Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Операционные системы**

Студент: Козека Е. М.

ФИТ 3 курс 4 группа

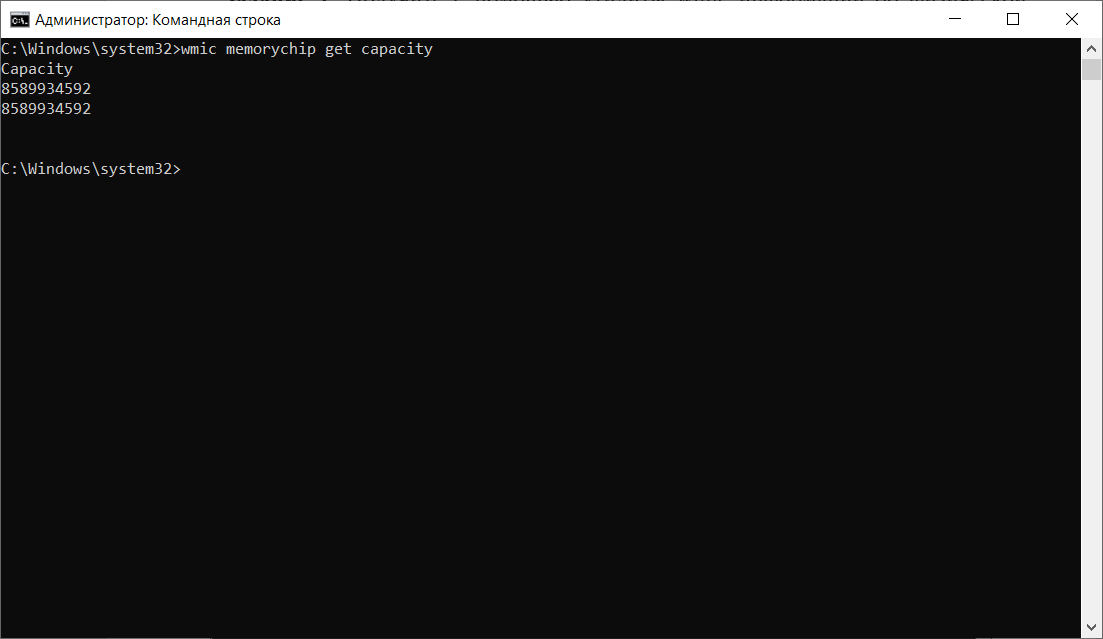
Преподаватель: Комкова А. В.

Минск 2024

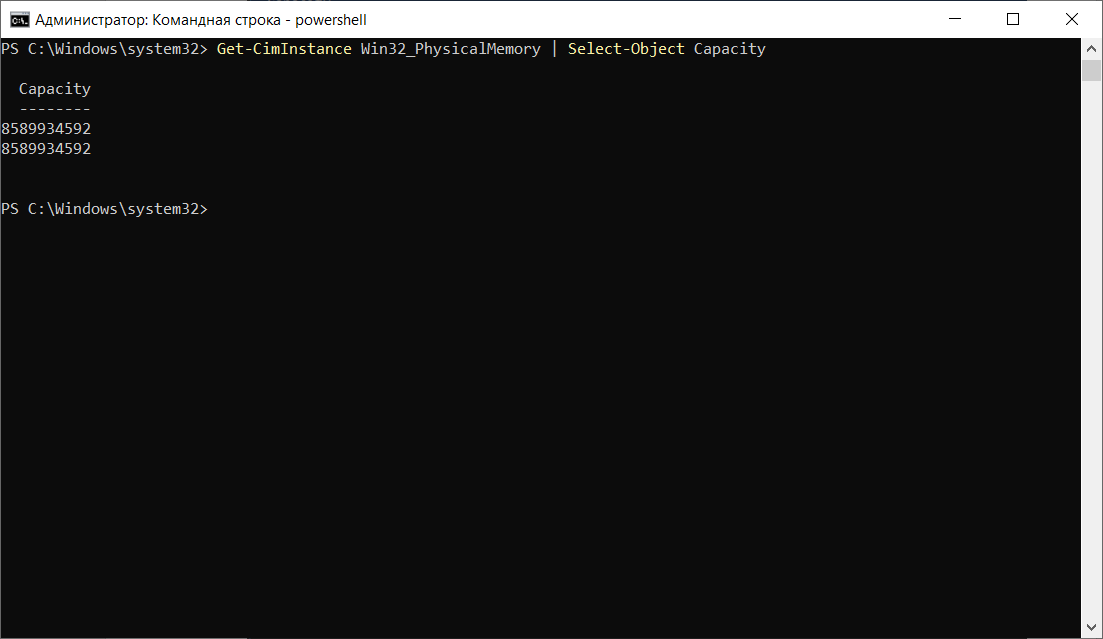
**Лабораторная работа №8. Управление памятью**

**Задание 1.** Получите с помощью утилиты wmic информации об физической оперативной памяти компьютера, поясните эту информацию. Получите с помощью утилиты powershell информации об физической оперативной памяти компьютера, поясните эту информацию. Получите с помощью утилиты systeminfo информации об оперативной памяти компьютера, поясните эту информацию. Получите с помощью утилиты performance monitor информации об оперативной памяти компьютера, поясните эту информацию.

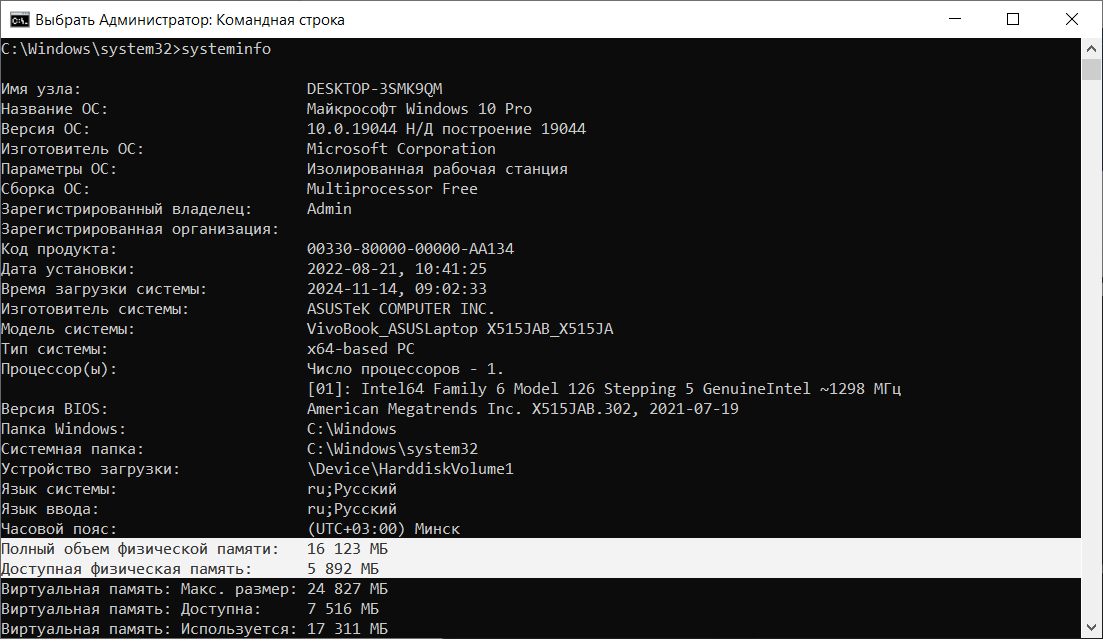
wmic:



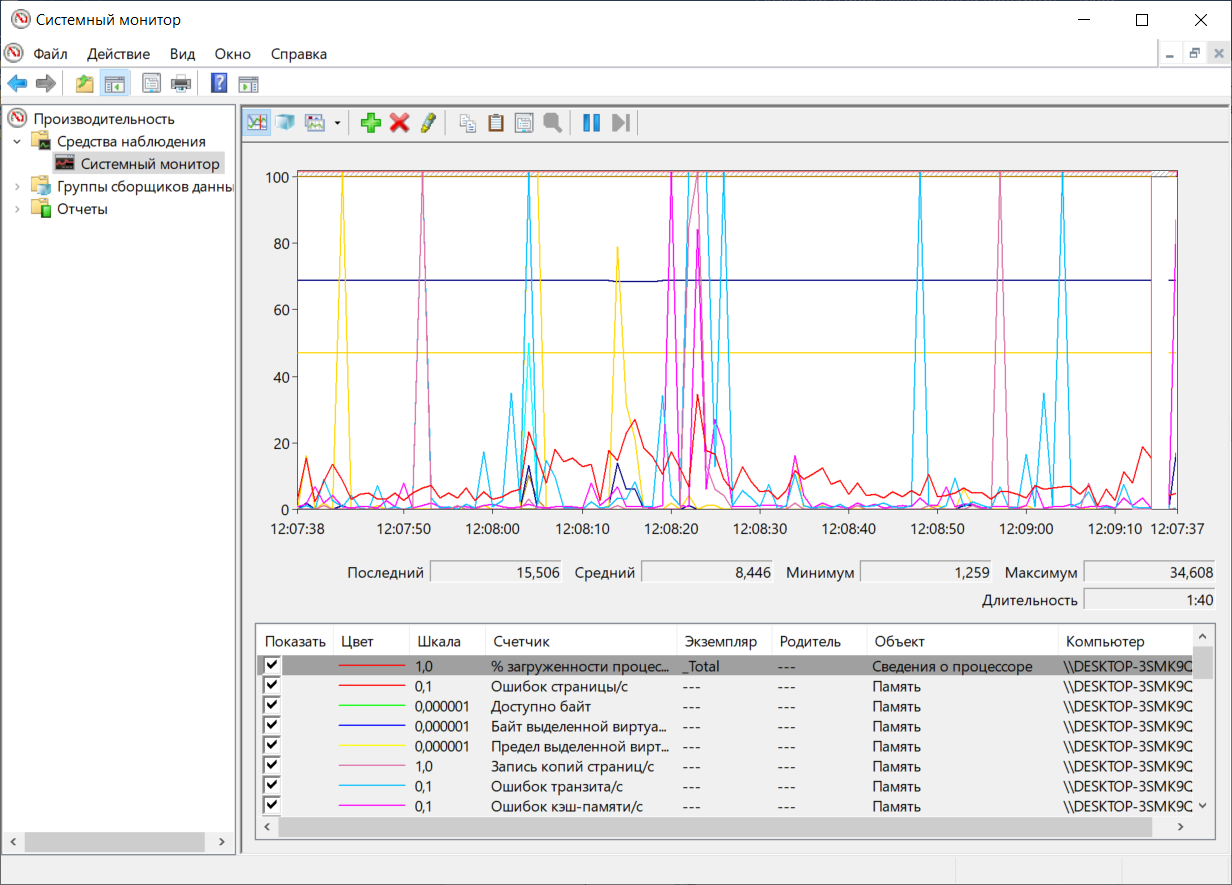
powershell:



systeminfo:

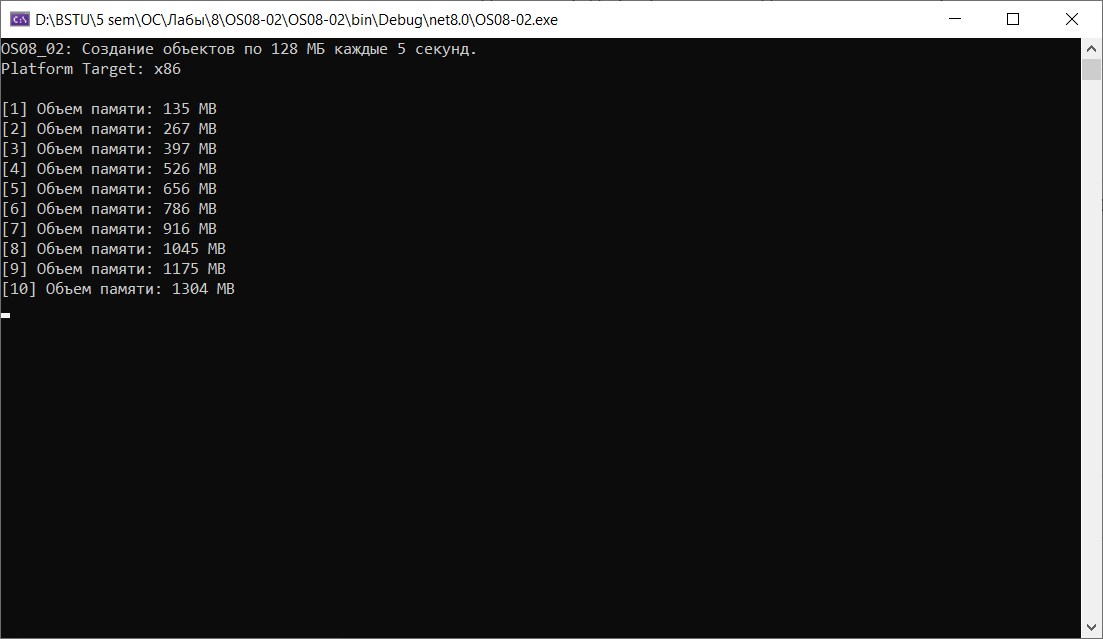


Performance Monitor:

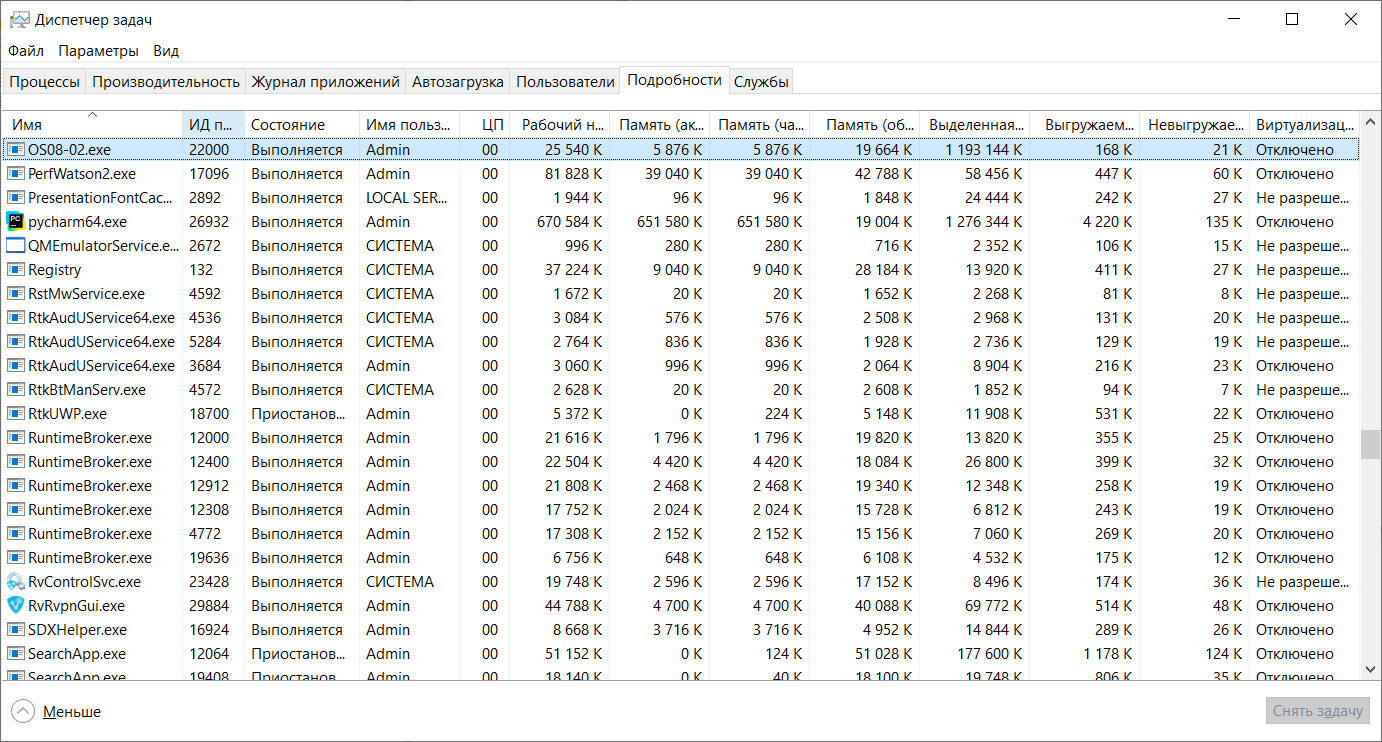


**Задание 2.** Разработайте на языке программирования C# консольное приложение OS08\_02, которое каждые 5 секунд создает новый объект размером 128 МБ и выводит объем используемой памяти, установите Platform Target x86.

Результат выполнения в консоли:



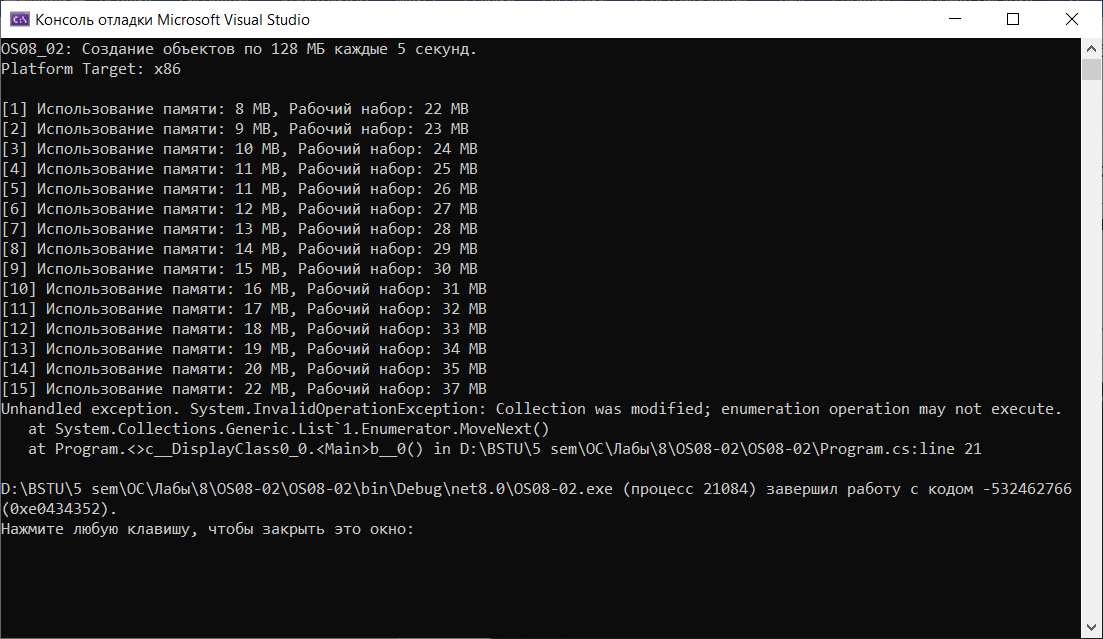
Понаблюдайте за выделенной памятью и рабочим набором памяти приложения OS08\_02 в Диспетчере задач. Будет ли приложение работать бесконечно?

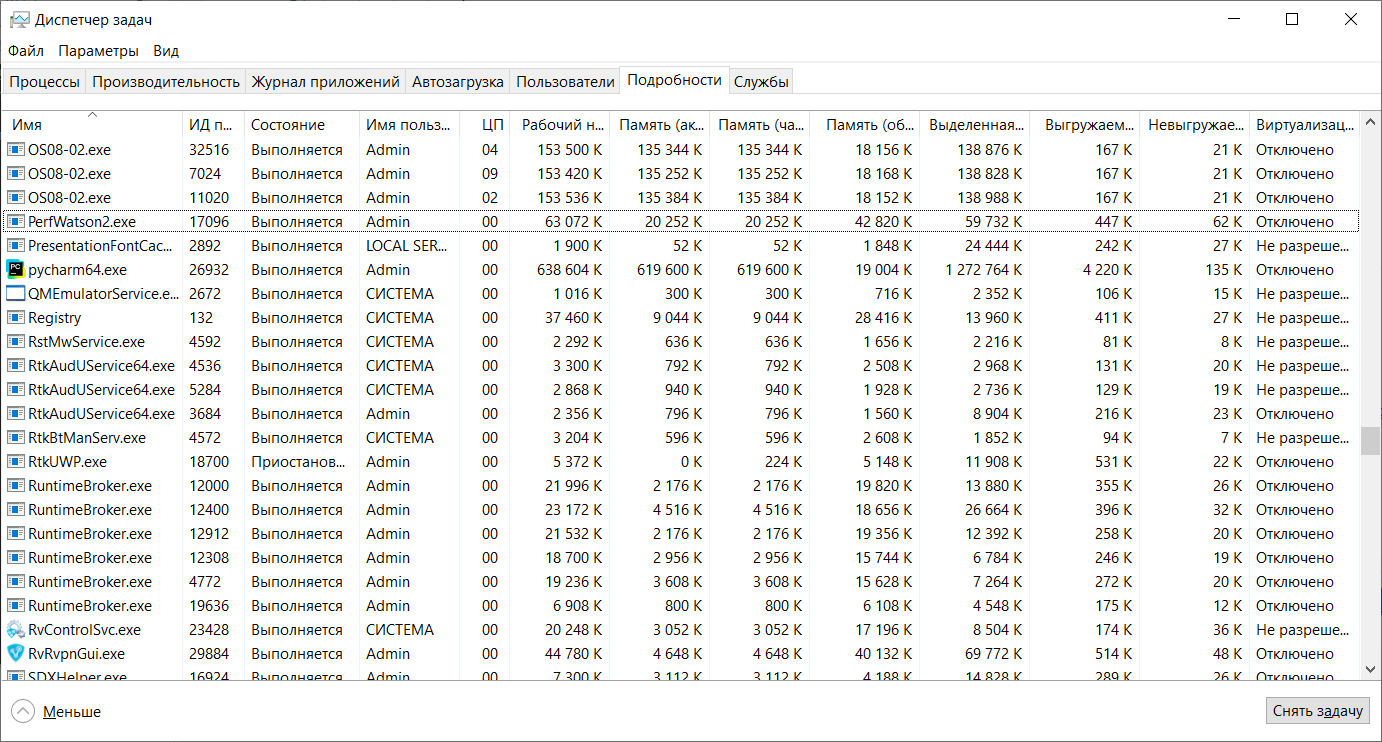


Приложение OS08\_02 не будет работать бесконечно, так как оно ограничено памятью (1,14 ГБ). Постоянное выделение памяти приводит к её исчерпанию, после чего программа аварийное завершится с исключением OutOfMemoryException.

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

Результат выполнения в консоли:



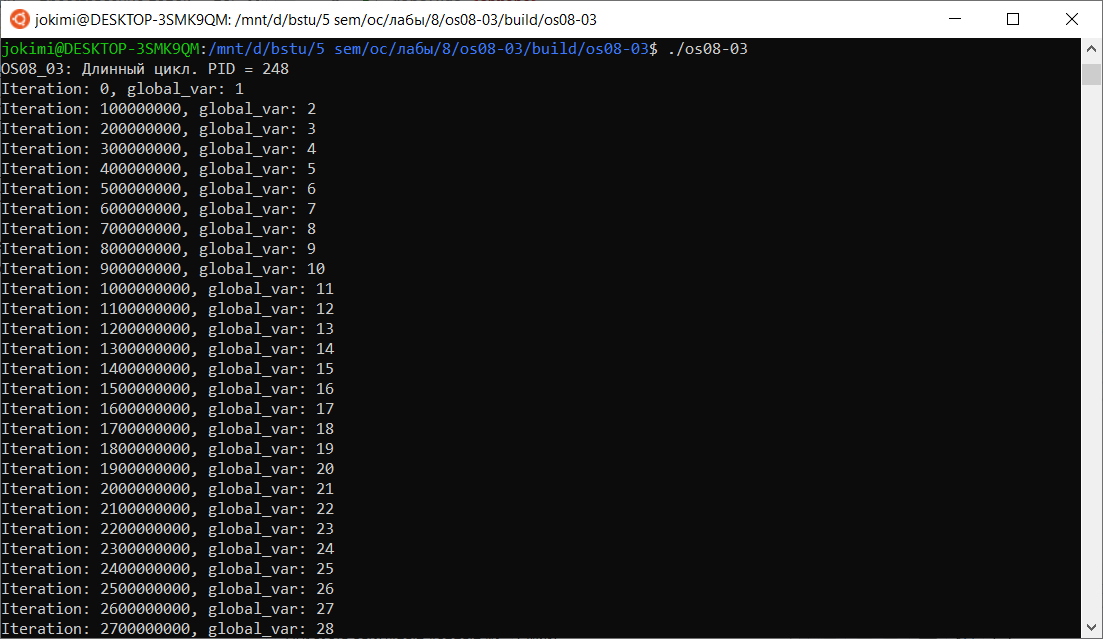


Рабочий набор стал больше, чем в исходной версии программы. Это связано с тем, что память действительно используется, так как массивы заполняются данными, и их содержимое подгружается в оперативную память. В предыдущей версии программы рабочий набор был значительно меньше выделенной памяти, так как память была выделена, но не использовалась активно.

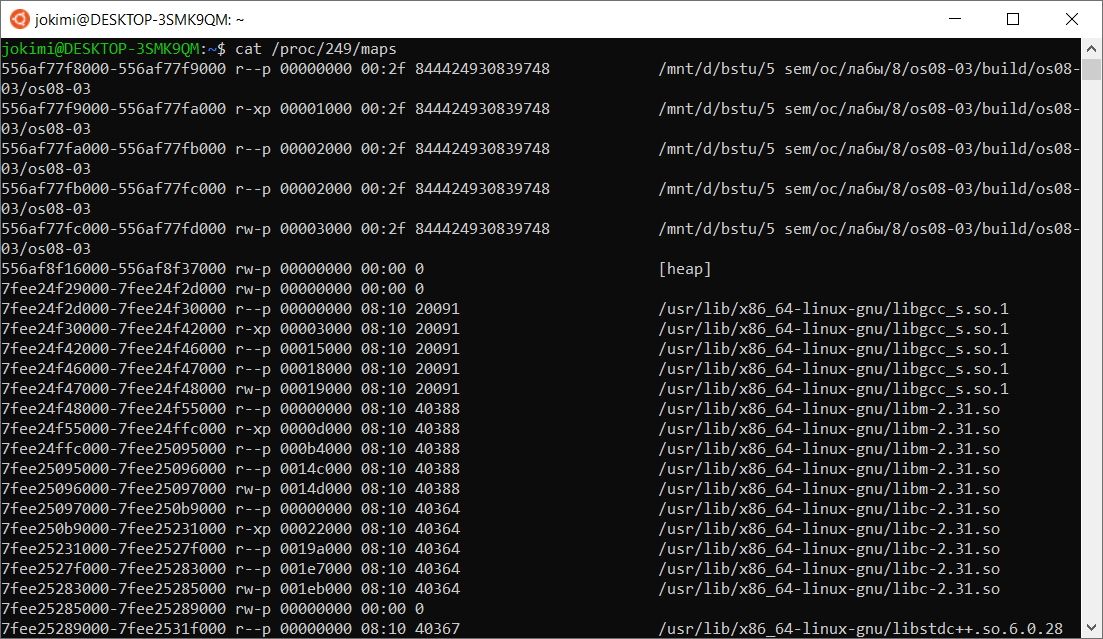
Общий рабочий набор будет увеличиваться пропорционально количеству экземпляров, так как каждый экземпляр требует свою память. Частный рабочий набор отображает объем памяти, использованный конкретным процессом.

**Задание 3.** Разработайте консольное приложение OS08\_03, выполняющее длинный цикл. Продемонстрируйте с помощью файловой системы /proc структуру адресного пространства. Продемонстрируйте с помощью pmap структуру адресного пространства. Определите с помощью утилиты objdump адрес загрузки main-модуля, секций с кодом, данными, неинициализированными глобальными переменными.

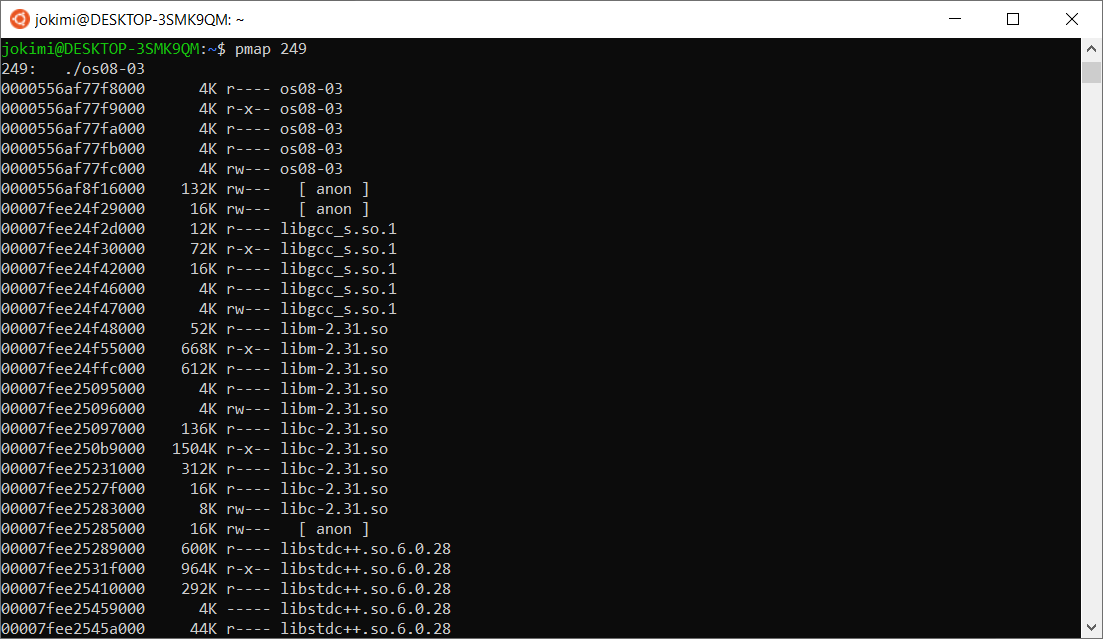
Результат выполнения в консоли:



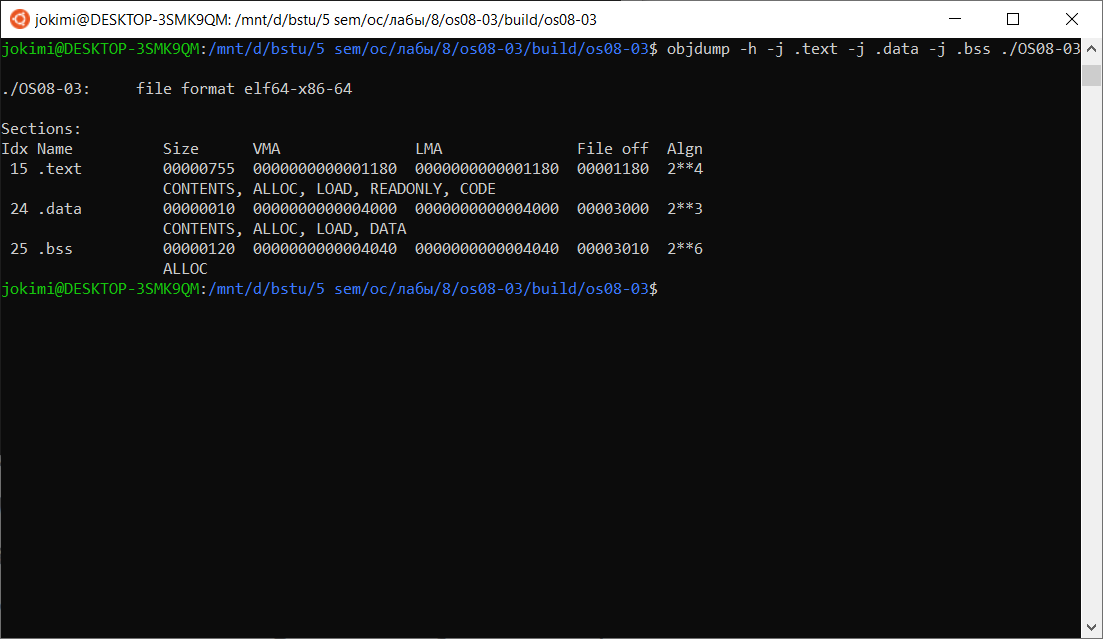
proc:



pmap:



objdump:



**Задание 4.** Укажите содержимое байта по виртуальному адресу 0x0167015F, если регистр CR3 содержит значение 0x13907000.

Виртуальный адрес в двоичном виде:

0000 0001 0110 0111 0000 0001 0101 1111.

Индекс каталога страниц: 0000 0001 01 → 5 (биты 31-22)

Индекс таблицы страниц: 10 0111 0000 → 0x270 (биты 21-12)

Смещение: 0001 0101 1111 → Ox15F (биты 11-0)

Базовый адрес физической страницы памяти: 0x139089CO

catAddr = CR3 + 4 \* PCI = 0x13907000 + (4\*5) = 0x13907014

tabBase = 0x13908000

tabAddr = tabBase + 4 \* PTI = 0x13908000 + (4\*624) = 0x139089CO

Искомый физический адрес: 0x2540315F

addr = tab + offset = 0x25403000 + 0x15F = 0x2540315F

Требуемые данные: 0x4A

0x25403150 → 78D06E5D

0x25403154 → 78B0E75A

0x25403158 → 0ABBB18B

0x2540315C → 4A24AF2A

0x2540315F → 4A

Ответы на вопросы:

**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.

**2. Поясните понятие «свопинг».**

Свопинг — это процесс передачи данных, хранящихся в оперативной памяти, на диск и обратно. Когда ОС замечает, что физ. память заполняется, она может переместить некоторые неиспользуемые данные на диск в файл подкачки, освобождая физ. память для новых данных. При необходимости ОС может снова загрузить данные из файла подкачки в оперативную память.

**3. Поясните понятие «страничная память».**

Страничная память — это механизм управления памятью, используемый ОС. В рамках концепции страничной памяти виртуальное адресное пространство программы разбивается на небольшие блоки, называемые страницами. Физ. память также разбивается на блоки одинакового размера (фреймы). Страничная память позволяет отображать вирт. страницы на физ. фреймы, позволяя программам эффективно использовать физ. память.

**4. Поясните понятие MMU.**

MMU — это устройство, обычно встроенное в процессор, которое отвечает за преобразование вирт. адресов, используемых программами, в соответствующие физ. адреса в оперативной памяти. Он играет ключевую роль в реализации вирт. памяти.

**5. Поясните понятие TLB.**

TLB — 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.

**6. Какая информация содержится в строке таблицы страниц?**

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

**7. 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.

**8. Поясните применение «инвертированной таблицы физических страниц» страничной памяти.**

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

**9. 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:

* Нижняя область адресного пространства:

1. Сегмент кода — исполняемый код программы.
2. Сегмент данных — глобальные и статические данные программы.
3. Сегмент стека — стек вызовов функций и локальные переменные.

* Область адресного пространства, зарезервированная для системы — зарезервирована для нужд ОС и не доступна для польз. приложений.
* Область адресного пространства, зарезервированная для пользовательских приложений — доступна для пользовательских приложений для выделения памяти и размещения данных.

**10. Windows: какой стандартный начальный размер области heap?**

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

**11. Windows: какой стандартный размер области памяти stack?**

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

**12. Windows: что такое «рабочее множество»? Поясните принцип управления рабочим множеством с помощью OS API.**

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.

**13. 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 — это основной heap, создаваемый для каждого процесса при его запуске.

Пользовательская heap — это heap, создаваемый польз. приложением для выделения и управления памятью.

Устройство heap основано на списке блоков памяти различных размеров, известном как список свободных блоков. 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.

**14. 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:

* Нижняя область адресного пространства:

1. Область кода — исполняемый код программы.
2. Область данных — глобальные и статические данные программы.
3. Область BSS — неинициализированные глобальные и статические пер-ные.
4. Область стека — стек вызовов функций и локальные переменные.

* Область адресного пространства, зарезервированная для библиотек и разделяемых объектов — библиотеки, используемые программой, и разделяемые объекты, кот. могут быть загружены и используемы неск. процессами одновременно.
* Область адресного пространства, зарезервированная для отображения файлов в память — 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.
* Область адресного пространства, зарезервированная для стека сигналов — стек, используемый для обработки сигналов, кот. могут быть отправлены процессу.
* Область адресного пространства, зарезервированная для динамической загрузки библиотек — библиотеки, которые загружаются динамически во время выполнения программы.
* Область адресного пространства, зарезервированная для кучи — выделение и управление динамической памятью с помощью ф-ий, таких как malloc и calloc.

**15. 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, 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.