Минобрнауки России  
Федеральное государственное автономное образовательное  
Учреждение высшего образования  
«Санкт-Петербургский государственный электротехнический  
Университет им. В.И. Ульянова (Ленина)»  
(СПГЭТУ «ЛЭТИ»)  
Факультет компьютерных технологий и информатики

Кафедра вычислительной техники  
  
  
  
**Отчет по лабораторной работе №2**

**на тему: «Управление памятью»**

**по дисциплине «Операционные системы»**

Выполнил студент группы 9308: Семенов А.И.

Принял: к.т.н., доцент Тимофеев А.В.

Санкт-Петербург  
2021 г.

Оглавление

[Цель работы 3](#_Toc84108923)

[Управление дисками, каталогами и файлами 3](#_Toc84108924)

[Указания к выполнению 3](#_Toc84108925)

[Примеры работы программы 4](#_Toc84108926)

[Копирование файла с помощью операций перекрывающегося ввода-вывода 10](#_Toc84108927)

[Указания к выполнению 10](#_Toc84108928)

[Примеры выполнения программы 11](#_Toc84108929)

[Измерения времени на копирование и анализ 11](#_Toc84108930)

[Вывод 16](#_Toc84108931)

# Цель работы

Исследовать механизмы управления виртуальной памятью Win32.

# Исследование виртуального адресного пространства процесса

## Указания к выполнению

Создайте консольное приложение с меню (каждая выполняемая функция и/или операция должна быть доступна по отдельному пункту меню), которое выполняет:

* получение информации о вычислительной системе (функция Win32 API – **GetSystemInfo**);
* определение статуса виртуальной памяти (функция Win32 API – **GlobalMemoryStatus**);
* определение состояния конкретного участка памяти по заданному с клавиатуры адресу (функция Win32 API – **VirtualQuery**);
* резервирование региона в автоматическом режиме и в режиме ввода адреса начала региона (функция Win32 API – **VirtualAlloc**);
* резервирование региона и передача ему физической памяти в автоматическом режиме и в режиме ввода адреса начала региона (функция Win32 API – **VirtualAlloc**);
* запись данных в ячейки памяти по заданным с клавиатуры адресам;
* установку защиты доступа для заданного (с клавиатуры) региона памяти и ее проверку (функция Win32 API – **VirtualProtect**);
* возврат физической памяти и освобождение региона адресного пространства заданного (с клавиатуры) региона памяти (функция Win32 API –**VirtualFree**).

## Примеры работы программы

Программа может запущена с помощью флагов:

* -cb – конвертация байт в другие единицы измерения (максимально возможные)
* -hi – помощь в вводе констант в некоторых пунктах (вывод доступных констант на экран)
* -sa – вывод виртуальных адресов процесса в некоторых пунктах

При запуске программы открывается меню, показанное на рисунке 1.

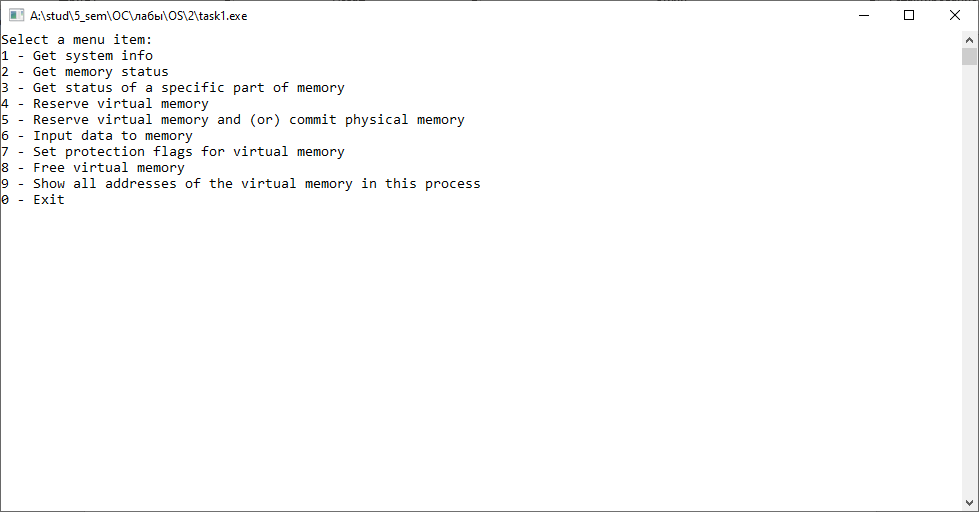


Рисунок . Главное меню

Получение информации о вычислительной системе осуществляется через 1-ый пункт меню. Результат получения информации представлен на рисунке 2.

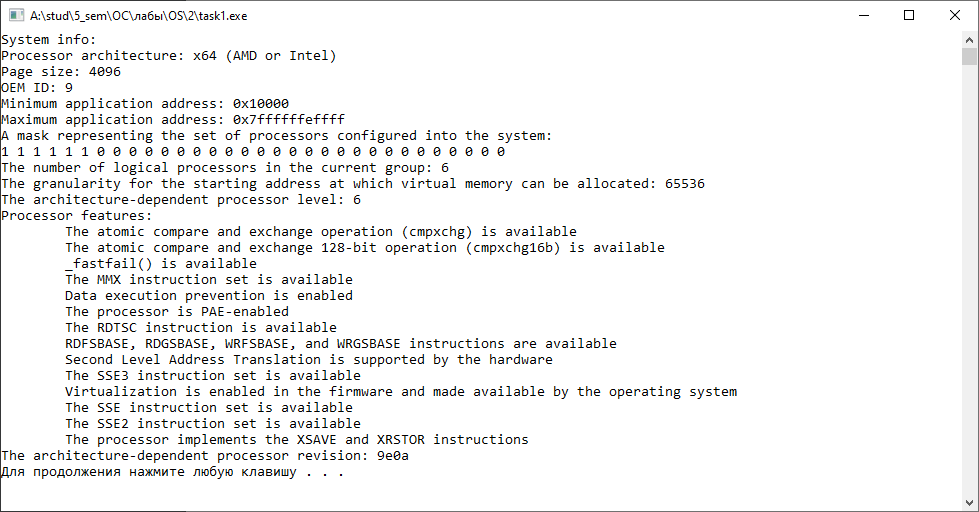


Рисунок . Получение информации о вычислительной системе

Определение статуса виртуальной памяти осуществляется через 2-ой пункт меню. Результат получения статуса представлен на рисунке 3.

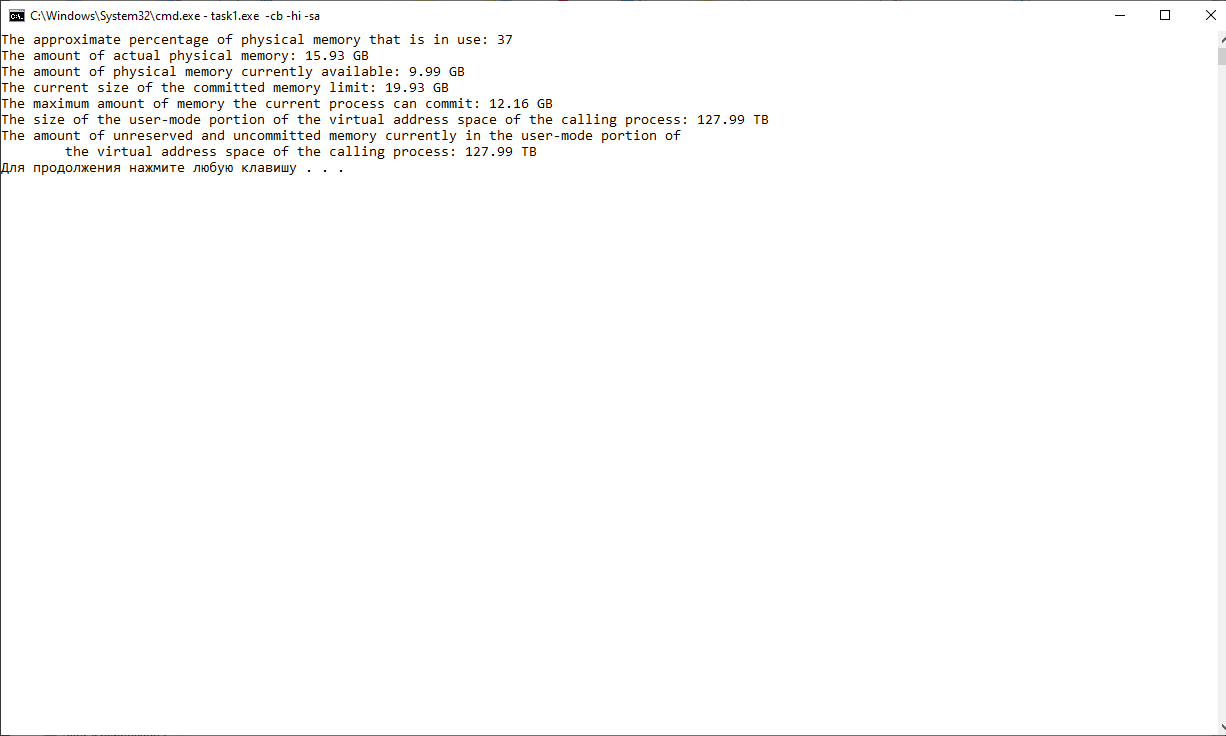


Рисунок . Получение статуса виртуальной памяти

Чтобы определить состояния конкретного участка памяти по указанному адресу, необходимо использовать пункт 3. Результат определения состояния представлен на рисунке 4.

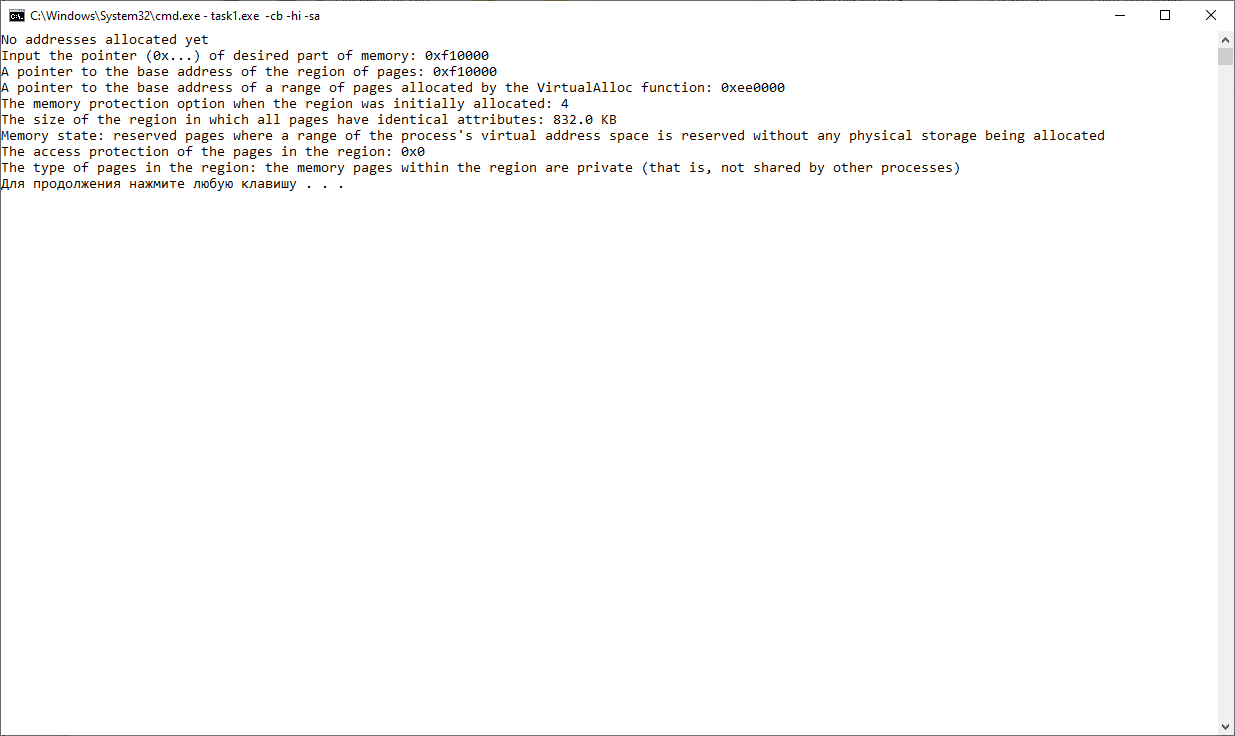


Рисунок . Определение состояния конкретного участка памяти

Для резервирования региона предназначен пункт 4 меню. Результат резервирования представлен на рисунках 5 и 6.

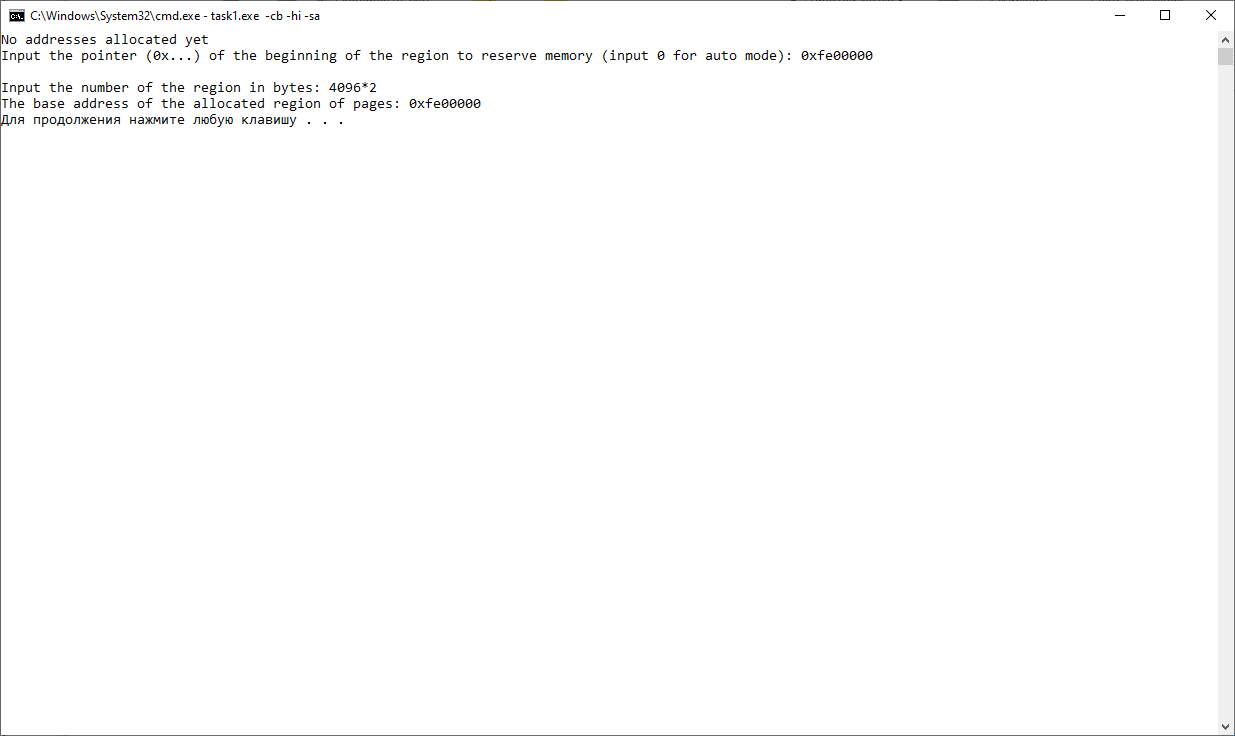


Рисунок . Резервирование региона по указанному адресу

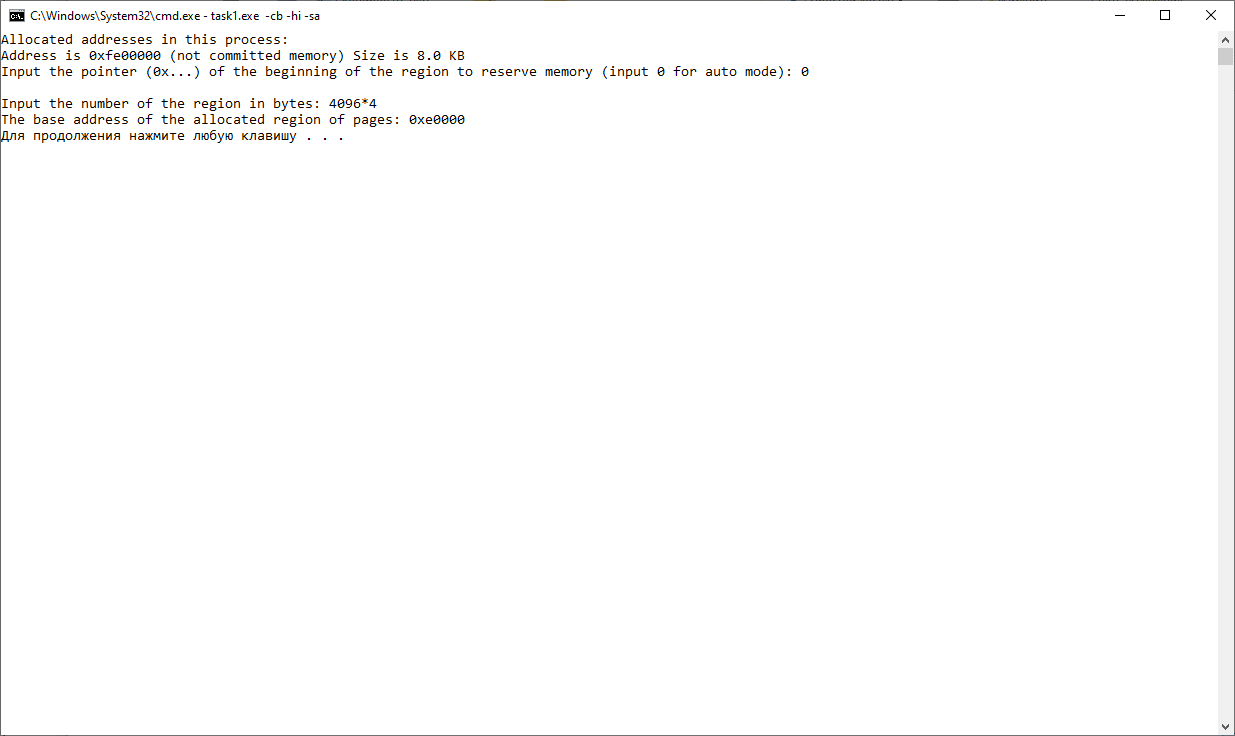


Рисунок . Резервирование региона в автоматическом режиме

Для резервирования и передачи региону физической памяти используется пункт 5 меню. Он работает в двух режимах:

1. Резервирование и передача физической памяти новому региону;
2. Передача физической памяти региону, который уже зарезервирован.

Результат работы данного пункта представлен на рисунках 7 и 8.

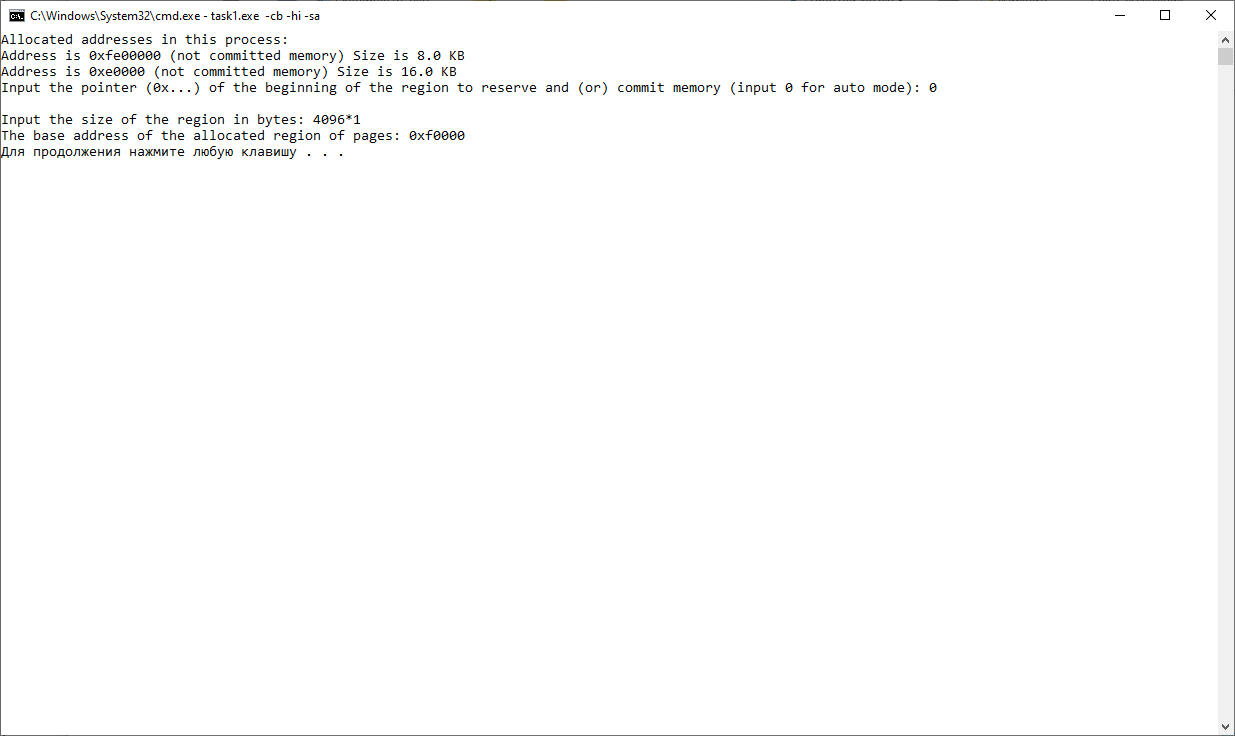


Рисунок . Резервирование и передача физической памяти новому региону

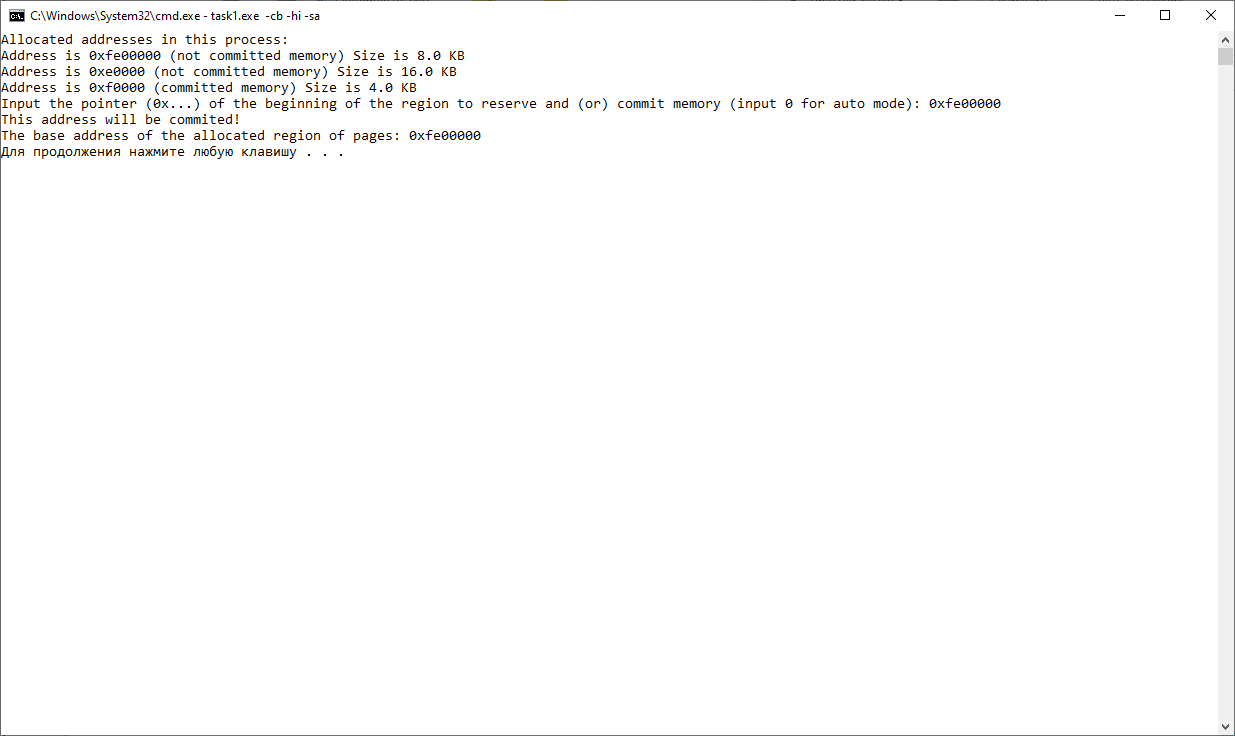


Рисунок . Передача физической памяти зарезервированному региону

Запись данных в ячейки памяти по заданному адресу производится в пункте 6 меню. Результат работы записи представлен на рисунке 9.

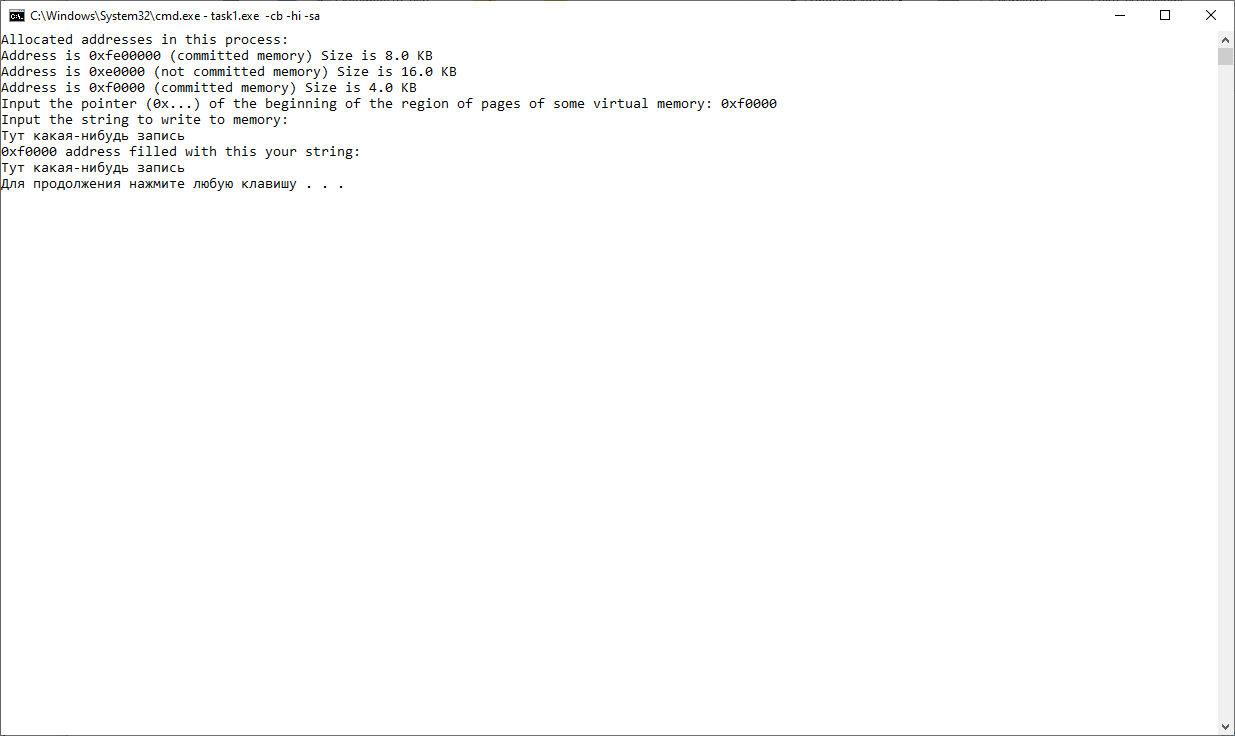


Рисунок . Запись данных в ячейки памяти по заданному адресу

Установка защиты доступа для заданного региона памяти производится через пункт 7 меню. Пример установки защиты представлен на рисунке 10.

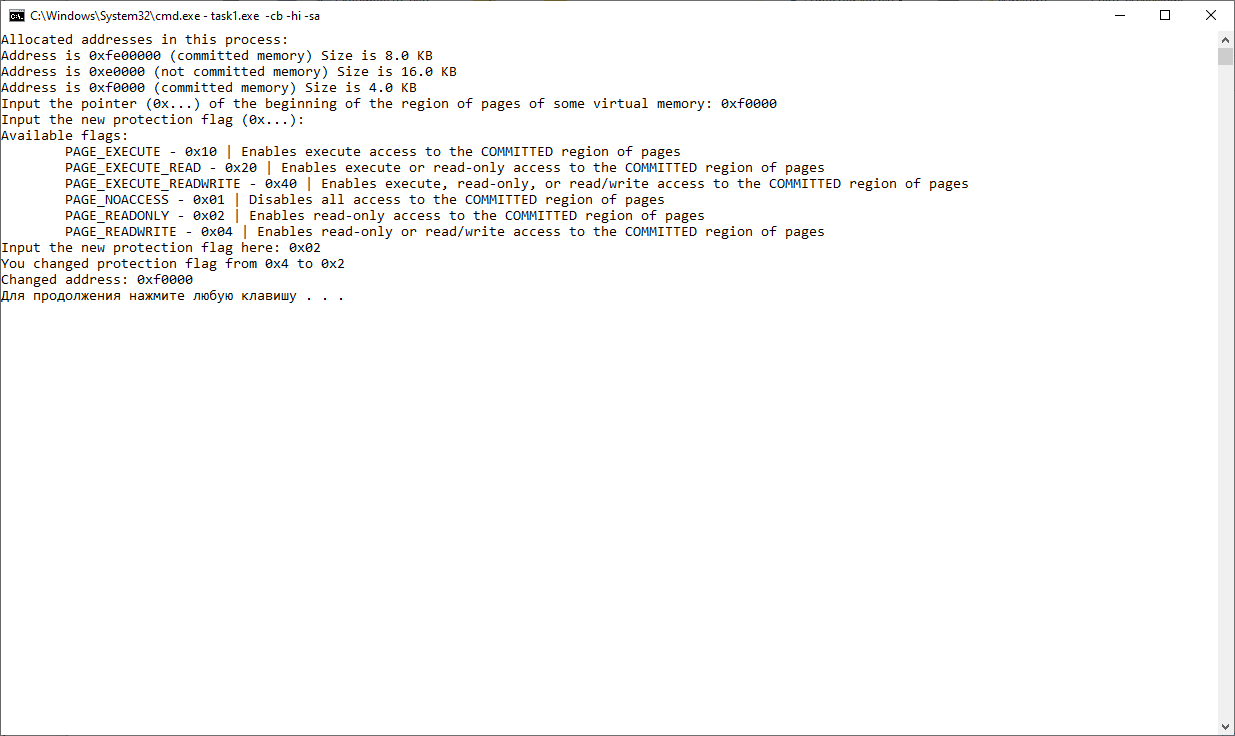


Рисунок . Установка защиты доступа

Освобождение регионов адресного пространства региона памяти производится через пункт 8 меню. Освобождение может происходить в нескольких режимах:

1. Освободить абсолютно все регионы, которые были зарезервированы. Если им передана физическая память, то происходит возврат памяти
2. Освободить конкретный участок, которому передана физическая память:
   1. Возвратить физическую память и оставить участок зарезервированным
   2. Возвратить физическую память и снять резервирование
3. Снять резервирование с конкретного участка регионов, которому не передана физическая память.

Все режимы представлены на рисунках 11, 12, 13 и 14 соответственно.

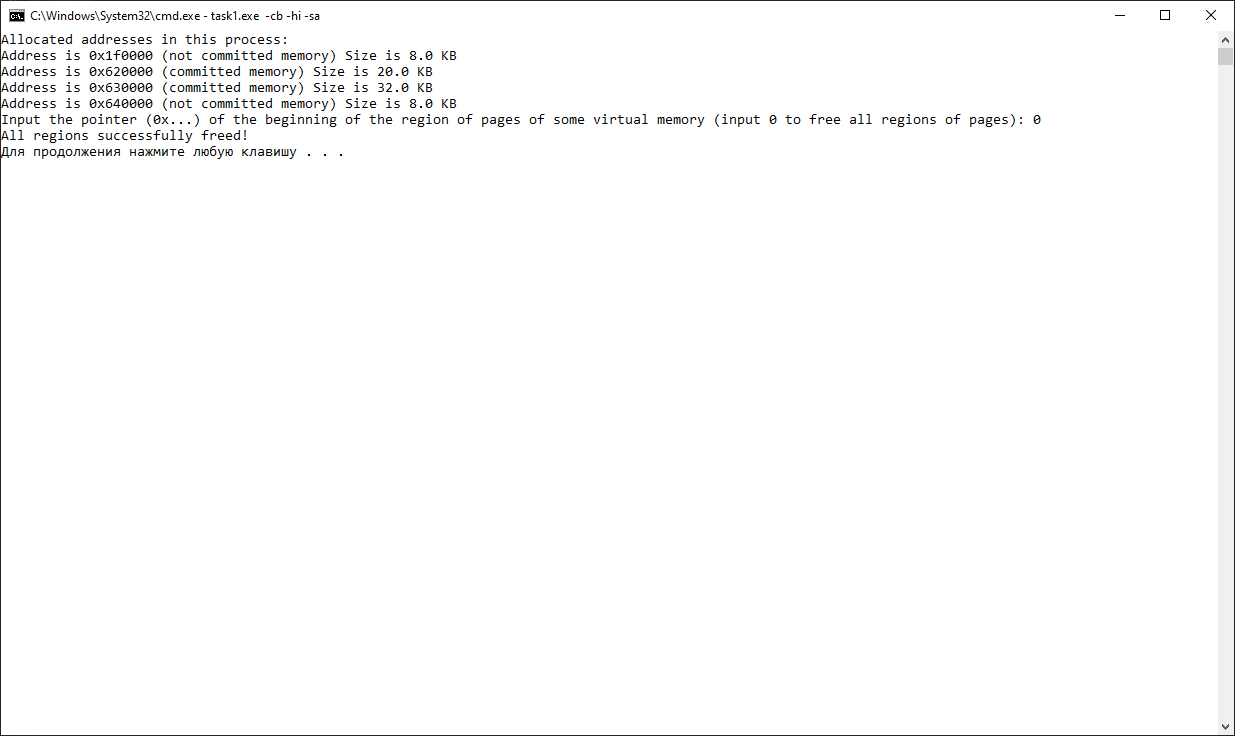


Рисунок . Освобождение всех регионов

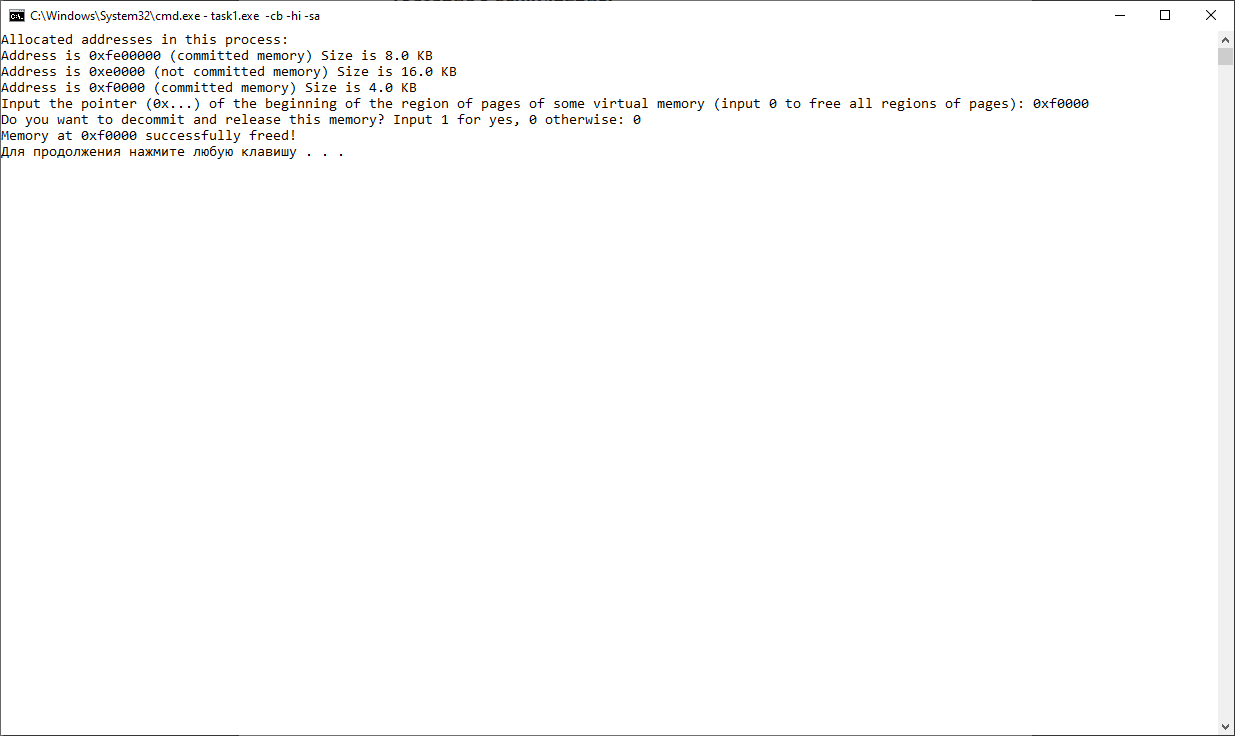


Рисунок . Возврат физической памяти без снятия резервирования

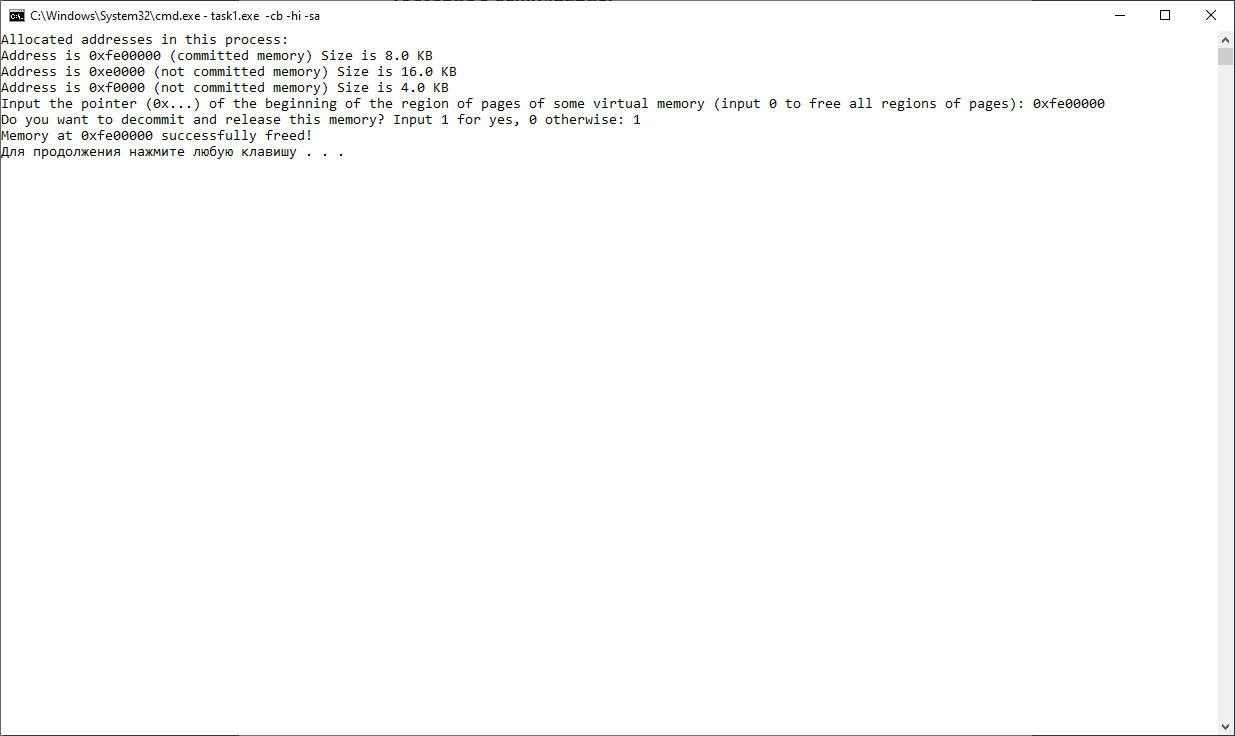


Рисунок . Возврат физической памяти со снятием резервирования

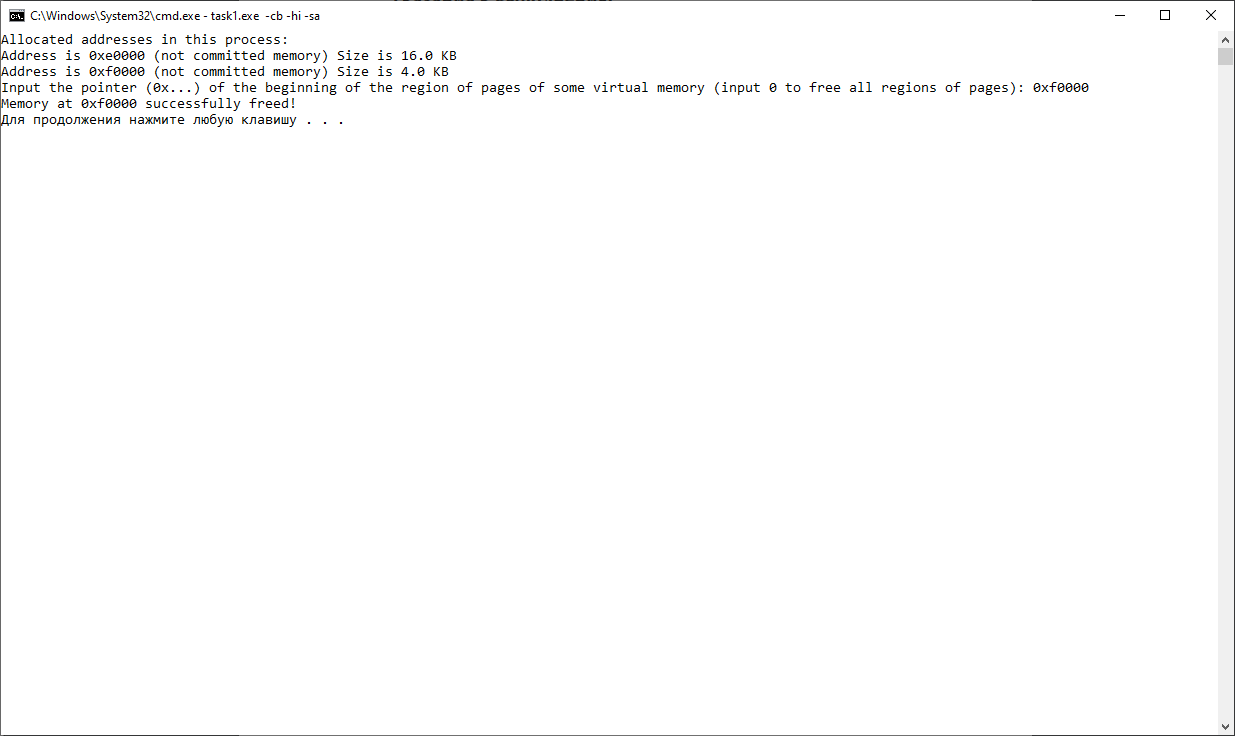


Рисунок . Снятие резервирования с указанного региона

Для того, чтобы узнать, какие участки были зарезервированы и была бы передана им физическая память, используется пункт 9 меню. Пример вывода представлен на рисунке 15.

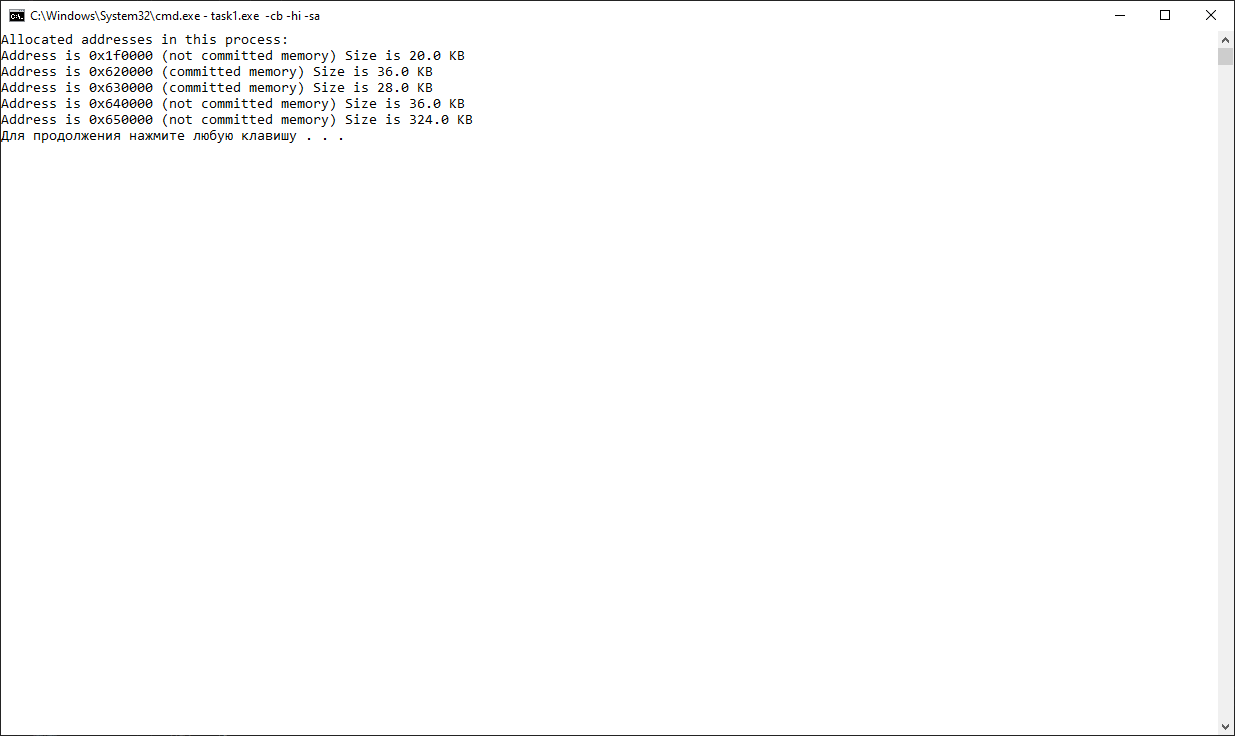


Рисунок . Вывод всех регионов, зарезервированных в процессе работы программы

**Вывод**

Были изучены различные функции Win32 API, позволяющие работать с виртуальным адресным пространством. Рассмотренные функции позволяют получить информацию об участках памяти по указанным адресам; зарезервировать необходимый участок и передать ему физическую память, если это нужно; устанавливать защиту доступа к определенному региону памяти; записывать данные в ячейки памяти по указанному адресу.

# Использование проецируемых файлов для обмена данными между процессами

## Указания к выполнению

Создайте два консольных приложения с меню (каждая выполняемая функция и/или операция должна быть доступна по отдельному пункту меню), которые выполняют:

* приложение-писатель создает проецируемый файл (функции Win32 API – **CreateFile**, **CreateFileMapping**), проецирует фрагмент файла в память (функции Win32 API – **MapViewOfFile**, **UnmapViewOfFile**), осуществляет ввод данных с клавиатуры и их запись в спроецированный файл;
* приложение-читатель открывает проецируемый файл (функция Win32 API – **OpenFileMapping**), проецирует фрагмент файла в память (функции Win32 API – **MapViewOfFile**, **UnmapViewOfFile**), считывает содержимое из спроецированного файла и отображает на экран.

## Примеры выполнения программы

Программа-писатель работает по следующему принципу:

1. Вводится имя файла для проецирования.
2. Вводится имя, которое используется для проецирования (оно будет использовано программой-читателем).
3. Вводятся сами данные.
4. Программа-писатель не закрывается, пока читатель не закончит свою работу.

Программа-читатель работает по следующему принципу:

1. Вводится имя файла, которое используется для проецирования (оно было задано программой-писателем).
2. Выводятся сами данные.

Пример работы написания представлен на рисунке 16. Пример работы чтения – на рисунке 17.

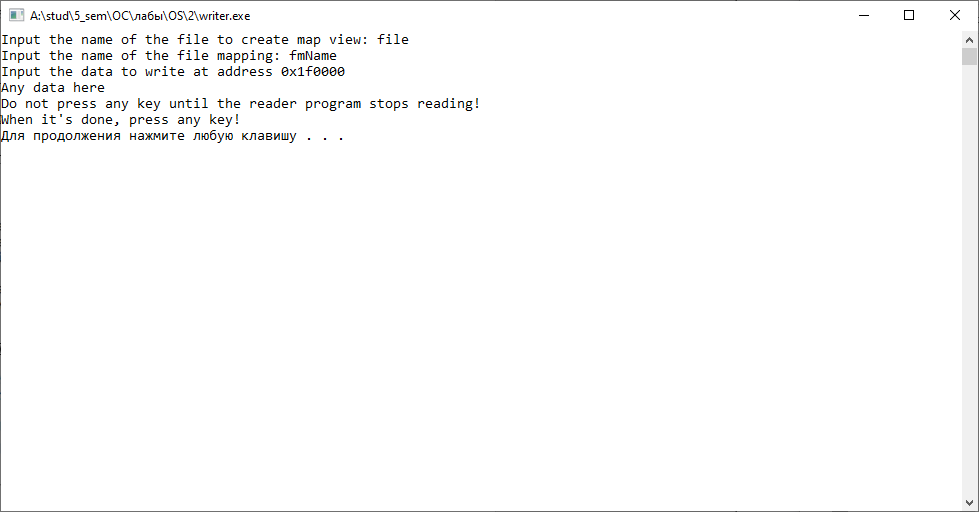


Рисунок . Пример работы программы-писателя

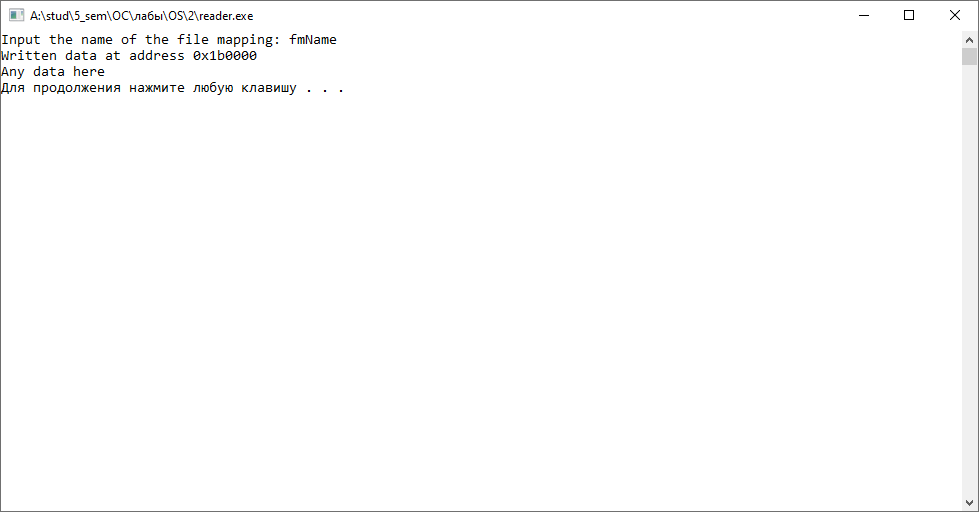


Рисунок . Пример работы программы-читателя

## Вывод

Был изучен механизм обмена данными между двумя процессами посредством проецирования файла в память. Наблюдаемое различие при проецировании файла в программе-писателе и программе-читателе в адресах, по которым производится запись/чтение, основывается на работе отображения файла в адресном пространство: используется непосредственно адресное пространство вызывающего отображение процесса.