Написать 2 программы, выполняемые под управлением ОС Linux.

Язык: C++. Для разделяемой памяти, потоков, семафоров использовать функции POSIX.

Описание первой программы:

1) Создаёт разделяемую память.
2) Принимает введённые пользователем с клавиатуры строки и складывает их в очередь в разделяемой памяти

Описание второй программы:

1) Состоит из двух потоков
2) Первый поток подключается к разделяемой памяти, считывает из очереди строки и записывает их в файл (название определяется программистом)
3) Второй поток по мере наполнения файла строками сортирует его содержимое по возрастанию

Обязательно использование ООП.

Первая программа first.cpp

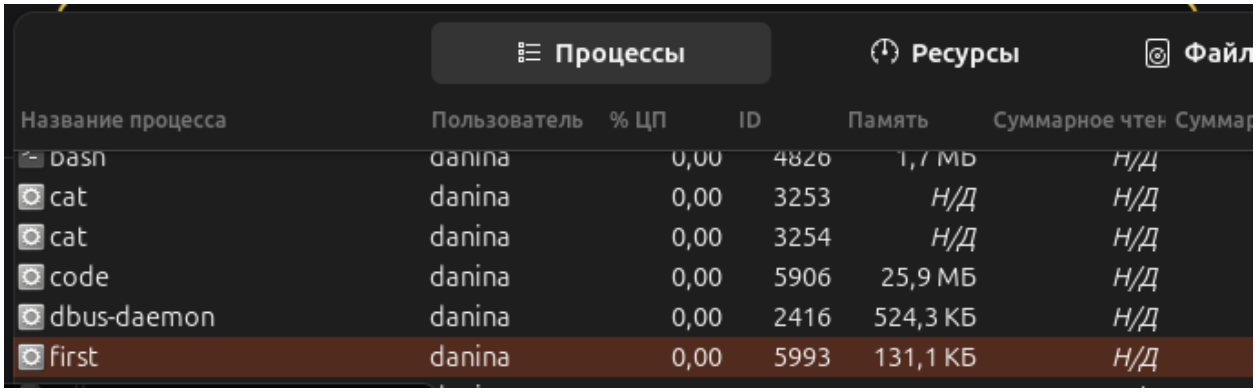
```cpp
#include <iostream>
#include <string>
#include <cstring>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <fcntl.h>
#include <semaphore.h>
int main() {
    //Размер
    const int shmSize = 1024;
    //Ключ
    key_t key = 2025;
    //Создание сегмента, IPC_EXCL возвращает ошибку
    int shmid = shmget(key, shmSize, IPC_CREAT | 0666);
    //Пролёт в создании
    if (shmid == -1) {
        perror("shmget");
        return 1;
    }
    //Его подключение
    char* data = static_cast<char*>(shmat(shmid, nullptr, 0));
    //Ошибка подключения
    if (data == (char*)-1) {
        perror("shmat");
        return 1;
    }
    //Создание семафора (имя, флаг, права доступа, разблокировка)
    sem_t* sem = sem_open("/my_semaphore", O_CREAT, 0666, 1);
    //Проверка открытия семафора
    if (sem == SEM_FAILED) {
        perror("sem_open");
        return 1;
    }
    /*Бесконечный цикл ввода значений,
      до тех пора пока не будет ввода "exit"*/
    std::string input;
    while (true) {
        std::cout << "Введите строку или \"exit\" для выхода: ";
        std::getline(std::cin, input);
        if (input == "exit") break;
        sem_wait(sem);
        std::strcpy(data, input.c_str());
        sem_post(sem);
    }
    //Закрытие семафора
    sem_close(sem);
    //Удаление имени семафора
    sem_unlink("/my_semaphore");
    //Откл. сегмента
    shmdt(data);
    //Удаления сегмента
    shmctl(shmid, IPC_RMID, nullptr);
    return 0;
}
```

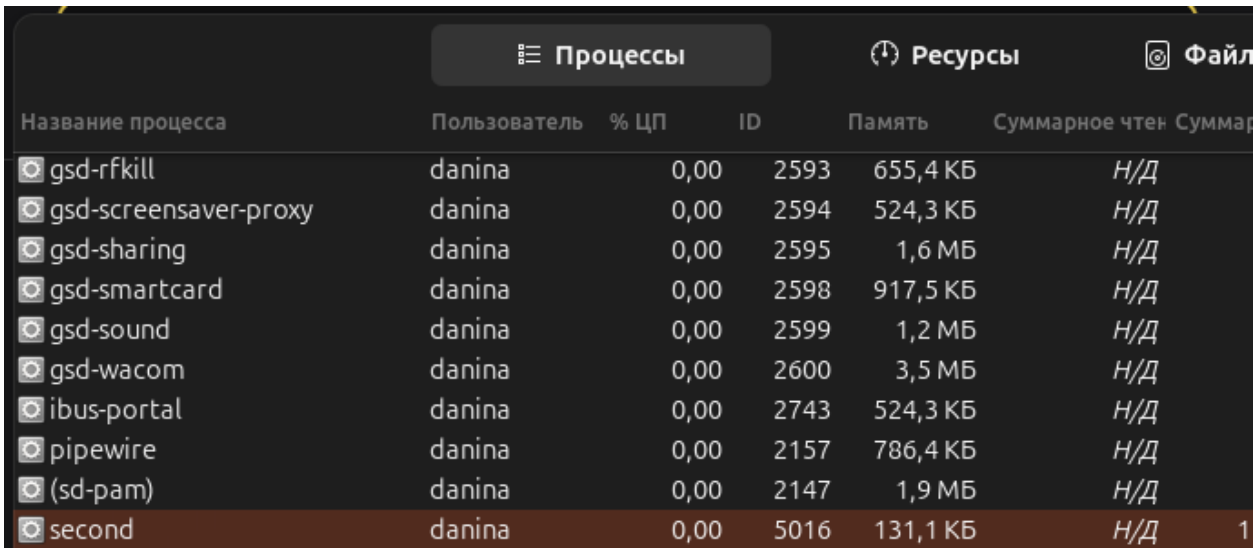
Вторая программа second.cpp

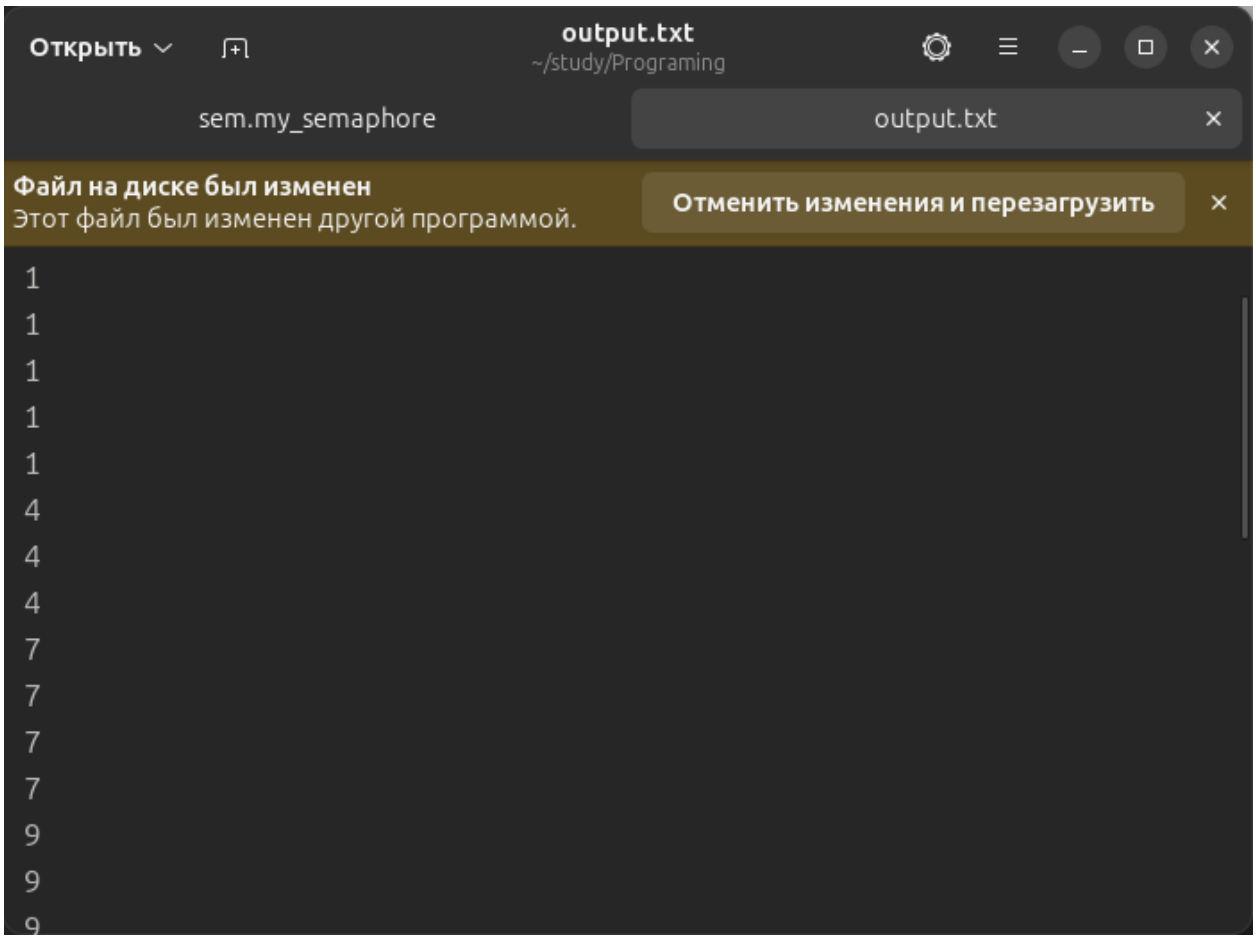
```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <algorithm>
#include <cstring>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <fcntl.h>
#include <semaphore.h>
#include <pthread.h>
#include <unistd.h>
//Запись имени файла
const std::string filename = "output.txt";
sem_t* file_sem;
//Поток записи
void* writerThread(void* arg) {
    //Указатель на разделяемую память
    char* data = static_cast<char*>(arg);
    //Подключение к уже существующему семафору
    sem_t* shm_sem = sem_open("/my_semaphore", 0);
    //Открытие файла в режиме добавления
    std::ofstream file(filename, std::ios::app);
    if (!file.is_open()) {
        std::cerr << "Ошибка открытия файла!" << std::endl;
        return nullptr;
    }
    while (true) {
        //Блокировка доступа разделяемой памяти
        sem_wait(shm_sem);
        //Чтение
        std::string str(data);
        //Разблокировка
        sem_post(shm_sem);
        if (!str.empty()) {
            //Блокировка
            sem_wait(file_sem);
            //Запись
            file << str << std::endl;
            //Принудительная запись
            file.flush();
            //Разблокировка
            sem_post(file_sem);
        }
        sleep(1);
    }
    //Закрытие семафора
    sem_close(shm_sem);
    return nullptr;
}
//Поток сортировки
void* sorterThread(void* arg) {
    while (true) {
        //Выполнение каждые 5 секунд
        sleep(5);
        //Блокировка файла
        sem_wait(file_sem);
        //Чтение всех строк файла
        std::ifstream inFile(filename);
        std::vector<std::string> lines;
        std::string line;
        while (std::getline(inFile, line)) {
            if (!line.empty()) lines.push_back(line);
        }
        inFile.close();
        //Сортировка
        std::sort(lines.begin(), lines.end());
        std::ofstream outFile(filename);
        for (const auto& l : lines) {
            outFile << l << std::endl;
        }
        outFile.close();
        //Разблокировка файла
        sem_post(file_sem);
    }
    return nullptr;
}
int main() {
    //Размер
    const int shmSize = 1024;
    //Ключ
    key_t key = 2025;
    //Создание сегмента, IPC_EXCL возвращает ошибку
    file_sem = sem_open("/file_semaphore", O_CREAT, 0666, 1);
    if (file_sem == SEM_FAILED) {
        perror("sem_open(file)");
        return 1;
```

```cpp
    if (file_sem == SEM_FAILED) {
        perror("sem_open(file)");
        return 1;
    }
    // Подключение к разделяемой памяти
    int shmid = shmget(key, shmSize, 0666);
    if (shmid == -1) {
        perror("shmget");
        return 1;
    }
    // Подключение памяти
    char* data = static_cast<char*>(shmat(shmid, nullptr, 0));
    if (data == (char*)-1) {
        perror("shmat");
        return 1;
    }
    //Поток записи
    pthread_t writer_thread, sorter_thread;
    if (pthread_create(&writer_thread, nullptr, writerThread, data) != 0) {
        std::cerr << "Ошибка создания потока записи!" << std::endl;
        return 1;
    }
    //Поток сортировки
    if (pthread_create(&sorter_thread, nullptr, sorterThread, nullptr) != 0) {
        std::cerr << "Ошибка создания потока сортировки!" << std::endl;
        return 1;
    }
    //Ожидание заврешение потока (бесконечно)
    pthread_join(writer_thread, nullptr);
    pthread_join(sorter_thread, nullptr);
    //Освобождение ресурсов
    sem_close(file_sem);
    sem_unlink("/file_semaphore");
    shmdt(data);
    return 0;
}
```

Проверка




Процессы
Ресурсы
Файл
Название процесса
Пользователь
% ЦП
ID
Память
Суммарное чтен
Суммар
gsd-rfkill danina 0,00 2593 655,4 КБ Н/Д
gsd-screensaver-proxy danina 0,00 2594 524,3 КБ Н/Д
gsd-sharing danina 0,00 2595 1,6 МБ Н/Д
gsd-smartcard danina 0,00 2598 917,5 КБ Н/Д
gsd-sound danina 0,00 2599 1,2 МБ Н/Д
gsd-wacom danina 0,00 2600 3,5 МБ Н/Д
ibus-portal danina 0,00 2743 524,3 КБ Н/Д
pipewire danina 0,00 2157 786,4 КБ Н/Д
(sd-pam) danina 0,00 2147 1,9 МБ Н/Д
second danina 0,00 5016 131,1 КБ Н/Д 1


Открыть
output.txt
~/study/Programing
sem.my_semaphore
output.txt
Файл на диске был изменен
Этот файл был изменен другой программой.
Отменить изменения и перезагрузить
1
1
1
1
1
4
4
4
7
7
7
7
9
9
9


Ubuntu / dev / shm
sem.file_semaphore
sem.my_semaphore