deep se
dependable evolvable pervasive software engineering group

Exercise Session – Taxi Call Center

Federica Filippini

Politecnico di Milano

federica.filippini@polimi.it

POLITECNICO DI MILANO

Goal

- Develop the core class `CallCenter` of the `MyTaxi` application for a small city with one railroad station.
- `CallCenter` provides an interface to the user to call taxi and store information about all the taxis of the city.
- Each taxi is identified in a unique way by the license id (of type `std::string`) of its driver.

- **Note:** since most of the rides are from or to the rail station, ad-hoc functionalities for this type of rides have to be implemented.

Code Structure

Place

- x_coordinate
- y_coordinate

+ // getters
+ operator==

(friend) ComputeDistance(place1,place2)

Taxi

- license_id
- total_distance
- Place last _ride_source
- Place last_ride_destination

+ // getters
+ SetRide (source, destination)
+ pair<Place, Place> CGetLastRide()
+ AddDistance (distance)

(friend) operator<

Date

- day, month, year

+ print()

(friend) operator<
(friend) operator!=

CallCenter

- **???** available_taxis
- **???** station_available_taxis
- unordered_map<string, Taxi> taxis
- Place station

+ // required methods

Required methods

- **`Taxi Call(const Place&, const Place&)`**, that returns the taxi, if any, whose current position is closest to the source of the ride.
- **`Taxi CallAtRailStation(const Place&)`**, that returns the available taxi which: 1) is currently at the rail station, and 2) has run the smallest distance among all available taxis located at the rail station.
- **`Taxi CallToRailStation(const Place&)`**, that returns the taxi, if any, whose current position is closest to the source of the ride
- **`Arrived(const string&)`**, that updates information about `available_taxis`, overall distance of taxi, and eventually `station_available_taxis`.

- **Note:** in all methods, if there are multiple taxis matching the request, whatever of them can be returned.