```
import java.util.*;

/**
 * TicketCounter demonstrates the use of a queue for simulating a line of customers.
 */
public class TicketCounter
{
   private final static int PROCESS = 120;
   private final static int MAX_CASHIERS = 10;
   private final static int NUM_CUSTOMERS = 100;

   public static void main(String[] args)
   {
      Customer customer;
      Queue<Customer> customerQueue = new LinkedList<Customer>();
      int[] cashierTime = new int[MAX_CASHIERS];
      int totalTime, averageTime, departs, start;

      // run the simulation for various number of cashiers
      for (int cashiers = 0; cashiers < MAX_CASHIERS; cashiers++)
      {
         // set each cashiers time to zero initially
         for (int count = 0; count < cashiers; count++)
            cashierTime[count] = 0;

         // load customer queue
         for (int count = 1; count <= NUM_CUSTOMERS; count++)
            customerQueue.add(new Customer(count * 15));

         totalTime = 0;

         // process all customers in the queue
         while (!(customerQueue.isEmpty()))
         {
            for (int count = 0; count <= cashiers; count++)
            {
               if (!(customerQueue.isEmpty()))
               {
                  customer = customerQueue.remove();
                  if (customer.getArrivalTime() > cashierTime[count])
                                    start = customer.getArrivalTime();
                  else
                     start = cashierTime[count];
                                 departs = start + PROCESS;
                              customer.setDepartureTime(departs);
                  cashierTime[count] = departs;
                  totalTime += customer.totalTime();
               }
            }
         }

         // output results for this simulation
         averageTime = totalTime / NUM_CUSTOMERS;
```

```
            System.out.println("Number of cashiers: " + (cashiers + 1));
            System.out.println("Average time: " + averageTime + "\n");
        }
    }
}
```

```
----------------------------------------
----------------------------------------
----------------------------------------
----------------------------------------
import java.util.*;

/**
 * Codes demonstrates the use of queues to encrypt and decrypt messages.
 */
public class Codes
{
   /**
    * Encode and decode a message using a key of values stored in
    * a queue.
    */
   public static void main(String[] args)
   {
      int[] key = {5, 12, -3, 8, -9, 4, 10};
      Integer keyValue;
      String encoded = "", decoded = "";
      String message = "You can post anything " +
               "that you like here.";
      Queue<Integer> encodingQueue = new LinkedList<Integer>();
            Queue<Integer> decodingQueue = new LinkedList<Integer>();

      // load key queues
      for (int scan = 0; scan < key.length; scan++)
      {
         encodingQueue.add(key[scan]);
         decodingQueue.add(key[scan]);
            }

      // encode message
      for (int scan = 0; scan < message.length(); scan++)
      {
            keyValue = encodingQueue.remove();
         encoded += (char) (message.charAt(scan) + keyValue);
         encodingQueue.add(keyValue);
      }

      System.out.println ("Encoded Message:\n" + encoded + "\n");

      // decode message
      for (int scan = 0; scan < encoded.length(); scan++)
      {
         keyValue = decodingQueue.remove();
         decoded += (char) (encoded.charAt(scan) - keyValue);
         decodingQueue.add(keyValue);
      }

      System.out.println ("Decoded Message:\n" + decoded);
   }
}
```