

Sistemas Operativos

Ayudantía 6: Memoria

Profesores: Martín Gutiérrez, Víctor Reyes

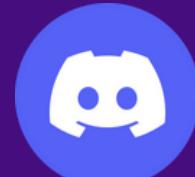
Ayudantes: Diego Banda, Dante Hortuvia



Contacto



dante.hortuvia@mail_udp.cl



doshuertos



github.com/Doshuertos/Ayudantias_SO_CIT2010

Sección 1



diego.banda@mail_udp.cl



darklouds



github.com/DiegoBan/SO2025-I

Sección 2

Respecto a la Tarea 2...

Algunas reglas generales

- Entrega individual o en parejas!!!!.
- Cualquier tipo de copia a compañeros/internet/LLMs será penalizado con nota I.
- Se debe realizar un informe explicando sus códigos realizados, decisiones y resultados.
- Debe incluir un archivo README que explique como ejecutar su código.
- Entrega: 1 de Junio a las 23:59.



Tarea 2

Enunciado

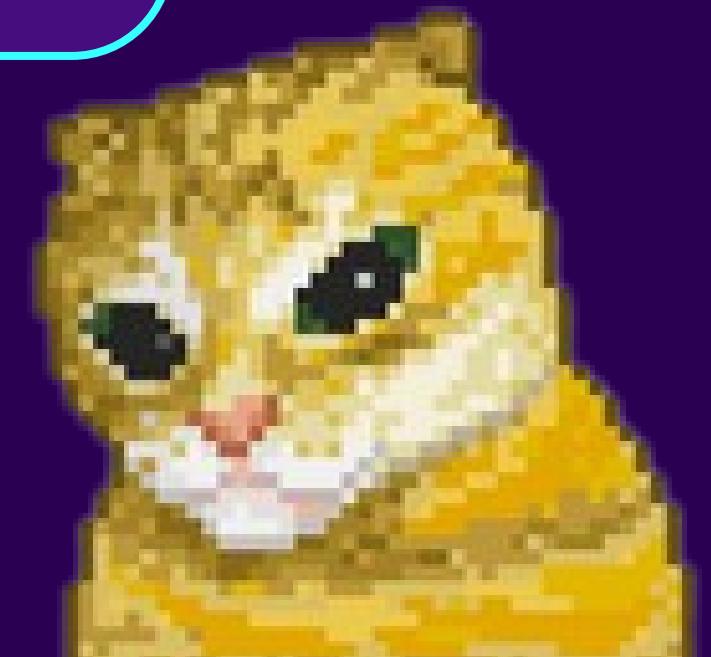
Rúbrica

Ejemplo informe

Recomendaciones

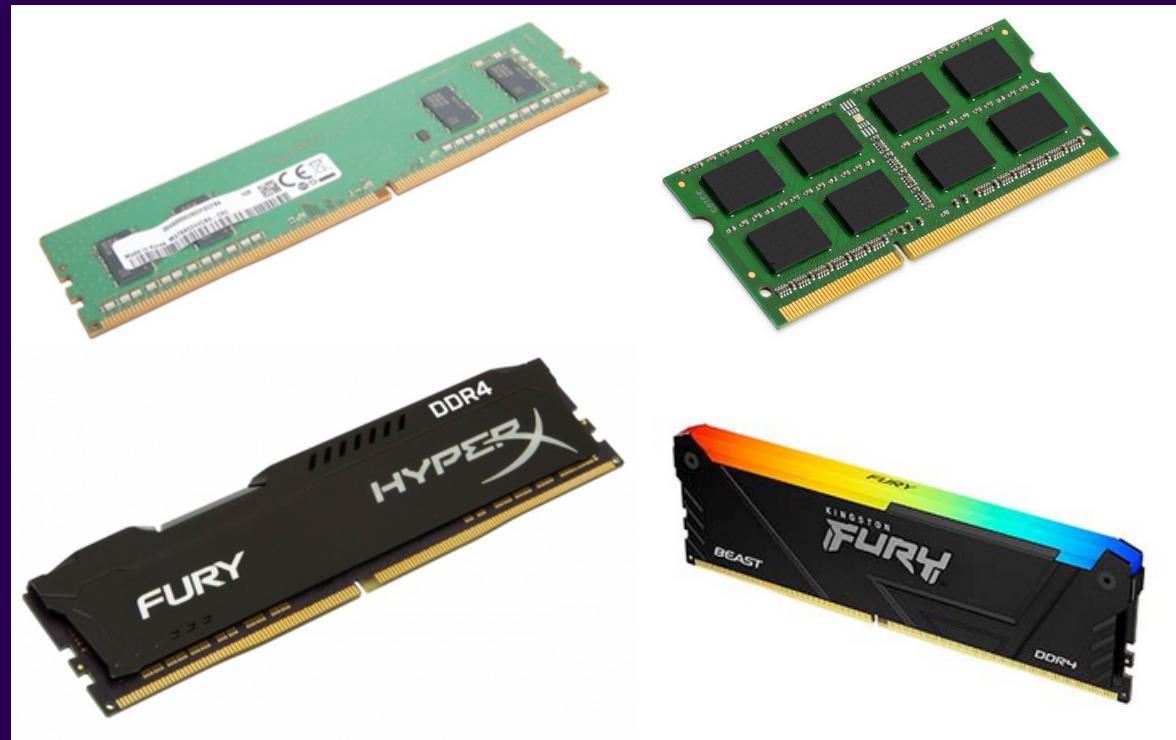
Recordar:

- Grafos no cílicos (DAG)
- Threads
- Herramientas de sincronización



Memoria Principal

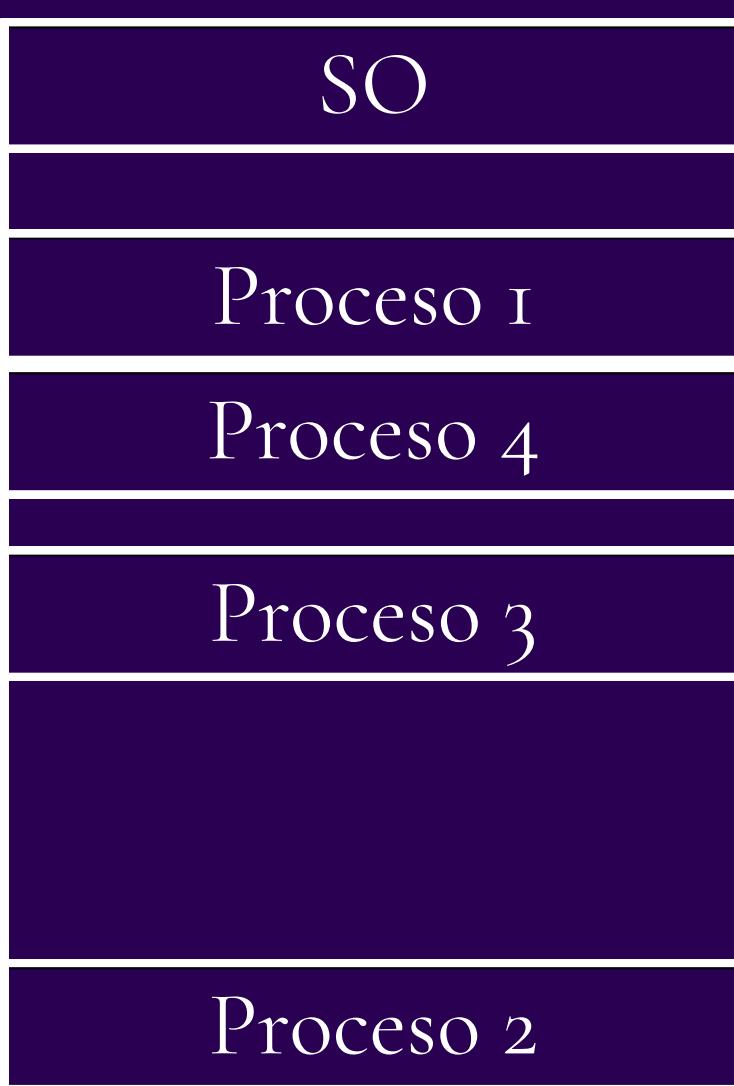
La memoria principal o también llamada RAM (Random Access Memory) es donde se aloja todo proceso en ejecución del computador.



Por fuera es
algo como
esto...

Y por dentro
algo como
esto

Lo que importa es lo de adentro...



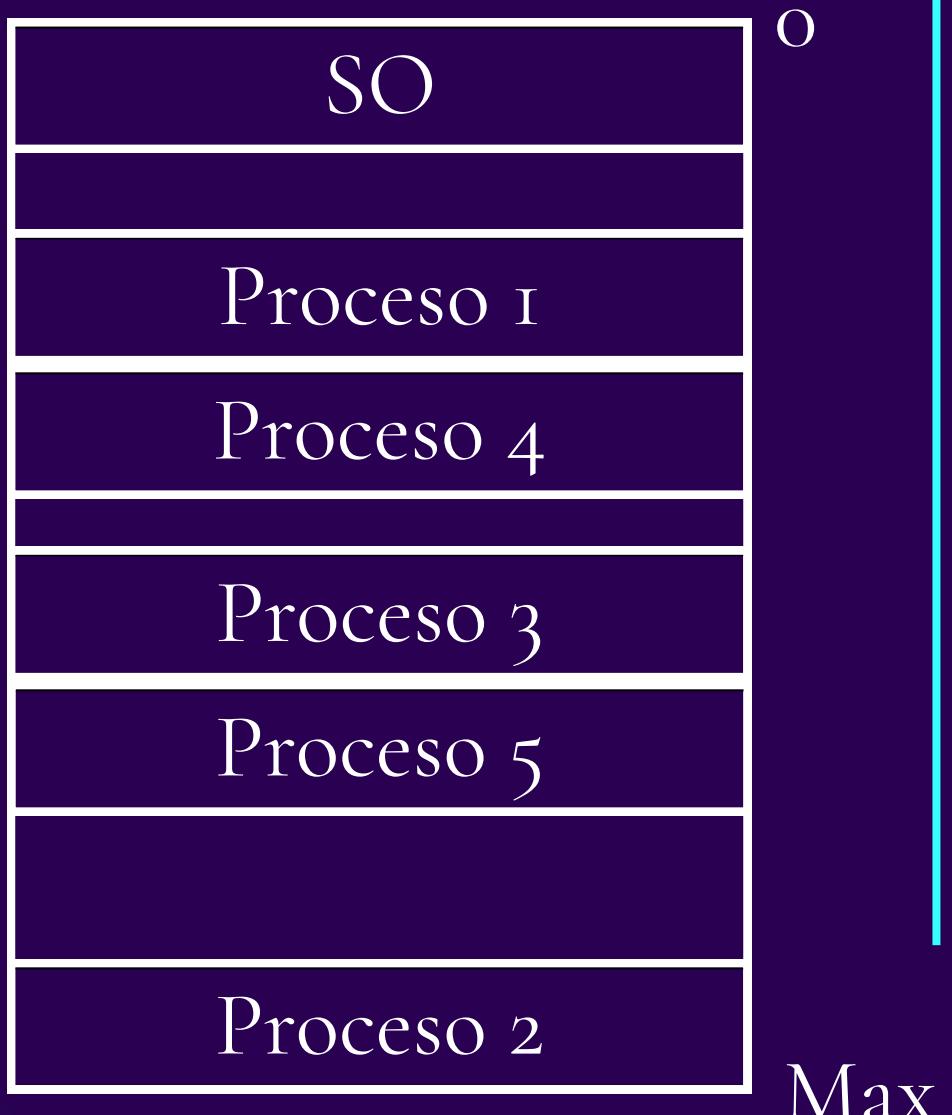
O

Max

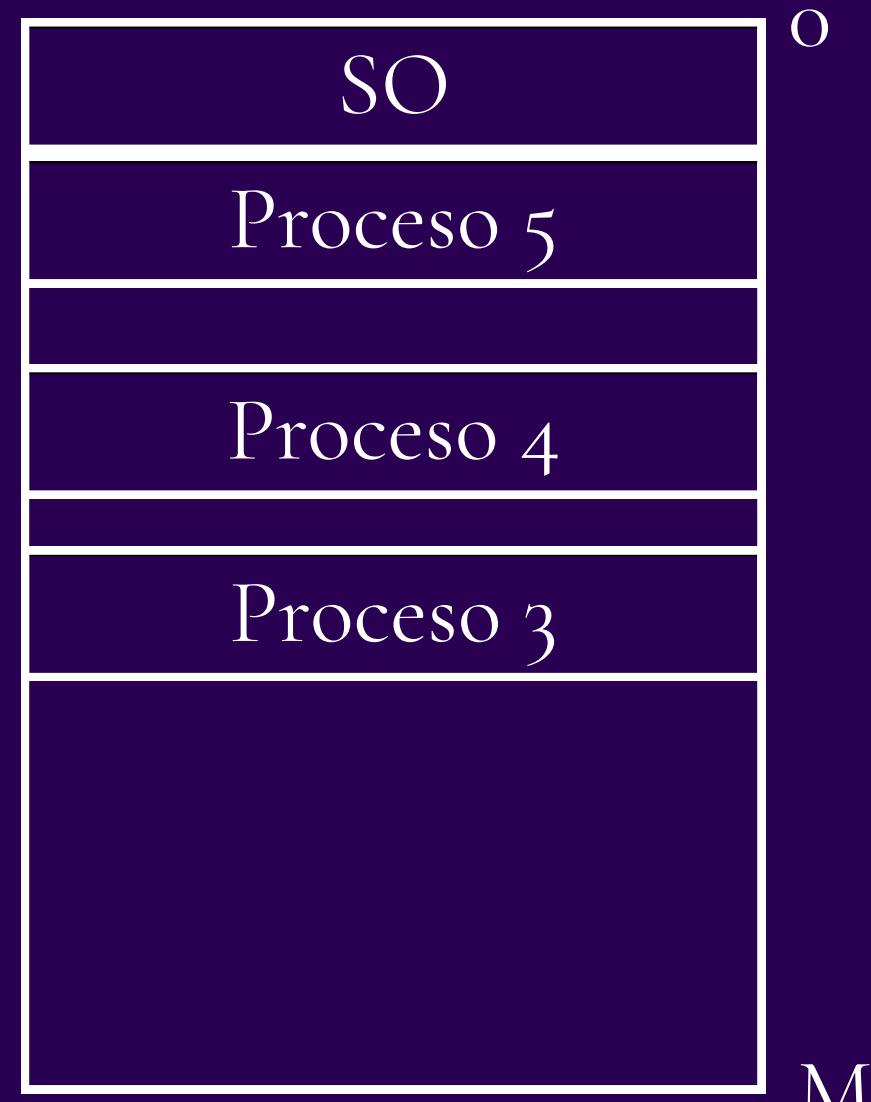
Memoria Principal: Dinamismo

Esta memoria principal es altamente rápida, y muy cambiante en el tiempo, un proceso casi nunca vuelve a usar los mismos espacios de memoria. Ej:

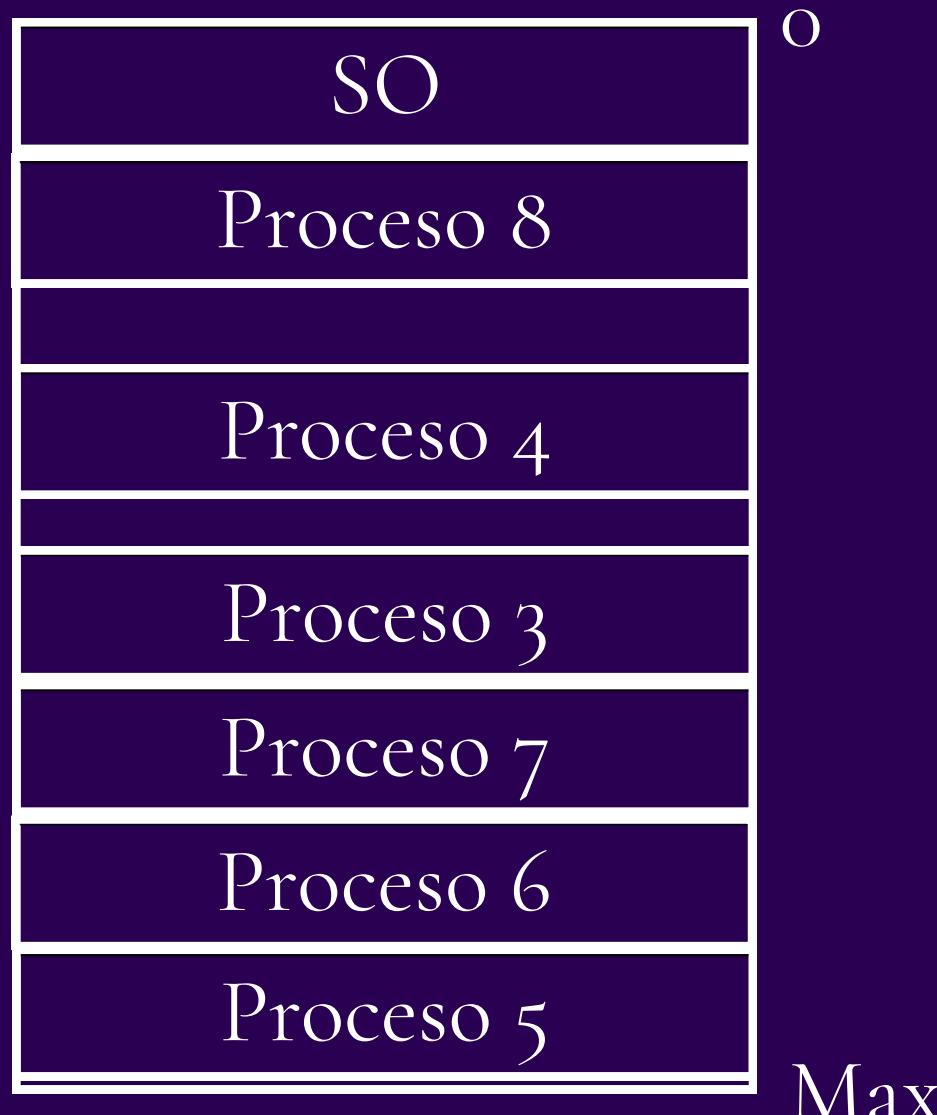
Primera vez que se ejecuta proceso 5:



Segunda vez que se ejecuta proceso 5:



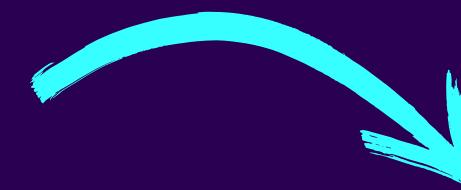
Tercera vez que se ejecuta proceso 5:



Memoria Principal: Dinamismo

Si siempre es dinámico...

- ¿Cómo guardo y obtengo datos guardados en esas mismas direcciones de memoria?
- ¿Cómo evito que un proceso acceda a los datos de otro proceso?



Utilizando:

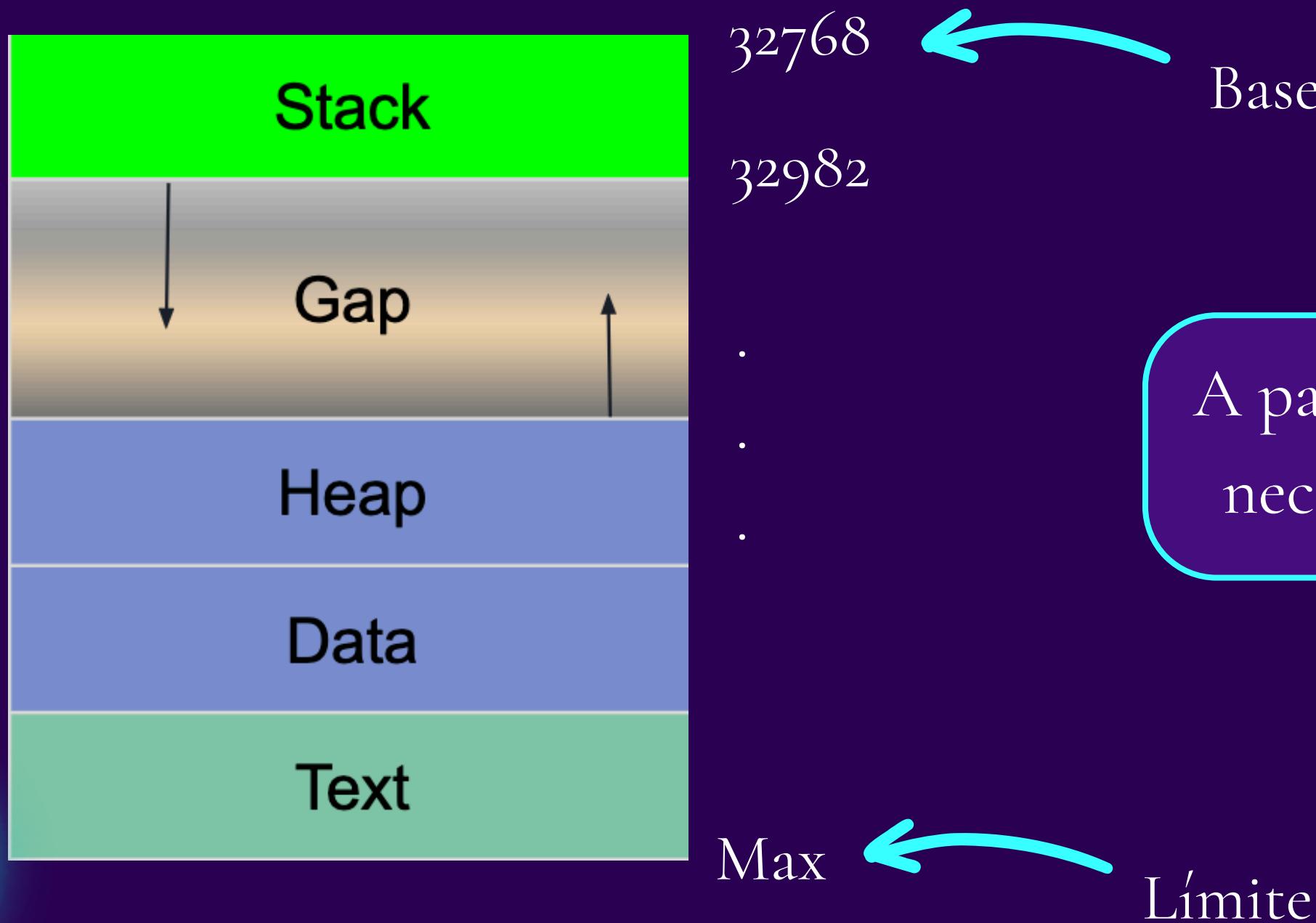
- Localización dinámica de variables.
- Protección de memoria.

Respectivamente... jeje



Localización Dinámica de Variables

La ubicación de cada dato necesario para un proceso se parametriza...

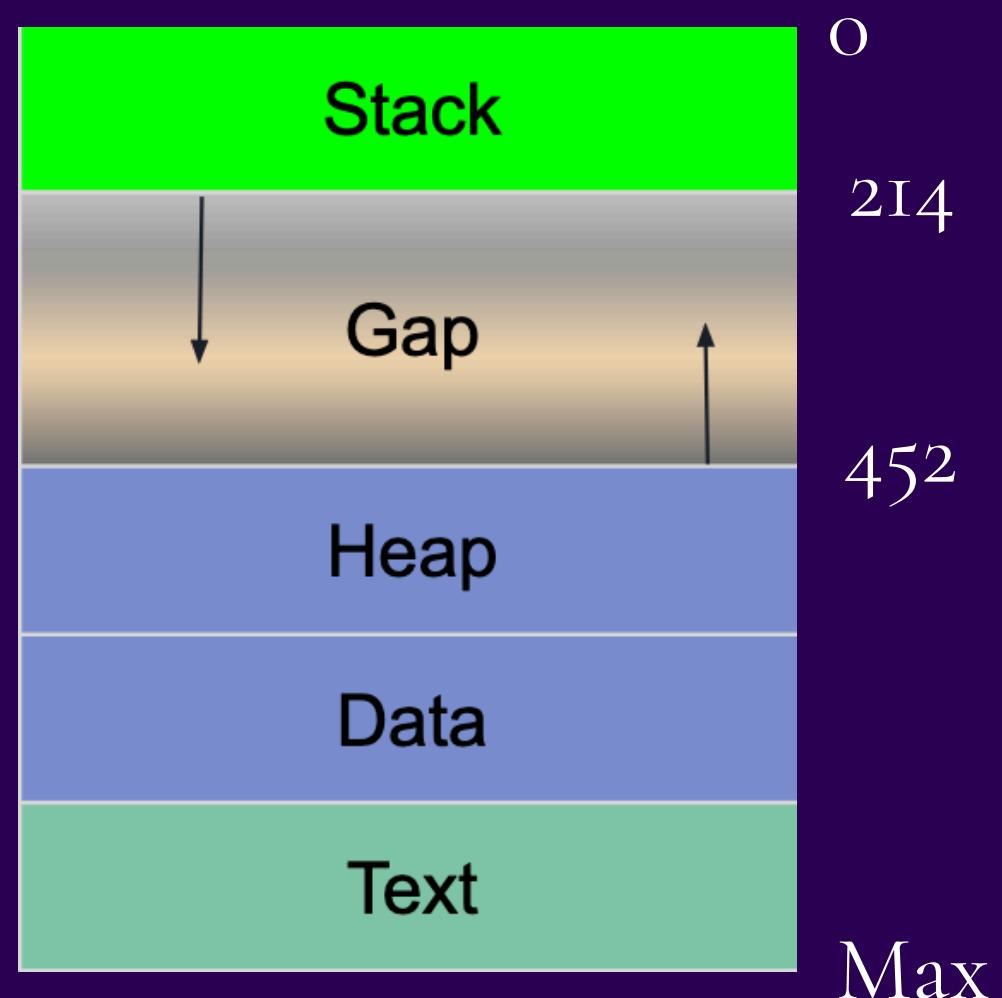


Localización Dinámica de Variables

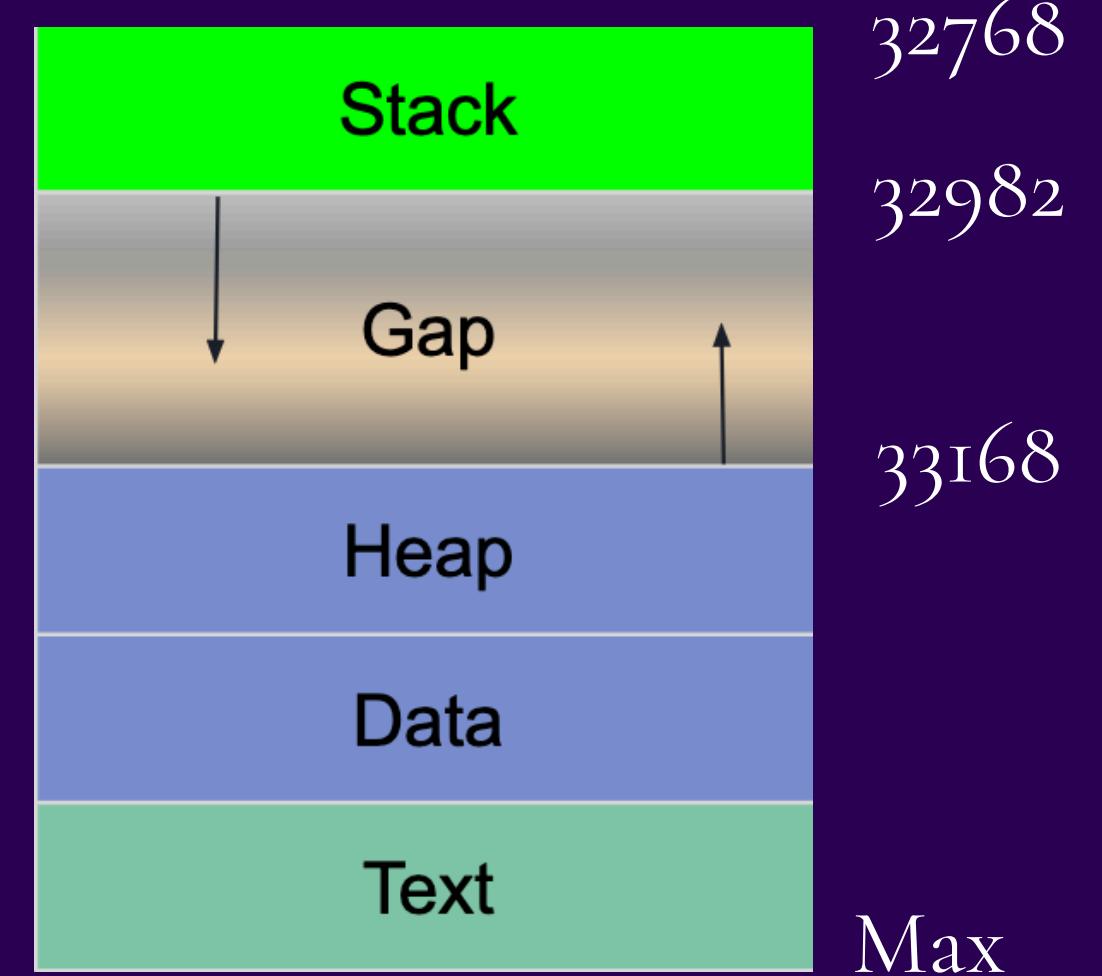
Entonces se crean estas direcciones de memoria variables y se denominan:

Direcciones Virtuales

*Para el proceso, esto se ve
más o menos así:*



*Para el SO, esto se
ve así:*



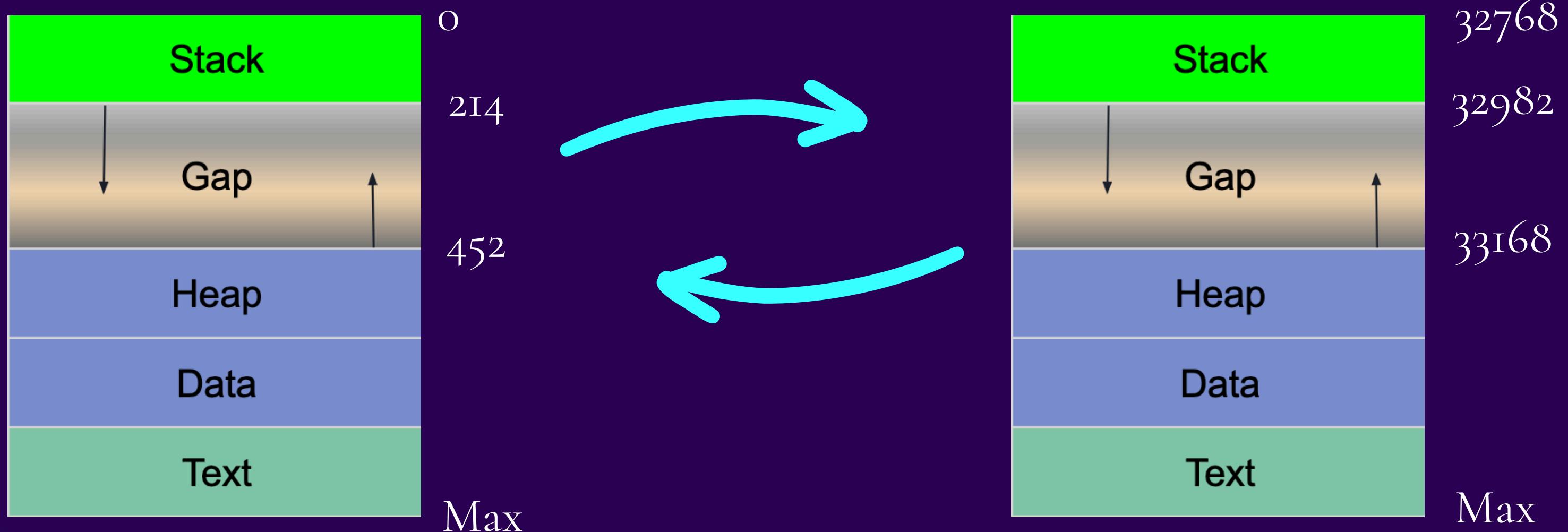
Localización Dinámica de Variables

Para calcular la dirección física sería:

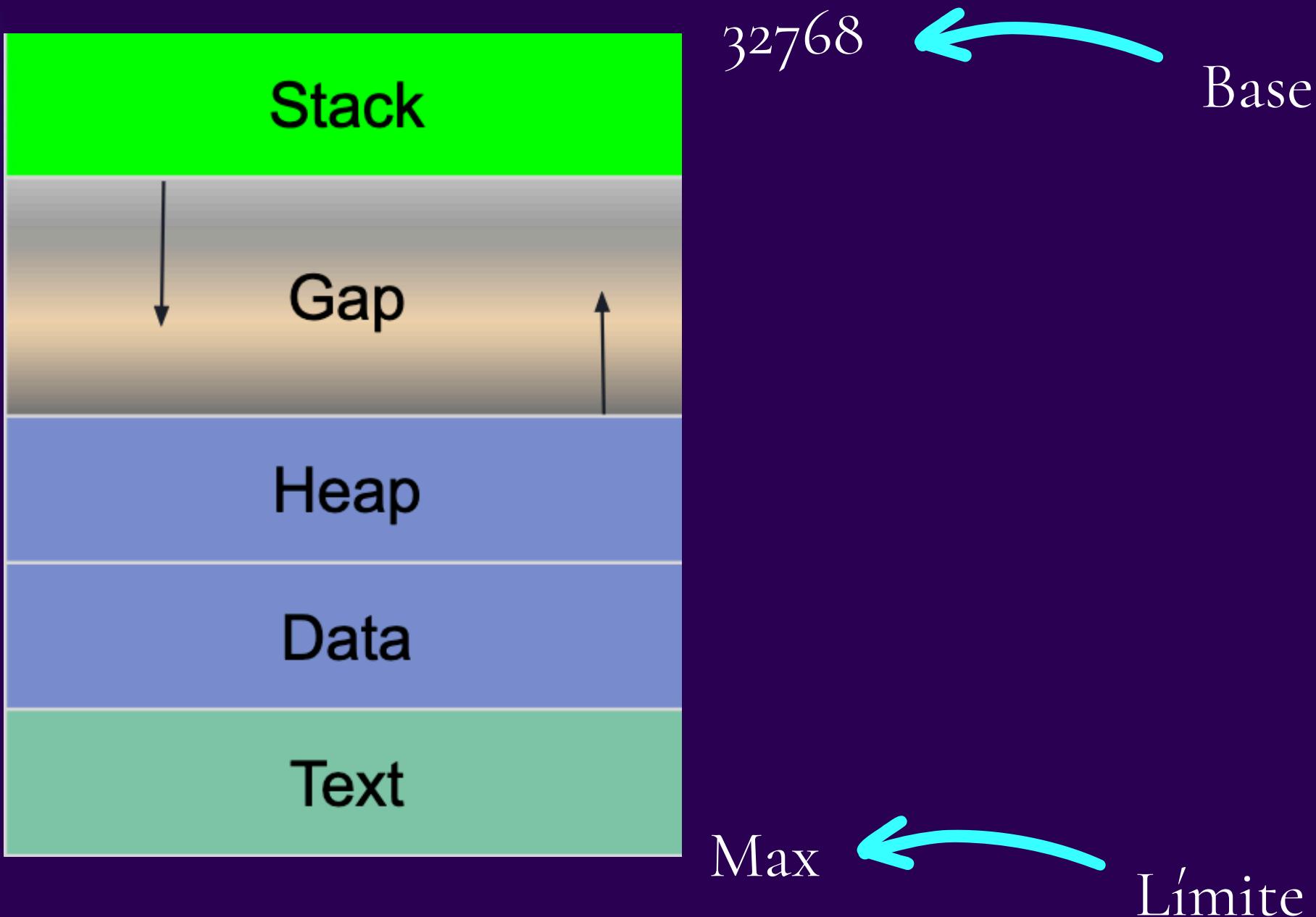
Base + dir. Virtual

Para calcular la dirección virtual sería:

dir. Física - Base



Protección de Memoria



Para asegurarse que un proceso no sobrepase sus límites y acceda a otros procesos (jugada totalmente ilegal) se tiene a una Base y Límite. Cualquier acción de lectura/escritura debe ser realizada entre los propios límites de cada proceso.



Memoria Principal: Administración

¿Por qué utilizar memoria principal?



Más rápida, pero... Sin tanto tamaño y más costosa.

Al tener poco espacio solo queda una cosa por hacer... Una buena administración: Módulo de SO llamado **MMU** (Memory Management Unit)

Se encarga de cosas como:

- Cálculo de direcciones virtuales a físicas.
- Ubicación de procesos en memoria.
- Algunos algoritmos...



Swap

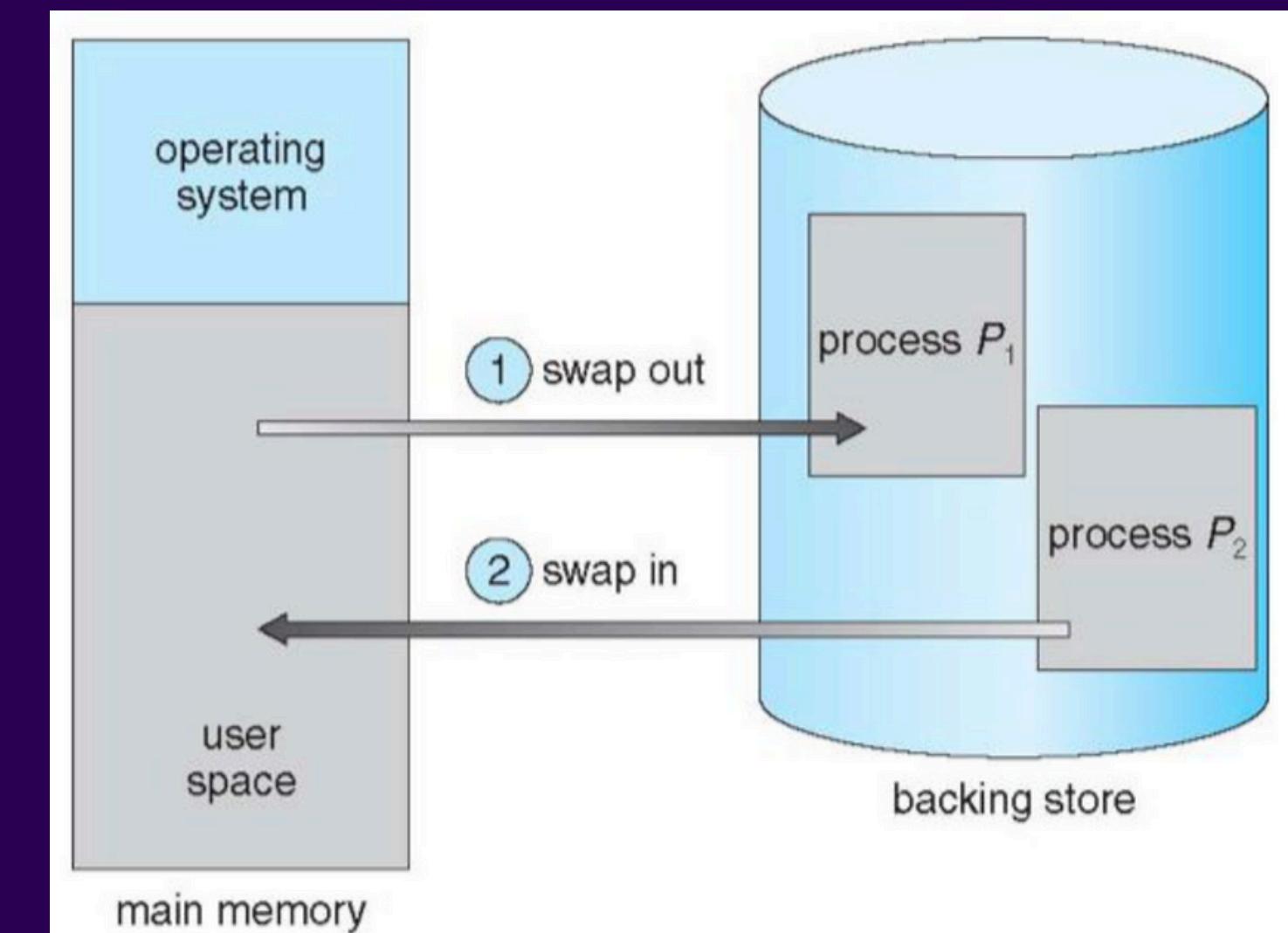
¿Qué pasa si tenemos muchos procesos y en nuestra memoria RAM no caben?

Swap: Guarda temporalmente datos de memoria principal en la secundaria.

- Swap-out: RAM ---> mem. secundaria.
- Swap-in: mem. secundaria ---> RAM

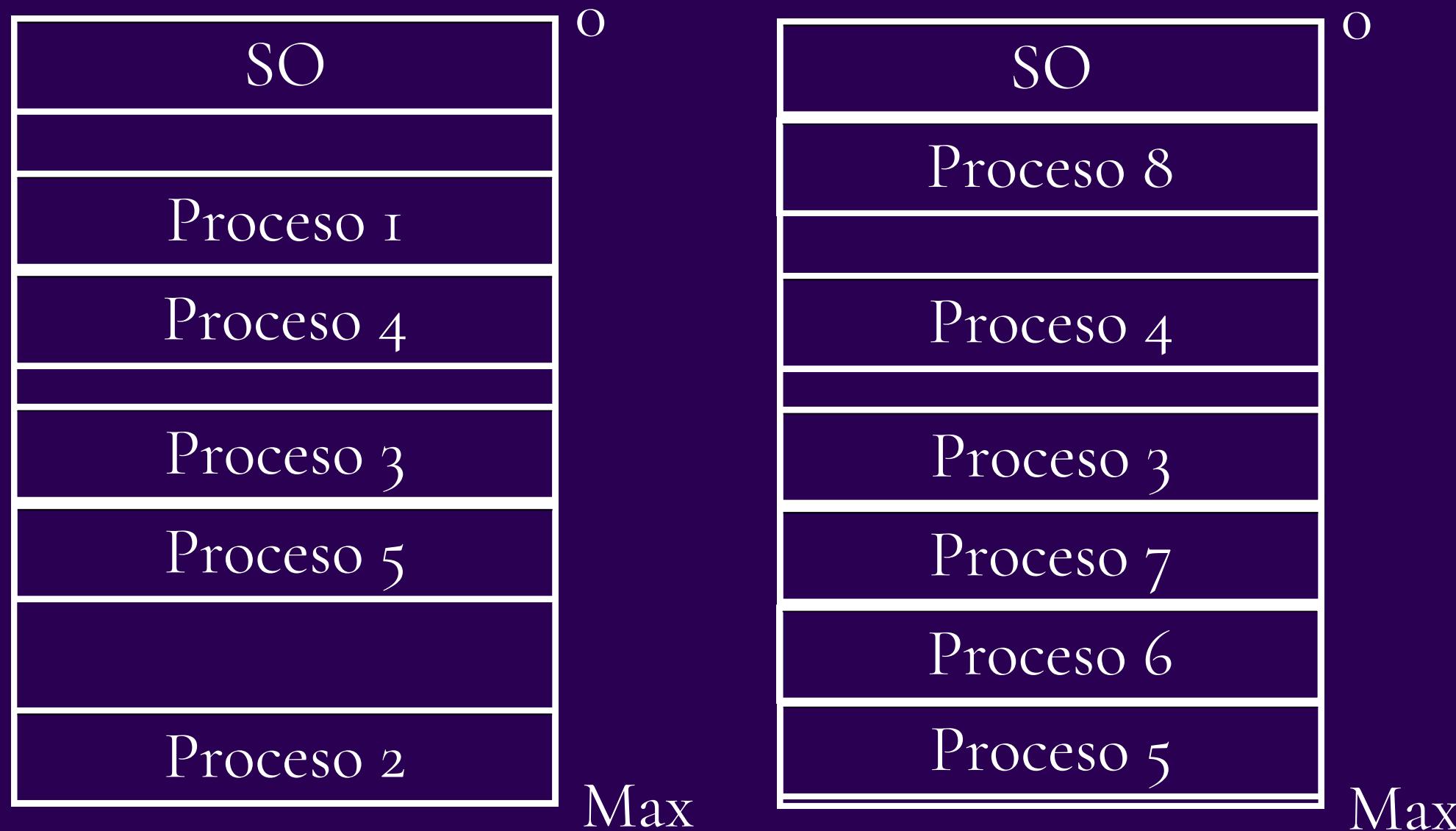
¿Problema?: \$\$\$

zzz

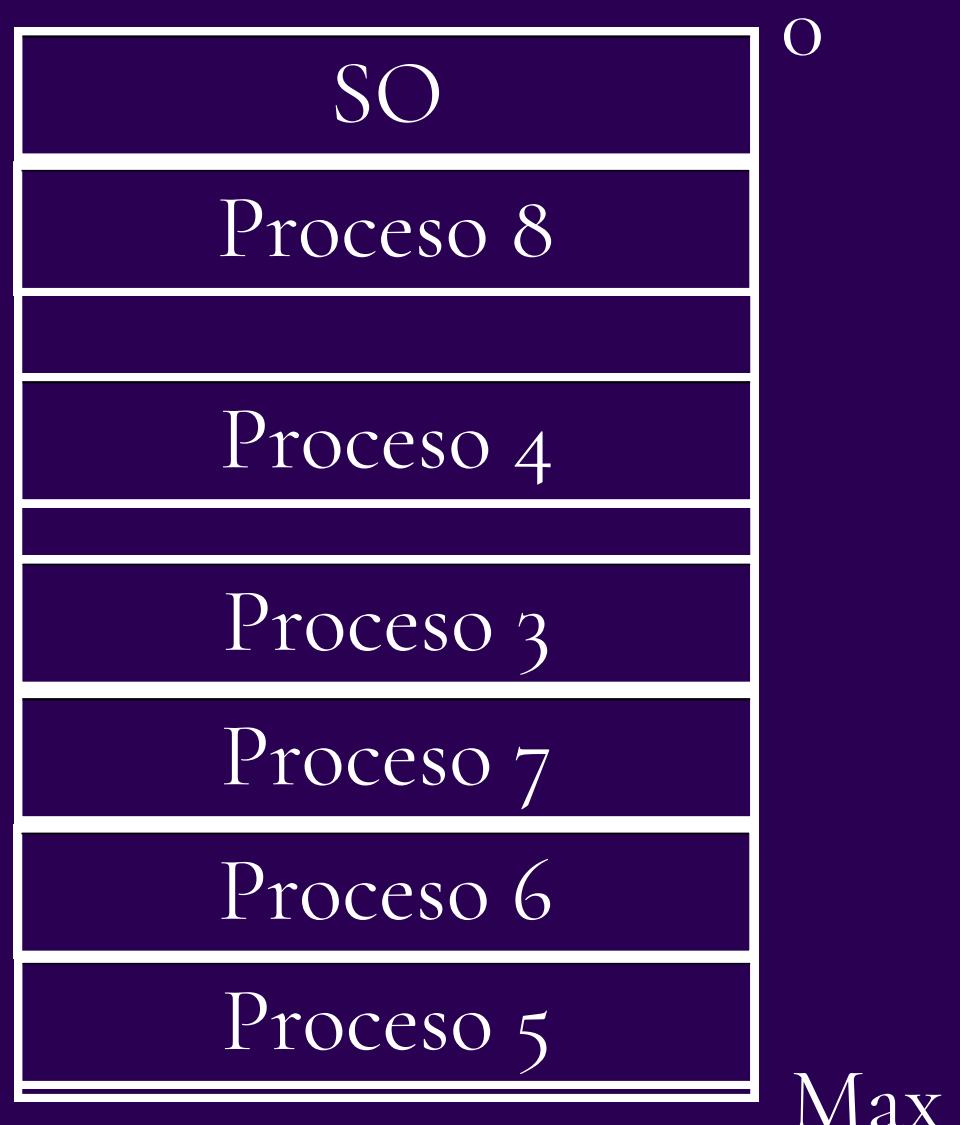
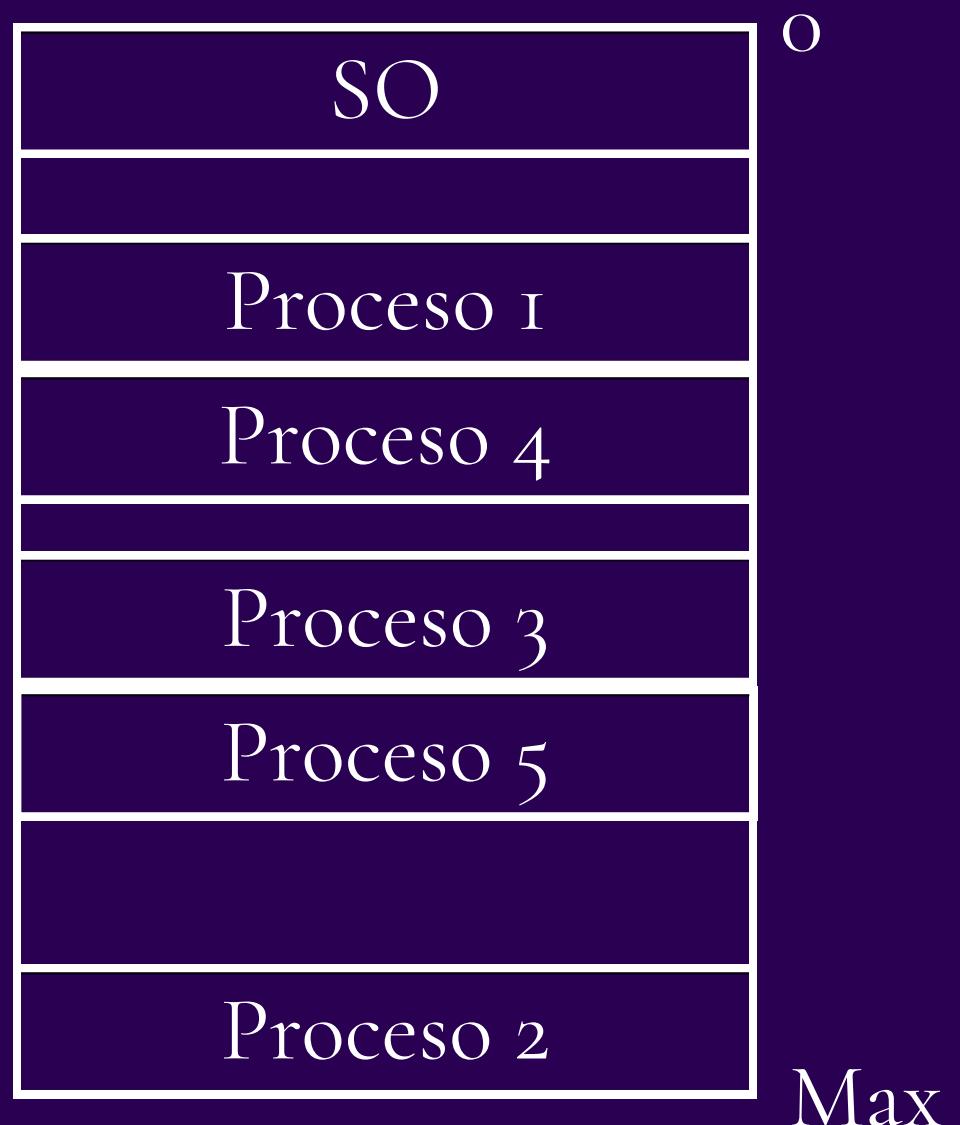


Fragmentación Externa

¿Problemas?



Fragmentación Externa



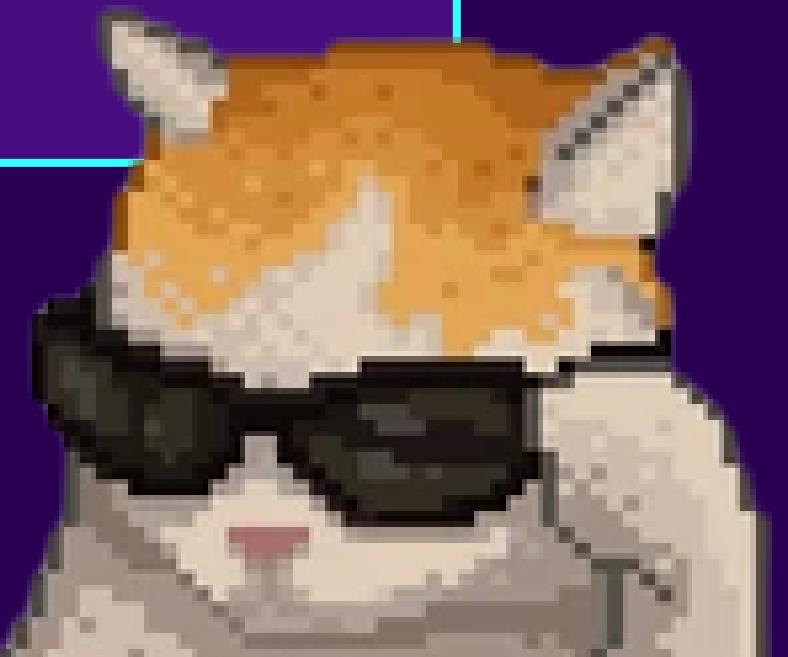
Pequeños espacios entre procesos
no pueden ser aprovechados por un
nuevo proceso.

**Se desaprovechan
espacios**

¿Qué hacer?

Existen estrategias para intentar evitar a toda costa este terrible sucesooo!!:

- **Compactación (Desfragmentación):** Ajustar todos los procesos actuales juntitos jeje.
- **First-fit:** Se asigna el proceso en el primer lugar encontrado que quepa.
- **Best-fit:** Se asigna el proceso en el lugar que mejor se ajuste al tamaño del proceso (que sobre menos espacio).
- **Worst-fit:** Se asigna el proceso en el lugar que peor se ajuste al tamaño del proceso (que sobre más espacio).

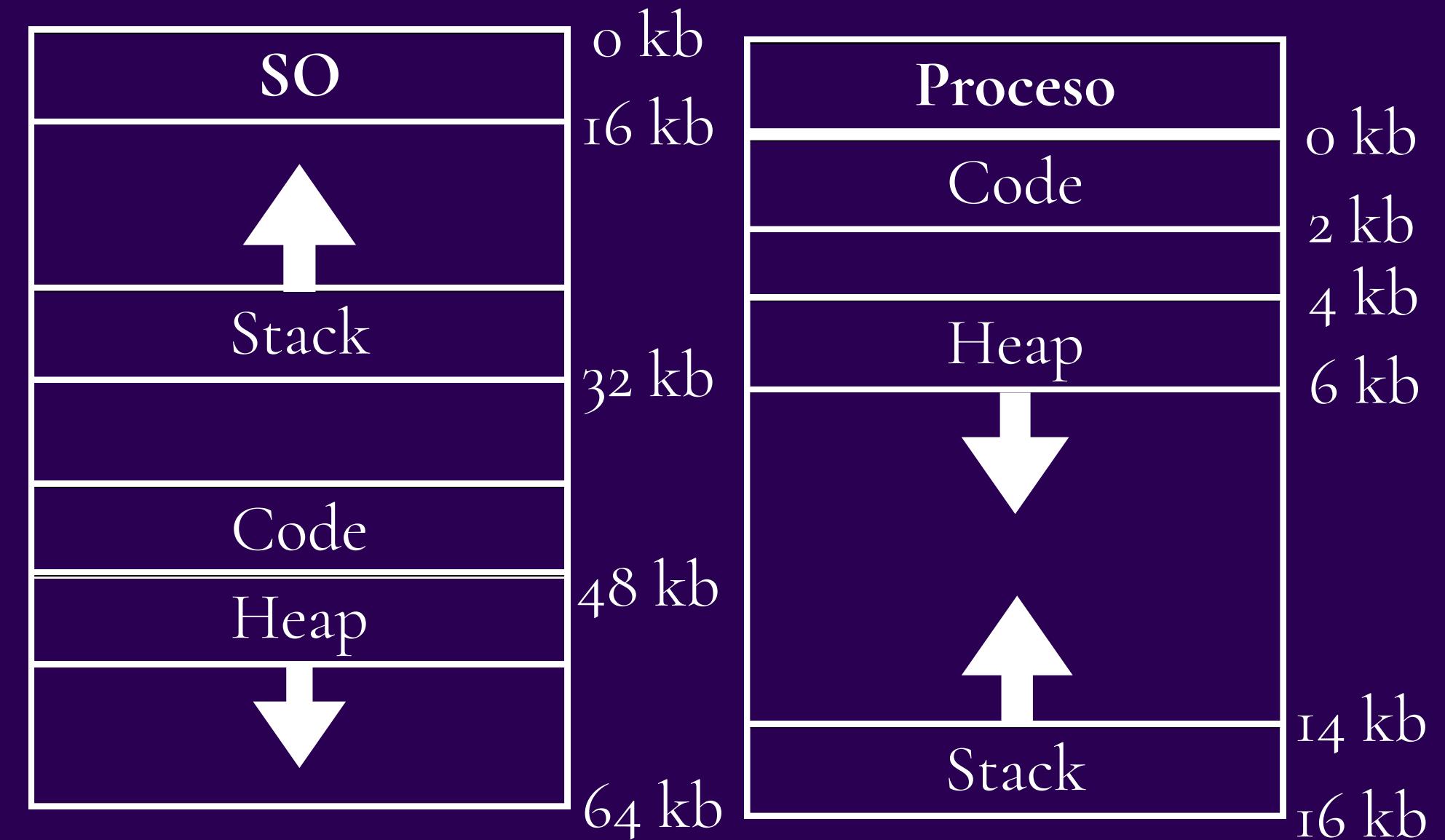


¿Qué hacer?

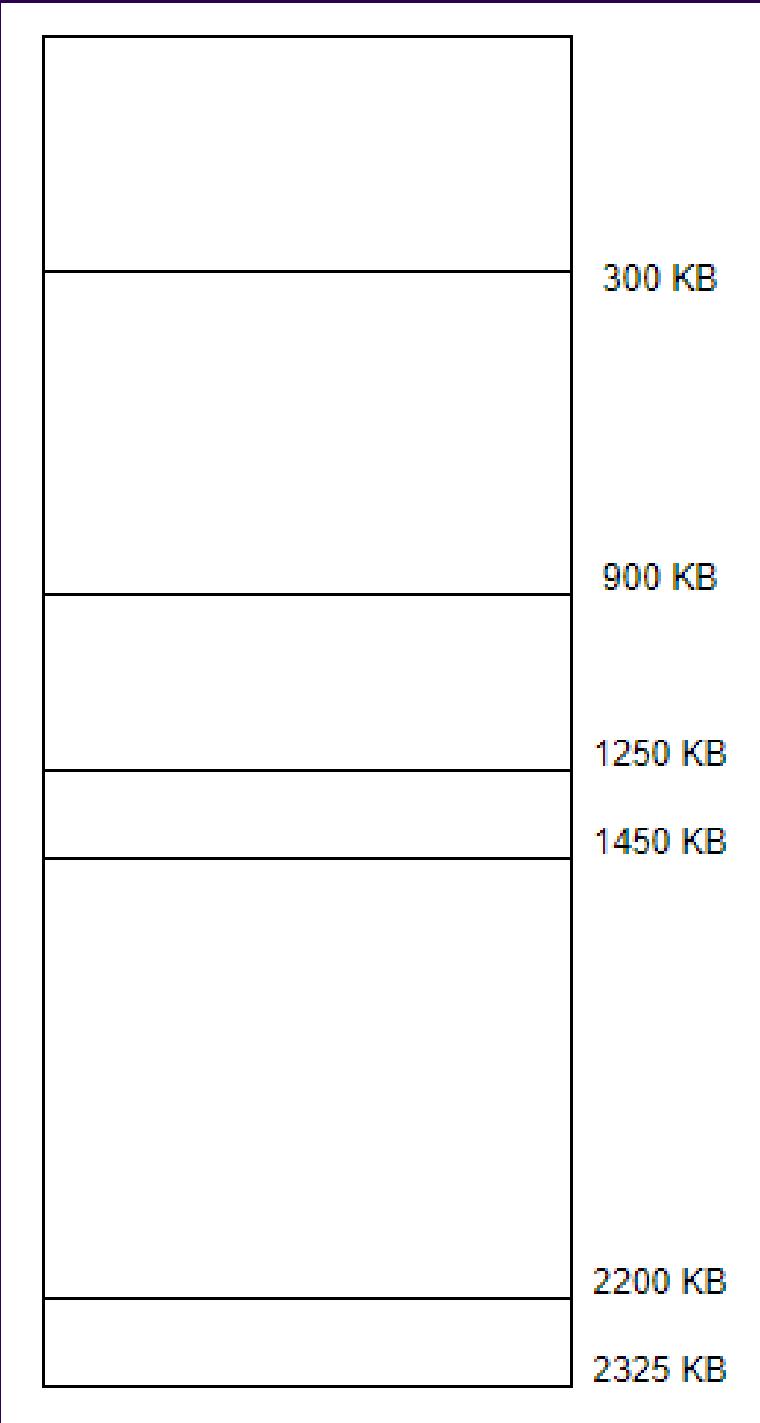
También existe la **segmentación**, dividir el proceso en fracciones (bloques) que sea más fácil acomodar. Estos bloques tendrán su base y su límite, el MMU lleva registro de esto en su **segment table**.

Segment table

Segment	Base	Size
Code	32768 (32KB)	2048 (2KB)
Heap	34816 (34KB)	2048 (2KB)
Stack	8672 (28KB)	2048 (2KB)



Ejercicios



Dado seis espacios de memoria de 300 KB, 600 KB, 350 KB, 200 KB, 750 KB y 125 KB (en ese orden), ¿Como colocarían los algoritmos first-fit, best-fit y worst-fit los siguientes procesos P₁ 500 KB, P₂ 95 KB, P₃ 358 KB, P₄ 200 KB y P₅ 375 KB (en ese orden)?



Ejercicios

Imagine un escenario en el que la memoria principal tiene una capacidad de 300KB y, en un principio, no alberga ningún proceso. A partir de este punto de partida, considere la siguiente secuencia de eventos:

- (a) Llegada del proceso P₁ con un tamaño de 47KB.
- (b) Llegada del proceso P₂ con un tamaño de 85KB.
- (c) Llegada del proceso P₃ con un tamaño de 108KB.
- (d) Finalización del proceso P₂.
- (e) Llegada del proceso P₄ con un tamaño de 60KB.
- (f) Finalización del proceso P₁.
- (g) Llegada del proceso P₅ con un tamaño de 25KB.
- (h) Llegada del proceso P₆ con un tamaño de 10KB.

Se solicita mostrar la evolución de la memoria, paso a paso, utilizando la política worst-fit.



Ejercicios

A partir de la Segmentation Table y de la figura, transformar las direcciones virtuales en físicas

- 1. 4200
- 2. 16300
- 3. 1520

Segmento	Base	Tamaño	Up
Code	32768 (32 KB)	2048 (2 KB)	I
Heap	34816 (34 KB)	2048 (2 KB)	I
Stack	28672 (28 KB)	2048 (2 KB)	O

Ejercicios

a. Dado un direccionamiento lógico binario 1001 0110 0000, determine el segmento al que pertenece y su dirección física en binario.

b. Si se intenta acceder a la dirección lógica en binario 1101 1010 1100, determine el segmento al que pertenece y su dirección física en binario.

Segmento	Base	Tamaño
Code	1000	2000
Heap	3000	4000
Stack	7000	500