(1) **Take the following flow-network and run the Ford-Fulkerson algorithm on it. Once you are done, write the following items in your response:**

- **Write the original / starting residual graph Gf for this network BEFORE execution of Ford-Fulkerson begins.**

	s	**u**	**v**	**t**
s	0	10	5	0
u	0	0	15	5
v	0	0	0	10
t	0	0	0	0

- **Write out the final / ending residual Graph Gf after Ford-Fulkerson completes. This should be an Adjacency Matrix and make sure to include backflow edges.**

	s	**u**	**v**	**t**
s	0	0	0	0
u	10	0	10	0
v	5	5	0	0
y	0	5	10	0

- **What is the final maximum flow f for this Graph?**

  The final maximum flow is 15 as can be seen from the sum of the “s” column in the final Residual Graph Gf in the red border

(2) Consider the flow network. What is the maximum number of iterations that Ford-Fulkerson could make. Explain how this maximum could be reached and whit it is a major issue.

As stated in the problem, the first path that Ford-Fulkerson will find from the start node to the end node is s→ u→ v → t. It will push the initial 1,000,000 from edge (s,u), then push only one unit on the edge (u,v) as its capacity is only 1. It will then send that one unit through edge (v,t) and change its capacity and backedges capacity accordingly. Now, in the worst case scenario, the DFS algorithm accesses the s→ v→ u→ t path next, where it will see that the backedge from (v,u) has one unit of open flow to it, and it will go there and reverse the backedge and normal edge capacity. The algorithm will go back and forth like this until the (u,t) and (v,t) edges are full, one at a time. This means that the maximum possible iterations that the algorithm can make is **2,000,000**.

(3) Suppose we want to take a flow-network as input and return a set of at most |E| augmenting paths that produce the maximum flow on that network. Describe an algorithm to do this.

First, we need to find the maximum flow using the Ford-Fulkerson algorithm. Now that we have the maximum flow, we have the necessary flow values for each edge that contribute to this maximum flow number using our adjacency matrix. We then take the paths in and try to send the maximum amount of the maximum flow as possible on that path. The flow value described previously is only as high as the edge with the lowest max-flow capacity. Once the flow is pushed along every path possible, the minimum edge is eliminated from the group of possible edges and won't be included in other paths caluculated. Since we've already moved all of its max-flow, there is no more capacity left. So, since each iteration of the algorithm takes an edge out of the possible pool, the set at the end can contain at most |E| augmenting paths as, in the worst case scenario, all edges will be removed from the pool.