**Пометка к заданию №12**

**Задание 1: Неименованный канал**

Описание

Программа демонстрирует взаимодействие между родительским и дочерним процессами через неименованный канал (pipe).

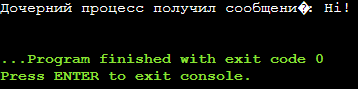
* Родитель пишет сообщение "Hi!" в канал.
* Дочерний процесс читает сообщение из канала и выводит его.

Работа программы

1. Создаётся неименованный канал с помощью pipe(pipefd).
2. Выполняется форк:
   * В родительском процессе закрывается дескриптор чтения pipefd[0], и сообщение записывается в канал.
   * В дочернем процессе закрывается дескриптор записи pipefd[1], сообщение считывается и выводится.

Ключевые функции

* pipe(): Создаёт канал.
* fork(): Создаёт дочерний процесс.
* write(): Записывает данные в канал.
* read(): Читает данные из канала.

Пример вывода:  


**Задание 2: Именованный канал**

**Описание**

Сервер и клиент взаимодействуют через именованный канал (FIFO).

* Сервер создаёт FIFO, пишет в него сообщение "Hi!" и завершает работу.
* Клиент открывает FIFO, читает сообщение и удаляет FIFO.

**Работа программы**

1. Сервер:
   * Создаёт FIFO с помощью mkfifo().
   * Открывает канал для записи и пишет сообщение.
   * Завершает работу.
2. Клиент:
   * Открывает FIFO для чтения и считывает сообщение.
   * Удаляет FIFO после завершения.

**Ключевые функции**

* mkfifo(): Создаёт именованный канал.
* open(): Открывает FIFO для чтения или записи.
* write() и read(): Передают данные через FIFO.
* unlink(): Удаляет FIFO.

Пример вывода:  




**Задание 3: Командный интерпретатор с поддержкой пайпов**

**Описание**

Командный интерпретатор поддерживает конструкции вида команда1 | команда2.

* Реализовано выполнение двух команд, соединённых пайпом.
* Например: ls -la | grep a.

**Работа программы**

1. Парсинг строки команды:
   * Разделяет команды по символу |.
2. Создание канала с помощью pipe().
3. Выполнение команд:
   * Родительский процесс перенаправляет стандартный вывод первой команды в канал.
   * Дочерний процесс читает данные из канала и перенаправляет их на вход второй команды.
4. Обработка завершения команд.

**Ключевые функции**

* pipe(): Создаёт канал между процессами.
* dup2(): Перенаправляет стандартный ввод/вывод.
* fork(): Создаёт процессы для выполнения команд.
* execlp(): Выполняет команды.

**Пример ввода и вывода**

