Zad 1

Dla każdego z kodów zostało to samo wysłane i póki daje to samo każdemu klientowi

Kod:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/shm.h>

#include <sys/sem.h>

#define SHM\_SIZE 1024

struct sembuf sem\_op;

//funkcja do czekania

void wait\_semaphore(int sem\_id, int sem\_num) {

sem\_op.sem\_num = sem\_num;

sem\_op.sem\_op = -1;

sem\_op.sem\_flg = 0;

semop(sem\_id, &sem\_op, 1);

}

//oblsuga syngalow

void signal\_semaphore(int sem\_id, int sem\_num) {

sem\_op.sem\_num = sem\_num;

sem\_op.sem\_op = 1;

sem\_op.sem\_flg = 0;

semop(sem\_id, &sem\_op, 1);

}

int main() {

//tworzenie pamieci dzielonej

key\_t key = ftok("keyfile", 'R');

int shmid = shmget(key, SHM\_SIZE, IPC\_CREAT | 0666);

char \*shmaddr = shmat(shmid, NULL, 0);

//tworzenie semeforow

int sem\_id = semget(key, 1, IPC\_CREAT | 0666);

sem\_op.sem\_num = 0;

sem\_op.sem\_op = 1;

sem\_op.sem\_flg = 0;

semop(sem\_id, &sem\_op, 1);

pid\_t pid;

//przechodzenie po klientach

for (int i = 0; i < 3; ++i) {

pid = fork();

if (pid == 0) {

// Klient

while (1) {

wait\_semaphore(sem\_id, 0);

printf("Klient %d otrzymał: %s\n", i + 1, shmaddr);

signal\_semaphore(sem\_id, 0);

sleep(1); // Symulacja przetwarzania przez klienta

}

} else if (pid < 0) {

perror("Błąd fork");

exit(EXIT\_FAILURE);

}

}

// Producent, odbieranie stdin

while (1) {

wait\_semaphore(sem\_id, 0);

printf("Podaj słowo: ");

fgets(shmaddr, SHM\_SIZE, stdin);

signal\_semaphore(sem\_id, 0);

}

//usuwanie pamieci

shmdt(shmaddr);

shmctl(shmid, IPC\_RMID, NULL);

return 0;

}

A screenshot of a computer

Description automatically generated