CS917: Seminar 3

1 Exercise 1

Many of the problems you will see in the real world don't have an obvious, clear cut, solution like the toy problems we look at in lectures do. Often, you have to really sit down and think about the problem at hand, and apply the knowledge you know and are comfortable with. Often it doesn't have to be an optimal solution, it can simply be good enough for the task at hand. With that in mind, have a look at the following problem and see what the best solution you can come up with is (and don't worry if it takes a bit of time, it's not easy but you will get there with some effort!):
You are given two eggs, and access to a 100-storey building (both eggs are identical). The aim is to find out the highest floor from which an egg will not break when dropped out of a window from that floor. If an egg is dropped and does not break, it is undamaged and can be dropped again. However, once an egg is broken, that's it for that egg. If an egg breaks when dropped from floor n, then it would also have broken from any floor above that. If an egg survives a fall, then it will survive any fall shorter than that.

- What strategy should you adopt to minimize the number egg drops it takes to find the solution?.
- What is the worst case for the number of drops it will take?
- How does this change if I give you a third egg?
- What about if we have a bigger building, with more stories?
- Can we generalise for n eggs, and a k story building?

Try out your solutions in code and print out the number of drops you are performing!

2 Exercise 2

For this task, much like you may have to in real life, you have to design your own algorithm to solve a given problem. Two people, Alice and Brian, are interested in the results of the football final:

- Each person knows all of the n teams in the league
- Alice knows the two teams who play in the final, but not who won
- Brian knows which team won, but not who they were playing against

Assuming they can agree a protocol beforehand:

- Design an algorithm to communicate the winner of the event
- Calculate how much information they would need to exchange in order to do this
- How many messages need to be sent?
- How many bits/bytes of information need to be sent?
- Can you think of a more efficient way to do this (i.e. fewer bits/bytes)?
- What is the algorithmic complexity of each of your solutions?

Try this question out by thinking of a theoretical solution at first, then have a go at adding methods to the base example (or write your own if you prefer the data stored in a different format).

Hint: The complexity of the optimal solution for this is very low, and also fairly uncommon.