LaSalle Universidad	**ESCUELA PROFESIONAL DE INGENIERÍA DE SOFTWARE**	**NOTA**

EXAMEN FINAL 2024-A

NOMBRE Y APELLIDOS DEL ESTUDIANTE	Jhosep Alonso Mollapaza Morocco	
CURSO	SISTEMAS OPERATIVOS	**SEMESTRE** : V
DOCENTE	MARIBEL MOLINA BARRIGA	**FECHA**: 09-05-2024 (HORA: 10:15 a.m.)
LUGAR	LABORATORIO B"	

PARTE I: Sincronización, Interbloqueos y Memoria principal, Programación en C o C++ (20 puntos)

1. Contestar las siguientes preguntas:

a. ¿Qué es la paginación y cómo ayuda en la gestión de la memoria?

La paginación es una técnica que divide la memoria física y lógica en bloques de tamaño fijo. Esto ayuda a la gestión de la memoria permitiendo que los procesos se carguen en áreas no contiguas de memoría física, facilita la implementación de la memoria virtual y mejora el uso eficiente del espacio en memoria.

b. ¿Cuáles son las condiciones necesarias para que ocurra un interbloqueo?

Las condiciones que se requieren llegan a ser que los recursos no puedan ser forzosamente quitados de otro proceso, que un proceso retenga recursos mientras este espera adquirir otros y que al menos un recurso no debe ser compartible.

2. Realice el siguiente código:

```
/*Ejercicio Matriz */
# include <stdio.h>
# include <stdlib.h>
# include <string.h>
# include <unistd.h>
# include <pthread.h>

struct parametros {
int id;
float escalar ;
float matriz [3][3];
};
void init (float m [3][3]) {
int i;
int j;
for ( i = 0 ; i < 3 ; i++ ) {
for ( j = 0 ; j < 3 ; j++ ) {
m[i][j] = random () *100;
  }
 }
```

```
}
void * matrizporescalar( void *arg ) {
struct parametros *p;
int i;
int j;
p = (struct parametros *) arg ;
for ( i = 0 ; i < 3 ; i++ ) {
printf (" Hilo %d multiplicando fila %d\n", p -> id , i);
for ( j = 0 ; j < 3 ; j++ ) {
p -> matriz [i][j] = p -> matriz [i][j] * p -> escalar ;
sleep (5) ;
  }
 }
}
int main(int argc , char *argv []) {
   pthread_t h1;
   struct parametros p1;
   p1.id = 1;
   p1.escalar = 5.0;

   init (p1.matriz );
   pthread_create (&h1 , NULL , matrizporescalar , ( void *)&p1);
   pthread_join(h1 , NULL);
   printf ("Fin \n");
}
```

a) Analice, comente y Ejecute el código entregado

```c
/* Ejercicio Matriz */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>

// Estructura para pasar parámetros al hilo
struct parametros {
    int id;           // Identificador del hilo
    float escalar;    // Valor escalar para multiplicar
    float matriz[3][3]; // Matriz 3x3 a modificar
};

// Función para inicializar la matriz con valores aleatorios
void init(float m[3][3]) {
    int i, j;
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++) {
            m[i][j] = random() * 100; // Genera valores aleatorios entre 0 y 100
        }
    }
}

// Función que ejecutará el hilo
void *matrizporescalar(void *arg) {
    struct parametros *p; // Declara un puntero a la estructura de parámetros
    int i, j; // Variables de los bucles
    p = (struct parametros *)arg; // Convierte el argumento void a un puntero

    for (i = 0; i < 3; i++) { // Bucle que itera sobre las filas de la matríz
        printf("Hilo %d multiplicando fila %d\n", p->id, i); // Imprime mensaje indicando que fila se procesa
        for (j = 0; j < 3; j++) { // Bucle que itera sobre las columnas de la matriz
            p->matriz[i][j] = p->matriz[i][j] * p->escalar; // Multiplica el elemento actual de la matriz
            sleep(5); // Pausa la ejecución del hilo durante 5 segundos
        }
    }
    return NULL;
}

int main(int argc, char *argv[]) {
    pthread_t h1;            // Es la declaración del hilo
    struct parametros p1;    // Estructura para pasar parámetros al hilo
```

```
45      p1.id = 1;                  // Se asigna ID 1 al hilo
46      p1.escalar = 5.0;           // Define el escalar como 5.0
47      init(p1.matriz);            // Inicializa la matriz con valores aleatorios
48
49      // Crea el hilo y le pasa la función a ejecutar y los parámetros
50      pthread_create(&h1, NULL, matrizporescalar, (void *)&p1);
51
52      // Espera a que el hilo termine su ejecución
53      pthread_join(h1, NULL);
54
55      printf("Fin \n");           // Imprime mensaje de finalización
56      return 0;
57  }
```

b) Describa qué actividad realiza el código que se muestra.

El código crea una matriz 3x3 con valores aleatorios, luego este utiliza un hilo para multiplicar cada elemento por un escalar que es 5.0 en este caso. El proceso se realiza fila por fila con retardo de 5 segundos entre cada elemento.

c) Verifique en el código si existe una función que sea invocada por el hilo principal para mostrar el contenido de la matriz antes y después de su modificación.

No existe en el código una función que muestre el contenido de la matriz antes y después de su modificación.

Enlace de Repositorio: https://onlinegdb.com/Z3ZkpsCZZ

PARTE II: Investigación sobre virtualización de sistema operativos (20 puntos)

Adjuntar actividades realizadas:

Actividad 01: Búsqueda de artículos indexados sobre el tema. Precisar a)¿Qué realizan en el artículo?, b) ¿Qué resultados obtuvieron? c) ¿Cuál es el aporte para su estudio?. Adjuntar el PDF de los artículos encontrados.

Actividad 02: Realizar virtualización del sistema operativo en modo consola. Realizar instalaciones: gcc, g++, screenfetch, vim y ejecutar su última versión de Filósofos comensales en la máquina virtual creada por ud.

Actividad 03: Realizar ejecución de GUI (Interfaz gráfica de usuario desde una máquina virtual. Ejecutar cualquier programa en modo gráfico: editor de texto, navegador web o gestor de escritorio.

CPU asignada a la Máquina virtual	4 Procesadores
Memoria asignada a la máquina virtual	4 GB de Ram
Tamaño en disco de la Máquina virtual	80 GB de espacio

Actividad 04: Realizar una comunicación de dos máquinas virtuales conectadas en red. Probar conectividad.

Entregables finales

Artículo

Presentación de Diapositivas, realizando una exposición.

Enlace de Repositorio:

https://drive.google.com/drive/folders/1R_GNYhsrLUdBWwDI29dMr-2YsBqU9APS?usp=drive_link