Exercise 3

Let $\mathcal{A}$ be an algorithm that has a transaction database D over an itemset I with threshold t as input parameters and outputs all frequent itemsets $\mathcal{F}$ where $\mathcal{A}$ is deterministic and only has access to D via the query "Is itemset X frequent?".
Claim: $\mathcal{A}$ needs evaluate that query at least for $|Bd(\mathcal{F})|$ number of itemsets.
Proof:
The border $Bd(\mathcal{F})$ is defined by the union of the positive and the negative border.

$$Bd(\mathcal{F}) = Bd^{+}(\mathcal{F}) \cup Bd^{-}(\mathcal{F})$$

We can proof the claim by showing that every set in the negative resp. positive border needs to be evaluated by every algorithm of the form described above. The negative border consists of all sets that are minimal infrequent itemsets. If $\mathcal{A}$ knows that an itemset X is frequent we we can conclude that every proper subset of X is frequent. On the other hand side if $\mathcal{A}$ evaluates X as infrequent we know that every set that contains X if infrequent. Let Y be a minimal infrequent itemset. There is no way of evaluating Y as infrequent by finding an infrequent proper subset of Y because every proper subset of Y is frequent. Hence $\mathcal{A}$ has to evaluate "is Y frequent?". The positive border on the other hand side consist of all maximal frequent itemsets. Let Z be a maximal frequent intemset. There is no way of identifying Z as frequent by finding a frequent proper superset of Z since all proper supersets of Z are infrequent. Hence $\mathcal{A}$ needs to evaluate "Is Z frequent?". That means $\mathcal{A}$ needs to evaluate the query at least $|Bd(\mathcal{F})|$ times. q.e.d.

Exercise 4

We just showed that the Apriori algorithm has the evaluate the query at least $|Bd(\mathcal{F})|$ times. To make this more precise we notice that the algorithm evaluates the query only in step 4 where all infrequent candidates are removed. So we can say that the query needs to be evaluated for every candidate that gets generated by the **CANDIDATEGENERATION** algorithm. (This is based on the assumption that candidate counting works by evaluating a query for every set in the set of candidates. There might be implementations where this is solved more efficient). The candidate-generation algorithm is based on the fact that every k-1-itemsubset of a frequent k-itemset is also frequent. Let $\mathcal{C}$ be the total number of candidates generated during the total execution of the algorithm. Then we can state "Number of times the query gets evaluated" = $|\mathcal{C}|$. We can describe C by:

$$c \in C \iff$$

$$\exists X, Y \in \mathcal{F} : size(X) = size(Y) = size(c) - 1,$$
$$X, Y \text{ differ only in the last element } (x_k,\, y_k)\ ,$$

c equals the first k-1 elements concatinated with x_k and y_k ,
all size(c)-1 subsets if c are frequent

The above is basically the characterization defined in step 34 in the candidate generation algorithm. Where size(c)=k, when c is a k-itemset.