**CS2030S RECITATION 08**

**Q1 (supposed to be 100k to 200k, not 10k to 20k)**

1. In ArrayList, all the primes between 10000 to 20000 are evaluated eagerly so even if we only wanted to find the 5th prime number, the subsequent prime numbers are also evaluated ie 5th, 6th, .., ith where 10000 <= ith prime <= 20000 and 20000 < (i+1)th prime

In LazyList, the programme only evaluates up till the 5th Prime between 10000 and 20000, and subsequent evaluations are not made thus lesser isPrime calls ok-ish

* Arraylist: has to loop through the entire range, LL: only loops until the kth term is found, the remaining are still frozen.
* LLmake is synthetic sugar based on the #define code (based on C)
* Key difference lies in intRange (for LL) vs intRangeArray (for ArrayList)
* Memoization only useful when you need to thaw the same thing multiple times. ie thaw(the same head twice or more etc)

1. For LL: depends on k and interval (because more primes in smaller numbers. Number of primes get more sparse with larger numbers)

For AL: depends on interval correct

1. Not possible. AL iterates through all the values in the interval and tests if each value isPrime while LL will stop once the kth isPrime returns true to return the kth prime number.

Correct. At most only as much work as the AL version. If the range is 2 (inclusive) to 8 (exclusive) and we are asked to get the 4th prime, then the number of calls on LL = AL

**Q2**

1. See code
2. I don’t know how to run it…
3. See code
4. Why cannot infer?
5. Can’t run it if d is wrong…

**Q3 [LINKED TO Q2 PART D AND E…]. use using LazyList rather than ArrayList to avoid doing the 6 deep for loops.**