EGCI 213

Group Project 2 – Travel Management

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 late submitting group copies code from a graded group, the graded group will still be penalized.
- If I suspect that you don't do the project all by yourself (taking code from ChatGPT is counted as not doing the project by yourself), I may ask you to do programming quizzes about the suspicious points in person.

1. This project uses only 1 input file (config.txt). first column of each line indicates the type of input data.

```
days,                5
agency_num_arrival,  5, 50
tour_num_capacity,   4, 50
place_num,           2
```

   1.1 Line "days" is followed by #days of simulation.
   1.2 Line "`agency_num_arrival`" is followed by #travel agencies and max daily arrival of customers at each agency.
   1.3 Line "`tour_num_capacity`" is followed by #tours and the capacity of each tour.
   1.4 Line "`place_num`" is followed by #places

** Don't hard code these values. I may change some of them to check whether your calculation is correct.
- There are always 4 lines with columns as stated above.
- But numbers may be changed.
- 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 AgencyThread** that represents an individual travel agency as thread. Thread activities are done in loop. Each iteration of a loop = 1 day. In each day:
   2.1 Wait until 1 thread (main, AgencyThread, or OperatorThread) prints day number.
   2.2 Receive customers and update remaining customers (from previous days + today). The number of arriving customers is random (<= max daily arrival). Print thread activities as in the demo.
   2.3 Send as many customers as it can to a tour and update seats taken in that tour. The choice of tour is random. Print thread activities as in the demo.
      - All AgencyThreads must see the same list of Tours

3. Implement **class Tour** that represents an individual tour, and **class OperatorThread** that represents an individual tour operator as thread. Each tour is operated by only 1 OperatorThread. Thread activities are done in loop. Each iteration of a loop = 1 day. In each day:
   3.1 Wait until 1 thread (main, AgencyThread, or OperatorThread) prints day number, and all AgencyThreads finish sending customers.
   3.2 If the tour has no customer, simply report no customer.
   3.3 If the tour has >=1 customer, take all of them to a place and update visitor count at that place. The choice of place is also random. Print thread activities as in the demo.
      - All OperatorThreads must see the same list of Places.
   3.4 Also update total customers received by the tour. This total is accumulated over all days of simulation, and will be reported in the summary (see 5.3).
   3.5 But seats taken in the tour is reset every day.

4. Implement **class Place** that represents an individual place. Update visitor count when OperatorThread takes customers to the place. This count is accumulated over all days of simulation.

5. Implement main class with main method.
   5.1 Read simulation parameters from config.txt.
   5.2 Create AgencyThreads, Tours, OperatorThreads, and Places. Start all threads. You are recommended to use ArrayLists to keep objects for flexibility.
   5.3 After all threads complete all days of simulation, let main thread report total customers received by the tours, sorted in decreasing order of customers then by tour's name.

** 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()

6. Package and folder structure must be correct
   6.1 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.
   6.2 Input files must be read from Project2_XXX. Don't use absolute path that is valid only on your PC.
   6.3 Add readme.txt containing names & IDs of all members in Project2_XXX.

Submission

1. Group representative zips and submits Project2_XXX to Google classroom
2. Other members submit only readme.txt to Google classroom

Grading

3	point	correct steps + results by AgencyThread (arrival step, sending step)
1	points	correct steps + results by OperatorThread (visiting step)
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

Late submission: -0.5 points for <1 week late; -1 point for each 1 full week late

```
days,                5
agency_num_arrival,  5, 50
tour_num_capacity,   4, 50
place_num,           2
```

```
--- compiler:3.11.0:compile (default-compile) @ solutions ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ solutions ---
java.io.FileNotFoundException: src\main\java\Project2\config.txt (The system cannot find the file specified)
New file name =
configs.txt
java.io.FileNotFoundException: src\main\java\Project2\configs.txt (The system cannot find the file specified)
New file name =
config1
java.io.FileNotFoundException: src\main\java\Project2\config1 (The system cannot find the file specified)
New file name =
config_1.txt
```

Missing file handling

```
            main  >>  ==================== Parameters ====================
            main  >>  Days of simulation = 5
            main  >>  Max arrival        = 50
            main  >>  AgencyThreads      = [AgencyThread_0, AgencyThread_1, AgencyThread_2, AgencyThread_3, AgencyThread_4]
            main  >>  Tour capacity      = 50
            main  >>  OperatorThreads    = [OperatorThread_0, OperatorThread_1, OperatorThread_2, OperatorThread_3]
            main  >>  Places             = [Place_0, Place_1]
            main  >>
             main  >>  ====================================================
             main  >>  Day 1
             main  >>
  AgencyThread_2  >>  new arrival = 29                 remaining customers =  29
  AgencyThread_0  >>  new arrival = 44                 remaining customers =  44
  AgencyThread_1  >>  new arrival = 44                 remaining customers =  44
  AgencyThread_3  >>  new arrival = 34                 remaining customers =  34
  AgencyThread_4  >>  new arrival = 24                 remaining customers =  24
  AgencyThread_4  >>  send  24 customers to Tour_1     seats taken = 24
  AgencyThread_0  >>  send  44 customers to Tour_3     seats taken = 44
  AgencyThread_1  >>  send  44 customers to Tour_2     seats taken = 44
  AgencyThread_2  >>  send  26 customers to Tour_1     seats taken = 50
  AgencyThread_3  >>  send   6 customers to Tour_2     seats taken = 50
  AgencyThread_3  >>
OperatorThread_3  >>  take  44 customers to Place_0    visitor count =   44
OperatorThread_0  >>  no customer
OperatorThread_1  >>  take  50 customers to Place_1    visitor count =   50
OperatorThread_2  >>  take  50 customers to Place_1    visitor count =  100
            main  >>
            main  >>  ====================================================
            main  >>  Day 2
            main  >>
  AgencyThread_3  >>  new arrival = 19                 remaining customers =  47
  AgencyThread_4  >>  new arrival = 44                 remaining customers =  44
  AgencyThread_0  >>  new arrival =  5                 remaining customers =   5
  AgencyThread_2  >>  new arrival = 17                 remaining customers =  20
  AgencyThread_1  >>  new arrival = 20                 remaining customers =  20
  AgencyThread_1  >>  send  20 customers to Tour_2     seats taken = 20
  AgencyThread_0  >>  send   5 customers to Tour_2     seats taken = 25
  AgencyThread_3  >>  send  47 customers to Tour_3     seats taken = 47
  AgencyThread_2  >>  send   3 customers to Tour_3     seats taken = 50
  AgencyThread_4  >>  send   0 customers to Tour_3     seats taken = 50
  AgencyThread_4  >>
OperatorThread_2  >>  take  25 customers to Place_1    visitor count =  125
OperatorThread_3  >>  take  50 customers to Place_0    visitor count =   94
OperatorThread_0  >>  no customer
OperatorThread_1  >>  no customer
            main  >>
```

Must not exceed tour capacity

+28 from yesterday

```
            main  >>  ==================================================
            main  >>  Day 3
            main  >>
  AgencyThread_2  >>  new arrival = 10                  remaining customers =  27
  AgencyThread_0  >>  new arrival = 44                  remaining customers =  44
  AgencyThread_4  >>  new arrival = 30                  remaining customers =  74
  AgencyThread_1  >>  new arrival = 40                  remaining customers =  40
  AgencyThread_3  >>  new arrival = 19                  remaining customers =  19
  AgencyThread_3  >>  send  19 customers to Tour_2      seats taken = 19
  AgencyThread_0  >>  send  44 customers to Tour_3      seats taken = 44
  AgencyThread_4  >>  send  50 customers to Tour_1      seats taken = 50
  AgencyThread_1  >>  send  40 customers to Tour_0      seats taken = 40
  AgencyThread_2  >>  send  27 customers to Tour_2      seats taken = 46
  AgencyThread_2  >>
OperatorThread_1  >>  take  50 customers to Place_0     visitor count =   144
OperatorThread_0  >>  take  40 customers to Place_1     visitor count =   165
OperatorThread_3  >>  take  44 customers to Place_0     visitor count =   188
OperatorThread_2  >>  take  46 customers to Place_0     visitor count =   234
            main  >>
            main  >>  ==================================================
            main  >>  Day 4
            main  >>
  AgencyThread_1  >>  new arrival =  3                  remaining customers =   3
  AgencyThread_2  >>  new arrival = 18                  remaining customers =  18
  AgencyThread_3  >>  new arrival =  3                  remaining customers =   3
  AgencyThread_0  >>  new arrival = 34                  remaining customers =  34
  AgencyThread_4  >>  new arrival =  8                  remaining customers =  32
  AgencyThread_4  >>  send  32 customers to Tour_3      seats taken = 32
  AgencyThread_1  >>  send   3 customers to Tour_1      seats taken =  3
  AgencyThread_2  >>  send  18 customers to Tour_2      seats taken = 18
  AgencyThread_3  >>  send   3 customers to Tour_3      seats taken = 35
  AgencyThread_0  >>  send  34 customers to Tour_1      seats taken = 37
  AgencyThread_0  >>
OperatorThread_2  >>  take  18 customers to Place_0     visitor count =   252
OperatorThread_1  >>  take  37 customers to Place_1     visitor count =   202
OperatorThread_0  >>  no customer
OperatorThread_3  >>  take  35 customers to Place_1     visitor count =   237
            main  >>
            main  >>  ==================================================
            main  >>  Day 5
            main  >>
  AgencyThread_3  >>  new arrival = 20                  remaining customers =  20
  AgencyThread_0  >>  new arrival = 21                  remaining customers =  21
  AgencyThread_4  >>  new arrival = 43                  remaining customers =  43
  AgencyThread_1  >>  new arrival = 10                  remaining customers =  10
  AgencyThread_2  >>  new arrival = 43                  remaining customers =  43
  AgencyThread_2  >>  send  43 customers to Tour_1      seats taken = 43
  AgencyThread_3  >>  send  20 customers to Tour_0      seats taken = 20
  AgencyThread_0  >>  send  21 customers to Tour_2      seats taken = 21
  AgencyThread_1  >>  send  10 customers to Tour_0      seats taken = 30
  AgencyThread_4  >>  send  29 customers to Tour_2      seats taken = 50
  AgencyThread_4  >>
OperatorThread_3  >>  no customer
OperatorThread_2  >>  take  50 customers to Place_0     visitor count =   302
OperatorThread_1  >>  take  43 customers to Place_1     visitor count =   280
OperatorThread_0  >>  take  30 customers to Place_0     visitor count =   332
            main  >>
            main  >>  ==================================================
            main  >>  Summary
            main  >>  Tour_2     total customers =  189
            main  >>  Tour_1     total customers =  180
            main  >>  Tour_3     total customers =  173
            main  >>  Tour_0     total customers =   70
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```