

- 1) Сразу после обработки прерывания режимом процесса является
 - * режим ядра;
 - режим задачи;
 - режим ядра или режим задачи;
 - режим готовности
- 2) Подсистема управления процессами
 - является частично машиннозависимой;
 - распознает системные вызовы `fork()`, `exit()` и пр.;
 - * функционирует на уровне ядра;
 - функционирует на уровне аппаратуры.
- 3) Проверка поступления сигналов выполняется
 - при переходе из режима задачи в режим ядра;
 - * при переходе в режим приостанова, режим ядра и возврата из режима готовности
 - при переходе из режима ядра в режим задачи;
 - при переходе из режима заблокировано в режим готовности.
- 4) После открытия процессом двух файлов и использования системного вызова `dup()` с параметром, являющимся номером дескриптора одного из открытых ранее файлов, число записей в таблице открытых файлов равно
 - 3;
 - * 6;
 - 4;
 - 7;некоторому другому значению.
- 5) Среди указанных ниже операций единственной операцией, выполняемой средствами DOS является
 - * вывод строки символов;
 - перемещение курсора в заданную точку;
 - чтение положения курсора;
 - задание новой активной страницы;
 - прокрутка активной страницы;
 - чтение символа из текущей позиции курсора и его атрибута;
 - запись символа и его атрибута в текущую позицию курсора;
 - запись символа без изменения атрибута в текущую позицию курсора;
 - задание видеорежима.
- 6) Компонента ОС "demand paging"
 - выполняет замещение страницы, путём выгрузки её во внешнюю память по некоторому алгоритму;
 - предотвращает возникновение ситуации пробуксовки (`trashing`);
 - занимается своппингом процессов;
 - * выделяет страницу ОП, перемещая в неё копию страницы из внешней памяти;
 - выполняет откачку страниц во внешнюю память;
 - оперирует с понятием рабочего набора.
- 7) Критическая секция
 - * предотвращает использование несколькими процессами критичных данных;
 - обеспечивает целостность данных пользователя;
 - создается в процессе обработки прерываний;
 - служит для предотвращения использования несколькими пользователями критичных данных;
 - доступна только процессам, созданным ОС.
- 8) При сегментной организации памяти
 - виртуальная память процесса не может превышать размера оперативной памяти;
 - * каждый блок виртуальной памяти может иметь произвольный размер;
 - имеет место двухуровневая трансляция виртуального адреса в физический;
 - каждый блок виртуальной памяти имеет одинаковый размер;
 - сумма виртуальных пространств всех процессов не может превышать размера оперативной памяти.
- 9) Таблица векторов прерываний – это

- * область, где хранятся адреса программ обработки прерываний;
 - область старших адресов оперативной памяти;
 - часть сегмента данных;
 - область префикса программного сегмента;
 - область, где хранятся параметры программ обработки прерываний.
- 10) Функция 4Ah прерывания 21h служит для
- выделения блока памяти;
 - * изменения размера памяти, отведенного программе;
 - освобождения блока памяти;
 - для запуска одной программы из другой.
- 11) Для обмена двух процессов данными через программный канал минимальный набор системных вызовов составляет
- ```
open(), read(), write(), close();
pipe();
* pipe(), read(), write();
pipe(), dup(), read(), write();
dup(), read(), write().
```
- 12) Выполнение P-операции P(S) над классическим семафором
- ведет к уменьшению значения аргумента на 1;
  - равносильно операции S=S+1;
  - \* неделимая операция, уменьшающая положительное значение аргумента на 1
  - ведет к увеличению значения аргумента на 1;
  - равносильно операции S=S-1;
  - производится над любым целочисленным аргументом;
  - подразумевает нечто иное.
- 13) В текстовом режиме каждой позиции экрана соответствует в памяти
- 1 бит;
  - 2 бита;
  - 4 бита;
  - 1 байт;
  - 4 байта;
  - \* 2 байта.
- 14) Результатом нормального выполнения системного вызова wait() является нулевой код завершения;
- \* идентификатор завершившегося процесса;
  - идентификатор ожидаемого процесса;
  - статус завершения.
- 15) Информация о первом введенном символе записывается в буфер клавиатуры по адресу 0040:001E, о втором - по адресу 0040:0020, . . . , о пятнадцатом - по адресу 0040:003A, о шестнадцатом - по адресу
- ```
0040:003B;
* 0040:001E;
0040:001A;
0040:003C;
0040:001C
```
- 16) Механизм очередей сообщений служит для обмена сообщениями родственных процессов;
- процессов, не связанных отношением родства;
 - процессов, имеющих общего предка;
 - * любых процессов;
 - процессов, имеющих общего владельца.
- 17) Таблица описателей файла содержит
- * сведения о типе файла, правах доступа к нему, размере файла, а также счетчик ссылок на запись таблицы;
 - информацию о режиме открытия файла, указатель чтения/записи и число ссылок на запись таблицы;
 - идентификатор (дескриптор) файла;
 - номера блоков, составляющих файл;
 - номер процесса.
- 18) Выполнение V-операции V(S) над классическим семафором

- ведет к уменьшению значения аргумента на 1;
равносильно операции $S=S+1$;
- ведет к увеличению значения аргумента на 1 для любого целочисленного аргумента;
равносильно операции $S=S-1$;
- производится над любым целочисленным аргументом;
- * неделимая операция, увеличивающая неотрицательное значение аргумента на 1
- 19) Минимальный объём динамически запрашиваемой памяти равен
- * 16 байт;
 - 1 байт;
 - 128 байт;
 - 256 байт.
- 20) После открытия процессом двух файлов и создания канала число записей в таблице открытых файлов равно
- * 7;
 - 6;
 - 4;
 - 3;
- некоторому другому значению.
- 21) Из режима задачи возможен переход
- в режим ядра и режим приостановки;
 - * в режим ядра;
 - в режим ядра и режим готовности;
 - в любой другой режим.
- 22) Процесс - это
- * единица работы, управления и потребления ресурсов;
 - последовательность команд программы;
 - объект, созданный интерпретатором команд;
 - объект, созданный процессом ядра;
 - нечто иное.
- 23) Связывание обычных файлов системным вызовом `link()` может быть выполнено
- только процессом, созданным интерпретатором `shell`;
 - * любым процессом;
 - только процессом, принадлежащим суперпользователю;
 - только процессом, созданным интерпретатором `shell`, либо ядром.
- 24) Контекст процесса - это
- адресное пространство процесса;
 - * состояние процесса в любой момент времени;
 - образ процесса в любой момент времени;
 - процедурный сегмент, сегмент данных и сегмент стека.
- 25) Своппинг - это
- * перемещение процессов из оперативной памяти на диск и ввод их по мере необходимости обратно;
 - управление процессами в оперативной памяти;
 - управление процессами во внешней памяти;
 - управление внешней и оперативной памятью;
 - нечто иное.
- 26) После открытия первым процессом файлов `file1, file2, file3, file4`, вторым процессом - файлов `file1, file2`, третьим - файла `file2` и последующим закрытием вторым процессом всех открытых ранее им файлов, максимальное значение счетчика в таблице описателей файлов равно
- 1;
 - 3;
 - * 2;
 - 4;
 - 5;
 - 7.
- 27) Кэш-память
- ускоряет работу с байториентированными устройствами;

- * ускоряет работу с блокориентированными устройствами;
ускоряет работу с любыми устройствами;
является аппаратно реализованным механизмом.
- 28) Интерпретатор команд shell
 - является процессом, выполняющимся в режиме ядра;
 - для выполнения любой команды создает новый процесс;
 - осуществляет ввод командной строки, не пользуясь услугами ядра;
 - не пользуется системными вызовами;
 - * является процессом, выполняющимся в режиме задачи.
- 29) Смена контекста выполняется
 - при переходе из режима ядра в режим задачи;
 - при переходе из режима ядра в заблокированное состояние;
 - при переходе из режима задачи в режим ядра;
 - при переходе из заблокированного состояния в режим готовности;
 - * при любой смене режима.
- 30) ОС UNIX – является мобильной ОС, поскольку
 - позволяет легко работать в сети;
 - обладает средствами восстановления после возникновения сбоев в системе;
 - * допускает перенос в текстах на различные платформы;
 - построена по архитектуре "Клиент-сервер";
- 31) В системной фазе могут выполняться
 - только процессы ядра ОС UNIX;
 - * любые процессы ОС UNIX;
 - только пользовательские процессы;
 - только процессы интерпретатора команд.
- 32) В режиме ядра
 - выполняются только процессы, созданные ОС;
 - * выполняется код ядра ОС;
 - процесс не может быть прерван;
 - недоступен аппарат системных вызовов;
 - нечто иное.
- 33) В случае нулевого первого аргумента системного вызова Signal
 - процесс игнорирует все последующие получения сигнала;
 - * по получению сигнала процесс завершается;
 - сигнал посыпается всем процессам, входящим с данным процессом в одну группу;
 - сигнал посыпается всем процессам, у которых код иден-ра пользователя совпадает с тем, под которым выполняется процесс;
 - процесс немедленно завершается.
- 34) Удаление связи с файлом после его открытия
 - приводит к аварийному завершению процесса;
 - * оставляет возможность для работы с ним;
 - приводит к автоматическому закрытию файла;
 - приводит к блокировке этого файла.
- 35) После открытия процессором 2-х файлов и создания потомка по системному вызову fork() общее число записей в таблице файлов равно
 - * 2;
 - 5;
 - 8;
 - 10;
 - некоторому другому значению.
- 36) Процесс обязательно включает
 - секции текста, стека, данных;
 - * секции текста, стека;
 - секцию текста;
 - секции текста, данных;
 - секции стека, данных.
- 37) Чтение потока символов с терминала интерпретатором shell
 - * выполняется отдельным процессом в режиме задачи;
 - выполняется процессом-интерпретатором в режиме ядра;

- выполняется процессом-интерпретатором в режиме задачи;
выполняется отдельным процессом в режиме ядра.
- 38) Прерывание 9h генерируется
любым перефрийным устройствам;
центральным процессором;
программой пользователя;
* клавиатурой;
операционной системой.
- 39) Прерывание - это
сигнал, прерывающий работу внешнего устройства и сообщающий о
необходимости выполнить некоторую работу;
* сигнал, прерывающий работу центрального процессора и сообщающий о
необходимости выполнить некоторую работу;
сигнал, генерируемый аппаратурой;
реакция на выполнение команды int.
- 40) Критическая секция служит для
обеспечения целостности данных пользователя;
реализации механизма событий (семафора);
учета процессов;
реализации системных вызовов;
* обеспечения целостности данных ядра.
- 50) Стока среды в блоке параметров при динамическом вызове одной
программы из другой содержит
* спецификации, используемые в файле config.sys;
сведения из PSP
команды, используемые в файле autoexec.bat;
параметры функции 4Bh;
нечто иное.
- 51) Промежуточная таблица областей процессов
* обеспечивает совместное использование областей независимыми
процессами;
обеспечивает ссылки к таблице процессов;
содержит управляющую информацию о состоянии процесса;
указывает, где размещены сегменты текста, стека и данных.
- 52) Среди указанных ниже операций единственной операцией, выполняемой в
графическом режиме является
вывод строки символов;
перемещение курсора в заданную точку;
чтение положения курсора;
задание новой активной страницы;
прокрутка активной страницы;
чтение символа из текущей позиции курсора и его атрибута;
запись символа и его атрибута в текущую позицию курсора;
запись символа без изменения атрибута в текущую позицию курсора;
* задание видеорежима.
- 53) Метод FCB
вызывается через прерывание BIOS;
работает с отдельными секторами;
использует идентификационный номер файла;
* работает только с файлами текущей директории;
служит для работы с любыми перефрийными устройствами.
- 54) При сегментно-страничной организации памяти
виртуальная память процесса не может превышать размера оперативной
памяти;
каждый блок виртуальной памяти может иметь произвольный размер;
* имеет место двухуровневая трансляция виртуального адреса в
физический;
каждый блок виртуальной памяти имеет одинаковый размер;
сумма виртуальных пространств всех процессов не может превышать
размера оперативной памяти.
- 55) Анализ скан-кода выполняется

- прерыванием 16h;
прерыванием 21h;
любым прерыванием;
* прерыванием 9h.
- 56) Длина таблицы прерываний
512 б;
* 1024 б;
256 б;
4096 б.
- 57) При запросе последнего блока из списка в суперблоке (*s_free*)
в данный блок переписывается содержимое массива *s_free*, выполняется сортировка и блок включается в цепочку;
* содержимое этого блока переписывается в массив *s_free*;
просматривается таблица блоков данных для поиска свободных;
в данный блок переписывается содержимое массива *s_free* и он включается в цепочку;
содержимое этого блока переписывается в массив *s_free* и выполняется сортировка.
- 58) Расширенный код имеет длину
4 байта;
8 байт;
1 байт;
* 2 байта;
переменная длина.
- 60) Исходный файл содержит последовательно 128 значений "a", "b", "c", "d", "e", "f" и т. д. Программа дважды открывает указанный файл и читает с использованием первого дескриптора две записи по 128 байт, а затем с использованием второго дескриптора три записи по 128 байт. Последним прочитанным символом из файла является
"a";
"b";
* "c";
"d";
"e";
"f".
- 61) Образ процесса состоит из
процедурного сегмента, сегмента данных и сегмента стека;
* процедурного сегмента и сегмента стека;
процедурного сегмента и сегмента данных;
процедурного сегмента, сегмента данных и сегмента стека и У-области;
процедурного сегмента, сегмента стека и У-области.
- 62) При страничной организации памяти
виртуальная память процесса не может превышать размера оперативной памяти;
каждый блок виртуальной памяти может иметь произвольный размер;
имеет место двухуровневая трансляция виртуального адреса в физический;
* каждый блок виртуальной памяти имеет одинаковый размер;
сумма виртуальных пространств всех процессов не может превышать размера оперативной памяти.
- 63) Длина префикса программного сегмента (PSP) составляет
* 100h байт;
100 байт;
128 байт;
128h байт.
- 64) Прерывания 13h, 25h, 26h используются
в методе FCB;
в методе дескриптора файла;
* для работы с отдельными секторами;
при работе с физической нумерацией диска;
при работе с логической нумерацией диска.

- 65) Обработка сигналов выполняется
- * при переходе из режима ядра в режим задачи;
 - при переходе в режим приостанова;
 - при выходе из режима приостанова;
 - при переходе из режима ядра в режим задачи и обратно;
 - при переходе из режима задачи в режим ядра.
- 66) Системный процесс - stealer
- выполняет замещение страницы, путём выгрузки её во внешнюю память по некоторому алгоритму;
 - предотвращает возникновение ситуации пробуксовки (trashing);
 - занимается своппингом процессов;
 - выделяет страницу ОП, перемещая в неё копию страницы из внешней памяти;
 - * выполняет откачуку страниц во внешнюю память;
 - оперирует с понятием рабочего набора.
- 67) Приоритет процесса является
- * функцией от времени с момента последней загрузки в ОП;
 - функцией от времени с момента предоставления процессора;
 - функцией от времени использования процессора;
 - функцией от времени нахождения в системной фазе;
 - функцией от времени нахождения в пользовательской фазе.
- 68) Блокировка описателя файла в алгоритме Link
- * порождает туниковые ситуации;
 - предотвращает туниковые ситуации;
 - создает условия для конкуренции процессов;
 - ведет к возникновению некорректных ситуаций.
- 69) Функция 35h прерывания 21h служит для
- защиты таблицы векторов прерываний;
 - установки вектора прерываний;
 - определение критической секции программы;
 - * чтение вектора прерывания;
 - задание режима обработки прерываний.
- 70) Принцип локальности ссылок
- выполняет замещение страницы, путём выгрузки её во внешнюю память по некоторому алгоритму;
 - предотвращает возникновение ситуации пробуксовки (trashing);
 - занимается своппингом процессов;
 - выделяет страницу ОП, перемещая в неё копию страницы из внешней памяти;
 - выполняет откачуку страниц во внешнюю память;
 - * оперирует с понятием рабочего набора.
- 71) Код ASCII имеет длину
- 4 байта;
 - 8 байт;
 - * 1 байт;
 - 2 байта;
 - переменная длина.
- 72) Исходный файл содержит последовательно 128 байт значений "a", "b", "c", "d", "e", "f" и т.д. Программа создает новый процесс в рамках порожденного процесса дважды открывает указанный файл и читает с использованием первого дескриптора две записи по 64 байта, а затем с использованием второго дескриптора три записи по 64 байта. Последним прочитанным символом из файла является
- "f";
 - "e";
 - "d";
 - "c";
 - * "b";
 - "a".

Массовые операции над семафорами в UNIX (набор семафоров) введены с целью

- * расширения понятия классического семафора;
- * увеличения числа выполняемых операций над семафором;
- * уменьшить вероятность возникновения тупиковых ситуаций;
- * увеличения числа процессов, одновременно использующих семафоры.

Процесс - это

- объект, созданный в результате выполнения системного вызова exec();
- объект, созданный интерпретатором команд;
- объект, созданный процессом ядра;
- * объект, созданный в результате выполнения системного вызова fork();
- нечто иное.

Для передачи командной строки в динамически вызываемую программу используется

- поле из PSP;
- * поле блока параметров;
- строка с полным именем запускаемой программы;
- нечто иное.

Своппингу

- менее подвергаются процессы с большим приоритетом.
- более подвержены процессы, находящиеся в системной фазе;
- * не подвергаются процессы, созданные в режиме ядра;
- более подвержены процессы, находящиеся в пользовательской фазе;

Один элемент FAT-таблицы соответствует

- * одному кластеру;
- одному блоку;
- одному элементу оглавления;
- одному файлу;
- одному сектору.

При вызове функции printf() из функции main() число записей активации составляет:

- 1
- 2
- * 3

Два процесса взаимодействуют через программный канал. Когда один запишет в канал n записей по 128 байт, сколько прочитает другой?

- n записей по 128 байт
- 128 байт
- * сколько угодно

Системный вызов mount служит для связывания :

- файлов
- папок
- * файловых систем

Таблица областей процессов

обеспечивает совместное использование областей независимыми процессами;

- обеспечивает ссылки к таблице процессов;
- содержит управляющую информацию о состоянии процесса;
- * указывает, где размещены сегменты текста, стека и данных.

Таблица процессов

обеспечивает совместное использование областей независимыми процессами;

- обеспечивает ссылки к таблице процессов;
- * содержит управляющую информацию о состоянии процесса;
- указывает, где размещены сегменты текста, стека и данных.

Таблица файлов содержит

сведения о типе файла, правах доступа к нему, размере файла, а также счетчик ссылок на запись таблицы;

- * информацию о режиме открытия файла, указатель чтения/записи и число ссылок на запись таблицы;
- идентификатор (дескриптор) файла;
- номера блоков, составляющих файл;
- номер процесса.

Таблица открытых файлов содержит

сведения о типе файла, правах доступа к нему, размере файла, а также счетчик ссылок на запись таблицы;

информацию о режиме открытия файла, указатель чтения/записи и число ссылок на запись таблицы;

* идентификатор (дескриптор) файла;

номера блоков, составляющих файл;

номер процесса.

Связывание каталогов файловой системы системным вызовом link() может быть выполнено

только процессом, созданным интерпретатором shell;
любым процессом;

* только процессом, принадлежащим суперпользователю;

только процессом, созданным интерпретатором shell, либо ядром.

Функция 25h прерывания 21h служит для

защиты таблицы векторов прерываний;

* установки вектора прерываний;

определение критической секции программы;

чтение вектора прерывания;

задание режима обработки прерываний.

Каждому введенному символу в буфере клавиатуры соответствует

4 байта;

8 байт;

1 байт;

* 2 байта;

переменная длина.

Неравенство значений по адресам 40:1A и 40:1C

свидетельствует об отсутствии символов в буфере клавиатуры

* свидетельствует о наличии символа в буфере клавиатуры

свидетельствует о записи символа в буфер клавиатуры

свидетельствует о переполнении буфера клавиатуры

Ввод строки символов на Ассемблере осуществляется отдельной функцией прерывания

* 21h

13h

9h

10h

Область DTA в PSP содержит

параметры программы

* командную строку программы

иное

Страницочный механизм работы с памятью используется

для управления режимом вывода информации в файл

во всех операциях с памятью

* для управления графическим режимом вывода информации

Метод дескриптора файла

* использует идентификационный номер

использует условное обозначение

использует сокращение файла

нечто иное

Мультипрограммирование - это

* управление последовательностью выполнения процессов и последовательностью выполнения свопинга

управление последовательностью выполнения процессов

управление последовательностью выполнения свопинга

Суперблок, в частности, содержит информацию о

количестве занятых блоков файловой системы

* количество свободных блоков файловой системы

блоках файловой системы

При освобождении блока в случае заполненности списка в суперблоке

в данный блок переписывается содержимое массива `s_free`, выполняется сортировка и блок включается в цепочку;
содержимое этого блока переписывается в массив `s_free`;
просматривается таблица блоков данных для поиска свободных;
* в данный блок переписывается содержимое массива `s_free` и он включается в цепочку;
содержимое этого блока переписывается в массив `s_free` и выполняется сортировка.

После открытия первым процессом файлов `file1`, `file2`, `file3`, `file4`, вторым - файлов `file1`, `file2`, третьим - файла `file2` число записей в таблице файлов - равно

```
1  
2  
4  
* 7
```

После открытия первым процессом файлов `file1`, `file2`, `file3`, `file4`, вторым - файлов `file1`, `file2`, третьим - файла `file2` число записей в таблице описателей файлов - равно

```
1  
2  
* 4  
7
```

После открытия процессом 2-х файлов число записей в таблице открытых файлов - равно

```
1  
2  
3  
4  
* 5
```

Исходный файл содержит последовательно 128 значений 'a', 'b', 'c',.. и т.д. После открытия файла и получения копии дескриптора файла системным вызовом `dup()` с использованием оригинального дескриптора файла выполнено чтение 2-х записей по 128 байт, а с использованием копии - 3 чтения.

Последним прочитанным символом - будет

```
a  
b  
c  
d  
* e  
f
```

Исходный файл содержит последовательно 128 значений 'a', 'b', 'c',.. и т.д. После открытия файла и получения копии дескриптора файла системным вызовом `dup()` с использованием оригинального дескриптора файла выполнено чтение 2-х записей по 64 байта, а с использованием копии дескриптора файла - прочитано 3 записи по 64 байт . Последним прочитанным символом - будет

```
a  
b  
* c  
d  
e  
f
```

Системный вызов `exec()` служит для

- * запуска программы из любого процесса
- запуска программы из родительского процесса
- запуска программы из дочернего процесса
- запуска нового процесса

Завершение процесса выполняется в программе

- * системным вызовом `exit()`, либо при завершении последнего оператора главной функции `main()`
- системным вызовом `exit()`

при завершении последнего оператора главной функции main()
другими вызовами

Системный вызов kill() служит для
уничтожения любого процесса
уничтожения дочернего процесса
* посылки сигнала любому процессу или группе
для завершения процесса
задания режима обработки сигнала

Системный вызов signal() служит для
уничтожения любого процесса
уничтожения дочернего процесса
посылки сигнала любому процессу или группе
для завершения процесса
* задания режима обработки сигнала

Выполнение системного вызова

- * связано с переходом из пользовательской фазы в системную
- связано с переходом из системной фазы в пользовательскую
- связано с другими переходами

В конвейере команд

- команды(программы) выполняются синхронно к друг другу
- * команды(программы) выполняются асинхронно к друг другу
- команды(программы) выполняются последовательно

Системный вызов mount() может быть выдан

- только процессом, созданным интерпретатором shell;
- любым процессом;
- * только процессом, принадлежащим суперпользователю;
- только процессом, созданным интерпретатором shell, либо ядром.

Набор программных средств IPC является средством взаимодействия

- * любых процессов
- родственных процессов
- независимых процессов

Произвольный алгоритм подкачки

- выполняет замещение страницы, путём выгрузки её во внешнюю память по некоторому алгоритму;
- * предотвращает возникновение ситуации пробуксовки (trashing);
занимается своппингом процессов;
выделяет страницу ОП, перемещая в неё копию страницы из внешней памяти;
- выполняет откачку страниц во внешнюю память;
оперирует с понятием рабочего набора.

Системный вызов msgget позволяет

- получить дескриптор существующей очереди
- образовать новую очередь сообщений
- * образовать новую очередь сообщений и получить дескриптор существующей очереди

Системный вызов shmget позволяет

- * образовать сегмент разделяемой памяти или найти сегмент разделяемой памяти по ключу
- образовать сегмент разделяемой памяти
- найти сегмент разделяемой памяти по ключу

Системные вызовы, связанные со временем

- оперируют с глобальными переменными, определенными на уровне пользователя
- * оперируют с глобальными переменными, определенными на уровне ядра
- оперируют с локальными переменными, определенными на уровне ядра
- оперируют с локальными переменными, определенными на уровне пользователя

Ядро ОС UNIX

- * выполняет диспетчерские функции
- выполняет только контролирующие функции
- используется для других целей

Реентерабельная программа
не допускает совместное свое использование
* допускает совместное свое использование

При нехватке ОП кандидатом на выгрузку является процесс
* находящийся в системной фазе
занимающий больший объем памяти
занимающий меньший объем памяти
выполняющий ввод или вывод

Подсистема управления файлами функционирует на уровне
* ядра
аппаратуры
другое

После открытия процессом файла и создания канала число записей в таблице открытых файлов процесса равно
2
3
4
5
* 6

После открытия первым процессом файлов file1, file2, file3, file4 вторым - файлов file1, file2, третьим - файла file2 и последующим открытием вторым процессом file3 , максимальное значение счётчика в таблице описателей файлов - равно
2
* 3
4
5
6

Процесс открывает существующий файл длиной 500 байт в режиме `o_wronly` и записывает в него 10 байт. какова длина после окончания записи ?
10
* 500
510

Процесс открывает существующий файл длиной 500 байт в режиме `o_wronly|o_append` и записывает в него 10 байт. какова длина после окончания записи ?
10
500
* 510

Процесс открывает существующий файл длиной 500 байт в режиме `o_wronly` и записывает в него 10 байт. а потом еще 10 раз по 50. какова длина после окончания записи ?
10
500
* 510

при условии, что `filedes` - дескриптор файла, системный вызов `lseek(filedes, (off_t)0, seek_end);` позволит определить - не пуст ли файл показать, сколько в нем записей
* определить длину файла
определить количество `off_t`

системный вызов `unlink("/temp/usedfile");` выполненный сразу после создания файла очищает файл
* уничтожает файл
убирает ссылку на файл

каковы права доступа, когда владелец может выполнять все операции, а все остальные - только чтение ?
* 0744
1755
0766

каковы права доступа, когда владелец может выполнять чтение и запись, а все остальные - только чтение ?

- * 0644
- 0755
- 0744

системный вызов alarm

- ставит процес в положение "тревога"

- * устанавливает интервал времени, через которое процессу будет послан сигнал

- посыпает сигнал процессу-потомку

ключ объекта ipc является

- случайным числом

- * уникальным для данной вычислительной системы

- числом, определенным заранее

номер семафоров (индекс) в наборе семафоров

- должен быть больше числа семафоров

- * должен быть меньше числа семафоров

- должен быть равен числу семафоров

в случае заполненности очереди сообщений и невозможности поместить в нее, процесс выдавший системный вызов msgsnd()

- переходит в режим ядра

- получает значение -1, говорящее о невозможности отправить сообщение

- * замораживается до появления возможности занести сообщение в очередь при использовании метода дейтограмм в отличие от модели tcp соединения не используется следующий этап клиентского процесса

- * подключение к сокету

- подключение к серверу

при загрузке exe программы на начало psp указывают регистры

- * ds,es

- ax,es

- bx,es

- es,ch

- все сегментные регистры

при загрузке com программы на начало psp указывают регистры

- ds,es

- ax,es

- bx,es

- es,ch

- * все сегментные регистры

регистры палитры

- * позволяют изменить цвета без изменений в видео-буфере

- позволяют изменить цвета с незначительными изменениями в видео-буфере

- не позволяют изменить цвета без изменений в видео-буфере

системный вызов exec

- применяется для запуска дочернего процесса

- применяется для заморозки процесса-отца

- * применяется для запуска процесса из любого процесса

при условии, что блок 8192 максимальный размер файла с использованием только прямой адресации

- 92 кб

- 128 кб

- * 96 кб

- 64 кб

- 8192 байта

системный вызов alarm -

- * процесс в режиме задач

- процесс в режиме ядра

- замораживает процесс

именованые каналы

- не являются файлами

- * полностью аналогичны файлам файловой системы

являются обычными файлами