Studio6

**Exercise1**: Qitao Xu, Jiangnan Liu, Zhe Wang

**Exercise2**:

#include <stdio.h> //For printf()

#include <stdlib.h> //for exit() and atoi()

#include <signal.h>

#include <string.h>

#include <errno.h>

#include <unistd.h> // for write()

#include <sys/types.h>

#include <sys/socket.h> // for bind(), socket()

#include <sys/un.h> // for unix()

#include <netinet/ip.h>

#include <arpa/inet.h>

const int num\_expected\_args = 3;

unsigned int num\_comm\_times;

char \* IPC\_mechanism;

int main( int argc, char\* argv[] ) {

int ret\_fork;

if (argc != num\_expected\_args) {

printf("Usage: ./ipc <# communication times> <IPC mechanism>\n");

exit(-1);

}

num\_comm\_times = atoi(argv[1]);

IPC\_mechanism = argv[2];

printf("Here is parent process. num\_comm\_times = %d, IPC\_mechanism is %s\n", num\_comm\_times, IPC\_mechanism);

fflush(stdout);

ret\_fork = fork();

if (ret\_fork < 0) {

printf("fork() system call failed!\n Reason: %s", strerror(errno));

exit(-1);

}

if (ret\_fork > 0) {

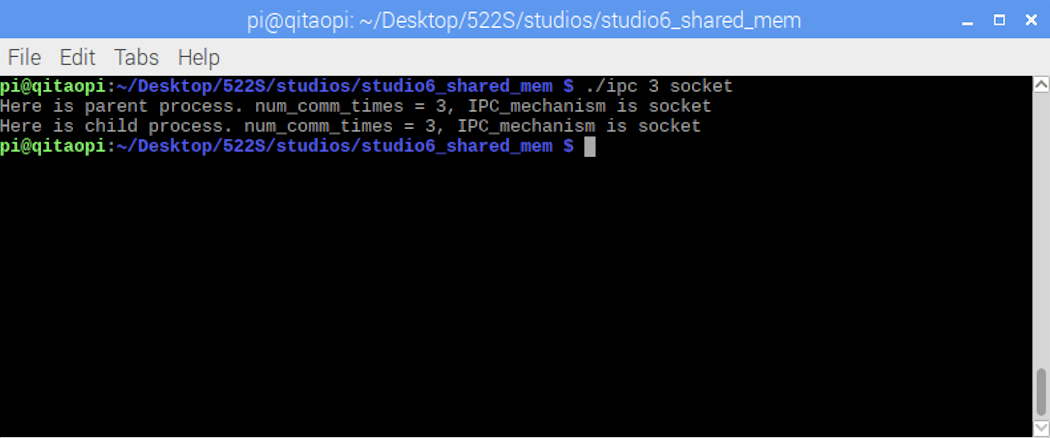
printf("Here is child process. num\_comm\_times = %d, IPC\_mechanism is %s\n", num\_comm\_times, IPC\_mechanism);

fflush(stdout);

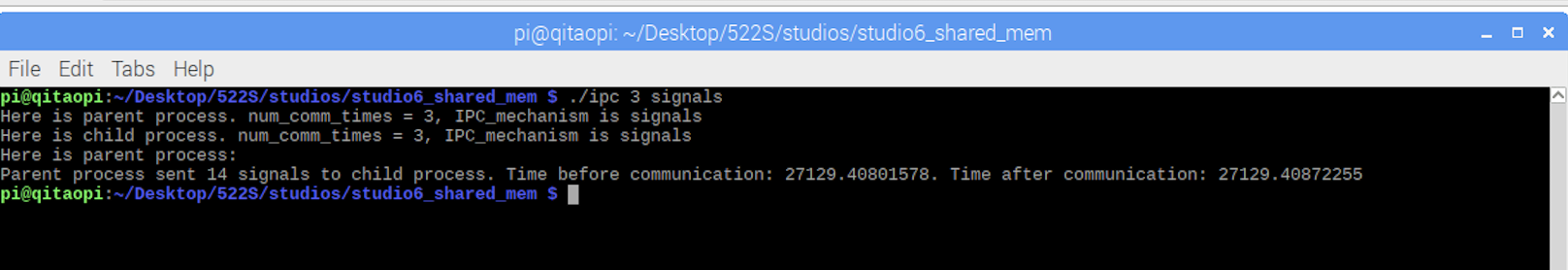
}

return 0;

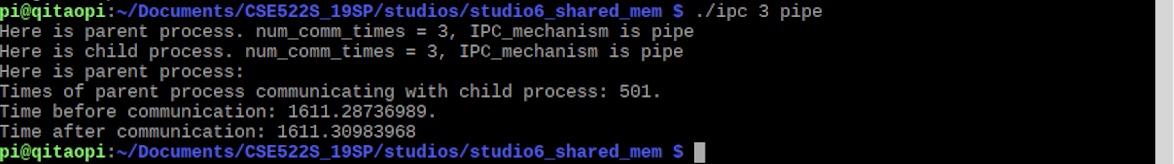
}



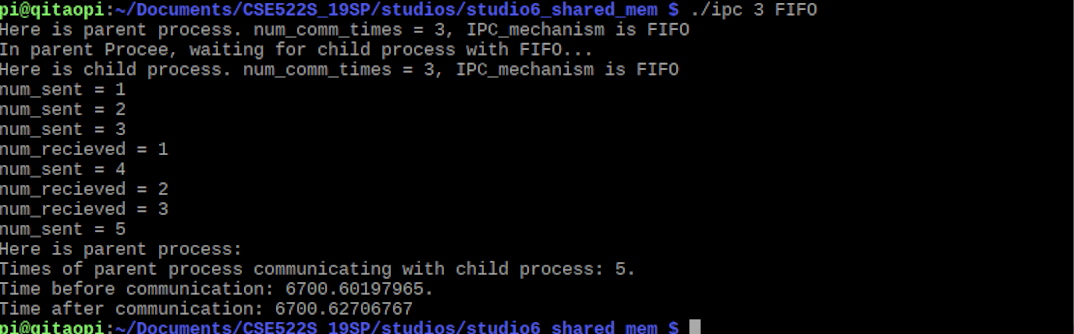
**Exercise3**:



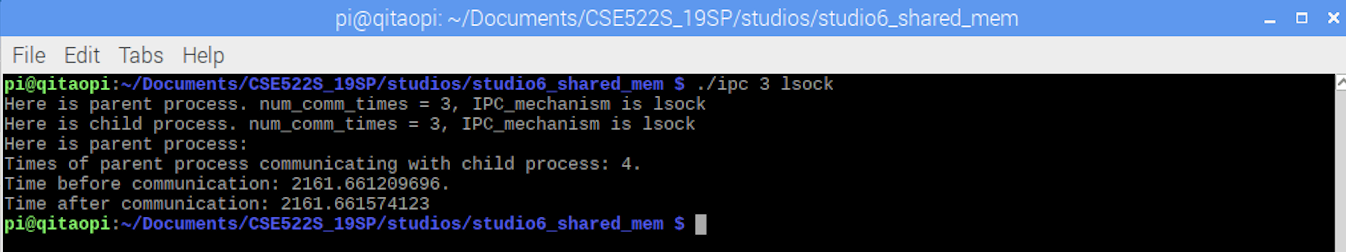
**Exercise4**:



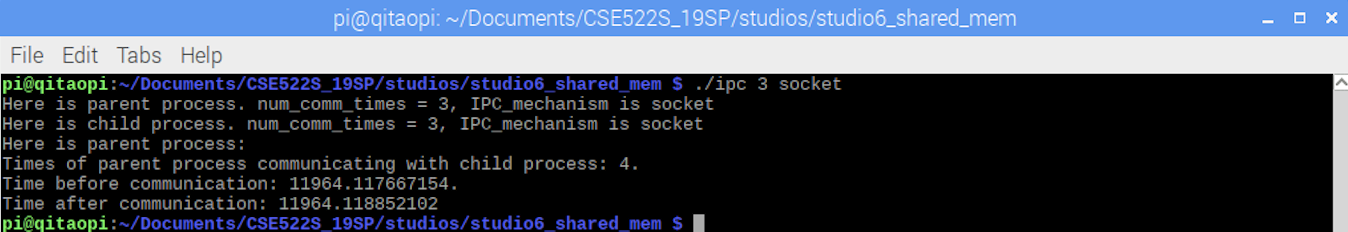
**Exercise5**:



**Exercise6**:



**Exercise7**:



**Exercise8**:

Pipe and signals can work more efficiently while fifo is the slowest mechanism in most case. Local socket can work better than Internet socket in terms of time.