Mechanics Problem Solutions

Problem 1.

a.)

```
Declare/Initialize Semaphores:
Crowbar=3; Wrench=4; Jack=2;
Stand=3; TireTools=3;

P(Crowbar);
loop
  Remove Hubcap
  P(Jack)              Is order important
  P(Stand)                    here?
  Raise Car
  V(Jack)            Order Important Here?
  V(Crowbar)
  P(Wrench)
  Remove Lug Nuts
  V(Wrench)
  P(Crowbar)       Order Important
  P(tireTools)               Here?
  Change Tire
  V(TireTools)
  V(Crowbar)
  P(Wrench)
  Replace Lug Nuts
  V(Wrench)
  P(Jack)
  P(Crowbar)
  Lower Car
  V(Jack)
  V(Stand)
  Replace Hubcap
end loop
```

b.) The Mechanic

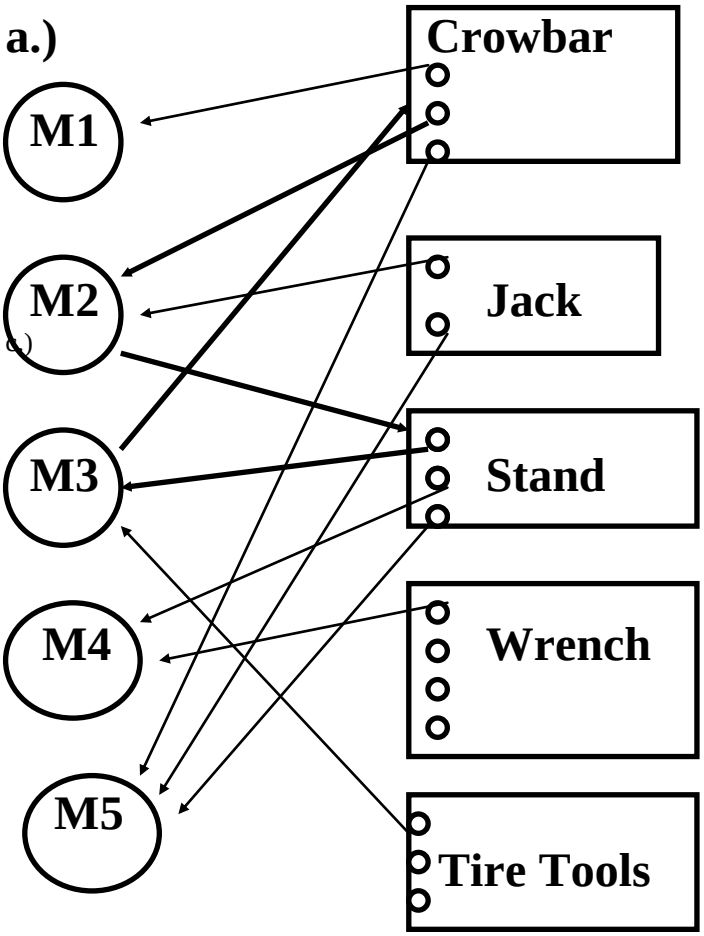
```
Process Mechanic
var TC:TireChange;
begin
  TC.StartMechanic
  loop
    RemoveHubcap()
    TC.Raise
    RaiseCar()
    TC.Raised
    RemoveLugNuts()
    TC.Change
    ChangeTire()
    TC.Changed
    ReplaceLugNuts()
    TC.Lower
    LowerCar()
    TC.Lowered
    ReplaceHubcap
  end loop
end
```

b.) The Monitor

```
Monitor TireChange
const
  Crowbar=3; Wrenches=4; Jacks=2;
  Stands=2; TireTools=3;

var
  AvailCrowBars, AvailJacks, AvailStands,
  AvailWrenches,  AvailTireTools:Integer;

  Bar, Jack, Stand, Wrench, TTool:Condition;

Procedure Entry StartMechanic;
{Need Crowbar to Start}
begin
 if AvailCrowbars=0 then
    Bar.wait;
 AvailCrowbars-=1;
end;

Procedure Entry Raise;
begin
  If AvailJacks=0 then
    Jack.wait;
  AvailJacks-=1;
  if AvailStands=0 then
    Stand.wait;
  AvailStands-=1;
end;

Procedure Entry Raised
begin
  AvailJacks+=1;
  Jack.signal;
  AvailCrowbars+=1;
  Bar.signal;
  If AvailWrenches=0 then
    Wrench.wait
  AvailWrenches-=1;
end;

...
begin {Init Code}
 AvailCrowBars:=CrowBar;
 AvailJacks:=Jacks;
 AvailStands:=Stands;
 ...
end;
```

Problem 2

a.)



c.)

b.) There is no deadlock, even though there is a cycle (highlighted).

M5 will release its stand when it finishes step 6, M@ will get the stand, do step 2 and release its crowbar

c) The answer is 6. Can't have deadlock with 5. BUT, this is dependant on ordering of requests.