(1) 1 (a, b)

In this problem, we were tasked with finding an order of courses. We had to implement two approach such as DFS and BFS.

The DFS used a recursive function to explore the courses and use a stack to keep track of order. It gives a valid order if there is one, if none. then it gives Impossible.

The BFS approach use a queue to explore the courses. It calculate in degree and iteratively remove courses with indegree of zero while update the indegree. If the valid order exists, it gives it otherwise prints Impossible.

(2)

In the function g(a, pre), it takes a as the number of courses and pre as a prerequisite pairs as input. It performs BFS to find course sequence. Then returns the course sequence or Impossible. In the function r_input(f2);, it reads input and extracts the number of courses I and prerequisite pair l2. In the function, w_out. it writes output to the given file f2 and writes course sequence or Impossible based on s. The code basically determines a valid course sequence based on prerequisites and writes the result

(3)

The program reads input, The DFS is used on the original graph to fill the stack. Using the vertex finishing times from the DFS, traverse, the stack is filled. After getting the stack with finishing times, the code find the strongly connected components. It then performs another traversal on the reversed graph using stack. Each ~~vertices~~ DFS call identifies strongly connected component Finally it prints the strongly connected components.