Ответы на асм

1. **По каким соображениям в процессорах Pentium запрещено вызвать менее привилегированные процедуры, но разрешено вызывать менее привилегированные задачи?**

В первом случае запрет предотвращает передачу некоторой работы (функции) от более надежной процедуры менее надежной, во втором случае такой запрет не нужен, так как каждая задача выполняет собственную работу, и ее вызов не уменьшает надежность более привилегированного вызывающего кода, который, как правило, является кодом ядра ОС. Если бы такой запрет существовал, то ОС не смогла бы выполнять свои функции по переключению задач.

2. **Поддерживает ли процессор Pentium приоритезацию запросов прерывания между несколькими внешними устройствами?**

Все линии прерываний имеют свой приоритет. Чем выше приоритет у линии прерывания, тем быстрее процессор ответит на запрос от устройства находящегося на этой линии.
Чем больше ресурсов от компьютера требует устройство, тем более высокий приоритет должен быть у линии IRQ присвоенной этому устройству.
Таблица, линии расположены в порядке убывания приоритета 15->0

Системность	Линия IRQ	Устройство
S	0	Системный таймер
S	1	Клавиатура
S	2	Cascad (Вывод на вторую микросхему контроллера линий прерываний)
S	8	Часы реального времени
	9	ACPI Controller
	10	*Свободно*
	11	USB
S	12	PS2
S	13	Сопроцессор
	14	IDE Primary (Контроллер жестких дисков)
	15	IDE Secondary (Контроллер жестких дисков)
	3	Com Port 1 (Мышь)
	4	Com Port 2 (Модем)
	5	*Свободно*
S	6	Floppy (Дисковод)
	7	LPT (Принтер)

3. Основные отличия файловых систем FAT и NTFS.

FAT32: по сути, эта файловая система представляет собой электронную таблицу размещения файлов, использующую 32-разрядные записи. Кстати, аббревиатура расшифровывается как File Allocation Table.

NTFS — файловая система, представляющая собой определенную структуру: в начале диска сводную таблицу (или каталог) всех файлов, далее — собственно данные. Аббревиатура расшифровывается как New Technology File System.

На практике сегодня файловая система FAT32 чаще применяется на съемных носителях небольшого объема, NTFS — на системных дисках и для хранения файлов большого размера. Кластеры FAT32 больше, следовательно, дисковое пространство при хранении большого числа маленьких файлов используется нерационально. Большое количество программ, требующих наличия, например, множества библиотек, файлов шрифтов и других, в системе FAT32 отзовется медленной работой. NTFS обеспечивает быстрый доступ к небольшому файлу или части файла.

В общем и целом NTFS работает ощутимо медленнее FAT32, зато NTFS эффективнее при обращении к файлам больших размеров. Фрагментация никак не влияет на NTFS, тогда как FAT32 производительность заметно снизит (особенно это касается работы с каталогами средних размеров).

4. **Основные функции работы с файлами (ОС).**

HANDLE CreateFile(LPCTSTR lpFileName, DWORD dwDesiredAccess, DWORD dwShareMode, LPSECURITY_ATTRIBUTES lpSecurityAttributes, DWORD dwCreationDisposition, DWORD dwFlagsAndAttributes, HANDLE hTemplateFile)

возвращаемое значение - дескриптор файла или INVALID_HANDLE_VALUE в случае ошибки
lpFileName - указатель на строку с именем файла
dwDesiredAccess - тип доступа к файлу (чтение, запись)
dwShareMode - совместное использование
lpSecurityAttributes - структура с атрибутами безопасности
dwCreationDisposition - создать, перезаписать, открыть, etc.
dwFlagsAndAttributes - установка флагов и атрибутов
hTemplateFile - дескриптор шаблона файла для копирования его атрибутов

BOOL CloseHandle(HANDLE hObject)

http://msdn.microsoft.com/en-us/library/ms724211%28v=vs.85%29.aspx

BOOL ReadFile(HANDLE hFile, LPVOID lpBuffer, DWORD nNumberOfBytesToRead, LPDWORD lpNumberOfBytesRead, LPOVERLAPPED lpOverlapped)

BOOL WriteFile(HANDLE hFile, LPCVOID lpBuffer, DWORD nNumberOfBytesToWrite, LPDWORD lpNumberOfBytesWritten, LPOVERLAPPED lpOverlapped)

BOOL DeleteFile (LPCTSTR lpFileName)

BOOL CopyFile (LPCTSTR lpExistingFileName, LPCTSTR lpNewFileName, BOOL fFailIfExists)

BOOL CreateHardLink(LPCTSTR lpFileName, LPCTSTR lpExistingFileName, LPSECURITY_ATTRIBUTES lpSecurityAttributes)

BOOL MoveFile (LPCTSTR lpExistingFileName, LPCTSTR lpNewFileName)

переименовывает или перемещает файл
завершится ошибкой, если новый файл уже существует

BOOL MoveFileEx (LPCTSTR lpExistingFileName, LPCTSTR lpNewFileName, DWORD dwFlags)

Ex for extended
dwFlags задаёт различные опции

стр. 78 и 54 у Харта
ИМНО: вообще функций намного больше (см. MSDN Library http://msdn.microsoft.com/en-us/library/aa364232%28v=vs.85%29.aspx), однако их нет у Харта, и вряд ли нам необходимо их упоминать (upd.: не заметил в вопросе слово "основные").
Если кто не знал: в прототипах повсюду трансильванская ересь, т.е. префиксы переменных обозначают их тип.

5. **Способы копирования файлов (не меньше 3-х).**
   - С использованием библиотеки C
   - С использованием средств Windows
   - С использованием вспомогательных функций Windows –copyfile

6. **Функции управления каталогами.**

BOOL CreateDirectory (LPCTSTR lpPathName, LPSECURITY_ATTRIBUTES lpSecurityAttributes)

BOOL RemoveDirectory (LPCTSTR lpPathName)

BOOL SetCurrentDirectory (LPCTSTR lpPathName)
DWORD GetCurrentDirectory (DWORD cchCurDir, LPCTSTR lpCurDir)

cchCurDir - размер буфера для имени каталога (в символах, а не байтах)
lpCurDir - указатель на буфер
возвращаемое значение - длина строки, содержащей путь доступа к текущему каталогу, или требуемый размер буфера, если буфер недостаточно велик; в случае ошибки - ноль

Функции MoveFile и MoveFileEx, описанные в 4 вопросе, так же применимы и к директориям.
стр. 81 у Харта
IMHO: прототипы функций не нужны, хватит и просто их имён, но я на всякий случай их привёл.

7. Поясните понятие «реестр».

Реестр - это централизованная иерархическая база данных, хранящая информацию о параметрах конфигурации операционной системы и установленных приложений. Доступ к реестру осуществляется через разделы, или ключи, реестра, играющие ту же роль что и каталоги в файловой системе. Раздел может содержать подразделы или пары “имя-значение”, в которых между именем и значением существует примерно та же взаимосвязь, что и между именами файлов и их содержимым. (страница 112 у Харта).

8. Как получить структуру памяти системы?

VOID GetSystemInfo(LPSYSTEM_INFO lpSystemInfo) - параметром является адрес структуры LPSYSTEM_INFO, в которой содержится информация относительно размера системной стрницы, а также адресах физической памяти, доступных для приложений. (Харт, страница 157)

9. Основные функции для работы с пулами памяти, называемыми куча(heaps).

HANDLE GetProcessHeap (void) - возвращаемое значение - дескриптора кучи процесса, в случае неуспеха - NULL

HANDLE HeapCreate (DWORD flOptions, SIZE_T dwInitialSize, SIZE_T dwMaximumSize) возвращает дескриптор кучи или NULL в случае неуспеха. SIZE_T - 32-битное или 64-битное число без знака, в зависимости от флагов компилятора. flOptions объединяет 2 флага - HEAP_NO_SERIALIZE и HEAP_GENERATE_EXCEPTIONS. Если dwMaximumSize равен 0, то куча будет расти, иначе этот парметр определяет максимальный размер кучи. dwInitialSize соотвественно размер кучи при инициализации.

BOOL HeapDestroy (HANDLE hHeap) - уничтожение объекта кучи. Параметр - указатель на кучу. (Харт - страница 158)

10. При установлении размера кучи параметром dwInitialSize до какого значения ОС округляет его величину?

Начальный размер кучи, устанавливаемый параметром dwInitialSize (который может быть нулевым), всегда округляется до величины, кратной размеру страницы, и определяет объём физической памяти (в файле подкачки), которой передаётся в распоряжение кучи первоначально (для последующего распределения памяти по запросам), а не в ответ на запросы распределения (allocation) памяти из кучи. (Харт, стр. 160)

11. Как управляется память кучи?

Для получения блока памяти из кучи следует указать дескриптор области памяти кучи, размер блока некоторые флаги.
LPVOID HeapAlloc (HANDLE hHeap, DWORD dwFlags, SIZE_T dwBytes) - возвращает указатель на распределённый блок памяти или NULL в случае неуспеха. hHeap - дескриптор кучи, из которой должен быть распредеён блок памяти. dwFlags - флаги HEAP_GENERATE_EXCEPTIONS,

HEAP_NO_SERIALIZE и HEAP_ZERO_MEMORY. dwBytes - размер блока памяти, которые будет распределён.

Для освобождения блока памяти, распределённого из кучи используется функция:
BOOL HeapFree (HANDLE hHeap, DWORD dwFlags, LPVOID lpMem). lpMem - указатель, возвращаемый функцией HeapAlloc.

Для повторного распределения блоков памяти с целью изменения их размера используется:
LPVOID HeapreAlloc (HANDLE hHeap, Dword dwFlags, LPVOID lpMem, SIZE_T dwBytes) - возвращает указатель на перераспределённый блок памяти или в случае неуспеха вызывается исключение или NULL (в зависимости от значения соответствующего флага).

DWORD HeapSize (HANDLE hHeap, DWORD dwFlags, LPVOID lpMem) - возвращает размер блока, в случае неуспеха - 0. (Харт страницы 162-164)

12. Понятие отображения файлов и получаемые преимущества.

Динамическая память, распределенная в кучах должны физически размещаться в файле подкачки**.** Управление перемещением страниц между физ. памятью и файлом подкачки, а также отображением файла подкачки на виртуальное адресное пространство процесса осуществляется ОС средствами управления памятью. По завершении выполнения процесса физ. пространство в этом файле освобождается. Те же функции Windows, которые обеспечивают отображение файла подкачки, позволяют отображать и обычные файлы. Отображение файлов дает следующие преимущества:

1)Отпадает необходимость в выполнении операций непосредственного файлового ввода\вывода

2)Структуры данных, созданные в памяти, будут сохраняться в файле для последующего использования этой же или другими программами.

3)Становится возможным применение удобных и эффективных алгоритмов ориентированных на работу с файлами “в памяти” (сортировка, деревья поиска, обработка строк и .тп.), которые позволяют обрабатывать хранящиеся в файлах данные даже в тех случаях, когда размеры файлов значительно превышают доступный объем физической памяти. при больших размерах файлов особенности организации страничного обмена могут оказывать заметное влияние на производительность.

4)В некоторых случаях значительно повышается эффективность обработки файлов

5)Исчезает обходимость в управлении буферами и манипулирования содержащимися в них данными файлов. Всю эту работу выполняет ОС.

6)Обеспечивается возможность разделения памяти несколькими параллельно выполняющимися процессами за счет отображения на их виртуальные адресные пространства одного и того е обычного файла или файла подкачки (разделение файла несколькими процессами является одной из основных причин использования объекта отображения файла подкачки)
7)Отпадает необходимость в расходовании излишнего пространства файла подкачки.

13. **Объект отображения файла и функция получения его дескриптора**.

Объект отображения файла создается для открытого файла, который затем отображается на вирт. адресное пространство процесса. Объектам отображения можно присваивать имена, по которым к ним смогут обращаться другие процессы, разделяющие память совместно с данным процессом. Кроме того, объекты отображения файлов имеют параметры размера и атрибуты защиты.
Функция **CreateFileMapping** возвращает дескриптор объекта отображения файла.

HANDLE CreateFileMapping(HANDLE hFile, LPSECURITY_ATTRIBUTES lpsa, DWORD dwProtect, DWORD dwMaximumSizeHigh, DWORD dwMaximumSizeLow, LPCTSTR lpMapName)

Возвращаемое значение: в случае успешного выполнения — дескриптор объекта отображения файла, иначе — NULL.

hFile - дескриптов открытого файла, LPSECURITY_ATRIBUTES - позволяет указать атрибуты защиты объекта отображения, dwProtect - определяет с помощью флагов возможности доступа к представлению файла при его отображении (PAGE_READONLY, PAGE_READWRITE,PAGE_WRITECOPY),dwMaximumSizeHigh и dwMaximumSizeLow - старшая и младшая 32-итовые части значения максимального размера объекта отображения файла.

lpMapName - указатель на строку,содержащую имя объекта отображения, которые другие процессы могут использовать для разделения объекта;

14. Поясните понятие «базовый указатель».

Базовый указатель используется для обозначения начала стекового кадра функции или области стека, управляемой этой функцией. Локальные переменные хранятся под базовым указателем и над стековым указателем. Каждая функцию начинается с сохранения старого базового указателя и объявления нового. Заканчивается восстановлением старого базового указателя.