PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

SISTEMAS OPERATIVOS
Examen 1
(Segundo semestre de 2022)

Horarios 0781, 0782: prof. V. Khlebnikov

Duración: 3 horas
Notas: No está permitido el uso de ningún material o equipo electrónico.
La presentación, la ortografía y la gramática influirán en la calificación.
Puntaje total: 20 puntos

Pregunta 1 (3 puntos – 24 min.)

(a) (1 punto) In general, the term *preemption* is defined to be the reclaiming of a resource from a process before the process has finished using it. Suppose, for example, that process A is running at a given priority level, and process B, at a higher priority level, is blocked. If the OS learns that the event upon which process B has been waiting has occurred, moving B to a ready state, then process A can be preempted. By who? Who realize the preemption and what is a resource to reclaim?

(b) (1 punto) ¿Qué hay en común y cuál es la diferencia entre los estados **Blocked/Suspend** y **Ready/Suspend**?

(c) (1 punto) ¿Qué contiene un imagen de proceso?

Pregunta 2 (5 puntos – 40 min.) Analice el siguiente programa y explique el comportamiento de cada proceso involucrado.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <sys/stat.h>

void die(char *s);

int
main(void)
{
    int i=26, p, a='a';
    FILE *f; struct stat sb;

    if (unlink("f") < 0 && errno != ENOENT) die("unlink error\n");
    while(i-- && (p=fork())) a++;
    while(!p) {
        f = fopen("f", "a");
        fprintf(f, "%c", a);
        fclose(f);
        if (!i) break;
        exit(0);
    }
    if (p) exit(0);
    while(1) {
        stat("f", &sb);
        if (sb.st_size == 26) break;
    }
    execl("/bin/cat", "cat", "f", (char *)0);
    die("exec error\n");
}

void die(char *s)
{
  if ( s != (char *) NULL ) { while (*s) (void) write(2, s++, 1); }
  exit( (s == (char *) NULL) ? 0 : 1 );
}
```

Pregunta 3 (2 puntos – 16 min.) (*OSIDP7E*) Consider the following program:

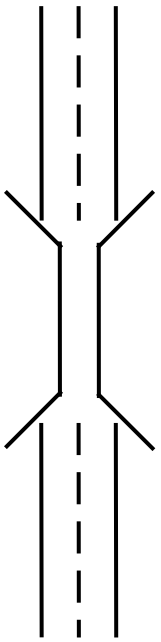
```
 A: {                                    B: {
      shared int x;                           shared int x;

1     x = 10;                           1     x = 10;
2     while (1) {                       2     while (1) {
3         x = x - 1;                    3         x = x - 1;
4         x = x + 1;                    4         x = x + 1;
5         if (x != 10)                  5         if (x != 10)
6             printf("x is %d", x);     6             printf("x is %d", x);
      }                                       }
     }                                       }
```

Note that the scheduler in a uniprocessor system would implement pseudo-parallel execution of these two concurrent processes by interleaving their instructions, without restriction on the order of the interleaving.

a) Show a sequence (i.e., trace the sequence of interleaving of statements) such that the statement "`x is 10`" is printed.
b) Show a sequence such that the statement "`x is 8`" is printed. You should remember that the increment/decrements at the source language level are not done atomically.

Pregunta 4 (6 puntos – 48 min.) (*East-West Bridge Problem*) Es un puente donde los coches pueden pasar en una sola dirección y no se permiten que pasen más de 3 coches simultáneamente en una dirección. La solución propuesta es una variación del problema de los lectores y los escritores. Aquí se proporciona el código para el proceso del lector. El código del proceso del escritor es simétrico, solamente cambie las ocurrencias correspondientes de la letra `r` por `w` y `w` por `r`. Explique cómo funciona este algoritmo (y no el código en sí) cuando hay coches (diferentes cantidades) de un lado, de ambos lados, etc. Su explicación debe ser clara para un no profesional, o sea, no puede limitarse a la presentación de la secuencia de las sentencias ejecutadas. Explique el papel de cada semáforo y de las variables usadas.

```
 1      P(lock);
 2      if ((nw_active + nw_waiting == 0) && (nr_active < 3)) {
 3        nr_active++;
 4        V(r_sem);
 5      } else nr_waiting++;
 6      V(lock);
 7      P(r_sem);
 8      READING... (west to east crossing bridge)
 9      P(lock);
10      nr_active--;
11      if ((nr_active == 0) && (nw_waiting > 0)) {
12        count = 0;
13        while ((nw_waiting > 0) && (count < 3)) {
14          V(w_sem);
15          nw_active++;
16          nw_waiting--;
17          count++;
18        }
19      } else if ((nw_waiting == 0) && (nr_waiting > 0)) {
20               V(r_sem);
21               nr_active++;
22               nr_waiting--;
23             }
24      V(lock);
```



Pregunta 5 (4 puntos – 32 min.) (*William Stallings*) Considere un sistema de un dron que recolecta y procesa datos de dos sensores: G-sensor que proporciona la aceleración en 3 ejes y GPS que proporciona las coordenadas. Los datos deben ser recogidos del 1er sensor cada 20 ms y del 2do sensor cada 50 ms. Cada porción de datos del 1er sensor se procesa en 10 ms, incluyendo el trabajo del sistema operativo, y en 25 ms se procesa cada porción de datos recogidos del 2do sensor. El dispositivo es capaz de tomar la decisión sobre planificación de procesos cada 10 ms. Se puede aplicar un esquema de planificación por prioridad asignando a la tarea de G-sensor o a la tarea de GPS una mayor prioridad. La tercera opción es primero atender la tarea que tenga su plazo (*deadline*) más cercano. Intentando procesar cada porción de datos de ambos sensores, ¿cuál de los tres esquemas proporciona el mejor resultado? ¿Cuál es el peor? Presente los diagramas de planificación.

Preparado por VK
con LibreOffice Writer en Linux Mint 21 "Vanessa"

Profesor del curso: V. Khlebnikov

Lima, 12 de octubre de 2022