Примеры решения задачи на анализ поведения параллельных процессов

Что будет выведено на экран в результате работы программы? Если возможны несколько вариантов – привести все. Предполагается, что обращение к функции вывода на экран прорабатывает атомарно и без буферизации. Все системные вызовы прорабатывают успешно. Подключение заголовочных файлов опущено.

Пример 1.

```
int main()
{
      pid_t pid;
      int fd[2];
      int x ;//отсутствие инициализации не считается в языке Си ошибкой
      pipe(fd);
      write(1,"a",1);
      if( (pid = fork()) > 0 ) /* точка 1 */
          {  write(1,"b",1);
             read(fd[0], &x, sizeof(int)); /*  точка 2 */
             write(1,"c",1);
             wait(NULL); /* точка 3 */
             write(1,"d",1);
      }
      else { write(1,"e",1);
              write(fd[1], &x, sizeof(int)); /*точка 4*/
              write(1,"f",1);
      }

      return 0;
}
```

Отметим основные моменты, влияющие на синхронизацию процессов.

1) До точки 1 существует один процесс, после нее -- два параллельных процесса: отец и сын.
2) Точка 2 является точкой синхронизации: отец ждет, пока в канале появится информация от сына, если ее еще нет там. (Если нет потенциального писателя, то синхронизации не будет)
3) Точка 3 является точкой синхронизации: отец ждет, пока завершится сын, если он еще не завершился.
4) Эта точка не синхронизирует сына, поскольку он пишет в пустой канал (если бы канал до этого момента полностью заполнился, сын ожидал бы чтения из канала от другого процесса; если нет потенциального читателя, то писатель получает сигнал SIGPIPE)

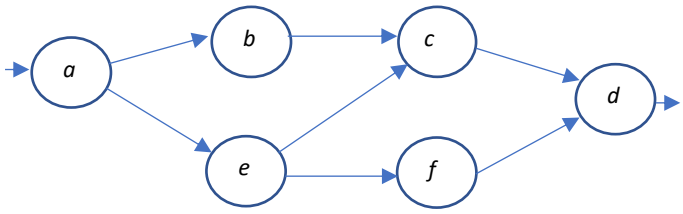
Введем на множестве печатаемых символов отношение частичного порядка:
$x \prec y$ тогда и только тогда, когда x печатается раньше y.

В соответствии с замечаниями выше о синхронизации имеем:
(1) $'a' \prec 'b'$, $'a' \prec 'c'$, $'a' \prec 'd'$, $'a' \prec 'e'$, $'a' \prec 'f'$
(2) $'e' \prec 'c'$, $'e' \prec 'd'$
(3) $'f' \prec 'd'$
(4) нет соотношений

В соответствии с порядком следования операторов в программе имеем дополнительные соотношения:

$$'b' \prec 'c', 'b' \prec 'd', 'c' \prec 'd',$$
$$'e' \prec 'f'$$

Построим граф, отражающий частичный порядок:



В соответствии с этим графом возможны следующие полные порядки обхода вершин:

abecfd
aebcfd
abefcd
aebfcd
aefbcd

Это и есть все возможные варианты ответа.

Пример 2.

```
int main()
{
      pid_t pid;
      int fd[2];
      int x ;
      pipe(fd);
       write(1,"a",1);
      if( (pid = fork()) > 0 ) /* точка 1 */
          {  write(1,"b",1);
             read(fd[0], &x, sizeof(int)); /*  точка 2 */
             write(1,"c",1);
             kill(pid, SIGKILL);
             wait(NULL); /* точка 3 */
             write(1,"d",1);
      }
      else { write(1,"e",1);
              write(fd[1], &x, sizeof(int)); /*точка 4  */
              write(1,"f",1);
      }

      return 0;
}
```

По сравнению с предыдущим примером отец посылает сыну сигнал SIGKILL , причем делает это не раньше, чем сын запишет информацию в канал в точке 4, так как отец ждет

сына в точке 2. Доставка сигнала не регламентирована по времени, может быть мгновенно, а может очень долго. Поэтому сын может успеть напечатать *'f'*, а может и не успеть.

Следовательно, ко всем вариантам ответа из Примера 1 добавляются варианты с отсутствующим *'f'*.

abecfd
aebcfd
abefcd
aebfcd
aefbcd
abecd
aebcd

Это все варианты ответа к Примеру 2.