CSC148, Assignment #1
Restaurant Simulation
Due June 17th, 4:30 pm
Late submissions are not accepted.

1 Introduction

Pat & Mat [1] are two friends who try to solve different problems—many of which are self-made. You can see them in action here, if you wish. They have recently decided to begin their culinary careers by opening their first restaurant at the University of Toronto. Their restaurant in the first floor of Bahen Centre for Information Technology would have a small kitchen that can prepare only one order at a time. They expect to have a large number of customers during rush hour. So, they will need to have some approach to pick the next customer to serve among all waiting ones, such that they maximize their profit and/or customer satisfaction.

There are different parameters that they need to consider when deciding which customer should be served next. Each customer's order requires a specific amount of time to prepare, and will earn a specific profit. For example, one customer's order may require 10 minutes to prepare for a profit of $\$2^{00}$, while another order may be prepared in 15 minutes for a $\$3^{50}$ profit. Further, customers will not wait in the line indefinitely; each customer will leave after some time based on their level of patience. For example, one customer will leave if their order is not picked in 10 minutes, where another customer will wait up to 30 minutes.

Pat & Mat invited two friends, Max and Pac, to help them decide how to serve their customers. All four proposed a different serving approach, and they cannot agree unanimously on any:

1) Pat suggested serving customers based on their arrival time, (i.e., selecting the customer who arrives earliest). He argued that this is a fair way of doing it.
2) Mat thinks that it will be easier to just select the last person who arrives.
3) Max only thinks about money, so he suggested selecting the customer who can bring the highest profit.
4) Pac thinks food should come fast, so he suggested selecting the customer whom can be served in the shortest time.

When Pat & Mat failed to decide on the best approach, they decided to get your help.

They are asking you to write a simulator to see how each approach works in different scenarios. In this project ("Assignment 1"), you will design and develop a restaurant simulator to simulate these four different approaches to find out which is the best. Note that there might be multiple strategies are the best.

2 Simulation Overview

Computer simulations are great for exploring "what if" questions about scenarios before, or instead of, letting those scenarios play out in the real world. Real world problems are usually very complex due to having many parameters involved, which makes it difficult to simulate them precisely. Hence, we usually simulate a simplified version of the real problem in which less-yet-important parameters are involved only. This enables us to study the effect of each of those parameters on the outcomes.

[1] For more information on Pat & Mat, you may want to see https://en.wikipedia.org/wiki/Pat %26 Mat

In this assignment—simulation of a simplified restaurant—you will implement a turn-based simulation. In this model, each turn represents one unit of time (here minute) in the real world. Further, events can happen at the beginning of a turn. For example, a customer is considered entering the restaurant or leaving it only at the beginning of a turn. As another example, a customer order is started to be prepared at the beginning of a turn, and it will be prepared at the beginning of another turn. In other words, each order needs an integer number of turns to be completed, e.g. 4 turns, 15 turns, etc. We also assume that the order of only one customer is prepared at a time.

Here, it is a simple example of how one run of the simulation would look like: Alice and Bob enter the restaurant at time 1. With some serving approach, Alice's order is picked to be prepared, and it needs 4 turns to be prepared. Carol enters the restaurant at time 3, joining Bob who is still waiting in the line. At time 5, Alice's order is ready, and (with some serving approach) Carol's order is picked to be prepared, and it needs 8 turns. At time 10, Bob's patience is over and he leaves the restaurant. At time 13, Carol's order is ready, …

There are two criteria that we want to use to measure the success of a serving approach:

1) **Restaurant Profit**. What is the total profit for each scenario? Obviously, Pat & Mat are not running a charity restaurant. They want to maximise their profit.
2) **Customer Satisfaction**. How many customers have they served? Although they want to maximize their profit, they also want to have a good reputation. If many customers are turned away, they will lose customers over time, which is not good. Furthermore, those customers who leave may post negative reviews on Yelp, which will hurt their brand and reduce their profit in the future.

3 Simulation Classes

We will review the purpose and functionality of main classes that you need to complete in this assignment.

Simulator

This class is responsible for reading the simulation scenario from a file, creating instances of different serving approaches, dispatching different events (a new customer arrives, and start of a next turn), and printing results at the end.

All of the functions that are necessary for it to work are already implemented in the starter code. However, you can add any additional function or attribute if it helps your development. However, note that in the auto-testing, you can only call methods of a simulator object from another object that are defined in the starter code.

The starter code for the simulator (simulator.py) also shows you some of the methods of other classes that you need to implement. So read it carefully, and be sure that your implementation of other functions are compatible with it.

Customer

This class represents a customer that will enter the restaurant. Each customer has the following information:

- ID. A unique id that is used to easily identify that customer.
- Profit. The profit the restaurant will earn if the customer's order is prepared.
- Prepare Time. The number of turns needed to prepare the order.
- Patience. The maximum number of turns that this customer will wait for their order to be picked for preparation.

The starter code (customer.py) gives you some overall hint of what you should implement. However, this is your job to implement all necessary methods that will be called by other objects.

Restaurant

The restaurant abstracts the common functions among different serving approaches. It also defines the function that will be called by the simulator:

- Event receiver for arrival of a new customer.
- Processing the events that happen at the turn (including which customers leave due to not being served, and which customer's order is picked if the previous order is finished).
- Printing the final results at the end of simulation (total profit and number of customers that were served).

You will implement four subclasses that implement the different serving approaches that were suggested by Pat, Mat, Max, and Pac. These classes will be called `PatApproach`, `MatApproach`, `MaxApproach`, and `PacApproach`.

The starter code (`restaurant.py`) includes the function signature of three main functions that you should implement: `add_customer`, `process_turn`, and `write_report`. Note that you may decide to leave one or more of these functions as abstract in the superclass, and implement them in subclasses. Alternatively, you may implement some common parts in the superclass, and complete them in subclasses. Please note that you should follow the function design recipe for all of the functions that you implement. In the starter code, you may be given part of it, and should complete them with any missing part.

4 Report File Format

The function `write_report` of the restaurant class (and/or its subclasses) should write a short report of simulation results at the end of simulation. Here, it is the format of the output that should be written in the report file for a specific restaurant. There should be three lines.

The first line should be:

Results for the serving approach using NAME's suggestion:

Where NAME is replaced by Pat, Mat, Max, or Pac. For example, if this is Pat's suggested serving approach, then it should be:

Results for the serving approach using Pat's suggestion:

The second and third line should print the total profit and number of customers that served in the following way:

Total profit: PROFIT

Customer served: CUSTOMER_COUNT

For example, if the total profit is $25.3 and it served 11 customers, then it will be:

Total profit: $25.30

Customer served: 11

Please pay attention to the exact formatting or requested output. Your code will be tested with an automated script. If there is any difference between your output and requested output (even if it is a single typo or additional dot or colon), your answer will be considered wrong.

5 Scenario File Format

Each scenario of customer arrivals is presented in a text file. In this file, each line represents a customer, and contains 5 numbers that are separated by tabs:

- The first number is the turn that this customer will enter the restaurant. You can assume that customers appear in the file in the order that they arrive, i.e., the turn number is non-decreasing. However, note that multiple customers may enter the restaurant in one turn.
- The second number is the unique customer id. The only assumption that you can have about customer ids is that they are unique integers (so, no two customers will have the same id).
- The third number is the profit that the restaurant will get if they fulfill this customer's order.
- The fourth number is the number of turns that we need to prepare this customer's order.
- The last number is the maximum number of turns that this customer will wait for their order to be selected for preparation before leaving the restaurant.

The scenario file of the example that we described in Section 2 will be:

```
1   23215   13.00   4   8
1   58923   5.99    6   9
3   38623   11      8   3
```

6 Declaring Your Assignment Team

You may do this assignment alone or in a team of 2 students. Your teammate should be different from any other partners that you have had (or going to have) in the labs (tutorials) and Assignment 2. You must declare your team (even if you are working solo) using the MarkUs online system. Assignment and team declaration will be active from June 2. Teams are required to be declared from June 2nd to June 12th.

Navigate to the MarkUs page for the assignment and find "Group Information". If you are working solo, say so. If you are working with other(s):

First: one of you needs to "invite" the other to be partner, providing MarkUs with their cdf user name.

Second: the invited student must accept the invitation.

Important: there must be only one inviter, and the other group member accepts after being invited, if you want MarkUs to set up your group properly.

To accept an invitation, find "Group Information" on the appropriate Assignment page, find the invitation listed there, and click on "Join".

7 Submitting Your Work

Submit the following files on MarkUs by 4:30 p.m. June 17:

```
customer.py
restaurant.py
simulator.py
```

Click on the "Submissions" tab near the top. Click "Add a New File" and either type a file name or use the "Browse" button to choose one. Then click "Submit". You can submit a new version of a file later (before the deadline, of course); look in the "Replace" column.

Only one team member should submit the assignment. Because you declared your team, each of you will get the same credit for the work. Note that late submissions are not accepted.

8 Special Office Hours

There will be special office hours for this assignment on **June 10**, from **10 to 12** as well as from **2 to 4**, both in BA3201. This will be the best time to discuss your questions in person with the Assignment Lead TA. In particular, we will NOT answer questions asked one day before the deadline (in person or online). Start Early!