1) What are the criteria for a recursive solution?

- There must be a way of reducing the problem to smaller repetitive elements
- There must be a way of identifying, recognizing and dealing with the end point of the solution
- There must be a way of achieving a larger result from the component elements

2) Think of another everyday example. NOT mentioned on this assignment or during the lesson!

When two devices calling each other are placed nearby and they begin to repeat the same sounds and create an awful noise . One device's microphone listens to the other's speaker and repeats the noise it originally made.

3) Give two specific problems associated with recursion, and two reasons for using recursion

Cons:

- Space: since function calls require space for parameters and local variables, and for an indication of where to return to. A loop on the other hand, only needs to return to the beginning and recreate variables. This space is typically allocated on the stack, and released automatically when the function returns, but there will still be an amount of space required which grows proportionally to the number of nested calls in the function.
- Time: since operations for calling a funcion - allocating and releasing local memory, copying values into local memory for parameters, branching to/returning from the function. Everything adds up and hinders performance.

Pros:

- They are cleaner, simple, shorter, and as such, simpler to understand than non-recursive solutions
- The program directly reflects the solution strategy

4) What does an iterative solution mean?

Iterative solutions are solutions which use a loop that they iterate through to solve a problem. For example, an iterative solution to adding all the values in an array could be to store an index number, then access each value in the array through that index, and increase the value with each loop/iteration until each value in the array has been checked.