Zad 1 lacze

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <ctype.h>

#include <string.h>

int main() {

// Stworzenie pipe

int pipe1[2];

int pipe2[2];

int pipe3[2];

int pipe4[2];

if (pipe(pipe1) == -1 || pipe(pipe2) == -1 || pipe(pipe3) == -1 || pipe(pipe4) == -1) {

perror("pipe");

return 1;

}

pid\_t pp1 = fork();

if (pp1 == -1) {

perror("fork");

return 1;

}

if (pp1 == 0) {

close(pipe1[0]); // Close read end in pp1

close(pipe4[1]);

char word[10];

printf("PM: %d; PP1:%d; ", getppid(), getpid());

printf("\n2. PP1 (%d); Podaj słowo: ", getpid());

scanf("%s", word);

//wysłanie do PM

write(pipe1[1], word, sizeof(word));

close(pipe1[1]); // Close write end in pp1

printf("3. PP1 (%d, %d): Wysłano do PM: %s\n", getpid(), pipe1[1], word);

//odebranie od PM

char changed\_word[10];

read(pipe4[0], changed\_word, sizeof(changed\_word));

close(pipe4[0]);

printf("10. PP1 (%d, %d): Odczytano od PM: %s\n", getpid(), pipe4[0], changed\_word);

printf("PP1: %s \n", changed\_word);

return 0;

}

pid\_t pp2 = fork();

if (pp2 == -1) {

perror("fork");

return 1;

}

char changed\_word[10];

if (pp2 == 0) {

printf(" \nPP2:%d; \n", getpid());

close(pipe2[1]);

close(pipe3[0]);

char received\_word[10];

//odebranie od PM

read(pipe2[0], received\_word, sizeof(received\_word));

close(pipe2[0]); // Close read end of pp1\_pipe

printf("6. PP2 (%d, %d): Odczytano od PM: %s \n", getpid(), pipe2[0], received\_word);

// zmiana słowa

for (int i = 0; i < sizeof(changed\_word); ++i) {

if (islower(received\_word[i])) {

changed\_word[i] = toupper(received\_word[i]);

} else if (isupper(received\_word[i])) {

changed\_word[i] = tolower(received\_word[i]);

}

}

//wysłanie do PM

write(pipe3[1], changed\_word, sizeof(changed\_word));

close(pipe3[1]); // Close write end in pp3

printf("7. PP2 (%d, %d): Wysłano do PM: %s \n", getpid(), pipe3[1], changed\_word);

return 0;

}

//odebranie od PP1

close(pipe1[1]); // Close read end in PM

char wordP1[10];

read(pipe1[0], wordP1, sizeof(wordP1));

close(pipe1[0]); // Close read end of pp1\_pipe

printf("4. PM (%d, %d): Odczytano od P1: %s\n", getpid(), pipe1[0], wordP1);

//wysłanie do PP2

close(pipe2[0]); // Close write end in PM

write(pipe2[1], wordP1, sizeof(wordP1));

close(pipe2[1]);

printf("5. PM (%d, %d): Wysłano do P2: %s\n", getpid(), pipe2[1], wordP1);

//odebranie od PP2

char wordP2[10];

close(pipe3[1]); // Close read end in PM

read(pipe3[0], wordP2, sizeof(wordP2)); // Close write end in PM

close(pipe3[0]);

printf("8. PM (%d, %d): Odczytano od P2: %s\n", getpid(), pipe3[0], wordP2);

//wysłanie do PP1

close(pipe4[0]); // Close read end in PM

write(pipe4[1], wordP2, sizeof(wordP2));

close(pipe4[1]); // Close write end in PM

printf("9. PM (%d, %d): Wysłano do P1: %s\n", getpid(), pipe4[1], wordP2);

waitpid(pp1, NULL, 0);

waitpid(pp2, NULL, 0);

printf("\nProgram zakonczyl sie.\n");

return 0;

}

A screenshot of a computer

Description automatically generated