IPC - kolejki komunikatów

Przydatne funkcje:

System V:

```
<sys/msg.h> <sys/ipc.h> - msgget, msgctl, msgsnd, msgrcv, ftok
```

POSIX:

```
<mqueue.h> - mq_open, mq_send, mq_receive, mq_getattr, mq_setattr, mq_close,
mq_unlink, mq_notify
```

Zadanie 1. Prosty chat - System V (50%)

Napisz prosty program typu klient-serwer, w którym komunikacja zrealizowana jest za pomocą kolejek komunikatów.
Serwer po uruchomieniu tworzy nową kolejkę komunikatów systemu V. Za pomocą tej kolejki klienci będą wysyłać komunikaty do serwera. Wysyłane zlecenia mają zawierać rodzaj zlecenia jako rodzaj komunikatu oraz informację od którego klienta zostały wysłane (ID klienta), w odpowiedzi rodzajem komunikatu ma być informacja identyfikująca czekającego na nią klienta.
Klient bezpośrednio po uruchomieniu tworzy kolejkę z unikalnym kluczem IPC i wysyła jej klucz komunikatem do serwera (komunikat INIT). Po otrzymaniu takiego komunikatu, serwer otwiera kolejkę klienta, przydziela klientowi identyfikator (np. numer w kolejności zgłoszeń) i odsyła ten identyfikator do klienta (komunikacja w kierunku serwer->klient odbywa się za pomocą kolejki klienta). Po otrzymaniu identyfikatora, klient może wysłać zlecenie do serwera(zlecenia są czytane ze standardowego wyjścia w postaci typ_komunikatu).

Rodzaje zleceń

- LIST:
  Zlecenie wypisania listy wszystkich aktywnych klientów.
- 2ALL string:
  Zlecenie wysłania komunikatu do wszystkich pozostałych klientów. Klient wysyła ciąg znaków. Serwer wysyła ten ciąg wraz z identyfikatorem klienta-nadawcy oraz aktualną datą do wszystkich pozostałych klientów.
- 2ONE id_klienta string:
  Zlecenie wysłania komunikatu do konkretnego klienta. Klient wysyła ciąg znaków podając jako adresata konkretnego klienta o identyfikatorze z listy aktywnych klientów. Serwer wysyła ten ciąg wraz z identyfikatorem klienta-nadawcy oraz aktualną datą do wskazanego klienta.
- STOP:
  Zgłoszenie zakończenia pracy klienta. Klient wysyła ten komunikat, kiedy kończy pracę, aby serwer mógł usunąć z listy jego kolejkę. Następnie kończy pracę, usuwając swoją kolejkę. Komunikat ten wysyłany jest również, gdy po stronie klienta zostanie wysłany sygnał SIGINT.

Serwer powinien zapisywać do pliku czas otrzymania zlecenia, identyfikator klienta i treść komunikatu.
Zlecenia powinny być obsługiwane zgodnie z priorytetami,najwyższy priorytet ma STOP, potem LIST i reszta. Można tego dokonać poprzez sterowanie parametrem MTYPE w funkcji msgsnd.
Poszczególne rodzaje komunikatów należy identyfikować za pomocą typów komunikatów systemu V. Klucze dla kolejek mają być generowane na podstawie ścieżki `$HOME`. Małe liczby do wygenerowania kluczy oraz rodzaje komunikatów mają być zdefiniowane we wspólnym pliku nagłówkowym. Dla uproszczenia można założyć, że długość komunikatu jest ograniczona pewną stałą (jej definicja powinna znaleźć się w pliku nagłówkowym).
Klient i serwer należy napisać w postaci osobnych programów (nie korzystamy z funkcji fork). Serwer musi być w stanie pracować z wieloma klientami naraz. Przed zakończeniem pracy każdy proces powinien usunąć kolejkę którą utworzył (patrz IPC_RMID oraz funkcja atexit). Dla uproszczenia można przyjąć, że serwer przechowuje informacje o klientach w statycznej tablicy (rozmiar tablicy ogranicza liczbę klientów, którzy mogą się zgłosić do serwera).
Serwer może wysłać do klientów komunikaty:

- inicjujący pracę klienta
- wysyłający odpowiedzi do klientów (kolejki klientów)
- informujący klientów o zakończeniu pracy serwera - po wysłaniu takiego sygnału i odebraniu wiadomości STOP od wszystkich klientów serwer usuwa swoją kolejkę i kończy pracę. (kolejki klientów)

Należy obsłużyć przerwanie działania serwera lub klienta za pomocą CTRL+C. Po stronie klienta obsługa tego sygnału jest równoważna z wysłaniem komunikatu STOP.

Zadanie 2. Prosty chat - POSIX (50%)

Zrealizuj zadanie analogiczne do Zadania 1, wykorzystując kolejki komunikatów POSIX. Kolejka klienta powinna mieć losową nazwę zgodną z wymaganiami stawianymi przez POSIX. Na typ komunikatu można zarezerwować pierwszy bajt jego treści. Obsługa zamykania kolejek analogiczna jak w zadaniu 1, z tym, że aby można było usunąć kolejkę, wszystkie procesy powinny najpierw ją zamknąć. Przed zakończeniem pracy klient wysyła do serwera komunikat informujący, że serwer powinien zamknąć po swojej stronie kolejkę klienta. Następnie klient zamyka i usuwa swoją kolejkę. Serwer przed zakończeniem pracy zamyka wszystkie otwarte kolejki, informuje klientów, aby usunęli swoje kolejki oraz zamknęli kolejkę serwera i usuwa kolejkę, którą utworzył.