Conceptos de Sistemas Operativos (IC)
Introducción a los Sistemas Operativos (LI, LS, APU, ATIC)

Trabajo Práctico N° 2

Objetivo

Comprender conceptos básicos de los sistemas operativos, los procesos y su planificación para la administración de la CPU. Introducir y comprender la administración de memoria, considerando las diferentes posibilidades acordes a las características del hardware, distinguiendo el apoyo de este así como las actividades a realizar para la eficiente administración del recurso por parte del Kernel.

Se recomienda, para la autocorrección de los ejercicios, la utilización del simulador de planificación que se publicará en la plataforma.

Temas incluidos

Definición de SO. Componentes de un SO. Apoyo del Hardware: Modos de ejecución, interrupciones. Llamadas al sistema. Programa y Proceso. Planificación de CPU. Colas de planificación. Context Switch. Creación de procesos. Espacio de direcciones. Direcciones lógicas y físicas. Particiones fijas y dinámicas. Paginación y Segmentación. Fragmentación. MMU. Algoritmos de administración.

1. Responda en forma sintética sobre los siguientes conceptos:
    a. ¿Qué es un Sistema Operativo?
    b. Enumere qué componentes/aspectos del Hardware son necesarios para cumplir los objetivos de un Sistema Operativo.
    c. Enumere componentes de un Sistema Operativo
    d. ¿Que es una llamada al sistema (*system call*)? ¿Cómo es posible implementarlas?
    e. Defina y diferencie Programa y Proceso.
    f. ¿Cuál es la información mínima que el Kernel debe tener sobre un proceso? ¿En qué estructura de datos asociada almacena dicha información?
    g. ¿Qué objetivos persiguen los algoritmos de planificación (scheduling).
    h. ¿Qué significa que un algoritmo de scheduling sea apropiativo o no apropiativo (Preemptive o Non-Preemptive)?
    i. ¿Qué tareas realizan los siguientes módulos de planificación?:
        i. Short Term Scheduler
        ii. Long Term Scheduler
        iii. Medium Term Scheduler
    j. ¿Qué tareas realiza el Dispatcher? ¿Y el Loader?
    k. ¿Qué significa que un proceso sea “CPU Bound” y “I/O Bound”?
    l. ¿Cuáles son los estados posibles por los que puede atravesar un proceso? ¿Qué representa que un proceso se encuentre en los estados

enumerados? Utilizando un diagrama explique las transiciones entre los estados.

m. ¿Cuáles de los schedulers mencionados anteriormente se encargan de las transiciones entre los estados enumerados?
n. Defina Tiempo de retorno (**TR**) y Tiempo de espera (**TE**) para un proceso.
o. Defina Tiempo Promedio de Retorno (**TPR**) y Tiempo promedio de espera (**TPE**) para un lote de procesos.
p. Defina tiempo de respuesta.

2. Para los siguientes algoritmos de scheduling:
   - ➢ FCFS (First Come First Served)
   - ➢ SJF (Shortest Job First)
   - ➢ Round Robin
   - ➢ Prioridades

   a. Explique su funcionamiento mediante un ejemplo.
   b. ¿Alguno de ellos cuentan con parámetros para su funcionamiento? Identifique y enunciarlos
   c. Cual es el más adecuado según los tipos de procesos y/o SO.
   d. Cite ventajas y desventajas de su uso.

3. Dado el siguiente lote de procesos en el que todos arriban al sistema en el instante 0 (cero):

Job	Unidades de CPU
1	7
2	15
3	12
4	4
5	9

   a. Realice los diagramas de Gantt según los siguientes algoritmos scheduling:
      i. FCFS (First Come, First Served)
      ii. SJF (Shortest Job First)
      iii. Round Robin con quantum = 4
   b. Para cada algoritmo calcule el TR y TE para cada job así como el TPR y el TPE.
   c. En base a los tiempos calculados compare los diferentes algoritmos.

4. Dado el siguiente lote procesos:

Job	Llegada	Unidades de CPU
1	0	4
2	2	6
3	3	4

4	6	5
5	8	2

a. Realice los diagramas de Gantt según los siguientes algoritmos de scheduling:
   i. FCFS (First Come, First Served)
   ii. SJF (Shortest Job First)
   iii. Round Robin con quantum = 1
   iv. Round Robin con quantum = 6
b. Para cada algoritmo calcule el TR y TE para cada job así como el TPR y el TPE.
c. En base a los tiempos calculados compare los diferentes algoritmos.
d. En el algoritmo Round Robin, qué conclusión se puede sacar con respecto al valor del quantum.
e. ¿Para el algoritmo Round Robin, en qué casos utilizará un valor de quantum alto y qué ventajas y desventajas obtendría?

5. Una variante al algoritmo SJF es el algoritmo SJF apropiativo o SRTF (Shortest Remaining Time First):
   a. Realice el diagrama de Gantt para este algoritmo según el lote de trabajos del ejercicio 5.
   b. ¿Nota alguna ventaja frente a otros algoritmos?

6. Suponga que se agregan las siguientes prioridades al lote de procesos del ejercicio 5, donde un menor número indica mayor prioridad:

Job	Prioridad
1	3
2	4
3	2
4	1
5	2

   a. Realice el diagrama de Gantt correspondiente al algoritmo de planificación por prioridades según las variantes:
      i. No Apropiativa
      ii. Apropiativa
   b. Calcule el TR y TE para cada job así como el TPR y el TPE.
   c. ¿Nota alguna ventaja frente a otros algoritmos? ¿Bajo qué circunstancias lo utilizaría y ante qué situaciones considera que la implementación de prioridades podría no ser de mayor relevancia?

7. Inanición (*Starvation*)
   a. ¿Qué significa?

b. ¿Cuál/es de los algoritmos vistos puede provocarla?
c. ¿Existe alguna técnica que evite la inanición para el/los algoritmos mencionados en el inciso b?

8. Los procesos, durante su ciclo de vida, pueden realizar operaciones de I/O como lecturas o escrituras a disco, cintas, uso de impresoras, etc. El Kernel mantiene para cada dispositivo, que se tiene en el equipo, una cola de procesos que espera por la utilización del mismo (al igual que ocurre con la Cola de Listos y la CPU, ya que la CPU es un dispositivo más).

Cuando un proceso en ejecución realiza una operación de I/O el mismo es expulsado de la CPU y colocado en la cola correspondiente al dispositivo involucrado en la operación.
El Kernel dispone también de un "I/O Scheduling" que administra cada cola de dispositivo a través de algún algoritmo (FCFS, Prioridades, etc.). Si al colocarse un proceso en la cola del dispositivo, la misma se encuentra vacía el mismo será atendido de manera inmediata, caso contrario, deberá esperar a que el Kernel lo seleccione según el algoritmo de scheduling establecido.
Los mecanismos de I/O utilizados hoy en día permiten que la CPU no sea utilizada durante la operación, por lo que el Kernel puede ejecutar otro proceso que se encuentre en espera una vez que el proceso bloqueado por la I/O se coloca en la cola correspondiente.
Cuando el proceso finaliza la operación de I/O el mismo retorna a la cola de listos para competir nuevamente por la utilización de la CPU.
Para los siguientes algoritmos de Scheduling:

- FCFS
- Round Robin con quantum = 2

Y suponiendo que la cola de listos de todos los dispositivos se administra mediante FCFS, realice los diagramas de Gantt según las siguientes situaciones:

a. Suponga que al lote de procesos del ejercicio 5 se agregan las siguientes operaciones de entrada salida:

Job	I/O (Recurso,Instante,Duración)
1	(R1, 2, 1)
2	(R2, 3, 1) (R2, 5, 2)
4	(R3, 1, 2) (R3, 3, 1)

b. Suponga que al lote de procesos del ejercicio 5 se agregan las siguientes operaciones de entrada salida:

Job	I/O (Recurso,Instante,Duración)
1	(R1, 2, 3) (R1, 3, 2)
2	(R2, 3, 2)
3	(R2, 2, 3)

4	(R1, 1, 2)

9. Algunos algoritmos pueden presentar ciertas desventajas cuando en el sistema se cuenta con procesos ligados a CPU y procesos ligados a entrada salida. Analice las mismas para los siguientes algoritmos:
    a. Round Robin
    b. SRTF (Shortest Remaining Time First)

10. Para equiparar la desventaja planteada en el ejercicio 10, se plantea la siguiente modificación al algoritmo:

Algoritmo VRR (Virtual Round Robin): Este algoritmo funciona igual que el Round Robin, con la diferencia que cuando un proceso regresa de una I/O se coloca en una cola auxiliar. Cuando se tiene que tomar el próximo proceso a ejecutar, los procesos que se encuentran en la cola auxiliar tienen prioridad sobre los otros. Cuando se elije un proceso de la cola auxiliar se le otorga el procesador por tantas unidades de tiempo como le faltó ejecutar en su ráfaga de CPU anterior, esto es, se le otorga la CPU por un tiempo que surge entre la diferencia del quantum original y el tiempo usado en la última ráfaga de CPU.

    a. Analice el funcionamiento de este algoritmo mediante un ejemplo. Marque en cada instante en que cola se encuentran los procesos.
    b. Realice el ejercicio 9)a) nuevamente considerando este algoritmo, con un quantum de 2 unidades

11. Suponga que un Kernel utiliza un algoritmo de VRR para planificar sus procesos. Para ello, el quantum es representado por un contador, que es decrementado en 1 unidad cada vez que ocurre una interrupción de reloj. ¿Bajo este esquema, puede suceder que el quantum de un proceso nunca llegue a 0 (cero)? Justifique su respuesta.

12. El algoritmo SJF (y SRTF) tiene como problema su implementación, dada la dificultad de conocer la duración de la próxima ráfaga de CPU. Es posible realizar una estimación de la próxima, utilizando la media de las ráfagas de CPU para cada proceso. Así, por ejemplo, podemos tener la siguiente fórmula (1):

$$S_{n+1} = \frac{1}{n} T_n + \frac{n-1}{n} S_n$$

Donde:

T_i = duración de la ráfaga de CPU i-ésima del proceso.

S_i = valor estimado para el i-ésimo caso

S_i = valor estimado para la primera ráfaga de CPU. No es calculado.

    a. Suponga un proceso cuyas ráfagas de CPU reales tienen como duración: 6, 4, 6, 4, 13, 13, 13. Calcule qué valores se obtendrían como estimación para las ráfagas de CPU del proceso si se utiliza la fórmula 1, con un valor inicial

estimado de S_1=10.

La fórmula 1 le da el mismo peso a todos los casos (siempre calcula la media). Es posible reescribirla permitiendo darle un peso mayor a los casos más recientes y menor a casos viejos (o viceversa). Se plantea la siguiente fórmula 2:

$$S_{n+1} = \alpha T_n + (1 - \alpha) S_n$$

Con $0 < \alpha < 1$.

b. Analice para qué valores de α se tienen en cuenta los casos más recientes.
c. Para la situación planteada en a) calcule qué valores se obtendrían si se utiliza la fórmula 2 con $\alpha = 0,2$; $\alpha = 0,5$ y $\alpha = 0,8$.
d. Para todas las estimaciones realizadas en a y c ¿Cuál es la que más se asemeja a las ráfagas de CPU reales del proceso?

13. Colas Multinivel

Actualmente los algoritmos de planificación vistos se han ido combinando para formar algoritmos más eficientes. Así surge el algoritmo de Colas Multinivel, donde la cola de procesos listos es dividida en varias colas, teniendo cada una su propio algoritmo de planificación.

a. Suponga que se tiene dos tipos de procesos: *Interactivos* y *Batch*. Cada uno de estos procesos se coloca en una cola según su tipo. ¿Qué algoritmo de los vistos utilizará para administrar cada una de estas colas?.

A su vez, se utiliza un algoritmo para administrar cada cola que se crea. Así, por ejemplo, el algoritmo podría determinar mediante prioridades sobre qué cola elegir un proceso.

b. Para el caso de las dos colas vistas en a: ¿Qué algoritmo utilizaría para planificarlas?

14. Suponga que en un Kernel se utiliza un algoritmo de planificación de colas multinivel. El mismo cuenta con 3 colas de procesos listos, en las que los procesos se encolan en una u otra según su prioridad. Hay 3 prioridades (1 , 2 , 3), donde un menor número indica mayor prioridad. Se utiliza el algoritmo de prioridades para la administración entre las colas.

Se tiene el siguiente lote de procesos a ser procesados con sus respectivas operaciones de I/O:

Job	Llegada	CPU	I/O (rec,ins,dur)	Prioridad
1	0	9	(R1, 4, 2) (R2, 6, 3) (R1, 8, 3)	1
2	1	5	(R3, 3, 2) (R3, 4, 2)	2
3	2	5	(R1, 4, 1)	3
4	3	7	(R2, 1, 2) (R2, 5, 3)	2
5	5	5	(R1, 2, 3) (R3, 4, 3)	1

Suponiendo que las colas de cada recurso (dispositivo de entrada/salida) se

administran a través de FCFS y que cada cola de procesos listos se administra por medio de un algoritmo RR con un quantum de 3 unidades, realice un diagrama de Gantt:

a. Asumiendo que NO hay apropiación entre los procesos.
b. Asumiendo que hay apropiación entre los procesos.

15. En el esquema de Colas Multinivel, cuando se utiliza un algoritmo de prioridades para administrar las diferentes colas los procesos pueden sufrir starvation.

La técnica de envejecimiento se puede aplicar a este esquema, haciendo que un proceso cambie de una cola de menor prioridad a una de mayor prioridad, después de cierto periodo de tiempo que el mismo se encuentra esperando en su cola. Luego de llegar a una cola en la que el proceso llega a ser atendido, el mismo retorna a su cola original.

Por ejemplo: Un proceso con prioridad 3 está en cola su cola correspondiente. Luego de X unidades de tiempo, el proceso se mueve a la cola de prioridad 2. Si en esta cola es atendido, retorna a su cola original, en caso contrario luego de sucederse otras X unidades de tiempo el proceso se mueve a la cola de prioridad 1. Esta última acción se repite hasta que el proceso obtiene la CPU, situación que hace que el mismo vuelva a su cola original.

Para los casos a y b del ejercicio 15 realice el diagrama de Gantt considerando además que se tiene un envejecimiento de 4 unidades.

16. La situación planteada en el ejercicio 16, donde un proceso puede cambiar de una cola a otra, se la conoce como Colas Multinivel con Realimentación.

Suponga que se quiere implementar un algoritmo de planificación que tenga en cuenta el tiempo de ejecución consumido por el proceso, penalizando a los que más tiempo de ejecución tienen. (Similar a la tarea del algoritmo SJF que tiene en cuenta el tiempo de ejecución que resta).

Utilizando los conceptos vistos de Colas Multinivel con Realimentación indique qué colas implementaría, qué algoritmo usaría para cada una de ellas así como para la administración de las colas entre sí.

Tenga en cuenta que los procesos no deben sufrir inanición.

17. A cuáles de los siguientes tipos de trabajos:

- cortos acotados por CPU
- cortos acotados por E/S
- largos acotados por CPU
- largos acotados por E/S

benefician las siguientes estrategias de administración:

a. prioridad determinada estáticamente con el método del más corto primero (SJF).
b. prioridad dinámica inversamente proporcional al tiempo transcurrido desde la última operación de E/S.

18. Analizar y explicar por qué si el quantum en Round-Robin se incrementa sin límite, el algoritmo de planificación se aproxima a FIFO.

19. Dado los siguientes programas en un pseudo-código que simula la utilización de llamadas al sistema para crear procesos en Unix/Linux, indicar qué mensajes se imprimirán en pantalla (sin importar el orden en que saldrían):
    a. Caso 1

```
printf("hola")
x = fork()
if x <  1 {
      execv("ls")
      printf("mundo")
      exit(0)
}
exit(0)
```

    b. Caso 2

```
printf("hola")
x = fork()
if x <  1 {
      execv("ps")
       printf("mundo")
       exit(0)
}
execv("ls")
printf("fin")
exit(0)
```

    c. Caso 3

```
printf("Anda    a   rendir   el   Primer   Parcial   de
Promo!")
newpid = fork()
if newpid == 0 {
    printf("Estoy comenzando el Examen")
    execv("ps")
    printf("Termine el Examen")
}
printf("¿Como te fue?")
exit(0)
printf("Ahora anda a descansar")
```

20. Dado el siguiente programa en C, analice su objetivo sin necesidad de ejecutarlo. (Investigue el funcionamiento del iterador *for* en C para poder responder: https://ccia.ugr.es/~jfv/ed1/c/cdrom/cap4/cap43.htm)

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main ( void ) {
     int c ;
     pid_t pid ;
     printf( " Comienzo . : \ n " ) ;
     for ( c = 0 ; c < 3 ; c++ ) {
```

```
            pid = fork ( ) ;
        }
        printf ( " Proceso \n " ) ;
        return 0 ;
}
```

a. ¿Cuántas líneas con la palabra "Proceso" aparecen al final de la ejecución de este programa?
b. *¿El número de líneas es el número de procesos que han estado en ejecución?.* Ejecute el programa y compruebe si su respuesta es correcta. Modifique el valor del bucle for y compruebe los nuevos resultados.

21. Modifiquemos el programa anterior. Ahora, además de un mensaje, vamos a añadir una variable y antes de finalizar el programa vamos a mostrar el valor de la misma:

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main ( void ) {
      int c ;
      pid_t pid ;
      int p = 0;
      printf( " Comienzo . : \ n " ) ;
      for ( c = 0 ; c < 3 ; c++ ) {
            pid = fork ( ) ;
      }
      p++;
      printf ( " Proceso %d\n ",p ) ;
      return 0 ;
}
```

a. ¿Qué valores se imprimirán en la consola?
b. ¿Todas las líneas tendrán el mismo valor o algunas líneas tendrán valores distintos?
c. ¿Cuál es el valor (o valores) que aparece*?.* Compile y ejecute el programa y compruebe si su respuesta es correcta.
d. Realice modificaciones al programa, por ejemplo: modifique el valor del bucle for, cambie el lugar dónde se incrementa la variable p, y compruebe los nuevos resultados.

22. ¿Qué es la MMU y qué funciones cumple?

23. ¿Que representa el espacio de direcciones de un proceso?

24. Explique y relacione los siguientes conceptos:
    Dirección Lógica o Virtual
    Dirección Física

25. En la técnica de Particiones Múltiples, la memoria es dividida en varias particiones y

los espacios de direcciones de los procesos son ubicados en estas, siempre que el tamaño del mismo sea menor o igual que el tamaño de la partición.

Al trabajar con particiones se pueden considerar 2 métodos (independientes entre sí):

Particiones Fijas
Particiones Dinámicas

a. Explique cómo trabajan estos 2 métodos. Cite diferencias, ventajas y desventajas.
b. ¿Qué información debe disponer el Kernel para poder administrar la memoria principal con estos métodos?
c. Realice un gráfico indicando cómo se realiza la transformación de direcciones lógicas a direcciones físicas.

26. Al trabajar con particiones fijas, los tamaños de las mismas se pueden considerar:
    - ➢ Particiones de igual tamaño.
    - ➢ Particiones de diferente tamaño.

Cite ventajas y desventajas entre las alternativas.

27. Fragmentación. Ambos métodos de particiones presentan el problema de la fragmentación:

Fragmentación Interna (Para el caso de Particiones Fijas)
Fragmentación Externa (Para el caso de Particiones Dinámicas)

a. Explique a qué hacen referencia estos 2 problemas
b. El problema de la Fragmentación Externa es posible de subsanar. Explique una posible técnica que permita al Kernel mitigar este problema.

28. Analice y describa cómo funcionan las siguientes técnicas de asignación de particiones: Best Fit, Worst Fit, First Fit y Next Fit. Tenga en cuenta que información debe mantener el Kernel. Indique, para los métodos de particiones dinámicas y fijas, con cuál técnica se obtienen mejores resultados y justifique.

29. Segmentación
    a. Explique cómo trabaja este método de asignación de memoria.
    b. ¿Qué estructuras son necesarias en el Kernel para poder ejecutarse sobre un Hardware que utiliza segmentación?
    c. Explique, utilizando gráficos, cómo son resueltas las direcciones lógicas a físicas.
    d. En este esquema: ¿se puede producir fragmentación (interna y/o externa)?
    e. De las técnicas enumeradas en el ejercicio 29 ¿Cuál se ajusta mejor para la asignación de memoria principal?

30. Dado un esquema de segmentación donde cada dirección hace referencia a 1 byte y la siguiente tabla de segmentos de un proceso, traduzca, de corresponder, las

direcciones lógicas indicadas a direcciones físicas. La Dirección lógica está representada por: **segmento:desplazamiento**

Segmento	Dir. Base	Tamaño
0	102	12500
1	28699	24300
2	68010	15855
3	80001	400

i. 0000:9001
ii. 0001:24301
iii. 0002:5678
iv. 0001:18976
v. 0003:0

31. Paginación
    a. Explique cómo trabaja este método de asignación de memoria.
    b. ¿Qué estructuras son necesarias en el Kernel para poder ejecutarse sobre un Hardware que utiliza paginación ?
    c. Explique, utilizando gráficos, cómo son resueltas las direcciones lógicas en físicas.
    d. En este esquema: ¿se puede producir fragmentación (interna y/o externa)?

32. Suponga un sistema donde la memoria es administrada mediante la técnica de paginación, y donde:
    El tamaño de la página es de 512 bytes
    Cada dirección de memoria referencia 1 byte.
    Se tiene una memoria principal de 10240 bytes (10 KiB) y los marcos se enumeran desde 0 y comienzan en la dirección física 0.

Suponga además un proceso P1 con un tamaño lógico de 2000 bytes y con la siguiente tabla de páginas:

Página	Marco
0	3
1	5
2	2
3	6

    a. Realice los gráficos necesarios (de la memoria principal, del espacio de direcciones del proceso, de tabla de páginas y de la tabla de marcos) en el que refleje el estado descripto.
    b. Indicar si las siguientes direcciones lógicas corresponden al espacio lógico del proceso P1 y en caso afirmativo indicar la dirección física a la que corresponden:
        i. 35
        ii. 512
        iii. 2051
        iv. 0
        v. 1325
        vi. 602

c. Indicar, en caso de ser posible, las direcciones lógicas del proceso P1 que se corresponden a las siguientes direcciones físicas:
   i. 509
   ii. 1500
   iii. 0
   iv. 3215
   v. 1024
   vi. 2000

33. Dado un esquema donde cada dirección hace referencia a 1 byte, con páginas de 2 KiB (KibiBytes), donde el frame 0 se encuentra en la dirección física 0. Con las siguientes siguientes primeras entradas de la tabla de páginas de un proceso,

Página	Marco
0	16
1	13
2	9
3	2
4	0

traduzca las direcciones lógicas indicadas a direcciones físicas:
   i. 5120
   ii. 3242
   iii. 1578
   iv. 2048
   v. 8191

34. Tamaño de la Página. Compare las siguientes situaciones con respecto al tamaño de página y su impacto, indicando ventajas y desventajas:
   a. Un tamaño de página pequeño.
   b. Un tamaño de página grande.

35. Existen varios enfoques para administrar las tablas de páginas:
   - Tablas de páginas de 1 nivel.
   - Tablas de páginas de 2 niveles o más
   - Tablas de páginas invertidas.

Explique brevemente cómo trabajan estos mecanismos. Tenga en cuenta analizar cómo se resuelven las direcciones y qué ventajas/desventajas presentan una sobre otra.

36. Dado un esquema de paginación, donde cada dirección hace referencia a 1 byte, se tiene un tamaño de dirección de 32 bits y un tamaño de página de 2048 bytes. Suponga además un proceso P1 que necesita 51358 bytes para su código y 68131 bytes para sus datos.
   a. De los 32 bits de las direcciones ¿Cuántos se utilizan para identificar página? ¿Cuantos para desplazamiento?
   b. ¿Cuál sería el tamaño lógico máximo de un proceso?
   c. ¿Cuántas páginas máximo puede tener el proceso P1?
   d. Si se contaran con 4GiB (Gibibytes) de RAM, ¿cuantos marcos habría disponibles para utilizar?
   e. ¿Cuántas páginas necesita P1 para almacenar su código?
   f. ¿Cuántas páginas necesita P1 para almacenar sus datos?

g. ¿Cuántos bytes habría de fragmentación interna entre lo necesario para almacenar su código y sus datos? Ayuda: Recuerde que en una misma página no pueden convivir código y datos.
h. Asumiendo que el hardware utiliza tabla de páginas de un (1) nivel, que la dirección se interpreta como 20 bits para identificar página y 12 bits para desplazamiento, y que cada Entrada de Tabla de Páginas (PTE) requiere de 1 Kib. ¿Cuál será el tamaño mínimo que la tabla de páginas del proceso P1 ocupará en RAM?
i. Asumiendo que el hardware utiliza tabla de páginas de dos (2) niveles, que la dirección se interpreta como 10 bits para identificar primer nivel, 10 bits para identificar el segundo nivel, 12 bits para desplazamiento y que cada Entrada de Tabla de Páginas (PTE) requiere de 1 Kib. ¿Cuál será el tamaño mínimo que la tabla de páginas del proceso P1 ocupará en RAM?
j. Dado los resultados obtenidos en los dos incisos anteriores (h, i) ¿Qué conclusiones se pueden obtener del uso de los 2 modelos de tablas de página?

37. Cite similitudes y diferencias entre las 4 técnicas vistas: Particiones Fijas, Particiones Dinámicas, Segmentación y Paginación.

38. Dada la técnica de administración de memoria por medio de segmentación paginada:
    a. Explique cómo funciona.
    b. Cite ventajas y desventajas respecto a las técnicas antes vistas.
    c. Teniéndose disponibles las siguientes tablas:

Tabla de Segmentos

Núm. Seg.	Dir. base
1	500
2	1500
3	5000

Tabla de Páginas

Nro. Segmento	Nro. Página	Direc. Base
1	1	40
	2	80
	3	60
2	1	20
	2	25
	3	0
3	1	120
	2	150

Indicar las direcciones físicas correspondientes a las siguientes direcciones lógicas (segmento,página,desplazamiento):

i. (2,1,1)
ii. (1,3,15)
iii. (3,1,10)
iv. (2,3,5)