Ahmet DENİZLİ 161044020

CSE344 System Programming HW4 Report

1. Design

Program checks the correctness of the arguments. Sets STDOUT without buffering. Makes supplier thread to a detached thread. Creates supplier and consumers threads and create semaphores for the amount of ‘1’s and the amount of ‘2’s. Then join to the consumer threads. Finally closes the opened files and semaphores.

There is only 2 semaphore. One for the amount of ‘1’s and one for the amount of ‘2’s. In every character read, supplier thread posts a semaphore representing its amount. Consumer threads only gets when both are available at the same time. This is done with the system v semaphores.

2. Signal Handling

Defined sigint handler function with sigaction for the threads to exit. When takes SIGINT signal, changes the global sigint flag to 1. All processes breaks, free their resources and exits.

3. Functions

void *supplierFun(void *arg): This function for supplier thread. It reads input file’s contents. If it reads a ‘1’ it will post the semaphore representing the amount of ‘1’s read so far, and if it reads a ‘2’ it will post the semaphore representing the amount of ‘2’s read so far. It prints messages concerning its activity.

void * consumerFun(void *arg): This function for consumer threads. It loops N times. At each iteration it remove one ‘1’ and one ‘2’ by reducing the corresponding semaphores’ values. It either takes two items (one ‘1’ and one ‘2’) or waits until two (one ‘1’ and one ‘2’) are available. It prints messages concerning its activity.

void errExit(char *s): This function prints given error via perror then exits.

void sigint_handler(int signum): Handler for SIGINT and SIGTERM for wholesaler.

unsigned long get_time_microseconds(): This function returns current timestamp in microseconds.

unsigned long get_time_seconds(): This function returns current timestamp in seconds.

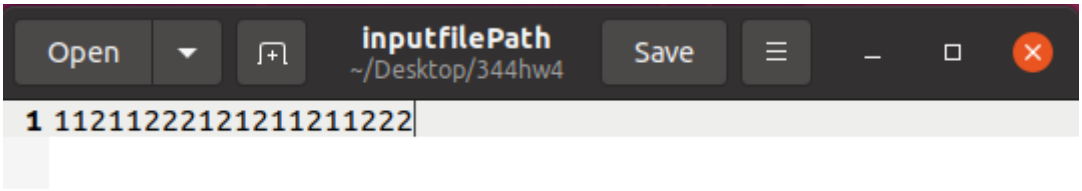
int main(int argc, char *argv[]): Checks the correctness of the arguments.

Sets STDOUT without buffering. Makes supplier thread to a detached thread.

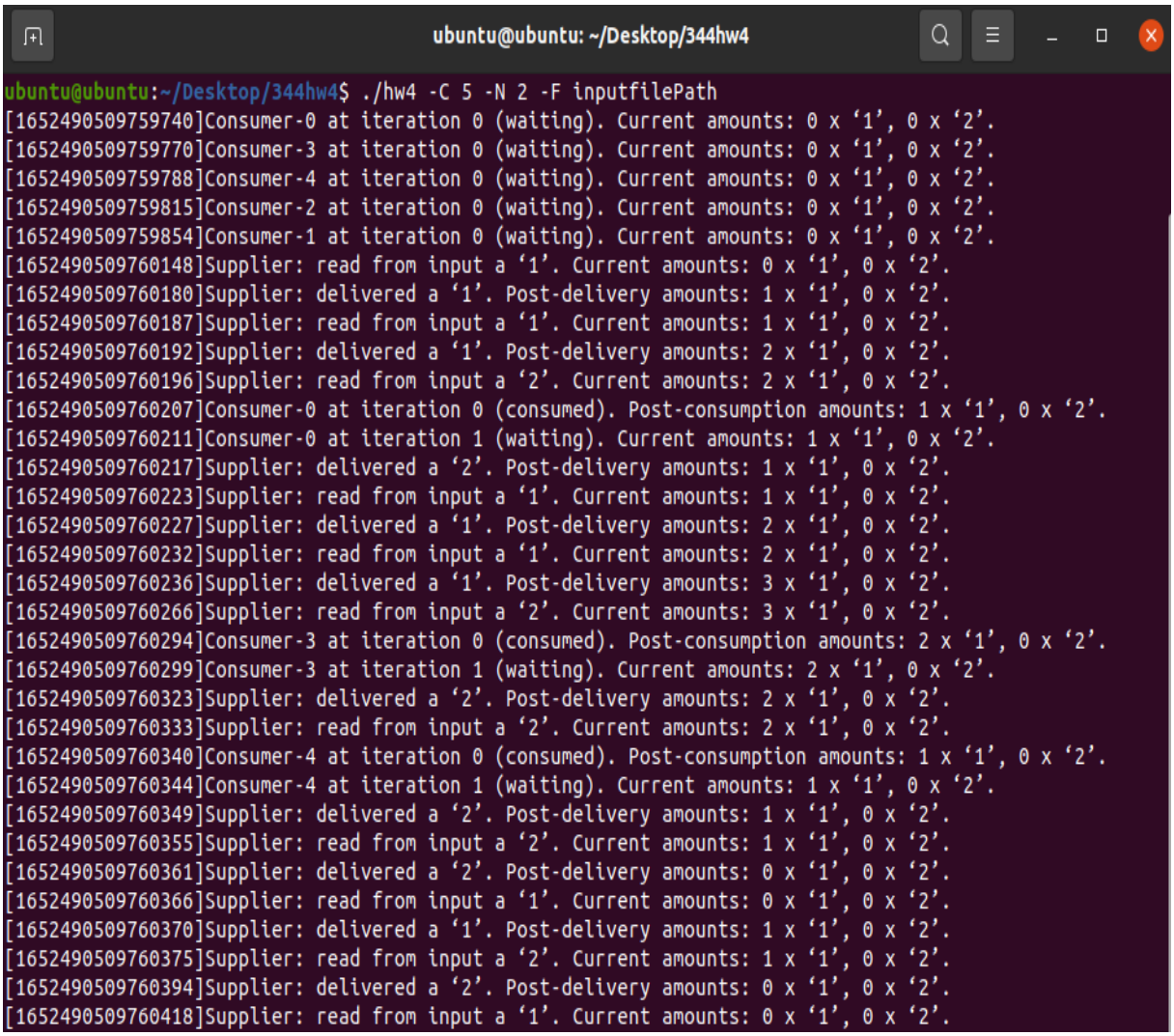
Creates supplier and consumers threads and create semaphores for the amount of ‘1’s and the amount of ‘2’s. Then join to the consumer threads. Finally closes the opened files and semaphores.

4. Sample Screenshots

Input File



Output



```
[1652490509760424]Supplier: delivered a ‘1’. Post-delivery amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760428]Supplier: read from input a ‘2’. Current amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760423]Consumer-2 at iteration 0 (consumed). Post-consumption amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760436]Consumer-0 at iteration 1 (consumed). Post-consumption amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760437]Consumer-2 at iteration 1 (waiting). Current amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760439]Consumer-0 has left.
[1652490509760447]Consumer-1 at iteration 0 (consumed). Post-consumption amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760452]Consumer-1 at iteration 1 (waiting). Current amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760554]Supplier: delivered a ‘2’. Post-delivery amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760584]Supplier: read from input a ‘1’. Current amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760590]Supplier: delivered a ‘1’. Post-delivery amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760595]Supplier: read from input a ‘1’. Current amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760599]Supplier: delivered a ‘1’. Post-delivery amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760603]Supplier: read from input a ‘2’. Current amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760612]Consumer-3 at iteration 1 (consumed). Post-consumption amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760615]Consumer-3 has left.
[1652490509760638]Supplier: delivered a ‘2’. Post-delivery amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760643]Supplier: read from input a ‘1’. Current amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760666]Supplier: delivered a ‘1’. Post-delivery amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760690]Supplier: read from input a ‘1’. Current amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760698]Supplier: delivered a ‘1’. Post-delivery amounts: 3 x ‘1’, 0 x ‘2’.
[1652490509760702]Supplier: read from input a ‘2’. Current amounts: 3 x ‘1’, 0 x ‘2’.
[1652490509760710]Consumer-4 at iteration 1 (consumed). Post-consumption amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760713]Consumer-4 has left.
[1652490509760745]Supplier: delivered a ‘2’. Post-delivery amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760752]Supplier: read from input a ‘2’. Current amounts: 2 x ‘1’, 0 x ‘2’.
[1652490509760774]Supplier: delivered a ‘2’. Post-delivery amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760782]Supplier: read from input a ‘2’. Current amounts: 1 x ‘1’, 0 x ‘2’.
[1652490509760787]Supplier: delivered a ‘2’. Post-delivery amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760790]The Supplier has left.
[1652490509760856]Consumer-2 at iteration 1 (consumed). Post-consumption amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760888]Consumer-2 has left.
[1652490509760919]Consumer-1 at iteration 1 (consumed). Post-consumption amounts: 0 x ‘1’, 0 x ‘2’.
[1652490509760923]Consumer-1 has left.
ubuntu@ubuntu:~/Desktop/344hw4$
```