**1я прога.**

1. В какие моменты и какие действия выполняет система для сохранения иерархии предок-потомок?

При завершении любого процесса система анализирует таблицы дескрипторов и если находит незавершившиеся потомки этого процесса выполняет следующие действия: PPID потомков присваивает 1 (это PID процесса, открывшего терминал), меняет указатель на родителя каждого потомка на init, у init’a указатели на предков добавляет эти потомки.

1. Что возникает в процессе выполнения 1й программы?

Процесс сирота. Это видно по PID Child’a, который равен 1.

1. Почему родитель завершается раньше потомка?

Потому что нет системного вызова wait()

Что такое процесс сирота? Процесс сирота – процесс, чей предок закончился раньше потомка.

**2я прога.**

1. Что есть такого в предке, чего нет в потомке?

Системный вызов wait().

1. Что делает системный вызов wait()?

Cистемный вызов wait() блокирует родительский процесс до момента завершения дочернего.

1. Что возвращает wait()?

Процесс-предок получает статус завершения процесса-потомка.

1. После усыновления init’ом потомка, что ожидает init от wait’a?(что то в этом духе)

Статус завершения потомка.

1. Что значит процесс блокировки sleep() с точки зрения системы?

Не стоит в очереди на получение главного ресурса – процессорного времени.

1. Что происходит с предком после вызова wait()?

Он блокирован и не получает процессорное время.

**3я прога.**

1. Что делает системный вызов exec(). Особенности. Почему он низкоуровневый?

Exec() создает новое адресное пространство. Создаются таблицы страниц для адресного пространства программы, указанной в exec(), но программа на выполнение не запускается, поэтому низкоуровневый.

В результате системного вызова exec() адресное пространство процесса будет заменено на адресное пространство новой программы, а сам процесс будет возвращен в режим задачи с установкой указателя команд на первую выполняемую инструкцию этой программы.

1. Что такое зомби и как они появляются.

Процесс-зомби – это процесс, у которого отобраны все ресурсы, кроме последнего – строки в таблице процессов.

Если потомок быстро завершился, а предок не успел вызвать wait() или завершился не нормально, возникает зомби.

**4я прога.**

1. Что делает системный вызов pipe()?

Системный вызов pipe() создает неименованный программный канал. Неименованные программные каналы могут использоваться для обмена сообщениями между процессами родственниками. В отличие от именованных программных каналов неименованные не имеют идентификатора, но имеют дескриптор.

1. В какой области памяти размещаются программные каналы и почему?

В системной области памяти, т.к. адресные пространства процессов защищены и недоступны для других процессов.

**5я прога.**

1. Что это у вас за функция?

Обработчик сигнала.

1. Как задать обработчик сигнала?

С помощью системного вызова Signal().

1. Какой сигнал используете вы?

Ctrl+C.

1. Для чего можно использовать сигналы?

Для того, чтобы менять ход выполнения программы извне.

Рязанова требует в пятой программе вызов signal переместить в main, kill убрать и по флагу менять ход выполнения программы. Этот вариант программы prog05\_new.c.

Еще прочтите Вакхалию Главу 2 начиная с 2.7 – там все более менее понятно написано и Оптимизацию fork.