Assignment 5 Problem 2

a) In the forward direction, suppose e is in G but it is not in the MST, then consider the way we construct the MST in Kruskal's Algorithm, we go over each edges, thus this includes e. The only reason why it is not in the MST is that the two endpoints of e (say u and v) must in a same connected component. In this case, consider the uv-path in the current connected component. It is exist since the component is connected. And since we go over edges from light weight to heavy weight, all edges on such path must be less than the weight of e. Therefore, by adding e to such uv-path form a cycle, and in this cycle, e is the heaviest edge.

   In the opposite direction, if there exists a cycle on which e is the maximum weight edge. Let two endpoints of e be u and v. When we applying Kruskal's Algorithm, e is the last edge that we go over in this cycle. And because u and v are both in a same connected component, so e=uv will not be selected as an edge of the MST.

   Therefore, an edge e is not in the MST if and only if there exists a cycle on which e is the maximum weight edge.

b) Since the vertex set does not change, the MST of $T \cup \{e\}$ is also a spanning tree of $G \cup \{e\}$. The only thing needs to justify is it is the minimum one. Since tree has the fixed number of edges, if e is in MST, then one and only one edge is removed from T. Otherwise, new MST remains the same as T. Consider the process of Kruskal's algorithm, when we go over edges, edges are from light to heavy. So relative orders of edges in $T \cup \{e\}$ is exactly the same as the relative orders of them in $G \cup \{e\}$. So if the e is not in the MST of $T \cup \{e\}$, then which means when we go over edges in $G \cup \{e\}$, e is still not chosen since both endpoints are in the same component. If e is in the MST of $T \cup \{e\}$, since we will form a cycle by adding e to T, then from part 1, we need to remove an heaviest edge from that cycle to remain the structure of tree. Therefore, we obtain an lighter spanning tree. Similarly, the relative orders of edges in $T \cup \{e\}$ is exactly the same as the relative orders of them in $G \cup \{e\}$. When we traversing the edges in $G \cup \{e\}$ we will make the same decision as the edges in $T \cup \{e\}$.

c) The algorithm is as follows.

Adding the edge e into T, then let one of its endpoints as a source vertex, run BFS to find a cycle, note that this cycle must be return to the source vertex due to the property of tree. Then find the edge with the maximum weight in such cycle, remove such edge.

After removing the edge, we get the MST of $T \cup \{e\}$, from the conclusion we got in part b), this MST is also the MST of $G \cup \{e\}$.

The runtime of this algorithm is runtime of BFS plus runtime of finding and removing edge, which is O(n) in total. (Note that since we are working on a tree, m is in $\Theta(n)$).