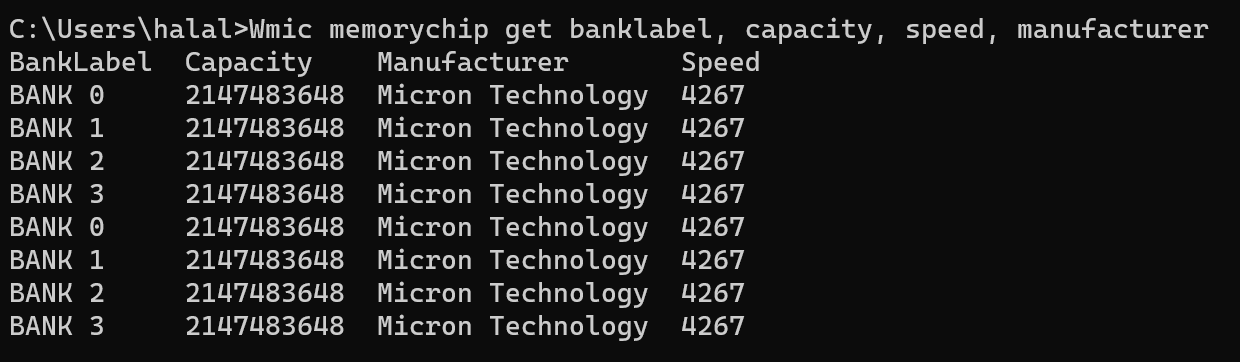
Лабораторная работа 06

Управление памятью

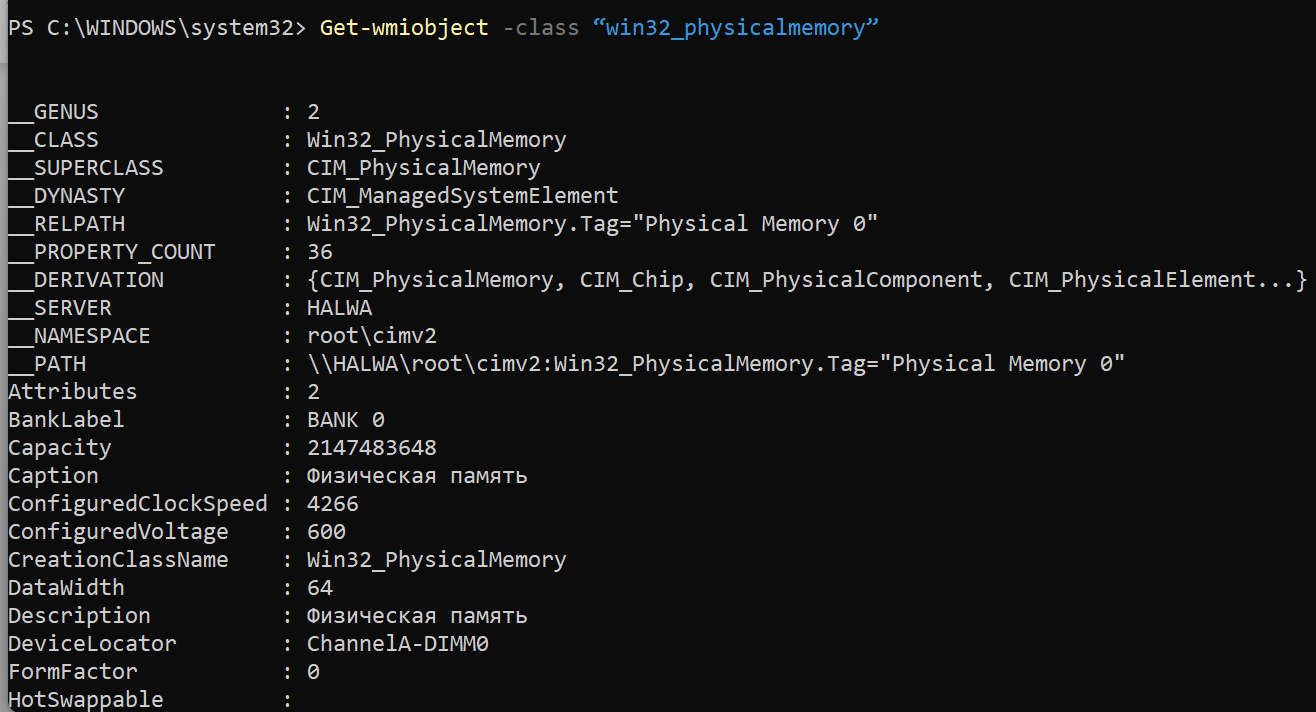
OC, ПОИТ-3

**Задание 01. Windows**

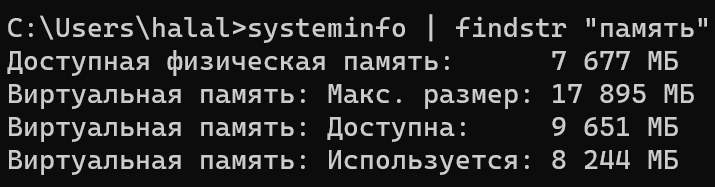
1. Получите с помощью утилиты **wmic** информации об физической оперативной памяти компьютера, поясните эту информацию.



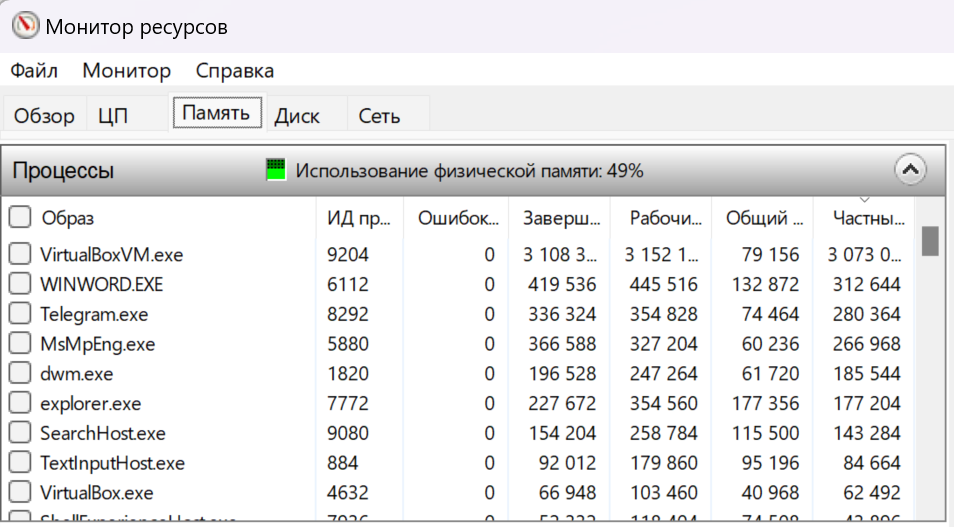
1. Получите с помощью утилиты **powershell** информации об физической оперативной памяти компьютера, поясните эту информацию.



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 **OS06\_02**, выполняющее длинный цикл.
2. Продемонстрируйте с помощью отладчика адреса расположения модулей приложения **OS06\_02**.
3. Установите для приложения **OS06\_02** стандартный адрес загрузки в память.
4. Продемонстрируйте с помощью отладчика стандартный адрес расположения модулей приложения **OS06\_02**

**Задание 03.Windows**

1. Разработайте консольное приложение **OS06\_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.

**Задание 04.Windows**

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

**Задание 06.Linux**

1. Разработайте консольное приложение **OS06\_06**, выполняющее длинный цикл.
2. Продемонстрируйте с помощью файловой системы **/proc** структуру адресного пространства.
3. Продемонстрируйте с помощью **pmap**  структуру адресного пространства.
4. Определите с помощью утилиты objdump адрес загрузки main-модуля, секций с кодом, данными, неинициализированными глобальными переменными.

Cat /proc/pid/maps

Pmap pid

objdump -h ./06\_06

**Задание 07.Linux**

1. Разработайте консольное приложение **OS06\_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.

**Задание 08** Ответьте на следующие вопросы

1. Поясните понятие «виртуальная память».
2. Поясните понятие «свопинг».
3. Поясните понятие «страничная память».
4. Поясните понятие MMU.
5. Поясните понятие TLB.
6. Какая информация содержится в строке таблицы страниц
7. Поясните принцип применения хэш-таблиц.
8. Поясните применение «инвертированной таблицы физических» страничной памяти.
9. Поясните понятие «рабочий набор страниц».
10. Поясните принцип работы алгоритма LRU.
11. Windows: поясните назначение сервиса SysMain.
12. Windows: поясните назначение файла hiberfil.sys.
13. Windows: поясните назначение файла pagefile.sys.
14. Windows: поясните назначение файла swapfile.sys.
15. Windows: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.
16. Windows: какой стандартный начальный размер области heap?
17. Windows: каким образом можно изменить начальный размер области памяти heap приложения?
18. Windows: какой стандартный размер области памяти stack?
19. Windows: каким образом можно изменить размер области памяти stack приложения?
20. Windows: поясните назначение функции Windows API: GlobalMemoryStatus.
21. Windows: поясните назначение функции Windows API: VirtualQuery; перечислите значения атрибутов Protect, State и Type.
22. Windows: что такое «рабочее множество»? поясните принцип управления рабочим множеством с помощью OS API.
23. Windows: что означает «страница заблокирована»? с помощью каких функций OS API можно установить блокировку страниц и снять блокировку? Какое максимальное количество страниц можно заблокировать?
24. Windows: что такое «heap»? Что такое «heap процесса»? Что такое «пользовательская heap»? Поясните принцип устройства heap.
25. Linux: перечислите области адресного пространства (от младших к старшим адресам) и поясните их назначения.
26. Linux: в какой части адресного пространства выделяется памяти с помощью функций malloc, calloc?

1. Поясните понятие «виртуальная память».

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

2. Поясните понятие «свопинг».

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

3. 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

4. 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

5. Поясните понятие TLB.

TLB(Translation Lookaside Buffer) – буфер быстрого преобразования адреса; компонент MMU, предназначенный для вычисления реальных адресов, хранит 64 строки таблицы страниц, полностью таблица хранится во вторичной (диск) памяти без свопинга;

6. Какая информация содержится в строке таблицы страниц

1. блокировка кэша – блокируется кэширование реальных страниц, которые содержат регистры портов ввода/вывода.

2. защита, обращение и изменение - используется в алгоритмах вытеснения.

7. Поясните принцип применения хэш-таблиц.

решает проблему инвертированной таблицы: которая усложняла преобразование виртуальных адресов в физические

Одним из приемлемых способов осуществления этого поиска является ведение хэш-таблицы, созданной на основе виртуальных адресов. все находящиеся на данный момент в памяти виртуальные страницы, имеющие одинаковые хэш-значения, связываются в одну цепочку. Как только будет найден номер страничного блока, в TLB будет введена новая пара значений (виртуального, физического)

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

инвертированная таблица страниц – таблица для физических страниц;

В конструкции имеется одна запись для каждого страничного блока в реальной памяти, а не одна запись на каждую страницу в виртуальном адресном пространстве

9. Поясните понятие «рабочий набор страниц».

Набор страниц, который процесс использует в данный момент, известен как рабочий набор

10. Поясните принцип работы алгоритма LRU.

LRU (least recently used) — это алгоритм, при котором вытесняются значения, которые дольше всего не запрашивались. Необходимо хранить время последнего запроса к значению, как только число закэшированных значений превосходит N необходимо вытеснить из кеша значение, которое дольше всего не запрашивалось.

11. Windows: поясните назначение сервиса SysMain.

SysMain - это функция, которая работает в фоновом режиме , постоянно анализирует модели использования ОЗУ и выясняет, какие приложения запускаются чаще всего. Со временем SysMain отмечает эти приложения как «часто используемые» и предварительно загружает их в оперативную память, когда вы хотите запустить приложение, оно запустится намного быстрее, потому что оно уже загружено в память.

12. Windows: поясните назначение файла hiberfil.sys.

hiberfile.sys - файл для сохранения памяти в режиме «сон» (гибернация)

13. Windows: поясните назначение файла pagefile.sys.

pagefile.sys - файл подкачки;

14. Windows: поясните назначение файла swapfile.sys.

swapfile.sys - файл подкачки отдельных (предварительно скаченных из магазина приложений UWP) для быстрого применения (в случае надобности)

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

PEB – Process Environment Block

TEB – Thread Environment Block

DLL – Dynamic Link Library

heap – фрагмент памяти адресного пространства (по умолчанию 1MB), предназначенный для динамического использования

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

16. Windows: какой стандартный начальный размер области heap?

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

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

Можно установить стартовое значение величины HEAP в параметрах Linker.

в Visual Studio : Проект -> Свойства -> Свойства конфигурации -> Компоновщик -> Все параметры -> Резервный размер кучи.

если компилировать через консоль: /HEAP:”2097152” компоновщик

18. Windows: какой стандартный размер области памяти stack?

1MB (Лекция)

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

в Visual Studio : Проект -> Свойства -> Свойства конфигурации -> Компоновщик -> Все параметры -> Резервный размер стека.

если компилировать через консоль: /STACK:”2097152”

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

GlobalMemoryStatus получает информацию о текущем состоянии памяти компьютера

MEMORYSTATUS ms; // структура в которую записывается инфа

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

Извлекает информацию о диапазоне страниц в виртуальном адресном пространстве вызывающего процесса.

Атрибут Protect (защита): Определяет уровень защиты доступа к виртуальной памяти и может принимать значения, такие как PAGE\_NOACCESS (нет доступа), PAGE\_READONLY (только для чтения), PAGE\_READWRITE (чтение/запись) и другие.

Атрибут State (состояние): Определяет текущее состояние виртуальной памяти и может быть MEM\_COMMIT (выделена и используется), MEM\_RESERVE (зарезервирована, но не выделена), MEM\_FREE (освобождена) и т. д.

Атрибут Type (тип): Определяет тип региона виртуальной памяти и может быть MEM\_IMAGE (регион содержит образ исполняемого файла), MEM\_MAPPED (регион отображен на файл или устройство), MEM\_PRIVATE (регион является частным для процесса) и т. д.

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

Рабочее множество - количество памяти, требующееся процессу в заданный интервал времени.

Процесс уменьшает или опустошает рабочий набор, вызывая функцию SetProcessWorkingSetSize (Устанавливает минимальный и максимальный размеры рабочего набора для указанного процесса.)

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

VirtyalLock() Блокирует указанную область виртуального адресного пространства процесса в физической памяти, гарантируя, что последующий доступ к области не приведет к ошибке страницы.

Страницы, заблокированные процессом, остаются в физической памяти до тех пор, пока процесс не разблокирует их или не завершит работу. Эти страницы гарантированно не будут записаны в файл подкачки, пока они заблокированы.

Максимальное количество страниц, которые может заблокировать процесс, равно количеству страниц в его минимальном рабочем наборе за вычетом небольших накладных расходов.

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

Heap (куча) - это область виртуальной памяти операционной системы, используемая для динамического выделения памяти в процессе выполнения программы (malloc, new).

Heap процесса - это куча, специфичная для каждого процесса. Каждый процесс имеет собственную область памяти, выделенную под heap, в которой он может выделять и освобождать память по мере необходимости.

Пользовательская heap - это управляемая область кучи в рамках процесса, которую программист может использовать для динамического выделения памяти. Она предоставляет программисту возможность выделять и освобождать память по требованию, что позволяет гибко управлять памятью во время выполнения программы.

Принцип устройства heap включает в себя ведение списка свободных блоков памяти и их аллокацию для удовлетворения запросов выделения памяти.

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

Code: отображение программного кода в памяти

Data: отображение инициализированных переменных

Heap: используется для выделения памяти во время выполнения программы. Но в отличие от стека, память, выделенная в куче, сохраняется и после того, как функция, вызвавшая выделение этой памяти, завершит работу. Язык С предоставляет программисту целый ряд средств управления памятью в куче.

NotUsed

Stack: стек работает по схеме LIFO (последним вошел, первым вышел). Всякий раз, когда вызывается новый метод, содержащий примитивные значения или ссылки на объекты, то на вершине стека под них выделяется блок памяти.

Когда метод завершает выполнение, блок памяти, отведенный для его нужд, очищается, и пространство становится доступным для следующего метода. Переменные в стеке существуют до тех пор, пока выполняется метод в котором они были созданы.

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

heap