1. ¿Qué es Docker?

a) Un sistema operativo basado en Linux
b) Un software de virtualización de contenedores
c) Un lenguaje de programación para desarrollo web
d) Un servicio de almacenamiento en la nube

2. ¿Cuál es la función principal de un contenedor en Docker?

a) Ejecutar múltiples sistemas operativos simultáneamente
b) Virtualizar la capa de Kernel del sistema operativo
c) Empaquetar y ejecutar aplicaciones con todas sus dependencias
d) Administrar servidores físicos en la nube

3. ¿Por qué Docker facilita el desarrollo de aplicaciones?

a) Porque elimina la necesidad de escribir código
b) Porque ofrece entornos aislados y preconfigurados
c) Porque no requiere instalación de software adicional
d) Porque solo funciona en sistemas Windows

4. ¿Por qué Docker facilita el despliegue de aplicaciones?

a) Porque elimina completamente la necesidad de configurar el servidor
b) Porque permite ejecutar aplicaciones sin importar el sistema operativo
c) Porque abstrae dependencias y configuraciones necesarias
d) Porque utiliza máquinas virtuales con menos recursos

5. ¿Cuál es una de las principales diferencias entre Docker y una máquina virtual tradicional?

a) Docker virtualiza el sistema operativo completo
b) Las máquinas virtuales son más ligeras que los contenedores de Docker
c) Docker solo virtualiza la capa de aplicación, no el sistema operativo completo
d) Docker no requiere de un hipervisor para funcionar

6. ¿Qué capa adicional añade Docker para permitir compatibilidad con Windows y MacOS?

a) Un sistema de archivos compartido
b) Un hipervisor con una distribución de Linux ligera
c) Un motor de contenedores de Windows
d) Una capa de compatibilidad con hardware

7. ¿Qué comando en la CLI de Docker permite interactuar con el servidor de Docker?

a) `docker-compose`
b) `dockerd`

c) docker
d) `docker-run`

8. ¿Cuál de las siguientes herramientas NO está incluida en Docker Desktop?

a) Docker Engine
b) Docker Compose
c) Docker Virtual Machine
d) Kubernetes

9. ¿Cuál es la principal diferencia entre una imagen y un contenedor en Docker?

a) No hay diferencia, son lo mismo
b) La imagen es el paquete con la aplicación y sus dependencias, mientras que el contenedor es la instancia en ejecución de esa imagen
c) El contenedor almacena imágenes de Docker
d) La imagen se ejecuta directamente en el hardware, mientras que el contenedor requiere un sistema operativo

10. ¿Dónde se almacenan y distribuyen las imágenes de Docker de manera oficial?

a) Docker Compose
b) Docker Hub
c) Docker Desktop
d) Docker Daemon

11. ¿Qué servicio se encarga de almacenar y administrar imágenes de Docker en la nube?

a) Docker Compose
b) Docker Registry
c) Docker Machine
d) Docker Build

12. ¿Cuál es la relación entre Docker Registry y Docker Repository?

a) Docker Repository es un servicio dentro de Docker Registry donde se almacenan diferentes versiones de una misma imagen
b) Docker Registry y Docker Repository son términos sinónimos
c) Docker Repository almacena múltiples registros de Docker Registry
d) Docker Registry solo permite almacenar imágenes oficiales, mientras que Docker Repository almacena imágenes personalizadas

13. ¿Cuál de las siguientes afirmaciones sobre Docker Hub es correcta?

a) Es el único servicio de almacenamiento de imágenes de Docker
b) Solo permite almacenar imágenes privadas
c) Es el registro de Docker oficial que ofrece imágenes verificadas
d) No permite almacenar imágenes propias

14. ¿Qué herramienta de Docker se utiliza para definir y administrar múltiples contenedores como un solo servicio?

a) Docker Build
b) Docker Compose
c) Docker Hub
d) Docker Daemon

15. ¿Cuál de los siguientes archivos se usa para definir la configuración de un contenedor en Docker Compose?

a) `dockerfile.txt`
b) `compose.yaml` o `docker-compose.yml`
c) `docker.config`
d) `container.json`

16. ¿Cuál de los siguientes NO es un componente de Docker Desktop?

a) Docker Engine
b) Docker CLI
c) Docker VirtualBox
d) Docker Credential Helper

17. ¿Cuál de las siguientes afirmaciones sobre imágenes y contenedores es correcta?

a) Un contenedor es una imagen en ejecución
b) Las imágenes contienen solo el código fuente de la aplicación
c) Los contenedores pueden ejecutarse sin una imagen previa
d) Las imágenes de Docker son incompatibles entre sí

18. ¿Cuál de los siguientes comandos permite crear una imagen de Docker a partir de un Dockerfile?

a) `docker run`
b) `docker build`
c) `docker create`
d) `docker-compose`

19. ¿Qué hace la herramienta Docker Content Trust?

a) Cifra las imágenes de Docker para mayor seguridad
b) Verifica la autenticidad y el origen de las imágenes de Docker
c) Convierte imágenes de Docker en contenedores ejecutables
d) Permite compartir imágenes de forma pública

20. ¿Para qué se usa Kubernetes en Docker Desktop?

a) Para ejecutar imágenes de Windows dentro de Docker
b) Para administrar y orquestar contenedores en entornos de producción

c) Para convertir imágenes de Docker en contenedores
d) Para almacenar imágenes en Docker Hub

21. ¿Qué ventaja principal tienen los contenedores sobre las máquinas virtuales?

a) Los contenedores son más ligeros y rápidos que las máquinas virtuales
b) Los contenedores pueden ejecutar múltiples sistemas operativos simultáneamente
c) Los contenedores incluyen un kernel propio e independiente del sistema host
d) Los contenedores requieren más recursos que las máquinas virtuales

22. ¿Cuál de las siguientes afirmaciones sobre el arranque de contenedores es correcta?

a) Arrancar un contenedor es más rápido que iniciar una máquina virtual
b) Arrancar un contenedor requiere instalar previamente un sistema operativo
c) Un contenedor no puede ejecutar múltiples aplicaciones a la vez
d) Para arrancar un contenedor es necesario usar VirtualBox

23. ¿Cuál de las siguientes NO es una ventaja de usar Docker?

a) Facilita la estandarización del entorno de desarrollo
b) Reduce la cantidad de espacio en disco necesario en comparación con máquinas virtuales
c) Permite empaquetar aplicaciones junto con sus dependencias
d) Aumenta la dependencia del sistema operativo host

24. ¿Cuál es la función del demonio de Docker (`dockerd`)?

a) Gestionar la ejecución de contenedores y la comunicación con la CLI
b) Ejecutar aplicaciones en un entorno virtualizado
c) Administrar las credenciales de los usuarios
d) Controlar el acceso a Docker Hub

25. ¿Cómo se denomina el archivo en el que se define una imagen de Docker?

a) `docker-compose.yml`
b) `Dockerfile`
c) `container.conf`
d) `registry.json`

26. ¿Cómo se crea un contenedor a partir de una imagen en Docker?

a) `docker start imagen`
b) `docker run imagen`
c) `docker create imagen`
d) `docker launch imagen`

27. ¿Cómo se puede listar los contenedores en ejecución en Docker?

a) `docker list`
b) `docker show containers`
c) `docker ps`
d) `docker containers`

28. ¿Cómo se puede detener un contenedor en ejecución en Docker?

a) `docker stop <ID o nombre del contenedor>`
b) `docker pause <ID o nombre del contenedor>`
c) `docker terminate <ID o nombre del contenedor>`
d) `docker kill <ID o nombre del contenedor>`

29. ¿Qué comando permite eliminar un contenedor en Docker?

a) `docker remove <ID>`
b) `docker delete <ID>`
c) `docker kill <ID>`
d) `docker rm <ID>`

31. ¿Cómo se ejecuta un contenedor en segundo plano (modo detached)?

a) `docker run -b imagen`
b) `docker run --bg imagen`
c) `docker run -d imagen`
d) `docker run --detach imagen`

33. ¿Qué hace el comando `docker logs <ID>`?

a) Muestra los archivos dentro del contenedor
b) Muestra los registros (logs) de un contenedor **en ejecución**
c) Lista todas las imágenes descargadas en Docker
d) Elimina un contenedor del registro

34. ¿Cómo se construye una imagen de Docker desde un Dockerfile?

a) `docker create -f Dockerfile`
b) `docker build -t nombre-imagen .`
c) `docker image create -f Dockerfile`
d) `docker make -t nombre-imagen`

36. ¿Cuál de las siguientes afirmaciones sobre Docker Hub es correcta?

a) Solo almacena imágenes de contenedores de pago
b) Es una plataforma donde se pueden alojar y compartir imágenes de Docker
c) No permite almacenar imágenes privadas
d) Requiere una suscripción obligatoria para su uso

37. ¿Cómo se ejecuta un contenedor desde una imagen que está en Docker Hub?

a) `docker run nombre-imagen`
b) `docker download nombre-imagen`
c) `docker fetch nombre-imagen`
d) `docker pull nombre-imagen (lo mismo, PERO NO LA EJECUTA INMEDIAT)`

38. ¿Qué hace el comando `docker pull`?

a) Extrae un contenedor en ejecución
b) Descarga una imagen desde un Docker Registry
c) Muestra la lista de contenedores disponibles
d) Elimina un contenedor

39. ¿Qué comando se usa para eliminar una imagen de Docker del sistema local?

a) `docker remove imagen`
b) `docker delete imagen`
c) `docker rm imagen`
d) `docker rmi imagen`

40. ¿Cómo se verifica qué imágenes están almacenadas en el sistema local?

a) `docker list`
b) `docker images`
c) `docker containers`
d) `docker ps`

41. ¿Qué es un volumen en Docker?

a) Un contenedor con permisos especiales
b) Una imagen optimizada para múltiples instancias
c) Un mecanismo para almacenar datos persistentes fuera del contenedor
d) Una versión específica de una imagen de Docker

42. ¿Cómo se crea un volumen en Docker?

a) `docker volume create nombre-volumen`
b) `docker volume start nombre-volumen`
c) `docker make volumen nombre-volumen`
d) `docker persist volumen nombre-volumen`

43. ¿Cómo se puede asignar un volumen a un contenedor en Docker?

a) `docker run -v nombre-volumen:/ruta-en-contenedor imagen`
b) `docker start --volume nombre-volumen imagen`

c) `docker create --persist nombre-volumen imagen`
d) `docker attach volumen nombre-volumen imagen`

45. ¿Qué hace el comando `docker-compose up`?

a) Elimina todos los contenedores activos
b) Descarga imágenes de Docker Hub
c) Inicia los contenedores definidos en `docker-compose.yml`
d) Compila una imagen de Docker

46. ¿Qué comando permite ver el historial de cambios de una imagen de Docker?

a) `docker history <nombre-imagen>`
b) `docker logs <nombre-imagen>`
c) `docker inspect <nombre-imagen>`
d) `docker show <nombre-imagen>`

49. ¿Qué hace el comando `docker network ls`?

a) Muestra la configuración de red del sistema operativo
b) Lista todas las redes creadas en Docker
c) Muestra los contenedores conectados a internet
d) Cierra todas las conexiones de red de Docker

50. ¿Qué tipo de red crea Docker por defecto para sus contenedores?

a) `bridge`
b) `host`
c) `overlay`
d) `none`

51. ¿Cómo se crea una nueva red en Docker?

a) `docker network create <nombre-red>`
b) `docker create network <nombre-red>`
c) `docker network add <nombre-red>`
d) `docker config network <nombre-red>`

52. ¿Cuál es la función de `docker exec`?

a) Ejecutar un nuevo contenedor
b) Ejecutar un comando dentro de un contenedor en ejecución
c) Eliminar un contenedor en ejecución
d) Descargar una imagen desde Docker Hub

53. ¿Cómo se elimina una red en Docker?

a) `docker network delete <nombre-red>`
b) `docker remove network <nombre-red>`
c) `docker network rm <nombre-red>`
d) `docker drop network <nombre-red>`

55. ¿Cómo se define una variable de entorno en Docker al ejecutar un contenedor?

a) `docker run --env VAR=valor imagen`
b) `docker run --set VAR=valor imagen`
c) `docker run --define VAR=valor imagen`
d) `docker run --export VAR=valor imagen`

60. ¿Qué herramienta en Docker Desktop permite verificar la autenticidad de una imagen?

a) Docker Content Trust
b) Docker Guard
c) Docker Inspect
d) Docker Defender

61. ¿Qué significa el flag `-p` en `docker run -p 8080:80 imagen`?

a) Asigna memoria extra al contenedor
b) Define variables de entorno
c) Mapea un puerto del host al contenedor
d) Especifica una política de reinicio

62. ¿Cuál de los siguientes comandos elimina TODOS los contenedores detenidos?

a) `docker rm $(docker ps -aq)`
b) `docker stop $(docker ps -aq)`
c) `docker remove all`
d) `docker cleanup`

68. ¿Qué ocurre si un contenedor en Docker falla y tiene la política de reinicio `always`?

a) Se reinicia automáticamente sin importar la causa del fallo
b) No se reinicia nunca
c) Solo se reinicia si el fallo fue causado por falta de recursos
d) Se elimina automáticamente del sistema

69. ¿Cómo se configuran múltiples servicios en Docker de manera declarativa?

a) Mediante un archivo `docker-compose.yml`
b) Usando un script de Bash

c) Mediante un archivo `docker-multi.conf`
d) Usando el comando `docker services`

70. ¿Cómo se detienen todos los contenedores en ejecución en Docker?

a) `docker stop $(docker ps -q)`
b) `docker kill all`
c) `docker remove --all-containers`
d) `docker shutdown`

Aquí tienes un examen tipo test completo sobre Docker, cubriendo todos los temas clave que mencionaste. ¡Espero que te ayude a prepararte bien! 🚀

EXAMEN TIPO TEST - DOCKER

(Selecciona la opción correcta en cada pregunta)

SECCIÓN 1: CONCEPTOS BÁSICOS DE DOCKER

1. ¿Qué es Docker?
   a) Un sistema operativo basado en contenedores
   b) Una herramienta para crear, desplegar y ejecutar contenedores
   c) Un lenguaje de programación para virtualización
   d) Un gestor de bases de datos distribuido

2. ¿Cuál es la diferencia principal entre una máquina virtual y un contenedor?
   a) Los contenedores incluyen un sistema operativo completo
   b) Las máquinas virtuales son más ligeras que los contenedores
   c) Los contenedores comparten el kernel del sistema operativo host
   d) No hay ninguna diferencia

SECCIÓN 2: IMÁGENES Y CONTENEDORES

3. ¿Cuál de los siguientes comandos se usa para descargar una imagen desde Docker Hub?
   a) `docker download imagen`
   b) `docker pull imagen`
   c) `docker fetch imagen`
   d) `docker import imagen`

4. ¿Qué comando se usa para ejecutar un contenedor basado en una imagen?
   a) `docker start imagen`
   b) `docker run imagen`
   c) `docker exec imagen`

d) `docker init imagen`

5. ¿Qué sucede si ejecutas `docker run nginx` y la imagen no está en tu máquina?
   a) Se generará un error porque la imagen no está disponible
   b) Docker buscará la imagen localmente y la ejecutará si está disponible
   c) Docker descargará automáticamente la imagen desde Docker Hub y la ejecutará
   d) La imagen se eliminará automáticamente

SECCIÓN 3: GESTIÓN DE CONTENEDORES

6. ¿Cómo se listan los contenedores en ejecución?
   a) `docker list`
   b) `docker show`
   c) `docker ps`
   d) `docker inspect`

7. ¿Cómo se listan todos los contenedores (en ejecución y detenidos)?
   a) `docker ps --all`
   b) `docker ps -a`
   c) `docker show --all`
   d) `docker containers list`

8. ¿Qué comando se usa para detener un contenedor?
   a) `docker stop <ID>`
   b) `docker shutdown <ID>`
   c) `docker pause <ID>`
   d) `docker remove <ID>`

9. ¿Cómo se visualizan los logs de un contenedor en ejecución?
   a) `docker show logs <ID>`
   b) `docker view logs <ID>`
   c) `docker logs <ID>`
   d) `docker inspect logs <ID>`

SECCIÓN 4: PUERTOS Y REDES

10. ¿Cómo se asigna un puerto del host a un contenedor al ejecutarlo?
    a) `docker run -p <puerto_host>:<puerto_contenedor> imagen`
    b) `docker expose <puerto_host>:<puerto_contenedor>`
    c) `docker link <puerto_host>:<puerto_contenedor>`
    d) `docker network <puerto_host>:<puerto_contenedor>`

11. ¿Cómo se listan las redes creadas en Docker?
    a) `docker list networks`
    b) `docker networks show`
    c) `docker network ls`
    d) `docker inspect networks`

12. ¿Cómo se crea una nueva red en Docker?
    a) `docker network create <nombre_red>`
    b) `docker create network <nombre_red>`
    c) `docker add network <nombre_red>`
    d) `docker network new <nombre_red>`

SECCIÓN 5: DOCKERFILES Y CONSTRUCCIÓN DE IMÁGENES

13. ¿Qué es un Dockerfile?
    a) Un archivo JSON con la configuración del contenedor
    b) Un script de shell para iniciar contenedores
    c) Un archivo de configuración YAML para Docker
    d) Un archivo con instrucciones para construir una imagen

14. ¿Qué instrucción en un Dockerfile se usa para definir la imagen base?
    a) `IMAGE`
    b) `FROM`
    c) `BASE`
    d) `USE`

15. ¿Qué instrucción en un Dockerfile se usa para copiar archivos del host al contenedor?
    a) `ADD`
    b) `COPY`
    c) `MOVE`
    d) `IMPORT`

16. ¿Qué comando se usa para construir una imagen a partir de un Dockerfile?
    a) `docker create <ruta>`
    b) `docker build <ruta>`
    c) `docker compile <ruta>`
    d) `docker make <ruta>`

SECCIÓN 6: PERSISTENCIA Y VOLUMENES

17. ¿Para qué sirve Docker Volumes?
    a) Para aumentar el tamaño de los contenedores

b) Para almacenar datos persistentes en contenedores
c) Para mejorar la velocidad de ejecución de los contenedores
d) Para eliminar imágenes automáticamente

18. ¿Cómo se crea un volumen en Docker?
    a) `docker volume create <nombre_volumen>`
    b) `docker create volume <nombre_volumen>`
    c) `docker add volume <nombre_volumen>`
    d) `docker new volume <nombre_volumen>`

19. ¿Cómo se monta un volumen en un contenedor al ejecutarlo?
    a) `docker run -v <nombre_volumen>:<ruta_contenedor> imagen`
    b) `docker run --mount <nombre_volumen>:<ruta_contenedor> imagen`
    c) `docker run -m <nombre_volumen>:<ruta_contenedor> imagen`
    d) `docker run --link <nombre_volumen>:<ruta_contenedor> imagen`

SECCIÓN 7: DOCKER COMPOSE

20. ¿Qué es Docker Compose?
    a) Un sistema para orquestar múltiples contenedores
    b) Un editor de archivos YAML
    c) Un gestor de imágenes en Docker Hub
    d) Un entorno de desarrollo de contenedores

21. ¿Cuál es la extensión de los archivos de Docker Compose?
    a) `.docker`
    b) `.compose`
    c) `.yml` o `.yaml`
    d) `.cfg`

22. ¿Qué comando se usa para iniciar los servicios definidos en un archivo de Docker Compose?
    a) `docker-compose start`
    b) `docker-compose run`
    c) `docker-compose up`
    d) `docker-compose init`

23. ¿Cómo se detienen los contenedores gestionados por Docker Compose?
    a) `docker-compose stop`
    b) `docker-compose down`
    c) `docker-compose terminate`
    d) `docker-compose exit`

Sección 2: Gestión de contenedores

6. ¿Qué hace el comando docker ps?
   a) Muestra solo los contenedores detenidos
   b) Muestra todos los contenedores, en ejecución o no
   c) Muestra solo los contenedores en ejecución
   d) No tiene ninguna función

7. ¿Qué diferencia hay entre docker run y docker run -d?
   a) docker run -d ejecuta el contenedor en modo **detached** (en segundo plano)
   b) docker run -d ejecuta el contenedor en modo depuración
   c) docker run -d detiene el contenedor tras iniciarlo
   d) docker run -d inicia el contenedor con permisos de root

8. ¿Cómo se ejecuta un contenedor asignándole un nombre personalizado?
   a) docker run --rename [nombre] [imagen]
   b) docker run --name [nombre] [imagen]
   c) docker start --name [nombre] [imagen]
   d) docker create --name [nombre] [imagen]

9. ¿Qué comando permite ver los logs de un contenedor en ejecución?
   a) docker logs [ID_Contenedor]
   b) docker ps --logs [ID_Contenedor]
   c) docker inspect logs [ID_Contenedor]
   d) docker show logs [ID_Contenedor]

10. Si se desea reutilizar un contenedor detenido sin crear uno nuevo, ¿qué comando se debe usar?
   a) docker restart [ID_Contenedor]
   b) docker start [ID_Contenedor]
   c) docker resume [ID_Contenedor]
   d) docker boot [ID_Contenedor]

Sección 3: Gestión de imágenes y Dockerfile

11. ¿Para qué sirve el comando docker pull?
   a) Para ejecutar un contenedor desde una imagen
   b) Para descargar una imagen desde Docker Hub

c) Para subir una imagen a Docker Hub
d) Para eliminar una imagen local

12. ¿Qué hace el siguiente comando? docker build -t mi-app:1.0 .
a) Construye una imagen llamada "mi-app" con la versión "1.0"
b) Descarga una imagen llamada "mi-app" de Docker Hub
c) Inicia un contenedor con la imagen "mi-app:1.0"
d) Elimina la imagen "mi-app:1.0"

13. ¿Cuál de las siguientes instrucciones **no** forma parte de un Dockerfile?
a) FROM
b) RUN
c) WORKDIR
d) EXECUTE

14. ¿Cuál es la última instrucción de un Dockerfile que define el comando a ejecutar en el contenedor?
a) RUN
b) WORKDIR
c) CMD
d) COPY

Sección 4: Redes y Volúmenes

16. ¿Cómo se puede exponer un puerto de un contenedor para que sea accesible desde el host?
a) docker run -p [Puerto_Host]:[Puerto_Contenedor] [imagen]
b) docker expose -p [Puerto_Host]:[Puerto_Contenedor] [imagen]
c) docker publish -p [Puerto_Host]:[Puerto_Contenedor] [imagen]
d) docker network -p [Puerto_Host]:[Puerto_Contenedor] [imagen]

17. ¿Para qué se utiliza Docker Network?
a) Para conectar contenedores entre sí
b) Para conectar contenedores con redes externas
c) Para definir redes privadas dentro de Docker

d) Todas las anteriores

18. ¿Cómo se crea una red personalizada en Docker?
    a) docker create network [nombre_red]
    b) docker network create [nombre_red]
    c) docker new network [nombre_red]
    d) docker network --new [nombre_red]

19. ¿Cómo se asocia un contenedor a una red personalizada?
    a) docker run --network [nombre_red] [imagen]
    b) docker network connect [nombre_red] [imagen]
    c) docker network attach [nombre_red] [imagen]
    d) docker net-add [nombre_red] [imagen]

20. ¿Cuál es la función de los volúmenes en Docker?
    a) Permitir la persistencia de datos en contenedores
    b) Ampliar el tamaño del contenedor
    c) Crear copias de seguridad de los contenedores
    d) Optimizar la memoria del host

Sección 5: Docker Compose

21. ¿Para qué sirve Docker Compose?
    a) Para gestionar múltiples contenedores con un solo comando
    b) Para descargar imágenes automáticamente
    c) Para optimizar el rendimiento de los contenedores
    d) Para crear imágenes de contenedores

22. ¿Cuál es el nombre del archivo por defecto que utiliza Docker Compose?
    a) dockerfile.yml
    b) compose.yaml
    c) docker-compose.yml
    d) containers.yml

23. ¿Qué comando se usa para iniciar los contenedores definidos en un archivo docker-compose.yml?
    a) docker compose run
    b) docker-compose up
    c) docker-compose start

d) docker run compose

24. ¿Qué hace el comando docker-compose down?
a) Elimina todos los contenedores y la red definida en el archivo YAML
b) Detiene los contenedores pero los mantiene creados
c) Reinicia los contenedores definidos
d) Nada, el comando no existe

TIPO TEST DE PRÁCTICA 10

Aquí tienes todas las preguntas en **formato de test** listo para imprimir o usar en exámenes.

TEST PRÁCTICA 10 - DOCKER

Sección 1: Creación de contenedor con Dockerfile

1. ¿Cuál es la función del Dockerfile en un proyecto Docker?
a) Ejecutar un contenedor
b) Administrar bases de datos
c) Definir los pasos para construir una imagen de contenedor
d) Iniciar servicios de red

2. ¿Cuál de las siguientes imágenes es adecuada para ejecutar PHP con Apache en Docker?
a) php:8.2-apache
b) apache:latest
c) mysql:8.0
d) ubuntu:latest

3. ¿Dónde se debe copiar el archivo `index.php` dentro del contenedor para que Apache lo sirva correctamente?
a) /app
b) /var/www/html
c) /home/php
d) /usr/src

4. ¿Cómo se define en un Dockerfile la copia de archivos desde el host al contenedor?
   a) ADD index.php /var/www/html/
   b) RUN index.php /var/www/html/
   c) COPY index.php /var/www/html/
   d) MOVE index.php /var/www/html/

5. ¿Qué puerto debe exponerse en el Dockerfile para que el servidor web sea accesible?
   a) 80
   b) 3306
   c) 22
   d) 443

6. ¿Qué hace el comando `docker build -t mi-app .`?
   a) Crea un contenedor desde la imagen "mi-app"
   b) Construye una imagen llamada "mi-app" desde el Dockerfile en el directorio actual
   c) Inicia un contenedor con el nombre "mi-app"
   d) Ejecuta un comando dentro del contenedor "mi-app"

7. ¿Cuál de los siguientes comandos ejecuta un contenedor basado en una imagen creada desde un Dockerfile?
   a) docker start mi-app
   b) docker create mi-app
   c) docker ps -a mi-app
   d) docker run -p 8080:80 --name web-container mi-app

8. ¿Cómo se detiene un contenedor en ejecución desde la terminal?
   a) docker kill [ID_CONTENEDOR]
   b) docker pause [ID_CONTENEDOR]
   c) docker stop [ID_CONTENEDOR]
   d) docker rm [ID_CONTENEDOR]

Sección 2: Docker Compose y MySQL

9. ¿Qué es Docker Compose?
   a) Una herramienta para definir y gestionar múltiples contenedores usando un solo archivo YAML
   b) Un gestor de paquetes para contenedores Docker
   c) Un servicio web para ejecutar contenedores en la nube

d) Un plugin para Docker Desktop

10. ¿Cómo se llama el archivo de configuración principal de Docker Compose?
a) compose.yml
b) docker-config.yml
c) docker-compose.yml
d) compose.json

11. En un archivo docker-compose.yml, ¿cómo se define el servicio para MySQL?
a)

```
services:
  mysql:
    image: mysql
```

b)

```
services:
  db:
    image: mysql:latest
    environment:
      MYSQL_ROOT_PASSWORD: mypassword
      MYSQL_DATABASE: mydatabase
      MYSQL_USER: myuser
      MYSQL_PASSWORD: mypassword
```

c)

```
database:
```

```
  image: mysql:8.0
```

d)

```
services:
  database:
    container: mysql
```

12. ¿Qué variable de entorno define la contraseña del usuario root en un contenedor de MySQL?
    a) MYSQL_USER
    b) MYSQL_DB
    c) MYSQL_PASSWORD
    d) MYSQL_ROOT_PASSWORD

13. ¿Por qué se usa un volumen en el contenedor de MySQL?
    a) Para mejorar la seguridad de la base de datos
    b) Para que los datos de la base de datos persistan aunque el contenedor se elimine
    c) Para reducir el uso de memoria del contenedor
    d) Para ejecutar múltiples instancias de MySQL en el mismo contenedor

14. ¿Cómo se define un volumen en Docker Compose para persistencia de datos en MySQL?
    a)

```
volumes:
  - ./datadir:/var/lib/mysql
```

b)

```
persistencia:
  - mysql_data:/var/mysql
```

c)

```
mount:
  - local:/data/mysql
```

d)

```
storage:
  path: /var/mysql
```

15. ¿Cuál es el comando para iniciar los contenedores definidos en `docker-compose.yml`?
    a) docker start compose
    b) docker-compose init
    c) docker-compose up -d
    d) docker run compose.yml

Sección 3: PHPMyAdmin y Despliegue de Aplicación Web

16. ¿Qué servicio permite gestionar gráficamente bases de datos MySQL en Docker?
    a) MySQL Workbench
    b) Adminer
    c) PHPMyAdmin
    d) PgAdmin

17. ¿Cómo se define un contenedor de PHPMyAdmin en `docker-compose.yml`?
    a)

```
services:
  pma:
    image: phpmyadmin
```

b)

```
services:
  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    environment:
      PMA_HOST: db
      MYSQL_ROOT_PASSWORD: mypassword
    ports:
      - "8081:80"
```

c)

```
services:
  mysqladmin:
    image: phpmyadmin:latest
```

d)

```
phpmyadmin:
  host: mysql
```

18. ¿En qué dirección se accede a PHPMyAdmin en este proyecto?
    a) http://localhost:3306
    b) http://localhost:8081
    c) http://localhost:8080
    d) http://localhost:8888

19. ¿Qué usuario se debe emplear para acceder a PHPMyAdmin con privilegios completos?
 a) root
 b) admin
 c) phpmyadmin
 d) usuario

20. ¿Qué cambios hay que hacer en `index.php` para conectar con la base de datos MySQL en Docker Compose?
 a) Agregar `$db = "db";`
 b) Cambiar `$host = "db";, $user = "nombre_usuario";, $pass = "contraseña";, $db = "nombre_db";`
 c) Definir `$connection = new MySQL();`
 d) Instalar Apache dentro del contenedor

21. ¿Cómo se verifica que los contenedores están en ejecución después de iniciar Docker Compose?
 a) docker ps
 b) docker check
 c) docker verify
 d) docker status

22. ¿Cómo se eliminan completamente los datos persistentes de MySQL si se quiere hacer una instalación limpia?
 a) `docker-compose down --clean`
 b) `docker remove volume`
 c) Eliminando el volumen con `docker volume rm` o borrando la carpeta `datadir`
 d) Reiniciando el servicio de MySQL en Docker

23. **¿Cómo se actualiza la imagen Docker cuando se han hecho cambios en los archivos del proyecto?**
✅ b) `docker-compose build`

24. **Si después de reiniciar los contenedores, los datos de la base de datos siguen presentes, ¿qué componente de Docker es responsable de esto?**
✅ d) El volumen vinculado a `/var/lib/mysql`

Sección 1: Dockerfile y construcción de imágenes

1. ¿Qué hace la instrucción COPY index.php /var/www/html en un Dockerfile?
   a) Copia el archivo index.php desde el host al contenedor
   b) Copia el archivo index.php desde el contenedor al host
   c) Descarga el archivo index.php de internet
   d) Ejecuta el archivo index.php en el contenedor

2. En un Dockerfile, ¿cuál es el propósito de la instrucción EXPOSE 80?
   a) Permite que el contenedor se comunique con otros usando el puerto 80
   b) Expone el puerto 80 en el host
   c) Es obligatorio para que el contenedor funcione
   d) Configura el firewall del host

3. ¿Qué hace el comando docker build -t mi_imagen .?
   a) Crea un contenedor a partir de una imagen
   b) Construye una imagen de Docker usando el Dockerfile en el directorio actual
   c) Ejecuta una imagen existente
   d) Descarga una imagen desde Docker Hub

4. ¿Qué comando se usa para eliminar una imagen de Docker?
   a) docker remove
   b) docker delete
   c) docker rmi
   d) docker rm

Sección 2: Gestión de contenedores

5. ¿Cómo se puede asignar un nombre específico a un contenedor al iniciarlo?
   a) docker run --name mi_contenedor mi_imagen
   b) docker create --name mi_contenedor mi_imagen
   c) docker start --name mi_contenedor mi_imagen
   d) docker assign --name mi_contenedor mi_imagen

Sección 3: Docker Compose y múltiples servicios

8. ¿Cuál es el propósito de docker-compose.yml?
   a) Automatizar la creación y gestión de múltiples contenedores
   b) Crear imágenes de Docker automáticamente
   c) Ejecutar comandos dentro de un contenedor
   d) Descargar imágenes desde Docker Hub

9. ¿Cómo se inicia un conjunto de contenedores definidos en docker-compose.yml?
   a) docker-compose start
   b) docker-compose run
   c) docker-compose up
   d) docker-compose init

10. ¿Qué hace el comando docker-compose down?
   a) Reinicia los contenedores
   b) Detiene y elimina los contenedores, redes y volúmenes definidos en docker-compose.yml
   c) Solo detiene los contenedores sin eliminarlos
   d) Solo elimina las imágenes creadas

11. ¿Qué ocurre si se especifica un volumen en docker-compose.yml?
   a) Los datos del contenedor serán persistentes aunque el contenedor se detenga o elimine
   b) Se asigna más memoria RAM al contenedor
   c) Se borra la base de datos automáticamente al detener el contenedor
   d) Se aumenta la velocidad del contenedor

12. ¿Cómo se define una variable de entorno en docker-compose.yml?
   a) Usando environment: dentro del servicio
   b) Usando ENV_VAR en el terminal antes de ejecutar docker-compose
   c) Modificando el Dockerfile
   d) No es posible definir variables de entorno en

docker-compose.yml

Sección 4: MySQL en Docker

13. ¿Para qué se usa la variable de entorno MYSQL_ROOT_PASSWORD en docker-compose.yml?
    a) Para establecer la contraseña del usuario root en MySQL
    b) Para definir el puerto de MySQL
    c) Para habilitar la autenticación anónima
    d) Para cambiar el nombre de usuario predeterminado

14. ¿Por qué es recomendable definir un volumen para MySQL en Docker?
    a) Para mejorar la velocidad de consulta
    b) Para evitar perder datos cuando el contenedor se detiene o elimina
    c) Para reducir el uso de CPU
    d) Para evitar conflictos con PHPMyAdmin

15. ¿Cómo se accede a la base de datos MySQL dentro de un contenedor?
    a) docker exec -it <nombre_contenedor> mysql -u root -p
    b) docker mysql start
    c) docker-compose mysql connect
    d) docker run mysql --interactive

Sección 5: PHP y conexión con MySQL

16. En db_test.php, ¿qué hace $conn = new mysqli("db", "user", "password", "nombre_db");?
    a) Conectar el script PHP con el servicio de MySQL definido en docker-compose.yml
    b) Ejecutar un contenedor de MySQL
    c) Crear una nueva base de datos
    d) Crear un usuario nuevo en MySQL

17. Si al ejecutar db_test.php en el navegador aparece el error "**Connection refused**", ¿cuál podría ser la causa?
    a) El contenedor de MySQL no está en ejecución
    b) La contraseña del usuario de MySQL es incorrecta
    c) El nombre del host en la conexión PHP es incorrecto
    d) Todas las anteriores

18. ¿Cuál es el propósito de phpMyAdmin en esta práctica?
    a) Permitir una interfaz gráfica para gestionar la base de datos MySQL
    b) Ejecutar código PHP dentro del contenedor
    c) Reemplazar MySQL dentro de Docker
    d) Mejorar el rendimiento del contenedor

19. En docker-compose.yml, ¿cómo se enlaza phpMyAdmin con MySQL?
    a) Usando la opción links:
    b) Definiendo la variable de entorno PMA_HOST=db
    c) Definiendo la variable MYSQL_ADMIN_LINK
    d) No es posible enlazarlos