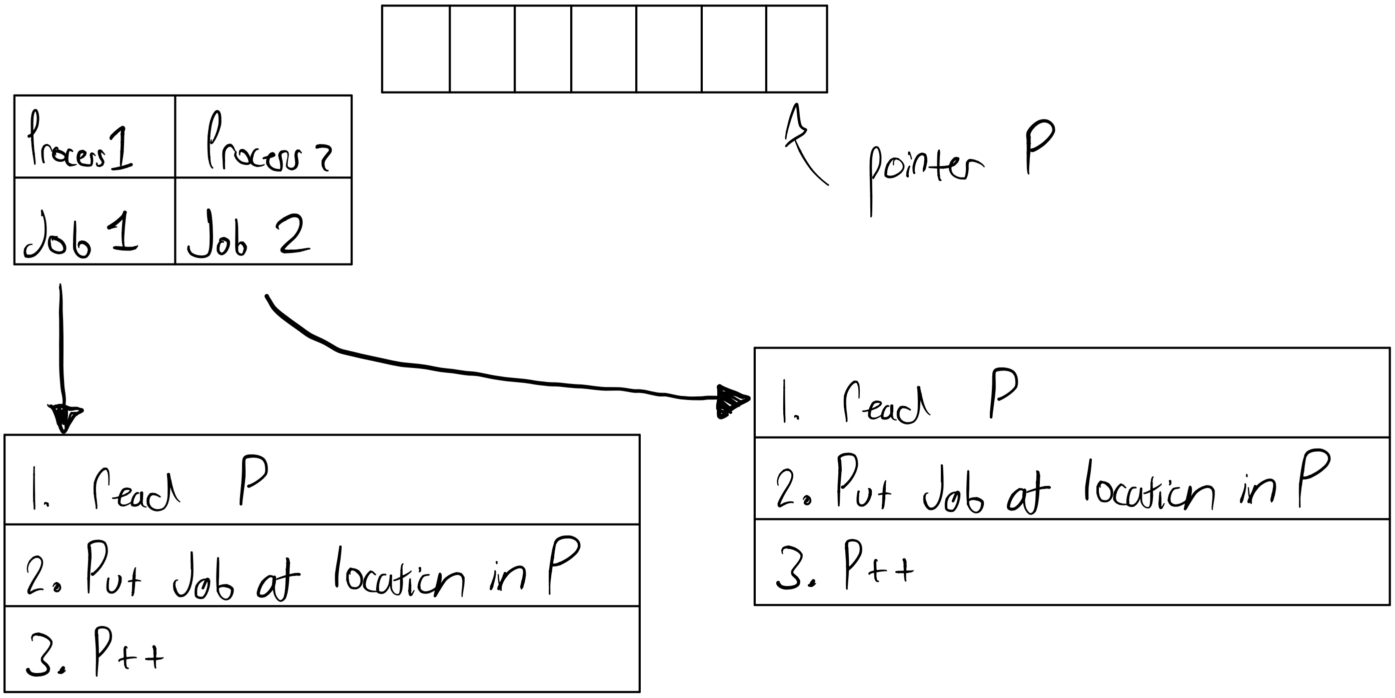
9/17/18 Day 11

PT

- Events
  - listen for programs
    ```
    - if ( today == "off" ) {
    - if program poc.exe is "seen" close operation
    }
    ```

Mutual exclusion - Chapter 5

Suppose we have a print spooler (Queue for print)



pointer P

Process 1	Process 2
Job 1	Job 2

Process 1:

1. read P
2. Put Job at location in P
3. P++

Process 2:

1. read P
2. Put Job at location in P
3. P++

Note if a timeout occurs after reading P an error occurs where one will write over the other this happens when resources are shared

Critical section the area where we do not want to edit memory. Programs won't have infinite loops in critical sections.

How do we avoid this?

Process 1	Process 2

while true
{
while (turn == 0);
{ < Critical code >
turn = 1;
⋮
}
}
	while true
	{
	while (turn == 1);
	{ < Critical code >
	turn = 0;
	⋮
	}
	}

// is this mutually exclusive.?

we cannot assume that both processes are taking the same time, so we do have mutual exclusion but we have to take turns through strict alternation,

Process 1	Process 2

while flag[1] do ;
flag[0] = true
<critical section>
flag[1] = false
⋮
	while flag[0] do ;
	flag[1] = true
	<critical section>
	flag[1] = false
	⋮

// here both processes could grab the flag at the
// same time if they finish at the same tim

Process 1	Process 2
1. flag[0] - true while flag[1] do <critical section> flag[0] = false ⋮	flag[1] - true while flag[1] do <critical section> flag[1] = false ⋮

// this is live lock

if after line 1 a timeout occurs we can deadlock

- Race condition - when things happen they may or may not work.
  hard to recreate.

9/19/18 Day 12

$\int \frac{1}{e^{x^2}} = e^u$ $u = -x^2$ $du = -2x\,dx$

$\frac{du}{-2x} = dx$

$\frac{e^u}{-2x} = \left[\frac{e^{-x^2}}{-2x}\right]$

$\lim \frac{e^{-x^2}}{-2x} = \frac{1}{-2xe^{x^2}}$

$\frac{1}{e^{-x^2}} \cdot \frac{1}{-2x}$ can't apply lahopitals

Grading criteria paper

- unified
- to the point
- logical
- correct in grammar
- appropriate to audience

9/21/18 Day 13

Deadlock

- For the next test we need to be ready to read psuedo code determine its code is mutually exclusive, based on turn order

define TRUE

Process 0

```
while true
{ flag [0] = true
  while ( flag[1] )
  {  if  (turn=1 )
     {  flag[0] = false
        while  (turn==1 );
        flag [0] = true
     }
  }
  <critical sect >
   turn = 1
   flag [0] = false
       ⋮
}
```

Process 1

```
while true
{ flag [1] = true
  while ( flag[0] )
  {  if  (turn=0)
     {  flag[1] = false
        while  (turn==0 );
        flag [1] = true
     }
  }
  <critical sect >
   turn = 0
   flag [1] = false
    ⋮
}
```

// this method works

// Dekker's Algorithm

Next paper

- these methods in distributed environments
- processes need to communicate to be able to control access to resources
  - I.e: token ring

P0

```
flag[0] = true
turn = 1
while(flag[1] && turn == 1) // busy wait only when both
<C.S.>                      // true can we proceed
flag[0] = false
⋮
```

Peterson's Algorithm

TODO: he is going to rearrange one of these on the test.

Hw 1 - Software solutions

#7, chapter 5 pg. 249

- works for any number of processes

- **Hardware Solutions**
  - disable interrupts
    - Not a good idea
  - Special instructions - Machine instruction
    - instructions cannot be interrupted

Example

Test and set instruction psuedo Code

```
boolean testset (int i)
{ if (i==0)
   { i=1
     return true;
   }
   else
    { return false; }
}
   ⋮
int bolt;
while true
{  while (! testset (bolt)); // busy wait lock
   Critical section
   bolt = 0;
}
```

Ex change instruction - Intel

Psuedo code

```
Void exchange (int register, int memory)
{ int temp
  tem = memory;
  memory = register;
  register = temp;
}
 // Hw how would this work as a lock
```

HW