Gebze Technical University
Computer Engineering

System Programming

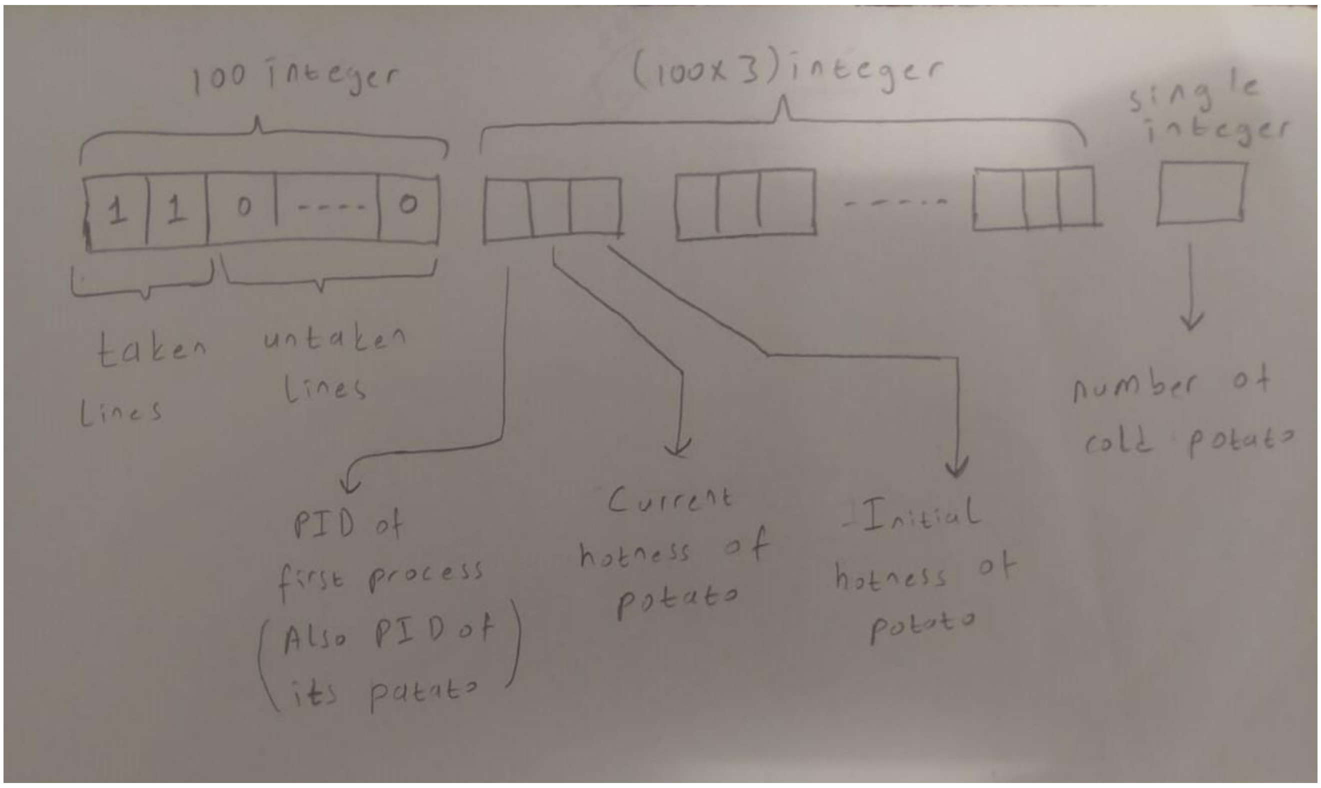
CSE344 – 2021

HW3-REPORT

Yusuf Abdullah ARSLANALP

151044046

Shared Memory Structure



Structure of shared memory are in above. Initially shared memory is empty. N is the number of fifo file. When a process created it iterates on the first N integer on the shared memory. And seek a zero. “1” means fifo file are taken. “0” means fifo file is not taken. In the example third integer is zero. It means that third fifo file are not taken by any process.

Shared memory operations are protected by semaphore. So only one process can use shared memory at the same time.

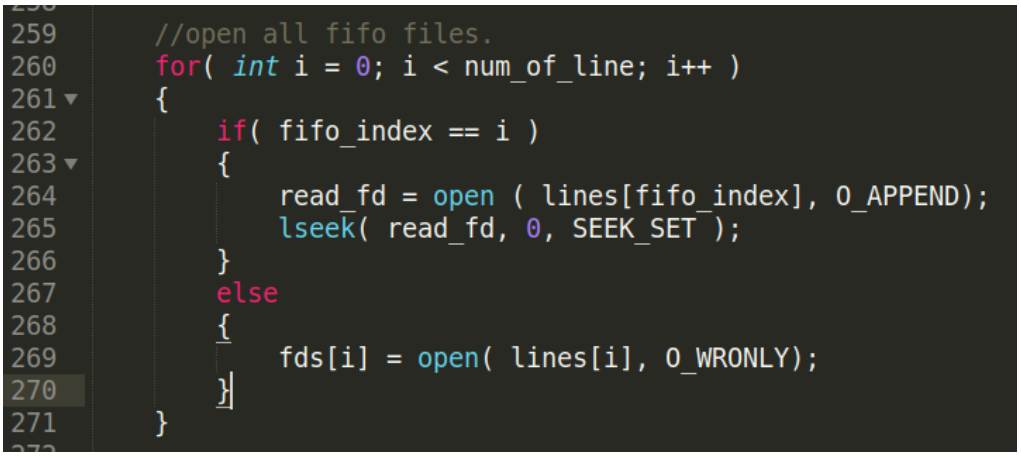
Synchronization Barrier

“The FIFO must be opened on both ends before data can be passed.

Normally, opening the FIFO blocks until the other end is opened also”

Thanks to above property of fifos, all processes start sending and receiving potatoes at the same time.

In the below code snippet, all processes opens fifo files in order. Thanks to above property all processes exit from for loop at the same time.



```
    //open all fifo files.
    for( int i = 0; i < num_of_line; i++ )
    {
        if( fifo_index == i )
        {
            read_fd = open ( lines[fifo_index], O_APPEND);
            lseek( read_fd, 0, SEEK_SET );
        }
        else
        {
            fds[i] = open( lines[i], O_WRONLY);
        }
    }
```

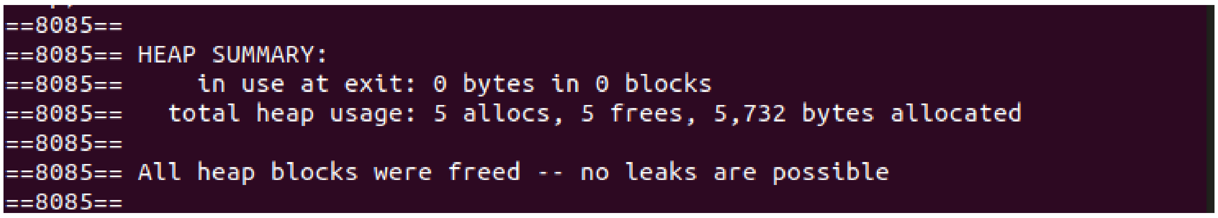
Termination of All Processes

When a potato cooled down the process increment the value of “number of cold potatoes” in shared memory. If number of cooled down potatoes equals to number of potatoes, then current process delivers a message to all other processes. Message send via fifo. And the message is -1.

If a process receives -1 instead of potato_id it terminates itself. Because -1 means no hot potato left.

What did I do:

- When CTRL-C pressed all processes terminated.
- No warning
- I have a make file which compiles program.
- If the required command line arguments are missing/invalid program prints usage information
- No memory leaks
- No busy waiting



```
==8085==
==8085== HEAP SUMMARY:
==8085==     in use at exit: 0 bytes in 0 blocks
==8085==   total heap usage: 5 allocs, 5 frees, 5,732 bytes allocated
==8085==
==8085== All heap blocks were freed -- no leaks are possible
==8085==
```