MANUEL UTRERA GISBERT
31266016T
UNED CADIZ

SISTEMAS OPERATIVOS

Primera Prueba de Evaluación a Distancia (PED1)

1. Explique **razonadamente** si las siguientes afirmaciones son verdaderas o falsas:

I) *(1 p)* Se denomina *sobrecarga del sistema* al número de procesos que se encuentran en el estado bloqueado en espera de que se produzca un determinado evento.

FALSA. Se denomina sobrecarga del sistema al tiempo que el procesador se encuentra ocupado ejecutando código del sistema operativo asociado a tareas y servicios de administración que no se pueden contabilizar a ningún proceso en particular.

II) *(1 p)* El *planificador a largo plazo* de un sistema operativo da más prioridad a los trabajos limitados por CPU que a los trabajos limitados por E/S.

FALSA. El planificador a largo plazo debe intentar seleccionar trabajos de tal forma que en el sistema exista una mezcla adecuada de procesos limitados por CPU y procesos limitados por E/S. Si la mayoría de los trabajos seleccionados están limitados por CPU, entonces los dispositivos de E/S estarían la mayor parte del tiempo inactivos. Por el contrario, si la mayoría de los trabajos seleccionados están limitados por E/S, sería el procesador el que estaría la mayor parte del tiempo inactivo.

III) *(1 p)* El micronúcleo de un sistema operativo con estructura extensible se encarga, entre otras tareas, de la gestión de la memoria virtual.

FALSA. El micronúcleo de un sistema operativo con estructura extensible se encarga de realizar únicamente los servicios absolutamente esenciales del sistema operativo, aquellos que dependen de la arquitectura de la máquina y que son independientes del tipo de sistema operativo, como por ejemplo, la gestión de memoria a bajo nivel, la comunicación entre procesos, la gestión de la E/S y la gestión de las interrupciones. La gestión de la memoria virtual se implementa como extensiones del núcleo.

2. Una sala de exposiciones tiene una capacidad para V visitantes y dispone de un guía. Si la sala esta llena los visitantes deben esperar en una cola para entrar. Una vez dentro de la sala se distinguen dos tipos de visitantes: los visitantes guiados y los visitantes libres. Un visitante guiado cuando entra en la sala se dirige a un punto de encuentro existente donde espera junto con otros visitantes a que el guía les avise para iniciar la visita guiada. El visitante libre realiza la visita por su cuenta. Ambos tipos de visitantes cuando terminan la visita abandonan la sala. El guía si no hay nadie en el punto de encuentro descansa sentado en una silla, solo cuando hay 10 visitantes en el punto de encuentro se levanta y avisa a los visitantes allí presentes para que le sigan. Cuando termina una visita guiada vuelve a su silla y descansa al menos 7 minutos antes de iniciar otra visita guiada. Escribir el pseudocódigo de un programa que coordine la actividad de los visitantes y del guía en la sala de exposiciones usando:

a) *(1 p)* Semáforos generales.

b) *(1.5 p)* Semáforos binarios.

c) *(1.5 p)* Paso de mensajes, suponer que la comunicación es indirecta a través de buzones y que se dispone de la operación `send` sin bloqueo y de la operación `receive` con bloqueo.

El pseudocódigo del programa que se realice en cada apartado debe tener cinco partes: declaración de variables, código visitante guiado, código visitante libre, código del guía, y código para inicializar los elementos de sincronización (semáforos o cola de mensajes) y lanzar la ejecución concurrente de los procesos.

```
PROGRAMA sala_de_exposiciones;

semaforo_binario  cont, dispo, encu, avisoguia, finvisita;
int contador=0, encuentro=0;
#define TRUE 1

void visitante_libre()
{
     wait_sem(cont);
     contador=contador+1;
     if (contador >50)
     {
          signal_sem(cont);
          wait_sem(dispo);
     }
     else signal_sem(cont);

     entrar();
     visita_libre();
     signal_sem(dispo);
     wait_sem(cont);
     contador=contador-1;
     signal_sem(cont);
     salir();

}
```

```
void visitante_guiado()
{
     wait_sem(cont);
     contador=contador+1;
     if (contador >50)
     {
          signal_sem(cont);
          wait_sem(dispo);
     }
     else signal_sem(cont);

     entrar();
     ir_puntoencuentro();
     wait_sem(encu);
     encuentro=encuentro+1;
     signal_sem(encu);
     wait_sem(avisoguia);
     visita_guiada();
     wait_sem(finvisita);
     signal_sem(dispo);
     wait_sem(cont);
     contador=contador-1;
     signal_sem(cont);
     salir();
}



void guia()
{
     while TRUE
     {
          if  (encuentro>=10)
          {
               levantarse();
               wait_sem(encu);
               encuentro=0;
               signal_sem(encu);
               signal_sem(avisoguia);
               guía_visita();
               signal_sem(finvisita);
               vuelve_silla();
               descansa_7min();
          }
          else  descansa();
     }
}
```

```
void main()
{
     init_sem(cont,1);
     init_sem(dispo,0);
     init_sem(encu,1);
     init_sem(avisoguia,0);
     init_sem(finvisita,0);
     ejecucion_concurrente(visitante_libre, … ,visitante_libre,
     visitante_guiado, … ,    visitante_guiado , guia);
}
```