Primer Parcial Mayo 2013

Lea detenidamente las siguientes instrucciones. No cumplir los requerimientos puede implicar la pérdida de los puntos del parcial.

Formato

- **Indique su nombre completo y número de cédula en cada hoja (No se corregirán las hojas sin nombre, sin excepciones).**
- **Numere todas las hojas e indique la cantidad total de hojas que entrega en la primera.**
- **Escriba las hojas de un solo lado y empiece cada problema en una hoja nueva.**
- **Si se entregan varias versiones de un problema solo se corregirá el primero de ellos según el orden de hojas.**

Dudas

- **Sólo se contestarán dudas de letra.**
- **No se aceptarán dudas en los últimos 15 minutos del parcial.**

Material

- **El examen es SIN material (no puede utilizarse ningún apunte, dispositivo móvil, libro ni calculadora). Sólo puede tenerse las hojas del examen, lápiz, goma y lapicera en su banco. Todas sus demás pertenencias debe colocarlas en el piso debajo de su asiento.**

Finalización

- **El parcial dura 2 horas.**

Problema 1 (15 puntos)

1. Indique tres métodos posibles para realizar una operación de E/S con una controladora
2. ¿Para qué sirve la protección de memoria? ¿Quién realiza este control?
3. Describa brevemente el proceso de cambio de contexto de la cpu
4. Describa el modelo the threads MxN (many to many)
5. Indique una posible estategia de scheduling para usar en un sistema de tiempo real estricto (hard) y justifique la respuesta

Problema 2 (16 puntos) (10,6)

Se tiene un sistema operativo simétrico multiprogramado con un planificador con dos colas de prioridad (alta/baja) con retroalimentación (multi-level feedback queue con dos niveles de prioridad). En cada una de las dos colas se utiliza un algoritmo round-robin, en la cola de alta prioridad se utiliza un quantum de 80ms y la de baja prioridad un quantum de 40ms.

Se sabe que el sistema cumple con las siguientes características:

- Los procesos al ser creados se les asigna la prioridad alta.
- Un proceso sube a la cola de mayor prioridad si en los últimos 100ms no ha usado el recurso procesador.
- Un proceso baja a la cola de menor prioridad si en los últimos 80ms ha utilizado completamente el recurso procesador.
- Nunca debe detenerse o retrasarse la ejecución de un proceso con alta prioridad si solo existen en el sistema procesos con baja prioridad.
- Cuando un proceso pasa a un estado listo, luego de estar bloqueado, su quantum es reiniciado según su prioridad.

Sean los procesos P1 y P2 creados en el instante t=0ms que ejecutan lo siguiente:

- Ejecutan tarea de cómputo durante 50ms.
- Se boquean durante 100ms.
- Ejecutan tarea de cómputo durante 150ms.

Sea el procesos P3 creado en el instante t=20ms que ejecuta lo siguiente:

- Ejecuta tarea de cómputo durante 120ms
- Se bloquea durante 50ms
- Ejecuta tarea de cómputo durante 100ms

1) Asumiendo un sistema mono-procesador, realice un esquema o diagrama de planificación (tiempo vs. procesos), en el que se indique el estado de cada proceso (listo/ejecutando/bloqueado/terminado) para cada intervalo de tiempo. Además, se debe indicar, en cada caso, el nivel de prioridad (alta/baja) en el que se encuentra el proceso.
2) Calcule las siguientes métricas para la parte anterior
   a) Tiempo de espera del proceso P1
   b) Tiempo de retorno del proceso P2
   c) Tiempo de respuesta del proceso P3

Solución Problema 2:

1)

El siguiente diagrama muestra estado/prioridad de los proceso en función del tiempo

P3		L/A	L/A	E/A	E/A	L/B	L/B	L/B	L/A	E/A	E/A
P2	L/A	L/A	E/A	B/A	B/A	B/A	L/A	E/A	E/A	L/B	L/B
P1	E/A	E/A	B/A	B/A	L/A	E/A	E/A	L/B	L/B	L/B	L/A

0 20 50 100 150 180 200 260 280 340 360 380

P3	B/A	L/A	L/A	E/A	L/B	E/B	fin	
P2	L/B	L/B	L/A	L/A	E/A	fin		
P1	E/A	E/A	E/A	fin				

380 430 440 450 530 600 620

2)

a)

Tiempo de espera (Waiting time): Es la suma de los intervalos de tiempo que un proceso estuvo en la cola de procesos listos (ready queue).

El proceso P1 esta en la cola de procesos listos en los intervalos: 150ms a 180ms y 260ms a 380ms

Tiempo de espera P1 = (180ms – 150ms) + (380ms – 260ms) = 30ms + 120ms = 150ms

b)

Tiempo de retorno (Turnaround time): Es el intervalo de tiempo desde que un proceso es cargado hasta que este finaliza su ejecución.

P2 se cargado en el sistema en t = 0ms y finaliza su ejecución en t = 600ms

Tiempo de retorno P2 = 600ms – 0ms = 600 ms

c)

Tiempo de respuesta (Response time): Es el intervalo de tiempo desde que un proceso es cargado hasta que brinda su primer respuesta.

P3 es cargado en el intante t = 20 ms y su primera tareas las realiza en el intante t = 100ms

Tiempo de respuesta P3 = 100ms – 20ms = 80 ms

****

Problema 3 (7 puntos) (3,4)

1. Corregir los procedimientos generador y controlador usando semáforos de forma de asegurar la consistencia del siguiente programa. Se asume que el pasaje de parámetros a funciones se realiza por valor y que las funciones son externas y no usan las variables globales del programa.

```
program ejercicio3.1
int a, c;

procedure generador()
int b;
begin
      loop
            b := genero(); // genera dato en b
            a := b*2;
            if a = 8000 then
                  print('Listo');
                  exit;
            end
      end
end

procedure controlador()
int b;
begin
      loop
            b := controlo(a); // devuelve dato de control según el
                              // parámetro a
            c := b*2;
            if c = 8000 then
                  print('Controlado');
                  exit;
            end
      end
end
begin
      a := 1;
      c := 1;
      cobegin
            generador();
            controlador();
      coend
end.
```

2. Indique las posibles salidas de este programa:

```
        a := 2;
        cont := 2;
        print(a);
        fork L1;
        a := 5;
        print(a);
        goto L2;
L1:     a := 7;
L2:     join cont,L3;
        quit;
L3:     a := a+3;
        print(a);
```

Solución Problema 3:

1)

```
program ejercicio3.1
      int a, c;
      semaphore mutexA;

      procedure generador()
      int b;
      begin
            loop
                  b := genero(); // genera dato en b
                  P(mutexA);
                  a := b*2;
                  V(mutexA);
                  if a = 8000 then
                        print('Listo');
                        exit;
                  end
            end
      end
```

```
procedure controlador()
int b;
begin
      loop
            P(mutexA);
            b := controlo(a); // devuelve dato de control según el
                              // parámetro a
            V(mutexA);
            c := b*2;
            if c = 8000 then
                  print('Controlado');
                  exit;
            end
      end
end

begin
      a := 1;
      c := 1;
      init(mutexA, 1);
      cobegin
            generador();
            controlador();
      coend
end.
```

2)

Las salidas posibles son:

(a) 2,5,8 – (b) 2,5,10 – (c) 2,7,10

- El primer número que se imprime siempre es 2 porque se ejecuta antes de realizar el fork.

- El segundo número que se imprime puede ser un 5 o un 7, dependiendo de cual haya sido la última asignación a la variable a previo a la sentencia que la imprime.

- El último número que se imprime puede ser un 8 o un 10:

Se imprime un 8 en caso en que en la parte concurrente se haya asignado primero 7 y luego 5 previo al join, en dicho caso queda (a)2,5,8.

Se imprime un 10 en caso en que en la parte concurrente se haya asignado primero 5 y luego 7 previo al join, en dicho caso puede quedar (b)2,5,10 o (c)2,7,10 dependiendo del valor de a previo al segundo print.