Sieci Komputerowe 2021/22

Projekt zaliczeniowy II: Księga skarg i wniosków

Napisz programy klienta i serwera realizujące „księgę skarg i wniosków" za pomocą **pamięci współdzielonej** Systemu V, z dostępem synchronizowanym poprzez **semafory** (opcjonalnie). Dokładniej:

1. Serwer

- uruchamiany z **dwoma parametrami**:
  - nazwa pliku (służąca do generowania klucza IPC),
  - liczba całkowita n oznaczająca rozmiar księgi, tj. maksymalną liczbę wpisów;
- tworzy klucz IPC na podstawie pliku podanego jako argument powyżej;
- korzystając z powyższego klucza tworzy segment pamięci dzielonej zawierający n rekordów („slotów") do przechowywania wpisów;
- ewentualnie tworzy drugi klucz (korzystając z tego samego pliku co wyżej) i za jego pomocą odpowiedni semafor / zbiór semaforów;
- każdy rekord w księdze zawiera nazwę użytkownika klienta i jego wpis, ew. inne techniczne dane pozwalające poprawnie zrealizować zakładany scenariusz;
- czeka na sygnały od użytkownika:
  - Ctrl+Z powoduje wypisanie aktualnej zawartości księgi na ekran (tylko niepuste wpisy) jak w poniższych przykładach;
  - Ctrl+C powoduje zakończenie pracy serwera (poprzedzone „posprzątaniem").

2. Klient

- również uruchamiany z **dwoma parametrami**:
  - nazwa pliku (służąca do generowania klucza IPC),
  - nazwa użytkownika (pod tą nazwą będzie widniał jego wpis w księdze);
- dołącza uprzednio utworzony przez serwer segment pamięci dzielonej zawierający księgę (i ew. semafory);
- wyświetla informację ile jest jeszcze wolnych rekordów w księdze (por. przykłady poniżej);
- jeżeli w księdze nie ma już wolnego miejsca, to informuje o tym użytkownika i kończy pracę;
- jeżeli jeszcze jest wolne miejsce w księdze, to pyta użytkownika o komunikat, a następnie wpisuje ten komunikat w pierwsze wolne miejsce w księdze (wraz z dodatkowymi danymi pozwalającymi poprawnie zrealizować zakładany scenariusz – m.in. klient powinien przekazać nazwę użytkownika – drugi argument programu), dbając o poprawną synchronizację dostępu do księgi zgodnie z wybranym wariantem rozwiązania opisanym poniżej.

Warianty rozwiązania:

Implementujemy jeden z poniższych trzech wariantów (o rosnących stopniach trudności i rosnącej maksymalnej liczbie punktów). Wybrany wariant powinien być opisany w pierwszym komunikacie serwera jak w przykładzie poniżej.

- **WARIANT C**: implementacja bez synchronizacji semaforowej: **maks. 6 pkt.**,
- **WARIANT B**: synchronizacja z użyciem jednego semafora dla całej księgi, dbającego o to, by w danym momencie co najwyżej jeden proces czytał z / pisał do księgi; **maks. 8 pkt.**,
- **WARIANT A**: synchronizacja z użyciem n oddzielnych semaforów, po jednym dla każdego rekordu („slota") w księdze; **maks. 10 pkt**.

Należy zadbać o:

- obsługę błędów i inne „dobre praktyki" programistyczne opisane na ostatniej stronie w sekcji *Jak przygotować dobry projekt zaliczeniowy?*,
- precyzyjne i estetycznie sformatowane komunikaty informujące użytkownika na każdym etapie co się dzieje, czego program oczekuje, jakie opcje / skróty klawiaturowe są dostępne (patrz przykłady),
- obsługa polskich znaków przez program nie jest wymagana.

Film:

- należy przygotować i przesłać film z prezentacją programu, wskazówki jak przygotować film znajdują się na stronie wykładowcy,
- podkreślmy, że film powinien trwać 5-8 min. (filmy trwające więcej niż 10 min. nie będą akceptowane) i mieć formę screencastu z komentarzem głosowym,
- na filmie powinno znaleźć się:
    - krótkie (!) omówienie kodu źródłowego (proszę nie omawiać każdej linii kodu ani rzeczy elementarnych, należy krótko skupić się na kilku wybranych najważniejszych fragmentach kodu / pomysłach, na które (subiektywnie) warto zwrócić uwagę, np. fragmenty dotyczące pracy z obiektami IPC, synchronizacją, organizacja pamięci itp.),
    - prezentacja kompilacji,
    - pokaz działania – główna część filmu (treść w zależności od wariantu, szczegóły patrz niżej).

Punktacja:

Poniżej specyfikuję dokładniej za co będą przyznawane punkty (równoważnie, za brak czego można stracić punkty). Warunkiem koniecznym przyznania punktów za daną rzecz jest to, że prezentacja tej rzeczy będzie obecna na filmie.

WARIANT C [razem maks. 6 punktów]:

- kody serwera i klienta: estetyka, obecność komentarzy, obsługa błędów przynajmniej w kluczowych miejscach, kompilacja obu kodów z opcją `-Wall` bez warningów: **1 punkt**,
- odpowiednia reakcja na wywołanie klienta oraz serwera z błędnymi argumentami (na filmie po jednym przykładzie takiego wywołania dla klienta i serwera): **0,5 punkta**,
- interfejs serwera: estetyka i zgodność ze specyfikacją (w tym poprawne wyświetlanie księgi po Ctrl+Z): **1 punkt**,
- interfejs klienta: estetyka i zgodność ze specyfikacją (w tym poprawna reakcja na brak miejsca w księdze): **1 punkt**,
- poprawne zamykanie serwera (poprzez Ctr+C) wraz z prezentacją, że obiekty IPC znikają z systemu po zamknięciu serwera: **1 punkt**,
- poprawne umieszczanie komunikatu w pierwszym wolnym slocie: **1,5 punkta**,
- **UWAGA 1**: programy będą także testowane przez sprawdzającego; jeśli zostaną wykryte jakieś rażące błędy (np. program zakończy się błędem wykonania i zrzutem obrazu pamięci (`Core dumped`)), wówczas, oprócz utraty punktów wyżej, można dostać dodatkowo do **-1 punkta** (przy czym proszę się nie obawiać, nie będę testował sytuacji zupełnie skrajnych typu 3 klientów chce jednocześnie pisać do ostatniego slota a jeden klient w trakcie pisania zostaje zabity sygnałem itp.; testuję typowe scenariusze – choć oczywiście powinniśmy dążyć do tego by nasze programy były odporne również na skrajne sytuacje);
- **UWAGA 2**: kody uczestników będą też porównywane między sobą, wykrycie plagiatu skutkuje niezaliczeniem obu wersji, bez dociekania kto był autorem, a kto plagiatorem.

WARIANT B [razem maks. 8 punktów]:

- wszystkie wymagania jak w Wariancie C (**6 punktów**) oraz:
- prezentacja poprawnej synchronizacji dla dwóch klientów uruchomionych „jednocześnie" w wersji dla jednego semafora: **2 punkty**.

WARIANT A [razem maks. 10 punktów]:

- wszystkie wymagania jak w Wariancie C (**6 punktów**) oraz:
- prezentacja poprawnej synchronizacji dla dwóch klientów uruchomionych „jednocześnie" w wersji dla n semaforów: **4 punkty**.

Uwaga: dokładniejsze wyjaśnienie czym jest *poprawna synchronizacja* w Wariantach B i A powyżej – patrz **pokaz i dyskusja** na zajęciach.

Przesyłka:

Należy przesłać (w terminie, za pomocą mechanizmu Moodle poniżej) trzy pliki:

- kod klienta (plik o nazwie `NazwiskoImie_klientksiega.c`),
- kod serwera (plik o nazwie `NazwiskoImie_serwerksiega.c`),
- film (plik o nazwie `NazwiskoImie_prezentacja.???`), gdzie w miejscu `NazwiskoImie` należy wpisać swoje nazwisko i imię.

Proszę nie przysyłać wszystkiego w jednym skompresowanym pliku, proszę by były trzy oddzielne pliki j.w.

Przykłady:
Uwaga: Na niebiesko zaznaczone są dane podawane przez użytkownika.

Przykładowe wykonania klientów:

```
jan@xxx> klientksiega /admin/serwerksiega jasio
Klient ksiegi skarg i wnioskow wita!
[Wolnych 10 wpisow (na 10)]
Napisz co ci doskwiera:
> lubudubu lubudubu niech zyje nam admin!
Dziekuje za dokonanie wpisu do ksiegi

jan@xxx> klientksiega /admin/serwerksiega anonim
Klient ksiegi skarg i wnioskow wita!
[Wolnych 9 wpisow (na 10)]
Napisz co ci doskwiera:
> wszystko dziala szybko a system jest niezawodny
Dziekuje za dokonanie wpisu do ksiegi

kazimierz@xxx> klientksiega /admin/serwerksiega krnabrnykazio
Klient ksiegi skarg i wnioskow wita!
[Wolnych 8 wpisow (na 10)]
Napisz co ci doskwiera:
> admin jest gupi
Dziekuje za dokonanie wpisu do ksiegi
```

Odpowiadające powyższym przykładowe wykonanie serwera:

```
admin@xxx> ./serwerksiega ./serwerksiega 10
[Serwer]: ksiega skarg i wnioskow (WARIANT B)
[Serwer]: tworze klucz na podstawie pliku ./serwerksiega... OK (klucz: 16915235)
[Serwer]: tworze segment pamieci wspolnej dla ksiegi na 10 wpisow po 128b...
          OK (id: 9371648, rozmiar: 1640b)
[Serwer]: dolaczam pamiec wspolna... OK (adres: 800640000)
[Serwer]: nacisnij Crtl^Z by wyswietlic stan ksiegi
[Serwer]: nacisnij Crtl^C by zakonczyc program
^Z
Ksiega skarg i wnioskow jest jeszcze pusta
^Z
----------- Ksiega skarg i wnioskow: -----------
[jasio]: lubudubu lubudubu niech zyje nam admin!
[anonim]: wszystko dziala szybko a system jest niezawodny
^Z
----------- Ksiega skarg i wnioskow: -----------
[jasio]: lubudubu lubudubu niech zyje nam admin!
[anonim]: wszystko dziala szybko a system jest niezawodny
[krnabrnykazio]: admin jest gupi
^C
[Serwer]: dostalem SIGINT => koncze i sprzatam...
          (odlaczenie: OK, usuniecie: OK)
```

Jak przygotować dobry projekt zaliczeniowy?

Programistyczne zadanie „zaliczeniowe" zostanie ocenione na maksymalną liczbę punktów, jeśli spełnia wszystkie poniższe warunki:

- Program działa poprawnie (w pełni zgodnie ze specyfikacją) na serwerach dostępnych w sieci wydziałowej.
- Kod kompiluje się poprawnie z opcją `-Wall` (*print all warnings*) bez wyświetlania ostrzeżeń.
- Program jest napisany samodzielnie i autor rozumie użyte techniki.
- Kod jest napisany estetycznie i czytelnie (odstępy, wcięcia, czytelne nazwy zmiennych, itp.).
- Kod jest dobrze udokumentowany (komentarze opisujące najważniejsze fragmenty kodu i najważniejsze zmienne).
- Jest zaimplementowana obsługa błędów wywołania funkcji systemowych oraz odpowiednia reakcja na niepoprawne dane od użytkownika.
- Wyjściowe komunikaty wypisywane przez program są czytelne, zrozumiałe i estetyczne (w szczególności, komunikaty podają wszystkie informacje niezbędne dla użytkownika).
- Jeżeli kod źródłowy składa się z więcej niż jednego pliku lub wymaga dodatkowych („niestandardowych") opcji kompilacji, to wówczas należy dołączyć odpowiedni plik `Makefile`.
- Do plików źródłowych dołączony jest krótki (5-8 minutowy) film zawierający prezentację projektu.