1. Create a list of all possible sets from the given list

3a. Whenever there is a arithmetic expression with a real and an int convert the int into a real and then solve the expression

3b. overload the + symbol so that whenever it has a real and an int you add them and return the answer as a real number

3c. You overload the + symbol so that whenever it has a real and an int you convert the int into a real and then return the resulting answer as a real

3d. both real and int could be a subtype of the type number.

5a. guess: ‘a -> int , answer: ‘a -> int

5b. guess: ‘a,’b -> int, answer: ‘a \* ‘b -> int

5c. guess: ‘a -> ‘a, answer: ‘a -> ‘a

5d. guess: ‘a \* ‘b -> ‘a, answer: ‘a \* ‘b -> ‘a

5e. guess: ‘a -> int, answer: (int -> ‘a) -> ‘a

5f. guess: (‘a -> ‘a) \* ‘b -> ‘a, answer: (‘a -> ‘b) \* ‘a -> ‘b

5g. guess: (‘a \* ‘b -> ‘c) \* ‘a \* ‘b -> ‘d, answer: (‘a \* ‘b -> ‘c) \* ‘a \* ‘b -> ‘c

5h. guess: ((‘a -> ‘b) -> ‘c) \* (‘a -> ‘b) \* ‘a -> ‘b, answer: (‘a -> ‘b) \* (‘c -> ‘a) \* ‘c -> ‘b

5i. guess: ((‘a -> ‘b) -> ‘c) \* ‘b -> ‘c, answer: (‘a -> ‘a) \* a -> ‘a

6a fun test c = c ^ " is a character."; The major benefits would be that you could concatenate a char to a string and there wouldn’t need to be any need to explicitly convert a char to a string. However ML would lose some of its type safety from doing this.

6b. fun test x = Math.sqrt x; The major benefits would be that you could do any arithmetic operations on an int that you could do on a real. However because real numbers are not an equality type then neither would integers.