# Project 1 Analysis

1) If you used different arrays/structures/classes to represent the different types of sets, would it be possible to have overloaded methods or operations that would provide the correct functionality regardless of whether or not you were using sets, multi sets, or fuzzy sets? Why or why not? (You do not need to code this, just answer the question).

When it comes to some set operations on plain sets vs. fuzzy sets, we would have a problem since the nature of a set operation on a fuzzy set is entirely different when operated on a normal set. For instance, computing the complement of a normal set means you take the opposite truth value of the bit string representing what is included in the set. So, everything not included in the set is included in the complement. When it comes to taking the complement of a fuzzy set, since we are dealing with members and degrees of membership, we look at the degree of membership, which is a number between 0 and 1. The complement will be the difference of 1 and the degree of membership. An overloaded method for the complement set operation would have to have two entirely different implementations to handle the different types of sets.

2) Would it be possible to use the same data set/structure/class to store sets, multi sets, and fuzzy sets? Why or why not?

It would be possible to have one class that holds fields for each of the different types of sets. For instance, you could have a class called Sets and then fields: setA and setB (normal sets), multiSetA and multiSetB (multisets), and fuzzySetA and fuzzySetB (fuzzysets). Set specific set operation methods could be implemented where applicable.

3) How easy or difficult is it to determine the type of set that you need to use based on the users query? Why?

4) Is it possible to store the data from one type of set (plain sets, multi sets, fuzzy sets) in another type? Would you need to lose data in order to do so? Why?

5) Discuss what implications your answers to questions 1 – 4 have for someone trying to code an interface which would allow users to type in natural language queries.

6) Discuss what implications your answers to questions 1 – 5 have for someone trying to code an interface which would allow users to access arbitrary types of data using natural language.

In a very fundamental sense, the answers to questions 5 and 6 have not only shaped how the internet has evolved, but has equally shaped how people interact with the internet. One of the definitions of a ‘digital native’ is someone who is comfortable and adept at finding and interpreting answers to questions in an online world.

7) Considering your answers to 5 and 6 above, do you in particular, and programmers in general, have a good feel as to how people use data in today’s world? Why do you say that? There is no wrong answer to this, I just want you to think deeply about how you view data vs how a typically internet user mighty view data.