Asignación

Primer código de estudiante, Apellido
Segundo código de estudiante, Apellido
Tercer código de estudiante, Apellido
Cuarto código de estudiante, Apellido

sistemas operativos
jefferson a.
Cali, 2024-II

Asignación

Descripción General

A continuación se detallan los lineamientos para una actividad teórico-práctica para los estudiantes de la carrera de Sistemas Operativos. La actividad evalúa específicamente el dominio de algunos conceptos básicos, definiciones relacionadas con la gestión de CPU, procesos, programación y la implementación de un simulador en C o C++.

Objetivos Durante el desarrollo de las actividades los estudiantes lograrán lo siguiente:

- Enumerar los hitos en la evolución de los sistemas informáticos.
- Describir los objetivos y funciones de los sistemas operativos modernos.
- Simular el comportamiento de la CPU y la gestión de procesos.
- Aplicar los principios de programación.

Antes de comenzar

Lea lo siguiente:

- Conceptos de sistemas operativos (novena edición) de Silberschatz y Galvin: capítulos 1, 2 y 3
- Sistemas operativos modernos de Tanenbaum: capítulos 1 y 2

Cree un repositorio git y agregue el enlace:

Utilice este documento como plantilla [aquí]

Actividad No. 1: Conceptualización [40%]

Después de completar las lecturas recomendadas en este documento, responda las siguientes preguntas con sus propias palabras, de manera concisa y precisa.

1. Enumerar los hitos en el evolución de los sistemas informáticos.
2. ¿Cuáles son los cuatro componentes de un sistema informático? Describe cada uno.
3. ¿Cuál es la diferencia entre un ¿núcleo monolítico y un micronúcleo?
4. Definir un Sistema Operativo desde dos perspectivas diferentes.
5. ¿Cuál es el propósito de las llamadas al sistema?
6. ¿Qué es un sistema operativo multiprogramado?
7. ¿Qué es un proceso?
8. ¿Cuáles son los estados de un proceso?
9. ¿Qué información se almacena en el ¿Bloque de control de procesos (PCB) asociado a un proceso?
10. ¿Cuáles son las principales actividades de una sistema operativo en relación con la gestión de procesos?

Actividad No. 2: Recordando los lenguajes C y C++ [50%]

1. Implementa los programas y funciones siguientes:

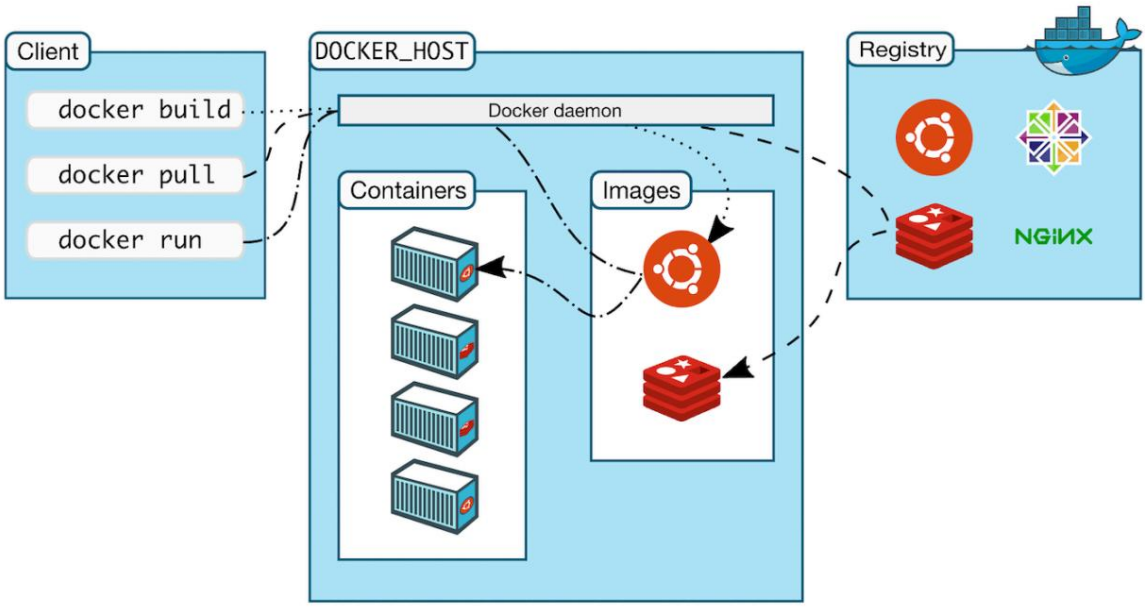
Write a program in c to find the leap year.
Write a progran to calculate factorial of a number.
Write a program in c to calculate power using recursion.
Write a program in c to find even or odd numbers.
Write a program in c to print fabonnaci series.
Write a Function to check uppercase letter.
Write a program in c to function to check lowercase letter
Find the greater of the three numbers
Write a program in c to type casting implicit explicit.
Write program to display number 1 to 10 in octal, decimal and hexadecimal system.

2. Escriba un programa en C que reciba y procese calificaciones para el curso de Sistemas Operativos (SO) usando estructuras.
3. Escriba un programa C++ que realice las siguientes tareas: Primero, el programa debe recibir un valor numérico y determinar si es un número primo, mostrando el resultado. En segundo lugar, el programa debería aceptar una lista de números e identificar los números primos dentro de esa lista. Por último, el programa debería permitir al usuario ingresar un rango numérico y mostrar todos los números primos dentro de ese rango especificado.
4. Cree un programa con C++ para modelar formas geométricas y realizar operaciones de área, perímetro y color. Se deben considerar formas de rectángulo, cuadrado, triángulo, etc. Consejo.
   Utilice clases y herencia.

Actividad No. 3: Sesión práctica de Docker [10%]

Docker es un software basado en línea de comandos que permite a los usuarios manipular imágenes y crear contenedores de aplicaciones.

Como se presentó en el curso, Docker consta de dos elementos: • un
cliente, para recibir comandos del usuario • un servidor,
para ejecutar comandos y administrar imágenes y contenedores



Arquitectura de comandos de Docker

Al escribir este comando, aparecerán las versiones de Cliente y Servidor disponibles en su computadora. Pegue una captura de pantalla del resultado

```
versión acoplable
```

Se puede acceder al uso, las opciones y una lista completa de comandos disponibles a través de la línea de comando en una terminal. Escriba el siguiente comando

```
ventana acoplable --ayuda
```

El uso general de una línea de comando de Docker es el siguiente:

```
ventana acoplable [OPCIONES] COMANDO [arg...]
```

Preguntas 1.

¿Cuántos argumentos requiere absolutamente el comando 'docker pull'?

2. ¿Recuerdas qué es un registro?

Descargue una imagen predefinida disponible en DockerHub

En un navegador web, navegue hasta DockerHub: https://hub.docker.com/ En la barra de búsqueda superior, escribe: japeto/pujgcc y pega una captura de pantalla.

Nuestro curso dispone de imágenes para el desarrollo de las sesiones prácticas.

Preguntas 1.

¿Cuántas veces se descargó la imagen del japeto ?

Ejecute el comando dentro de una terminal.

Pegue una captura de pantalla del resultado

ventana acoplable japeto/pujgcc:v0.12

Recibirá un error ya que esta imagen no tiene una etiqueta predeterminada (“más reciente”). Entonces necesitamos especificar uno en la línea de comando.

Vaya a la pestaña "Etiquetas" y copie el comando de extracción de la última versión

Pegue una captura de pantalla del resultado

ventana acoplable japeto/pujgcc:v0.12

Pregunta: 1.

¿Cuántas veces aparece "Extracción completa"? Por qué ?

Ahora, para asegurarnos de que la imagen se extrajo correctamente, veamos la lista de todas las imágenes descargadas disponibles dentro de nuestro espacio de trabajo. Pegue una captura de pantalla del resultado

imagen acoplable

Pregunta: 1.

¿Cuál es el tamaño de la imagen japeto/pujgcc?

Realizar una tarea usando una imagen extraída

Entre los comandos de Docker, ahora usaremos el comando 'ejecutar'.

Pregunta: 1.

¿Cuáles son las opciones y parámetros del comando 'ejecutar'?

ejecución de la ventana acoplable --ayuda

Como se muestra en la terminal, la descripción del comando es "Ejecutar un comando en un nuevo contenedor".

Pregunta: 1.

¿Cuál es la diferencia entre una imagen y un contenedor?

Ahora, para ejecutar la aplicación, ejecute el siguiente comando:

Pegue una captura de pantalla del resultado

ventana acoplable ejecuta japeto/pujgcc bash --ayuda

¡Felicidades!

¡Acabas de descargar y utilizar con éxito tu primera imagen de Docker!

Ejecutar PUJGCC sin parámetros fue interesante como demostración de las características de Docker.

Pero si realmente queremos ejecutar PUJGCC, también debemos proporcionar parámetros y, lo más importante, archivos de entrada.

Encuentre las rutas para vincular

Para vincular nuestra carpeta actual a la carpeta /data/ ubicada dentro de un contenedor, primero necesitamos la ruta absoluta de la carpeta actual, obtenida mediante el comando pwd de Unix.

Pegue una captura de pantalla del resultado

persona con discapacidad

Esta ruta se utilizará en más comandos a través de ${PWD}.

En lugar de ejecutar el comando ls en los archivos /home/, ahora simplemente enumeraremos el contenido de la carpeta /data/ dentro del contenedor pero lo vincularemos con el host.

Pegue una captura de pantalla del resultado

ventana acoplable ejecuta japeto/pujgcc:último ls /data

Si no aparece nada, es normal: la carpeta está vacía y sólo sirve como “punto de bifurcación”.

Ahora tenemos las rutas de las dos carpetas que queremos unir.

Vincular una carpeta local a un contenedor

Para realizar la asignación de carpetas entre la carpeta actual y /data dentro de la imagen, la sintaxis es simple. Pegue una captura de pantalla del resultado

docker run -v ${PWD}:/data/ japeto/pujgcc:latest ls /data/

Pregunta:

1. ¿La lista que se muestra es la misma que la que hay en su carpeta actual?

Finalmente, podemos ejecutar C en un archivo C o C++ ubicado en la carpeta Datos. Cambie el nombre del archivo a cualquiera de los archivos proporcionados.

Pegue una captura de pantalla del resultado

docker run -v ${PWD}:/data/ japeto/pujgcc:latest gcc /data/helloworld.c

Reiniciar y desconectar un contenedor

Aprenda a reutilizar un contenedor donde instaló algo

Utilice el comando de inicio para reiniciar el contenedor creado en el último ejercicio.

Pegue una captura de pantalla del resultado

inicio de la ventana acoplable -ti micontenedor /bin/bash

Regrese al contenedor usando el comando exec en lugar del comando ejecutar.

Pegue una captura de pantalla del resultado

docker exec -ti micontenedor /bin/bash

Pregunta: 1.

¿Qué sucede ahora cuando sales del contenedor? ¿Está detenido?

Verifique y pegue una captura de pantalla del resultado.

```
ventana acoplable ps-l
```

De hecho, el contenedor sigue funcionando. Esto se debe a que reiniciar un contenedor lo convierte en un "proceso independiente" que se ejecuta en segundo plano. Alternativamente, podríamos haber agregado la opción -d al primer comando de ejecución de Docker, creando directamente un contenedor separado.

Finalmente, puedes detener el contenedor.

```
Docker detiene mi contenedor
```

Este es el final de la sesión práctica. Esperamos que lo hayas disfrutado. ¡No dude en hacernos cualquier pregunta y no dude en contactarnos en cualquier momento después de la sesión!

Lista de comandos

Busque las versiones disponibles de una imagen en el registro de Docker:

```
búsqueda acoplable
```

Tirando de una imagen:

```
tirón de la ventana acoplable
```

Iniciar un contenedor en una imagen determinada ejecutando un solo comando:

```
ventana acoplable ejecutar -ti
```

Iniciar un contenedor en una imagen determinada ejecutando un solo comando (separado):

```
ejecución de la ventana acoplable -d
```

Enumere todos los contenedores y su estado.

```
ventana acoplable ps-l
```

Listar todas las imágenes extraídas

```
Imágenes de docker
```

Quitar un contenedor local

```
habitación acoplable
```

Eliminando una imagen local

```
acoplador rmi
```

Limpiar todos los contenedores

```
docker rm $(docker ps -aq)
```

Limpiar todas las imágenes (después de limpiar los contenedores)

```
ventana acoplable rmi $ (imágenes de ventana acoplable -aq)
```

Observaciones

- Las entregas deben realizarse en equipos de 4. Utilizando un repositorio público en github y un pdf informe
- Si no comprende las instrucciones de alguna de las actividades, no dude en consultarnos. jefferson.amado.pena@correounivalle.edu.co