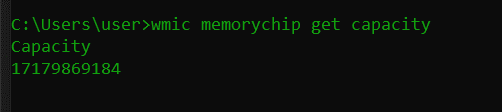
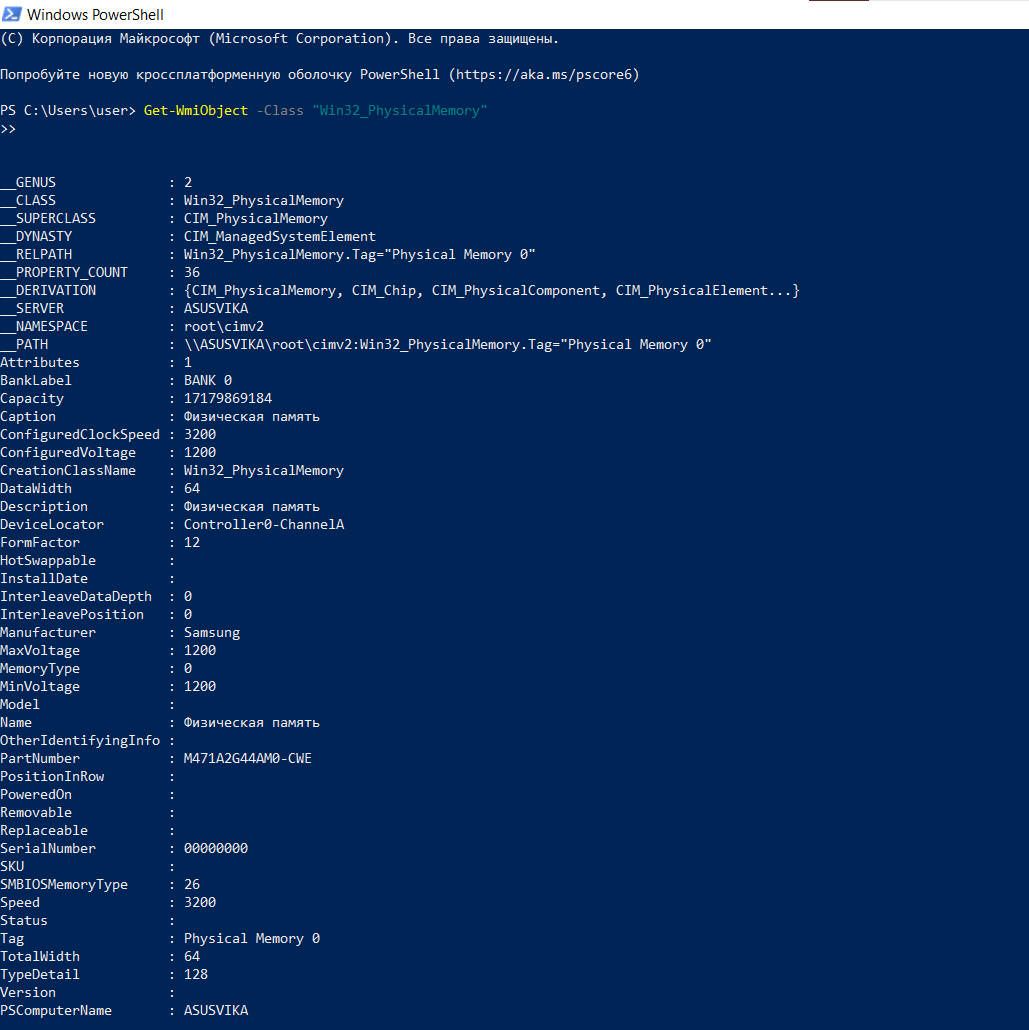
**Задание 01. Windows**

1. Получите с помощью утилиты **wmic** информации об физической оперативной памяти компьютера, поясните эту информацию.

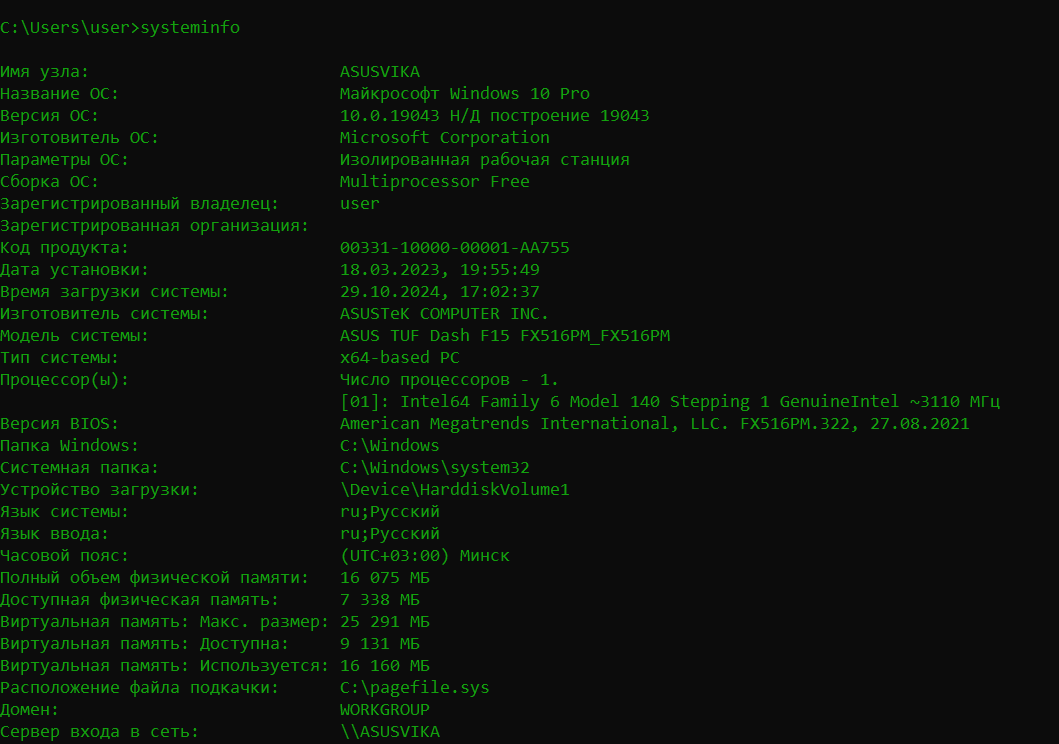


1. Получите с помощью утилиты **powershell** информации об физической оперативной памяти компьютера, поясните эту информацию.

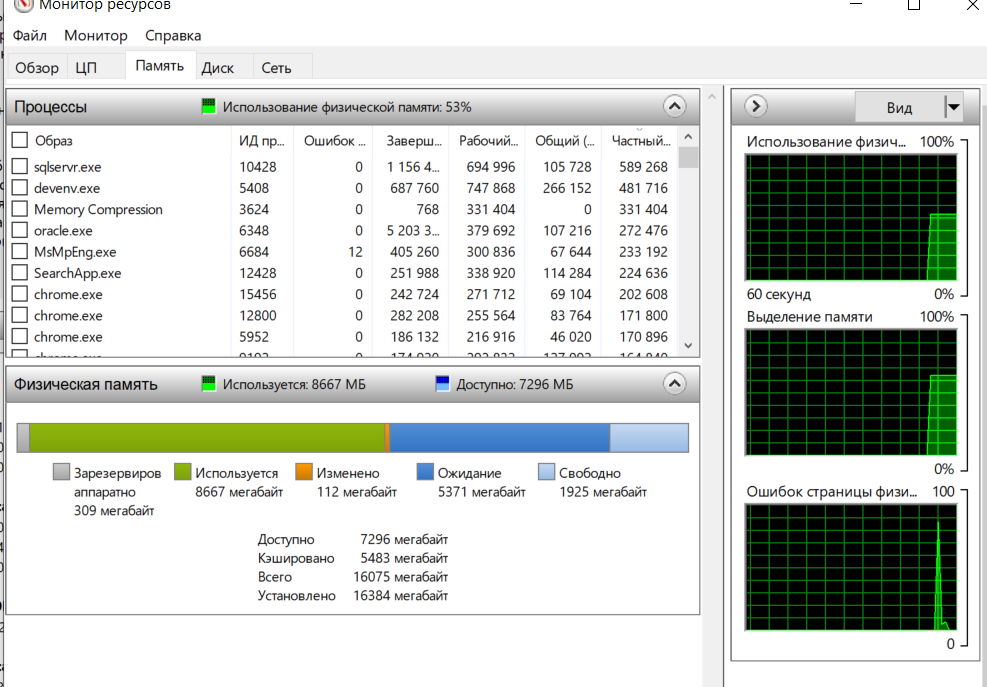
Get-WmiObject -Class “Win32\_PhysicalMemory”



1. Получите с помощью утилиты **systeminfo** информации об оперативной памяти компьютера, поясните эту информацию.

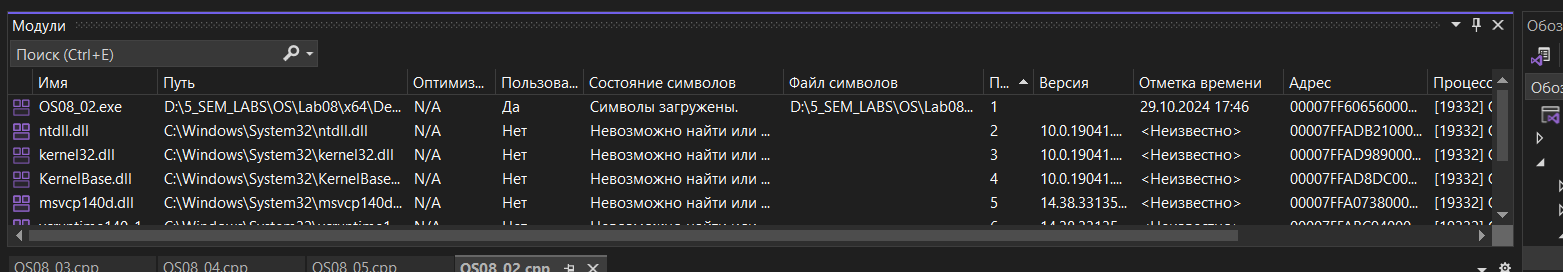


1. Получите с помощью утилиты **performance monitor** информации об оперативной памяти компьютера, поясните эту информацию.

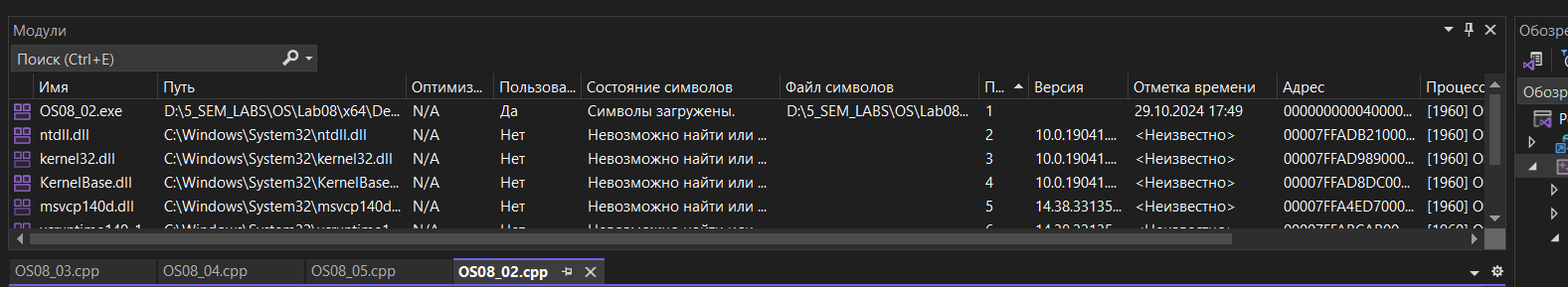
****

**Задание 02.Windows**

1. Разработайте консольное приложение **OS08\_02**, выполняющее длинный цикл.
2. Продемонстрируйте с помощью отладчика адреса расположения модулей приложения **OS08\_02**.



1. Установите для приложения **OS08\_02** стандартный адрес загрузки в память.
2. Продемонстрируйте с помощью отладчика стандартный адрес расположения модулей приложения **OS08\_02**



**Задание 03.Windows**

1. Разработайте консольное приложение **OS08\_03**, выполняющее получение 256 страниц оперативной памяти.
2. Разместите в этой памяти массив типа **int,** полностью занимающее все 256 страниц.
3. Заполните этот массив нарастающей последовательностью чисел с шагом 1.
4. Запишите 3 первых буквы своей фамилии в 16-ричными числами в кодировке Windows-1251.
5. Найдите в полученной области памяти с помощью отладчика значение в байте, имеющем адрес вычисленный по следующему принципу: номер страницы = число в нулевом байте, смещение в странице = число 12 бит в 1ом и втором байтах.

*Пример: Иванов*

*И = C8*

*в = E2*

*а = E0*

*Страница C8 = 200, смещение E2E = 3630*

*Б - 194(10) - C2(16)*

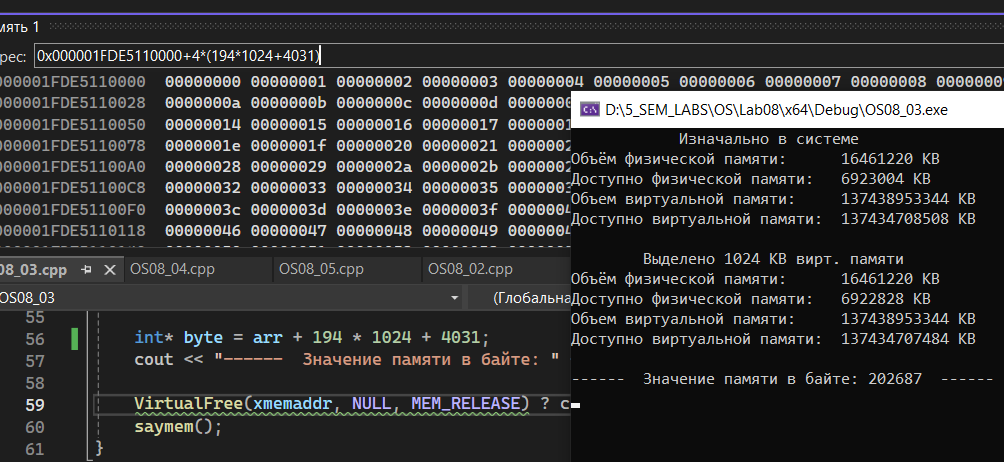
*ы - 251(10) - FB(16)*

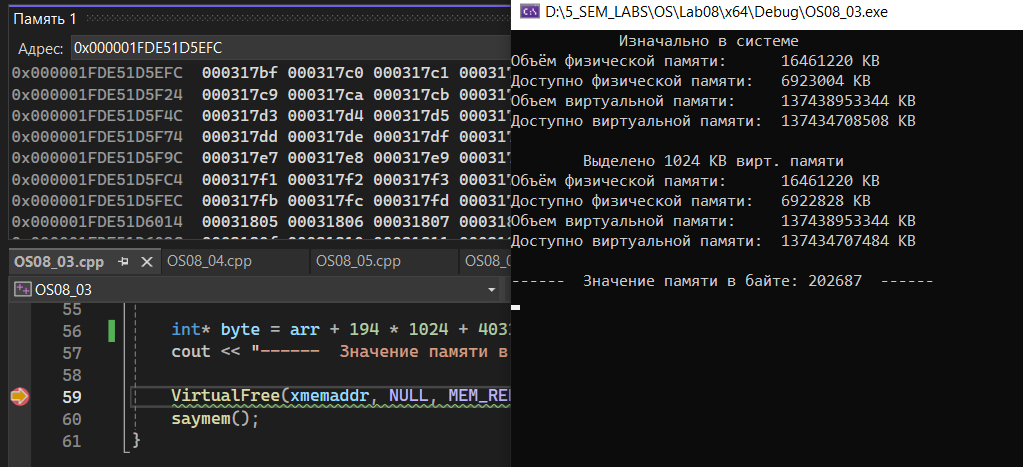
*ч - 247(10) - F7(16)*

*Страница с2 = 194*

*FBF = (15 \* 16^2) + (11 \* 16^1) + (15 \* 16^0) = 4031*

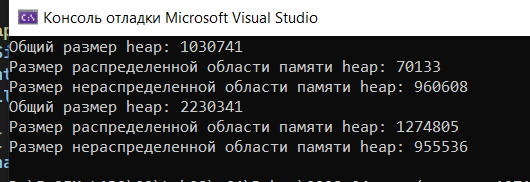
*Смещение FBF = 4031(10) = 0xFBF*





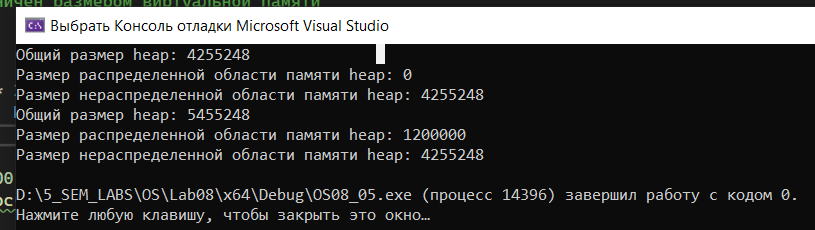
**Задание 04.Windows**

1. Разработайте консольное приложение **OS08\_04,** которое включает функцию **sh**, принимающую 1 параметр: дескриптор (HANDLE) heap.
2. Функция **sh** выводит на консоль, общий размер heap, размеры распределенной и нераспределенных областей памяти heap.
3. Приложение **OS08\_04** размещает в стандартной heap процесса int-массив размерности 300000.
4. Выведите с помощью функции **sh** информацию до размещения массива и после.
5. Объясните результат.



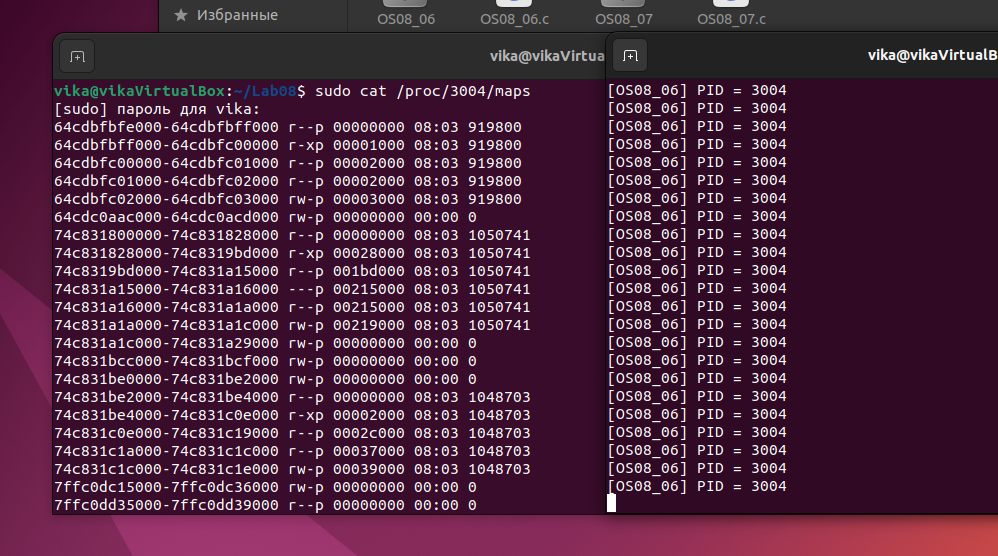
**Задание 05.Windows**

1. Разработайте консольное приложение **OS08\_05** аналогичное приложению **OS06\_05,** но использующее пользовательскую heap, которая имеет первоначальный размер 4MB.
2. Объясните результат.

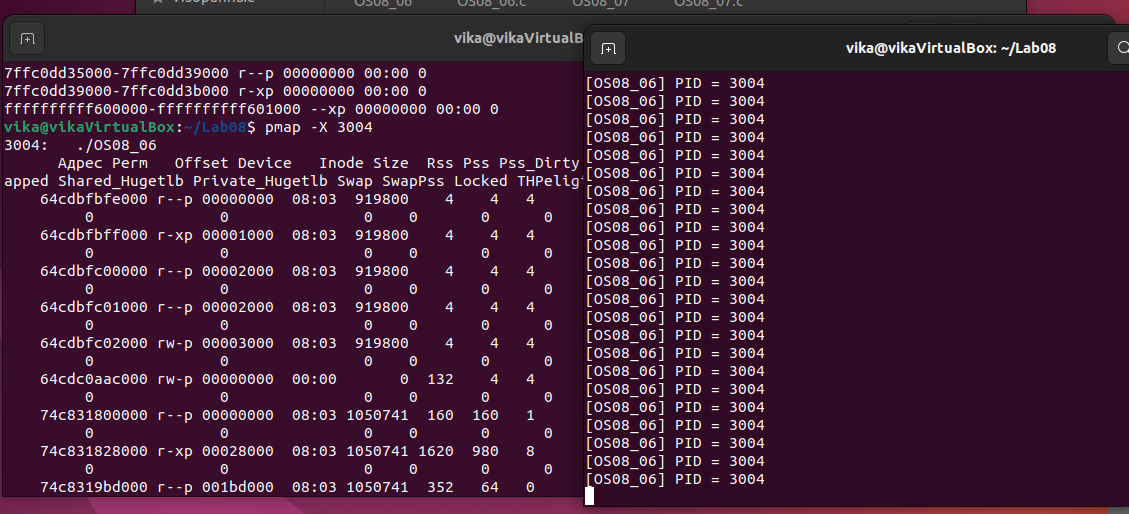


**Задание 06.Linux**

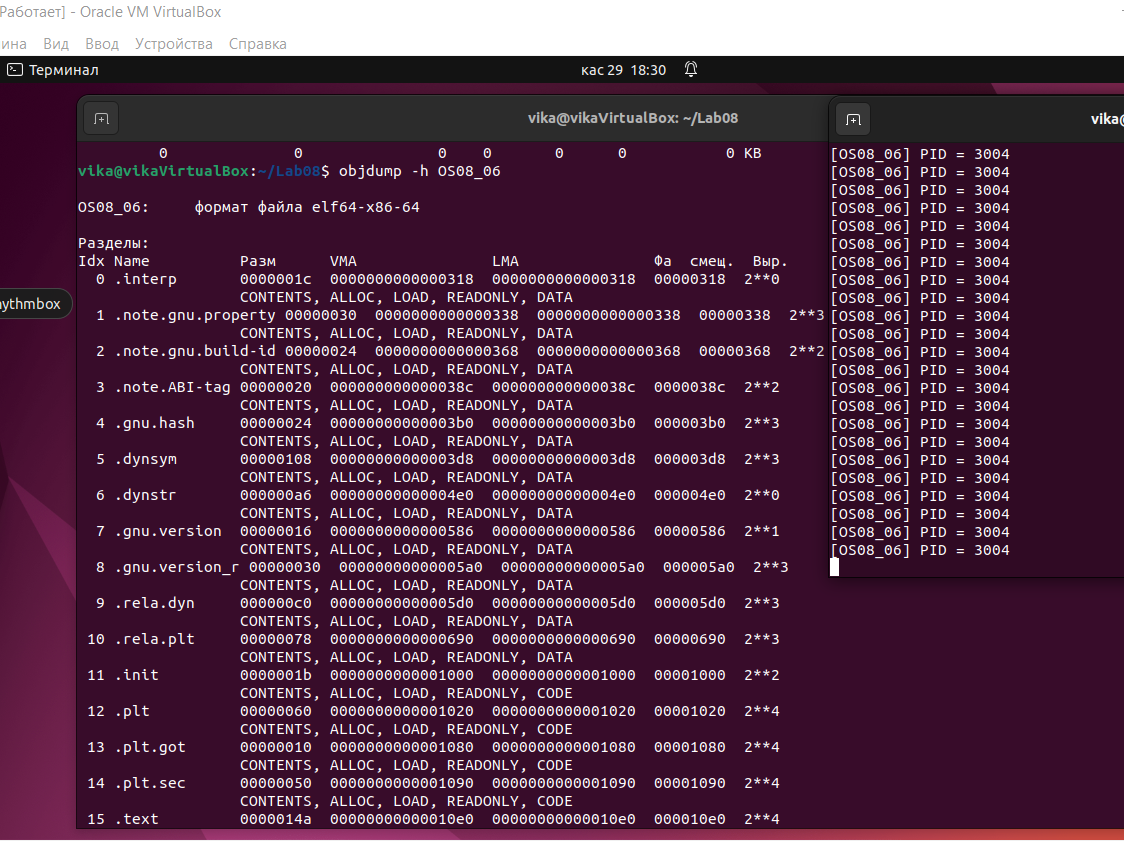
1. Разработайте консольное приложение **OS08\_06**, выполняющее длинный цикл.
2. Продемонстрируйте с помощью файловой системы **/proc** структуру адресного пространства.

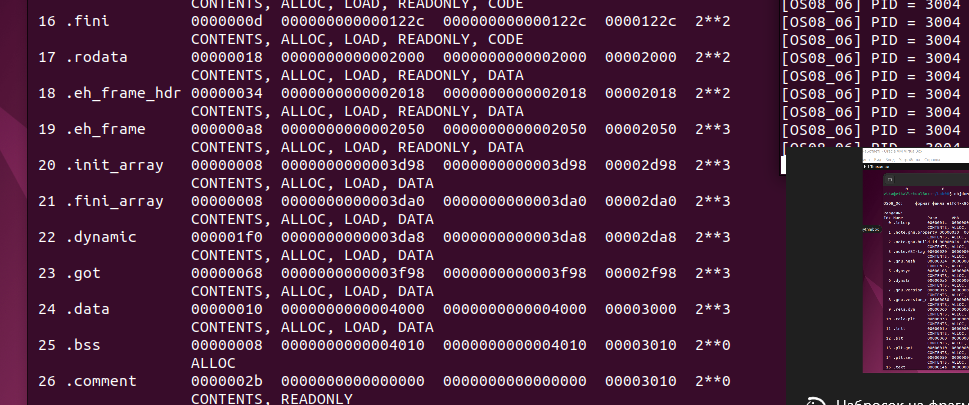


1. Продемонстрируйте с помощью **pmap**  структуру адресного пространства.



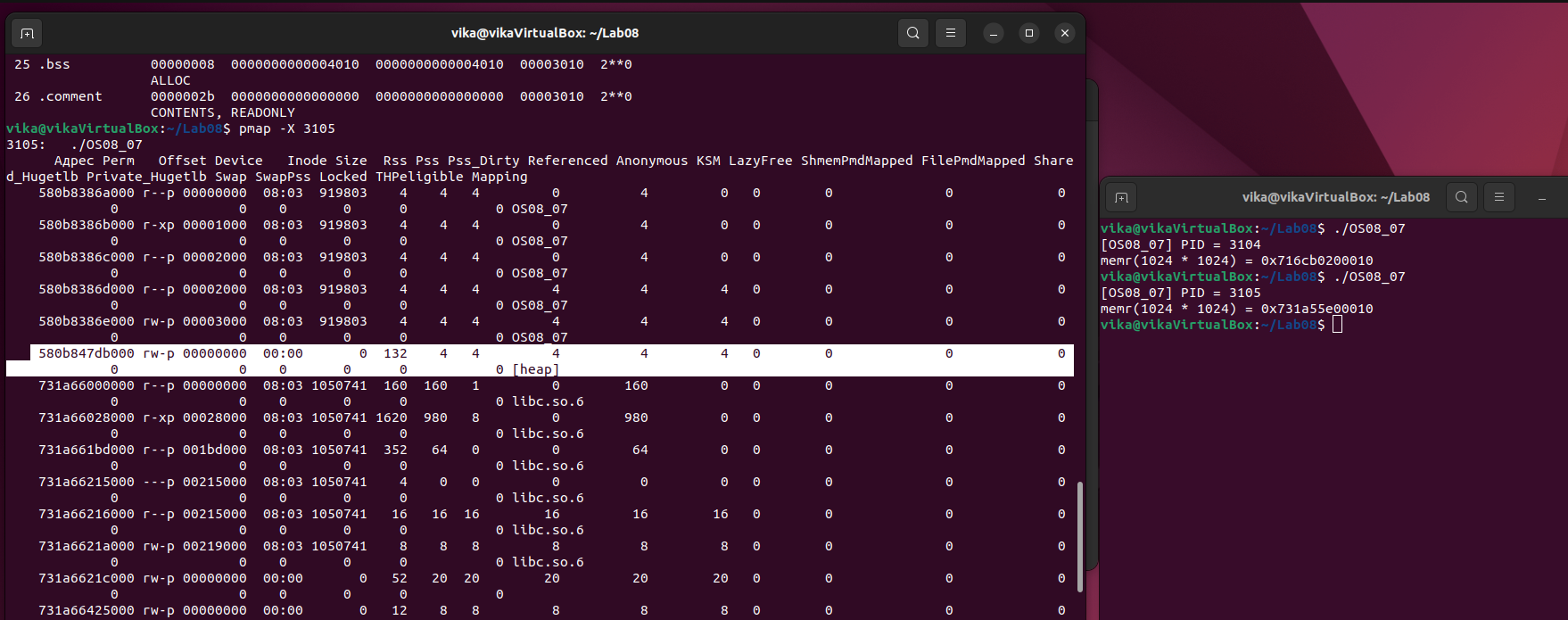
1. Определите с помощью утилиты objdump адрес загрузки main-модуля, секций с кодом, данными, неинициализированными глобальными переменными.





**Задание 07.Linux**

1. Разработайте консольное приложение **OS08\_07**, которое динамически выделяет 256 МБ памяти.
2. В выделенной памяти разместите int-массив максимальной размерности. Проинициализируйте массив последовательными значениями с шагом 1.
3. Выведите на консоль адрес выделенной памяти.
4. После инициализации приложение должно приостановить свое выполнение на длительный интервал времени.
5. С помощью утилиты pmap определите область памяти в которой выделена память.



1. Поясните понятие «виртуальная память».

Виртуальная память - это концепция, позволяющая программам исполняться, как если бы у каждой из них было свое собственное пространство оперативной памяти. Виртуальная память используется для создания иллюзии большего объема оперативной памяти, чем физически доступно на компьютере. Когда программы выполняются, они работают с виртуальной памятью, а операционная система отслеживает и управляет фактическим распределением данных между виртуальной и физической памятью.

1. Поясните понятие «свопинг».

Свопинг (англ. swapping) - это процесс передачи данных, хранящихся в оперативной памяти, на диск и обратно. Когда операционная система замечает, что физическая память заполняется, она может переместить некоторые неиспользуемые данные на диск в файл подкачки, освобождая физическую память для новых данных. При необходимости операционная система может снова загрузить данные из файла подкачки в оперативную память.

1. Поясните понятие «страничная память».

Страничная память - это механизм управления памятью, используемый операционными системами. В рамках концепции страничной памяти виртуальное адресное пространство программы разбивается на небольшие блоки, называемые страницами. Физическая память также разбивается на блоки одинакового размера, называемые фреймами. Страничная память позволяет отображать виртуальные страницы на физические фреймы, позволяя программам эффективно использовать физическую память.

1. Поясните понятие MMU.

MMU (Memory Management Unit):

MMU - это устройство, обычно встроенное в процессор, которое отвечает за преобразование виртуальных адресов, используемых программами, в соответствующие физические адреса в оперативной памяти. Он играет ключевую роль в реализации виртуальной памяти.

1. Поясните понятие TLB.

TLB (Translation Lookaside Buffer) - это кэш ассоциативной памяти, используемый MMU для ускорения преобразования виртуальных адресов в физические адреса. TLB хранит последние использованные соответствия виртуальных адресов и физических адресов, что позволяет MMU избежать частых обращений к таблице страниц и ускоряет доступ к памяти.

1. Какая информация содержится в строке таблицы страниц

Строка таблицы страниц содержит информацию о соответствии виртуальных страниц программы физическим фреймам памяти. Она может включать в себя информацию, такую как номер виртуальной страницы, номер физического фрейма, флаги доступа и прав доступа (например, чтение, запись, выполнение).

1. Поясните принцип применения хэш-таблиц.

Хэш-таблицы используются для эффективного хранения и поиска данных. Они основаны на хэш-функциях, которые преобразуют ключи данных в хэш-коды. Хэш-таблица состоит из массива ячеек, где каждая ячейка содержит список элементов с одинаковыми хэш-кодами. Это позволяет быстро находить элементы по ключу, обычно за константное время.

1. Поясните применение «инвертированной таблицы физических» страничной памяти.

Инвертированная таблица физических страниц - это особый способ организации страничной памяти, где вместо того, чтобы хранить соответствия виртуальных страниц программы и физических фреймов в каждом процессе, используется общая таблица, известная как инвертированная таблица. Каждая запись в инвертированной таблице содержит информацию о соответствии виртуальной страницы идентификатору процесса и физическому адресу. Это позволяет экономить память, но требует дополнительных операций для поиска и обновления записей в таблице.

1. Поясните понятие «рабочий набор страниц».

Рабочий набор страниц - это набор страниц виртуальной памяти, которые активно используются программой в данный момент. Рабочий набор страниц обычно содержит страницы кода программы, активно используемые данные и стек вызовов. Понимание рабочего набора страниц позволяет операционной системе эффективно управлять страничной памятью, перемещая наиболее активные страницы в физическую память и перемещая менее активные страницы на диск.

1. Поясните принцип работы алгоритма LRU.

LRU (Least Recently Used) - это алгоритм замещения страниц в страничной памяти, который основывается на принципе, что страницы, которые дольше всего не использовались, наименее вероятно будут использоваться в ближайшем будущем. При необходимости замещения страницы операционная система выбирает страницу, которая была наименее недавно использована, и замещает ее новой страницей. Это позволяет максимизировать использование физической памяти для наиболее активных страниц.

1. Windows: поясните назначение сервиса SysMain.

Служба SysMain (ранее известная как SuperFetch) в Windows предназначена для улучшения производительности системы путем адаптивного предварительного загрузки и кэширования данных на основе использования программ. Она может предварительно загружать часто используемые приложения и данные в оперативную память, чтобы ускорить их запуск и обеспечить быстрый доступ к ним.

1. Windows: поясните назначение файла hiberfil.sys.

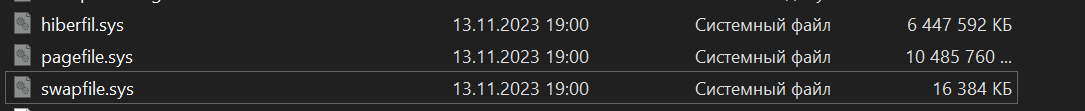
Файл hiberfil.sys - это системный файл в Windows, который используется для хранения снимка оперативной памяти при переходе компьютера в режим гибернации. В режиме гибернации компьютер сохраняет состояние операционной системы и отключается, а при следующем включении восстанавливает состояние из файла hiberfil.sys.

1. Windows: поясните назначение файла pagefile.sys.

Файл pagefile.sys - это файл подкачки в Windows, который используется для вспомогательного хранения данных, когда физическая память компьютера заполняется. Когда оперативная память исчерпывается, операционная система может перемещать неиспользуемые данные на диск в файл pagefile.sys, освобождая физическую память.

1. Windows: поясните назначение файла swapfile.sys.

Файл swapfile.sys - это файл подкачки, используемый в Windows 10 и более новых версиях вместо файла pagefile.sys. Swapfile.sys выполняет ту же функцию - предоставляет дополнительное пространство на диске для размещения неиспользуемых данных из физической памяти.



1. Windows: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.

В Windows области адресного пространства перечисляются от младших к старшим адресам следующие области:

а) Нижняя область адресного пространства (Lower Address Space) - эта область подразделяется на несколько сегментов:

- Сегмент кода (Code Segment) - в этом сегменте размещается исполняемый код программы.

- Сегмент данных (Data Segment) - в этом сегменте размещаются глобальные и статические данные программы.

- Сегмент стека (Stack Segment) - в этом сегменте размещается стек вызовов функций и локальные переменные.

б) Область адресного пространства, зарезервированная для системы (Reserved System Space) - эта область зарезервирована для нужд операционной системы и не доступна для пользовательских приложений.

в) Область адресного пространства, зарезервированная для пользовательских приложений (Reserved User Space) - это область адресного пространства, которая доступна для пользовательских приложений для выделения памяти и размещения данных.

1. Windows: какой стандартный начальный размер области heap?

Стандартный начальный размер области heap в Windows зависит от версии операционной системы и настроек системы. В общем случае, стандартный начальный размер heap составляет 1 мегабайт (1 MB). Но вообще, он динамически определяется.

1. Windows: каким образом можно изменить начальный размер области памяти heap приложения?

Начальный размер области памяти heap приложения можно изменить с помощью функции HeapCreate из Windows API. Эта функция позволяет создать heap с указанным начальным размером. При создании heap можно указать размер в байтах или в виде процентного соотношения от размера физической памяти компьютера. Также можно изменить размер heap с помощью функции HeapReAlloc, которая позволяет изменить размер уже существующего heap.

1. Windows: какой стандартный размер области памяти stack?

Стандартный размер области памяти stack в Windows зависит от версии операционной системы и настроек системы. В общем случае, стандартный размер stack составляет 1 мегабайт (1 MB).

1. Windows: каким образом можно изменить размер области памяти stack приложения?

Размер области памяти stack приложения можно изменить с помощью функции SetThreadStackGuarantee из Windows API. Эта функция позволяет установить минимальный гарантированный размер stack для новых потоков, создаваемых в приложении. При увеличении размера stack необходимо учесть доступное виртуальное адресное пространство и ограничения операционной системы.

1. Windows: поясните назначение функции Windows API: GlobalMemoryStatus.

Функция Windows API GlobalMemoryStatus предоставляет информацию о доступной физической и виртуальной памяти в системе. Она возвращает структуру MEMORYSTATUS, содержащую информацию о размере памяти, используемой и доступной в системе.

1. Windows: поясните назначение функции Windows API: VirtualQuery; перечислите значения атрибутов Protect, State и Type.

Функция Windows API VirtualQuery используется для получения информации о виртуальной памяти в процессе. Она возвращает информацию о регионе виртуальной памяти, включая его адрес, размер, атрибуты защиты (Protect), состояние (State) и тип (Type).

Значения атрибута Protect:

- PAGE\_NOACCESS: Регион памяти не доступен для чтения или записи.

- PAGE\_READONLY: Регион памяти доступен только для чтения.

- PAGE\_READWRITE: Регион памяти доступен для чтения и записи.

- PAGE\_WRITECOPY: Регион памяти доступен для записи, но изменения не применяются к другим процессам.

- и другие значения, определяющие различныеправа доступа к памяти.

Значения атрибута State:

- MEM\_COMMIT: Регион памяти выделен и использован.

- MEM\_FREE: Регион памяти свободен и не выделен.

- MEM\_RESERVE: Регион памяти зарезервирован для будущего использования.

Значения атрибута Type:

- MEM\_IMAGE: Регион памяти содержит образ исполняемого файла.

- MEM\_MAPPED: Регион памяти сопоставлен с файлом или устройством.

- MEM\_PRIVATE: Регион памяти предназначен только для использования процессом.

1. Windows: что такое «рабочее множество»? поясните принцип управления рабочим множеством с помощью OS API.

В контексте Windows термин "рабочее множество" (working set) относится к набору физических страниц памяти, используемых процессом в данный момент. Рабочее множество включает в себя активные страницы памяти, к которым процесс часто обращается, и может изменяться динамически во время выполнения программы. Управление рабочим множеством осуществляется с помощью OS API, таких как функции SetProcessWorkingSetSize и QueryWorkingSet из Windows API. С помощью SetProcessWorkingSetSize можно установить ограничения на размер рабочего множества процесса, а с помощью QueryWorkingSet можно получить информацию о текущем состоянии рабочего множества.

1. Windows: что означает «страница заблокирована»? с помощью каких функций OS API можно установить блокировку страниц и снять блокировку? Какое максимальное количество страниц можно заблокировать?

В контексте Windows термин "страница заблокирована" (locked page) означает, что физическая страница памяти зарезервирована и не может быть выгружена на диск в случае нехватки физической памяти. Заблокированные страницы памяти обычно используются для хранения критически важных данных, которые должны быть всегда доступны в памяти.

Для установки блокировки страниц памяти можно использовать функцию VirtualLock, а для снятия блокировки - функцию VirtualUnlock из Windows API. Максимальное количество страниц, которые можно заблокировать, зависит от операционной системы и доступной физической памяти.

1. Windows: что такое «heap»? Что такое «heap процесса»? Что такое «пользовательская heap»? Поясните принцип устройства heap.

В Windows термин "heap" относится к области памяти, используемой для динамического выделения памяти в процессе. Heap представляет собой блоки памяти различного размера, которые могут быть выделены и освобождены во время выполнения программы.

Heap процесса (process heap) - это основной heap, создаваемый для каждого процесса при его запуске. Он предоставляет механизм для выделения и освобождения памяти внутри процесса.

Пользовательская heap (user heap) - это heap, создаваемый пользовательским приложением для выделения и управления памятью. Пользовательские heap'ы являются частью process heap'а и могут быть созданы с помощью функций HeapCreate или GetProcessHeap из Windows API.

Устройство heap основано на списке блоков памяти различных размеров, известном как список свободных блоков (free list). При запросе памяти heap ищет подходящий блок в списке свободных блоков и выделяет его для приложения. При освобождении памяти блок возвращается в список свободных блоков для последующего использования. Heap также выполняет операции сборки мусора и дефрагментации данных.

1. Linux: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.

В Linux области адресного пространства перечисляются от младших к старшим адресам следующие области:

а) Нижняя область адресного пространства:

- Область кода (Code Segment) - в этой области размещается исполняемый код программы.

- Область данных (Data Segment) - в этой области размещаются глобальные и статические данные программы.

- Область BSS (Block Started by Symbol) - в этой области размещаются неинициализированные глобальные и статические переменные.

- Область стека (Stack Segment) - в этой области размещается стек вызовов функций и локальные переменные.

б) Область адресного пространства, зарезервированная для библиотек и разделяемых объектов (Shared Libraries and Shared Objects) - в этой области размещаются библиотеки, используемые программой, и разделяемые объекты, которые могут быть загружены и используемы несколькими процессами одновременно.

в) Область адресного пространства, зарезервированная для отображения файлов в память (Memory-Mapped Files) - в этой области размещаются файлы, которые могут быть отображены в память и использованы как обычная память. Это позволяет обращаться к содержимому файла напрямую через указатели и использовать его без необходимости чтения и записи на диск.

г) Область адресного пространства, зарезервированная для стека сигналов (Signal Stack) - в этой области размещается стек, используемый для обработки сигналов, которые могут быть отправлены процессу.

д) Область адресного пространства, зарезервированная для динамической загрузки библиотек (Dynamic Linking) - в этой области размещаются библиотеки, которые загружаются динамически во время выполнения программы.

е) Область адресного пространства, зарезервированная для кучи (Heap Segment) - в этой области происходит выделение и управление динамической памятью с помощью функций, таких как malloc и calloc.

1. Linux: в какой части адресного пространства выделяется памяти с помощью функций malloc, calloc?

В Linux память, выделяемая с помощью функций malloc и calloc, выделяется в области адресного пространства, называемой "кучей" (Heap Segment). Куча представляет собой область памяти, используемую для динамического выделения и освобождения памяти во время выполнения программы.

Функции malloc и calloc взаимодействуют с менеджером памяти (memory manager), который управляет выделением и освобождением блоков памяти в куче. Когда вызывается функция malloc или calloc, менеджер памяти ищет свободный блок памяти в куче достаточного размера для удовлетворения запроса и возвращает указатель на этот блок. При освобождении памяти с помощью функции free, блок памяти возвращается обратно в кучу и может быть повторно использован.

Область адресного пространства, выделенная для кучи, находится в верхней части адресного пространства процесса и растет в направлении старших адресов при выделении памяти.