Esta prueba tiene un valor de 1 punto, y consta de 10 cuestiones tipo test. Cada cuestión plantea 4 alternativas y tiene una única respuesta correcta. Cada respuesta correcta aporta 0.1 puntos, y cada error descuenta 0.033 puntos. Debe contestar en la hoja de respuestas.

1 *En la práctica 3 de laboratorio se ha construido una aplicación docker, donde una de las imágenes se encarga de registrar eventos en un fichero de log. A esta componente la hemos llamado* `logger`. *Señale la opción correcta.*

- **a** El fichero de log que se genera puede corromperse debido a condiciones de carrera en el logger.
- **b** Mediante docker-compose hemos conectado el resto de componentes al logger, haciendo que cada componente tenga acceso al fichero de log, de forma que todas las componentes pueden escribir en el mismo fichero concurrentemente.
- **c** El fichero de log será visible en la máquina `host` dado que así lo hemos indicado en la configuración.
- **d** La componente logger utiliza un socket `rep` para recibir los mensajes de log.

2 *Sobre el sistema con un componente* `worcli`*, el código suministrado acepta como parámetros:*

- **a** Dos url de dos brokers, un retardo y el tipo de trabajo a procesar
- **b** Un url del broker
- **c** Dos url de dos brokers y un retardo
- **d** Dos url de dos brokers y el tipo de trabajo a procesar

3 *Al final del apartado 2 se sugiere reflexionar ante la posibilidad de desplegar* `cbw_ftcl` *en dos escenarios concretos. Selecciona la única afirmación cierta.*

- **a** En un despliegue con dos brokers, únicamente uno ellos será capaz de interactuar con workers, clients y logger
- **b** En un despliegue con dos loggers, los mensajes del broker se repartirán alternativamente (round-robin) entre dichos loggers
- **c** En un despliegue con dos loggers, docker-compose encontrará dificultades para asignar valores a la variable `LOGGER_URL` porque depende de `log`
- **d** Un despliegue con dos brokers no es posible porque ambos realizan bind en la misma URL

4 *En el sistema en el que consideramos un cliente externo:*

- **a** Para redirigir la petición del cliente al broker se realiza una entrada `ROUTE ip_localhost:ip_container` en el Dockerfile del componente broker
- **b** Los clientes externos pueden interactuar con el logger
- **c** La IP del broker es fija y visible desde el nodo donde está ejecutándose el cliente externo
- **d** El acceso al contenedor broker se realiza a través de un puerto del anfitrión

5 *En la práctica 3, en cuanto al cliente externo, no se puede desplegar conjuntamente con* `cbw_ftcl` *porque...*

- **a** El enunciado es falso: sí se puede desplegar conjuntamente
- **b** El despliegue con docker-compose no puede afectar a varios anfitriones
- **c** El cliente externo es incompatible con los clientes normales
- **d** En los equipos de escritorio LINUX no hay Docker instalado

6 *Imagina que acabas de ejecutar sin errores la orden* `docker build -t provisional .`, *indica qué afirmación es cierta:*

- **a** Al ejecutar ahora `docker inspect provisional` puedo consultar su IP (casi al final de la información devuelta) entre otras cosas
- **b** El `Dockerfile` empleado seguro que incluye una orden `CMD` o `ENTRYPOINT`
- **c** Aunque cambie de directorio, es posible ejecutar `docker run provisional`
- **d** En el directorio actual, ahora existe un archivo llamado `provisional`

7 *La orden docker-compose:*

- **a** Toma como entrada únicamente un fichero `Dockerfile` donde se especifican las relaciones entre las imágenes docker.
- **b** Todas las afirmaciones son ciertas.
- **c** Permite ejecutar aplicaciones docker multi-contenedor.
- **d** Permite ejecutar contenedores en máquinas remotas. Para ello se incluyen directivas para la conexión remota dentro del fichero de configuración.

8 *Considerando el escenario del cliente externo (*`4_CBW_FTCL_CEXT`*) y el* `docker-compose.yml` *suministrado*

- **a** Es necesario eliminar la sección `expose:`
- **b** Eliminando la sección `expose:` del `docker-compose.yml` es posible conectar al broker desde un cliente externo
- **c** El `docker-compose.yml` suministrado permite el acceso de un cliente externo al broker
- **d** Es necesario añadir una sección `ports:` mapeando el puerto de clientes desde el anfitrión al contenedor (`9999:9999`)

9 *Si el componente logger se encontrara en un anfitrión diferente al del resto del sistema* `cbw`, *indíquese qué enunciado sería cierto:*

- **a** El archivo de despliegue `yml` deberá dividirse en dos, de manera que la parte referente al logger formará un nuevo archivo (añadiendo el resto de instrucciones necesarias)
- **b** El despliegue automatizado será posible si el anfitrión del nuevo logger permite conexiones desde otros equipos que encamina al contenedor adecuado
- **c** El despliegue automatizado será posible si el anfitrión de `cbw_ftcl` permite conexiones desde otros equipos que encamina a los contenedores adecuados
- **d** Las herramientas empleadas en el laboratorio 3 no permitirían resolver este despliegue de forma automatizada

10 *Si deseamos conocer la dirección IP de un contenedor docker:*

- **a** Utilizaremos la orden `docker ps`
- **b** Utilizaremos la orden `docker images`
- **c** No se puede lograr esa información, pues no es una propiedad de la imagen docker.
- **d** Utilizaremos la orden `docker inspect`

Esta prueba tiene un valor de 1 punto, y consta de 10 cuestiones tipo test. Cada cuestión plantea 4 alternativas y tiene una única respuesta correcta. Cada respuesta correcta aporta 0.1 puntos, y cada error descuenta 0.033 puntos. Debe contestar en la hoja de respuestas.

1 *Si deseamos conocer la dirección IP de un contenedor docker:*

- **a** Utilizaremos la orden `docker inspect`
- **b** No se puede lograr esa información, pues no es una propiedad de la imagen docker.
- **c** Utilizaremos la orden `docker ps`
- **d** Utilizaremos la orden `docker images`

2 *Si el componente logger se encontrara en un anfitrión diferente al del resto del sistema* `cbw`*, indíquese qué enunciado sería cierto:*

- **a** El archivo de despliegue `yml` deberá dividirse en dos, de manera que la parte referente al logger formará un nuevo archivo (añadiendo el resto de instrucciones necesarias)
- **b** El despliegue automatizado será posible si el anfitrión del nuevo logger permite conexiones desde otros equipos que encamina al contenedor adecuado
- **c** Las herramientas empleadas en el laboratorio 3 no permitirían resolver este despliegue de forma automatizada
- **d** El despliegue automatizado será posible si el anfitrión de `cbw_ftcl` permite conexiones desde otros equipos que encamina a los contenedores adecuados

3 *Sobre el sistema con un componente* `worcli`*, el código suministrado acepta como parámetros:*

- **a** Un url del broker
- **b** Dos url de dos brokers y el tipo de trabajo a procesar
- **c** Dos url de dos brokers, un retardo y el tipo de trabajo a procesar
- **d** Dos url de dos brokers y un retardo

4 *En la práctica 3 de laboratorio se ha construido una aplicación docker, donde una de las imágenes se encarga de registrar eventos en un fichero de log. A esta componente la hemos llamado* `logger`*. Señale la opción correcta.*

- **a** Mediante docker-compose hemos conectado el resto de componentes al logger, haciendo que cada componente tenga acceso al fichero de log, de forma que todas las componentes pueden escribir en el mismo fichero concurrentemente.
- **b** La componente logger utiliza un socket `rep` para recibir los mensajes de log.
- **c** El fichero de log será visible en la máquina `host` dado que así lo hemos indicado en la configuración.
- **d** El fichero de log que se genera puede corromperse debido a condiciones de carrera en el logger.

5 *En la práctica 3, en cuanto al cliente externo, no se puede desplegar conjuntamente con* `cbw_ftcl` *porque…*

- **a** El despliegue con docker-compose no puede afectar a varios anfitriones
- **b** El enunciado es falso: sí se puede desplegar conjuntamente
- **c** El cliente externo es incompatible con los clientes normales
- **d** En los equipos de escritorio LINUX no hay Docker instalado

6 *Considerando el escenario del cliente externo (*`4_CBW_FTCL_CEXT`*) y el* `docker-compose.yml` *suministrado*

a Es necesario eliminar la sección `expose:`

b Es necesario añadir una sección `ports:` mapeando el puerto de clientes desde el anfitrión al contenedor (`9999:9999`)

c El `docker-compose.yml` suministrado permite el acceso de un cliente externo al broker

d Eliminando la sección `expose:` del `docker-compose.yml` es posible conectar al broker desde un cliente externo

7 *La orden docker-compose:*

a Todas las afirmaciones son ciertas.

b Permite ejecutar aplicaciones docker multi-contenedor.

c Permite ejecutar contenedores en máquinas remotas. Para ello se incluyen directivas para la conexión remota dentro del fichero de configuración.

d Toma como entrada únicamente un fichero `Dockerfile` donde se especifican las relaciones entre las imágenes docker.

8 *En el sistema en el que consideramos un cliente externo:*

a Los clientes externos pueden interactuar con el logger

b El acceso al contenedor broker se realiza a través de un puerto del anfitrión

c Para redirigir la petición del cliente al broker se realiza una entrada `ROUTE ip_localhost:ip_container` en el Dockerfile del componente broker

d La IP del broker es fija y visible desde el nodo donde está ejecutándose el cliente externo

9 *Imagina que acabas de ejecutar sin errores la orden* `docker build -t provisional .`, *indica qué afirmación es cierta:*

a Al ejecutar ahora `docker inspect provisional` puedo consultar su IP (casi al final de la información devuelta) entre otras cosas

b El `Dockerfile` empleado seguro que incluye una orden `CMD` o `ENTRYPOINT`

c Aunque cambie de directorio, es posible ejecutar `docker run provisional`

d En el directorio actual, ahora existe un archivo llamado `provisional`

10 *Al final del apartado 2 se sugiere reflexionar ante la posibilidad de desplegar* `cbw_ftcl` *en dos escenarios concretos. Selecciona la única afirmación cierta.*

a Un despliegue con dos brokers no es posible porque ambos realizan bind en la misma URL

b En un despliegue con dos loggers, docker-compose encontrará dificultades para asignar valores a la variable `LOGGER_URL` porque depende de `log`

c En un despliegue con dos brokers, únicamente uno ellos será capaz de interactuar con workers, clients y logger

d En un despliegue con dos loggers, los mensajes del broker se repartirán alternativamente (round-robin) entre dichos loggers

TSR	Laboratorio	Práctica 3	2019-01-09	D

Esta prueba tiene un valor de 1 punto, y consta de 10 cuestiones tipo test. Cada cuestión plantea 4 alternativas y tiene una única respuesta correcta. Cada respuesta correcta aporta 0.1 puntos, y cada error descuenta 0.033 puntos. Debe contestar en la hoja de respuestas.

1 *En el sistema en el que consideramos un cliente externo:*

- **a** El acceso al contenedor broker se realiza a través de un puerto del anfitrión
- **b** Para redirigir la petición del cliente al broker se realiza una entrada `ROUTE ip_localhost:ip_container` en el Dockerfile del componente broker
- **c** Los clientes externos pueden interactuar con el logger
- **d** La IP del broker es fija y visible desde el nodo donde está ejecutándose el cliente externo

2 *Si el componente logger se encontrara en un anfitrión diferente al del resto del sistema* `cbw`*, indíquese qué enunciado sería cierto:*

- **a** El archivo de despliegue `yml` deberá dividirse en dos, de manera que la parte referente al logger formará un nuevo archivo (añadiendo el resto de instrucciones necesarias)
- **b** Las herramientas empleadas en el laboratorio 3 no permitirían resolver este despliegue de forma automatizada
- **c** El despliegue automatizado será posible si el anfitrión del nuevo logger permite conexiones desde otros equipos que encamina al contenedor adecuado
- **d** El despliegue automatizado será posible si el anfitrión de `cbw_ftcl` permite conexiones desde otros equipos que encamina a los contenedores adecuados

3 *La orden docker-compose:*

- **a** Todas las afirmaciones son ciertas.
- **b** Permite ejecutar aplicaciones docker multi-contenedor.
- **c** Permite ejecutar contenedores en máquinas remotas. Para ello se incluyen directivas para la conexión remota dentro del fichero de configuración.
- **d** Toma como entrada únicamente un fichero `Dockerfile` donde se especifican las relaciones entre las imágenes docker.

4 *En la práctica 3, en cuanto al cliente externo, no se puede desplegar conjuntamente con* `cbw_ftcl` *porque…*

- **a** En los equipos de escritorio LINUX no hay Docker instalado
- **b** El enunciado es falso: sí se puede desplegar conjuntamente
- **c** El cliente externo es incompatible con los clientes normales
- **d** El despliegue con docker-compose no puede afectar a varios anfitriones

5 *En la práctica 3 de laboratorio se ha construido una aplicación docker, donde una de las imágenes se encarga de registrar eventos en un fichero de log. A esta componente la hemos llamado* `logger`. *Señale la opción correcta.*

- **a** El fichero de log será visible en la máquina `host` dado que así lo hemos indicado en la configuración.
- **b** La componente logger utiliza un socket `rep` para recibir los mensajes de log.
- **c** Mediante docker-compose hemos conectado el resto de componentes al logger, haciendo que cada componente tenga acceso al fichero de log, de forma que todas las componentes pueden escribir en el mismo fichero concurrentemente.
- **d** El fichero de log que se genera puede corromperse debido a condiciones de carrera en el logger.

6 *Sobre el sistema con un componente* `worcli`*, el código suministrado acepta como parámetros:*

- **a** Dos url de dos brokers y el tipo de trabajo a procesar
- **b** Dos url de dos brokers, un retardo y el tipo de trabajo a procesar
- **c** Dos url de dos brokers y un retardo
- **d** Un url del broker

7 *Imagina que acabas de ejecutar sin errores la orden* `docker build -t provisional .`*, indica qué afirmación es cierta:*

- **a** Al ejecutar ahora `docker inspect provisional` puedo consultar su IP (casi al final de la información devuelta) entre otras cosas
- **b** El `Dockerfile` empleado seguro que incluye una orden `CMD` o `ENTRYPOINT`
- **c** Aunque cambie de directorio, es posible ejecutar `docker run provisional`
- **d** En el directorio actual, ahora existe un archivo llamado `provisional`

8 *Si deseamos conocer la dirección IP de un contenedor docker:*

- **a** Utilizaremos la orden `docker images`
- **b** Utilizaremos la orden `docker inspect`
- **c** Utilizaremos la orden `docker ps`
- **d** No se puede lograr esa información, pues no es una propiedad de la imagen docker.

9 *Al final del apartado 2 se sugiere reflexionar ante la posibilidad de desplegar* `cbw_ftcl` *en dos escenarios concretos. Selecciona la única afirmación cierta.*

- **a** En un despliegue con dos brokers, únicamente uno ellos será capaz de interactuar con workers, clients y logger
- **b** En un despliegue con dos loggers, docker-compose encontrará dificultades para asignar valores a la variable `LOGGER_URL` porque depende de `log`
- **c** En un despliegue con dos loggers, los mensajes del broker se repartirán alternativamente (round-robin) entre dichos loggers
- **d** Un despliegue con dos brokers no es posible porque ambos realizan bind en la misma URL

10 *Considerando el escenario del cliente externo (*`4_CBW_FTCL_CEXT`*) y el* `docker-compose.yml` *suministrado*

- **a** Es necesario eliminar la sección `expose:`
- **b** Es necesario añadir una sección `ports:` mapeando el puerto de clientes desde el anfitrión al contenedor (`9999:9999`)
- **c** Eliminando la sección `expose:` del `docker-compose.yml` es posible conectar al broker desde un cliente externo
- **d** El `docker-compose.yml` suministrado permite el acceso de un cliente externo al broker