R-6.1 **Hint:** Review the definitions of separate chaining and load factors.

R-6.2 **Hint:** Review the definitions of open addressing and load factors.

R-6.3

R-6.4

R-6.5

R-6.6

R-6.7

R-9.2 **Hint:** Review the case for $d = 2$ and generalize this approach.

Solution: Preform the stable bucket sort on m, l, and then k. In general, perform the stable bucket sort on k_d down to k_1.

C-6.7 **Hint:** Think of how you might generalize the polynomial hash function.

Oppgave 2

Solution: [of Exercise **??**] Hash tables are not really made for that. In particular, they don't contain or represent *order* information. In a way, a good *hashing function* should best be completely unordered ....

Anyway, since that's the case, there's no other way to determine the maximum etc than just going throught the whole table and find out. So the complexity is $O(n)$. Note there is no distinction between best/worst/average case, one needs to search through all of it, as said.