Programowanie systemowe

Sprawozdanie

Aron Krajda - 283874

Data wykonania sprawozdania: 30.11.2025

Cele:

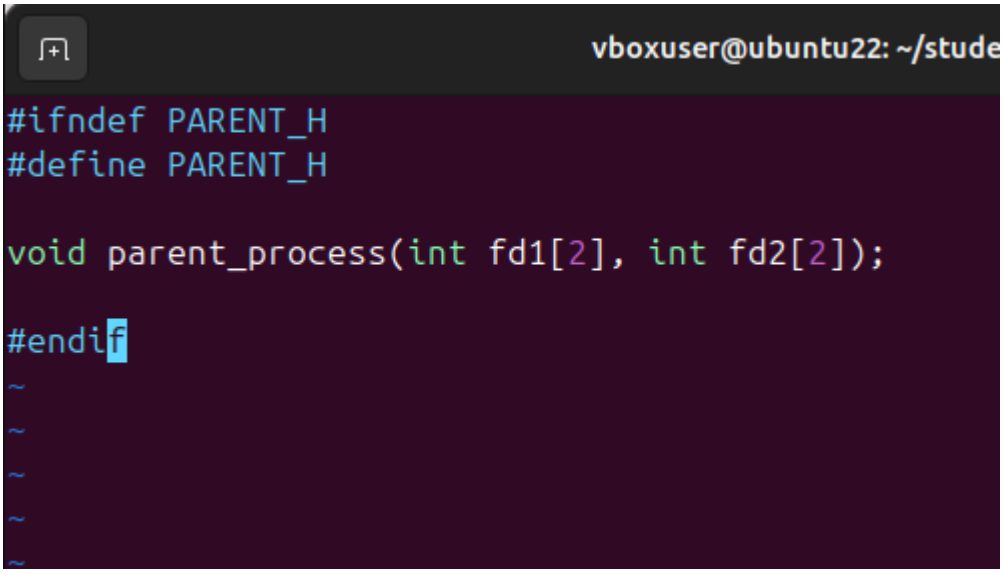
Celem zadania był dalszy rozwój napisanego na poprzednich zajęciach programu w C, którego dodatkowe (dodane na tych zajęciach) funkcje, poza tymi już zaimplementowanymi, miały być następujące:

- kod miał zostać zmodyfikowany tak, aby proces dziecko odbierał przez pipe tekst i go modyfikował, a następnie wysyłał go przez drugi pipe do rodzica, aby ten go wyświetlił
- kod miał zostać podzielony na osobne pliki dla każdej funkcji
- program miał wykonywać się w nieskończoność, chyba, że jako tekst zostanie wprowadzone słowo “exit”

Poprzedni plik nazwany wcześniej zadanie.c, podzielono na 3 nowe pliki .c: parent.c, child.c oraz main.c. Opis wszystkich z tych plików znajduje się poniżej.

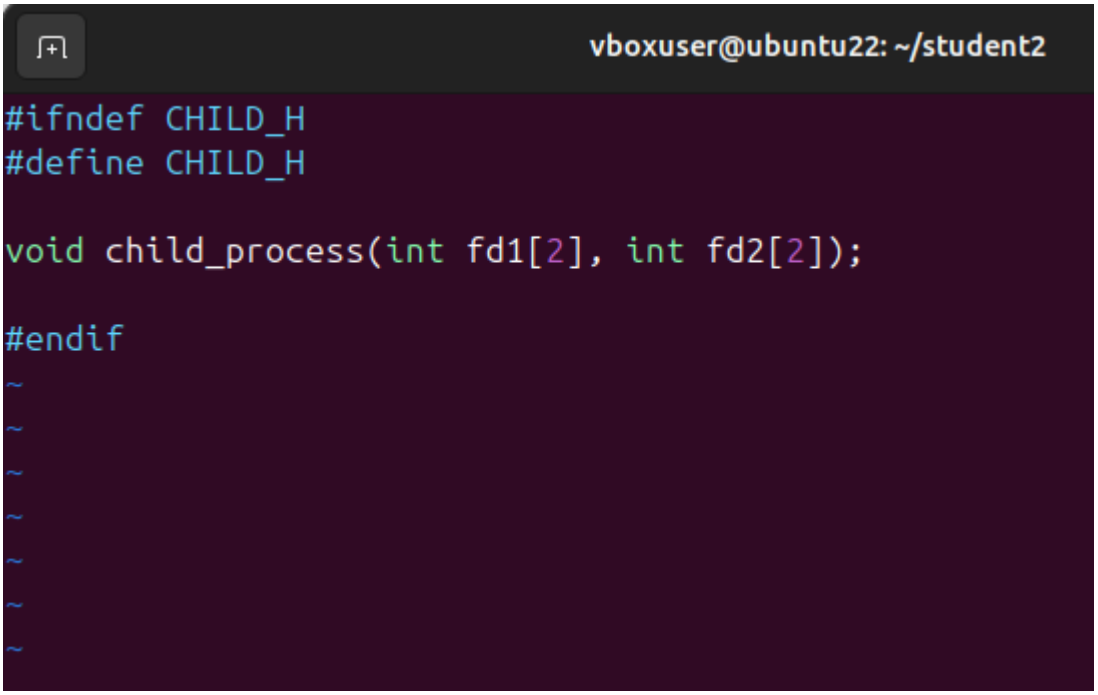
Plik parent.h:

W pliku parent.h znajduje się deklaracja parent_process, która wykonuje zadania przypisane procesowi rodzicowi.



Plik child.h:

W pliku child.h znajduje się deklaracja child_process, która wykonuje zadania przypisane procesowi dziecku.



Plik parent.c:

Plik parent.c zawiera funkcję parent_process.

Pierwsze 5 linijek to niezbędne include'y, w tym deklaracja parent_process.

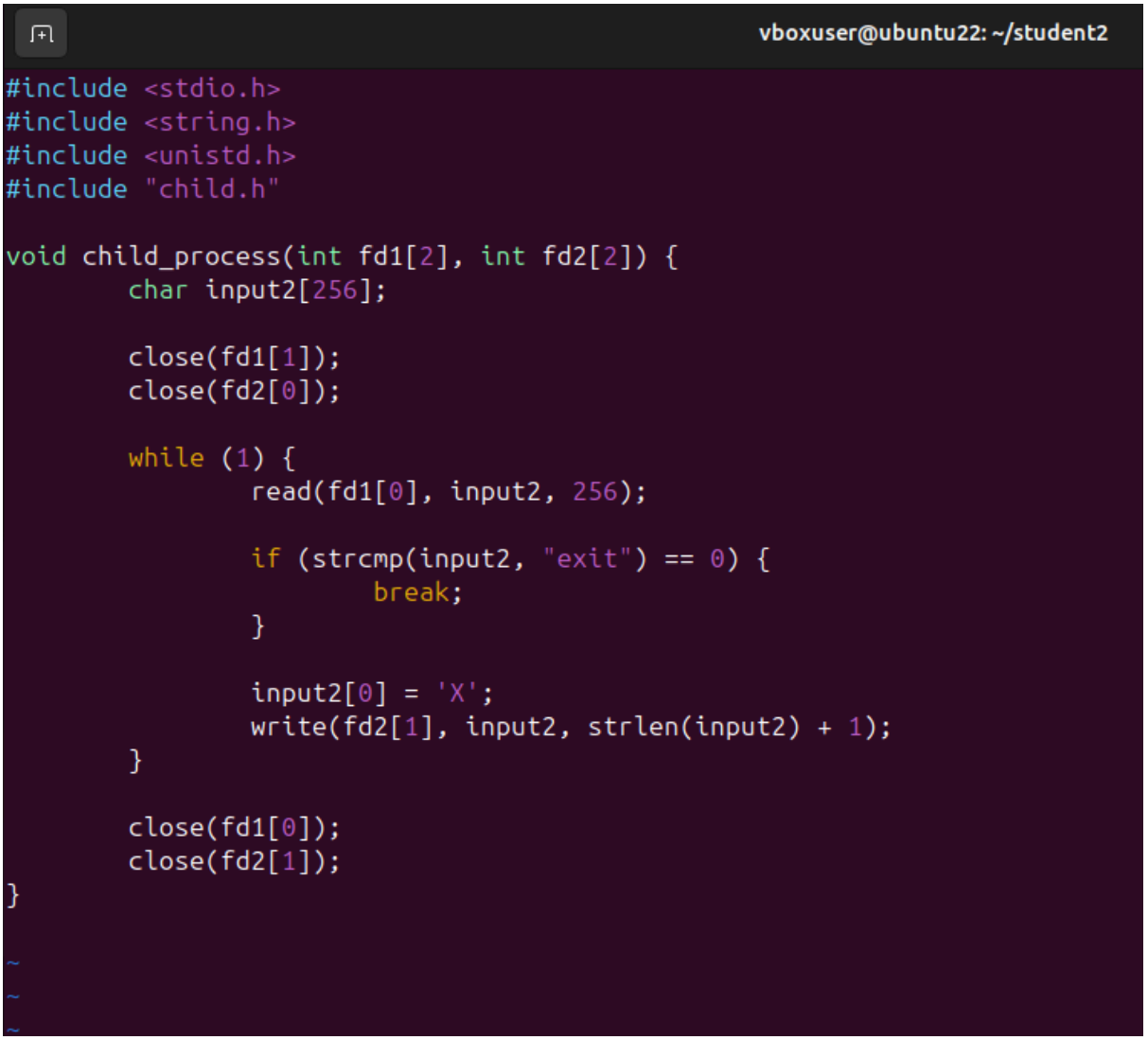
Dalej znajduje się definicja parent_process, a w nim linijki write oraz read służące do wysyłania przez proces rodzica tekstu przez pipe do procesu dziecka, a następnie odebrania innego tekstu od procesu dziecka przez drugiego pipe. Wszystko to zawiera się w pętli while, która działa póki nie zostanie wpisane słowo "exit" (co zostało ustalono pomiędzy linijkami do używania pipe). Następnie proces rodzic wyświetla nowy teskt. Pozostałe elementy kody zostały niezmienione względem poprzednich laboratoriów lub "podwojone" aby obsługiwać dwa pipe'y a nie jeden.

Plik child.c:

Plik child.c zawiera funkcję child_process.

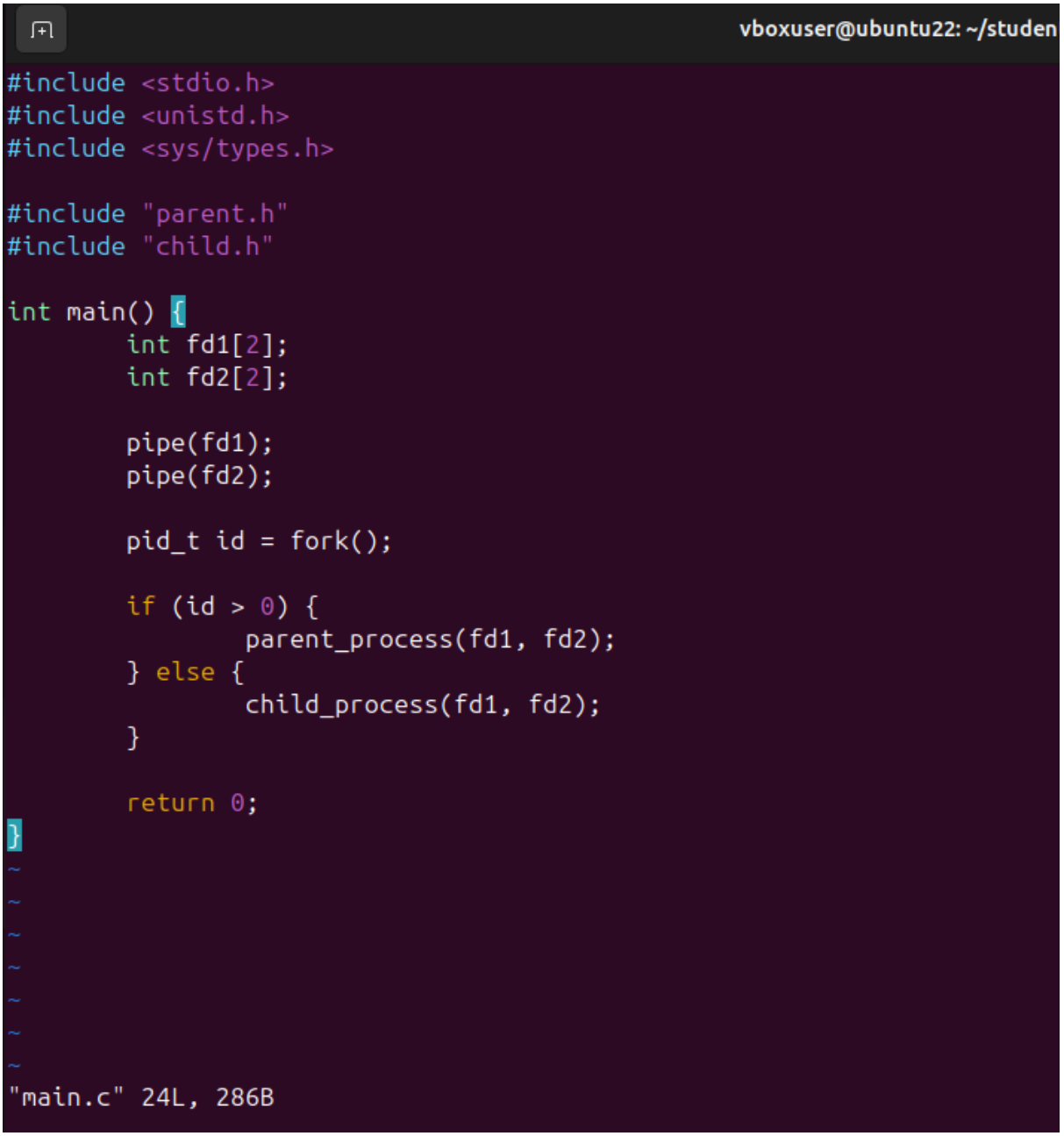
Pierwsze 4 linijki to niezbędne include'y, w tym deklaracja child_process.

Następnie w pętli while znajdują się linijki do użytkowania pipe, jedna do odbierania danych od procesu rodzica (read), a druga do ich wysyłania (write), podobnie jak w pliku parent.c. Ten plik również zawiera linijkę gwarantującą zakończenie pętli po wpisaniu słowa "exit" w działającym programie, aby mieć pewność, że proces również się zakończy i nie zostanie procesem osieroconym. Pozostałe elementy kody zostały niezmienione względem poprzednich laboratoriów lub "podwojone" aby obsługiwać dwa pipe'y a nie jeden, podobnie jak w pliku parent.c.



Plik main.c:

W tym pliku znajdują się deklaracje zarówno parent_process jak i child_process. Deklarowane są dwie pary deskryptorów - dla obydwu pipe'ów. Następnie wykonujący się program jest rozdzielany na dwa procesy.



Uruchomienie programu i wnioski:

Po skompilowaniu wszystkich plików przy użyciu Make oraz ich uruchomieniu można zauważyć, że program działa poprawnie w pętli i przerywa działanie po wpisaniu słowa “exit”. Wskazuje to na poprawne wykonanie wszystkich plików oraz poprawne zaimplementowanie wszystkich nowych funkcji.

```
vboxuser@ubuntu22:~/student2$ ./program
czx
Xzx
gd
Xd
exit
vboxuser@ubuntu22:~/student2$
```