1. Обработка ошибок:

- Команда начинается с символов “ | ”, “ & ”, “ ; ” => ошибка
- Очередная команда конвейера начинается с символов “ | ”, “ & ”, “ ; ” => ошибка
- Команда заканчивается символом “ | ” => ошибка
- Очередная команда последовательности, разделенной начинается “ | ”, начинается с символов “ | ”, “ & ”, “ ; ” => ошибка
- Команда заканчивается символом “ ; ” => ошибка
- Очередная команда последовательности, разделенной начинается “ & ”, начинается с символов “ | ”, “ & ”, “ ; ” => ошибка
- После “ > ”, “ < ”, “ >> ” стоят символы “ | ”, “ & ”, “ ; ”, “ > ”, “ < ”, “ >> ” => ошибка
- Стоит только одна кавычка => она игнорируется

2. Набор тестов

(a) Правильные:

• cd ..
pwd
Перейти в родительскую директорию, напечатать ее название

• echo homedir: $HOME username: $USER
Напечатать имя домашней директории и имя пользвателя

• cd
pwd
Перейти по умолчанию в домашнюю директорию. Убедиться, что попали туда

• pwd > fpwd
имя текущей директории записывается в файл fpwd

• who >> fpwd
имя пользователя добавляется в конец файла fpwd

• cat < file.in > file.out
Копирует содержимое файла file.in в file.out.

• cat < file.in >> file.out
Дописывает содержимое файла file.in в конец file.out.

• zcat < file.tar.gz | tar -xv
Распаковывает .tar.gz архив.

• cat < /dev/urandom | head -c 4096 > "file.bin"
Создаёт файл случайных чисел размером 4096 байт.

• yes | head
Печатает 10 раз букву y.

• yes | yes | yes | yes | yes | yes | yes | yes | head
Печатает 10 раз букву y. Проследить с помощью ps или top из другого окна xterm, чтобы не оставались
Зомби

• cat < /dev/null | head | head | head | head | head
Ничего не печатает. Проследить с помощью ps или top из другого окна xterm, чтобы не оставались зомби

• pwd > current_dir.txt
Печатает текущую директорию в файл current_dir.txt.

• pwd | cat >> my_home.txt
Дописывает домашнюю директорию пользователя в файл my_home.txt.

• ls | cat | cat | cat | cat
Печатает список файлов текущей директории на стандартный вывод.

• who | cat > who_am_i.txt | cat | cat | cat > who_am_i_not.txt
Печатает имя пользователя в файл who_am_i.txt. Файл who_am_i_not.txt создаётся и оказывается
пустым.

• yes | yes | yes | yes | sleep 10 | pwd
Печатает текущую директорию. Проследить, чтобы приглашение к следующей команде появилось не
раньше, чем через 10 сек.
Выводит пронумерованные строки, введенные с клавиатуры, и отсортированные в обратном порядке

• cat | sort –r | cat –n
Печатает вводимые строки, отсортированные в обратном порядке и пронумерованные

• pwd; pwd | cd..| wc ; pwd
Печатает одно и то же имя директории – внутри конвейера cd игнорируется, аналогичное поведение для
exit и других внутренних команд. В конвейере они игнорируются.

• pwd | exit | date | cat
Печатает только текущую дату.

• sleep 10; pwd
Директория печатается через 10 сек

• sleep 10 & pwd
Директория печатается сразу
Проверить, что не остается зомби от фонового процесса sleep 10 &

• sleep 5 & sleep 5 & sleep 5 & sleep 5 & sleep 5 &
Проверить зомби (с помощью команды ps или top)

• sleep 5 &
sleep 5 # набирать эту команду быстро
ps
Проверить зомби (с помощью команды ps или top)

• echo a b c > f; cat f & ls

• echo a b c > f; echo >> f; cat<f
Проверить зомби

• cat f; date; pwd > zz

(б) Демонстрирующие ошибки:

• | cat f; date; pwd > zz

• & ls & who & pwd

• ; ls | who | pwd

• pwd & sleep 5 |

• ls | who | cat ;

• who |cat ; | ls

• yes | yes | yes | ; yes | sleep 10 | pwd

• sleep 10 ; & sleep 10 ; sleep 10 | pwd

• pwd > | fpwd

• who >> ; fpwd

• cat < > file.in > & file.out

3. Описание внутренней структуры данных

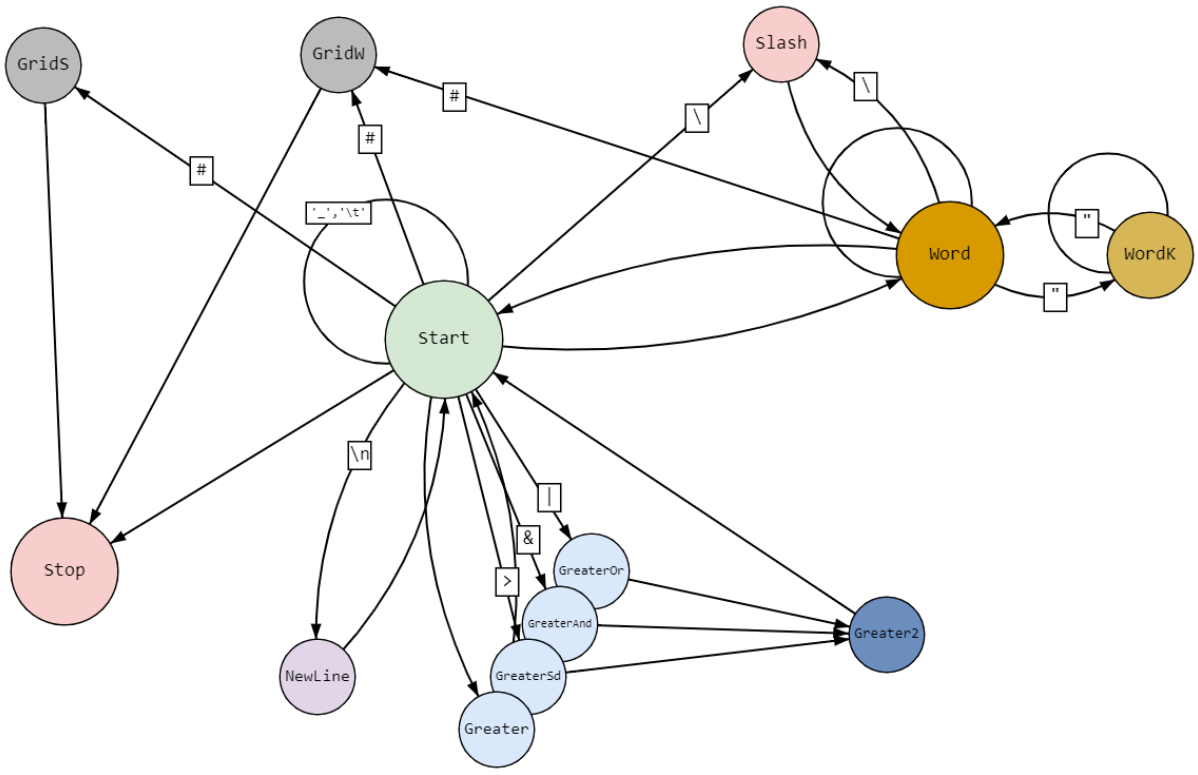
typedef char ** List;

```
struct cmd_inf
  {
    List argv;            /* список из имени команды и аргументов */
    char *infile;         /* переназначенный файл стандартного ввода */
    char *outfile;        /* переназначенный файл стандартного вывода */
    int append;           /* =1, если файл надо открыть на дозапись*/
    int backgrnd;         /* =1, если команда подлежит выполнению в фоновом режиме */
    struct cmd_inf* pipe;  /* следующая команда после “|” */
    struct cmd_inf* next;  /* следующая команда после “|” */
  };
```

4. Описание графов.

Граф программы List.c:

- Start - начало работы
- Slash - обработка символа \
- GridS, GridW - обработка символа #
- Word - обработка слов
- WordK - обработка кавычек
- Greater - обработка обычных спецсимволов (т.е. состоящих из 1 символа)
- GreaterOr - обработка спецсимволов | и ||
- GreaterAnd - обработка спецсимволов & и &&
- GreaterSd - обработка спецсимволов > и >>
- Greater2 - обработка двойных спецсимволов
- Stop - завершение работы



Граф программы Tree.c:

- Begin - начало работы
- Conv - обработка команды с ее аргументами
- End - завершение работы
- Backgrnd - обработка спесимвола &
- In, Out, Out1 - работа с файлами (вход, выход, выход с флагом append)
- Conv1 - обработка конвейера
- ConvP - обработка спесимвола ;