Отчет по лабораторной работе № 14

Программирование в командном процессоре ОС Unix. Расширенное программирование

Скобеева Алиса Алексеевна

Содержание

# 1 Цель работы

Изучить ОП в оболочке ОС Unix. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 2 Задание

Написать три командных файла с использованием логических управляющих конструкций и циклов.

# 3 Выполнение лабораторной работы

Пишем командный файл, реализующий упрощенный механизм семафоров. Также дорабатываем программу так, чтобы имелась возможность взаимодействия трёх и более процессов.



Рис. 1: Пишем программу

Запускаем файл и открываем еще два терминала, смотрим на работу файла также и в них

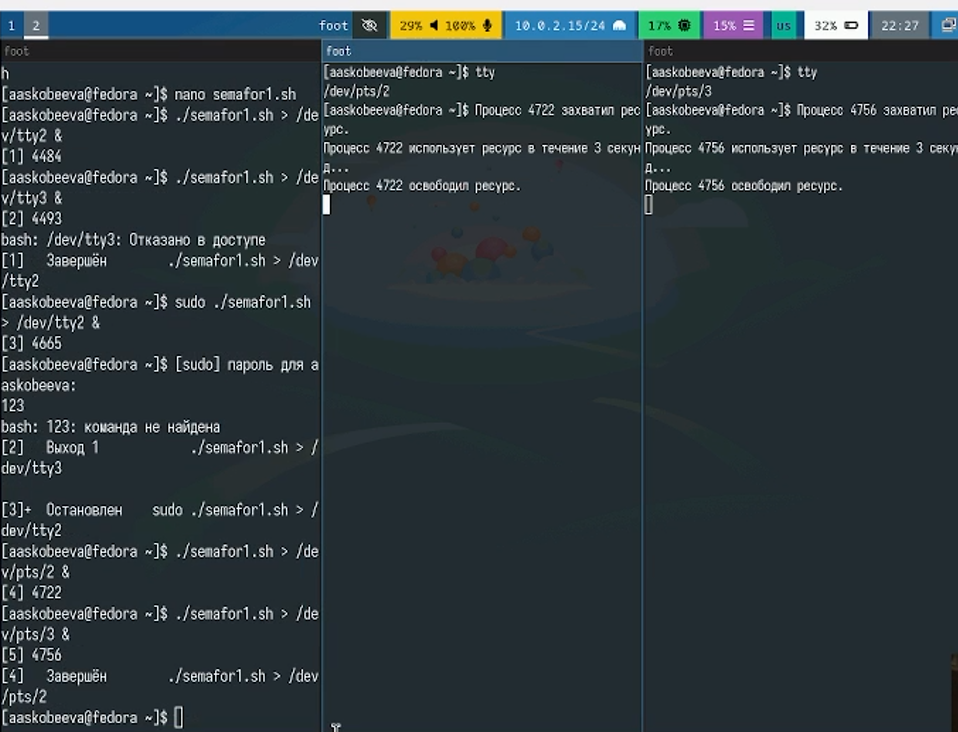


Рис. 2: Проверка работы программы

Реализовываем команду man с помощью командного файла

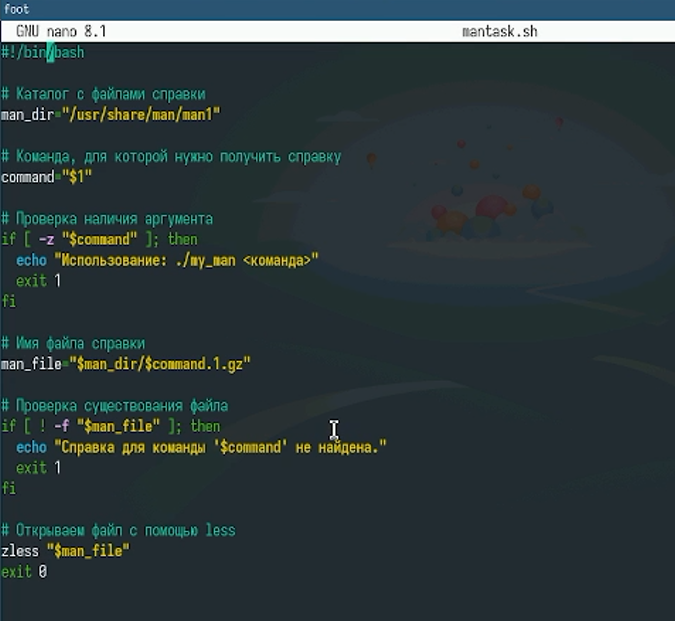


Рис. 3: Пишем программу

Запускаем файл

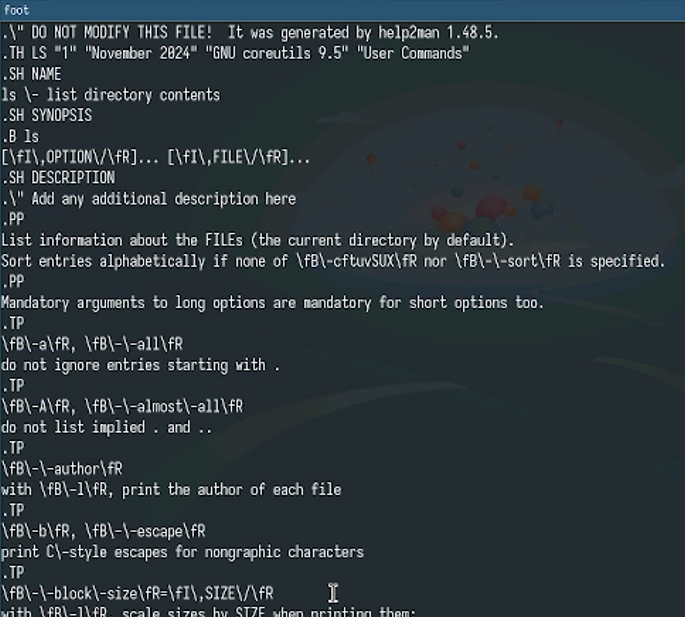


Рис. 4: Программа работает корректно

Пишем командный файл, который используя встроенную переменную $RANDOM генерирует случайную последовательность букв латинского алфавита

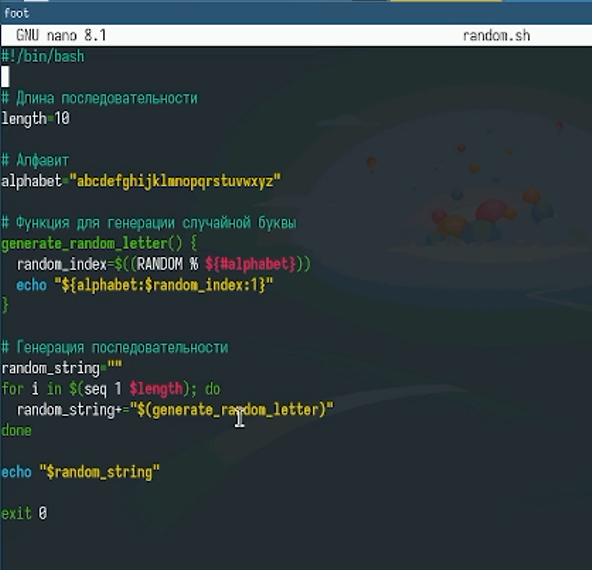


Рис. 5: Пишем программу

Запускаем файл и проверяем корректность работы программы

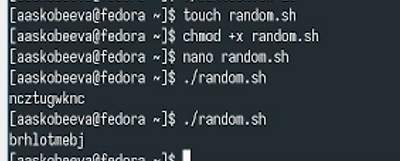


Рис. 6: Все работает корректно

# 4 Выводы

Мы успешно написали 3 командных файла с использованием логических управляющих конструкций и циклов.

# 5 Ответы на контрольные вопросы

1. Синтаксическая ошибка:

Bash while [ “$1” != “exit” ]

Необходимы пробелы вокруг [ и ], а также кавычки вокруг $1, чтобы избежать ошибок, если $1 не определена. 2. Конкатенация строк:

Bash result=“string2$string3”

1. Утилита seq: Генерирует последовательность чисел. Другие способы:
   * Цикл for: for i in $(eval echo {1..10}); do … done
   * while: i=1; while [ ((i+1)); done
2. Результат ((10/3)): 3 (целочисленное деление).
3. zsh vs. bash:
   * Расширенная автодополнение
   * Темы оформления и плагины
   * Более гибкая настройка
4. Синтаксис for ((a=1; a <= LIMIT; a++)): Синтаксис верен.
5. bash vs. другие ЯП:
   * Преимущества: Прямой доступ к командам ОС, удобство управления процессами, хорошо подходит для автоматизации задач.
   * Недостатки: Ограниченные возможности в обработке сложных данных, менее эффективен для вычислительных задач, сложнее в поддержке крупных проектов. Bash - скриптовый язык, другие языки могут быть скомпилированы.