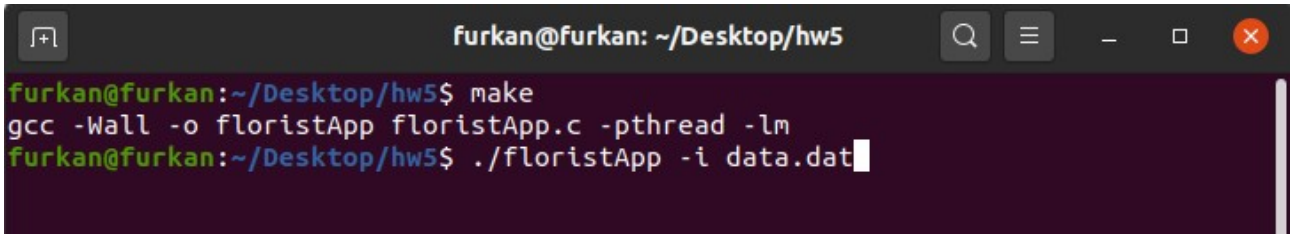
CSE344 – System Programming
REPORT - HW 5

Furkan ÖZEV
161044036

COMPILE & RUN:



MAIN IDEA:

→ 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.
→ These requests will be read from the file with some information in the specified format.
→ When determining the nearest florist, Chebyshev distance will be taken as basis.
→ The main thread that reads the requests from the file, the florists that will provide these requests are separate threads.
→ So, main thread will have a pool of one thread for every florist.
→ The central thread must process the requests as fast as possible. So, there will be a request queue for each florist and will add main thread requests to this queue.
→ Synchronization between threads will be provided with posix mutex.
→ All allocated memories will be deallocated when the program ends.
→ In case of CTRL-C program (and all its threads) shuts down gracefully, by deallocating all resources and printing an informative message.

MAIN THREAD FLOW:

→ The data file in the appropriate format given as argument opens.
→ All florists and clients to be created will be stored in dynamic struct arrays.
→ The information of the florists is read line by line.
→ For each new florist, the information in this line is sent to the fillFlorist () function.
→ This function creates a florist struct object, parses the information on the line and assigns it to the variables in that struct.
→ Florist name, coordinate x, coordinate y, speed, flower type amount, flowers, total sales and total time information is kept in this florist struct.
→ After all the florists are read from the file, the total number of florists is determined.
→ Queue is created for each florist.

→ The mutex to be used for each florist is initialized. And their initial state starts locked.
→ It is created in mutexes to be used for other synchronizations.
→ A thread is created for each florist.
→ From this moment, florist threads have been created and requests will be processed.
→ The information of the clients is read line by line.
→ For each new client, the information in this line is sent to the fillClient () function.
→ This function creates a client struct object, parses the information on the line and assigns it to the variables in that struct.
→ Client name, coordinate x, coordinate y, flowers and distance information is kept in this florist struct.
→ Chebyshev distances between this client and florists are calculated and stored in the array.
→ The closest florist with the flower desired by the client is determined.
→ The desired flower is added to the determined florist's request queue and the florist's mutex is unlocked.
→ The main thread uses one global condition variable to read all requests and directed them to the florists to indicate that it is done.
→ It then unlocks the mutexes of these threads to prevent deadlocks.
→ Then, join for each thread.
→ When all threads are finished, sales statistics are printed.
→ All allocated memory is deallocated and the file is closed.
→ The program ends successfully.

FLORIST THREADS FLOW:

→ If the main thread has not ended or the request list is not empty, it can already be a request or receive a new request.
→ The florist waits for its mutex to be unlocked.
→ This mutex is unlocked when there is a request in queue and locked when there is no request in queue.
→ If the main thread is terminated and there is no request left in the queue, the florist thread will end.
→ If the queue has a request, it will get the request.
→ The time of preparation and delivering the request is calculated and waits for this time using the sleep () function.
→ Preparation time is randomly between [1,250].
→ Delivering time is calculated by dividing the distance by speed.
→ The florist is delivered and makes the necessary printing.
→ If florist have an request in queue, unlock mutex.
→ Otherwise, the florist waits for the main thread to unlock this mutex.
→ When all requests are delivered, different mutexes are used to report main thread.
→ The main thread print requests are processed, then the florists will print that they close the shop.

→ The threads of florists end.
→ After all of these florist threads are finished, main thread prints the sales statistics.

FLORIST THREADS FLOW:

→ The SIGINT signal is handlered in the sigintHandler () function.
→ So when CTRL-C is done, this function works directly.
→ First, it empties all requests in the queues of florists.
→ It then unlocks all mutexes.
→ It then ends the threads with the pthread_cancel () and pthread_join() functions using the ids of the threads created.
→ All allocated memories are deallocated.
→ An informative message is printed and the program is successfully terminated.

RUN EXAMPLE :

INPUT FILE:

data.dat

```
Ayse (10,25; 1.5) : orchid, rose, violet
Fatma (-10,-15; 1.3) : clove, rose, daffodil
Murat (-10,8; 1.1) : violet, daffodil, orchid

client1 (0,4): orchid
client2 (1,5): clove
client3 (2,10): daffodil
client4 (4,15): orchid
client5 (8,-21): violet
client6 (-1,21): orchid
client7 (-6,20): rose
client8 (-16,18): rose
client9 (-12,-3): rose
client10 (23,0): violet
client11 (5,1): orchid
client12 (7,-8): violet
client13 (8,-3): clove
client14 (9,8): orchid
client15 (6,5): orchid
client16 (2,6): clove
client17 (-6,-4): daffodil
client18 (-9,-6): daffodil
client19 (-4,16): rose
client20 (-9,26): orchid
client21 (-4,-12): daffodil
client22 (9,13): rose
client23 (12,18): rose
client24 (11,15): orchid
```

NORMAL EXECUTION OUTPUT:

```
furkan@furkan: ~/Desktop/hw5
furkan@furkan:~/Desktop/hw5$ make
gcc -Wall -o floristApp floristApp.c -pthread -lm
furkan@furkan:~/Desktop/hw5$ ./floristApp -i data.dat
Florist application initializing from file: data.dat
3 florists have been created
Processing requests
Florist Ayse has delivered a orchid to client4 in 134ms
Florist Fatma has delivered a clove to client2 in 143ms
Florist Murat has delivered a orchid to client1 in 242ms
Florist Murat has delivered a daffodil to client3 in 11ms
Florist Ayse has delivered a orchid to client6 in 216ms
Florist Fatma has delivered a rose to client9 in 212ms
Florist Murat has delivered a violet to client5 in 111ms
Florist Murat has delivered a orchid to client11 in 22ms
Florist Ayse has delivered a rose to client7 in 65ms
Florist Ayse has delivered a rose to client8 in 51ms
Florist Fatma has delivered a clove to client13 in 142ms
Florist Ayse has delivered a violet to client10 in 46ms
Florist Murat has delivered a violet to client12 in 237ms
Florist Ayse has delivered a orchid to client14 in 209ms
Florist Fatma has delivered a clove to client16 in 236ms
Florist Fatma has delivered a daffodil to client17 in 96ms
Florist Murat has delivered a orchid to client15 in 256ms
Florist Ayse has delivered a rose to client19 in 192ms
Florist Murat has delivered a orchid to client20 in 77ms
Florist Ayse has delivered a rose to client22 in 132ms
Florist Fatma has delivered a daffodil to client18 in 229ms
Florist Ayse has delivered a rose to client23 in 51ms
Florist Fatma has delivered a daffodil to client21 in 58ms
Florist Ayse has delivered a orchid to client24 in 250ms
All requests processed.
Murat closing shop.
Fatma closing shop.
Ayse closing shop.
Sale statistics for today:
-------------------------------------------------
Florist          # of sales      Total time
-------------------------------------------------
Ayse             10              1346ms
Fatma            7               1116ms
Murat            7               956ms
-------------------------------------------------
furkan@furkan:~/Desktop/hw5$
```

CASE CONTROL-C OUTPUT:

```
furkan@furkan: ~/Desktop/hw5
furkan@furkan:~/Desktop/hw5$ ./floristApp -i data.dat
Florist application initializing from file: data.dat
3 florists have been created
Processing requests
Florist Murat has delivered a orchid to client1 in 71ms
Florist Fatma has delivered a clove to client2 in 106ms
Florist Ayse has delivered a orchid to client4 in 190ms
Florist Murat has delivered a daffodil to client3 in 209ms
Florist Fatma has delivered a rose to client9 in 193ms
Florist Ayse has delivered a orchid to client6 in 132ms
Florist Murat has delivered a violet to client5 in 234ms
Florist Fatma has delivered a clove to client13 in 219ms
Florist Ayse has delivered a rose to client7 in 201ms
Florist Murat has delivered a orchid to client11 in 126ms
Florist Ayse has delivered a rose to client8 in 182ms
Florist Murat has delivered a violet to client12 in 100ms
Florist Fatma has delivered a clove to client16 in 222ms
^C
Case CTRL-C: The program was terminated by deallocating all the resources.

furkan@furkan:~/Desktop/hw5$
```

VALGRIND RESULT :

Valgrind is a programming tool for memory debugging, memory leak detection, and profiling.

NORMAL EXECUTION:

```
furkan@furkan:~/Desktop/hw5$ valgrind ./floristApp -i data.dat
==11626== Memcheck, a memory error detector
==11626== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11626== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==11626== Command: ./floristApp -i data.dat
==11626==
Florist application initializing from file: data.dat
3 florists have been created
Processing requests
Florist Ayse has delivered a orchid to client4 in 47ms
Florist Ayse has delivered a orchid to client6 in 135ms
Florist Fatma has delivered a clove to client2 in 245ms
Florist Murat has delivered a orchid to client1 in 254ms
Florist Fatma has delivered a rose to client9 in 21ms
Florist Fatma has delivered a clove to client13 in 80ms
Florist Ayse has delivered a rose to client7 in 200ms
Florist Fatma has delivered a clove to client16 in 55ms
Florist Murat has delivered a daffodil to client3 in 208ms
Florist Ayse has delivered a rose to client8 in 206ms
Florist Fatma has delivered a daffodil to client17 in 202ms
Florist Ayse has delivered a violet to client10 in 21ms
Florist Fatma has delivered a daffodil to client18 in 88ms
Florist Ayse has delivered a orchid to client14 in 84ms
Florist Murat has delivered a violet to client5 in 236ms
Florist Fatma has delivered a daffodil to client21 in 21ms
Florist Ayse has delivered a rose to client19 in 35ms
Florist Ayse has delivered a rose to client22 in 82ms
Florist Murat has delivered a orchid to client11 in 146ms
Florist Ayse has delivered a rose to client23 in 108ms
Florist Murat has delivered a violet to client12 in 146ms
Florist Ayse has delivered a orchid to client24 in 175ms
Florist Murat has delivered a orchid to client15 in 149ms
Florist Murat has delivered a orchid to client20 in 232ms
All requests processed.
Fatma closing shop.
Ayse closing shop.
Murat closing shop.
Sale statistics for today:
-------------------------------------------------
Florist          # of sales      Total time
-------------------------------------------------
Ayse             10              1093ms
Fatma            7               712ms
Murat            7               1371ms
-------------------------------------------------
==11626==
==11626== HEAP SUMMARY:
==11626==     in use at exit: 0 bytes in 0 blocks
==11626==   total heap usage: 161 allocs, 161 frees, 9,835 bytes allocated
==11626==
==11626== All heap blocks were freed -- no leaks are possible
==11626==
==11626== For lists of detected and suppressed errors, rerun with: -s
==11626== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
furkan@furkan:~/Desktop/hw5$
```

CASE CONTROL-C:

```
furkan@furkan:~/Desktop/hw5$ valgrind ./floristApp -i data.dat
==11766== Memcheck, a memory error detector
==11766== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==11766== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==11766== Command: ./floristApp -i data.dat
==11766==
Florist application initializing from file: data.dat
3 florists have been created
Processing requests
Florist Murat has delivered a orchid to client1 in 72ms
Florist Fatma has delivered a clove to client2 in 106ms
Florist Ayse has delivered a orchid to client4 in 133ms
Florist Fatma has delivered a rose to client9 in 57ms
Florist Ayse has delivered a orchid to client6 in 91ms
Florist Murat has delivered a daffodil to client3 in 170ms
Florist Ayse has delivered a rose to client7 in 28ms
Florist Fatma has delivered a clove to client13 in 107ms
Florist Ayse has delivered a rose to client8 in 129ms
^CFlorist Fatma has delivered a clove to client16 in 165ms
Fatma closing shop.

Case CTRL-C: The program was terminated by deallocating all the resources.

==11766==
==11766== HEAP SUMMARY:
==11766==     in use at exit: 0 bytes in 0 blocks
==11766==   total heap usage: 166 allocs, 166 frees, 11,545 bytes allocated
==11766==
==11766== All heap blocks were freed -- no leaks are possible
==11766==
==11766== For lists of detected and suppressed errors, rerun with: -s
==11766== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
furkan@furkan:~/Desktop/hw5$
```

NOTICED:

1- I solved any and all synchronization issues using only mutexes and/or condition variables.

2- If the command line arguments are missing/invalid my program will print usage information and exit.

3- I assumed the file is not empty and that its contents have the proper expected format.

4- Each thread remove allocated resources explicitly.

5- I prevent the zombie process with using pthred_join.

6- In case of CTRL-C program (and all its threads) shuts down gracefully, by deallocating all resources and printing an informative message.

7- I don’t use busy waiting of any kind, don’t use timed waiting, or trylock.