

# Homework 3

☰ Student Name	Muhammed Oğuz
☰ Student Number	1801042634

## Problem Solution Approach

First of all, this is basically (as mentioned teams) a smoker consumer problem. In this problem, smoker only needs one ingredient, in our homework, chefs need two ingredient.

So I use my old function from previous homeworks, that reads the text file. It converts the text file to struct and keeps important things in structs fields.

After this, it creates initial processes for named and unnamed semaphores. And runs it.

## My Approach

I use 4 pushers for sending data to chefs. And they are also processors. So I created 4 more process too for this.

They are also uses `psuherWorking` semaphore to check if another pusher working or not.

## Validation for homework

As mentioned in homework pdf, I had limited time, so, first of all, I created necessary processes.

## Validation for Named Part

Creates all necessary chefs in this part

```

src > C named.c > chefSM()
262     // create chief
263     for (int i = 0; i < CHIEF_COUNT; i++)
264     {
265         pid = fork();
266         if (pid == -1)
267         {
268             GLOBAL_ERROR = FORK_ERROR;
269             return -1;
270         }
271         else if (pid == 0)
272         {
273             // child processes
274             if (i == 0)
275                 chefWS();
276             else if (i == 1)
277                 chefFW();
278             else if (i == 2)
279                 chefSF();
280             else if (i == 3)
281                 chefMF();
282             else if (i == 4)
283                 chefMW();
284             else if (i == 5)
285                 chefSM();
286             exit(0);
287         }
288         else
289         {
290             // parent process
291             pids[i + PUSHER_COUNT] = pid;
292         }
293     }
294 }

```

## Validation for unnamed Part

Creates all child chef processes

```

src > C unnamed.c > pusherW()
236 if (pid == 1)
237 {
238     GLOBAL_ERROR = FORK_ERROR;
239     return -1;
240 }
241 else if (pid == 0)
242 {
243     // child processes
244     if (i == 0)
245         chefWS();
246     else if (i == 1)
247         chefFW();
248     else if (i == 2)
249         chefSF();
250     else if (i == 3)
251         chefMF();
252     else if (i == 4)
253         chefMW();
254     else if (i == 5)
255         chefSM();
256
257     exit(0);
258 }
259 else
260 {

```

## Running Results

### Named Part

Named part working as expected even with 240 input

```

Chef5 (pid 9159) has delivered the desert
Chef5 (pid 9159) is waiting for Sugar and Milk
The wholesaler (pid 9149) has obtained the dessert and left.
the wholesaler (pid 9149) delivers Flour and Milk
The wholesaler (pid 9149) is waiting for the dessert.
Chef3 (pid 9157) has taken the Milk
Chef3 (pid 9157) has taken the Flour
Chef3 (pid 9157) is preparing the desert
Chef3 (pid 9157) has delivered the desert
Chef3 (pid 9157) is waiting for Milk and Flour
The wholesaler (pid 9149) has obtained the dessert and left.
the wholesaler (pid 9149) delivers Walnut and Sugar
The wholesaler (pid 9149) is waiting for the dessert.
Chef0 (pid 9154) has taken the Walnut
Chef0 (pid 9154) has taken the Sugar
Chef0 (pid 9154) is preparing the desert
Chef0 (pid 9154) has delivered the desert
Chef0 (pid 9154) is waiting for Walnut and Sugar
The wholesaler (pid 9149) has obtained the dessert and left.
the wholesaler (pid 9149) delivers Sugar and Milk
The wholesaler (pid 9149) is waiting for the dessert.
Chef5 (pid 9159) has taken the Sugar
Chef5 (pid 9159) has taken the Milk
Chef5 (pid 9159) is preparing the desert
Chef5 (pid 9159) has delivered the desert
Chef5 (pid 9159) is waiting for Sugar and Milk
The wholesaler (pid 9149) has obtained the dessert and left.
The wholesaler (pid 9149) is done.(total desserts: 240).

```

◀ ▶ ~/projects/GTU-University-Assignments/CSE344 - Systems Programming/HW6

## Unnamed Part

Unnamed part does not works correctly unfortunately.

It creates child pcorsseses (passes the validation), unlinks and free's its memory but can not produce correct output.

```

> make run_unnamed
./hw3unnamed -i data/ingredients.txt
Chef0 (pid 9253) is waiting for Walnut and Sugar
Chef1 (pid 9254) is waiting for Flour and Walnut
Chef2 (pid 9255) is waiting for Sugar and Flour
the wholesaler (pid 9248) delivers Sugar and Walnut
Chef3 (pid 9256) is waiting for Milk and Flour
The wholesaler (pid 9248) is waiting for the dessert.
The wholesaler (pid 9248) has obtained the dessert and left
the wholesaler (pid 9248) delivers Walnut and Sugar
The wholesaler (pid 9248) is waiting for the dessert.
Chef5 (pid 9258) is waiting for Sugar and Milk
Chef4 (pid 9257) is waiting for Milk and Walnut
The wholesaler (pid 9248) has obtained the dessert and left
The wholesaler (pid 9248) is done.(total desserts: 0).

```

## Leak results

After running `make shared_mem_leak a`, `make_zombies` and `make_memory`, there is no unfreed or zombie or unlinked shared mem

## Zombie result

```

> make zombies
ps aux | awk '"[Zz]" ~ $8 { printf("%s, PID = %d\n", $8, $2); }'

```

## Shared Mem Result

```

ps aux | awk '"[Zz]" ~ $8 { printf("%s, PID = %d\n", $8, $2); }'
> make shared_mem_leak
ipcs

----- Message Queues -----
key          msqid        owner         perms        used-bytes   messages
----- Shared Memory Segments -----
key          shmid        owner         perms        bytes        nattch       status
----- Semaphore Arrays -----
key          semid        owner         perms        nsems

ls /dev/shm -a
. .

```

## Valgrind Result

## Named Part

There are some little warnings but all content is freed

```
==9788== by 0x109388: main (in /home/mdu/projects/GTU-University-Assignment
ed)
==9788==
==9788==
==9788== HEAP SUMMARY:
==9788==   in use at exit: 0 bytes in 0 blocks
==9788== total heap usage: 44 allocs, 44 frees, 9,105 bytes allocated
==9788==
==9788== All heap blocks were freed -- no leaks are possible
==9788==
==9788== For lists of detected and suppressed errors, rerun with: -s
==9788== ERROR SUMMARY: 9 errors from 9 contexts (suppressed: 0 from 0)
❏ ~/projects/GTU-University-Assignments/CSE344 - Systems Programming/HW03
```

## Unnamed Part

```
==10344==
==10344== HEAP SUMMARY:
==10344==   in use at exit: 0 bytes in 0 blocks
==10344== total heap usage: 10 allocs, 10 frees, 8,202 bytes allocated
==10344==
==10344== All heap blocks were freed -- no leaks are possible
==10344==
==10344== For lists of detected and suppressed errors, rerun with: -s
==10344== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

## Not working parts of homework

unfortunately, unnamed part does not working correctly. But it tries to behaves as expected and creates all necessary child processes.