Assignment 4: Concurrency

Deadline: 14th October, 2020

Question 1: Concurrent Merge Sort

- Given a number n and n numbers, sort the numbers using Merge Sort.
- Recursively make two child processes, one for the left half, one for the right half. If the number of elements in the array for a process is less than 5, perform a selection sort.
- The parent of the two children then merges the result and returns back to the parent and so on. Compare the performance of the merge sort with a normal merge sort implementation. Print the time taken for each on the terminal. Report your findings in the README.

Bonus: Make a third implementation of merge sort using threads in place of processes for Concurrent Merge Sort. Add the performance comparison for this as well.

Question 2: Back to College

Risubrik is a fellow student of a IIIT-H and he is completely tired of the online semester and wants to head back to college as soon as possible. He wants to suggest the college to get all the students vaccinated using his huge internship money along with a foolproof plan to do so. Help him devise a plan that he can impress the college with. Here are the details:

Pharmaceutical Companies

- IIIT has partnered up with ‘n’ Pharmaceutical Companies that are producing a coronavirus vaccine.
- Each Pharmaceutical Company manufactures its own version of a ~~corona~~virus vaccine which has ‘x’ probability of successfully administering antibodies into the student.
- Each Pharmaceutical Company can prepare a random number (greater than or equal to 1) of coronavirus vaccine batches at a time.
- Each batch has a capacity of ‘p’ coronavirus vaccines.
- In order to save money, a pharmaceutical company will only start producing the next batch of vaccines after the current batch it has produced has been exhausted.

Wuhan

Vaccination Zones

- In order to maintain social distancing, IIIT has created ‘m’ secure Vaccination zones on campus with limited capacities.
- Each Vaccination Zone receives a batch of coronavirus vaccine from one of the pharmaceutical companies.

- As long as a Vaccination zone has enough vaccines left in the batch, it makes a random number of slots available for the students to get vaccinated. Note that the number of slots should always be less than or equal to the vaccines left in the batch. These students get vaccinated and once this over, you must update the number of vaccines left in the batch.
- There can only be one batch of ~~coronavirus~~ vaccine at each zone. A new batch of ~~coronavirus~~ vaccine will be delivered by a pharmaceutical company once the existing batch has been exhausted.

Students

- 'o' Students have registered for the vaccination drive.
- Since no IIIT student ever comes on time, different students become available for vaccination at various different times.
- Once a Student is vaccinated, he gets an antibody test to check if he has successfully received antibodies. If he tests negative (Did not receive antibodies) then he must re-enter the process immediately. If he tests positive, he can now attend college!

Implementation

Pharmaceutical Companies

- Each Pharmaceutical Company takes 'w' seconds (random between 2-5) to prepare 'r' batches (random between 1-5) of vaccine at any particular time. Each batch has a capacity to vaccinate 'p' students (random between 10-20).
- Each Vaccine has 'x' probability of successfully administering antibodies into the student which will be given to you as input.
- Once the Pharmaceutical Company is done preparing the batches, they signal to the vaccination zones that they have a batch ready. The pharmaceutical company does not resume manufacturing another batch until all the vaccines in the group of batches that it has manufactured has been used for vaccination on students(It does not resume straight after distributing to a vaccination zone).

Vaccination zones

- Each Vaccination zone is initially empty of students and vaccines. It waits for any of the Pharmaceutical Companies to send a batch of vaccine to the zone.
- Once the Vaccination zone receives a batch of coronavirus vaccine it can start vaccinating the students. At each iteration of vaccination(called a vaccination phase), the zone makes available 'k' slots. ('k' >=1 and k<=min(8,number of vaccines left in the

batch,number of waiting students)). Note that a student can be assigned a slot at any of the vaccination zones.
- While the Zone is in vaccination phase, it cannot allocate a free slot (if it has any) to the newly arriving students. The Zone can resume allocating slots only after it has finished its current vaccination phase.

Students

- Once a student arrives at the college gate, he/she signals that he is ready for vaccination. Once the student finds a vaccination zone with an empty slot, he heads over to the vaccination zone to get vaccinated.
- Once the student is in the slot, he/she will signal to the zone that he has arrived. Subsequently, he/she gets vaccinated.
- Once a Student is vaccinated, he gets an antibody test to check if he has successfully received antibodies. If he tests negative (Did not receive antibodies) then he must signal that he is ready for vaccination again immediately. If he tests positive, he can now attend college.
- Each Student can receive a maximum of 3 vaccinations, before the college gives up on him and he is sent home for another online semester.
- Once all the students have gone through the drive either successfully/unsuccessfully , you are done for the day.

Instructions

- Each Company, Vaccination Zone and Student ~~are threads~~ is a thread.
- Stop the simulation when all the students have gone through the drive.
- A Vaccination Zone is used multiple times for vaccinating. That means if there are still students left to be vaccinated, the Vaccination Zone should be on the lookout for waiting students.
- Use appropriate small delays for the simulation.
- Prepare a detailed report explaining your implementation and assumptions. Note that the report will be an important component of the grading.
- The use of semaphores is not allowed. You can use only one mutex lock per Pharmaceutical Company, Vaccination Zone and Student if required.
- Your simulation should allow for multiple students to arrive at a Vaccination zone simultaneously. Essentially, you cannot send students to the vaccination zone one-by-one (this defeats the point of multithreading) and your code should allow various students to reach the vaccination zone simultaneously.
- IIIT will not be happy if the code results in any deadlocks/infinite loops/segmentation faults and will ask Risubrik to pay his entire internship salary to the college. Ensure that your code **terminates successfully for all kinds of test cases.**
- You are allowed to code this question in whatever way seems fit to you. There isn't much restriction as long as the instructions aren't violated.

Input

First line will have ‘n’ pharmaceutical companies, ‘m’ vaccination zones, ‘o’ students (note: 0 <=n,m,o)
Second Line will have ‘n’ probabilities of the success rate of each vaccine from the n pharmaceutical companies.

Output

On the arrival of a student, print,

"Student X has arrived for his 1st/2nd/3rd round of Vaccination"

When the student is waiting to be allotted a vaccination zone, print,

"Student X is waiting to be allocated a slot on a Vaccination Zone"

When a Vaccination zone becomes ready to vaccinate, print.

"Vaccination Zone Y is ready to vaccinate with S slots", where S is the number of slots available in that zone for that particular round of vaccination.

When the student is assigned a slot on the Zone, print,

"Student X assigned a slot on the Vaccination Zone Y and waiting to be vaccinated"

When the Vaccination Zone begins vaccinating, print,

"Vaccination Zone Y entering Vaccination Phase"

When a student is vaccinated, print,

"Student X on Vaccination Zone Y has been vaccinated which has success probability Xj" Where Xj is the probability of that vaccine successfully administrating antibodies.

When a student does his antibody test, print,

"Student X has tested ‘positive/negative’ for antibodies."

When the batch at a Vaccination Zone becomes empty, print,

"Vaccination Zone Y has run out of vaccines”

When the Vaccines of a Vaccination Zone is refilled by a Pharmaceutical Company, print,

"Pharmaceutical Company J has delivered vaccines to Vaccination zone Y, resuming vaccinations now"

When a Pharmaceutical Company is preparing vaccine batches, print,

"Pharmaceutical Company J is preparing N batches of vaccines which have success probability Xj",
where N is the number of batches the Pharmaceutical Company is preparing for that particular round where Xj is the probability of that vaccine successfully administrating antibodies.

When a Pharmaceutical Company is done preparing N Vaccine Batches, print,

"Pharmaceutical Company J has prepared N batches of vaccines which have success probability Xj. Waiting for all the vaccines to be used to resume production"

When the Pharmaceutical Company is delivering a batch to a Vaccination Zone, print

"Pharmaceutical Company J is delivering a vaccine batch to Vaccination Zone Y which has success probability Xj""

When all the vaccines prepared by a Pharmaceutical Company are used, print,

"All the vaccines prepared by Pharmaceutical Company J are emptied. Resuming production now."

When the simulation is over, print,

"Simulation Over."

Note: You can always add more print statements but these are the minimum requirement.

Question 3: Musical Mayhem

The college is opening for the new semester, and the campus is overrun with excited first years and their parents. The Music Club has decided to host an event on the first day to entertain all the newcomers. You have been given the responsibility of managing the event. Here are the details:

1. There are stages set up all over the campus. They are of two types: acoustic and electric. There are **a** acoustic stages and **e** electric stages.

2. IIIT has a large number of talented musicians. Every musician plays one of the following instruments. Each type of instrument included in this event is either acoustic or electric, i.e. they are allowed to play on either the acoustic stage or the electric stage (or both).

Instrument Players	**Instrument Character**	**Stages eligible to perform**
Piano	p	Acoustic, Electric
Guitar	g	Acoustic, Electric
Violin	v	Acoustic
Bass	b	Electric
Singer* (special case)	s	Acoustic, Electric

3. **k** musicians and singers will arrive at Srujana at random times. They wait for a stage to become available. When they get a stage, the musicians will go to the stage and perform solo for a random duration t' such that t' >= **t1** sec and t' <= **t2** sec. At no time should a stage be empty if some valid musician is waiting at Srujana to perform.

4. **Singers** (s) are a special case. They can perform solo at both acoustic and electric stages for a random time t'. However, they also have the option of **joining a performing musician on the stage**. In such a case, the original time duration of the musician is extended by **two seconds**. At the end of the performance, both leave. **If both empty and occupied stages are available to the singer, either can be chosen.** No more than one singer can join a performer.

5. Musicians and singers are **impatient** individuals. They will wait for no more than **t** sec after they arrive to be given a chance to perform. If they do not get to play, they leave Srujana and don't return.

6. As a token of appreciation, the Music Club offers t-shirts to every musician who performed(**not singers**). The club has **c** coordinators who each take 2 seconds to check the t-shirt size and give it to the performer. Each coordinator can help only one person at a time. As soon as the musician's performance is over, they wait for one of the coordinators to become free so they can collect the t-shirt.

- Impatient musicians who left without performing do not get t-shirts.
- Musicians are not impatient about getting their reward, they can wait for however long after their performance to get the t-shirt :P
- Singers do not get t-shirts, they leave immediately

7. Musicians can take **7 statuses**: Not yet arrived, Waiting to perform, Performing solo, Performing with singer, Waiting for t-shirt, T-shirt being collected, Exited.
Singers can take **5 statuses**: Not yet arrived, Waiting to perform, Performing solo, Performing with musicians, Exited
Make sure your code handles all of these cases.

8. Some assumptions:
- The stage is pre-set. Ignore soundcheck time and time taken to travel to the stage. Once assigned, musicians and singers can instantly teleport there and begin performing. After the performance, they can instantly teleport to the coordinators to collect the t-shirt.
- Each instrumentalist can play only one type of instrument.
- The duration of each performance is not the same. It should be randomized every time.
- Two performances cannot happen at the same time on the same stage.
- Each musician can perform at most one time.
- No preference in allocating musicians, ie if one is waiting for a long time, and a new one comes, no need to explicitly ensure that the one waiting longer is given preference.
- There is no specific preference between acoustic and electric for pianists and guitarists. They will play wherever is available.

Assume all integer inputs, and wherever randomization is required, take it as an integer.
You **DO NOT** need to keep track of the individual stages.

Instructions:

1. You must use **semaphores** and **mutexes** to implement the problem.
2. According to the implementation in your logic, treat components as either threads or resources. Unlike Q2, there is no compulsion to make threads for each element, as long as your logic is valid.
3. If some musicians arrive at the same time, you cannot take care of their cases one-by-one. Your code should take care of the simultaneous arrival.
4. Your simulation must not result in deadlocks. Use mutex locks appropriately.
5. You are NOT permitted to use busy-waiting.

Bonuses:

1. Singers also collect t-shirts after their performance.
2. Keep track of which stage number is joined by each musician/singer.

You will need to print related output for both of these in case you implement it.

Your program:

Input:

First line will have k, a, e, c, t1, t2, t (note: 0 <= k, a, e, c, t1, t2, t and t1 <= t2)
The following k lines will have info of each instrumentalist. Each line is of the form:
<name> <instrument character> <time in secs of arrival>

Output:

You need to print relevant output statements whenever:

1. Instrumentalist arrives at Srujana.
   -> Show **the name** of who arrives and **the instrument** they play.
2. Performance starts by a musician
   -> show **name** of musician, **stage type**, **instrument** and for what **duration**.
3. Performance ends.
   -> show **name** of musician, and **which stage** type.
4. T-shirt collecting process begins. (not when starts waiting, but when gets to collect)
   -> show **name** of musician
4. Musician leaves Srujana due to impatience.
   -> show **name** of musician, and **instrument**.
5. Solo performance starts by singer
   -> show **name** of singer, **stage type**, and for what **duration**.
6. Singers joining a performance
   -> show **name** of singer, and **name** of musician they joined

You can also print any other relevant logs you wish to print, as long as it makes sense.

Once the simulation is done, **print 'Finished'** before exiting.

Sample 1:

Input:
4 1 1 2 10 5
Tanvi p 0
Sudh b 1
Manas g 0
Sriya v 1
Pahun s 1

Output:
Tanvi p arrived -> at t~0
Manas g arrived -> at t~0
Tanvi performing p at acoustic stage for 9 sec -> at t~0
Manas performing g at electric stage for 2 sec -> at t~0
Sudh b arrived -> at t~1
Sriya v arrived -> at t~1

Pahun s arived	-> at t~1
Pahun joined Tanvi's performance, performance extended by 2 secs	-> at t~1
Manas performance at electric stage finished	-> at t~2
Manas collecting t-shirt	-> at t~2
Sudh performing b at electric stage for 5 sec	-> at t~2
Sriya v left because of impatience	-> at t~6
Sudh performance at electric stage ended	-> at t~7
Sudh collecting t-shirt	-> at t~7
Tanvi performance at acoustic stage ended	-> at t~11
Tanvi collecting t-shirt	-> at t~11
Finished	-> at t~13

Sample 2:
Input:
1 1 1 2 7
Abhinav d 3

Output:

Abhinav d arrived	-> at t~3
Abhinav performing d at electric stage for 7 secs	-> at t~3
Abhinav performance at electric stage ended	-> at t~10
Abhinav collecting t-shirt	-> at t~10
Finished	-> at t~12

Note:
1. DO NOT print "at t~1" etc shown in the output sample. **It is only included here for the explanation.**
2. Your sentence structure of the output format need not exactly match what is given. Feel free to restructure the sentence as per your preference **as long as each line conveys what is required**. (Eg: you can expand t->tabla, etc)
3. Do not play around with the input structure, it should be exactly as specified.
4. Since this is a concurrency problem, if your output on some of the test cases does not match what we have given, it may still be accepted as long as it matches the constraints of the problem statement.

General Guidelines:

1. You are **highly encouraged to use coloring** while printing your outputs. Check out this link. This will not only make your debugging life easier but will also make it easy for the TA's to understand your code so you can be graded faster. Less hassle for both of us :)
2. Use concise, relevant **comments** in your code to mark the purpose of variables and conditional statements. This will make the reading of the code easier.
3. If you have made **assumptions** regarding the problem statement, clearly state them in the README. As long as it is a reasonable assumption, there will be no issues.

Submission:

- You are required to code every question in C.
- Libraries allowed are pthread.h and semaphore.h
- You have to submit a separate README (md/txt/pdf) for each question in which you should give a **clear and comprehensive** description of your logic for each question. You may include relevant code snippets to illustrate your logic, but it is not necessary. We will be evaluating this.
- Submission format: Include all files in a folder named with your roll number

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

- Compress <roll_number> and submit it as <roll_number>_assign4.tar.gz
- Plagiarism is strictly prohibited. Even if your code does not work perfectly, the emphasis is on development of a sound logic. If there is the slightest doubt that you are not clear about your implementation logic, you will be heavily penalized.