Homework1

Qinglin Li, 5110309074

Problem 1

Every process is either red or white. Initially, all processes are white.
All processes send message with same color to its own
The protocol:

1. Process p_0 turns red and send message to others
2. $\forall p$, when p receives a message in red color for the first time, it record and send its local state to p_0. At the same time, p turns red and send message to others.
3. The collection C p_0 received is a consistant global state

Proof:
$\forall e_j, e_j \in C$, e_j must happen before p_j turned red.
$\forall e_i \rightarrow e_j$, if p_i is red, the message p_i sent to p_j must be red and e_j cannot happen before p_j turned red. So p_i must be white when e_i happened and $e_i \in C$.
So it's a consistant global state.

Problem 2

1. a combination of safety and liveness
2. a safety property
3. a combination of safety and liveness
4. no property at all
5. a safety property
6. a liveness property

Problem 3

1. (a) For every general x, ($x \in \{A, B\}$):
If inp_x ="ready" let m_{send} ="yes", else let m_{send} ="no"
Send m_{send} for 11 times

Let $m_{deliver}$ be the delivered message
Decide "attack" if $m_{deliver}$ ="yes" and m_{send} ="yes". Otherwise, decide "no"

Agreement:
Since at most 10 messages are lost, both generals should deliver at least one message for the other. If a general decide "attack", he must send and deliver the message "yes" and the other general must decide "attack" because he send and deliver "yes" too. In any other case, every general would send "no" or deliver "no" and then both decide "no".

Validity:
If both inputs are "not ready", both generals must send and deliver "no", so they both decide "not attack".
If both inputs are "ready", both generals must send and deliver "yes", so they both decide "attack".

Termination:
Because both generals send only 11 messages and eventually deliver a message, so they eventually decide.

(b) If inp ="ready" let m_{send} ="yes", else let m_{send} ="no"

For general A:
Keep sending m_{send_A} until receiving a message from general B

For general B:
Send m_{send_B} whenever receiveing a message from general A

Let $m_{deliver}$ be the delivered message
Decide "attack" if $m_{deliver} = m_{send}$ ="yes". Decide "no" if inp="not ready"

Agreement:
Since finite messages are lost, both generals should eventually deliver at least one message for the other. If a general decide "attack", he must send and deliver the message "yes" and the other general must

decide "attack" because he send and deliver "yes" too. In any other case, every general would send "no" or deliver "no" and then both decide "no".

Validity:
If both inputs are "not ready", both generals must decide "not attack". If both inputs are "ready", both generals must send "yes" and eventually deliver "yes", so they both decide "attack".

Termination:
Because eventually one message should be delivered, they eventually decide.

2. (a) Because both generals send 11 messages, they halt.

   (b) Since finite messages are lost, general B should have received some messages from general A and general A should eventually have received a message from general B. So general A halts.
   Because general B receives no more messages than general A have sent, general B halts.

Problem 4

Property X: For any distinct process i and j, if i sent m before j sent m', no process would deliver m after m'.

FIFO ensures

$$send_i(m) \rightarrow send_i(m') \Rightarrow send_k(m) \rightarrow send_k(m')$$

Property X ensures

$$\forall i \neq j, send_i(m) \rightarrow send_j(m') \Rightarrow send_k(m) \rightarrow send_k(m')$$

So FIFO + Property X ensures

$$\forall i, j, send_i(m) \rightarrow send_j(m') \Rightarrow send_k(m) \rightarrow send_k(m')$$

And that's casual property

Problem 5

Every processor p maintains a vector clock.
Let m_i be the last message p_i delivered and $D[i] = TS(m_i)[i]$
When a message m is delivered by p_i, $VC(e_i) = max(VC[i], TS(m))$
When p_i is broadcasting a message, $VC(E_i)[i] = VC[i] + 1$
In other situations, p_i just keep VC
Deliver m from p_j as soon as both of the following conditions are satisfied:

1. $D[j] = TS(m)[j] - 1$
2. $D[k] \geq TS(m)[k], \forall k \neq j$