**Programowanie współbieżne i rozproszone – laboratorium**

**Instrukcja do ćwiczenia laboratoryjnego II**

**Pliki, potoki, sygnały**

Autor: S. Samolej

Rzeszów, 2013

Podziękowania:  
Składam podziękowania panu dr inż. Jędrzejowi Ułasiewiczowi za udostepnienie materiałów wykładowych i laboratoryjnych, które były inspiracją do opracowania tych materiałów.

1. Proszę przypomnieć sobie składnie funkcji **open()**, **write()**, **read()**, **close()** do podstawowej obsługi plików w systemach Unixowych.
   1. Należy skompilować i uruchomić przykładowe programy „simple\_write.c”, „simple\_read.c”, „copy\_system.c”. Na potrzeby wykonywania programu copy-system należy przygotować plik źródłowy, o nazwie „file.in”, który ma zostać skopiowany.
2. Proszę skompilować i uruchomić przykładowy program „**lockf.c**”. Należy zwrócić uwagę w jaki sposób zapewniono wzajemne wykluczanie w dostępie do współdzielonego pliku.
3. Proszę skompilować przykładowe programy „**polecenie2.c**” i „**polecenie1.c**”.
   1. Wywołując programy ./polecenie2 | ./polecenie1 należy przekazać strumień wyjściowy programu polecenie2 na wejście polecenia1.
   2. Należy przetestować działanie programu polecenie1 przez przekazanie mu na standardowe wejście danych w następującej postaci:  
      echo To jest tekst. | ./polecenie1  
      ./polecenie1 < Tekst1.txt (Plik Tekst1.txt należy uprzednio utworzyć i wpisać do niego jakiś tekst).
4. Proszę skompilować i uruchomić przykładowe programy „**popen1.c**” i „**popen2.c**” i **„popen3.c”**. Należy zwrócić uwagę na zaoferowane w nich sposoby uruchamiania zewnętrznych programów wraz z przesyłaniem lub odbieraniem od nich strumieni danych.
5. Proszę skompilować i uruchomić przykładowe programy „**pipe1.c**”. Należy zwrócić uwagę w jaki sposób program powołuje do życia nienazwany potok i przesyła sam do siebie przez niego strumień danych.
6. Proszę skompilować i uruchomić przykładowe programy „**pipe2.c**”. Należy zwrócić uwagę w jaki sposób w programie składającym się z 2 procesów zorganizowano komunikację pomiędzy procesami z zastosowaniem nienazwanego potoku.
7. Proszę skompilować przykładowe programy „**pipe3.c**” i „**pipe4.c**”. Pliki wykonywalne należy zgromadzić w jednym katalogu. Proszę zwrócić uwagę jak nazywa się program utworzony na podstawie pliku źródłowego pipe4.c.
   1. Należy uruchomić program utworzony na podstawie pliku źródłowego pipe3.c. Proszę zwrócić uwagę, jak ten program w nowym procesie uruchamia inny program i przekazuje mu informację o potoku nienazwanym, z którego ma pobierać dane w parametrach wywołania programu.
8. Proszę skompilować i uruchomić program „**fifo1.c**”. Program tworzy systemową kolejkę fifo widoczną w systemie plików jako **/tmp/my\_fifo**. Należy przetestować działanie kolejki z zastosowaniem następujących poleceń:  
   **cat < /tmp/fifo1 &**  
   **echo "Ala ma kota" > /tmp/fifo1**
9. Proszę skompilować programy „fifo3.c” i „fifo4.c” i zwrócić uwagę, w jaki sposób została zorganizowana w tych dwóch programach komunikacja z zastosowaniem kolejki. Proszę uruchomić programy sekwencją poleceń:  
   ./fifo3 &  
   time ./fifo4
10. Proszę skompilować program „**ctrlc1.c**”. Po uruchomieniu program oczekuje na sygnał od użytkownika: przyciśnięcie kombinacji klawiszy ctrl+C. Proszę o przeanalizowanie działania programu gdy po jego uruchomieniu dwukrotnie przyciśnie się kombinację klawiszy ctrl+C.
11. Proszę skompilować i uruchomić programu „**alarm.c**”. Należy zwrócić uwagę na wywołanie funkcji **kill** i **pause**.
12. Proszę skompilować i uruchomić program „**ctrlc2.c**”. Po uruchomieniu programu należy przesłać kilkakrotnie do niego przerwanie od terminala (Ctrl+C), a następnie rozkaz zamknięcia terminala (Ctrl+\). Proszę porównać działanie tego programu z działaniem programu „**ctrlc1.c**”.
13. Proszę napisać program, który stosuje funkcję popen do uruchomienia programu „**polecenie1.c**” i przesłania na jego strumień wejściowy pewnego tekstu złożonego z dużych i małych liter.
14. Proszę napisać aplikację składającą się z procesów P1 i P2. Proces P2 jest procesem potomnym procesu P1. Proces P1 przekazuje przez potok nienazwany co 1 sekundę do P2 kolejne liczby 1,2,...,10 które mają być wyświetlane przez P2.
15. Proszę napisać 2 programy. Jeden z nich ma wysyłać do potoku nazwanego łańcuchy tekstowe o stałej długości (np. „tekst 01”, „tekst 02”,…) co sekundę przez 10 sekund. Drugi program ma odbierać takie łańcuchy z tego samego potoku. Kolejki mają być otwarte w trybie blokującym. Proszę przeanalizować działanie systemu złożonego z 2 programów wysyłających i jednego programu odbierającego.
16. Napisz dwa procesy – macierzysty i potomny. Proces macierzysty czeka w pętli na sygnał. Proces potomny generuje cykliczne sygnały (za pomocą funkcji kill).