

Тест 5

# Вопросы

## 1. При назначении описателя файла новому файлу

- a) используется списковая структура, связанная с массивом в суперблоке;
- b) **в случае выделения номера, являющегося последним в массиве s\_inode, он трактуется как указатель на блок с номером блока данных;**
- c) выделение каждого описателя файла связано с просмотром таблицы описателей файлов;
- d) в случае выделения номера, являющегося последним в массиве s\_inode, он выделяется файлу.

## 2. При запросе последнего блока из списка в суперблоке (s\_free)

- a) в данный блок переписывается содержимое массива s\_free, выполняется сортировка и блок включается в цепочку;
- b) содержимое этого блока переписывается в массив s\_free;
- c) просматривается таблица блоков данных для поиска свободных;
- d) в данный блок переписывается содержимое массива s\_free и он включается в цепочку;
- e) содержимое этого блока переписывается в массив s\_free и выполняется сортировка.

## 3. При освобождении блока в случае заполненности списка в суперблоке

- a) в данный блок переписывается содержимое массива s\_free, выполняется сортировка и блок включается в цепочку;
- b) содержимое этого блока переписывается в массив s\_free;
- c) просматривается таблица блоков данных для поиска свободных;
- d) в данный блок переписывается содержимое массива s\_free и он включается в цепочку;
- e) содержимое этого блока переписывается в массив s\_free и выполняется сортировка.

## 4. Магнитные диски являются

- a) **байтоориентированными устройствами;**
- b) блокоориентированными устройствами;
- c) байто- или блокоориентированными устройствами в зависимости от типа операции.

# Вопросы

## 5. Описатель файла содержит, в частности, информацию о

- a) типе файла, его размере и адресах блоков данных, составляющих файл;
- b) количестве блоков файловой системы, занятых файлом;
- c) типе файла, его размере и идентификаторе;
- d) типе файла, его размере и имени.

## 6. Таблица файлов содержит

- a) сведения о типе файла, правах доступа к нему, размере файла, а также числе ссылок на запись таблицы;
- b) информацию о режиме открытия файла, указателе чтения/записи и числе ссылок на запись таблицы;
- c) идентификатор (дескриптор) файла;
- d) номера блоков, составляющих файл;
- e) номер процесса, работающий с файлом.

## 7. Таблица открытых файлов процесса содержит

- a) сведения о типе файла, правах доступа к нему, размере файла, а также числе ссылок на запись таблицы;
- b) информацию о режиме открытия файла, указателе чтения/записи и числе ссылок на запись таблицы;
- c) идентификатор (дескриптор) файла;
- d) номера блоков, составляющих файл;
- e) номер процесса, работающий с файлом.

# Вопросы

## 8. Таблица описателей файлов содержит

- a) сведения о типе файла, правах доступа к нему, размере файла, а также числе ссылок на запись таблицы;
- b) информацию о режиме открытия файла, указателе чтения/записи и числе ссылок на запись таблицы;
- c) идентификатор (дескриптор) файла;
- d) номера блоков, составляющих файл;
- e) номер процесса, работающий с файлом.

## 9. После открытия первым процессом файлов file1, file2, file3, file4, вторым процессом - файлов file1, file2, file4, третьим - файла file2, и последующим закрытием первым процессом файла file3 число записей в таблице описателей файлов равно

- a) 3;
- b) 4;
- c) 7;
- d) 8.

## 10. После открытия первым процессом файлов file1, file2, file3, file4, вторым процессом - файлов file1, file2, третьим - файла file2, file3 и последующим закрытием вторым процессом файла file1, максимальное значение счетчика ссылок в таблице описателей файлов равно

- a) 1;
- b) 2;
- c) 3;
- d) 4;
- e) 5;
- f) 6.

## 11. После открытия процессом двух файлов и двукратного использования системного вызова dup() с параметром, являющимся номерами дескрипторов первого и второго открытых ранее файлов, число записей в таблице открытых файлов равно

- a) 3;
- b) 4;
- c) 5;
- d) 6;
- e) 7.

# Вопросы

**12. После открытия первым процессом файлов file1, file2, file3, file4, вторым процессом - файлов file1, file2, file4, третьим - файла file2, и последующим закрытием первым процессом файла file3 число записей в таблице файлов равно**

- |       |              |
|-------|--------------|
| a) 3; | c) <u>7;</u> |
| b) 4; | d) 8.        |

**13. После открытия процессом двух файлов в режиме чтения, и последующим открытием одного из ранее открытых файлов в режиме записи число записей в таблице открытых файлов процесса равно**

- |              |       |
|--------------|-------|
| a) 2;        | d) 5; |
| b) <u>3;</u> | e) 6; |
| c) 4;        | f) 7. |

**14. После открытия процессом двух файлов в режиме чтения, создания канала и последующим закрытием одного из ранее открытых файлов число записей в таблице открытых файлов равно**

- |              |       |
|--------------|-------|
| a) 2;        | d) 5; |
| b) <u>3;</u> | e) 6; |
| c) 4;        | f) 7. |

**15. После открытия процессом двух файлов в режиме записи, создания канала и последующим закрытием одного из ранее открытых файлов число записей в таблице открытых файлов равно**

- |              |       |
|--------------|-------|
| a) 2;        | d) 5; |
| b) <u>3;</u> | e) 6; |
| c) 4;        | f) 7. |

# Тест 6

# Вопросы

## 1. Суперблок помимо прочего содержит

- a) указатели на описатели файлов файловой системы;
- b) список свободных описателей файлов;
- c) указатель на таблицу описателей файлов;
- d) списковую структуру с номерами описателей файлов;
- e) счетчик числа используемых описателей файлов.

## 2. Суперблок

- a) является первым блоком файловой системы;
- b) расположен между загрузочным блоком и таблицей индексных дескрипторов;
- c) расположен между таблицей индексных дескрипторов и информационными блоками;
- d) расположен между меткой и загрузочным блоком

## 3. Каковы права доступа, при которых владелец может читать, писать и исполнять файл, а все прочие пользователи - только исполнять файл?

- a) 0666;
- b) 0644;
- c) 0611;
- d) **0711.**

## 4. Какая из перечисленных функций возвращает дескриптор файла

- a) Read(); -
- b) Lseek();
- c) Chmod();
- d) Pipe();

# Вопросы

**5. Что из перечисленного является библиотечной функцией**

- a) Printf();
- b) Open(); -
- c) Creat(); -
- d) Read(); -
- e) Lseek();

**6. Исходный файл содержит последовательно 128 значений "a", 128 значений "b", "c", "d", "e", "f" и т. д. Программа дважды открывает указанный файл и читает с использованием первого дескриптора три записи по 128 байт, а затем с использованием второго дескриптора три записи по 128 байт. Последним прочитанным символом из файла является**

- a) "a";
- b) "b";
- c) "c";
- d) "d";
- e) "e";
- f) "f";

**7. Исходный файл содержит последовательно по 128 значений "a", 128 значений "b", "c", "d", "e", "f" и т.д. После открытия файла и получения копии дескриптора файла по системному вызову dup() с использованием оригинального дескриптора файла выполнено чтение трех записей по 128 байт, а с использованием копии дескриптора файла выполнено чтение трех записей по 128 байт. Последним прочитанным из файла символом является**

- a) "a";
- b) "b";
- c) "c";
- d) "d";
- e) "e";
- f) "f";

**8. Процесс открывает существующий файл длиной 500 байт в режиме O\_WRONLY | O\_APPEND и записывает в него 10 байт. Какова длина файла после окончания записи?**

- a) 510 б;
- b) 10 б;
- c) 500 б;
- d) 0 б.



# Вопросы

**9. При условии, что `filedes` - дескриптор файла, системный вызов `lseek(fildes, (off_t)0, SEEK_END)` позволит**

- a) выполнить переход в начало файла;
- b) **определить длину файла;**
- c) уменьшить длину файла до нуля;
- d) перевести указатель в начало файла.

**10. Исходный файл содержит последовательно по 128 значений "a", 128 значений "b", "c", "d", "e", "f" и т. д. После открытия файла и получения копии дескриптора файла по системному вызову `dup()` с использованием оригинального дескриптора файла выполнено чтение трех записей по 64 байта, а с использованием копии дескриптора файла выполнено чтение трех записей по 64 байт. Последним прочитанным из файла символом является**

- |                |         |
|----------------|---------|
| a) "a";        | d) "d"; |
| b) "b";        | e) "e"; |
| c) <b>"c";</b> | f) "f"  |

**11. Исходный файл содержит последовательно 128 байт значений "a", 128 значений "b", "c", "d", "e", "f" и т.д. Программа создает новый процесс и в рамках порожденного процесса дважды открывает указанный файл и читает с использованием первого дескриптора три записи по 64 байта, а затем с использованием второго дескриптора три записи по 64 байта. Последним прочитанным символом из файла является**

- |                |         |
|----------------|---------|
| a) "a";        | d) "d"; |
| b) <b>"b";</b> | e) "e"; |
| c) "c";        | f) "f"  |

# Вопросы

**12. Каковы права доступа, при которых владелец файла и все прочие пользователи могут читать и писать в файл?**

- a) 0666;
- b) 0644;
- c) 0555;
- d) 0744.

**13. Процесс открывает существующий файл длиной 500 байт в режиме O\_WRONLY и записывает в него 10 байт. Какова длина файла после окончания записи?**

- a) 510 б;
- b) 10 б;
- c) 500 б;
- d) 0 б.

**14. Процесс открывает существующий файл длиной 500 байт в режиме O\_WRONLY и записывает в него 10 байт. После этого указанная операция записи 10 байт повторяется еще 50 раз. Какова длина файла после окончания записи?**

- a) 510 б;
- b) 10 б;
- c) 500 б;
- d) 0 б.

Тест 7

# Вопросы

**1. Для обмена двух процессов данными через неименованный программный канал минимальный набор системных вызовов в программе составляет**

- a) open(), read(), write(), close();
- b) pipe();
- c) **pipe(), read(), write();**
- d) pipe(), dup(), read(), write();
- e) dup(), read(), write()

**2. Два родственных процесса, выполняющие ввод-вывод данных, используют системный вызов pipe() для обмена данными между собой. Первая программа пишет в канал, вторая - читает. В момент, когда первая программа поместила в канал n записей длиной 128 байт, вторая программа может обнаружить в канале**

- a) 128\*n байт;
- b) 128\*n+2 байта;
- c) 128\*n-2 байта;
- d) 128\*n+1 байт;
- e) любое число байт.

**3. Именованные каналы**

- a) не работают по алгоритму FIFO;
- b) могут использоваться неродственными процессами;
- c) могут использоваться родственными процессами;
- d) уничтожаются после работы с ними;
- e) полностью аналогичны файлам файловой системы.

# Вопросы

## 4. Фрагмент программы:

...

```
if(fork()) printf("text\n");
```

...

Каким процессом будет напечатан текст?

- a) процессом-родителем; c) процессом интерпретатора;  
b) процессом-потомком; d) процессом ядра.

## 5. Фрагмент программы:

...

```
if(i=fork()==0) printf("text\n");
```

...

Каким процессом будет напечатан текст?

- a) процессом-родителем; c) процессом интерпретатора;  
b) процессом-потомком; d) процессом ядра.

## 6. Фрагмент программы:

...

```
if(i=fork()!=0) printf("text\n");
```

...

Каким процессом будет напечатан текст?

- a) процессом-родителем; c) процессом интерпретатора;  
b) процессом-потомком; d) процессом ядра.

## 7. Фрагмент программы:

...

```
fork();
```

```
printf("text\n");
```

...

Каким процессом будет напечатан текст?

- a) процессом-родителем; c) процессом интерпретатора;  
b) процессом-потомком; d) процессом ядра.

# Вопросы

**8. Сколько строк будет напечатано при выполнении программы, содержащей следующий контекст?**

```
printf("One\n");  
fork();  
printf("Two\n");
```

- a) 1; c) 3;  
b) 2; d) 4.

**9. Сколько строк будет напечатано при выполнении программы, содержащей следующий контекст?**

```
printf("One\n");  
if(fork()) printf("Two\n");
```

- a) 1; c) 3;  
b) **2;** d) 4.

**10. Сколько строк будет напечатано при выполнении программы, содержащей следующий контекст?**

```
printf("One\n");  
if(i=fork()==0) printf("Two\n");
```

- a) 1; c) 3;  
b) **2;** d) 4.

**11. После открытия процессом 2-х файлов и неименованного программного канала и создания потомка по системному вызову fork() общее число записей в таблице файлов равно**

- a) 1; d) **4;**  
b) 2; e) 5.  
c) 3;

# Вопросы

**12. После открытия процессом 2-х файлов и неименованного программного канала и создания потомка по системному вызову `fork()` общее число записей в таблице описателей файлов равно**

- a) 1;
- b) 2;
- c) **3;**
- d) 4;
- e) 5.

**13. После открытия процессом 2-х файлов и неименованного программного канала и создания потомка по системному вызову `fork()` максимальное число ссылок на записи таблицы файлов равно**

- a) 1;
- b) **2;**
- c) 3;
- d) 4;
- e) 5.

**14. После открытия процессом 2-х файлов и неименованного программного канала и создания потомка по системному вызову `fork()` максимальное число ссылок на записи таблицы описателей файлов равно**

- a) 1;
- b) **2;**
- c) 3;
- d) 4;
- e) 5.

# Тест 8



# Вопросы

## 1. Обработка сигналов выполняется

- a) при переходе из режима задачи в режим ядра;
- b) **при переходе из режима ядра в режим задачи;**
- c) при переходе в режим приостанова;
- d) при выходе из режима готовности;
- e) при выходе из режима приостанова.

## 2. Результатом нормального выполнения системного вызова wait() является

- a) нулевой код завершения;
- b) **идентификатор завершившегося процесса;**
- c) идентификатор ожидаемого процесса;
- d) ненулевой код завершения;
- e) **статус завершения.**

## 3. Процесс последовательно создает три процесса-потомка, после чего ожидает их завершения тремя системными вызовами wait(). Завершающийся процесс в качестве статуса завершения возвращает свой идентификатор. С большей вероятностью первым будет получен

- a) **идентификатор первого процесса; ????????????**
- b) идентификатор второго процесса;
- c) идентификатор третьего процесса;
- d) **идентификатор любого из трех процессов.**

## 4. Системный вызов raise()

- a) моментально посылает сигнал некоторому процессу;
- b) **моментально посылает сигнал вызвавшему его процессу;**
- c) устанавливает интервал времени, через который данному процессу будет послан сигнал;
- d) устанавливает интервал времени, через который некоторому процессу будет послан сигнал;
- e) моментально посылает сигнал процессу ядра.

# Вопросы

## 5. Если первый аргумент системного вызова `kill()` меньше нуля и не равен -1, то

- a) ядро посылает сигнал процессу с идентификатором, равным модулю аргумента;
- b) ядро посылает сигнал всем процессам, входящим в одну группу с этим процессом;
- c) **ядро посылает сигнал всем процессам, идентификатор группы которых равен этому модулю первого аргумента;**
- d) ядро посылает сигнал всем процессам, идентификатор группы которых равен этому модулю второго аргумента;

## 6. При выполнении системного вызова `exec()` задействованы

- a) подсистема управления процессами;
- b) подсистема управления файлами;
- c) подсистема управления устройствами;
- d) **подсистема управления процессами и подсистема управления файлами;**

## 7. Проверка поступления сигналов выполняется

- a) при переходе из режима задачи в режим ядра;
- b) **при переходе из режима ядра в режим задачи;**
- c) **при переходе в режим приостанова;**
- d) при выходе из режима готовности;
- e) **при выходе из режима приостанова.**

## 8. Возможные действия в случае нулевого второго аргумента системного вызова `Signal`

- a) процесс игнорирует все последующие получения сигнала;
- b) **по получению сигнала процесс завершается;**
- c) сигнал посылается всем процессам, входящим с данным процессом в одну группу;
- d) сигнал посылается всем процессам, у которых код идентификатора пользователя совпадает с тем, под которым выполняется процесс;
- e) процесс передает управление функции обработки сигнала.

# Вопросы

**9. Статус завершения процесса, переданный из процесса-потомка в родительский процесс, доступен**

- a) в родительском процессе в любом случае;
- b) **в родительском процессе в случае, если в процесс-родитель выдан системный вызов wait();?**
- c) в родительском процессе только в случае успешного завершения потомка;
- d) в родительском процессе только в случае аварийного завершения потомка.

**10. Системный вызов alarm()**

- a) моментально посылает сигнал некоторому процессу;
- b) моментально посылает сигнал вызвавшему его процессу;
- c) **устанавливает интервал времени, через который данному процессу будет послан сигнал;**
- d) устанавливает интервал времени, через который некоторому процессу будет послан сигнал;
- e) моментально посылает сигнал процессу ядра.

**11. Результатом аварийного завершения системного вызова wait() является**

- a) нулевой код завершения;
- b) идентификатор завершившегося процесса;
- c) идентификатор ожидаемого процесса;
- d) ненулевой код завершения;
- e) **статус завершения. ???**

**12. Возможные действия в случае ненулевого второго аргумента системного вызова Signal**

- a) **процесс игнорирует все последующие получения сигнала;**
- b) по получению сигнала процесс завершается;
- c) сигнал посылается всем процессам, входящим с данным процессом в одну группу;
- d) сигнал посылается всем процессам, у которых код идентификатора пользователя совпадает с тем, под которым выполняется процесс;
- e) **процесс передает управление функции обработки сигнала.**

# Вопросы

## 13. При создании процесса системным вызовом `fork()`

- a) приоритет процесса-потомка меньше приоритета процесса-родителя;
- b) приоритет процесса-потомка больше приоритета процесса-родителя;
- c) **приоритет процесса-потомка равен приоритету процесса-родителя;**
- d) приоритет процесса-потомка может быть определен в момент создания процесса.

# Тест 9

# Вопросы

**1. Процесс открывает файлы f1, f2, делает копию дескриптора файла f1, порождает новый процесс, который закрывает файл f2, а затем вновь открывает его. Каково число записей в таблице открытых файлов?**

- |              |        |
|--------------|--------|
| a) 6; ?????? | e) 10; |
| b) 7;        | f) 11; |
| c) 8;        | g) 12; |
| d) 9;        | h) 13. |

**2. Процесс открывает файлы f1, f2, делает копию дескриптора файла f1, порождает новый процесс, который закрывает файл f2, а затем вновь открывает его. Каково число записей в таблице файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**3. Процесс открывает файлы f1, f2, делает копию дескриптора файла f1, порождает новый процесс, который закрывает файл f2, а затем вновь открывает его. Каково число записей в таблице описателей файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**4. Процесс открывает файлы f1, f2, делает копию дескриптора файла f1, порождает новый процесс, который закрывает файл f2, а затем вновь открывает его. Каково максимальное число ссылок в записях таблицы файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

# Вопросы

**5. Процесс открывает файлы f1, f2, делает копию дескриптора файла f1, порождает новый процесс, который закрывает файл f2, а затем вновь открывает его. Каково минимальное число ссылок в записях таблицы файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**6. Процесс открывает файл f1 и непоименованный программный канал, порождает новый процесс и переназначает стандартный вывод в канал. Каково число записей в таблице открытых файлов?**

- |                   |        |
|-------------------|--------|
| a) 6; ??????????? | e) 10; |
| b) 7;             | f) 11; |
| c) 8;             | g) 12; |
| d) 9;             | h) 13. |

**7. Процесс открывает файл f1 и непоименованный программный канал, порождает новый процесс и переназначает стандартный вывод в канал. Каково число записей в таблице файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**8. Процесс открывает файл f1 и непоименованный программный канал, порождает новый процесс и переназначает стандартный вывод в канал. Каково число записей в таблице описателей файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

# Вопросы

**9. Процесс открывает файл f1 и непоименованный программный канал, порождает новый процесс и переназначает стандартный вывод в канал. Каково максимальное число ссылок в записях таблицы файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**10. Процесс открывает файл f1 и непоименованный программный канал, порождает новый процесс и переназначает стандартный вывод в канал. Каково минимальное число ссылок в записях таблицы файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**11. Процесс открывает файл f1 и непоименованный программный канал, порождает новый процесс и переназначает стандартный вывод в канал. Каково максимальное число ссылок в записях таблицы описателей файлов?**

- |       |       |
|-------|-------|
| a) 1; | d) 4; |
| b) 2; | e) 5. |
| c) 3; |       |

**12. Интерпретатор команд выполняет командную строку: `ls -la | wc -c | sort`, порождая два дочерних процесса и корректно организовав между ними непоименованный канал. Каково число записей в таблице открытых файлов?**

- |       |        |
|-------|--------|
| a) 6; | e) 10; |
| b) 7; | f) 11; |
| c) 8; | g) 12; |
| d) 9; | h) 13. |



# Вопросы

**13. Интерпретатор команд выполняет командную строку: `ls -la | wc -c | sort`, порождая два дочерних процесса и корректно организуя между ними непоименованный канал. Каково максимальное число ссылок в записях таблицы файлов?**

- |              |       |
|--------------|-------|
| a) <b>1;</b> | d) 4; |
| b) 2;        | e) 5. |
| c) 3;        |       |

**14. Интерпретатор команд выполняет командную строку: `ls -la | wc -c | sort`, порождая два дочерних процесса и корректно организуя между ними непоименованный канал. Каково максимальное число ссылок в записях таблицы описателей файлов?**

- |              |       |
|--------------|-------|
| a) 1;        | d) 4; |
| b) <b>2;</b> | e) 5. |
| c) 3;        |       |

**15. Интерпретатор команд выполняет командную строку: `ls -la | wc -c | sort`, порождая два дочерних процесса и корректно организуя между ними непоименованный канал. Каково минимальное число ссылок в записях таблицы файлов?**

- |              |       |
|--------------|-------|
| a) <b>1;</b> | d) 4; |
| b) 2;        | e) 5. |
| c) 3;        |       |

# Тест 10

# Вопросы

## 1. Выполнение P-операции P(S) над классическим семафором

- a) ведет к уменьшению значения аргумента на 1;
- b) равносильно операции  $S=S+1$ ;
- c) **неделимая операция, уменьшающая положительное значение аргумента на 1;**
- d) ведет к увеличению значения аргумента на 1;
- e) равносильно операции  $S=S-1$ ;
- f) производится над любым целочисленным аргументом.

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

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

## 3. Системный вызов msgget() набора системных средств IPC позволяет

- a) получить сообщение из очереди сообщений;
- b) послать сообщение в очередь сообщений;
- c) образовать новую очередь сообщений;
- d) **получить дескриптор существующей очереди. ???**

## 4. Набор программных средств IPC может выступать как средство взаимодействия

- a) родственных процессов;
- b) **процессов, не связанных отношением родства; ???**
- c) процессов, имеющих общего предка;
- d) процессов, имеющих общего владельца.

# Вопросы

## 5. Ключ объекта IPC

- a) является уникальным в рамках программы пользователя;
- b) **является уникальным в рамках группы процессов, работающих с объектом;**
- c) является уникальным в рамках вычислительной системы (ОС);
- d) является уникальным в рамках вычислительной сети.

## 6. В случае заполненности очереди сообщений и невозможности поместить в нее сообщение процесс, выдавший системный вызов `msgsnd()`,

- a) заканчивается аварийно с соответствующим кодом завершения;
- b) **замораживается до появления возможности занести сообщение;**
- c) заносит сообщение в буфер и продолжает выполнение;
- d) игнорирует занесение сообщения в очередь и продолжает выполнение.

## 7. Семафор в ОС Unix не содержит

- a) значения семафора;
- b) идентификатора процесса, который хронологически последним работал с семафором;
- c) числа процессов, ожидающих увеличения значения семафора;
- d) **числа процессов, ожидающих уменьшения значения семафора;**
- e) числа процессов, ожидающих нулевого значения семафора;

## 8. В заголовке очереди не содержится

- a) указатель на первое сообщение в очереди;
- b) указатель на последнее сообщение в очереди;
- c) число сообщений в очереди;
- d) общий размер сообщений очереди в байтах;
- e) **указатель на следующий элемент очереди;**
- f) идентификаторы процессов, которые последними послали или приняли сообщение через данную очередь

# Вопросы

## 9. Выполнение V-операции V(S) над классическим семафором

- a) **ведет к увеличению значения аргумента на 1; ????**
- b) равносильно операции  $S=S+1$ ;
- c) ведет к увеличению значения аргумента на 1 для любого целочисленного аргумента;
- d) равносильно операции  $S=S-1$ ;
- e) производится над любым целочисленным аргументом;
- f) **неделимая операция, увеличивающая неотрицательное значение аргумента на 1.**

## 10. Механизм очередей сообщений служит для обмена сообщениями

- a) **родственных процессов; ?????????**
- b) процессов, не связанных отношением родства;
- c) процессов, имеющих общего предка;
- d) процессов, имеющих общего владельца.

## 11. Системный вызов shmget() набора системных средств IPC позволяет

- a) подключить сегмент разделяемой памяти к виртуальной памяти процесса;
- b) отключить сегмент разделяемой памяти от виртуальной памяти процесса;
- c) образовать новый сегмент разделяемой памяти;
- d) **найти сегмент разделяемой памяти по ключу.**

## 12. Номер семафоров (индекс) в наборе семафоров

- a) должен быть меньше или равен числу семафоров;
- b) **должен быть меньше числа семафоров;**
- c) должен быть положительным значением;
- d) может быть больше числа семафоров.

# Вопросы

**13. Родительский процесс создал неименованный канал, получив дескрипторы f[0], f[1], после чего породил дочерний процесс. Оба процесса используют канал для двухсторонней передачи данных, при этом каждый из процессов использует дескриптор f[0] для чтения, а f[1] - для записи. Описанная схема**

- a) обеспечивает гарантированную двухстороннюю передачу данных;
- b) дает непредсказуемый результат, включая возможные дедлоки;
- c) соответствует спецификации системного вызова pipe();
- d) аналогична схеме использования именованного канала;

**14. Родительский процесс создал неименованный канал, получив дескрипторы f[0], f[1], после чего породил дочерний процесс. Оба процесса используют канал для двухсторонней передачи данных, при этом и для чтения и для записи родительский процесс использует дескриптор f[0], а порожденный процесс - дескриптор f[1]. Описанная схема**

- a) соответствует спецификации системного вызова pipe();
- b) дает непредсказуемый результат, включая возможные дедлоки;
- c) обеспечивает гарантированную двухстороннюю передачу данных
- d) аналогична схеме использования именованного канала;