Laboratorium 5.

Pobierz plik lab5.tar.gz i rozpakuj go.

Zadanie 0. Demony 3p

a) Napisz program, który będzie działał jako demon. Zadaniem programu jest wpisywanie znacznika czasu do pliku co jedną minutę. Program ma przesyłać komunikat o uruchomieniu i ewentualnych błędach do demona syslogd. Proszę przygotować dwie wersje:
- Z własną funkcją uruchamiającą demona,
- Z funkcją biblioteczną daemon() .

b) Uzupełnij program o czytanie pliku konfiguracyjnego, w którym podawana jest częstotliwość generowania znacznika czasu.

c) Zmodyfikuj program tak, aby w trakcie działania programu możliwe było ponowne przeczytanie pliku konfiguracyjnego. Wskazówka: wykorzystaj sygnał SIGHUP.

Zadanie 1.

Zapoznaj się z programem `pipedemo.c`.
Napisz program, który będzie wysyłał list automatycznie generowany przez program. Do wysyłania listu użyj polecenia `mail`.
Użyteczna funkcja: do utworzenia składowych polecenia można użyć instrukcji:

```
sprintf(arg,"-s 'Wiadomosc od procesu PID %d'", getpid());
```

Zadanie 2.

Zapoznaj się z programem `simpleredirect.c`. Program ten jest równoważny wykonaniu polecenia:

```
ls -l | sort -n -k5.
```

a) Czy ma znaczenie, w którym procesie będzie wykonywane polecenie `ls`?
b) Co będzie się działo, jeśli deskryptory `fd[0]` i `fd[1]` nie będą zamknięte przed wywołaniem `execl`?
c) Rozbuduj program `simpleredirect.c` tak, aby polecenia do wykonania były pobierane z wiersza wywołania programu.

Zadanie 3.

Napisz program, składający się z procesu macierzystego i potomnego, w którym:
- proces macierzysty generuje komunikat,
- proces potomny odbiera komunikat za pomocą potoku, zamienia go na pisany wielkimi literami i przesyła z powrotem za pomocą potoku do procesu macierzystego
- proces macierzysty drukuje otrzymany komunikat.

Uwaga: należy utworzyć dwa potoki łączące proces macierzysty i potomny.

Zadanie 4.

Program czyta z dwóch plików, do których napływają dane. Napisz program, który będzie czytał te pliki do momentu, kiedy zostaną zakończone i wyświetlał sumę przeczytanych bajtów z obydwu plików.

Zadanie 5.

Napisz program, który prosi o podanie hasła i kończy działanie, jeśli użytkownik nie wprowadzi hasła w określonym czasie. Wprowadzane hasło ma być maskowane i ma być widać znaki (np. ‘x’) w czasie wpisywania.

Zadania domowe

Zadanie 6.
Rozbuduj program z zadania 2 tak, aby potok mógł być dowolnie duży. Przykład:

```
simpleredirect who sort head.
```

Jest to odpowiednik polecenia `who |sort |head`.

Zadanie 7
Uzupełnij program `shell` z laboratorium 5-6 o „ręczną” obsługę potoków.

Zadanie 8.
Napisz program, który zbiera komunikaty od wielu programów i wyświetla je na ekranie. Do komunikacji użyj potoku nazwanego.
Wskazówka: Utwórz program rdfifo, którego zadaniem jest utworzenie kolejki FIFO i czytanie z niej danych. Utwórz program wrfifo, który otwiera kolejkę FIFO tylko do zapisu i wpisuje do niej dane (np. swoj pid i czas). W jaki sposób przekażesz wspólną nazwę kolejki FIFO do tych programów? W jaki sposób zapewnić działanie programu zbierającego komunikaty również wtedy, kiedy nie ma programu piszącego do łącza? Jak zapewnić to, że komunikaty pochodzące od różnych programów wyświetlane są w całości, tzn. nie są rozdzielane komunikatami od innych programów?