Data Structures: Homework on Queues

October 3, 2023

In this assignment you will use queues to implement an algorithm to calculate all primes, in order, up to a number n. Here is how that algorithm will work:

1) Initialize a queue called `numbers` filled will all of the numbers from 2 (since 1 is technically not prime) up to n. Initialize another empty queue called `primes`.

2) Remove the smallest element in `numbers` (the first element in the queue), call this `p`, and add it to the end of `primes`.

3) Remove all elements of `numbers` that are divisible by `p`. To do this, remove elements in the front of `numbers` one by one and add them to the end of `numbers` only if `p` does not divide them. If `numbers` is not empty, go back to step 2.

4) Print the elements in `primes`.

This code should be implemented in a method called `primesTo(int n)` in a class called `PrimeCalculator`. If `primesTo` is given a number less than 2 then it should raise an exception. `PrimeCalculator` should have a `main` method to test your code. The class `ArrayQueue` from the textbook is provided to you. All other implementation details are up to you. Please zip all source files and submit on Brightspace.

Example input:

```
new PrimeCalculator().primesTo(20);
new PrimeCalculator().primesTo(2);
new PrimeCalculator().primesTo(0);
```

Returns:

```
Printing primes up to 20:
2, 3, 5, 7, 11, 13, 17, 19.

Printing primes up to 2:
2.

Error: Input must be a number greater than or equal to 2.
```