Sistemas Operativos
1° Parcial 1C2018 - TT - Resolución

<u>Aclaración</u>: La mayoría de las preguntas o ejercicios no tienen una única solución. Por lo tanto, si una solución particular no es similar a la expuesta aquí, no significa necesariamente que la misma sea incorrecta. Ante cualquier duda, consultar con el/la docente del curso.

Teoría

1.

	RR	VRR	HRRN
Overhead	Intermedio	Alto	Alto
Monop. CPU	Imposible	Imposible	Posible
Starvation	Imposible	Imposible	Imposible

2. Un proceso puede ser creado/finalizado por el sistema operativo o por otro proceso. En caso de ser realizado por otro proceso, la operación se realiza a través de una llamada al sistema. Si el proceso padre finaliza inesperadamente,| los procesos hijos pueden seguir ejecutando dado que son entidades autosuficientes (aún si son adoptados por otro proceso como su nuevo padre)
3.
    - Mutua exclusión obligatoria
    - No expropiación de recursos
    - Retención y espera
    - Espera circular

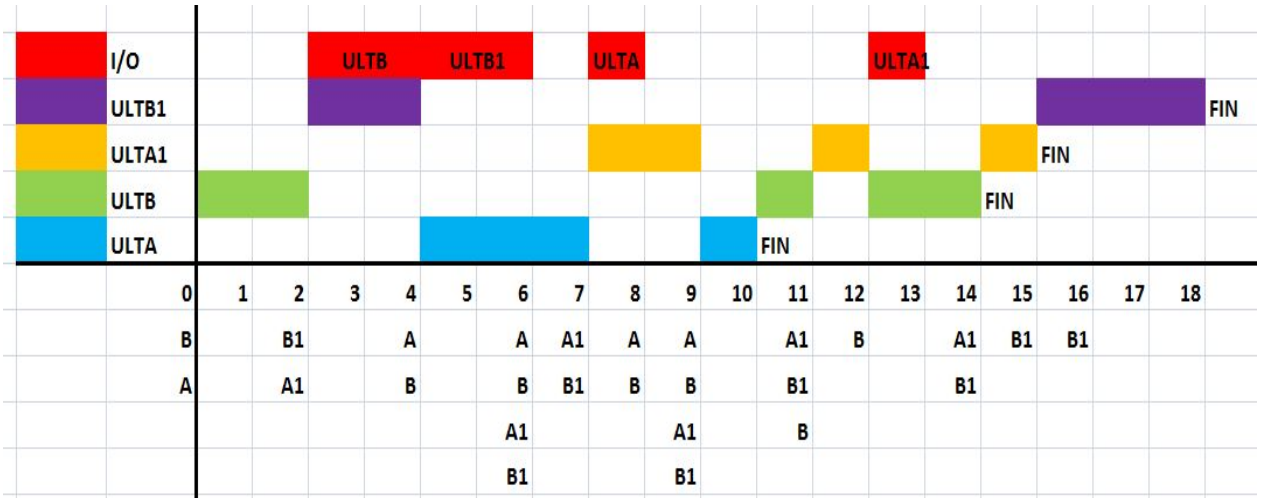
Las desventajas de una estrategia de recupero son que no pueden aplicarse en sistemas donde el deadlock no puede permitirse bajo ninguna circunstancia, y también el hecho de que al recuperarse del deadlock quitando un recurso o finalizando un proceso implica un impacto negativo para dicha víctima.

4.
    a. Verdadero. Para manipular los semáforos utilizamos syscalls provistas por el SO, por lo que es necesario cambiar a modo kernel.
    b. Falso. Los planificadores que modifican el grado de multiprogramación son los de mediano y largo plazo, ya que son los que pueden decidir cargar o sacar un proceso de memoria - disco.
5.
    a. Genera la interrupción de sw.

b. Cambio a modo kernel + cambio de la base del stack por la del stack del kernel.
c. Identificar en la syscall table qué rutina utilizar para atender la misma.
d. Ejecutar la rutina.
e. Mover el resultado de la syscall al stack del programa.

Practica

1.



1b1) En el instante cero que podría entrar ult A porque ambos inicializan en t=0 y al ser fifo ingresa ultA y luego ult a1 y al volver DEL QUANTUN elige alt A porque salió por S.O y vuelve el mismo hilo. Y después de la E/S elije ult B por aplicar algoritmo FIFO.

1B2) En T=2 ejecutaría ULTA porque al ser Jacketing no bloquea la primera instancia al irse ULTB a I/O. en T=4 Continuaría con ULTB 1 y luego con ULTA1

2.

a) SSOL# = 4, SRE = SDO = SSOL = 0

Saxo Tenor	Saxo Alto	Saxo Soprano	Saxo Sopranino

`while(true){`
`  wait(SRE)//x4`
`  tocar(DO#)`
`  tocar(DO#)`
`  signal(SDO)`
`}`
	`while(true){`
	`  wait(SSOL#)`
	`  tocar(RE#)`
	`  signal(SRE)`
	`}`
		`while(true){`
		`  wait(SDO)`
		`  tocar(SOL)`
		`  signal(SSOL)`
		`}`
			`while(true){`
			`  wait(SSOL)`
			`  tocar(SOL#)`
			`  signal(SSOL#)//x4`
			`}`

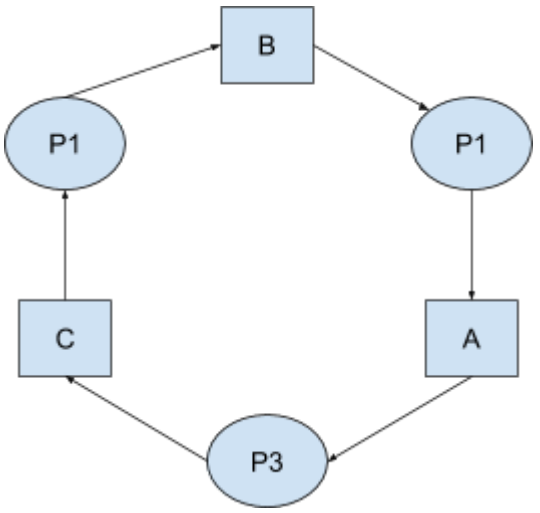
b)
SM = 3

Proceso incrementador/decrementador	Proceso principal o primer hilo

If (incrementar){
  signal(SM)
} else {
  wait(SM)
}
	wait(SM)
	creación_cuatro_hilos();
	esperar_finalizacion_hilos();
	signal(SM)

3.
a)

- P1: pedir B
- P2: pedir C
- P3: pedir A
- P1: pedir A (se bloquea)
- P2: pedir B (se bloquea)
- P3: pedir C (se bloquea)



b)
Técnica 1: evitar la retención y espera:

Proceso 1	Proceso 2	Proceso 3

`pedir_recursos(B,A,C)`
`usar_recursos(A,B,C)`
`liberar_recursos()`
	`pedir_recursos(B,A,C)`
	`usar_recursos(A,B,C)`
	`liberar_recursos()`
		`pedir_recursos(B,A,C)`
		`usar_recursos(A,B,C)`
		`liberar_recursos()`

Técnica 2: evitar la espera circular:

Proceso 1	Proceso 2	Proceso 3

`pedir_recurso(A)`
`pedir_recurso(B)`
`pedir_recurso(C)`
`usar_recursos(A,B,C)`
`liberar_recursos()`
	`pedir_recurso(A)`
	`pedir_recurso(B)`
	`pedir_recurso(C)`
	`usar_recursos(A,B,C)`
	`liberar_recursos()`
		`pedir_recurso(A)`
		`pedir_recurso(B)`
		`pedir_recurso(C)`
		`usar_recursos(A,B,C)`
		`liberar_recursos()`