EGCI 213
Group Project 2 – Parcel Delivery

The project can be done in a group of <= 5 students. Each group must do the project by themselves

- **Everyone involved in cheating, either as source or copier, will get ZERO point.**
- If I suspect that you don't do the project all by yourself, I may ask you to do programming quizzes about the suspicious points in person, in front of me, and all by yourself (i.e. in presence of only group members).

```
days,                      6
bike_num_maxload,         10,  20
truck_num_maxload,        10,  80
seller_num_maxdrop,        3, 200
delivery_bybike_bytruck,  2,   2
```

1. This project uses only 1 input file (config.txt). first column of each line indicates the type of input data.
   - 1.1 Line "days" is followed by #days of simulation.
   - 1.2 Line "`bike_num_maxload`" is followed by #bikes in bike fleet and max parcel load per bike.
   - 1.3 Line "`truck_num_maxload`" is followd by #trucks in truck fleet and max parcel load per truck.
   - 1.4 Line "`seller_num_maxdrop`" is followed by #sellers and max parcel drop per day.
   - 1.5 Line "`delivery_bybike_bytruck`" is followed by #delivery-by-bike and #deliver-by-truck shops.

** Don't hard code these values. I may change some of them to check whether your calculation is correct.
- There are always 5 lines with columns as stated above.
- But numbers e.g. #days, #bikes, #trucks, #sellers, #delivery shops may change.
- There won't be any input error (e.g. invalid input, negative number, wrong format, missing columns) in this file. But the program must still handle the case of missing file. Don't let it crash.

2. Implement **class Fleet** that represents a fleet of delivery vehicles. There are always 2 Fleets in the program: bike Fleet and truck Fleet.
   - Your class should have method for vehicle allocation (see 5).

3. Implement **class SellerThread** that represents an individual seller as thread. Thread activities are done in loop. Each iteration of a loop = 1 day. In each day:
   - 3.1 Wait until 1 thread (main, SellerThread, or DeliveryThread) prints day number.
   - 3.2 Drop parcels at 1 delivery shop & update parcel balance at that shop. The number of parcels to drop is randomed, but it must not exceed max parcel drop (read from config.txt). The choice of delivery shop is also randomed. Print thread activities as in the demo.
     - All SellerThreads must see the same list of DeliveryShops.

4. Implement **class DeliveryShop** that represents an individual shop, and **class DeliveryThread** that represents an individual shop manager as thread. Since each shop is managed by 1 thread, you can assign identical name to them for simplicity. Thread activities are done in loop. Each iteration of a loop = 1 day. In each day:
   - 4.1 Wait until 1 thread (main, SellerThread, or DeliveryThread) prints day number, and all SellerThreads finish dropping parcels.
   - 4.2 Report number of parcels it has to deliver, which is accumulated from previous days + previous step.
   - 4.3 Each shop has only 1 mean of delivery: bikes or trucks. So, try to get bikes/trucks for parcel delivery (see 5) & update remaining parcels. Print thread activities as in the demo.
     - All DeliveryThreads must see the same bike/truck Fleet.
   - 4.4 After completing all days of simulation, calculate success rate = total delivered parcels / total received parcels.

5. Number of available bikes/trucks is reset every day. For each thread to get vehicles for parcel delivery:
   5.1 From vehicle's max load, calculate #vehicles needed for the parcels.
   5.2 Each allocated vehicle must be at least half full. For example, if vehicle's max load = 20, it will be allocated only if parcels to deliver >= 10.
   5.3 Each thread will try to get as many available vehicles as possible. Remaining parcels that can't be delivered today will be accumulated for the next day.

6. Implement main class with main method.
   6.1 Read simulation parameters from config.txt.
   6.2 Create Fleets, SellerThreads, DeliveryShops & DeliveryThreads. Start all threads. You are recommended to use ArrayLists to keep objects for flexibility.
   6.3 After all threads complete all days of simulation, let main thread report total parcels received & delivered and success rates of all DeliveryShops, sorted in decreasing order of success rate. If success rates are equal, sort the shops by their names.

** Everything printed to the screen must be labelled by the name of the thread who prints it. Don't hard code thread's name but use Thread.currentThread().getName()

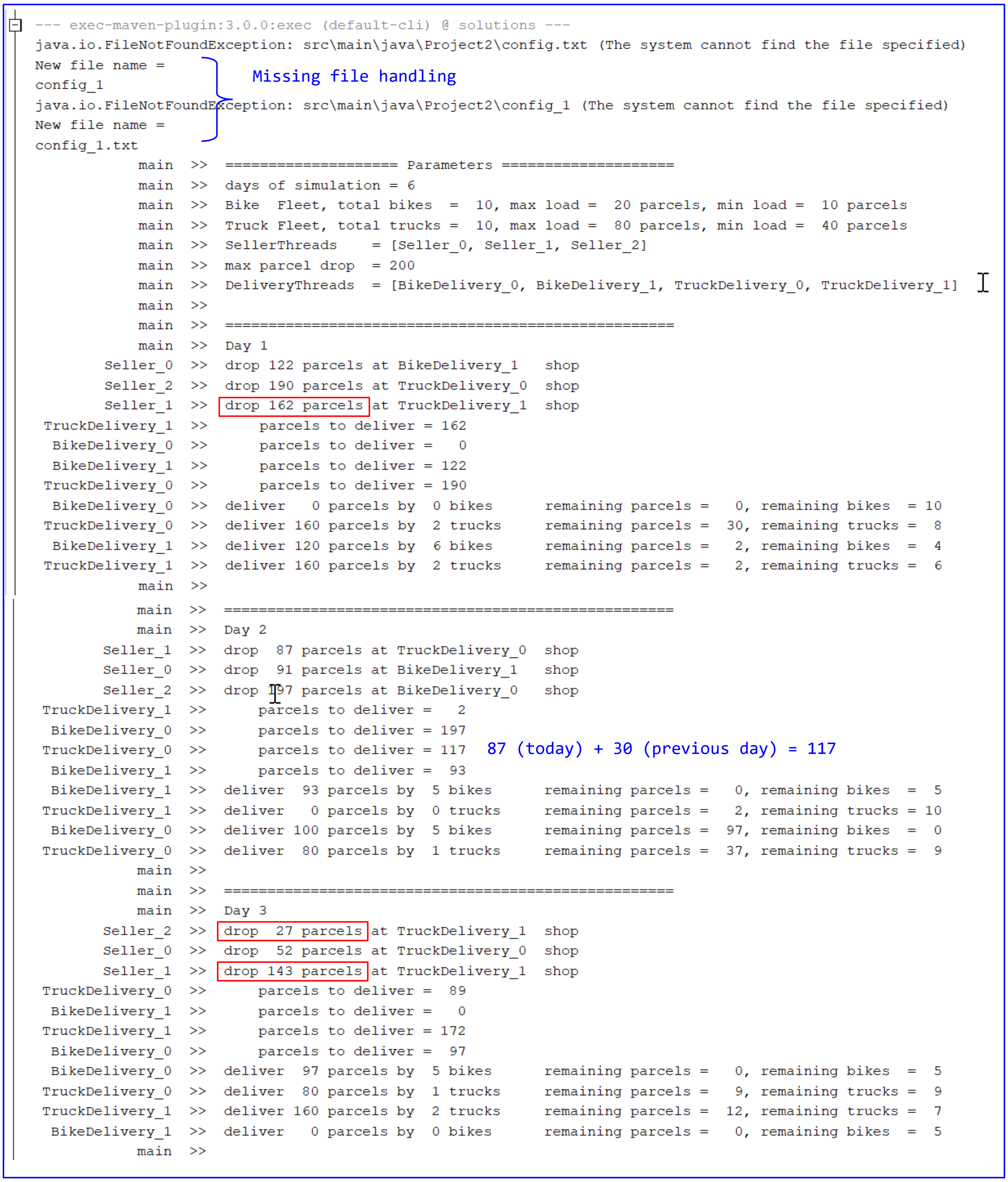
7. Package and folder structure must be correct
   7.1 Your source files (.java) must be in folder Project2_XXX where XXX = full ID of the group representative, assuming that this folder is under Maven's "src/main/java" structure. The first lines of all source files must be comments containing names & IDs of all members.
   7.2 Input files must be read from Project2_XXX. Don't use absolute path that is valid only on your PC.
   7.3 Add readme.txt containing names & IDs of all members in Project2_XXX.

<u>Submission</u>
1. Group representative zips and submits Project2_XXX to Google classroom
2. Other members submit only readme.txt to Google classroom

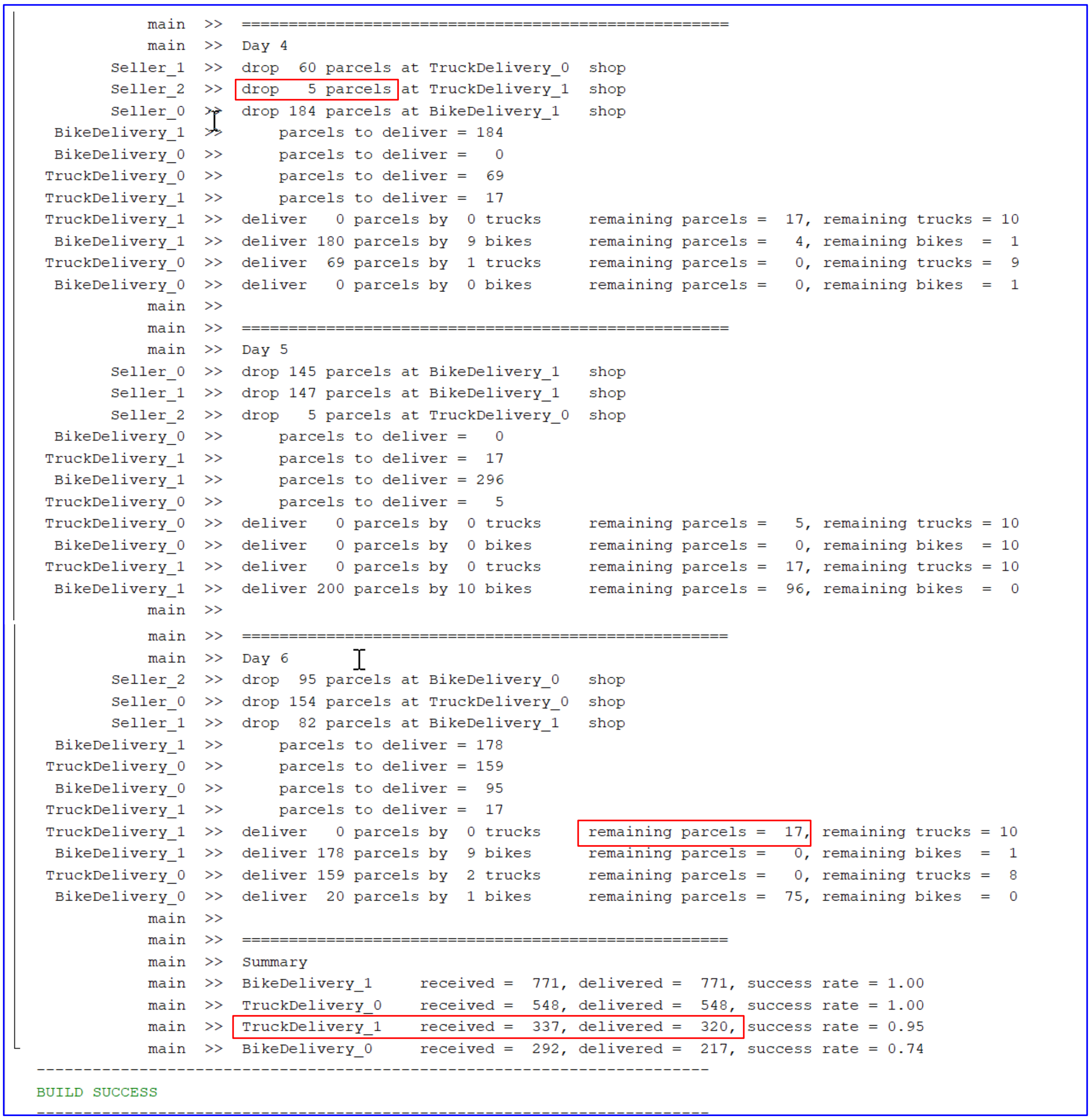
<u>Grading</u>

1	point	correct steps + results by SellerThread (dropping parcels)
3	points	correct steps + results by DeliveryThread (remaining parcels, bike/truck allocation)
1	point	correct summary by main thread
1	point	other requirements (thread name, missing file handling)
4	points	design & programming in proper OOP and multithreading style

```
days,                      6
bike_num_maxload,        10,  20
truck_num_maxload,       10,  80
seller_num_maxdrop,       3, 200
delivery_bybike_bytruck, 2,   2
```



```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ solutions ---
java.io.FileNotFoundException: src\main\java\Project2\config.txt (The system cannot find the file specified)
New file name =
config_1                                  Missing file handling
java.io.FileNotFoundException: src\main\java\Project2\config_1 (The system cannot find the file specified)
New file name =
config_1.txt
            main  >>  ==================== Parameters ====================
            main  >>  days of simulation = 6
            main  >>  Bike  Fleet, total bikes  =  10, max load =  20 parcels, min load =  10 parcels
            main  >>  Truck Fleet, total trucks =  10, max load =  80 parcels, min load =  40 parcels
            main  >>  SellerThreads    = [Seller_0, Seller_1, Seller_2]
            main  >>  max parcel drop  = 200
            main  >>  DeliveryThreads  = [BikeDelivery_0, BikeDelivery_1, TruckDelivery_0, TruckDelivery_1]
            main  >>
            main  >>  ====================================================
            main  >>  Day 1
        Seller_0  >>  drop 122 parcels at BikeDelivery_1   shop
        Seller_2  >>  drop 190 parcels at TruckDelivery_0  shop
        Seller_1  >>  drop 162 parcels at TruckDelivery_1  shop
 TruckDelivery_1  >>      parcels to deliver = 162
  BikeDelivery_0  >>      parcels to deliver =   0
  BikeDelivery_1  >>      parcels to deliver = 122
 TruckDelivery_0  >>      parcels to deliver = 190
  BikeDelivery_0  >>  deliver   0 parcels by  0 bikes      remaining parcels =   0, remaining bikes  = 10
 TruckDelivery_0  >>  deliver 160 parcels by  2 trucks     remaining parcels =  30, remaining trucks =  8
  BikeDelivery_1  >>  deliver 120 parcels by  6 bikes      remaining parcels =   2, remaining bikes  =  4
 TruckDelivery_1  >>  deliver 160 parcels by  2 trucks     remaining parcels =   2, remaining trucks =  6
            main  >>
            main  >>  ====================================================
            main  >>  Day 2
        Seller_1  >>  drop  87 parcels at TruckDelivery_0  shop
        Seller_0  >>  drop  91 parcels at BikeDelivery_1   shop
        Seller_2  >>  drop 197 parcels at BikeDelivery_0   shop
 TruckDelivery_1  >>      parcels to deliver =   2
  BikeDelivery_0  >>      parcels to deliver = 197
 TruckDelivery_0  >>      parcels to deliver = 117  87 (today) + 30 (previous day) = 117
  BikeDelivery_1  >>      parcels to deliver =  93
  BikeDelivery_1  >>  deliver  93 parcels by  5 bikes      remaining parcels =   0, remaining bikes  =  5
 TruckDelivery_1  >>  deliver   0 parcels by  0 trucks     remaining parcels =   2, remaining trucks = 10
  BikeDelivery_0  >>  deliver 100 parcels by  5 bikes      remaining parcels =  97, remaining bikes  =  0
 TruckDelivery_0  >>  deliver  80 parcels by  1 trucks     remaining parcels =  37, remaining trucks =  9
            main  >>
            main  >>  ====================================================
            main  >>  Day 3
        Seller_2  >>  drop  27 parcels at TruckDelivery_1  shop
        Seller_0  >>  drop  52 parcels at TruckDelivery_0  shop
        Seller_1  >>  drop 143 parcels at TruckDelivery_1  shop
 TruckDelivery_0  >>      parcels to deliver =  89
  BikeDelivery_1  >>      parcels to deliver =   0
 TruckDelivery_1  >>      parcels to deliver = 172
  BikeDelivery_0  >>      parcels to deliver =  97
  BikeDelivery_0  >>  deliver  97 parcels by  5 bikes      remaining parcels =   0, remaining bikes  =  5
 TruckDelivery_0  >>  deliver  80 parcels by  1 trucks     remaining parcels =   9, remaining trucks =  9
 TruckDelivery_1  >>  deliver 160 parcels by  2 trucks     remaining parcels =  12, remaining trucks =  7
  BikeDelivery_1  >>  deliver   0 parcels by  0 bikes      remaining parcels =   0, remaining bikes  =  5
            main  >>
```



```
           main  >>  ====================================================
           main  >>  Day 4
       Seller_1  >>  drop  60 parcels at TruckDelivery_0  shop
       Seller_2  >>  drop   5 parcels at TruckDelivery_1  shop
       Seller_0  >>  drop 184 parcels at BikeDelivery_1   shop
 BikeDelivery_1  >>      parcels to deliver = 184
 BikeDelivery_0  >>      parcels to deliver =   0
TruckDelivery_0  >>      parcels to deliver =  69
TruckDelivery_1  >>      parcels to deliver =  17
TruckDelivery_1  >>  deliver   0 parcels by  0 trucks     remaining parcels =  17, remaining trucks = 10
 BikeDelivery_1  >>  deliver 180 parcels by  9 bikes      remaining parcels =   4, remaining bikes  =  1
TruckDelivery_0  >>  deliver  69 parcels by  1 trucks     remaining parcels =   0, remaining trucks =  9
 BikeDelivery_0  >>  deliver   0 parcels by  0 bikes      remaining parcels =   0, remaining bikes  =  1
           main  >>
           main  >>  ====================================================
           main  >>  Day 5
       Seller_0  >>  drop 145 parcels at BikeDelivery_1   shop
       Seller_1  >>  drop 147 parcels at BikeDelivery_1   shop
       Seller_2  >>  drop   5 parcels at TruckDelivery_0  shop
 BikeDelivery_0  >>      parcels to deliver =   0
TruckDelivery_1  >>      parcels to deliver =  17
 BikeDelivery_1  >>      parcels to deliver = 296
TruckDelivery_0  >>      parcels to deliver =   5
TruckDelivery_0  >>  deliver   0 parcels by  0 trucks     remaining parcels =   5, remaining trucks = 10
 BikeDelivery_0  >>  deliver   0 parcels by  0 bikes      remaining parcels =   0, remaining bikes  = 10
TruckDelivery_1  >>  deliver   0 parcels by  0 trucks     remaining parcels =  17, remaining trucks = 10
 BikeDelivery_1  >>  deliver 200 parcels by 10 bikes      remaining parcels =  96, remaining bikes  =  0
           main  >>
           main  >>  ====================================================
           main  >>  Day 6
       Seller_2  >>  drop  95 parcels at BikeDelivery_0   shop
       Seller_0  >>  drop 154 parcels at TruckDelivery_0  shop
       Seller_1  >>  drop  82 parcels at BikeDelivery_1   shop
 BikeDelivery_1  >>      parcels to deliver = 178
TruckDelivery_0  >>      parcels to deliver = 159
 BikeDelivery_0  >>      parcels to deliver =  95
TruckDelivery_1  >>      parcels to deliver =  17
TruckDelivery_1  >>  deliver   0 parcels by  0 trucks     remaining parcels =  17, remaining trucks = 10
 BikeDelivery_1  >>  deliver 178 parcels by  9 bikes      remaining parcels =   0, remaining bikes  =  1
TruckDelivery_0  >>  deliver 159 parcels by  2 trucks     remaining parcels =   0, remaining trucks =  8
 BikeDelivery_0  >>  deliver  20 parcels by  1 bikes      remaining parcels =  75, remaining bikes  =  0
           main  >>
           main  >>  ====================================================
           main  >>  Summary
           main  >>  BikeDelivery_1     received =  771, delivered =  771, success rate = 1.00
           main  >>  TruckDelivery_0    received =  548, delivered =  548, success rate = 1.00
           main  >>  TruckDelivery_1    received =  337, delivered =  320, success rate = 0.95
           main  >>  BikeDelivery_0     received =  292, delivered =  217, success rate = 0.74
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```