CSE 344 – SYSTEM PROGRAMMING

HW3 - SEMAPHORES

151044072 – ALPER YAŞAR

Purpose of this Project?

Synchronization between processes using semaphores. There are 6 processes for chefs and 1 process for wholesaler. Wholesaler is the mother process and chefs are sub-processes. Every chef has an endless supply of two distinct ingredients and lacks the remaining two. Wholesaler delivering 2 ingredients in each visit, and the chef, who needs these 2 ingredients, taking these ingredients and making the dessert, and giving the wholesaler for sale.

Program calling and arguments

In this project there are 2 programs. First program for using unnamed semaphores. Calling like "`./hw3unnamed -i inputFilePath`" .

Second program for using named semaphores. Calling like "`./hw3named -i inputFilePath -n name`".

Programs little difference from each other.

In each program, firstly reading arguments. In named program there must 6 arguments. There are 2 semaphore so I need to 2 name. "`./hw3named -i inputFilePath -n /chef /wholesaler`".

In named program there must 3 arguments "`./hw3named -i inputFilePath -n /chef /wholesaler`".

Otherwise 2 program giving errors and exiting.

Signal

I used sigaction for SIGINT. This not mentioned in the homework but I also added. If signal handled in mother process calling "wholesaler()" function then waiting child process to termination and closing files and removing memory.

I also used this function for exiting when files end.

Chefs

Chef is structure and assigning which chef what has and what needs. Then it is using when process created. When process created in for loop calling "street()" function with index. So in this function running with just 1 chef for 1 process.

Shared memory

Opening a shared address for communication between processes. In unnamed program also has 7 semaphore for using communication. When open the shared memory address initialization the 7 unnamed semaphores. If there is any error when sem_init then giving error and exiting.

6 semaphores for chefs. Wholesaler reading line from file and posting first chef. If chef need these ingredients take it or posting to next chef and so on until 6. chef.

In named function semaphore names creating from arguments. Adding integer to each string and incremented.

In named program there is no semaphore in structure. Semaphores out of structure and using sem_open for initialization. Names coming from arguments.

I used 4 variable in structure.

Exit is for children. When wholesaler job is done then making exit variable as 1 and children terminating their jobs. waitToWrite is for wait wholesaler. Each child writing initial status and increasing the variable. When waitToWrite reach to 6 then post wholesaler process for progress.

And desserts for holding how many desserts are done.

Ingredients is for holding the ingredients are reading from file. Each line assigning this variable and child take this variable. Comparing with his/her what needs.

Checking file

Checking the file is it appropriate.

Then creating each process. And calling street(index) function. Index for which chef is running on this function.

Street(int index)

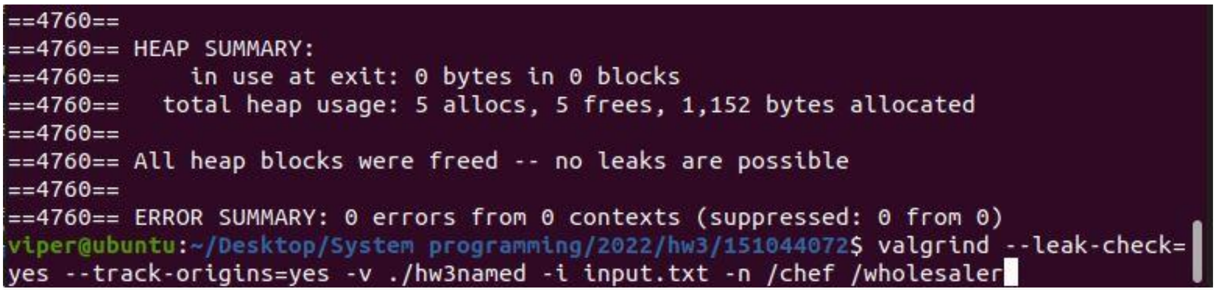
Index coming from wholesaler process. Firstly, writing which chef what needs. Then increment waitToWrite. Until all chefs printed, the wholesaler will wait for read the file. When all chef print their status then wholesaler reading each line from file and adding to array in shared memory. Then post the chef semaphore.

In street function there is endless loop but it is waiting for chef semaphore. When chef semaphore posted. Each chef checking and comparing arrays. Their needs ingredients holding in chef structure and the ingredients are taking by wholesaler is holding in shared memory array. After the comparison the correct chef making the deserts. Everything writing to screen each layer. After the desert finish then posting to wholesaler for selling.

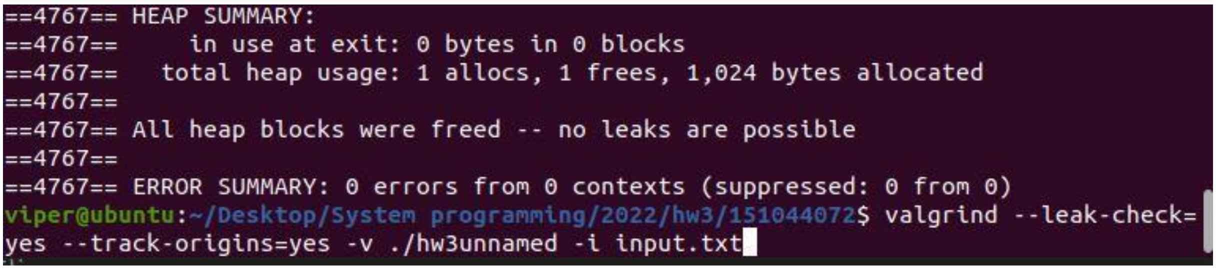
This is running until the file ends.

After the file ends, the wholesaler sending info to chefs with shared memory and chefs terminating. Chefs returning their total values and wholesaler collecting them. Wholesaler waiting their termination and then closing files and removing memories and exiting from system.

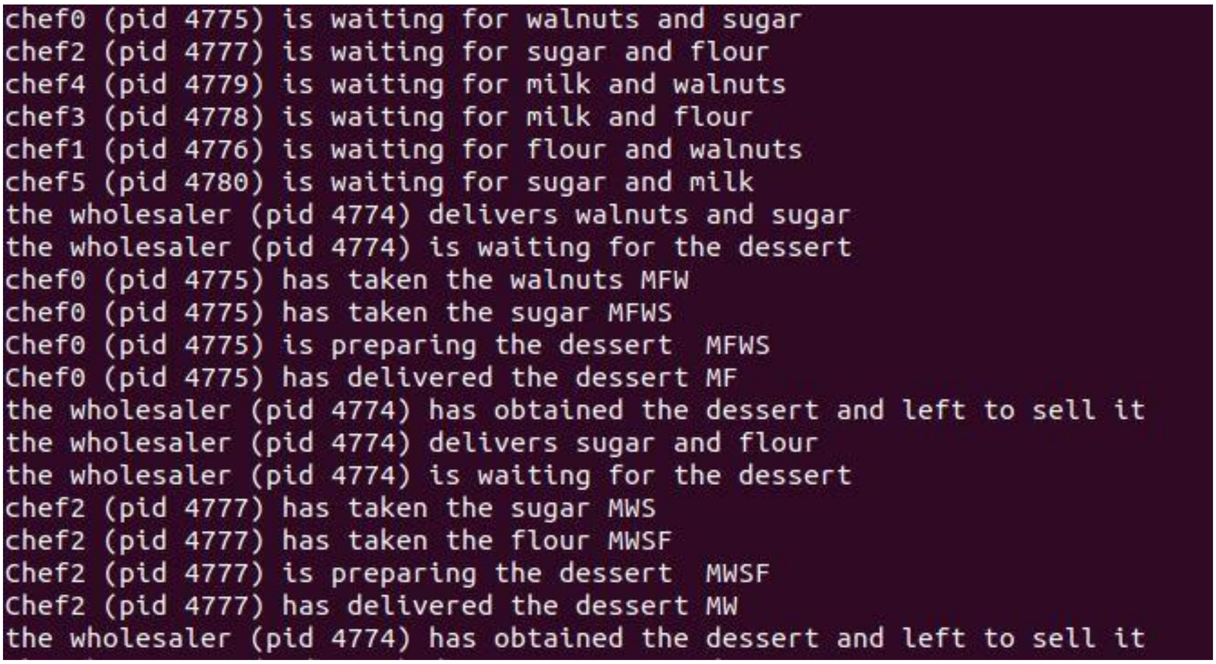
The wholesaler or any other process knows which chef needs which ingredients. When wholesaler take the ingrediens just adding them to array and posting the semaphore.



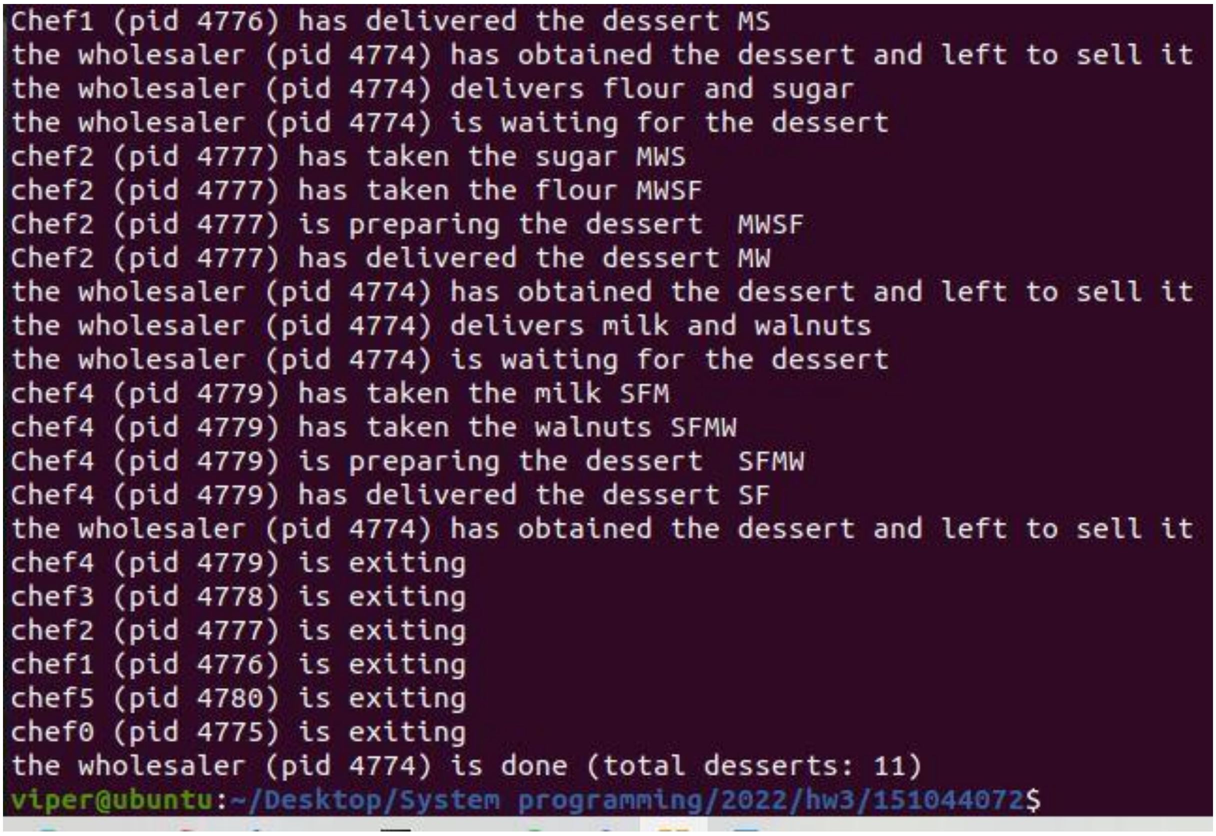
```
==4760==
==4760== HEAP SUMMARY:
==4760==     in use at exit: 0 bytes in 0 blocks
==4760==   total heap usage: 5 allocs, 5 frees, 1,152 bytes allocated
==4760==
==4760== All heap blocks were freed -- no leaks are possible
==4760==
==4760== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
viper@ubuntu:~/Desktop/System programming/2022/hw3/151044072$ valgrind --leak-check=
yes --track-origins=yes -v ./hw3named -i input.txt -n /chef /wholesaler
```



```
==4767== HEAP SUMMARY:
==4767==     in use at exit: 0 bytes in 0 blocks
==4767==   total heap usage: 1 allocs, 1 frees, 1,024 bytes allocated
==4767==
==4767== All heap blocks were freed -- no leaks are possible
==4767==
==4767== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
viper@ubuntu:~/Desktop/System programming/2022/hw3/151044072$ valgrind --leak-check=
yes --track-origins=yes -v ./hw3unnamed -i input.txt
```



```
chef0 (pid 4775) is waiting for walnuts and sugar
chef2 (pid 4777) is waiting for sugar and flour
chef4 (pid 4779) is waiting for milk and walnuts
chef3 (pid 4778) is waiting for milk and flour
chef1 (pid 4776) is waiting for flour and walnuts
chef5 (pid 4780) is waiting for sugar and milk
the wholesaler (pid 4774) delivers walnuts and sugar
the wholesaler (pid 4774) is waiting for the dessert
chef0 (pid 4775) has taken the walnuts MFW
chef0 (pid 4775) has taken the sugar MFWS
Chef0 (pid 4775) is preparing the dessert  MFWS
Chef0 (pid 4775) has delivered the dessert MF
the wholesaler (pid 4774) has obtained the dessert and left to sell it
the wholesaler (pid 4774) delivers sugar and flour
the wholesaler (pid 4774) is waiting for the dessert
chef2 (pid 4777) has taken the sugar MWS
chef2 (pid 4777) has taken the flour MWSF
Chef2 (pid 4777) is preparing the dessert  MWSF
Chef2 (pid 4777) has delivered the dessert MW
the wholesaler (pid 4774) has obtained the dessert and left to sell it
```



```
Chef1 (pid 4776) has delivered the dessert MS
the wholesaler (pid 4774) has obtained the dessert and left to sell it
the wholesaler (pid 4774) delivers flour and sugar
the wholesaler (pid 4774) is waiting for the dessert
chef2 (pid 4777) has taken the sugar MWS
chef2 (pid 4777) has taken the flour MWSF
Chef2 (pid 4777) is preparing the dessert  MWSF
Chef2 (pid 4777) has delivered the dessert MW
the wholesaler (pid 4774) has obtained the dessert and left to sell it
the wholesaler (pid 4774) delivers milk and walnuts
the wholesaler (pid 4774) is waiting for the dessert
chef4 (pid 4779) has taken the milk SFM
chef4 (pid 4779) has taken the walnuts SFMW
Chef4 (pid 4779) is preparing the dessert  SFMW
Chef4 (pid 4779) has delivered the dessert SF
the wholesaler (pid 4774) has obtained the dessert and left to sell it
chef4 (pid 4779) is exiting
chef3 (pid 4778) is exiting
chef2 (pid 4777) is exiting
chef1 (pid 4776) is exiting
chef5 (pid 4780) is exiting
chef0 (pid 4775) is exiting
the wholesaler (pid 4774) is done (total desserts: 11)
viper@ubuntu:~/Desktop/System programming/2022/hw3/151044072$
```