chap 5

Board examples - critical sections, race conditions, worked @ the whiteboard

Race Condition (Find it)

```
int balance = 50;
1   void withdraw(int amount) {
2       if ( balance > amount) {
3           cout << "approved";
4           balance -= amount;
5       }
    }
```

2 threads start w balance $50

T1 withdraw 40 } 65 > 50!

T2 withdraw 25 }

Balance after	
50	T1 starts gets to 3 (past the filter)
50	preempted by T2,
25	T2 runs to end
-15	T1 switched back n finishes

Race Condition

- when program output depends on what gets done first, tough to reproduce n debug

ex.

```
        int global = 2;

(2)     void func() {
          if (global == 0)
                doZero(); (3)
          else
                doNotZero(); (4)
        }

        int main() {
critical    thread myt(func);
section     global = 0;
(1)     }
```

race condition,

if (1) happens before 2

then (3)

else (4)

who's gonna win?

PSA - you may see code that "solves" this with delays. (sleep_for(), sleep_until()) this is a non-scalable solution.

a critical section is an area of code where only 1 thread can be @ a time

Question: if launch no threads, do you have CS's?

```
void fun(){
      gi++;
}
int gi = 0;

int main(){
    Thread t1(Fun);
    int i = gi;

    i = i+1;

    gi = i;
    t1.join();
}
```

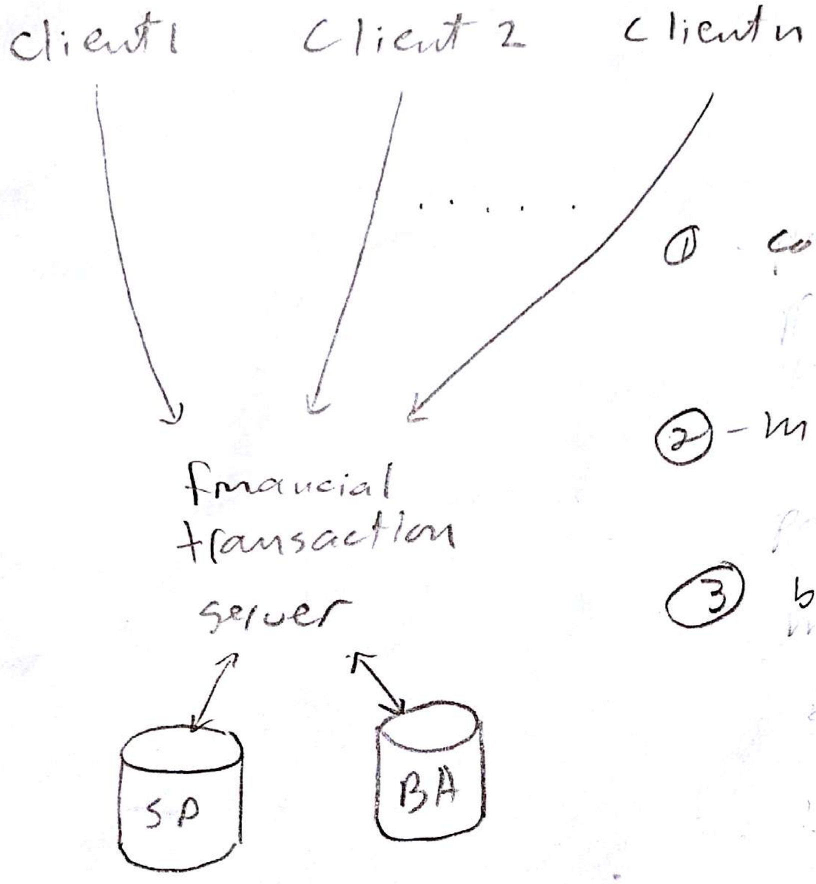
← where is crit sec.?

if no thread?

if thread does not access global?

if thread does access global?

chap 5 part 1   find critical sections



① cost = stockprice * shares

② - m = getmoney (acct, cost)

③ buy stock (sym, m, acct)

```
//global
int transactions=0;

getmoney (acct, cost){
        :
        :
    return m;
}

buystock (sym, m, acct){
    //record #xactions
    transactions ++;
        :
        :
}
```

① start w 1, 2, 3
global cost

② put ① ② ③ in func
no globals
(stack frame local vars)

③ explore getmoney buystock

④ add global transactions