МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №8**

**з дисципліни «Операційні системи»**

**на тему: «Взаємодія процесів з використанням черги повідомлень»**

Виконала:

Студент гр. ПЗ2011

Кулик С.В.

Прийняв:

Андрющенко В.О.

Дніпро, 2023

**Тема:** Вивчення механізму сигналів ОС UNIX

**Мета:** Ознайомитися з механізмом сигналів UNIX (види сигналів, способи і засоби посилки сигналів). Отримати практичні навики програмування передачі і обробки сигналів. Отримати практичні навики використання сигналів.

**Короткі теоретичні відомості**

Черги повідомлень надають механізм, який дозволяє одному процесу опублікувати повідомлення, яке може отримати інший процес. Коли повідомлення отримане, воно віддаляється з черги. Черги повідомлень унікальні тим, що обидва процеси необов'язково повинні протікати одночасно - один процес може опублікувати повідомлення і завершитися, а повідомлення може бути отримане іншим процесом декількома днями пізніше до перезавантаження системи.

Cистемні виклики для роботи з чергою повідомлень:

int msgget ( key\_t key, int msgflg );

int msgsnd (int msgid, struct msgbuf \* msgp, int msgzs, int msgflg);

int msgrcv (int msgid, struct msgbuf \* msgp, int msgzs, long msgtype, int msgflg);

int msgctl (int shmid, int cmd, struct msgid\_ds \*buf).

**Текст програми**

**Файл server.cpp**

#include "msg.h"

#include <string.h>

int main() {

  message msg;

  answer answ;

  strcpy(answ.text, "OK!");

  int err, last\_id = 3;

  int mq\_id = msgget(KEY, IPC\_CREAT | 0666);

  cout << "server start" << endl;

  while(true) {

    if(mq\_id != -1) {

      err = msgrcv(mq\_id, &msg, MESSAGE\_SIZE, 0, 0);

      if(err == -1) return 0;

      if(msg.type == 1) {

        msg.client\_id = last\_id;

        last\_id += 2;

        msg.type = 2;

        err = msgsnd(mq\_id, &msg, MESSAGE\_SIZE, 0);

      }

      else if (msg.type  & 1) {

        cout << "server -> client "

        << (msg.client\_id - 1) / 2

        << " : " << msg.text

        << ";" << endl;

        answ.type = msg.type + 1;

        err = msgsnd(mq\_id, &answ, ANSWER\_SIZE, 0);

      }

    }

    else

      cout << "server -> can`t create message queue;" << endl;

  }

  return 0;

}

**Файл client.cpp**

#include "msg.h"

#include <string>

#include <unistd.h>

using namespace std;

int main() {

  message msg;

  answer answ;

  msg.type = 1;

  int mq\_id = msgget(KEY, 0);

  int err;

  cout << "client start" << endl;

  if (mq\_id != -1) {

    err = msgsnd(mq\_id, &msg, MESSAGE\_SIZE, 0);

    err = msgrcv(mq\_id, &msg, MESSAGE\_SIZE, 2, 0);

    if(err != -1) {

      msg.type = msg.client\_id;

      string end;

      do {

        cout << "client " <<  (msg.client\_id - 1) / 2  << " -> enter text: ";

        cin.getline(msg.text, 256);

        err = msgsnd(mq\_id, &msg, MESSAGE\_SIZE, 0);

        err = msgrcv(mq\_id, &answ,   ANSWER\_SIZE, msg.client\_id + 1, 0);

        cout << "client " << (msg.client\_id - 1) / 2 << " -> " << answ.text << endl;

        cout << "do you wand to continue? (yes or no)" << endl;

        getline(cin, end);

      } while (end == "yes" || end == "Yes");

    }

    else

      cout << "client -> server not found;" << endl;

  }

  else

    cout << "client -> none message queue found;" << endl;

  return 0;

}

**Файл msg.h**

#include <sys/types.h>

#include <sys/stat.h>

#include <sys/msg.h>

#include <sys/ipc.h>

#include <iostream>

using std::cout;

using std::endl;

using std::cin;

const int KEY = 2132;

struct message {

long type;

int client\_id;

char text[256];

};

struct answer {

long type;

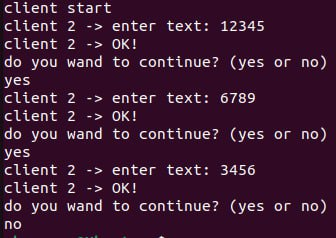
char text[4];

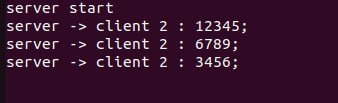
};

const int MESSAGE\_SIZE = sizeof(message) - sizeof(message::type);

const int ANSWER\_SIZE = sizeof(answer::text);

**Результати програми та тестові прилади**





**Висновки**

У результаті виконання лабораторної роботи ознайомився з механізмом сигналів UNIX (види сигналів, способи і засоби посилки сигналів). Отримано практичні навики програмування передачі і обробки сигналів. Отримано практичні навики використання сигналів.