1. (a) True.
   (b) False.
   (c) False.
   (d) True.
   (e) True.
   (f) True.
2. (a) False.
   (b) True.
   (c) True.
   (d) True.
   (e) False.
   (f) False.
3. Following Euclids algorithm:
   (a) Yes.

$$\begin{aligned} & GCD(10505, 1274) \\ = & GCD(1274, 313) \\ = & GCD(313, 22) \\ = & GCD(22, 5) \\ = & GCD(5, 2) \\ = & GCD(2, 1) \\ = & GCD(1, 1) \\ = & 1 \end{aligned}$$

   (b) No.

$$\begin{aligned} & GCD(8024, 7289) \\ = & GCD(7289, 740) \\ = & GCD(740, 629) \\ = & GCD(629, 111) \\ = & GCD(111, 74) \\ = & GCD(74, 37) \\ = & 37. \end{aligned}$$

4.

5. Let $\phi = (x \vee y) \wedge (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge (\bar{x} \vee \bar{y})$. The following truth-table shows that ϕ is not satisfiable.

x	y	$(x \vee y)$	$(x \vee \bar{y})$	$(\bar{x} \vee y)$	$(\bar{x} \vee \bar{y})$	ϕ
0	0	0	1	1	1	0
0	1	1	0	1	1	0
1	0	1	1	0	1	0
1	1	1	1	1	0	0

6. Let $\langle L_i, M_i \rangle$ be a polytime language and decider, such that $L_i = \{w \mid M_i\langle w \rangle \text{ accepts}\}$, and M_i always halts in polynomial time.

   Let $L_\cup = L_i \cup L_j$, $L_\circ = L_i \circ L_j$, $\bar{L_i} = \{w \in \Sigma^* \mid w \notin L_i\}$ be languages. To show that $L_\cup, L_\circ, \bar{L_i} \in \text{P}$, we construct respective polynomial time deciders $M_\cup, M_\circ, \bar{M_i}$.

   $M_\cup =$ "On input w:

   1. Run $M_i\langle w \rangle$. If M_i accepts, *accept*.
   2. Run $M_j\langle w \rangle$. If M_j accepts, *accept*.
   3. If neither $M_i\langle w \rangle$ nor $M_j\langle w \rangle$ accepted, *reject*."

   $M_\circ =$ "On input w:

   1. For each position $k = 0$ to $|w|$, divide w into substrings $w = w_1w_2$, where w_1 is the first k symbols in w.
   2. Run $M_i\langle w_1 \rangle$ and $M_j\langle w_2 \rangle$. If both accept, *accept*.
   3. If no k exists such that $M_i\langle w_1 \rangle$ and $M_j\langle w_2 \rangle$ both *accept*, *reject*."

   $\bar{M_i} =$ "On input w:

   1. Run M_i on w. If M_i accepts *reject*. If M_i rejects, *accept*.

7. Let $\langle L_i, N_i \rangle$ be a language and non-deterministic Turing machine, such that N_i decides L_i in polynomial time.

   Let $L_\cup = L_i \cup L_j$, $L_\circ = L_i \circ L_j$ be languages. To show that $L_\cup, L_\circ \in \text{P}$, we construct respective polynomial time non-deterministic deciders.

   $N_\cup =$ "On input w:

   1. Non-deterministically branch to simulate both $N_i\langle w \rangle$ and $N_j\langle w \rangle$.
   2. If either branch accepts, *accept*, else *reject*."

   $N_\circ =$ "On input w:

1. Non-determinsitically branch for each position $k = 0$ to $|w|$, with each branch dividing w into substrings $w = w_1 w_2$, where w_1 is the first k symbols in w.
    2. For each branch, run $N_i\langle w_1\rangle$ and $N_j\langle w_2\rangle$. If both accept, *accept*.
3. If no branch accepts, *reject*."

8. To show that CONNECTED $\in P$, we need to show that CONNECTED $\in$ TIME(n^k), where k is some constant. As M decides CONNECTED, we only need to prove that M runs in $O(n^k)$.

   - In step 1, a node is selected and marked. This is done in constant time.
   - In stages 2 and 3, each node in G is scanned and marked if it is a neighbour of a marked node. This is repeated until no new nodes are marked. Assuming we do not visit already marked nodes, this has a worst-case run-time of $n\sum_{i=1}^{n-1} i = \frac{n^2(n-1)}{2}$ steps.
   - In stage 4, each node of G is scanned to determine if it is marked. This takes n steps.

   Combining steps 2,3 and 4, we have a run time of $\frac{n^2(n-1)}{2} + n = \frac{n^3+n^2+2n}{2} = O(n^3)$. This is clearly in P, and the proof is complete.

9. To show that TRIANGLE $\in$ P, we need to show that TRIANGLE $\in$ TIME(n^k), where k is some constant. We define Turing machine M, which operates as follows.

   M = "On input $\langle G\rangle$:

   1. For each edge u, v in G, do
       2. For each vertex w, do
           3. If v, w is an edge and w, u is an edge, *accept*
   4. If no edge, vertex combination has accepted, *reject*.

   M runs in $O(|V||E|) = O(n^3)$. This is clearly in P, and the proof is complete."

10.

11.

12.

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.

23.

24.

25.

26. Assume an arbitrary upper configuration of size $k > 3$. Ignoring the lower configuration, there must then be at least two overlapping windows consisting of adjacent tape symbols abc, where $cell[i, j-1] = a$, $cell[i, j] = b$, $cell[i, j+1] = c$. We note that any cell adjacent to the tape head may be updated in the lower configuration. As both windows contain only tape symbols, each window cell is potentially adjacent to the head (with the exception of the case in which either window is at the beginning or end of the tape), meaning neither window can verify any corresponding cells in the lower configuration. As $cell[i, j]$ is not covered by any other windows, $cell[i+1, j]$ will never be verified, and the proof is complete.