#define FIFO1 "server_to_worker"

#define FIFO2 "worker_to_server"

Nazwy dwóch potoków FIFO (plików):

jeden do wysyłania danych z serwera do workera

drugi z workera do serwera

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

argc – liczba argumentów

argv – argumenty z linii poleceń

if (argc != 2) {

Sprawdza, czy podano dokładnie jeden argument (np. server albo worker)

printf("usage: %s server and worker\n", argv[0]);

Wypisuje, jak poprawnie uruchomić program

SERVER:

if (strcmp(argv[1], "server") == 0) {

Sprawdza, czy argument to "server"

mkfifo(FIFO1, 0666);

mkfifo(FIFO2, 0666);

Tworzy dwa FIFO (jak pliki, ale do komunikacji):

0666 → każdy może czytać i pisać

int write_fd = open(FIFO1, O_WRONLY);

Serwer pisze do FIFO1 (do workera)

int read_fd = open(FIFO2, O_RDONLY);

Serwer czyta z FIFO2 (od workera)

server_loop(write_fd, read_fd);

Uruchamia główną logikę serwera (działa, dopóki pętla się nie skończy)

close(write_fd);

close(read_fd);

Zamyka potoki

unlink(FIFO1);

unlink(FIFO2);

Usuwa FIFO z systemu (kasuje pliki)

printf("server: done\n");

Informacja, że serwer skończył

WORKER:

else if (strcmp(argv[1], "worker") == 0) {

Sprawdza, czy argument to "worker"

int read_fd = open(FIFO1, O_RDONLY);

Worker czyta z FIFO1 (od serwera)

int write_fd = open(FIFO2, O_WRONLY);

Worker pisze do FIFO2 (do serwera)

worker_loop(read_fd, write_fd);

Główna pętla workera (robi swoją robotę)

close(read_fd);

close(write_fd);

Zamyka FIFO.

else {

printf("Error:invalid argument");

return 1;

}

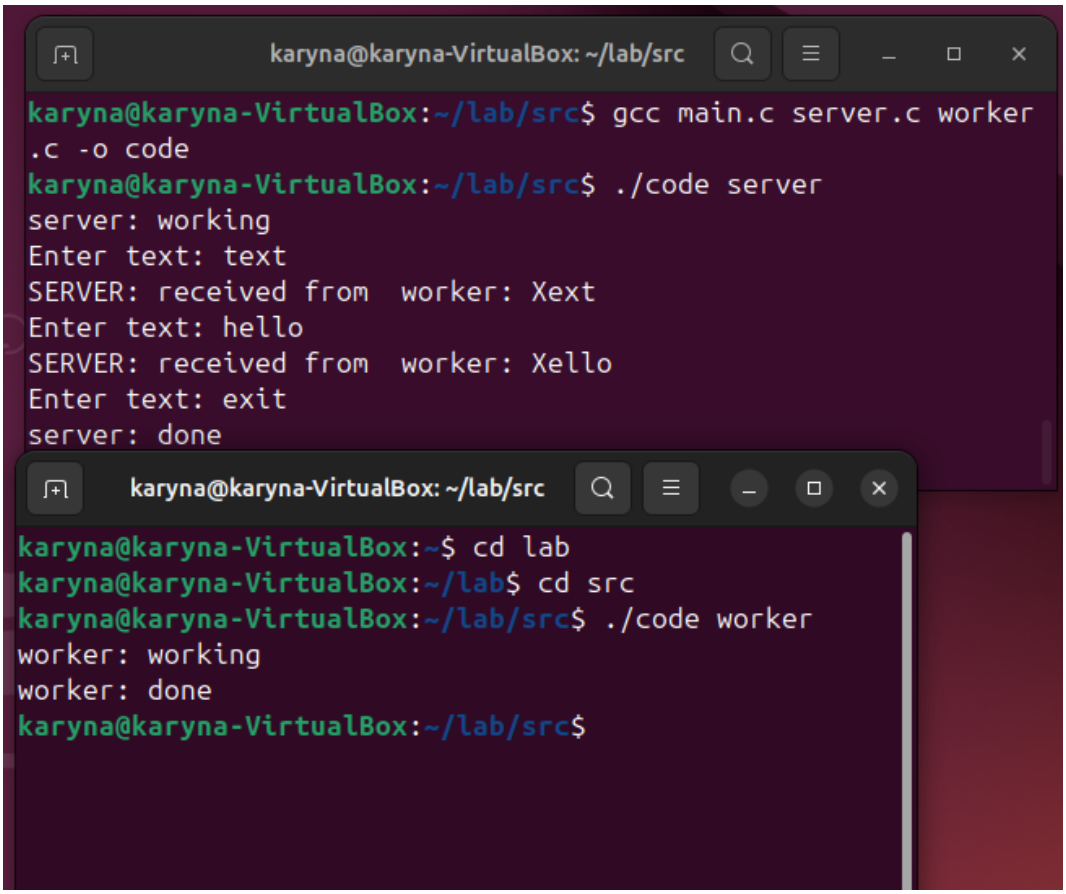
Jeśli argument nie jest ani server, ani worker.

return 0;

Program zakończył się poprawnie.

Wynik:

Po uruchomieniu serwera i workera możliwa jest wymiana danych między nimi.
Tekst wpisany po stronie serwera jest przesyłany do workera, a następnie przetworzony tekst wraca do serwera i zostaje wyświetlony. Po wpisaniu komendy `exit` serwer kończy działanie.


karyna@karyna-VirtualBox: ~/lab/src
karyna@karyna-VirtualBox:~/lab/src$ gcc main.c server.c worker
.c -o code
karyna@karyna-VirtualBox:~/lab/src$ ./code server
server: working
Enter text: text
SERVER: received from worker: Xext
Enter text: hello
SERVER: received from worker: Xello
Enter text: exit
server: done
karyna@karyna-VirtualBox: ~/lab/src
karyna@karyna-VirtualBox:~$ cd lab
karyna@karyna-VirtualBox:~/lab$ cd src
karyna@karyna-VirtualBox:~/lab/src$ ./code worker
worker: working
worker: done
karyna@karyna-VirtualBox:~/lab/src$