PREGUNTAS TAREA 5:

ALEJANDRO GÜEREÑA MORÁN

JESÚS DAVID SANCHEZ CARBALLO

1. Explica por qué con la solución con procesos, a diferencia de la solución con hilos es necesario solicitar memoria compartida al sistema operativo.

Porque los procesos no comparten variables globales como lo hacen los hilos, por eso es necesario compartir estas variables entre procesos con el uso de memoria compartida.

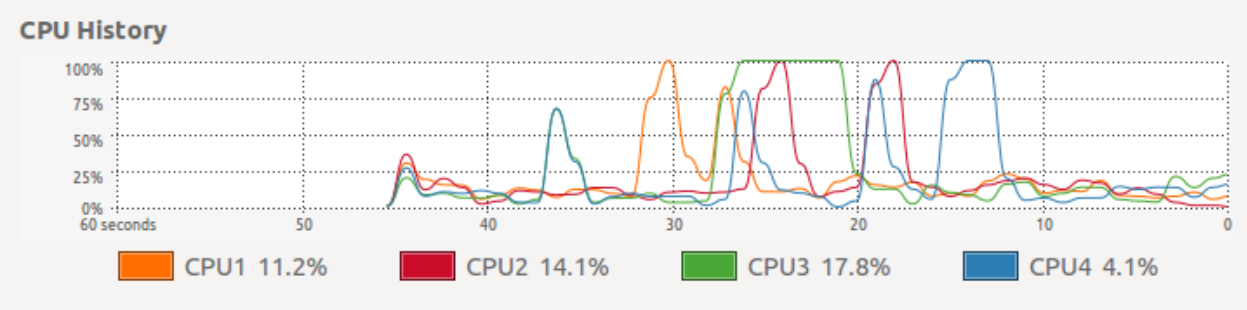
Ejecuta el monitor del sistema de Linux o cualquier utilería que te permita monitorear el uso del CPU en Linux.

Se uso el comando

while [1]; do ps --no-heading -C Ejemplo -L -o command,psr,pcpu|sort -k 2 -n; echo; sleep 1; done

para monitorear cada período de tiempo la utilización en % de los CPUs de cada ejemplo durante su ejecución.

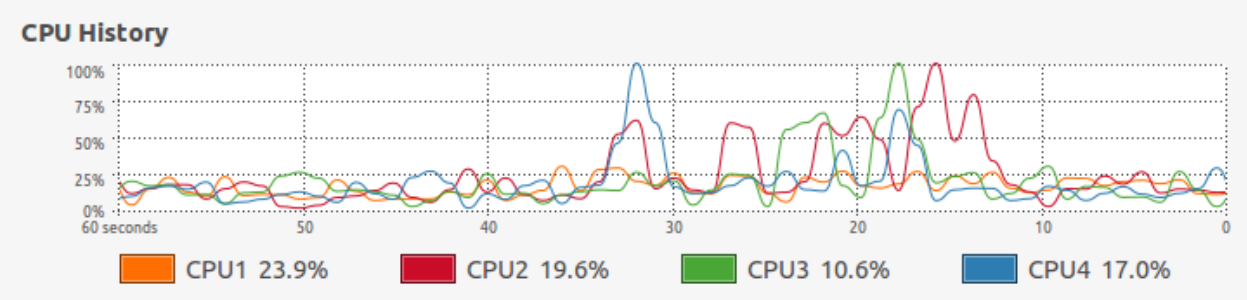
2. ¿Cuál es la utilización del CPU durante la ejecución del Ejemplo 1?



./xchg	**0 0.0**			**./xchg**	**3 0.0**
./xchg	**1 73.0**	**./xchg**	**0 24.7**		
./xchg	**2 73.0**	**./xchg**	**1 49.7**	**./xchg**	**0 14.1**
./xchg	**3 0.0**	**./xchg**	**2 24.7**	**./xchg**	**2 42.7**
		./xchg	**3 0.0**	**./xchg**	**3 0.0**
./xchg	**0 38.0**			**./xchg**	**3 55.5**
./xchg	**1 88.0**	**./xchg**	**0 19.8**		
./xchg	**2 49.5**	**./xchg**	**1 39.8**	**./xchg**	**0 24.0**
./xchg	**3 0.0**	**./xchg**	**2 36.8**	**./xchg**	**2 37.3**
		./xchg	**3 0.0**	**./xchg**	**3 0.0**
./xchg	**0 33.0**			**./xchg**	**3 61.3**
./xchg	**1 66.3**	**./xchg**	**0 16.5**		
./xchg	**2 33.0**	**./xchg**	**0 47.8**	**./xchg**	**0 22.1**
./xchg	**3 0.0**	**./xchg**	**2 48.0**	**./xchg**	**2 33.2**

./xchg	3 0.0	./xchg	3 56.1		
./xchg	3 65.8			./xchg	0 40.8
		./xchg	2 41.7	./xchg	2 23.8
./xchg	0 19.9	./xchg	3 0.0	./xchg	3 0.0
./xchg	2 29.9	./xchg	3 23.2	./xchg	3 44.3
./xchg	3 0.0	./xchg	3 52.8		
./xchg	3 69.4			./xchg	0 42.2
		./xchg	2 39.7	./xchg	3 0.0
./xchg	0 18.0	./xchg	3 0.0	./xchg	3 42.6
./xchg	2 36.1	./xchg	3 21.9		
./xchg	3 0.0	./xchg	3 49.9	./xchg	0 40.7
./xchg	3 63.5			./xchg	3 0.0
		./xchg	0 20.7	./xchg	3 41.0
./xchg	0 16.5	./xchg	2 43.1		
./xchg	2 41.5	./xchg	3 0.0	./xchg	0 39.6
./xchg	3 0.0	./xchg	3 47.3	./xchg	1 39.2
./xchg	3 58.2			./xchg	3 0.0
		./xchg	0 20.4		
./xchg	1 15.6	./xchg	2 45.4	./xchg	0 38.2
./xchg	2 39.0	./xchg	3 0.0	./xchg	1 37.8
./xchg	3 0.0	./xchg	3 44.9	./xchg	3 0.0
./xchg	3 53.7				
		./xchg	0 23.5	./xchg	0 36.9
./xchg	0 42.5	./xchg	2 43.3	./xchg	1 36.6
./xchg	1 21.0	./xchg	3 0.0	./xchg	3 0.0
./xchg	3 0.0	./xchg	3 42.8		
./xchg	3 50.7			./xchg	1 35.4
		./xchg	1 22.2	./xchg	3 0.0
./xchg	0 39.7	./xchg	3 0.0		
./xchg	1 20.5	./xchg	3 39.0	./xchg	1 34.3
./xchg	3 0.0	./xchg	3 40.2	./xchg	3 0.0
./xchg	3 54.1				
		./xchg	0 38.2	./xchg	1 33.3
./xchg	1 24.6	./xchg	1 24.7	./xchg	3 0.0
./xchg	2 38.1	./xchg	3 0.0		
./xchg	3 0.0	./xchg	3 42.8		

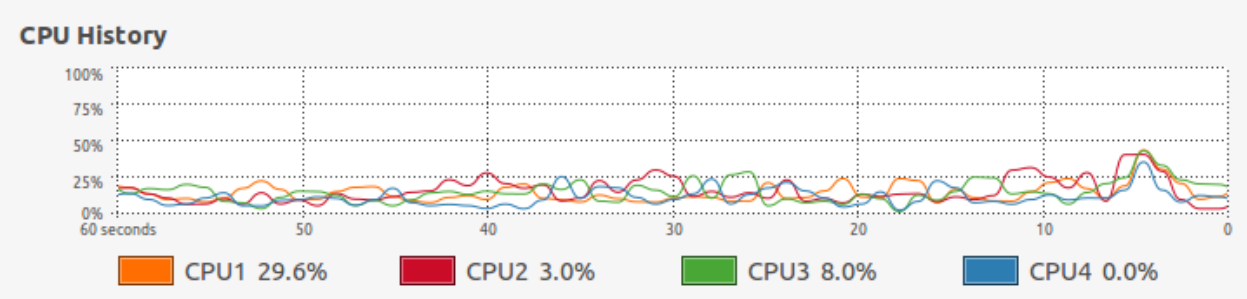
3. ¿Cuál es la utilización del CPU durante la ejecución del Ejemplo 2?



./shared_memory	0 0.0	./shared_memory	0 0.0	./shared_memory	0 0.0
./shared_memory	0 0.0	./shared_memory	0 0.0	./shared_memory	0 0.0
./shared_memory	1 0.0	./shared_memory	1 99.0	./shared_memory	1 49.5
./shared_memory	3 0.0	./shared_memory	3 132	./shared_memory	3 99.0

```
./shared_memory  0  0.0
./shared_memory  1 33.0
./shared_memory  1 66.3
./shared_memory  2  0.0

./shared_memory  0  0.0
./shared_memory  1 24.7
./shared_memory  1 49.7
./shared_memory  2  0.0

./shared_memory  0  0.0
./shared_memory  1 39.8
./shared_memory  2  0.0
./shared_memory  2 19.8

./shared_memory  0  0.0
./shared_memory  1 40.8
./shared_memory  2  0.0
./shared_memory  2 16.5

./shared_memory  0  0.0
./shared_memory  1 42.7
./shared_memory  2 14.1
./shared_memory  3  0.0

./shared_memory  0  0.0
./shared_memory  1 37.3
./shared_memory  2 12.3
./shared_memory  3  0.0

./shared_memory  0  0.0
./shared_memory  1 33.2
./shared_memory  2 17.2
./shared_memory  3  0.0

./shared_memory  0  0.0
./shared_memory  1 27.1
./shared_memory  2  0.0
./shared_memory  2 18.3

./shared_memory  0  0.0
./shared_memory  1 30.1
./shared_memory  2  0.0
./shared_memory  3 22.0

./shared_memory  0  0.0
./shared_memory  1 30.6
./shared_memory  2  0.0
./shared_memory  3 23.2

./shared_memory  0  0.0
./shared_memory  1 26.3
./shared_memory  2 28.5
./shared_memory  3  0.0

./shared_memory  0  0.0
./shared_memory  0  0.0
./shared_memory  1 26.8
./shared_memory  2 31.1

./shared_memory  0  0.0
./shared_memory  1 25.1

./shared_memory  2 35.6
./shared_memory  3  4.3

./shared_memory  0  0.0
./shared_memory  1 28.2
./shared_memory  2 35.2
./shared_memory  3  5.8

./shared_memory  0  0.0
./shared_memory  1 32.4
./shared_memory  2 33.2

./shared_memory  0  0.0
./shared_memory  1 31.6
./shared_memory  2 31.5

./shared_memory  0  0.0
./shared_memory  1 34.4
./shared_memory  2 30.0

./shared_memory  0  0.0
./shared_memory  1 33.4
./shared_memory  2 28.5

./shared_memory  0  0.0
./shared_memory  1 31.9
./shared_memory  2 27.2

./shared_memory  0  0.0
./shared_memory  3 30.5
```

4. ¿Cuál es la utilización del CPU durante la ejecución del Ejemplo 3 el cuál es se sincroniza utilizando nuestra implementación de semáforos?




```
/semaphores  0  0.0
./semaphores  0  0.0
./semaphores  1  0.0
./semaphores  3  0.0

./semaphores  0  0.0
./semaphores  0  0.0
./semaphores  1  0.0
./semaphores  3  0.0

./semaphores  0  0.0
./semaphores  0  0.0
./semaphores  1  0.0

./semaphores  2  0.0

./semaphores  0  0.0
./semaphores  0  0.0
./semaphores  1  0.0
./semaphores  2  0.0

./semaphores  1  0.0
./semaphores  2  0.0
./semaphores  2  0.0
./semaphores  2  0.0

./semaphores  1  0.0

./semaphores  2  0.0
./semaphores  2  0.0
./semaphores  2  0.0

./semaphores  1  0.0
./semaphores  2  0.0
./semaphores  2  0.0
./semaphores  2  0.0

./semaphores  1  0.0
./semaphores  2  0.0
./semaphores  2  0.0
./semaphores  3  0.0
```

```
./semaphores 1 0.0
./semaphores 2 0.0
./semaphores 2 0.0
./semaphores 3 0.0
```

./semaphores	**0 0.0**	**./semaphores**	**3 0.0**	**./semaphores**	**0 0.0**
./semaphores	**0 0.0**			**./semaphores**	**0 0.0**

5. Existe diferencia entre la utilización del CPU en la ejecución de las soluciones anteriores, ¿cuál de todas es mejor? explica por qué.

Si, si existe, principalmente entre el ejemplo de semáforos y los demás ejemplos. El ejemplo de semáforos tiene una utilización de CPU muy cercana a 0, debido a que los procesos no tienen espera ocupada para entrar a la sección crítica, ya que estos son bloqueados. Cuando pueden entrar a la sección crítica, es cuando son ejecutados.