Assignment 5: Multi-Threading

Question - 1: Washing Machines

You have been asked by the student council to monitor the use of washing machines in OBH. You have to report the events of the day in the format provided by the council with some statistics. In your conclusion line, you have to report if more washing machines are needed or not. More washing machines are needed if at least 25% of the students who came to the machines returned without getting their clothes washed.

During the entire day:

- **N** students will come to get their clothes washed
- There are **M** functioning washing machines
- The time at which the i-th student comes is **T_i** seconds after the execution of the program
- The time taken to wash the i-th student's clothes is **W_i** seconds
- The patience of the i-th student is **P_i** seconds, after which he leaves without getting his clothes washed

Statistics-

- Number of students who left without washing. Integer between 0 and N
- Total number of seconds wasted. Seconds wasted by a student
  = Time at which he starts washing/leaves without washing - time of arrival

Given appropriate inputs, simulate the events using multi-threading. **Avoid deadlocks and busy waiting.** Instead, use semaphores and mutex locks to implement the problem. You are required to write your algorithm, followed by implementational details in your report.

Input format-

The first line contains 2 integers **N** and **M**.
The next **N** lines contain 3 integers **T_i**, **W_i** and **P_i**, the i-th line corresponding to the detail's of the i-th student

Output format-

A relevant line has to be printed in the given color for the following events, with the time at which it occurs:

Event	Colour

- A student arrives to get their clothes washed	(White)
- A student gets an empty washing machine to wash his clothes	(Green)
- A student leaves after his washing is complete	(Yellow)
- A student leaves without getting his clothes washed	(Red)

A single integer stating the number of students who came but could not wash their clothes.
A single integer stating the total number of seconds wasted by all students waiting.
A final line saying "Yes" if more washing machines are needed, or "No" if they are not needed. (in White)

Example 1-

Input-

```
5 2
6   3   5
3   4   3
6   5   2
2   9   6
8   5   2
```

Output-

```
2: Student 4 arrives
2: Student 4 starts washing
3: Student 2 arrives
3: Student 2 starts washing
6: Student 1 arrives
6: Student 3 arrives
7: Student 2 leaves after washing
7: Student 1 starts washing
8: Student 5 arrives
8: Student 3 leaves without washing
9: Student 1 leaves after washing
9: Student 5 starts washing
10: Student 5 leaves after washing
11: Student 4 leaves after washing
1
4
No
```

Explanation-

Time wasted by the students waiting:

Student 1 = 1 second   (7-6)
Student 2 = 0 seconds   (3-3)
Student 3 = 2 second   (8-6)
Student 4 = 0 seconds   (2-2)
Student 5 = 1 second   (9-8)

Total = 4 seconds

Number of students who left without washing = 1 (Student 3)
Percentage of students left without washing = 20% which is less than 25%
Therefore, no washing machines needed

Example 2-

Input:

```
2 1
2 5 1
1 2 4
2 4 2
```

Output-

```
1: Student 2 arrives
1: Student 2 starts washing
2: Student 1 arrives
2: Student 3 arrives
3: Student 2 leaves after washing
3: Student 1 starts washing
4: Student 3 leaves without washing
7: Student 1 leaves after washing
1
3
Yes
```

Points to Ponder-
- If there are multiple students waiting, then the student who arrived first will use the machine first (FCFS)
- If there are multiple students arriving at the same time, the one with the lower index will go first
- If a student arrives at time **t_i** and has patience **p_i**, he can use the machine at time (**t_i + p_i**) if it gets empty at that second
- If a student arrives at time **t_i** and has patience **p_i**, he will leave at time (**t_i + p_i**) if there are no machines empty at that second

Question - 2: Pizzeria Gino Sorbillo

Gino's pizza is known to be the best pizza in the planet. It was featured in various movies, and people travel miles to eat the pizza. It was also featured in Eat, Pray Love. Since Gino Sorbillo is expanding, the pizzeria offers **M** pizzas. **N** chefs are employed in the pizzeria. Unfortunately, these chefs have other part-time jobs and hence cannot reach the pizzeria at the same time. The restaurant is also considering a drive-thru option. Given the rectangular layout of the restaurant, the drive-thru can only support **K** cars. Any other car waiting outside the drive-thru zone is allotted into the drive-thru zone depending on availability and a first-come-first-serve basis. All the time scales provided in the question are in seconds. If an event **E** is occurring at 1 second, it means it has been 1 second since the restaurant that the event **E** is occurring 1 second after the restaurant opened. The restaurant layout is as follows -

- **Chefs:** A chef arrives at time **T1** and exits at time **T2.** Every chef in the restaurant has been employed on competent basis, and hence, all the chefs can prepare a specific kind of pizza in a stipulated amount of time. At any given time, if no chef is present in the restaurant, the restaurant is shut down. All customers wait if there are any incoming chefs. If no chefs are incoming, the customers are rejected. A chef can only be assigned an order if they have enough time to prepare the order.

- **Pizza: M** kinds of pizzas are available. Each kind of pizza has a different preparation time **t**. If the pizza requires limited ingredients, it uses a single unit of the limited ingredient.

- **Drive-Thru Customers: K** cars can be supported through the drive-thru option. The customer present in each of the cars can order multiple pizzas of different kinds. The customer orders the instant they reach the drive-thru. If the customer is rejected by the restaurant, the customer can choose to exit the drive-thru by overtaking other cars. If the pizza can be served, the customer needs to wait till they reach the pickup spot to collect their pizzas. It takes **s** seconds to reach from the entrance to the pick up spot. If their pizza is not present at the pick up spot, but their order has not been rejected, they wait at the pick up spot.

- **Ingredients:** Some ingredients like flour, oil and yeast are available in unlimited quantity. Some of them like vegetables and meat are only available in limited quantity for the day. The restaurant then stocks these ingredients again for the next day. If all the limited ingredients available for making all the pizzas are over, then the restaurant shuts down for the day. If the restaurant has the ingredients to make at least one pizza, the restaurant stays open, but the drive-thru orders become more selective. Drive-thru orders having at least one pizza which can be made by the pizzeria are accepted. Other drive-thru orders are rejected.

- **Ovens: L** ovens are present in the restaurant. An oven can only cook one pizza at a time. If the preparation time for a pizza is **t**, 3 seconds are taken by all the chefs to arrange the ingredients (if available), and once allotted an oven, the pizza is baked in **(t-3)** seconds and immediately sent to the pick up spot.

- **Pick up spot:** The pick up spot is massive and can high quantities of pizzas, so no restrictions here!

We want to simulate the functioning of the pizzeria. Consider the following events for each of the following entities -

1. Chef

- Chef c arrives at time **t1**.
- Chef c exits at time **t2.**
- Chef c could not complete pizza **p** for order **o** due to ingredient shortage.
- Chef c is preparing the pizza **p** for order **o**.
- Chef c is waiting for oven allocation for pizza **p** for order **o**.
- Chef c has put the pizza **p** for order **o** in the oven at time **t**.
- Chef c has picked up the pizza **p** for order **o** from the oven at time **t**.

2. Car/Customer (interchangeable)

- Customer c arrives at time **t1**.
- Customer c is waiting for the drive-thru allocation.
- Customer c enters the drive-thru zone and gives out their order **o**.
- Customer is rejected. (Note that a customer can be rejected even after being accepted. If this case happens, the customer is notified immediately. The customer exits the restaurant immediately).
- Customer c is waiting at the pickup spot.
- Customer c picks up their pizza i.
- Customer c exits the drive-thru zone.

3. Order

- Order **o** placed by customer **c** has pizzas **{p1,p2,…}**.
- Pizza **x** in order **O** has been assigned to Chef **i.**
- Order **o** placed by customer **c** awaits processing.
- Order **o** placed by customer **c** is being processed.
- Order **o** placed by customer **c** has been processed.

- Order **o** placed by customer **c** partially processed and remaining couldn't be.
- Order **o** placed by customer **c** completely rejected.

Given appropriate inputs, simulate the pizzeria using multi-threading. **Avoid deadlocks and busy waiting**. Instead, use semaphores and mutex locks to implement the problem. **You are required to write your algorithm, followed by implementational details in your report. Answer the follow-up questions:**

- The pick-up spot now has a stipulated amount of pizzas it can hold. If the pizzas are full, chefs route the pizzas to a secondary storage. How would you handle such a situation?

- Each incomplete order affects the ratings of the restaurant. Given the past histories of orders, how would you re-design your simulation to have lesser incomplete orders? Note that the rating of the restaurant is not affected if the order is rejected instantaneously on arrival.

- Ingredients can be replenished on calling the nearest supermarket. How would your drive-thru rejection / acceptance change based on this?

The input format is as follows-

- The first line consists of the number of chefs (n), the number of pizza varieties (m), the number of limited ingredients (i), the number of customers (c), the number of ovens (o), and the time for a customer to reach the pickup spot (k).

- The second line to (2+m) consists of the pizza entity ID followed by its preparation time and the number of limited ingredients they use and its ID.

- The (2+m+1) line consists of the ingredient amount of each of the limited ingredient.

- The (2+m+2) line consists of the entry-exit times of each of the chefs ordered in the same line. (chef1_entry, chef1_exit, chef2_entry, chef2_exit, …).

- Followed by this, the rest of the lines consist of the customer's entry time, followed by the number of pizzas they want to order followed by the pizza IDs.

A sample input is as follows-
<u>Sample Input-</u>

```
3 4 3 3 5 3
1 20 3 1 2 3         *pizza 1 can be prepared in 20 seconds and requires 3 special ingredients, namely 1, 2 and 3. *
2 30 2 2 3
3 30 0               *margarita pizza; but they prolly ran out of good cheese*
10 5 2               *ingredient 1 has 10 units, 2 has 5 units and 3 has 2 units*
0 50 20 60 30 120
0 2 1 2
```

```
1 1 1
2 2 1 2
4 1 3
```

A sample output for the above input is as follows-
Sample Output-

```
Simulation Stared.
Chef 1 arrives at time 0.
Customer 1 arrives at time 1.
Order 1 placed by customer 1 has pizzas {1}.
Order 1 placed by customer 1 awaits processing.
Pizza 1 in order 1 assigned to chef 1.
Order 1 placed by customer 1 is being processed.
Chef 1 is preparing the pizza 1 from order 1.
Customer 2 arrives at time 2.
Order 2 placed by customer 2 has pizzas {1, 2}.
Chef 1 has put the pizza 1 for order 1 in oven 1 at time 4.
Customer 3 arrives at time 4.
Order 3 placed by customer 3 has pizzas {3}.
Customer 3 rejected.
Customer 3 exits the drive-thru zone.
Chef 2 arrives at time 20.
Pizza 1 in order 2 assigned to chef 2.
Chef 2 is preparing the pizza 1 from order 2.
Chef 1 has picked up the pizza 1 for order 1 from the over at time 21.
Customer 1 picks up their pizza 1.
Customer 1 exits the drive-thru zone.
Pizza 2 in order 2 assigned to chef 1.
Chef 1 is preparing the pizza 2 from order 2.
Chef 2 has put the pizza 1 for order 2 in oven 1 at time 23.
Chef 1 has put the pizza 2 for order 2 in oven 2 at time 24.
Chef 3 enters at time 30.
Chef 1 has picked up the pizza 2 for order 2 in oven 1 at time 44.
Customer 2 picks up their pizza 2.
Chef 2 has picked up the pizza 1 for order 2 in oven 1 at time 53.
Customer 2 picks up their pizza 2.
Customer 2 exits the drive-thru.
Chef 1 exits at time 50.
Chef 2 exits at time 60.
Chef 3 exits at time 120.
Simulation Ended.
```

Points to ponder

- Outputs are sequentially deterministic. The order of outputs can vary depending on synchronisation.
- We insist that you print colour coded statements for actions specified in the entities section. Use yellow text for customers, red text for orders and blue text for chefs.
- When the simulation starts, print *Simulation Started.* When the simulation is over, print *Simulation Over.*
- The test case provided above is very basic. You are not advised to create large test cases. You are rather advised to create more complex test cases. For example, create test cases where ingredients are used up while the customer is

in the middle of the drive-thru. A chef might have been assigned a pizza, but by the time they attempt to prepare the pizza, ingredients might have run short. Handle such corner cases.
- Intuitively check if the order of sequences in your outputs occurring are correct.

Question - 3: Internet Routing

A routing table is a set of rules, often viewed in table format, that is used to determine where data packets traveling over an Internet Protocol (IP) network will be directed. In this question you will be simulating the process of generating and using a routing table.

You will be creating two programs for this purpose.

1. Server
   - The server program will simulate all the nodes in the network with threads. These threads should only communicate with each other through message passing via sockets.
   - Initially the node only knows the communication delay with the nodes it is directly connected with. You will have to generate the routing table for every node after communicating with each other node.
   - When any data is received, the node must print the source, destination, data received and the forwarded destination if the current node had forwarded it. Example : *Data received at node: 4 : Source : 0; Destination : 5; Fowarded_Destination : 3; Message : "Hello!";*

   Input Format-

   - The first line of the input consists of the number of nodes in the network **n** and the number of direct connections between them **m**. The next **m** lines consist of three numbers **a**, **b**, **d** where **d** is the communication delay between **a** and **b**.
   - A sample input has been given as follows-

   Sample Input-
   ```
   4 5
   0 1 15
   0 2 50
   1 2 15
   2 3 20
   0 3 40
   ```

2. Client

- The client program will provide a shell to interact with the network simulated by the server. This shell implements the following commands.
- **Print table: *pt*** *pt* prints the routing table of the node 0. The routing table consists of the destination node, delay and the node to which the packet needs to be forwarded to.

- The output of the *pt* command is as follows-
```
dest   forw   delay
1      1      15
2      1      30
3      3      40
```
- **Send message: *send*** *send* sends a message to node 0 which will be sent through the network to the destination node.
- The format of the *send* command is as follows-
  - > *send* 4 Ping!
- This message will be sent to node 0, which will be forwarded through the network till it reaches the destination node 4. All the nodes that receive this message need to print the information as specified in the server program.

Instructions-

- For the purpose of this question, a process **p** can only communicate to its neighbours. Consider the source as 0, and the destination as the node in the send command.
- All the servers, at time instant 0, only has information about its neighbours. Each of these servers maintain a routing table: the shortest delays to all the servers to which a message can be passed to, followed by which server a message must be forwarded to so as to reach the intended destination. At time instant 0, this can only be availed for immediate neighbours.
- Construct an algorithm to update these routing tables. Post a send command, the message arrives at node 0. This message must then be forwarded to the intended destination with the shortest possible delay. Invoke message-passing between servers to update their routing tables.
- You are given a choice either optimise your routing tables post a send command, or you can pre-optimise your routing tables before a batch of send commands.

Follow-up Questions-

- You are required to write your implementational details in the report.
- How would you handle server failures?

General Guidelines-

1. Using colouring in question 1 and question 2 is **mandatory**.
2. Any assumptions must be written in the report alongside with code explanation.
3. Use concise and relevant **comments** in your code.
4. You are advised to test your assignments on more qualitative test cases. The essence of this assignment is to understand working of semaphores using real life examples. Hence, test on special cases. Testing on huge datasets won't be performed.

Submission Instructions-

1. You are required to use the C programming language.
2. Submission format-

```
<roll_number>
        |__ q1/
                |__ q1.c
                |__ readme/report
        |__ q2/
                |__ q2.c
                |__ readme/report
        |__ q3/
                |__ client.c
                |__ server.c
                |__ readme/report
        |__ <any other necessary files>
```

3. Compress <roll_number> and submit it as <roll_number>_assign4.tar.gz.
4. Along with the testing, marks will also be allotted to the logic and implementation.
5. Plagiarism is strictly prohibited.