PRÁCTICA 5 - CSO

1. **Explique a que hacen referencia los siguientes términos:**
   - **Dirección Lógica o Virtual**
   - **Dirección Física**

   - Dirección Lógica o Virtual: *dirección que hace referencia dentro del proceso*
     Dirección que enmascara o abstrae una dirección física. Es una referencia a una localidad en memoria y se la debe traducir a una dirección física.
   - Dirección Física: *hace referencia directamente a la memoria RAM del sistema de computo. Es decir es HW*
     Es la dirección real. Es con la que se accede efectivamente a memoria .
     Representa la dirección absoluta en memoria principal.

2. **Al trabajar con particiones se pueden considerar 2 métodos (independientes entre sí):**
   2.1 Particiones Fijas
   2.2 Particiones Dinámicas
   a) Explique cómo trabajan estos 2 métodos. Cite diferencias, ventajas y desventajas.
   b) ¿Qué información debe disponer el SO para poder administrar la memoria con estos métodos?
   c) Realice un gráfico indicando cómo realiza el SO la transformación de direcciones lógicas a direcciones físicas.

   a) 2.1Particiones fijas: memoria se divide en particiones de tamaño fijo en el que se guarda un **único proceso** . Cada proceso se coloca en alguna partición de acuerdo a los siguientes criterios:
      i) First Fit: vamos a tener una lista de particiones y de acuerdo al requerimiento del proceso, vamos a ir recorriendo esa lista y en la primera partición cuyo requerimiento de memoria por el proceso sea menor o igual que la partición se le asigna esa partición.
      ii) Best Fit: El mejor ajuste significa buscar la partición que se ajusta mejor a ese requerimiento de memoria.
      iii) Worst Fit: asignar el proceso a esa partición que deja espacio en memoria sin utilizar.
      iv) Next Fit:

      <u>Desventajas</u>
      Fragmentación interna: queda mucho espacio libre sin utilizar dentro de la partición porque el proceso es más corto o el requerimiento de memoria es menor

2.2. Particiones dinámicas: las particiones se van a ir ajustando al requerimiento del

proceso. Ya no vamos a tener una cierta cantidad de particiones limitadas sino que se van a ir creando a partir de la demanda. Cuando un proceso llega al sistema se va a crear una nueva particion y cuando deja esa partición esa partición se libera

Desventajas

Fragmentación Externa: al ir creando y eliminando particiones puede ser que la memoria quede muy fragmentada con espacios libres que se encuentran entre las particiones (huecos).

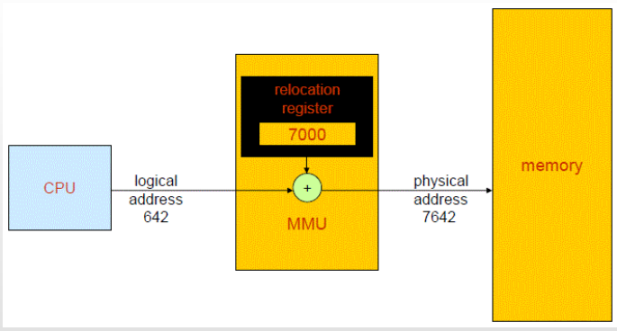
La solución es la compactación de memoria: se mueve todos los procesos hacia los extremos para que los espacios de memoria sin utilizar (huecos) puedan ser utilizados. Esto es muy costoso

Otra solución más avanzada:

Paginación se divide la memoria en porciones de igual tamaño de memoria (marcos) y a cada proceso lo divido en páginas del mismo tamaño de esos marcos

b) Se debe conocer el registro base y el registro límite, los cuales almacenan dónde comienza y termina la memoria asignada al proceso.

c) El mapeo entre direcciones virtuales y físicas se realiza mediante HW → MMU (Memory Management Unit)



El registro de relocación apunta al punto de entrada de carga de un proceso. Para calcular la dirección física suma la dirección virtual al punto de carga.

3. Al trabajar con particiones fijas los tamaños de las mismas se pueden considerar:

a. Particiones de igual tamaño

b. Particiones de diferente tamaño

Cite ventajas y desventajas de los dos métodos

a. Particiones de igual tamaño → fácil de implementar, pueden generar fragmentación interna, el uso de algoritmos implica una carga más para el procesador, tiene un límite de espacio máximo aunque haya espacio libre

b. Particiones de diferente tamaño → hacen mejor uso del espacio, genera fragmentación externa, el uso de algoritmos implica una carga más para el procesador.

4. 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 que hacen referencia estos 2 problemas

b) El problema de la Fragmentación Externa es posible de subsanar. Explique una técnica que evite este problema.

hecho en 1)

5. Paginación

a) Explique cómo trabaja este método de asignación de memoria.

b) ¿Qué estructuras adicionales debe poseer el SO para llevar a cabo su implementación?

c) Explique, utilizando gráficos, como son transformadas las direcciones lógicas en físicas.

d) En este esquema: ¿Se puede producir fragmentación (interna y/o externa)?

a) La memoria se divide de igual tamaño llamadas marco, es decir pedazos iguales de memoria. A cada proceso lo divido en páginas del mismo tamaño de esos marcos de memoria que voy a utilizar.

b) El SO debe tener una tabla de páginas por proceso, la cual contiene por cada página el marco de memoria en el cual se encuentra almacenada esa página.

6. Cite similitudes y diferencias entre la técnica de paginación y la de particiones fijas.

- Similitudes → segmentación de memoria y fragmentación interna
- Diferencias → paginación divide el proceso en varias particiones mientras que la partición fija coloca el proceso en una única partición

7. 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.**
- **Los marcos en memoria principal se encuentran desde la dirección física**

Suponga además un proceso con un tamaño 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, proceso y tabla de páginas) en el que reflejen el estado descrito.

Memoria Principal			
Marco	**Página**	**Direc Virtual**	**Direc Física**
	0		0..511
	1		512..1023
2	2	1024..1535	1024..1535
0	3	0..511	1536..2047
	4		2048..2559
1	5	512..1023	2560..3071
3	6	1536..2047	3072..3583

b) Indicar si las siguientes direcciones lógicas son correctas 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 que se corresponden si las siguientes direcciones físicas:
i) 509 ii) 1500 iii) 0 iv) 3215 v) 1024 vi) 2000
d) ¿Indique, en caso que se produzca, la fragmentación (interna y/o externa)

8.- Considere un espacio lógico de 8 páginas de 1024 bytes cada una, mapeadas en una memoria física de 32 marcos.

RECORDAR
Dir. Lógica div Tam. Página = N.º de Página
Dir. Lógica mod Tam. Página = Desplazamiento
Dir. Física = Inicio o base del frame + desplazamiento

a) ¿Cuántos bits son necesarios para representar una dirección lógica?
dirección lógica → indico el num de pagina y el desplazamiento dentro de la pagina para referenciar una posición de memoria
En este caso, el espacio lógico tiene 8 páginas, y cada página es de 1024 bytes. Para representar cada página, necesitamos

log2(8)=3
log 2(8)=3 bits
Para representar la dirección dentro de cada página, necesitamos
log2(1024)=10
log
2

(1024)=10 bits, ya que
210=1024
2
10
=1024.

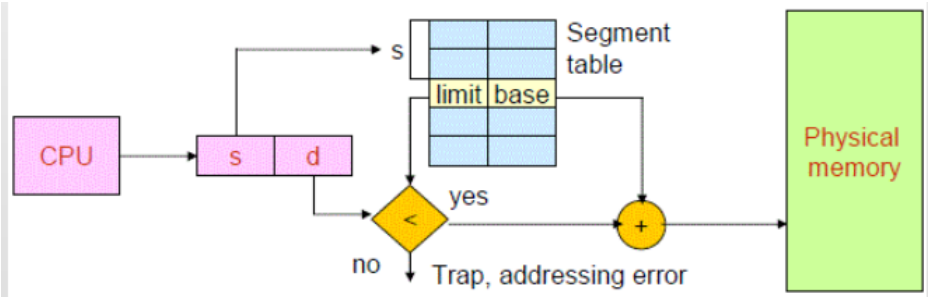
b) ¿Cuántos bits son necesarios para representar una dirección física?

9.- Segmentación
a) Explique cómo trabaja este método de asignación de memoria.
(info sacada de teoría)
Es un esquema el cual es similar a la visión de usuario. El programa se divide en partes/secciones. Este programa es una colección de segmentos, los cuales son unidades lógicas como un programa principal, variables globales o locales, stack, entre otras. La segmentación puede causar fragmentación externa.
Todos los segmentos de un programa pueden no tener el mismo tamaño
Las direcciones lógicas consisten en dos partes 1. Selector de segmento 2. Desplazamiento dentro del segmento

b) ¿Qué estructuras adicionales debe poseer el SO para llevar a cabo su implementación?

- Tabla de segmentos → permite mapear la direc lógica en física. Cada entrada contiene base (DF de comienzo del segmento) y limit (longitud del segmento)
- Segment-table base register (STBR): apunta a la ubicación de la tabla de segmentos
- Segment-table length register (STLR): cantidad de segmentos de un programa

c) Explique, utilizando gráficos, como son transformadas las direcciones lógicas en físicas.



d) En este esquema: ¿Se puede producir fragmentación (interna y/o externa)?
Fragmentación externa

10.- Cite similitudes y diferencias entre la técnica de segmentación y la de particiones dinámicas.
Similitudes: ambas pueden generar fragmentación externa y se basan en la idea de distribuir espacios de tamaño variable a los procesos en forma dinámica
Diferencias:
Partición dinámica → requiere que todo el proceso esté cargado en forma continua en memoria
Segmentación → subdivide y distribuye las divisiones de forma indiferente a su orden real, aunque sí mantiene la secuencia dentro de cada segmento

11.- Cite similitudes y diferencias entre la técnica de paginación y segmentación.

Similitudes: ambas permiten la distribución de sectores no contiguos en memoria y tampoco requieren que dichos sectores sean asignados en el mismo orden que en el programa principal.
Ambas requieren de estructuras adicionales
Diferencias:
Paginación → transparente al programador, elimina fragmentación externa
Segmentación → visible al programador, facilita modularidad, estructuras de datos grandes y da mejor soporte a la importación y protección

12.- Dado un S.O. que administra la memoria por medio de segmentación paginada, y teniéndose disponibles las siguientes tablas:

Tabla de Segmentos

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

Tabla de Paginas

Nro. Segmento	Nro. Pagina	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) = 1500 + 20 + 1 = 1521
ii) (1,3,15) = 500 + 60 + 15 = 575
iii) (3,1,10) = 5000 + 120 + 10 = 5130
iv) (2,3,5) = 1500 + 0 + 5 = 1505

13.- Memoria Virtual
a) Describa qué beneficios introduce este esquema de administración de la memoria.
La memoria virtual permite ejecutar procesos que requieren más memoria que la disponible en el sistema, manteniendo en MP solo aquella memoria que el proceso esté utilizando, el resto se almacenará en el disco.

b) ¿En qué se debe apoyar el SO para su implementación?
En el HW, este debe ser capaz de detectar cuando una instrucción está tratando de acceder a una dirección que no está en el momento cargada en memoria, y a partir de eso recién el SO puede generar las instrucciones necesarias para atender el fallo.
Si esta tarea dependiera únicamente del SO, se debería emplear mucho tiempo para detectar si una dirección lógica hace referencia a una porción de programa ya cargada.
Aún así, el SO debe dar al HW la información requerida para llevar a cabo la tarea

c) Al implementar está técnica utilizando paginación por demanda, las tablas de páginas de un proceso deben contar con información adicional además del marco donde se encuentra la página. ¿Cuál es está información? ¿ Por qué es necesaria?
Debe contar con ***un bit*** que indique la presencia de la página en memoria, esto es debido a que a partir de este dato es que el HW puede generar la interrupción necesaria para resolver el fallo de página.
También es necesaria contar con ***información sobre la presencia de modificaciones*** sobre las páginas cargadas, puesto que cualquier descarga de una página modificada

implica la necesidad de actualizar los datos cargados en disco, para mantener la consistencia
Por último es necesario saber los ***bits de control*** para saber con myor precisión cuál porción puede ser la mejor para realizar un reemplazo en caso de presentarse un fallo de página

14.- Fallos de Página (Page Faults):
a) ¿Cuándo se producen?
Se producen cuando el proceso intenta usar una dirección que está en una página que no se encuentra en la memoria principal.

b) ¿Quién es responsable de detectar un fallo de página?
El HW detecta la situación y genera un trap al SO

c) Describa las acciones que emprende el SO cuando se produce un fallo de página
El SO:

- Coloca al proceso en estado *blocked* (espera) mientras gestiona que la página que se necesite se cargue
- Busca un *marco libre* o *frame* en la memoria y genera una operación de E/S al disco para copiar en dicho frame la página del proceso que se necesita utilizar
- Puede asignarle la CPU a otro proceso mientras se completa la E/S, la cual al terminar notifica mediante interrupción su finalización
- Luego, el SO actualiza la tabla de páginas del proceso (coloca el Bit V en 1 en la página en cuestión, coloca la dirección base del Frame donde se colocó la página)
- El proceso que generó el fallo de página vuelve a estado de listo
- Cuando el proceso se ejecute, se volverá a ejecutar la instrucción que antes generó el fallo de página

1. Se genera el trap.
2. El SO bloquea al proceso (la CPU toma otro proceso).
3. El SO busca un marco libre en la memoria y genera una operación de E/S que le pide al disco, para copiar en dicho marco la página deseada.
4. La E/S avisa por interrupción cuando finaliza.
5. El SO actualiza la tabla de páginas del proceso.
    - Colocando el bit V en 1, en la página correspondiente.
    - Coloca la dirección base del marco donde se colocó la página.
6. El proceso pasa del estado bloqueado a listo para ejecutar
7. Cuando vuelva a ser asignado al CPU, el proceso comenzará desde la instrucción que generó el fallo en primera instancia

15.- Direcciones:
a) Si se dispone de una espacio de direcciones virtuales de 32 bits, donde cada dirección referencia 1 byte:
i) ¿Cuál es el tamaño máximo de un proceso (recordar "espacio virtual")?
(cant de direcciones) * (tamaño de referencia)
2^32 * 1 byte =2^32 = = 4.294.967.296 Bytes = 4GiB

ii) Si el tamaño de página es de 512Kb. ¿Cuál es el número máximo de páginas que puede tener un proceso?
Tamaño máx del proceso / tamaño de página
Tamaño máx del proceso → calculado anteriormente: cambio unidades
4.294.967.296 bytes / 1024 = 4194304 Kib

4194304 Kib / 512 Kib = 8192 páginas

iii) Si el tamaño de página es de 512Kb. y se disponen de 256 Mb. de memoria real ¿Cuál es el número de marcos que puede haber?
Página = 512 Kb
Memoria = 256 Mb → 256 * 1024 = 262144 Kb
262144 Kb / 512 Kb = 512 frames

iv) Si se utilizaran 2 Kb. para cada entrada en la tabla de páginas de un proceso: ¿Cuál sería el tamaño máximo de la tabla de páginas de cada proceso?

8192 * 2Kb = 16384 Kb = 16 MB

16.- Como se vio en el ejercicio anterior, la tabla de páginas de un proceso puede alcanzar un tamaño considerablemente grande, que incluso, no podría almacenarse de manera completa en la memoria real. Es por esto que el SO también realiza paginación sobre las tablas de páginas. Existen varios enfoques para administrar las tablas de páginas:

- **Tablas de páginas de 1 nivel.**
- **Tablas de páginas de 2 niveles.**
- **Tablas de páginas invertidas.**

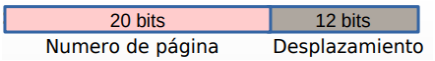
Explique brevemente cómo trabajan estos enfoques e indique cómo se realiza la transformación de la dirección virtual en dirección física.

Cada proceso tiene su tabla de páginas, el tamaño de las páginas depende del espacio de direcciones del proceso.
Organización:

- Tabla de 1 nivel: tabla única lineal
- Tabla de 2 niveles o más (multinivel)
- Tabla invertida: Hashing

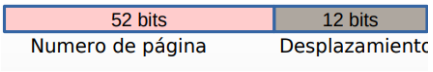
TABLA DE 1 NIVEL

- Direcciones de 32 bits



- ☑Ejemplo
  - ✓Cantidad de Page Table Entries (PTEs) máximas que puede tener un proceso = 2^{20} (1.048.576)
  - ✓El tamaño de cada página es de 4KB (2^{12})
  - ✓El tamaño de cada PTE es de 4 bytes
    - ✓Cantidad de PTEs que entran en un marco: 4KB/4B = 2^{10}
  - ✓Tamaño de tabla de páginas
    - Cantidad de marcos necesarios para todas las PTEs de la tabla de páginas de un proceso = $2^{20}/2^{10} = 2^{10}$
    - Tamaño tabla de páginas del proceso: 2^{10} * 4bytes = **4MB por proceso**

- Direcciones de 64 bits



- ☑Ejemplo
  - ✓Cantidad de Page Table Entries (PTEs) máximas que puede tener un proceso = 2^{52}
  - ✓El tamaño de cada página es de 4KB
  - ✓El tamaño de cada PTE es de 4 bytes
    - Cantidad de PTEs que entran en un marco: 4KB/4B = 2^{10}
  - ✓Tamaño de tabla de páginas
    - Cantidad de marcos necesarios para todas las PTEs de la tabla de páginas de un proceso = $2^{52}/2^{10} = 2^{42}$
    - Tamaño tabla de páginas del proceso = 2^{42} * 4bytes = 2^{54}
    - **Más de 16,000GB por proceso!!!**

TABLA DE 2 niveles

El propósito de la tabla de páginas multinivel *es dividir la tabla de páginas lineal en múltiples tablas de páginas*. Cada tabla de páginas suele tener el mismo tamaño pero se busca que tengan un menor número de páginas por tabla
La idea es que cada tabla sea más pequeña.
Se busca que la tabla de páginas no ocupe demasiada memoria RAM
Solo se carga una parcialidad de la tabla de páginas (solo lo que se necesite resolver)
Existe un esquema de direccionamientos indirectos

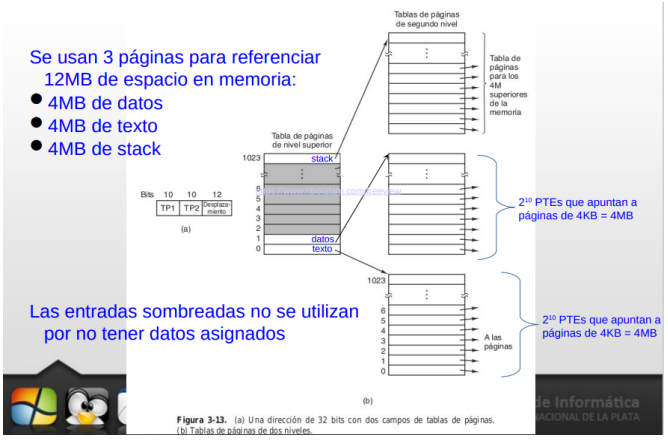


TABLA DE PÁGINAS INVERTIDA

Es utilizada en arquitecturas donde el espacio de direcciones es muy grande
Se adopta esta técnica debido a que las tablas de páginas ocuparían muchos niveles y la traducción sería costosa.
Hay una entrada por cada marco de página en la memoria real, es la visión inversa a la que veníamos viendo
Hay una sola tabla para todo el sistema
El espacio de direcciones de la tabla se refiere al espacio físico de la RAM, en vez del espacio de direcciones virtuales de un proceso
EL número de página es transformado en un valor de HASH
El HASH se usa como índice de la tabla invertida para encontrar el marco asociado
Se define un mecanismo de encadenamiento para solucionar colisiones→ cuando el hash da igual para 2 direcciones virtuales
Sólo se mantienen los PTEs de páginas presentes en memoria física
La tabla invertida es organizada como tabla hash en memoria principal Se busca indexadamente por número de página virtual Si está presente en tabla, se extrae el marco de página y sus protecciones Si no está presente en tabla, corresponde a un fallo de págin

Las tablas de páginas invertidas buscan solucionar el problema invirtiendo el proceso de conversión de direcciones (de ahí su nombre). La tabla de páginas invertida se encuentra siempre cargada en memoria, y contiene una entrada por cada marco de la memoria (en lugar de por cada página de cada proceso). El número de página de cada proceso se vincula con un marco a través de una función de hash relativamente sencilla (de otra forma los costos de procesamiento serían demasiado altos). Como varias páginas pueden dar como resultado de la función de hash un mismo valor, si un marco está asociado con más de una página, las entradas se encadenan, de modo que el punto de entrada siempre es la primera entrada en la tabla de páginas. La entrada de página, a su vez, contiene un identificador del proceso al que corresponde, a fin de poder ser reconocida de manera unívoca. Una vez resuelta la asociación entre página y tabla, se puede resolver la dirección física agregando el desplazamiento como en los demás métodos.

17.- Suponga que la tabla de páginas para un proceso que se está ejecutando es la que se muestra a continuación:

Página	Bit V	Bit R	Bit M	Marco
0	1	1	0	4
1	1	1	1	7
2	0	0	0	-
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

Asumiendo que:

- **El tamaño de la página es de 512 bytes**
- **Cada dirección de memoria referencia 1 byte**
- **Los marcos se encuentran contiguos y en orden en memoria (0, 1, 2..) a partir de la dirección real 0.**

¿Qué dirección física, si existe, correspondería a cada una de las siguientes direcciones virtuales? (No gestione ningún fallo de página, si se produce)
a) 1052 b) 2221 c) 5499 d) 3101

RECORDAR
Dir. Lógica div Tam. Página = N.º de Página
Dir. Lógica mod Tam. Página = Desplazamiento
Dir. Física = Inicio o base del frame + desplazamiento

a) 1052
número de página = 1052 / 512 = 2
desplazamiento = 1052 % 512 = 28 → fallo de página

b) 2221
número de página = 2221/ 512 = 4
desplazamiento = 2221 % 512 = 173 → fallo de página

c) 5499
número de página = 15499 / 512 = 10 → inválido?

d) 3101
número de página = 3101/ 512 = 6 → inválido?

18.- Tamaño de la Página:
La selección del tamaño de la página influye de manera directa sobre el funcionamiento de la memoria virtual. Compare las siguientes situaciones con respecto al tamaño de página, indicando ventajas y desventajas:

- **Un tamaño de página pequeño.** → menor fragmentación interna, más páginas requeridas por procesos (tablas de páginas más grandes), más páginas pueden residir en memoria
- **Un tamaño de página grande.** → mayor fragmentación interna, la memoria secundaria está diseñada para transferir grandes bloques de datos más eficientemente (más rápido mover páginas hacia la MP)

A mayor tamaño de página se transfiere mejor

- Relación con la E/S
    - Vel. De transferencia: 2 Mb/s
    - Latencia: 8 ms
    - Búsqueda: 20 ms

- Página de 512 bytes
    - 1 página → total: 28,2 ms
    - Sólo 0,2 ms de transferencia (1%)
    - 2 páginas → 56,4 ms

- Página de 1024 bytes
    - total: 28,4 ms
    - Sólo 0,4 ms de transferencia

19.- Asignación de marcos a un proceso (Conjunto de trabajo o Working Set):
Con la memoria virtual paginada, no se requiere que todas las páginas de un proceso se encuentren en memoria. El SO debe controlar cuantas páginas de un proceso puede tener en la memoria principal. Existen 2 políticas que se pueden utilizar: Asignación Fija y Asignación Dinámica.
a) Describa cómo trabajan estas 2 políticas.
Asignación fija → a cada proceso se le asigna una cantidad arbitraria de marco. A su vez para el reparto se puede usar:
reparto equitativo: se asigna la misma cantidad de marcos a cada proceso → m div p
reparto proporcional: se asigna marco en base a la necesidad que tiene cada proceso → Vp.m / Vt

Asignación dinámica → los procesos se van cargando en forma dinámica de acuerdo a la cantidad de marcos que necesiten

b) Dada la siguiente tabla de procesos y las páginas que ellos ocupan, y teniéndose 40 marcos en la memoria principal, cuántos marcos le corresponden a cada proceso si se usa la técnica de Asignación Fija:
m = 40
p = 4
Vp → páginas utilizadas por proceso
Vt = 15 + 20 + 20 + 8 = 63 → páginas utilizadas en total
i) Reparto Equitativo

m / p = 10 → marcos a cada proceso

ii) Reparto Proporcional

*Vp * m / Vt*

Proceso	Total de Paginas Usadas
1	15
2	20
3	20
4	8

P1 → 15 * 40 / 63 ≈ 10
P2 → 20 * 40 / 63 ≈ 13
P3→ 20 * 40 / 63 ≈ 13
P4 → 8 * 40 / 63 ≈ 5

c) ¿Cuál de los 2 repartos usados en b) resultó más eficiente? ¿Por qué?
El más eficiente es el *proporcional*, debido a que el equitativo deja para el proceso 4 dos marcos sin utilizar y hay otros procesos que requieren más marcos.
En el proporcional, se dividen los marcos de manera que a ningún proceso le sobre ni le falten marcos

20. Reemplazo de páginas (selección de una víctima):
¿Qué sucede cuando todos los marcos en la memoria principal están usados por las páginas de los procesos y se produce en fallo de página? El SO debe seleccionar una de las páginas que se encuentra en memoria como víctima, y ser reemplazada por la nueva página que produjo el fallo.
Considere los siguientes algoritmos de selección de víctimas básicos: LRU, FIFO, OPT (Óptimo), Segunda Chance
a) Clasifique estos algoritmos de malo a bueno de acuerdo a la tasa de fallos de página que se obtienen al utilizarlos.
FIFO - LRU - FIFO segunda chance - OPT

b) Analice su funcionamiento. ¿Cómo los implementaría?
Algoritmo FIFO: trata a los frames en uso como una cola circular, es simple de implementar. La página más vieja en la memoria es reemplazada.

Algoritmo LRU: reemplaza la página que no fue referenciada por más tiempo. Cada página debe tener información del instante de su última referencia (el overhead es mayor)

Algoritmo FIFO segunda chance: Se utiliza un bit adicional (bit de referencia). Cuando la página se carga en memoria, el bit R se pone en 0. Cuando la página es referenciada el bit R se pone en 1. La víctima se busca en orden FIFO. Se selecciona la primer página cuyo bit R está en 0
Mientras se busca la víctima cada bit R que tiene el valor 1, se cambia a 0

Algoritmo ÓPTIMO: selecciona la página cuya referencia se encuentra más lejana de la actual. Es imposible de implementar debido a que no se conoce los futuros eventos

c) Sabemos que la página a ser reemplaza puede estar modificada. ¿Qué acciones debe llevar el SO cuando se encuentra ante esta situación?
El SO reserva uno o varios marcos para la descarga asincrónica de páginas. Cuando es necesario descargar una página modificada

- La página que provocó el fallo se coloca en un frame designado a la descarga asincrónica.
- El SO envía la orden de descargar asincrónicamente la página modificada mientras continúa la ejecución de otro proceso
- El frame de descarga asincrónica pasa a ser el que contenía a la pa´gina víctima que ya se descargó correctamente

21.- Alcance del reemplazo
Al momento de tener que seleccionar una página víctima, el SO puede optar por 2 políticas a utilizar:

- **Reemplazo local**
- **Reemplazo global**

a) Describa cómo trabajan estas 2 políticas.

- Reemplazo local: el fallo de página de un proceso solo puede reemplazar sus propias páginas
- Reemplazo global: el fallo de página de un proceso puede reemplazar la página de cualquier proceso

b) ¿Es posible utilizar la política de "Asignación Fija" de marcos junto con la política de "Reemplazo Global? Justifique.
No es posible porque la asignación fija determina *un conjunto específico de marcos que son exclusivos de cada proceso*.
Si se implementara un reemplazo global, un fallo de página en el proceso A, podría generar la selección de una página víctima del proceso B, con lo que se estaría quitando un marco a B y asignándoselo a A, rompiendo por completo la idea de asignación fija de marcos

25.- Hiperpaginación (Thrashing)
a) ¿Qué es?
Un sistema estopa en thrashing cuando pasa más tiempo paginando que ejecutando procesos

b) ¿Cuáles pueden ser los motivos que la causan?
Motivos:
La hiperpaginación sucede cuando los procesos en ejecución no tienen suficientes marcos asignados como para mantener su conjunto de trabajo.
Ante esta situación, cuando una referencia a memoria genera un fallo de página, se seleccionaría una víctima que forma parte del conjunto de trabajo (propio o de otro proceso, según la política de asignación y reemplazo), la ausencia de esta víctima genera otro fallo de página, que al ser atendido selecciona a otra página del conjunto de trabajo, y así sucesivamente.

c) ¿Cómo la detecta el SO?
La detecta a través de la técnica del control de frecuencia de fallos de página.
Se establece a priori para el sistema una cota máxima y una mínima, a partir de las cuales se define el estado de un proceso, y ante la evaluación de la situación general de todos los procesos, el estado del sistema.
La tasa de fallos de página (p) es, de la cantidad de accesos a memoria de un determinado proceso, la proporción que representan los fallos de página. Es decir que $0 < p <= 1$
Si **p** → **1**, el proceso no tiene suficientes marcos asignados como para mantener su localidad, y se está en riesgo de hiperpaginación
Si **p** → **0**, el proceso está muy próximo a mantener su localidad (nunca puede suceder

que p=0, durante toda la vida de un proceso ya que se perdería por completo el propósito de la memoria virtual con paginación).
Las cotas máxima y mínima, entonces, son precisamente los indicadores de si p → 1 o p → 0 para un determinado proceso, o la totalidad de ellos.

d) Una vez que lo detecta, ¿qué acciones puede tomar el SO para eliminar este problema?

26.- Considere un sistema cuya memoria principal se administra mediante la técnica de paginación por demanda que utiliza un dispositivo de paginación, algoritmo de reemplazo global LRU y una política de asignación que reparte marcos equitativamente entre los procesos. El nivel de multiprogramación es actualmente, de 4.
Ante las siguientes mediciones:
a) Uso de CPU del 13%, uso del dispositivo de paginación del 97%.
b) Uso de CPU del 87%, uso del dispositivo de paginación del 3%.
c) Uso de CPU del 13%, uso del dispositivo de paginación del 3%.

Analizar: ¿Qué sucede en cada caso? ¿Puede incrementarse el nivel de multiprogramación para aumentar el uso de la CPU? ¿La paginación está siendo útil para mejorar el rendimiento del sistema?

a) *Hiperpaginación*, los procesos pasan más tiempo esperando la resolución de fallos de página de lo que pueden ocupar el CPU para continuar su ejecución. No es recomendable incrementar el nivel de multiprogramación, puesto que empeoraría la situación.
b) Los procesos están en este momento próximos a mantener su localidad, pudiendo hacer un buen aprovechamiento de su tiempo de procesador para avanzar con sus tareas.
   Si la situación persiste podría incrementarse el grado de multiprogramación, aunque dado el uso de CPU, es posible que aumentar el grado de multiprogramación incremente en mayor medida la tasa de fallos de página sin afectar en gran medida al uso del CPU. *La paginación en este caso está siendo útil para la mejoría de rendimiento del sistema.*
c) La CPU se encuentra mayormente ociosa y los procesos no están generando fallos de páginas. Lo recomendable sería incrementar el nivel de multiprogramación ya que bajo este esquema no parece que se esté haciendo un aprovechamiento de los recursos. Hay un amplio margen todavía para que aumente el uso del dispositivo de paginación sin entrar en hiperpaginación, pero sí obteniendo un mayor uso de la CPU.
   La paginación no está siendo útil para la mejoría de rendimiento del sistema

27.- Considere un sistema cuya memoria principal se administra mediante la técnica de paginación por demanda.
Considere las siguientes medidas de utilización:

- **Utilización del procesador: 20%**
- **Utilización del dispositivo de paginación: 97,7%**
- **Utilización de otros dispositivos de E/S: 5%**

Cuales de las siguientes acciones pueden mejorar la utilización del procesador:
a) Instalar un procesador más rápido
b) Instalar un dispositivo de paginación mayor
c) Incrementar el grado de multiprogramación
d) Instalar más memoria principal
e) Decrementar el quantum para cada proceso

28.- La siguiente fórmula describe el tiempo de acceso efectivo a la memoria al utilizar paginación para la implementación de la memoria virtual:

TAE = At + (1– p)* Am + p * (Tf + Am)

Donde:
TAE = tiempo de acceso efectivo
p = tasa de fallo de página (0 <= p <=1)
Am = tiempo de acceso a la memoria real
Tf = tiempo se atención de una fallo de pagina
At = tiempo de acceso a la tabla de páginas. Es igual al tiempo de acceso a la memoria (Am) si la entrada de la tabla de páginas no se encuentra en la TLB

Suponga que tenemos una memoria virtual paginada, con tabla de páginas de 1 nivel, y donde la tabla de páginas se encuentra completamente en la memoria. Servir una falla de página tarda 300 nanosegundos si hay disponible un marco vacío o si la página reemplazada no se ha modificado, y 500 nanosegundos si se ha modificado. El tiempo de acceso a memoria es de 20 nanosegundos y el de acceso a la TLB es de 1 nanosegundo

a) Si suponemos una tasa de fallos de página de 0,3 y que siempre contamos con un marco libre para atender el fallo
¿Cuál será el TAE si el 50% de las veces la entrada de la tabla de páginas se encuentra en la TLB (hit)?
TAE = At + (1– p)* Am + p * (Tf + Am)
TAE = 20 ns + (1 - 0,3) * 20 ns + 0.3 * (300 + 20)

b) Si suponemos una tasa de fallos de página de 0,3; que el 70% de las ocasiones la página a reemplazar se encuentra modificada. ¿Cuál será el TAE si el 60% de las veces la entrada de la tabla de páginas se encuentra en la TLB (hit)?
20 ns * 0.4 + 1 * 0.1 + (1 – 0,3 ns)* 20 ns + 0,3ns * ((300ns * 0.3 + 500 * 0.7) + 20ns) = 160.1

c) Si suponemos que el 60% de las veces la página a reemplazar está modificada, el 100% de las veces la entrada de la tabla de páginas requerida se encuentra en la TLB (hit) y se espera un TAE menor a 200 nanosegundos. ¿Cuál es la máxima tasa aceptable de fallas de página?
200ns = 1ns + (1 – p)* 20 ns + p * ((300ns * 0.4 + 500 * 0.6) + 20ns) = p ≈ 0.42

29.- Anomalía de Belady
a) ¿Qué es?
Fenómeno que ocurre en algunos algoritmos de reemplazo de páginas de memoria, por lo que la frecuencia de fallas de página puede aumentar con el número de fotogramas asignados a los procesos. Los algoritmos FIFO con algunas combinaciones de solicitudes de página sufren de esto.
Es posible tener más fallos de página al aumentar el número de marcos en la memoria física utilizando el método FIFO como algoritmo de reemplazo de páginas en sistemas de gestión de memoria virtual con paginación.

b) Dada la siguiente secuencia de referencias a páginas: 3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4
I. Calcule la cantidad de fallos de páginas si se cuentan con 3 marcos y se utiliza el algoritmo de reemplazo FIFO
II. Calcule la cantidad de fallos de páginas si se cuentan con 4 marcos y se utiliza el algoritmo de reemplazo FIFO Analice la situación