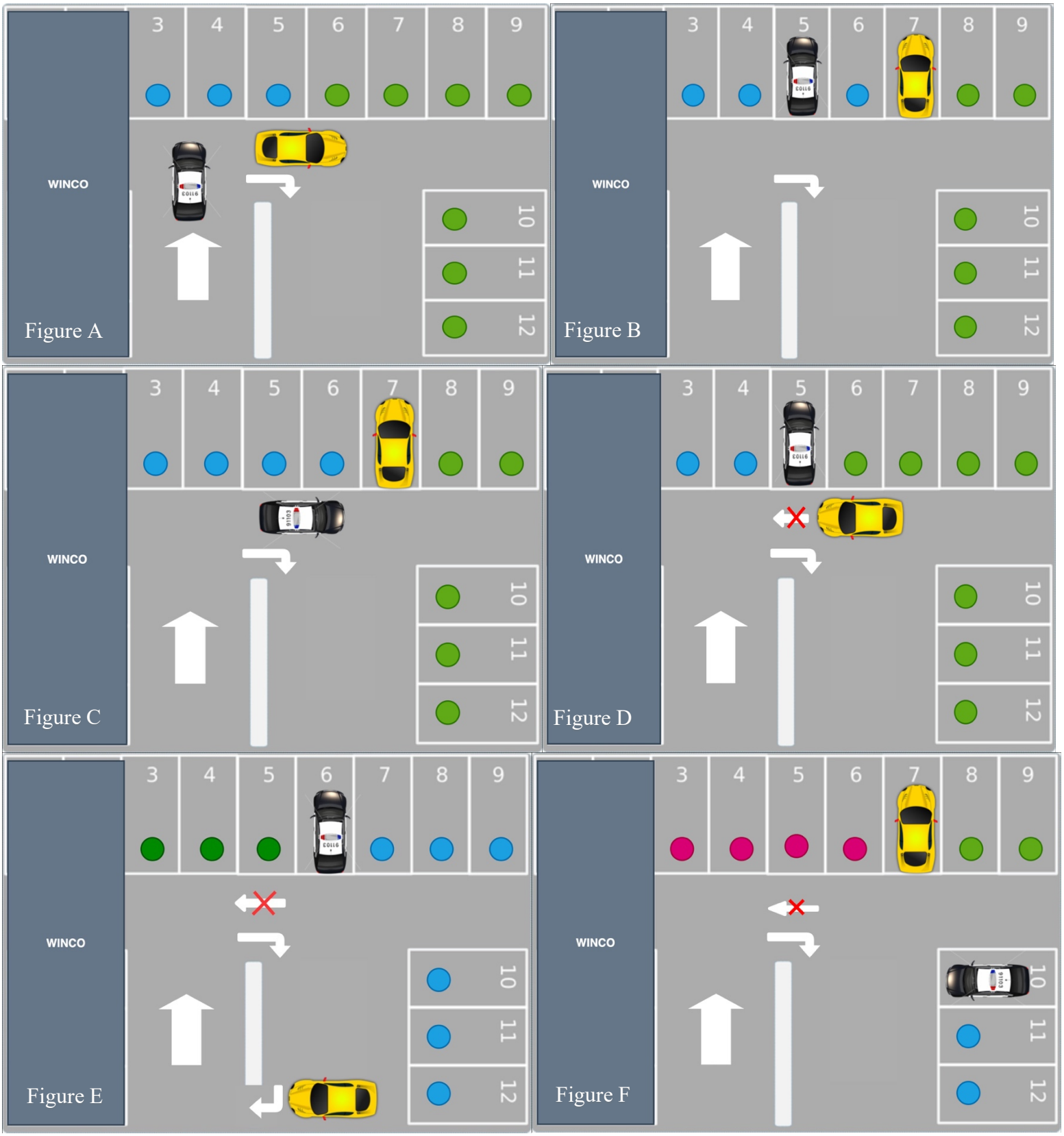
HOMEWORK#10

A transaction must acquire locks in a particular data order:

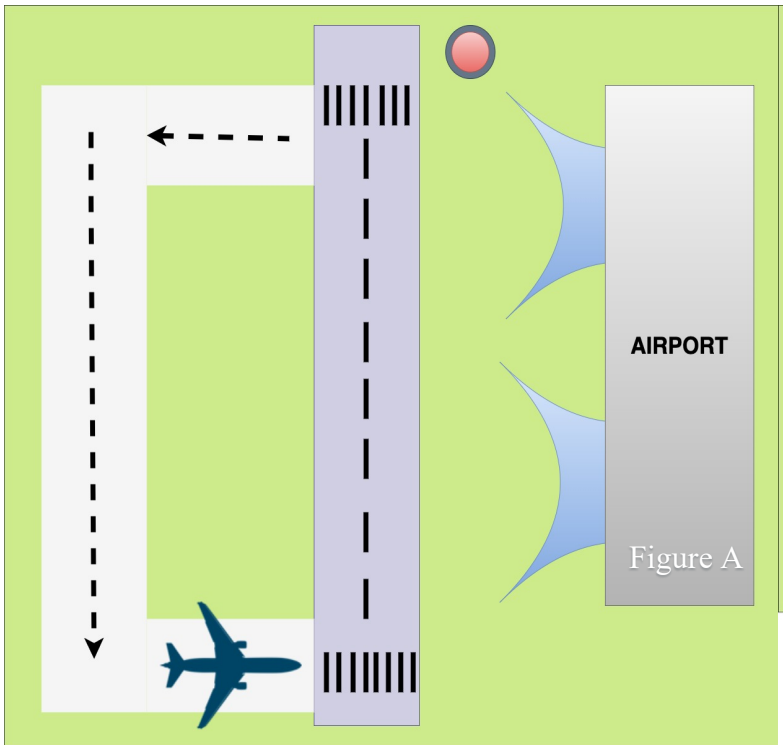
This can be explained by using following example of WINCO parking area.

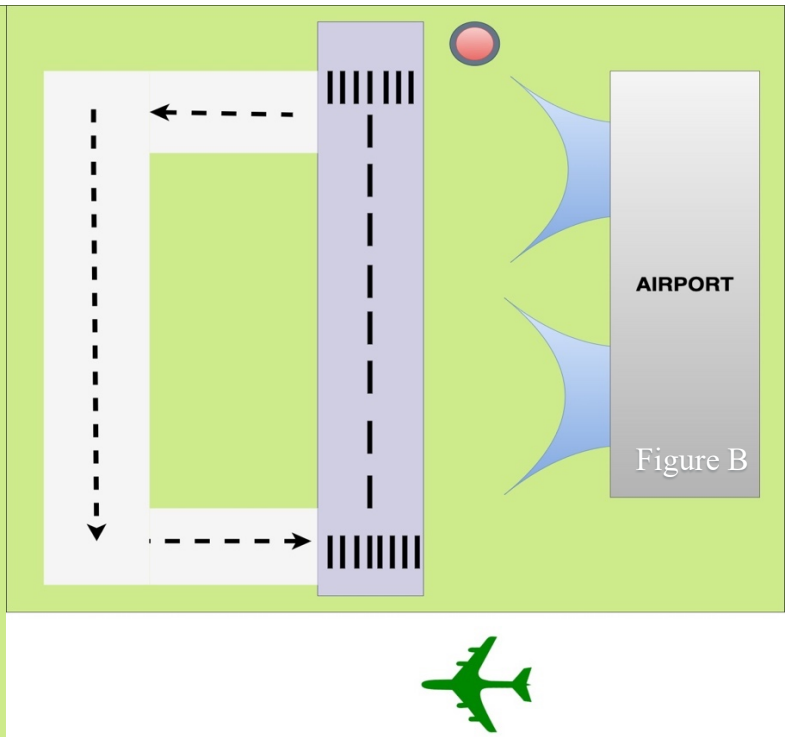


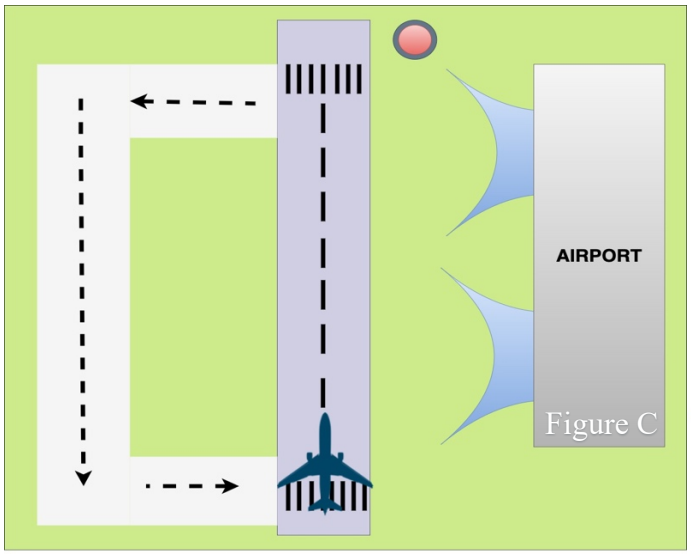
- In the parking area Figure A, we can see two car one is Black colour and other in Yellow colour.
- In Figure A, The Yellow car can access parking area from 6 to 12 denoted by **Green** Dot.
- In the same Figure A, the black car can access parking area from 3 to 12 denoted by **Blue** and **Green** dot.
- In Figure B, Black car locks parking no.5 and Yellow car locks parking no.7.
- In Figure C, Black car leaves parking no.5 and can lock any parking from 8 to 9.
- In same Figure C, the Black car can wait for Yellow car to leave the parking no.7 and then lock it for itself.
- In Figure D, Yellow car want to access parking area from 3 to 4 but it's a one-way road and it cannot lock it for itself.
- In Figure E, we can see to access and lock parking area from 3 to 5 the Yellow car have to go around the turn and then only it can access parking area 3 to 5.
- Alternative way for Yellow car is, to lock parking no. 3 from start and then move on remaining areas.
- In Figure E, the Black car locks parking area no.10.
- In same Figure, if Black car wants to lock any other parking areas it can only lock 10 or 12.
- If Black car wants to access parking area from 3 to 9 then it will have to go around the turn and then only it can lock any of these areas.
- Now in Figure E, Yellow car which is at Parking No.7 can now lock parking area from 8 to 12 including parking area 10.
- If Yellow car wants to access parking no.10 then it will have to wait for Black car to release the parking no.10 and once it did then it can lock it for itself.
- This way there is no deadlock were both cars want to lock same parking no. as the cars can lock the parking area in ascending order only.

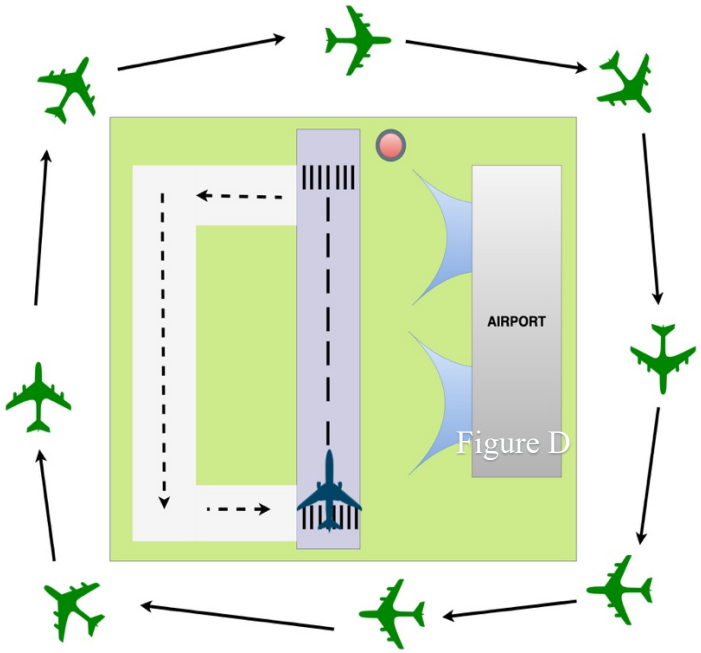
Timeout Based:

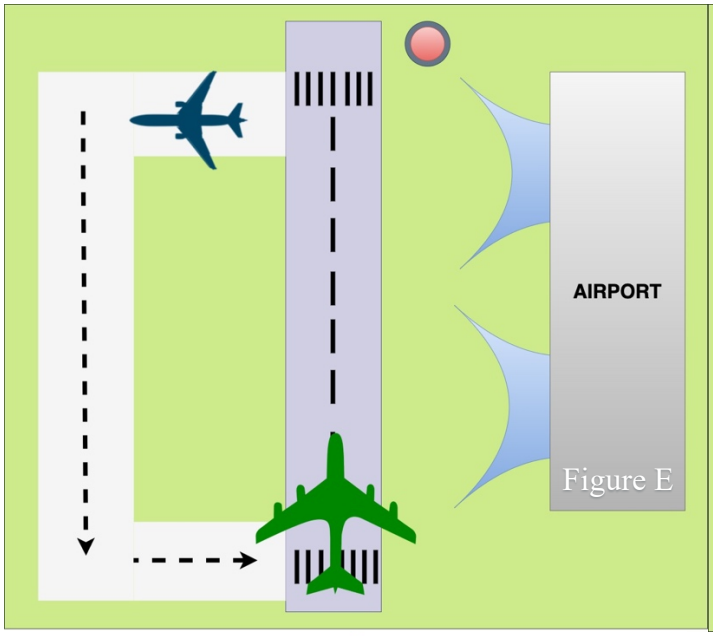
This can be explained by using following example of AIPORT RUNWAY.

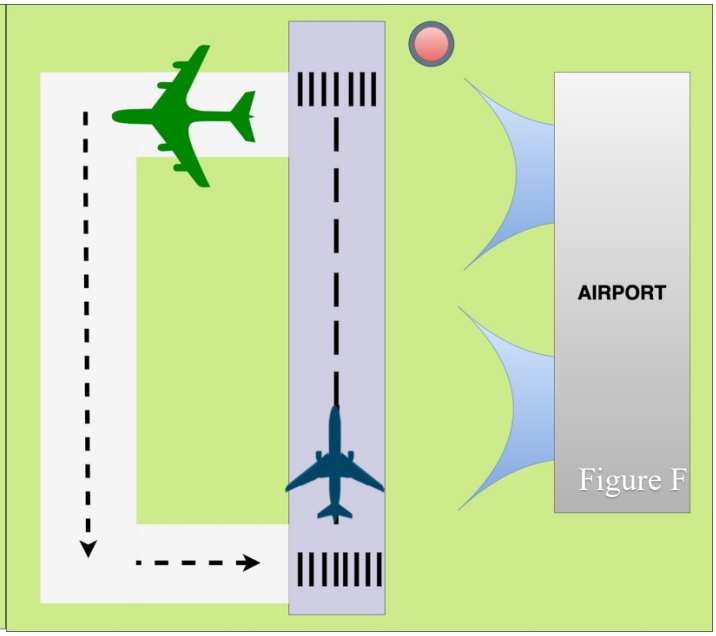


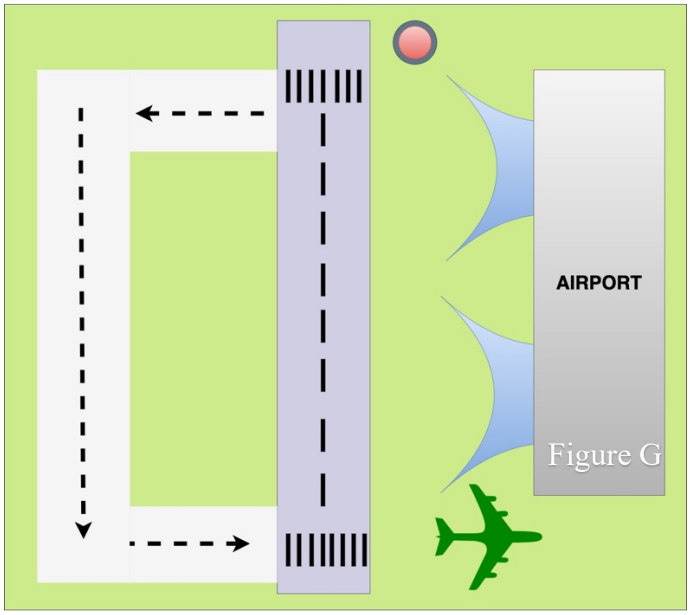



AIRPORT
Figure C


AIRPORT
Figure D


AIRPORT
Figure E

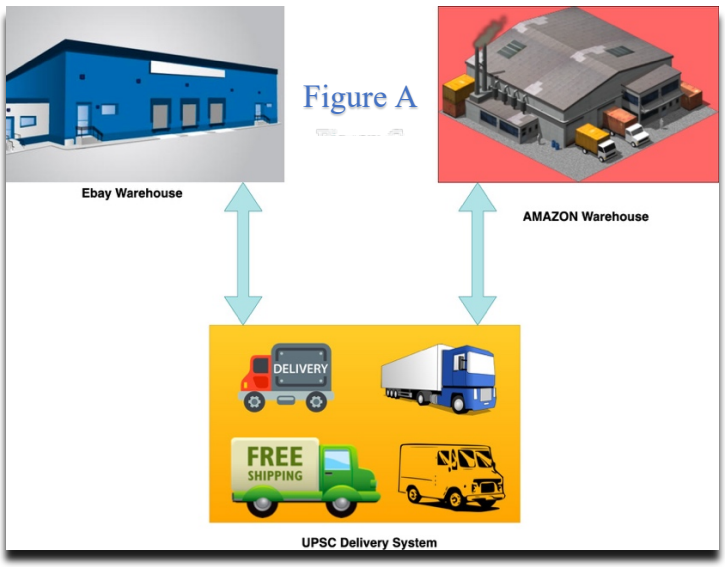

AIRPORT
Figure F

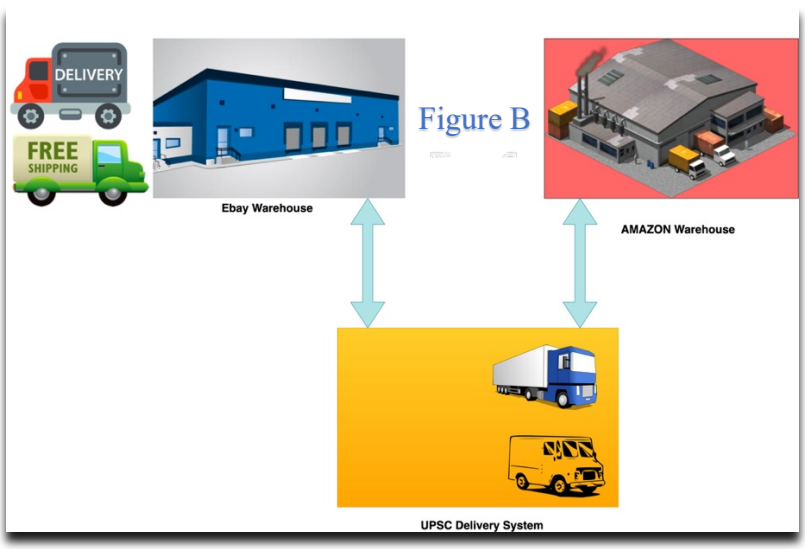

AIRPORT
Figure G

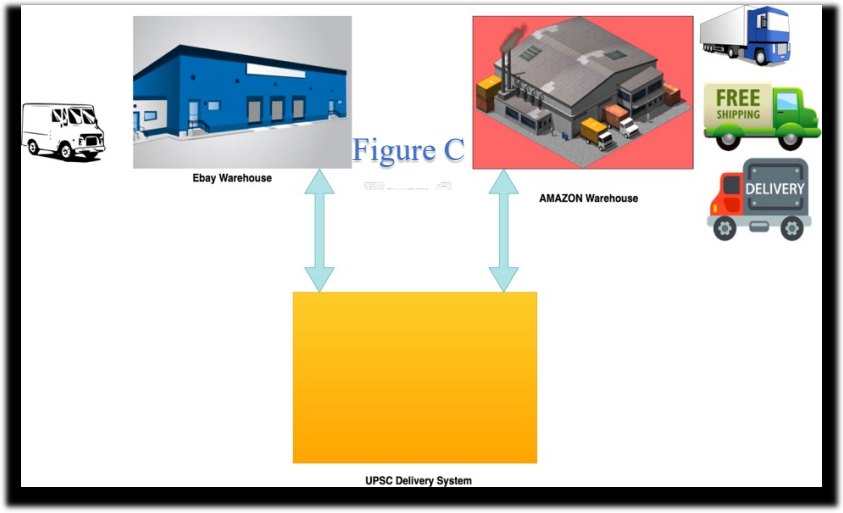
- In Figure A, we can a Blue flight which want to take-off and waiting for the control tower denoted by red circle to give green signal if runway can be used for takeoff.
- In Figure B, we can a Green flight which want to Land and waiting for the control tower denoted by red circle to give green signal if runway can be used for Landing the flight.
- Hence, as both the flight want to use runway that could cause a deadlock.
- We can see in Figure C, the Blue flight is already on runway track but it cannot take-off as control tower hasn't given green signal and as Blue flight is on runaway the Green flight cannot land.
- So, the Green flight will wait for some time for the Blue flight to clear the runway.
- In Figure D, The Green flight can see that even after waiting the Blue flight is still on runway, so it restarts its landing procedure and go around the airport assuming that till that round the Blue flight will clear the runway.
- The Blue flight can see that, even after waiting it cannot takeoff as runway is needed by other flight hence, it too restarts its procedure of takeoff and go around the sidetrack of runway.
- In Figure E, as Blue flight has cleared the runway and gone into the sidetrack, the control tower has given green signal for the Green flight to land and it has landed.
- Now, the Green flight will go from runway to sidetrack, so as that Blue flight can use the runway.
- In Figure F, the Blue flight can see that Green flight as gone into sidetrack and cleared the runway so, it comes on runway.
- In Figure G, we can see the Blue flight has finally taken offs, and the Green flight has gone into airport parking.
- This way the Deadlock is prevented using Timeout.

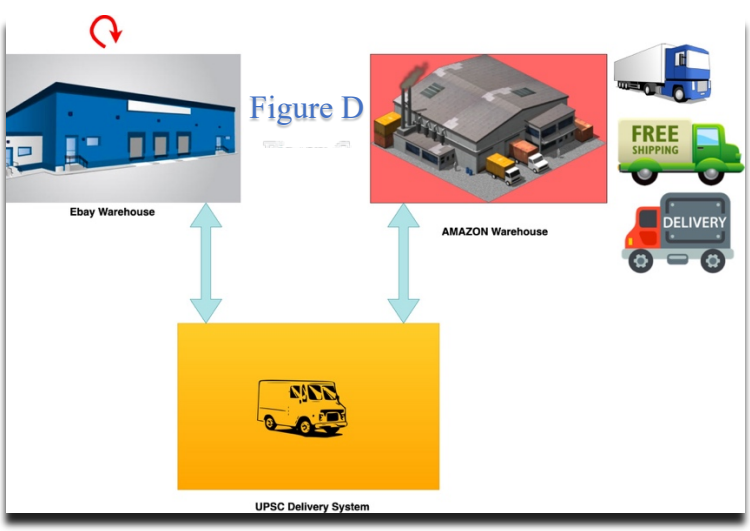
A transaction must acquire all locks before it begins:

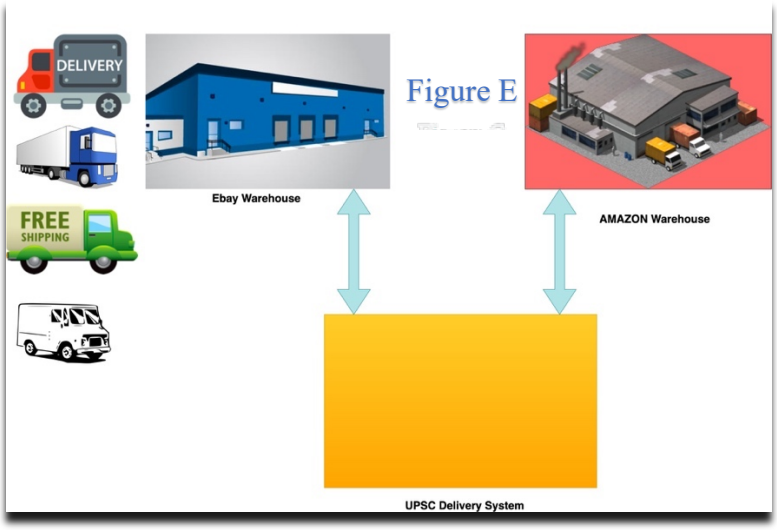
This can be explained by using following example of AMAZON and EBAY UPSC delivery system.











- In Figure A, we can see that Amazon warehouse uses UPSC delivery system truck to deliver the goods, also the eBay too uses UPSC delivery system truck to deliver its goods.
- The eBay requires 2 truck of UPSC to start delivering its Goods.
- The eBay will reserve the 2 trucks of UPSC for next 1hrs.
- In Figure B, we can see that eBay starts using the 2 trucks.
- Now, the Amazon requires 3 trucks of UPSC to start delivering it goods.
- But Amazon can see that UPSC only has 2 trucks available at the moment and other 2 trucks are used by eBay which are reserved for next 1 hr.
- Even though Amazon can start delivering goods with two truck and use 3rd truck later on when available, it waits for all the 3 trucks to be available.
- After 1hr, the eBay releases the previously reserved 2 trucks and return them to UPSC.
- Now, as UPSC has 4 truck available, the Amazon reserves the 3 truck for next 4hrs for itself.
- In Figure C, we can see Amazons requirements of 3 truck is finally fulfilled and it starts using the required 3 trucks for delivering goods.
- Now eBay again want 4 trucks of UPSC to deliver things.
- But in Figure C, we can see only 1 truck is available.
- Again eBay, can start using that 1 truck and use remaining 3 later on when available, but it sees that Amazon has reserved 3 trucks for next 4hrs.
- Hence, eBay waits for next 4hrs for the trucks to be available.
- Even though Amazon reserved the 3 trucks for 4hr it used them for 5hrs.
- In Figure D, we can see the eBay restarts its warehouse denoted by red arrow because the 3-truck needed are still unavailable even after 4hrs.
- After 5hrs Amazon releases the 3 truck and return them to UPSC.
- In Figure E, we can see that eBay now finally uses 4 truck, as they were available.
- This is how deadlock is prevented.
- Hence, unless and until the transaction can get a lock on all the resources it needed it does not start it process.

Wound wait:

Figure A

SENIOR

TOOLS

Figure B

TOOLS

SENIOR

Figure C

JUNIOR

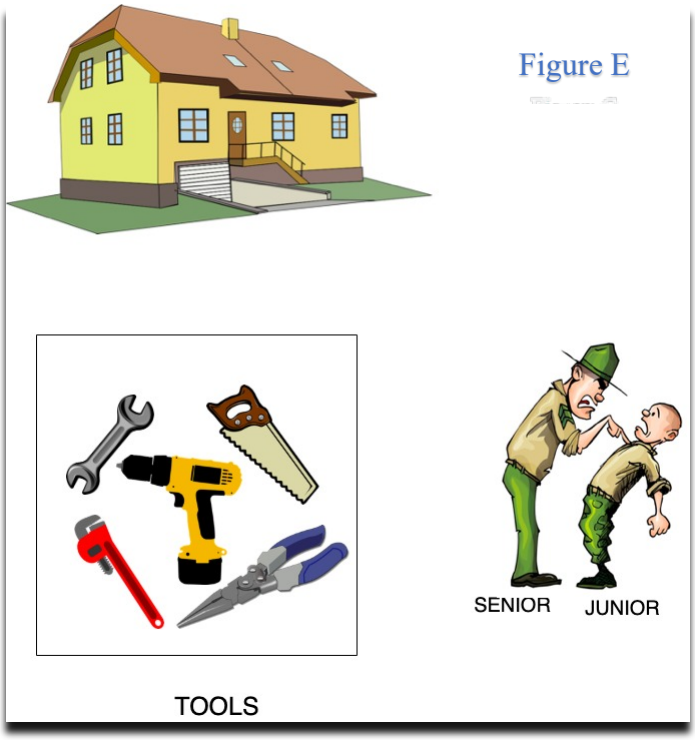
TOOLS

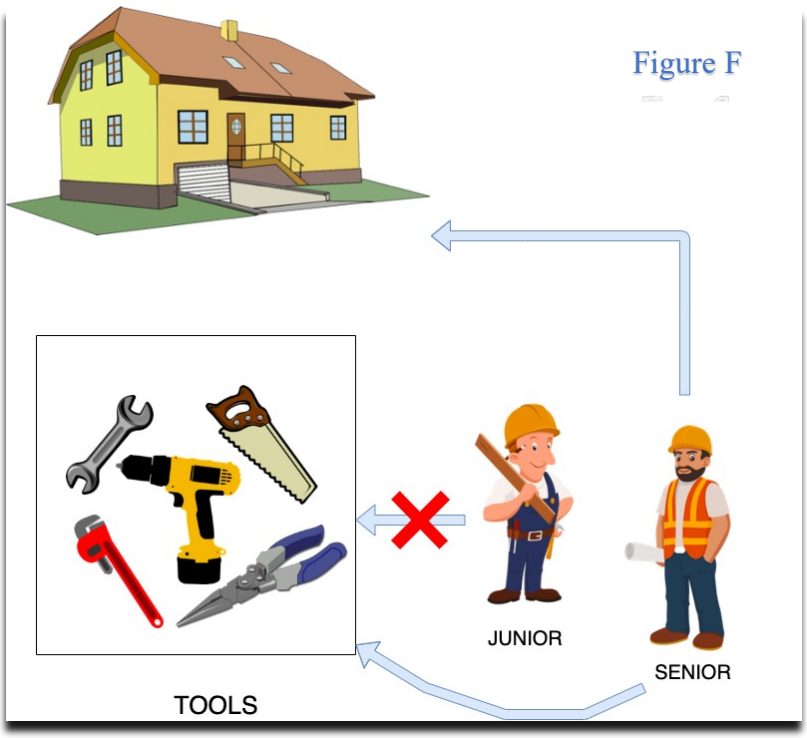
Figure D

JUNIOR

SENIOR

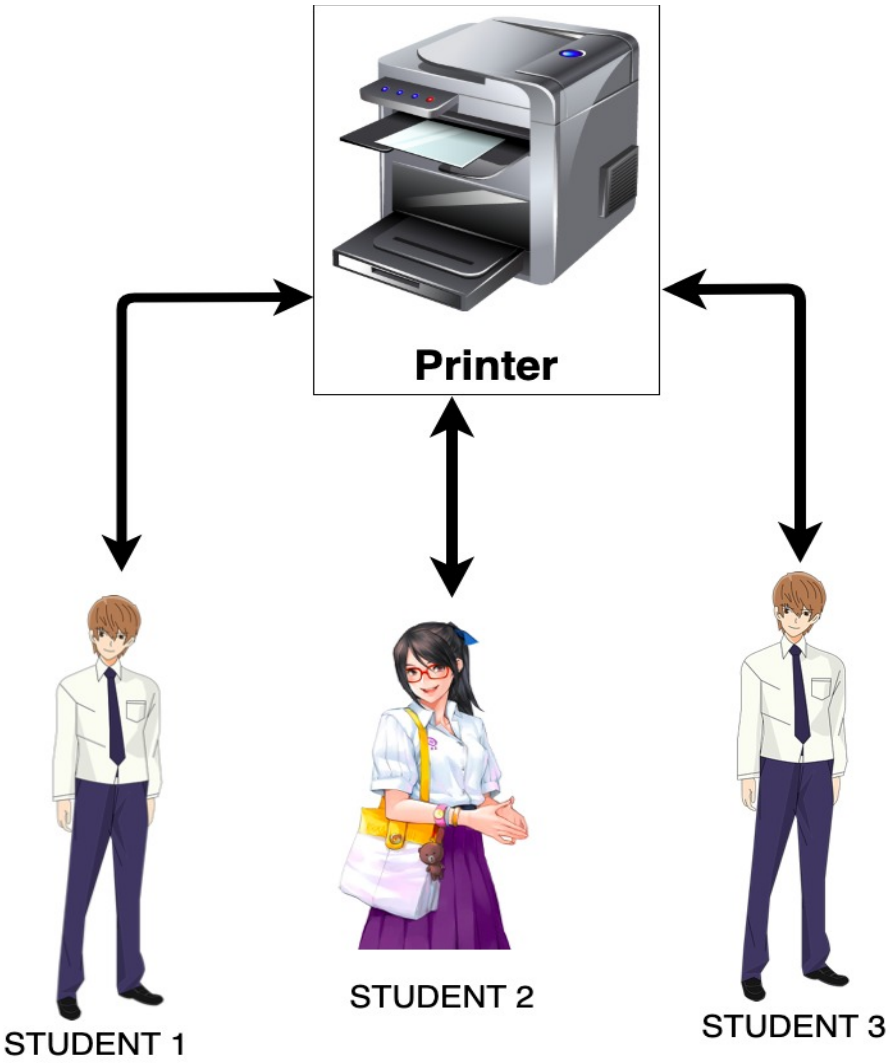
TOOLS





- In Figure A, we can see a Senior worker that is building a house, using tools.
- The Senior worker locks the tools for himself.
- In Figure B, we can see its 1pm and Senior worker go for the lunch leaving the tools behind.
- As the tools are free that means they are lock free.
- Now, a Junior worker comes working on 2nd shift sees that tools are not been used by anyone and aren't reserved by anyone.
- In Figure C, we can that Junior worker now locks the tools for himself and start building the house.
- In Figure D, we can see the Senior worker has return from lunch break and want to use the tools, but the tools are been used by Junior worker.
- This may cause deadlock.
- But, in Figure E we can see Senior worker arguing with Junior worker that, he's senior and working on building from start.
- He forces, Junior worker to stop using the tools and forces him to release the locks on them to use it for himself.
- In Figure F, we can see the Senior worker again acquiring the lock on tools and start building the house.
- The Junior worker can use the tools after restating the shift next day.
- This is how deadlock is prevented.
- In wound-wait if an older transaction requires certain resource that is been held by newer transaction then, it forces newer transaction to abort the process and release the lock on resource so that the older transaction can use it.

Cautious waiting:



- If all three students give print-out as the same time the printer will consider all the request and process them one at a time.
- If Student 1 gives, 20 pages print, Student 2 gives 5 pages print and Student 3 gives 8 pages print to the printer then Printer will print 5 pages of Student 1 then, it will process and print 1 page of Student 2 and finally will print 3 pages of Student 3.
- After this again it will print 5 pages of Student 1, 2 page of Student 2 and 2 pages of Student 3 and the process continuous.
- Hence, printer is always available to the student, it does not get blocked when one student sends a print request to the printer.
- If lots of print request are there on printer then it will just say that printer is busy currently, but still it will take the student print request and eventually will process.

Cautious waiting:

Here a transaction is allowed to wait if the resources are not blocked. But if resources are blocked, then the transaction restarts hoping that the resources will get unblocked after some time and then it can access them.