Exercise 2

Claim

The *DFS_Listing* Algorithm is correct and enumerates all frequent itemsets with polynomial delay.

Proof

Correctness: The itemsets are printed in step 1: "print F". At first the emptyset gets printed what is trivially correct. Then all infrequent items from D are removed and the algorithm gets called with D_i and $F \cup \{i\}$. Where D_i is the set of all transactions that contain i restricted to those that are strictly greater then i according to the linear order defined on the items. In the next step $F \cup \{i\}$ gets printed. That are all frequent items in D (frequent itemsets of size 1) since $F = \emptyset$. In general each step takes a frequent k-itemset F and appends a frequent item i and get $F \cup \{i\}$. Since F is a frequent itemset and D_i contains itemsets that are locally frequent without the item i the union of that is frequent in the complete transaction set. This is the general idea of that algorithm: Find all itemsets that are frequent containing the first item according to the linear order and then find those that are frequent containing the second item and not the first and then build then join those to find all frequent itemsets int the transaction table. $\Rightarrow$ the algorithm prints only frequent itemsets.
Now assume there exists an itemset F' that is frequent but does not get printed by the algorithm. That means there is no frequent F'' and frequent i such that $DFS\_Listing(D_i, F'' \cup \{i\})$ is called. That means F'' gets not printed. If F' is a k-itemset then F'' is a k-1-itemset. When we continue that we have a frequent 1-itemset that does not get printed. But that is a contradiction because every frequent 1-itemset j gets printed via $F = \emptyset \cup \{j\}$. $\Rightarrow$ every frequent itemset gets printed. Now assume G is not frequent but gets printed. Then there needs to be a frequent itemset F and a frequent item i such that $G = F \cup \{i\}$ but thats not possible because F is locally frequent in D without i and i is frequent then the union has to be frequent too.
$\Rightarrow$ the algorithm is correct.
Complexity: The print statement is the first one that gets called. So we need to proof that step 2-5 have polynomial time complexity. Lets say $|D| = n$. Removing all infrequent items from a set of size at most n can be done polynomially in n (quadratic). Defining a linear order is a linear operation. Depending on the implementation one might want to order that. But we know sorting algorithms with polynomial runtime. In the next step the alogorithm loops over all sets in D_i. In each iteration the set D_i is defined by picking all itemsets in D that contain elements greater then i this can easily be done in linear time when we sort (in poly time) D beforehand. Nevertheless this can be done in polynomial time. After that *DFS_Listing* gets called again and the next frequent set gets printed. $\Rightarrow$ the algorithm has polynomial delay.
Q.E.D.