# 1 Задание

-cmd/wmic

**wmic memorychip get banklabel, capacity, speed, manufacturer**

Столбец **Capacity** (с англ. Capacity - Вместительность, Ёмкость, Объём) показывает нам объём установленных модулей памяти в байтах для каждого отдельно (в данном примере установлено два модуля памяти).

Столбик "**Speed**" отобразит скорость в МГц каждой планки.

Столбец **Manufacturer** отображает производителя модуля памяти. В зависимомти от модели и производителя модуля памяти параметр Manufacturer может быть пустым.

**wmic memcache get purpose, maxcachesize (Кеш процессора ?)**

(Команда WMIC (Windows Management Instrumentation Command) используется для получения сведений об оборудовании и системе, управления процессами и их компонентами, а также изменения настроек с использованием возможностей инструментария управления Windows (Windows Management Instrumentation или WMI) .)

-powershell

**Get-WmiObject -Class "Win32\_PhysicalMemory"**

-cmd/systeminfo

systeminfo | findstr "памят"

-performance monitor

# **3 Задание**

**VirtualAlloc** - выделить или зарезервировать страницы виртуальной памяти.

1. задает начальный адрес. (**NULL** и тогда операционная система сама выберет этот адрес.)
2. Размер области в байтах
3. Тип выделения памяти

**MEM\_COMMIT** - Выполняется выделение страниц памяти для непосредственной работы с ними. Выделенные страницы заполняются нулями.

1. Защита памяти для области выделяемых страниц

**VirtualFree** - выпускает, списывает или освобождает и списывает область страниц в виртуальном адресном пространстве вызывающего процесса.

1. Указатель на базовый адрес области освобождаемых страниц.
2. Размер области памяти, подлежащей освобождению, в байтах.
3. Тип свободной операции.

**MEM\_RELEASE** - Освобождает указанную область страниц. После этой операции страницы находятся в свободном состоянии.

# **4 Задание**

Heap – область памяти адресного пространства, предназначенного для использования программной фрагментов динамически выделяемой памяти (malloc, new). По умолчанию – 1MB, из них 4K сразу забирает процесс. Можно установить стартовое значение величины HEAP в параметрах Linker. По мере new(malloc) размер HEAP прирастает. Память выделяется с учетом минимизации фрагментации.

Каждый процесс имеет кучу, предоставленную операционной системой. Извлекает дескриптор кучи процесса

Еще раз про дескриптор оно позволяет однозначно идентифицировать некий объект операционной системы и использовать себя (дескриптор) в качестве ссылки - указывать функции с каким над каким именно объектом нужно произвести операцию. По-русски, их можно назвать как описатели или манипуляторы. Т.е. как переменная классового типа позволяет сослаться на некий объект в вашем приложении, так дескриптор позволяет сослаться на некий объект операционной системы. Дескрипторы позволяют ссылаться на такие объекты, как окна, объекты ядра (файлы, процессы, потоки, объекты синхронизации и др), графические объекты (кисти, перья), некоторые другие объекты (исполняемый модуль, ловушка).

**GetProcessHeap -** Ссылку на кучу можно получить, воспользовавшись функцией

**PROCESS\_HEAP\_ENTRY** структура - содержит информацию об элементе кучи

**HeapWalk** - Выполняет обход кучи и возвращает сведения о следующей записи.

1. Буфер, который будет содержать данные кучи.

# **5 задание**

**HeapCreate** - Создает частный объект кучи, который может быть использован вызывающим процессом. Функция резервирует место в виртуальном адресном пространстве процесса и выделяет физическое хранилище для указанной начальной части этого блока.

1. Параметры выделения кучи.

**HEAP\_NO\_SERIALIZE** - Сериализованный доступ не используется, когда функции кучи обращаются к этой куче.

**HEAP\_ZERO\_MEMORY** - установить содержимое выделяемой памяти в 0

1. Начальный размер кучи, в байтах.
2. Максимальный размер кучи, в байтах.

**HeapAlloc** - Выделяет блок памяти из кучи. Выделенная память не является подвижной.

1. Дескриптор кучи, из которой будет выделена память.

# Параметры выделения кучи.

1. Количество выделяемых байтов.

# **6 задание**

Каждая строка в cat /proc/$PID/maps описывает область непрерывной виртуальной памяти в процессе или потоке. Каждая строка имеет следующие поля:



* **адрес**. Это начальный и конечный адрес региона в адресном пространстве процесса
* **разрешения**. Здесь описывается доступ к страницам в регионе. Существует четыре различных разрешения: чтение, запись, выполнение и совместное использование. Если чтение /запись /выполнение отключены, вместо - /w /x. Если регион не является общим, он является приватным, поэтому p появляются вместо s. Если процесс пытается получить доступ к памяти способом, который не разрешен, генерируется ошибка сегментации. Разрешения можно изменить с помощью системного вызова mprotect.
* **смещение** - если область была сопоставлена ​​из файла (с использованием mmap), это смещение в файле где начинается отображение. Если память не была отображена из файла, это просто 0.
* **устройство**. Если регион был сопоставлен из файла, это основной и младший номер устройства (в шестнадцатеричном формате), в котором находится файл.
* **inode**. Если регион был сопоставлен с файлом, это номер файла.
* **путь**. Если регион был сопоставлен с файлом, это имя файла. Это поле пустое для анонимных сопоставленных регионов. Существуют также специальные области с именами, такими как [heap], [stack] или [vdso]. [vdso] обозначает виртуальный динамический общий объект. Он используется системными вызовами для переключения в режим ядра.

**pmap** – Использование памяти процессами.

**pmap** PID

Команда детально расписывает использование оперативной памяти процессами в системе.

Параметр**-x** можно использовать для предоставления информации о выделении памяти и типах отображения для каждого отображения. Для каждого сопоставления отображается объем резидентной, не совместно используемой анонимной и заблокированной памяти

objdump -f ./OS06\_06

display information from object files

[**-f**|**--file-headers**]

**ТЕОРИЯ**

**адресное пространство** –непрерывный диапазон адресов, выделяемый OS процессу; у каждого процесса свое адресное пространство.

**1. Поясните понятие «виртуальная память».**

**виртуальная память (1961г)** –метод управления памятью процессора, предназначенный для выполнения программ, которым выделяется адресное пространство превышающее доступный физический объем памяти компьютера. (Лекция)

Второй подход называется виртуальной памятью, он позволяет программам запускаться даже в том случае, если они находятся в оперативной памяти лишь частично (Книга)

**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. (Лекция)

Самый простой подход для преодоления перегрузки памяти, называемый **свопингом**, заключается в размещении в памяти всего процесса целиком, его запуске на некоторое время, а затем сбросе на диск. (Книга)

**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**. (Лекция)**

Виртуальное адресное пространство состоит из блоков фиксированного размера, называемых страницами. Соответствующие блоки в физической памяти называются страничными блоками (**Книга**)

В основе виртуальной памяти лежит идея, что у каждой программы имеется собственное адресное пространство, которое разбивается на участки, называемые страницами. Каждая страница представляет собой непрерывный диапазон адресов. Эти страницы отображаются на физическую память, но для запуска программы одновременное присутствие в памяти всех страниц необязательно(**Книга**)

**4. Поясните понятие MMU.**

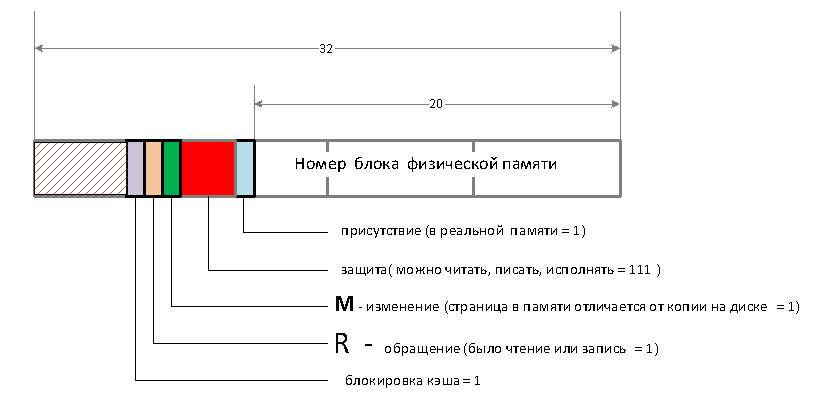
**MMU – Memory Management Unit – диспетчер памяти –** аппаратное (программируемое) устройство, входящее в состав процессора и предназначенное для трансляции виртуальных адресов оперативной памяти в реальные. (**Лекция**)

сгенерированные программным способом адреса называются виртуальными адресами. На компьютерах, не использующих виртуальную память, виртуальные адреса выставляются непосредственно на шине памяти, что приводит к чтению или записи слова физической памяти с таким же адресом. При использовании виртуальной памяти виртуальные адреса не выставляются напрямую на шине памяти. Вместо этого они поступают в диспетчер памяти **(Memory Management Unit (MMU))**, который отображает виртуальные адреса на адреса физической памяти(**Книга**)

**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; (**Лекция**)

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



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

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

(**ЛЕКЦИЯ**)

**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). (**КНИГА**)

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

инвертированная таблица страниц – таблица для физических страниц; (**Лекция**) (замена многоуровневых таблиц страниц)

. В данной конструкции имеется одна запись для каждого страничного блока в реальной памяти, а не одна запись на каждую страницу в виртуальном адресном пространстве(**Книга**)

. В каждой записи отслеживается, что именно находится в страничном блоке (процесс, виртуальная страница)

**9. Поясните понятие «рабочий набор страниц».**

Набор страниц, который процесс использует в данный момент, известен как рабочий набор(**КНИГА**)

Проще говоря, рабочий набор - это набор страниц памяти, которые в настоящее время принадлежат вашему процессу и не выгружены (то есть в ОЗУ).

**10. Поясните принцип работы алгоритма LRU.**

**LRU** (least recently used) — это алгоритм, при котором вытесняются значения, которые дольше всего не запрашивались. Соответственно, необходимо хранить время последнего запроса к значению. И 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 впервые была известна в Windows Vista под другим названием SuperFetch. Он спокойно сидит в фоновом режиме, постоянно анализируя модели использования ОЗУ и выясняя, какие приложения вы запускаете чаще всего. Со временем SysMain отмечает эти приложения как «часто используемые» и предварительно загружает их в оперативную память. Идея в том, что, 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** -файлдля сохранения памяти в режиме «сон» (гибернация);(Лекция)

Файл **hiberfil.sys** — это файл гибернации, используемый в Windows для хранения данных и их последующей быстрой загрузки в оперативную память при включении компьютера или ноутбука. (<https://remontka.pro/chto-za-file-hiberfil-sys-i-kak-udalit/>)

**13. Windows: поясните назначение файла pagefile.sys.**

**pagefile.sys** -файл подкачки;(Лекция)

Зачем он нужен? Дело в том, что какое бы количество оперативной памяти не было установлено на вашем компьютере, не всем программам для работы будет ее достаточно. Современные игры, видео и графические редакторы и многое другое программное обеспечение с легкостью заполнит Ваши 8 Гб RAM и попросит еще. В этом случае и используется файл подкачки. (<https://remontka.pro/pagefile-sys/>)

**14. Windows: поясните назначение файла swapfile.sys.**

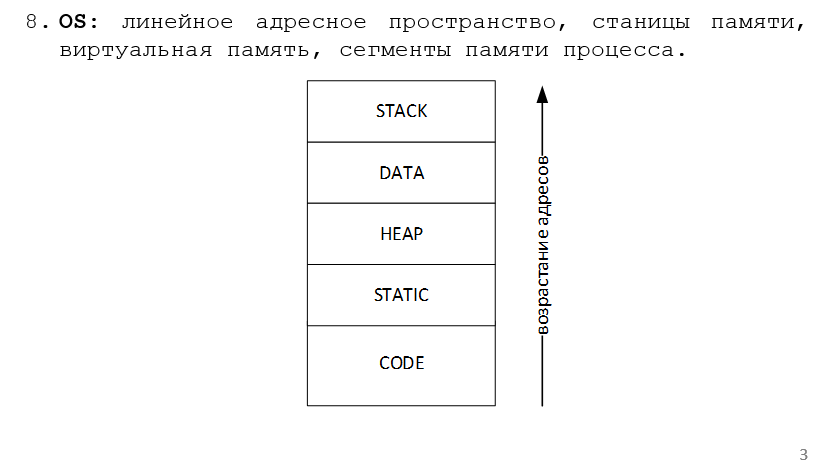
**swapfile.sys** -файл подкачки отдельных (предварительно скаченных из магазина приложенийUWP) для быстрого применения (в случае надобности). (Лекция)

Файл swapfile.sys ещё один файл подкачки (в дополнение к pagefile.sys), но служащий исключительно для приложений из магазина приложений (UWP).

В swapfile.sys записываются данные приложений из магазина (речь идет о «новых» приложениях Windows 10, ранее известных как Metro приложения, теперь — UWP), которые не требуются в настоящий момент времени, но могут внезапно потребоваться (например, при переключении между приложениями, открытии приложения из живой плитки в меню «Пуск»), и работает отличным от обычного файла подкачки Windows образом, представляя собой своего рода механизм «гибернации» для приложений.

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

CODE HEAP DATA STACK



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

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

**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”

**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.

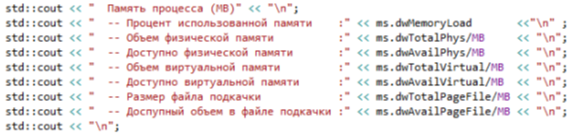
если компилировать через консоль: /STACK:”2097152”

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

**GlobalMemoryStatus** получает информацию о текущем состоянии памяти компьютера.

MEMORYSTATUS ms; // ствуктура в которую записывается инфа

GlobalMemoryStatus(&ms);



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.

****

****

****

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

**Рабочее множество** - количество памяти, требующееся [процессу](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) в заданный интервал времени.

*Рабочий набор процесса - это набор страниц в виртуальном адресном пространстве процесса, которые в настоящее время находятся в физической памяти.*

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

HANDLE hProcess, SIZE\_T dwMinimumWorkingSetSize, SIZE\_T dwMaximumWorkingSetSize

BOOL SetProcessWorkingSetSizeEx(

HANDLE hProcess,

SIZE\_T dwMinimumWorkingSetSize,

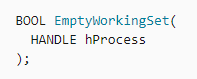
SIZE\_T dwMaximumWorkingSetSize,

DWORD Flags

);

Устанавливает минимальный и максимальный размеры рабочего набора для указанного процесса.

**или EmptyWorkingSet -** Удаляет как можно больше страниц из рабочего набора указанного процесса.

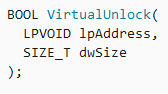
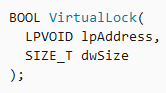


https://docs.microsoft.com/en-us/windows/win32/memory/working-set

Согласно **модели рабочего множества**, процесс может находиться в [ОЗУ](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BF%D0%BE%D0%BC%D0%B8%D0%BD%D0%B0%D1%8E%D1%89%D0%B5%D0%B5_%D1%83%D1%81%D1%82%D1%80%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE_%D1%81_%D0%BF%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%BE%D0%BB%D1%8C%D0%BD%D1%8B%D0%BC_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%BE%D0%BC) тогда и только тогда, когда множество всех его страниц, используемых в настоящее время (или множество последних по времени использования страниц, которое часто используется как его приближение) могут находиться в ОЗУ. Модель работает по принципу «всё или ничего», то есть, если число нужных процессу страниц памяти растёт и в ОЗУ нет свободного места, то процесс выгружается из памяти целиком, чтобы освободить память для использования другими процессами.

Часто сильно [загруженный](https://ru.wikipedia.org/wiki/Load_Average) компьютер может иметь столько процессов в очереди, что, если позволить им запуститься в один и тот же [квант](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%81%D0%BF%D0%B5%D1%82%D1%87%D0%B5%D1%80_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) времени, то объём памяти, на который они будут ссылаться, превысит объём ОЗУ, вследствие чего возникнет [пробуксовка](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B1%D1%83%D0%BA%D1%81%D0%BE%D0%B2%D0%BA%D0%B0_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) [виртуальной памяти](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C).

**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?**



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

При инициализации процесса система создает в его адресном пространстве кучу. (Ее размер по умолчанию — 1 Мб). Но система позволяет увеличивать этот размер, для чего надо указать компоновщику при сборке программы ключ /HEAP (Однако при сборке DLL этим ключом пользоваться нельзя, так как для DLL куча не создается.)

пользовательская heap - куча создаваемая пользователем

При запуске [процесса](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) [ОС](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) выделяет [память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) для размещения кучи. В дальнейшем [память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) для кучи (под кучу) может выделяться динамически.

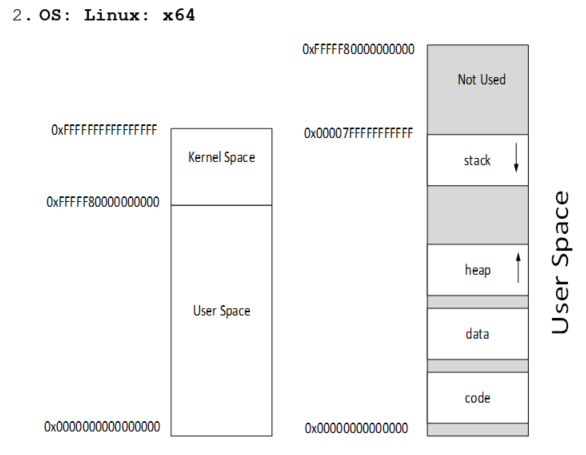
Программа пользователя, используя [функции](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), подобные malloc(), может получать [указатели](https://ru.wikipedia.org/wiki/%D0%A3%D0%BA%D0%B0%D0%B7%D0%B0%D1%82%D0%B5%D0%BB%D1%8C_(%D1%82%D0%B8%D0%BF_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)) на области [памяти](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C), принадлежащие куче. Программы используют кучу для размещения динамически создаваемых [структур данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Программа может освободить память с помощью [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), подобных free().

[Память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C) кучи можно разделить на **занятую** (выделенную программе с помощью [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), подобных malloc()) и **свободную** (ещё не занятую или уже освобождённую с помощью [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), подобных free()).

Для хранения данных о том, какая область кучи является занятой, а какая — свободной, обычно используется дополнительная область памяти.

Перед началом работы программы выполняется инициализация кучи, в ходе которой [память](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C), выделенная под кучу, отмечается как свободная.

**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?**

HEAP