German University in Cairo
Faculty of Media Engineering and Technology
Dr. Aysha Alsafty
Eng. Fadwa Elhussini
Eng. Eslam Osama

CSEN 602-Operating Systems, Spring 2018
Mini-project 3: Synchronization
Deadline: 22.4.2018
Grade: 8%

In this mini-project you will use Java threads to simulate the Big Wheel funfair ride. The wheel has a capacity of 5 players, where each player is simulated by a thread that has a unique ID. A new ride is started every *max_wait_time* msecs.

A player thread arrives to the wheel after an initial sleep period, specified in the input file. The player thread then boards the wheel to eventually take the ride. Boarding is managed by an operator.

The operator controls access to the wheel and operates it, where each operation is a ride. The operator admits players to the wheel and later makes sure all players leave the wheel after the ride is complete. The operator runs a ride in one of two cases: either the wheel is full (capacity is reached) or the *max_wait_time* of the wheel is reached, in which case a ride is run even if the wheel capacity has not been reached. You may assume the ride time to be negligible. Once the ride is operated, the wheel is emptied and *max_wait_time* for the next ride starts.

Input Format

You are provided with two sample input files with the format:

max_wait_time
total_players_count
thread_id Waiting_time
thread_id Waiting_time
...

Output Format

Before a ride is run, your simulation should print the IDs of the player threads currently in the wheel, in order of arrival. A sample output file is provided.

Suggested Design

This section provides a suggested design to guide your implementation. You may choose to follow an alternative design, as long as each player is represented by a thread and the behavior described above is realized by your design.

Player Thread

- A player has four main attributes: ID, waiting time, on-board flag, and a ride-complete flag. Both flags, on-board and ride-complete, are initially false.
- A Player thread is initially put to sleep according to its waiting time. Upon waking up, the player calls the operator to queue for the next ride.

Wheel Thread

- Wheel has four main attributes: capacity, count of currently on-board players, list of currently on-board players and the maximum waiting time of the wheel.
- The wheel is put to sleep for max_wait_time upon start.
- A method *load_players()* adds a player thread to the list of on-board players.
- A method *run_ride()* updates the state of on-board threads to *ride-complete*.
- A method *end_ride()* empties the list of on-board players and puts the wheel to sleep until the next ride is run.

Operator

- The operator is responsible for managing the simulation. It keeps an instance of the wheel. It loads the wheel until it reaches capacity, then runs the wheel and ends the ride.
- The operator creates, starts and keeps a list of player threads.
- The operator creates and starts the wheel thread.

German University in Cairo
Faculty of Media Engineering and Technology
Dr. Aysha Alsafty
Eng. Fadwa Elhussini
Eng. Eslam Osama

- The operator terminates when all players have successfully completed their rides.

Submission

- The project deadline is on 23/4 at 11:59 PM .
- **No late** submissions will be accepted.
- Cheating cases will be graded by 0 for all teams involved.
- It is your responsibility to make sure that the files were uploaded successfully to the website.

Submission guidelines:

- The submission will be through the following link: `https://goo.gl/forms/AjK96ckUqzTeVKjA2`
- The project files should be uploaded on a Google drive and you will be submitting the link to this drive file.
- The drive should have all your Java files.
- Please check the updated teams list on the MET website for your team number.