Exercise 4 (10 points) – can be done in pair or individually

- The first lines of all source files must be comments containing names & IDs of all members. Also create file readme.txt containing names & IDs of all members

- Put all files (source, input, readme.txt) in folder Ex4_xxx where xxx = ID of the group representative, i.e. your source files must be in package Ex4_xxx (assumedly in Maven's src/main/java). Input files must be read from this path

- The group representative zips Ex4_xxx & submits it to Google Classroom. The other members submit only readme.txt. Email submission is not accepted

==

1. Complete classes Customer and Shop. Add variables and methods as needed but the given variables must remain private.

```
class Customer {
   private static int runningID = 1;   // for running customer ID
   private int ID;
   private int order;                  // order amount (random value 1-20)
}

class Shop {
   private int itemsToRefill;        // items to refill in odd day
   private int maxDays;              // max day for simulation
   private PriorityQueue<Customer>  orderQueue;
   private ArrayDeque<Customer>     billingQueue;

   public void simulation()          { /* implement simulation */ }
}
```

Rules for orderQueue

- Customer with the highest order amount will have the highest priority.
- Between customers with the same order amounts, the one who comes first (i.e. having lowest ID) will have the highest priority.

2. Get refill items & max days for simulation from user.

3. Create Shop object. Before the simulation (i.e. in Day 0), create 5 Customers and put them in orderQueue. Simply run customer IDs 1, 2, 3, ... and random order amount 1-20.

4. The simulation runs in a loop from day=1 to maxDays. In each day:

   4.1 Refilling: if today is odd day, refill items.

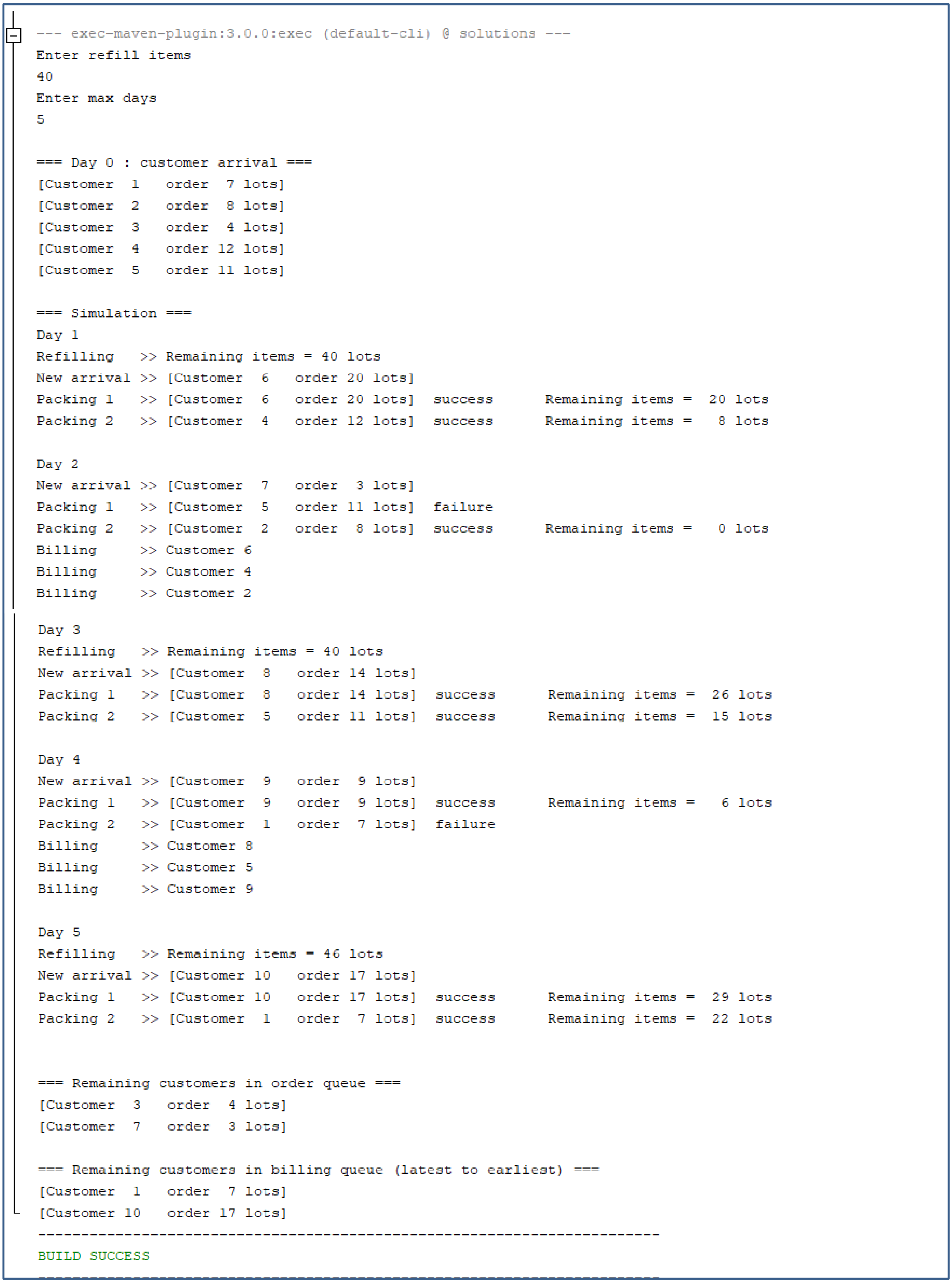
   4.2 New arrival: create a new Customer with random order amount. Put this new customer in orderQueue.

   4.3 Packing orders: take first 2 customers from orderQueue.
   - If there are enough items to pack, print success message & update remaining items. Also put this customer in billingQueue.
   - Otherwise, print failure message. Put this customer back in orderQueue so that he/she will be served in any next day.

   4.4 Billing: if today is even day, take all customers from billingQueue one-by-one & print billing message.

5. After completing maxDays, print remaining customers in both queues
   5.1 Customers remaining in orderQueue, in the order they would have been served (if the simulation continues)

   5.2 Customers remaining in billingQueue, starting from latest to earliest customers



```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ solutions ---
Enter refill items
40
Enter max days
5

=== Day 0 : customer arrival ===
[Customer  1   order  7 lots]
[Customer  2   order  8 lots]
[Customer  3   order  4 lots]
[Customer  4   order 12 lots]
[Customer  5   order 11 lots]

=== Simulation ===
Day 1
Refilling   >> Remaining items = 40 lots
New arrival >> [Customer  6   order 20 lots]
Packing 1   >> [Customer  6   order 20 lots]  success      Remaining items =  20 lots
Packing 2   >> [Customer  4   order 12 lots]  success      Remaining items =   8 lots

Day 2
New arrival >> [Customer  7   order  3 lots]
Packing 1   >> [Customer  5   order 11 lots]  failure
Packing 2   >> [Customer  2   order  8 lots]  success      Remaining items =   0 lots
Billing     >> Customer 6
Billing     >> Customer 4
Billing     >> Customer 2

Day 3
Refilling   >> Remaining items = 40 lots
New arrival >> [Customer  8   order 14 lots]
Packing 1   >> [Customer  8   order 14 lots]  success      Remaining items =  26 lots
Packing 2   >> [Customer  5   order 11 lots]  success      Remaining items =  15 lots

Day 4
New arrival >> [Customer  9   order  9 lots]
Packing 1   >> [Customer  9   order  9 lots]  success      Remaining items =   6 lots
Packing 2   >> [Customer  1   order  7 lots]  failure
Billing     >> Customer 8
Billing     >> Customer 5
Billing     >> Customer 9

Day 5
Refilling   >> Remaining items = 46 lots
New arrival >> [Customer 10   order 17 lots]
Packing 1   >> [Customer 10   order 17 lots]  success      Remaining items =  29 lots
Packing 2   >> [Customer  1   order  7 lots]  success      Remaining items =  22 lots


=== Remaining customers in order queue ===
[Customer  3   order  4 lots]
[Customer  7   order  3 lots]

=== Remaining customers in billing queue (latest to earliest) ===
[Customer  1   order  7 lots]
[Customer 10   order 17 lots]
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```