SYSTEMS PROGRAMMING

#HW4

jwan hussein
151044078

used semaphores :

sem_t mutex1 ; /* the helper thread wait for, and posted by wholesaler when new line is read */
sem_t mutex2 ; /* the wholesaler thread wait for , posted by chef when dessert is delivered */
sem_t mutexMF; /* the chef who has endless supply of (W S) wait for , posted by helper when line read by wholesaler is MF (chef[5] wait for) */
sem_t mutexMS; /* the chef who has endless supply of (W F) wait for , posted by helper when line read by wholesaler is MS (chef[4] wait for) */
sem_t mutexMW; /* the chef who has endless supply of (S F) wait for , posted by helper when line read by wholesaler is MW (chef[3] wait for) */
sem_t mutexFS; /* the chef who has endless supply of (W M) wait for , posted by helper when line read by wholesaler is FS (chef[2] wait for) */
sem_t mutexFW; /* the chef who has endless supply of (S M) wait for , posted by helper when line read by wholesaler is FW (chef[1] wait for) */
sem_t mutexSW; /* the chef who has endless supply of (M F) wait for , posted by helper when line read by wholesaler is SW (chef[0] wait for) */
sem_t F; /* flour mutex */
sem_t M; /* milk mutex */
sem_t S; /* sugar mutex*/
sem_t W; /* walnuts mutex */

threads :

the total number of threads used to solve the problem is 8
the main thread
the helper thread
the chefs threads and there is six of them
pthread_t chefsth[6] ; /* chef[0] endless supply of M F
chef[1] endless supply of M S
chef[2] endless supply of M W
chef[3] endless supply of F S
chef[4] endless supply of F W
chef[5] endless supply of W S
*/

mian thread :

the main thread first opens the input file then checks if the given file is valid ,
then initializing all semaphores with initial value of zero and then start to create all other threads (helper thread an the chefs threads).
the main thread then starts to read the file line by line and store the read line in a global char array so other threads functions can see the line , after the main thread read a line it posts(mutex1)which is the mutex that the helper thread waits for, then posts two of the following mutexes (F,M,S,W) according to the letters of the line , and waits for (mutex2) which is the mutex that is being posted by a chef after (s)he deliver the dessert .
When the file contains no more line to read , the main thread cancels all other threads then calls pthread_join for all the threads , close the file , destroy the semaphores and exits .

helper thread :

in an infinite loop
first wait for (mutex1) to be posted by main thread which means that a new line was read then it post one of the following mutex (mutexMF,mutexMS,mutexMW,mutexFS,mutexFW,mutexSW) according to the letters of the line

pseudo code :

```
while(true){
        sem_wait(mutex1);
        if(line == “MF” || line == “FM” ){
                sem_post(mutexFM)
        }
        else if(line == “MS” || line == “SM” ){
                sem_post(mutexMS)
        }
        else if(line == “SW” || line == “WS” ){
                sem_post(mutexSW)
        }
        else if(line == “MW” || line == “WM” ){
                sem_post(mutexMW)
        }
        else if(line == “SF” || line == “FS” ){
                sem_post(mutexFS)
        }
        else if(line == “WF” || line == “FW” ){
                sem_post(mutexFW)
        }
```

```
}
```

chefs thread :

chef 0 is the chef who has endless supply of M and F waits for (mutexSW) and (S) and (W) mutexes
chef 1 is the chef who has endless supply of M and S waits for (mutexFW) and (F) and (W) mutexes
chef 2 is the chef who has endless supply of M and W waits for (mutexFS) and (F) and (S) mutexes
chef 3 is the chef who has endless supply of F and S waits for (mutexMW) and (W) and (M) mutexes
chef 4 is the chef who has endless supply of F and W waits for (mutexMS) and (M) and (S) mutexes
chef 5 is the chef who has endless supply of W and S waits for (mutexMF) and (M) and (F) mutexes

then posts the (mutex2) mutex so the main thread continue
sleep() is only used to simulate dessert preparation .

pseudo code :

```
int I = the id of the chef
while(true){
        if (I == 0){
                sem_wait(mutexSW)
                sem_wait(S)
                sem_wait(W)
        }
        else if (I == 1){
                sem_wait(mutexFW)
                sem_wait(F)
                sem_wait(W)
        }
        else if (I == 2){
                 sem_wait(mutexFS)
                sem_wait(F)
                sem_wait(S)
        }
        else if (I == 3){
                sem_wait(mutexMW)
                sem_wait(M)
                sem_wait(W)
        }
        else if (I == 4){
                sem_wait(mutexMS)
                sem_wait(S)
                sem_wait(M)
        }
        else if (I == 5){
                sem_wait(mutexMF)
                sem_wait(F)
                sem_wait(M)
        }

        sem_post(mutex2)
}
```

parsing command line arguments :

if any wrong argument was found in the command line of a program the program is terminated immediately printing usage information .

getopt() function was used to parse the command line .