May 10th, 2018

CSE344 – System Programming - Homework #5

Submission deadline: 23:55, May 27th, 2018

Objective

The objective of this homework is to develop a multithreaded application using pthreads, condition variables and mutexes for flower delivery. Clients will make requests for flowers, a central thread will collect those requests, and then delegate the work to the closest florist that has the kind of flower requested.

How

You are provided with a file containing the list of florists and the type of flowers that they sell (they have an infinite amount of each).

e.g.
Ayse (10,25; 1.5) : orchid, rose, violet

means the florist Ayse sells the three mentioned types of flowers, and her shop is located at location (10,25) in a cartesian 2D grid and her deliveries are made with an average speed of 1.5 clicks per ms.

The file also contains the requests made by clients:

e.g.
client1 (55,76): orchid

means that the client1 located at location (55,76) has made a request for an orchid.

Your central thread will have a pool of one thread for every florist. Then it will start processing the requests one by one. For each request, it will delegate it to the closest florist with respect to the client using the Euclidean distance.

The florist admitting the request will prepare it and deliver it by printing on screen how long it took it in total to deliver it (time of preparation + time of delivery; use the speed and distance information). The preparation of every order at the florist requires a random amount of time between 10 to 50 ms (closed interval). You can simulate it by letting the thread sleep for a random duration (equal to the time of preparation + time of delivery).

Important: the central thread must process the requests as fast as possible, that means it should delegate the request to the closest florist even if she/he is busy preparing/delivering a request in the meantime (hint: use a request queue for each florist).

Once all requests have been processed, all florist threads should terminate and return their sale statistics (i.e. how much time each florist spent in total for preparing and delivering the requests) to the central thread which will then print on screen that information in a nicely formatted table along with the number of requests handled by each client.

If everything goes well your application should print on screen something similar to the following (the numbers are made up):

```
$floristApp data.dat
Florist application initializing from file: data.dat
3 florists have been created
```

```
Processing requests
Florist Fatma has delivered a clove to client2 in 56ms
Florist Murat has delivered a rose to client0 in 145ms
Florist Murat has delivered an orchid to client1 in 77ms
…
All requests processed.
Ayse closing shop.
Murat closing shop
Fatma closing shop.
Sale statistics for today:
-------------------------------------------------
Florist           # of sales        Total time
-------------------------------------------------
Ayse              7                 1543ms
Fatma             3                 750ms
Murat             6                 1343ms
-------------------------------------------------
```

Evaluation

Your code will be evaluated with a different data.dat file containing the same florists but different requests.

Rules

a) Your program must handle eventual errors gracefully according to the POSIX traditions.
b) Your program must accept the data file path as a commandline argument; and print usage information if it's missing.
c) Valgrind rule from previous homework still holds and will be hold on every homework.
d) *Your program MUST not give "segmentation fault" or any other fault that causes your program to end in a bad way. It is better for you to send a little part of the homework with clean and executing code instead of full homework with faults like this.*
e) **Provide a makefile to compile your homework. Do not run your program inside makefile. Just compile it.**
f) Test your homework using the Virtual Machine given to you.
g) No late submissions.
f) **Do not send any additional files like screenshots. Just send the makefile and source files.**
h) Taking any code from internet will result getting a grade of -100. Do no put links or references to internet, you don't need code from internet to do this homework. Man pages and your book is well enough.
j) Use POSIX and Standard C libraries. You can write your program using C11 standards.

Homework format:

StudentID_HW5_CSE344.tar.gz
|→ Makefile
|→ StudentID_main.c
|→ … (Any other source **files**, not directories!)

Good luck.