tipo de socket que no incluye acknowledgement ni re-envío** →  Socket datagrama
- 63. **Garantiza que todos los miembros de un grupo reciben los mensajes transmitidos, sin importar el orden en que se reciben** →  Multicast confiable
- 64. **Falla que se produce cuando el tiempo de respuesta del sistema es mayor al especificado en los requisitos no funcionales** →  Falla de tiempo
- 65. **El cliente utiliza éste método para poder recibir los mensajes enviados a un grupo** →  `socket.joinGroup(grupo)`
- 66. **Comunicación punto a punto dónde una computadora envía mensajes a otra computadora** →  Unicast
- 67. **Crea un datagrama para envío** →  `new DatagramPacket(buffer,buffer.length,grupo,puerto)`
- 68. **Rango del primer byte en una dirección IP V4 utilizada para multicast** →  224 a 239
- 69. **Multitransmisión en la cual una computadora envía mensajes a todas las computadoras en una red** →  Multicast
- 70. **Fallas que oculta el protocolo TCP** →  Fallas por omisión
- 71. **Un sistema que continúa operando con normalidad ante las fallas** →  Tolerancia fallas
- 72. **Capacidad que tiene un sistema de funcionar correctamente siempre** →  Disponibilidad
- 73. **Capacidad de un sistema de funcionar continuamente sin fallar** →  Confiabilidad
- 74. **Propiedad que tiene un sistema de no causar un evento catastrófico cuando falla** →  Seguridad
- 75. **Capacidad que tiene un sistema de ser reparado cuando falla** →  Mantenimiento

- 76. **Falla que se presenta cuando el sistema estaba funcionando normalmente de pronto se detiene** →  **Falla de congelación**
- 77. **Falla que se presenta cuando el sistema no recibe los mensajes o no envía los mensajes** →  **Falla de omisión**
- 78. **Falla que se produce cuando el tiempo de respuesta del sistema es mayor al especificado en los requerimientos no funcionales** →  **Falla de tiempo**
- 79. **Falla que produce una respuesta con un valor incorrecto** →  **Falla de respuesta**
- 80. **Falla que produce cualquier respuesta, en cualquier momento y con cualquier tiempo de respuesta** →  **Falla arbitraria**
- 81. **Rango del primer byte en una dirección IPV4 clase A** →  **A 1 a 126**
- 82. **Rango del primer byte en una dirección IPV4 utilizada para multicast** →  **224 a 239**
- 83. **Número máximo de redes y hosts por red en una dirección IP V4 clase C** →  **2097150 redes y 254 hosts por red**
- 84. **Establece una conexión virtual uno a uno utilizando el handshaking** →  **Socket stream**
- 85. **No hay un acknowledgement ni reenvío** →  **Socket datagrama**
- 86. **Protocolo utilizado generalmente para implementar comunicación unicast confiable** →  **TCP**
- 87. **¿Cómo se ocultan las fallas por omisión en la comunicación unicast confiable implementada mediante TCP?** →  **Mediante retransmisión de mensajes**
- 88. **Fallas que oculta el protocolo TCP** →  **Fallas por omisión**
- 89. **Garantiza que todos los miembros de un grupo reciben los mensajes transmitidos, sin importar el orden en que se reciben** →  **Multicast confiable**
- 90. **Implementación de multicast confiable que envía acuse solo cuando no se recibe el mensaje** →  **Acuse negativo**
- 91. **Garantiza que un mensaje llegue a todos a destinatarios o a ninguno** →  **Multicast atómico**

92. **Crea un datagrama para envío** →  `New DatagramPacket(buffer, buffer.length, grupo, puerto)`
93. **El cliente requiere este método para poder recibir los mensajes enviados a un grupo** →  `Socket.joinGroup(grupo)`
94. **En una aplicación multicasts. ¿Qué identifica el grupo al cual se envía los mensajes?** →  Una dirección IP de clase D