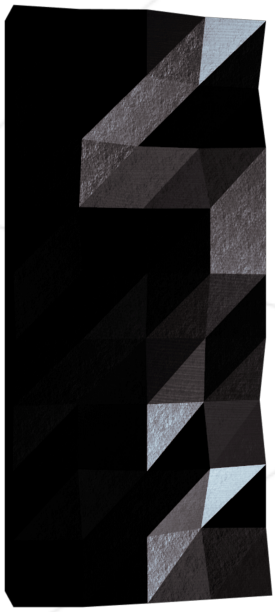

Мини-оболочка

Прекрасна как ракушка

Резюме: цель этого проекта - создать простую оболочку. Да, ваши маленький баш или зш. Вы узнаете много нового о процессах и файловых дескрипторах.

СОДЕРЖАНИЕ

Глава I

Вступление

Существование оболочек связано с самим существованием ИТ. Тогда все кодировщики согласились что общение с компьютером с помощью выровненных переключателей 1/0 было серьезно раздражает. Было вполне логично, что им в голову пришла идея общаться с компьютер, использующий интерактивные строки команд на языке, близком к английскому.

С участием Мини-оболочка, вы сможете путешествовать во времени и возвращаться к проблемам, с которыми сталкивались люди, когда Окна не существовало.

Глава II.

Общие инструкции

- Ваш проект должен быть написан в соответствии с Нормой. Если у вас есть бонусные файлы / функции, они будут включены в проверку нормы, и вы получите0 если внутри есть ошибка нормы.
- Ваши функции не должны завершаться неожиданно (ошибка сегментации, ошибка шины, двойное освобождение и т. Д.), За исключением неопределенного поведения. В этом случае ваш проект будет считаться нефункциональным и получит уведомление0 во время оценки.
- При необходимости все пространство памяти, выделенное кучей, должно быть правильно освобождено. Утечки недопустимы.
- Если предмет требует этого, вы должны отправить Makefile который скомпилирует ваши исходные файлы в требуемый вывод с помощью флагов -Стена, -Wextra а также -Веррор, и ваш Make-файл не должен повторно связываться.
- Ваш Makefile должен как минимум содержать правила $ (ИМЯ), all, clean, fclean а также ре.
- Чтобы превратить бонусы в свой проект, вы должны включить правило бонус в ваш Make-файл, который добавит все различные заголовки, библиотеки или функции, запрещенные в основной части проекта. Бонусы должны быть в другом файле _бонус. {c / h}.
  Оценка обязательной и бонусной части проводится отдельно.
- Если ваш проект позволяет вам использовать свой libft, вы должны скопировать его источники и связанные с ним Makefile в libft папка с соответствующим Make-файлом. Ваш проект Makefile должен скомпилировать библиотеку, используя ее Makefile, затем скомпилируйте проект.
- Мы рекомендуем вам создавать тестовые программы для вашего проекта, даже если эта работа **не должны быть отправлены и не будут оцениваться.** Это даст вам возможность легко проверить свою работу и работу ваших коллег. Вы найдете эти тесты особенно полезными во время защиты. Действительно, во время защиты вы можете использовать свои тесты и / или тесты партнера, которого вы оцениваете.
- Отправьте свою работу в назначенный репозиторий git. Оцениваться будет только работа в репозитории git. Если DeepThreadt назначен для оценки вашей работы, это будет сделано после ваших оценок коллег. Если во время выставления оценок Deepoughtt в каком-либо разделе вашей работы произойдет ошибка, оценка будет остановлена.

Глава III.

Обязательная часть

Название программы	мини-ракушка
Сдать файлы	
Сделать файл	да
Аргументы	
Внешние функции.	readline, rl_on_new_line, rl_replace_line, rl_redisplay, add_history, printf, malloc, free, write, open, read, close, fork, wait, waitpid, wait3, wait4, signal, kill, exit, getcwd, chdir, stat, lstat, fstat, unlink, execve, dup, dup2, pipe, opendir, readdir, closedir, strerror, errno, isatty, ttyname, ttyslot, ioctl, getenv
Libft авторизован	да
Описание	Напишите оболочку

Ваша оболочка должна:

- Не интерпретировать незакрытые кавычки или неуказанные специальные символы, такие как \ или ;.
- Не используйте более одной глобальной переменной, подумайте об этом и будьте готовы объяснить, почему вы это делаете.
- Показывать подсказку при ожидании новой команды.
- Иметь рабочую историю.
- Найдите и запустите правильный исполняемый файл (на основе переменной PATH или с использованием относительного или абсолютного пути)
- Он должен реализовывать встроенные функции:
    - эхо с опцией -n
    - CD только с относительным или абсолютным путем
    - pwd без вариантов
    - экспорт без вариантов
    - сброшен без вариантов

    - env без вариантов или аргументов
    - выход без вариантов
- ' запретить любую интерпретацию последовательности символов.
- " запретить любую интерпретацию последовательности символов, кроме символа $.
- Перенаправления:
    - < должен перенаправить ввод.
    - > должен перенаправить вывод.
    - «<<» считывает ввод из текущего источника до тех пор, пока не будет видна строка, содержащая только ограничитель. не нужно обновлять историю!
    - «>>» должен перенаправлять вывод в режиме добавления.
- Трубы | Выход каждой команды в конвейере соединяется через конвейер с входом следующей команды.
- Переменные среды ($, за которыми следуют символы) должны расшириться до своих значений.
- $? должен расшириться до статуса выхода последнего выполненного конвейера переднего плана.
- ctrl-C ctrl-D ctrl- \ должен работать как в bash.
- В интерактивном режиме:
    - ctrl-C напечатать новое приглашение на новой строке.
    - ctrl-D выйти из оболочки.
    - ctrl- \ ничего не делать.

Не требуется ничего, о чем не спрашивали.
Для каждого пункта, если у вас есть сомнения, возьмите трепать в качестве справки.

Глава IV.

Бонусная часть

- Если Обязательная часть не идеальна, даже не думайте о бонусах
- &&, || со скобками для обозначения приоритетов.
- wilcard * должен работать для текущего рабочего каталога.