Assignment 4

Task 1

GenerateOdd works by first checking whether you are past the endpoint of the list, and then if you aren't checks whether the current number is an odd number and add it to a list if it is, and set the tail to a recursive call to current number + 2. If the number is even then just recursively call the function with the current number incremented by 1.

Task 2

The function product works by pattern matching the input and if it is a list it multiplies the head with a recursive call on the tail. When the end of the list is reached it multiples with 1.

Task 3

The first 3 digits are 100. The code works by putting the execution of two different functions inside different thread closures. The benefit of running on two separate threads is that the functions can run in parallel, which increases performance of the code.

Task 4

Using the lazy annotation on the GenerateOdd function means the consumer (product function) has to wait for the producer (GenerateOdd) to produce the next odd number every time the consumer needs a new odd number. This means the execution speed is going to be slower, and in our case it's useless because we know we are going to need all the numbers in the list, so we won't save any resources.

Task 5

HammerFactory works by First waiting 1 second, thern creating a random number between 1 and 100 to determine whether to return a list where the head is either a working or defect hammer. The tail is a recursive call to HammerFactory.

HammerConsumer works by Taking an input hammer stream and a number N which determines how many hammers to consume. If N is greater than 0 it pattern matches the hammer steam to check whether it is a list with head and a tail, and if it is working it adds 1 to the number of working hammers and adds a recursive call to HammerConsumer with the tail as input stream and subtracting 1 from N. If the hammer is defect it adds 0 instead of 1, and if N <= 0 it adds 0 and ends the recursion.

BoundedBuffer works by creating a standard list by adding elements from the lazy input stream, and when the desired buffer size is reached, it appends the lazy input stream to the end of the list.