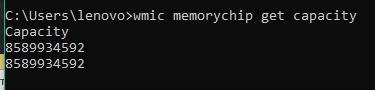
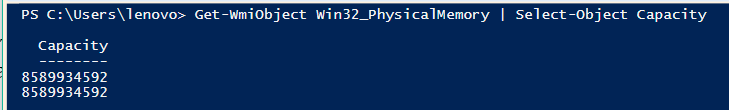
Задание 01. Windows

1. Получите с помощью утилиты wmic информации об физической

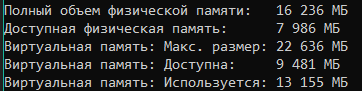
оперативной памяти компьютера, поясните эту информацию.



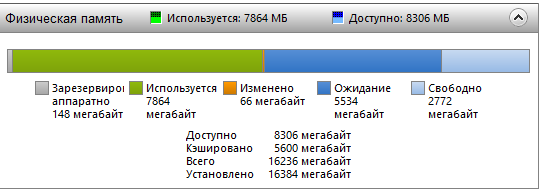
2. Получите с помощью утилиты powershell информации об физической оперативной памяти компьютера, поясните эту информацию.



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

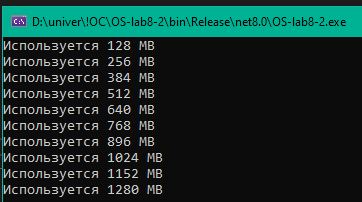


4. Получите с помощью утилиты performance monitor информации об оперативной памяти компьютера, поясните эту информацию.

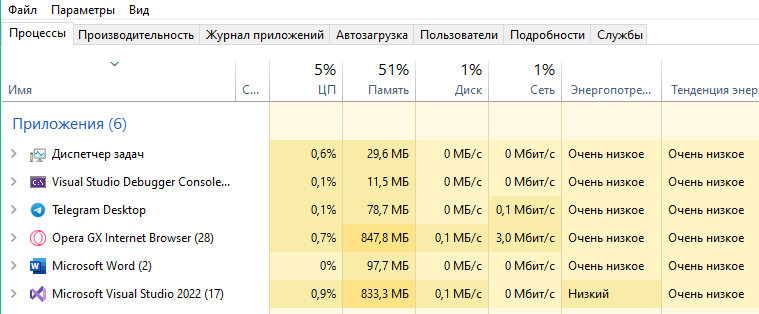


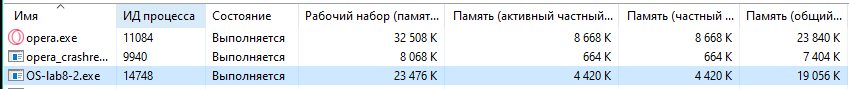
**Задание 02.Windows**

5. Разработайте на языке программирования C# консольное приложение OS08\_02, которое каждые 5 секунд создает новый объект размером 128 МБ и выводит объем используемой памяти, установите Platform Target x86.

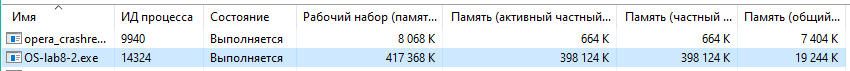
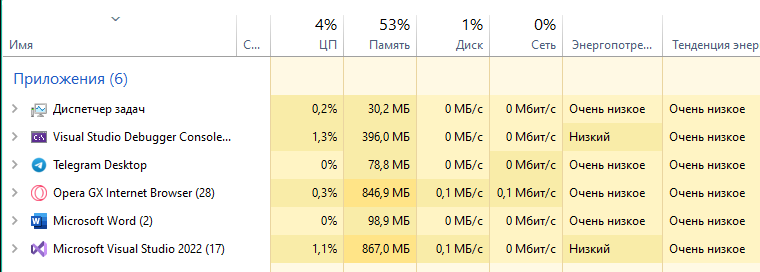


6. Понаблюдайте за выделенной памятью рабочим набором памяти приложения OS08\_02 в Диспетчере задач. Будет ли приложение работать бесконечно?

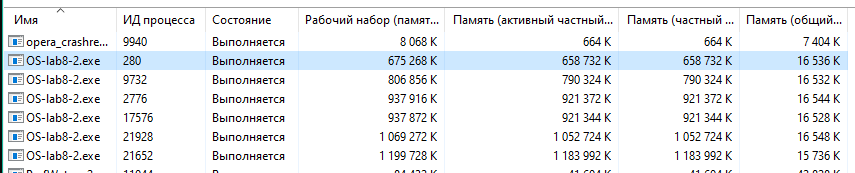


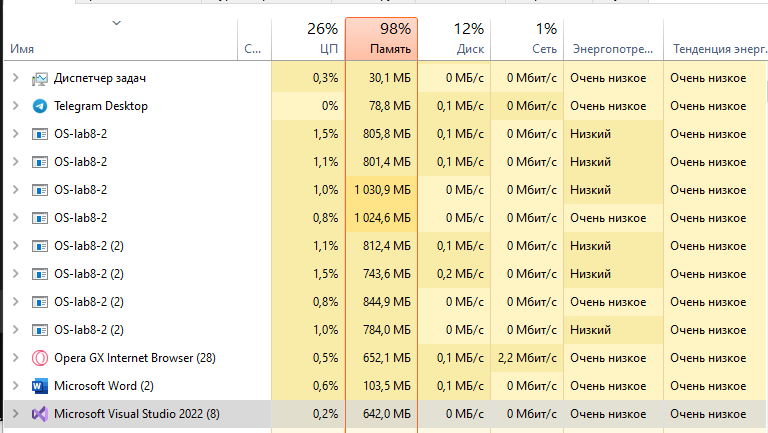


7. Добавьте в приложение OS08\_02 код, который заполняет произвольными значениями выделенную память, лучше всего в отдельном потоке. **Изменилось ли соотношение выделенной памяти и рабочего набора?** *Заполнение памяти данными может привести к более активному использованию выделенной памяти.* *Рабочий набор увеличится, так как страницы будут загружены в физическую память.*



8. Запустите несколько экземпляров приложения OS08\_02. Сравните частный и общий рабочие наборы.





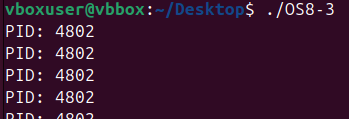
*Частный – не может использоваться другими процессами. Общий – может.*

*Частный рабочий набор включает в себя те страницы памяти, которые используются конкретным экземпляром приложения. Общий рабочий набор включает в себя страницы, которые используются всеми экземплярами приложения.*

*Общий рабочий набор будет меньше, если процессы используют общие данные. Если же каждый процесс использует свою память, то разница будет незначительной.*

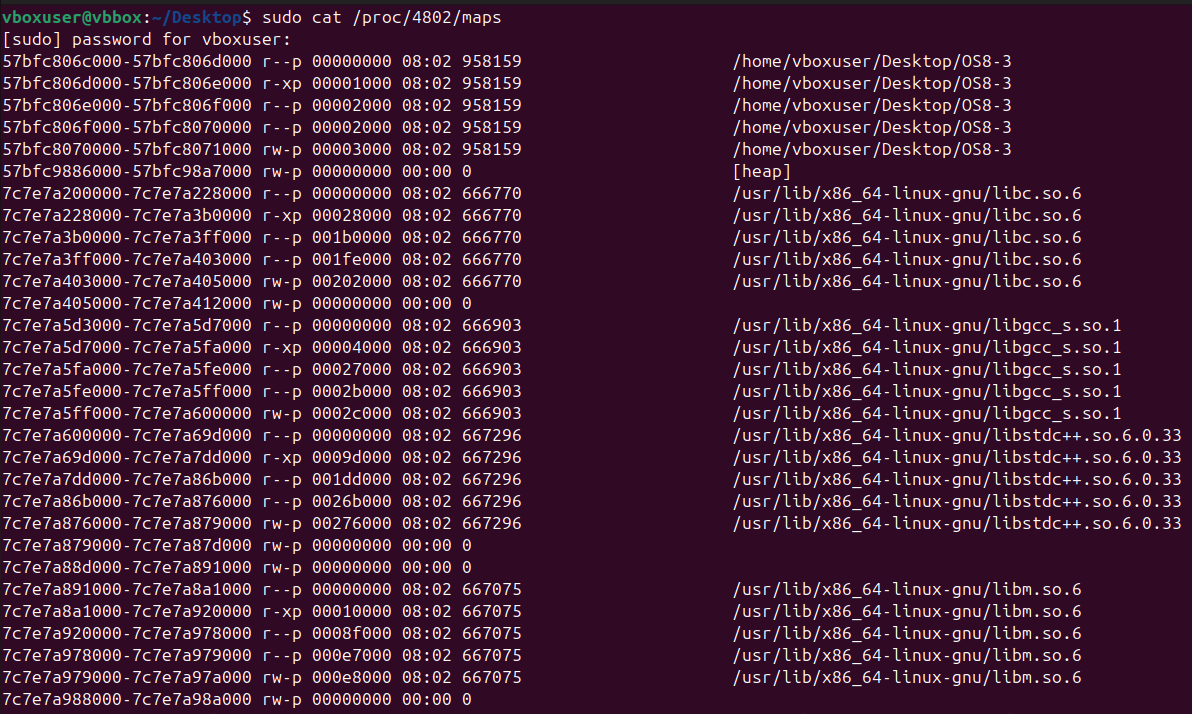
**Задание 03. Linux**

9. Разработайте консольное приложение OS08\_03, выполняющее длинный цикл.

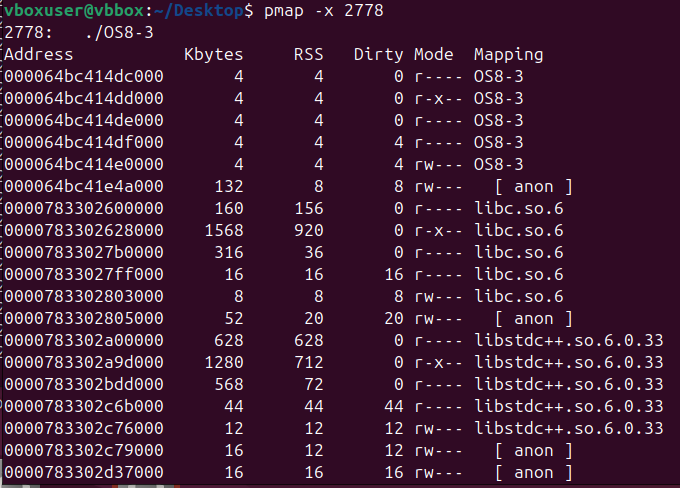


10. Продемонстрируйте с помощью файловой системы /proc структуру адресного пространства.

Показывает все области памяти, выделенные для процесса.



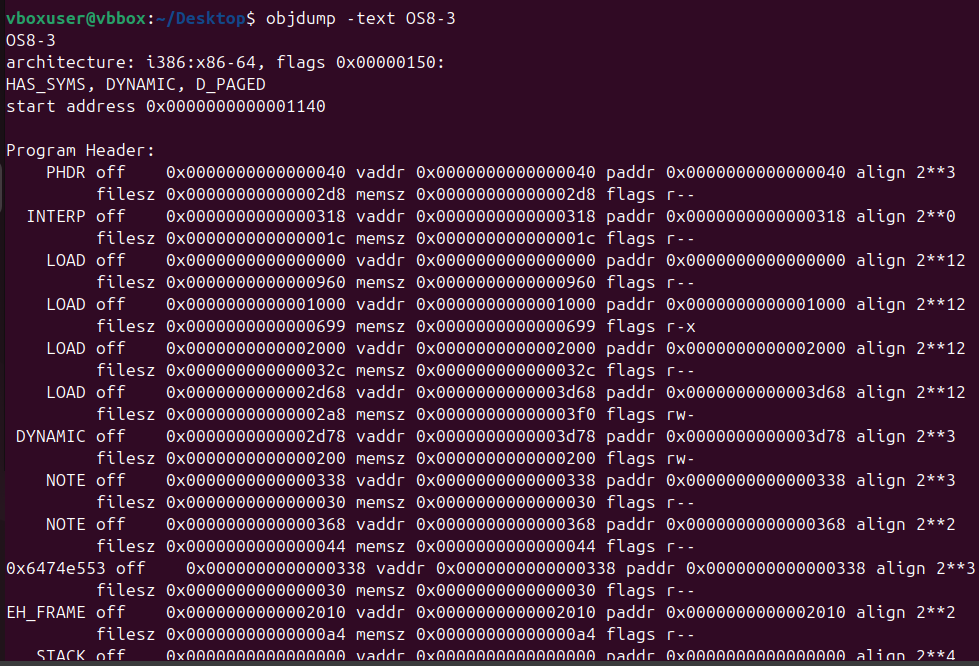
11. Продемонстрируйте с помощью pmap структуру адресного пространства.

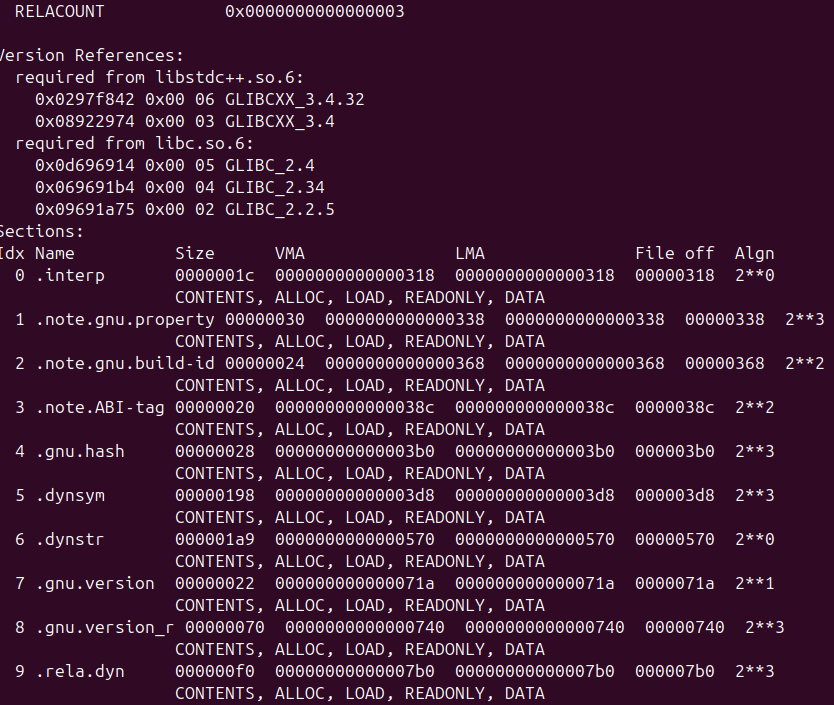


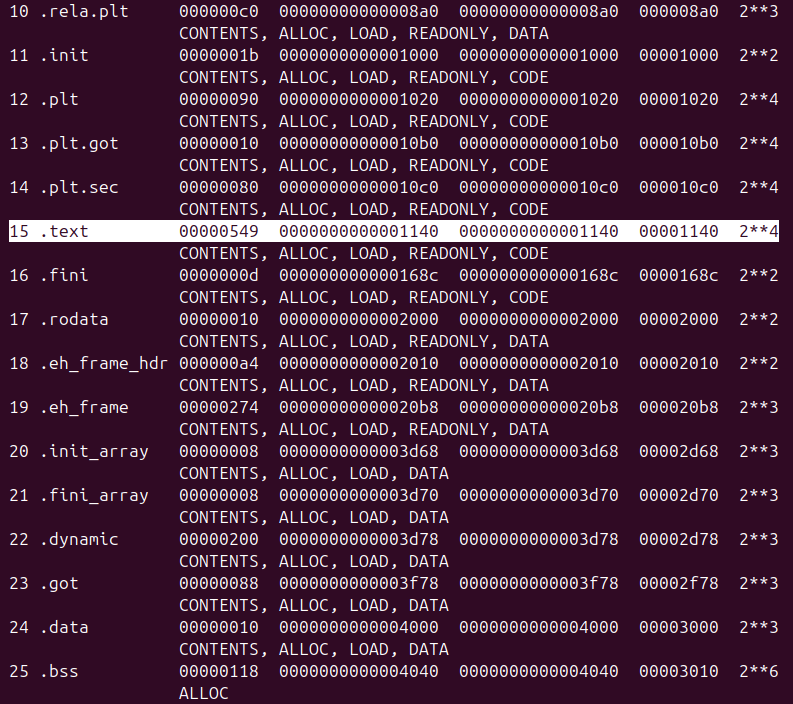
12. Определите с помощью утилиты objdump адрес загрузки main модуля, секций с кодом, данными, неинициализированными глобальными переменными.

 **VMA**: Адрес загрузки секции (Virtual Memory Address).

 **LMA**: Адрес загрузки в памяти (Load Memory Address).







Вариант 1.

Укажите содержимое байта по виртуальному адресу 0x01579333, если регистр CR3 содержит значение 0x13907000.

**0x01579333**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 5 | 7 | 9 | 3 | 3 | 3 |
| 0000 | 0001 | 0101 | 0111 | 1001 | 0011 | 0011 | 0011 |

0000 0001 0101 0111 1001 0011 0011 0011

**0x13907000**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 9 | 0 | 7 | 0 | 0 | 0 |
| 0001 | 0011 | 1001 | 0000 | 0111 | 0000 | 0000 | 0000 |

0001 0011 1001 0000 0111 0000 0000 0000

**Смотрим по битам:**

Старшие 10 бит – индекс каталога страниц- 0000 0000 0101 – 0х005

Средние 10 бит – индекс таблицы страниц - 0001 0111 1001 – 0х179

Младшие 12 бит – смещение по странице - 0011 0011 0011 – 0х333

Старшие 10: 0000 0001 01 – PDI – page directory index – каталог страницы.

0000 0000 0101 – 0x005

Средние 10: 0111 1001 00 – PTI – page table index – таблица страниц.

0001 0111 1001 – 0x179

Младшие 12: 0011 0011 0011 – смещение.

0011 0011 0011 – 0x333

**Базовый адрес страницы** – по смещению с индексом каталога страниц.

**Физический адрес записи** в каталоге страниц = адрес корневой таблицы страниц + PDI \* 4

0x13907000 + 0х005\*4(2010) = 0x13907 –> 000 + 0x14 = 0x13907014

|  |
| --- |
| 13907000: 13909127 1390D127 1390B127 1390F127 1390A127 13908127 1390E127 1390C127 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 13909127 | 1390D127 | 1390B127 | 1390F127 | 1390A127 | 13908127 | 1390E127 | 1390C127 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Чтобы выбрать нужное нам слово в этом дампе, смотрим на смещение = последние 12 бит = 0х014 = слово 13908127

Физический адрес страницы = физический адрес таблицы страниц + индекс таблицы страниц\*4

0х13908000 + 0х179\*4(150810) = 0x13908000 + 0x5E4 = 0x139085E4

|  |
| --- |
| 139085E0: 25023127 25024127 25025127 25026127 25027127 25028127 25029127 2502A127 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 25023127 | 25024127 | 25025127 | 25026127 | 25027127 | 25028127 | 25029127 | 2502A127 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Ищем смещение

25024000 + 0х333 = 25024333

Ищем ближайший адрес к искомому, дальше от нашего отнимаем выбранный и.

Смещение 13.

|  |
| --- |
| 25024320: 272ED887 43961093 6D8233A3 75197874 75BF6DD3 7ED36C4A 46515B0A 4804FEAF |

Ответ 75

03 02 01 00 | 07 06 05 04 | 0B 0A 09 08 | 0F 0E 0D 0C | 13 12 11 10 | 17 16 15 14 | 1B 1A 19 18 | 1F 1E 1D 1C

14. Поясните понятие «виртуальная память».

метод управления памятью процессора, предназначенный для выполнения программ, которым выделяется адресное пространство, превышающее доступный физический объем памяти компьютера.

15. Поясните понятие «свопинг».

механизм OS обмена (вытеснения и загрузки) содержимым блоков оперативной физической памяти компьютера с устройством хранения данных с целью расширения адресуемого объема оперативной памяти компьютера. Механизм является аппаратно-программным.

16. Поясните понятие «страничная память».

страничная память – реализации виртуальной памяти, при которой физическая память и адресное пространство разбивается на блоки (страницы), а также осуществляется страничный свопинг. Размеры страниц для X86-64: 4K, 2MB, 1GB.

17. Поясните понятие MMU.

MMU – Memory Management Unit – диспетчер памяти – аппаратное (программируемое) устройство, входящее в состав процессора и предназначенное для трансляции виртуальных адресов оперативной памяти в реальные.

18. Поясните понятие TLB.

TLB - Translation Lookaside Buffer – буфер быстрого преобразования адреса – ассоциативная память (параллельный поиск), x86\_64 – 64 строки.

19. Какая информация содержится в строке таблицы страниц

****

20. Поясните принцип применения хэш-таблиц.

Ключ преобразуется в индекс через хэш-функцию.

По индексу осуществляется доступ к элементу.

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

инвертированные таблицы для ускорения поиска номера реальной таблицы, обычно реализована в виде хэш-таблицы.

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

Код программы: область, где хранится исполняемый код.

Данные: область для глобальных и статических данных.

Heap (куча): область для динамического выделения памяти.

Stack (стек): область для вызовов функций, локальных переменных.

Маппированные файлы: память, отображенная на файлы.

Ядро (Kernel space): область памяти, зарезервированная для ОС.

23. Windows: какой стандартный начальный размер области heap?

По умолчанию – 1MB, из них 4K сразу забирает процесс.

24. Windows: какой стандартный размер области памяти stack?

stack – область стека для потоков (по умолчанию 1MB).

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

это набор страниц, которые процесс активно использует в оперативной памяти.

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

Windows: Heap – область памяти адресного пространства, предназначенного для использования программной фрагментов динамически выделяемой памяти (malloc, new).

Heap процесса — основная куча, автоматически создаваемая для процесса ОС.

Пользовательская heap — куча, создаваемая приложением с использованием API, например, HeapCreate.

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

Текст (код): содержит исполняемый код программы.

Данные (data): хранит глобальные и статические переменные.

BSS: область для неинициализированных данных.

Heap (куча): область для динамического выделения памяти.

Маппированные файлы: отображенные файлы и библиотеки.

Stack (стек): область для вызовов функций и локальных переменных.

Kernel space: область памяти ядра, недоступная для пользователя.

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

выделяется в области heap (кучи).