**Programowanie współbieżne i rozproszone – laboratorium**

**Instrukcja do ćwiczenia laboratoryjnego III**

**Problem czytelników i pisarzy, 5 filozofów,
programowanie interfejsu socket**

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 skompilować, uruchomić i przeanalizować przykładowy program „writer\_reader1.c”. Program jest rozwiązaniem problemu czytelników i pisarzy z uprzywilejowaniem czytelników. Proszę zaobserwować działanie programu, kiedy w programie wątku czytelnika usunie się funkcję „sleep(1)”. Można wtedy zaobserwować głodzenie pisarza.
2. Proszę skompilować, uruchomić i przeanalizować przykładowy program „writer\_reader2.c”. Program jest rozwiązaniem problemu czytelników i pisarzy z uprzywilejowaniem pisarzy. Proszę zaobserwować działanie programu, kiedy w programie wątku czytelnika usunie się funkcję „sleep(1)”.
3. Proszę skompilować, uruchomić i przeanalizować przykładowy program „five\_phil1.c”. Program jest rozwiązaniem problemu 5 ucztujących filozofów z możliwością impasu.
4. Proszę skompilować, uruchomić i przeanalizować przykładowy program „five\_phil2.c”. Program jest rozwiązaniem problemu ucztujących filozofów, ale zachowanie filozofów (parzystych i nieparzystych) się różni.
5. Proszę zaproponować rozwiązanie problemu czytelników i pisarzy z zastosowaniem blokad czytelników i pisarzy z biblioteki pthreads (POSIX).
6. Proszę zaproponować rozwiązanie problemu 5 ucztujących filozofów z zastosowaniem „kelnera” lub zmiennych warunkowych, przy założeniu, że zachowanie każdego z filozofów będzie identyczne.
7. Proszę skompilować, uruchomić i przeanalizować przykładowe programy „sock\_kient1.c” i „sock\_serwer1.c” . Komunikacja pomiędzy klientem, a serwerem odbywa się z zastosowaniem gniazda odwzorowanego w lokalnym systemie plików. Proszę wypróbować pracę sytemu powołując do pracy 1 serwera i kilku klientów.
8. Proszę skompilować, uruchomić i przeanalizować przykładowe programy „sock\_kient2.c” i „sock\_serwer2.c”. Komunikacja pomiędzy klientem a serwerem odbywa się z zastosowaniem gniazd sieci Internet. Proszę wypróbować pracę sytemu powołując do pracy 1 serwera i kilku klientów. Proszę zwrócić uwagę na różnice w wartościach portu zdefiniowanego na poziomie programu oraz numeru portu powiązanego z aplikacją otrzymanego z zastosowaniem polecenia **netstat**.
9. Proszę skompilować, uruchomić i przeanalizować przykładowe programy „sock\_kient3.c” i „sock\_serwer3.c”. Komunikacja pomiędzy klientem a serwerem odbywa się z zastosowaniem gniazd sieci Internet. Proszę wypróbować pracę sytemu powołując do pracy 1 serwera i kilku klientów. Proszę zwrócić uwagę na brak różnic w wartościach portu zdefiniowanego na poziomie programu oraz numeru portu powiązanego z aplikacją otrzymanego z zastosowaniem polecenia **netstat**.
10. Proszę skompilować, uruchomić i przeanalizować przykładowy program „sock\_serwer4.c”. Program może współpracować z klientem zdefiniowanym w programie „sock\_kient3.c”. Proszę z zastosowaniem polecenia **ps** przeanalizować ilość procesów serwera, gdy do serwera próbuje łączyć się kilku klientów.
11. Proszę skompilować, uruchomić i przeanalizować przykładowy program „sock\_serwer5.c”. Program może współpracować z klientem zdefiniowanym w programie „sock\_kient3.c”. Proszę przeanalizować pracę serwera współpracującego z kilkoma klientami.
12. Proszę przetestować działanie programów „sock\_kient3.c”, „sock\_serwer3.c” , „ sock\_serwer4.c”, „sock\_serwer5.c” w środowisku sieci lokalnej współpracując z sąsiednim zespołem, lub stosując dodatkowy komputer.
13. Proszę napisać rozproszoną aplikację o następującej specyfikacji:

- system składa się z 1 serwera i 3 klientów

- klienci co 1 sekundę przesyłają do serwera komunikat złożony z tekstu i numeru komunikatu i numeru klienta (struktura {1,2,”tekst1”})

- serwer odbiera i wypisuje komunikaty na ekranie

- komunikacja odbywa się ciągle przez jeden ustalony zestaw gniazd (nie powinny następować kolejne rozłączenia i przyłączenia, należy monitorować lub wypisywać porty, na których następuje komunikacja)