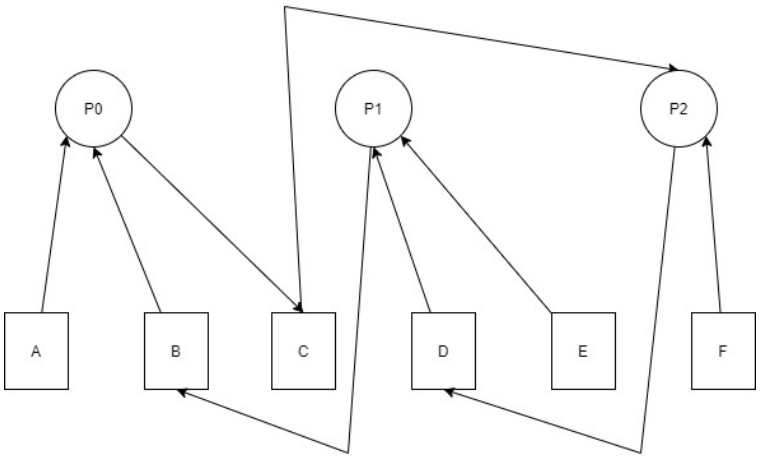
Stephen Sallas
03/19/2021
COMP-3500

HW2

1. a.

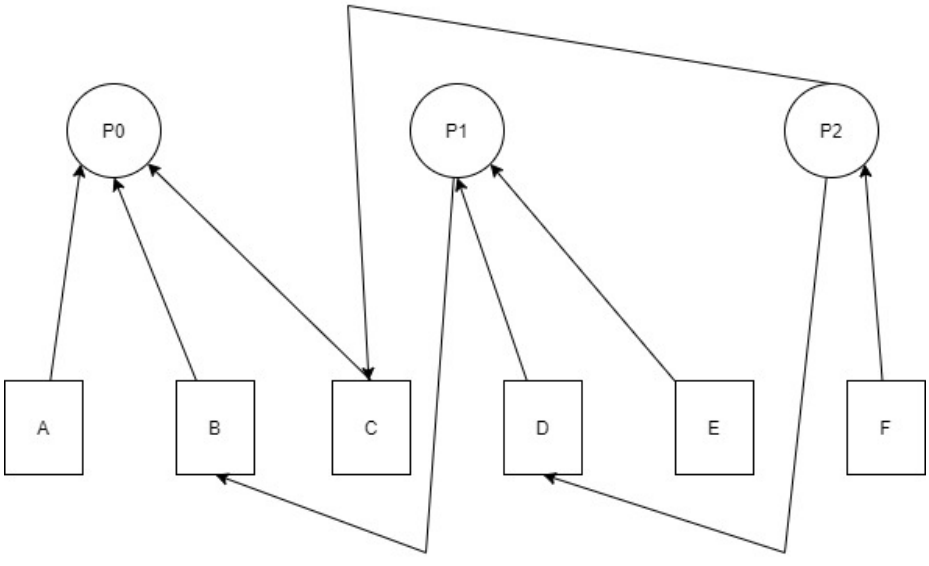


P0 holds resources A and B and waits on C. P1 holds resources D and E and waits on B. P2 holds C and F and waits on D. This cycle results in deadlock.

b.

Process P0	Process P1	Process P2

void p0()
{
while(true)
{
get(B);
get(C);
get(A);
release(A);
release(B);
release(C);
}
}
	void p1()
	{
	while(true)
	{
	get(D);
	get(E);
	get(B);
	release(D);
	release(E);
	release(B);
	}
	}
		void p2()
		{
		while(true)
		{
		get(F);
		get(D);
		get(C);
		release(F);
		release(D);
		release(C);
		}
		}



Stephen Sallas
03/19/2021
COMP-3500

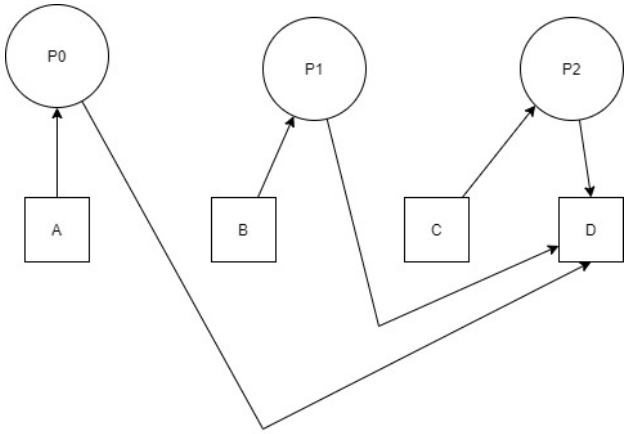
2. Yes, if both processes execute semWait(S) and semWait(R) in at the same time, so they will both be blocked when each process tries to execute the next instruction. This is because each process is waiting on the other to call semSignal() which will never happen, so execution will never resume.

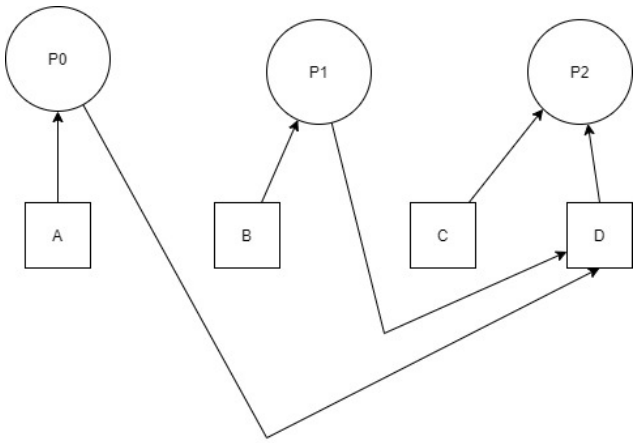
3. Deadlock avoidance means the user must analyze each resource request and determine if it could lead to deadlock, and in-turn ensure that the system does not go into deadlock.

   Deadlock prevention guarantees that deadlock won't occur by preventing either mutual exclusion, hold and wait, no preemption, or circular wait.

   Deadlock detection is the attempt to detect the presence of deadlock and take action to recover.

4. Here is a resource allocation graph reduced down to prove the system is deadlock free.





Stephen Sallas
03/19/2021
COMP-3500