Отчёт по лабораторной работе №13

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Николаев Дмитрий Иванович

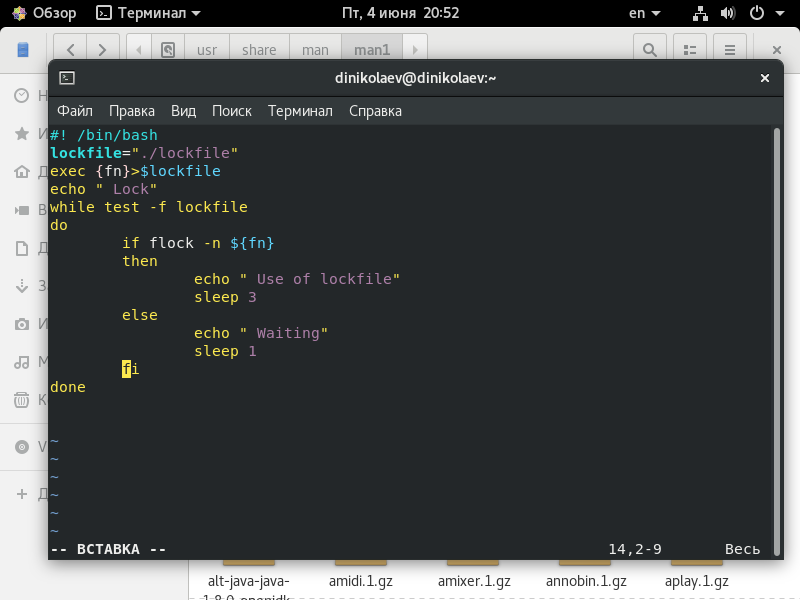
Содержание

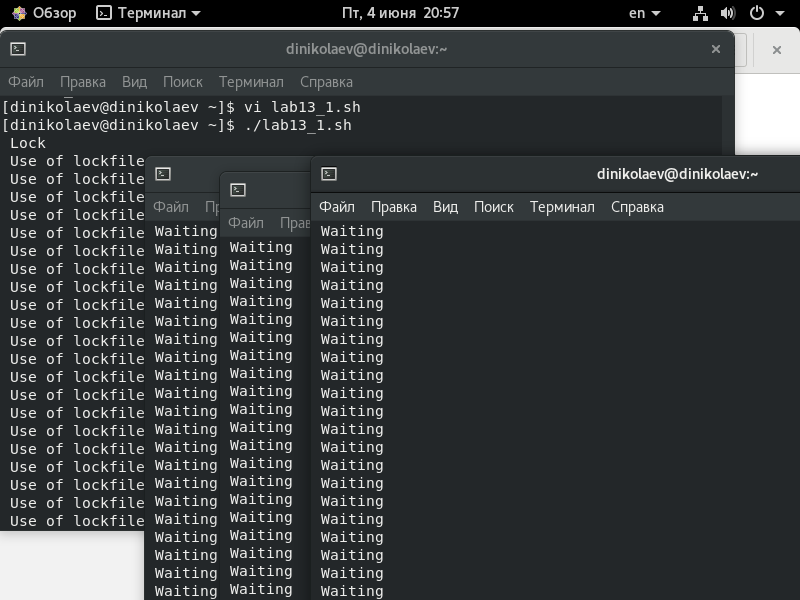
# Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

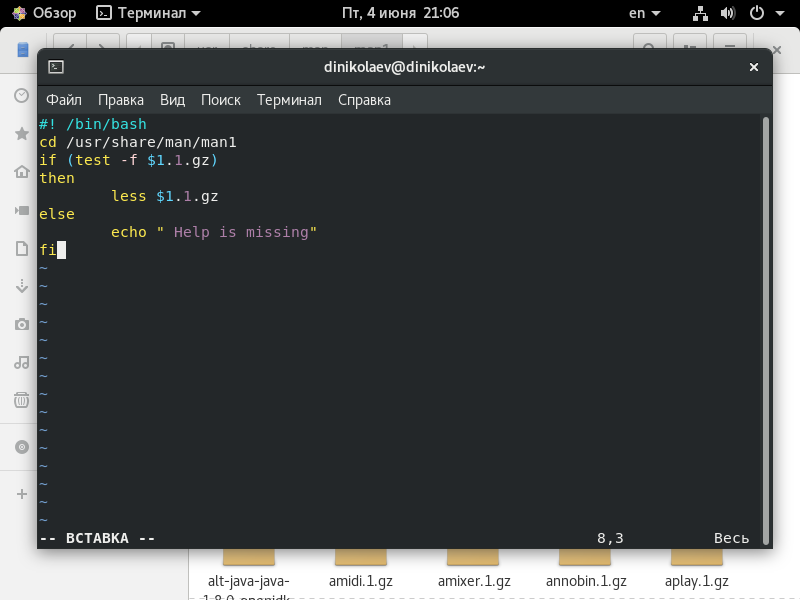
# Выполнение лабораторной работы

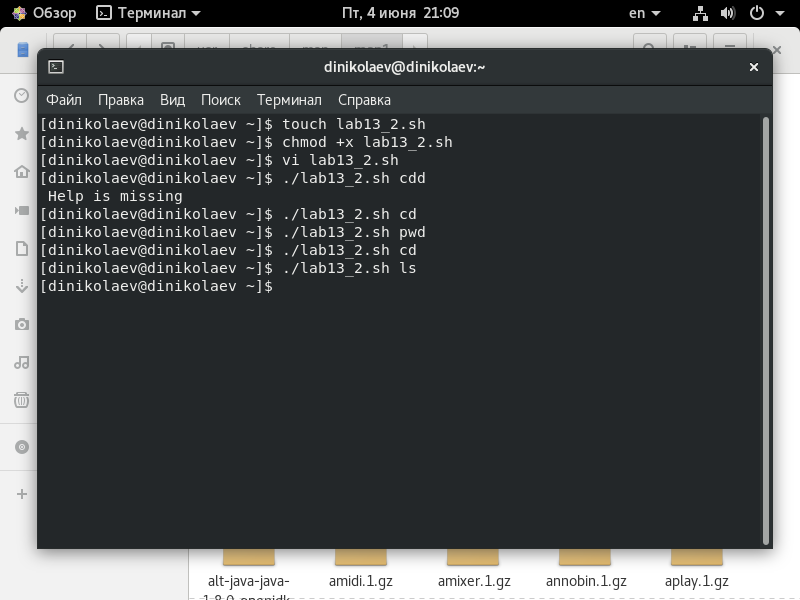
1. Написал командный файл, который реализует упрощённый механизм семафоров для доступа к необходимому ресурсу.

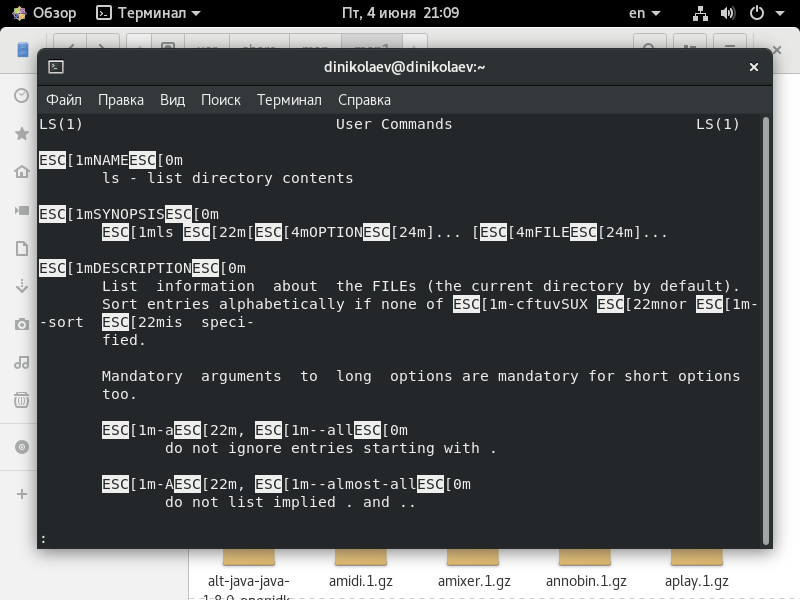
 - Командый файл, реализующий механизм семафоров

 - Результат использования командного файла

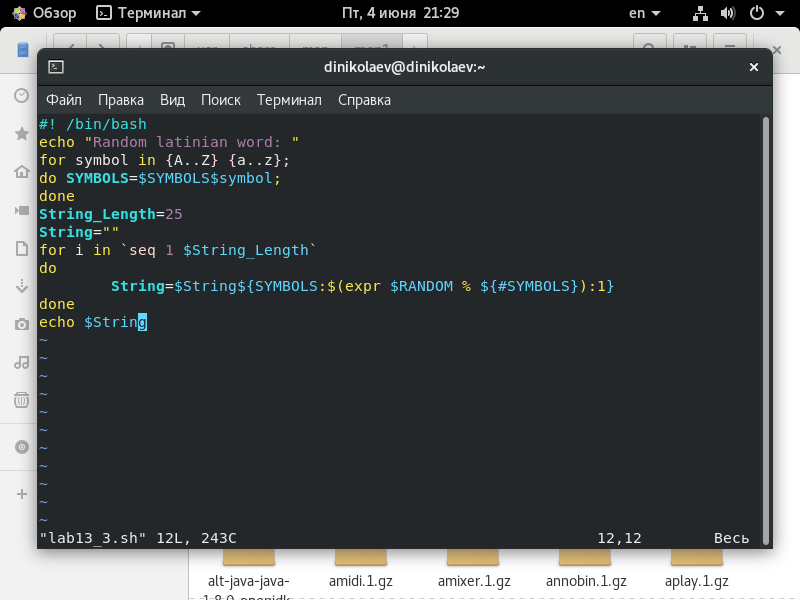
1. Реализовал аналог команды man с помощью командного файла (через каталог /usr/share/man/man1), если введённой команды нет, то файл выводит сообщение об её отсутствии.

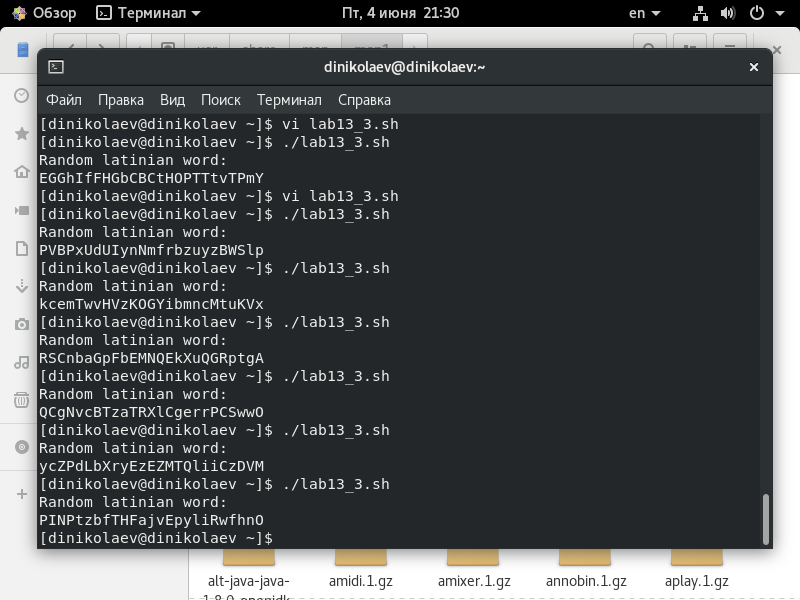
 - Командный файл аналог команды man

 - Вывод на несуществующую команду и вызов справки по другим командам

 - Вывод справки по команде ls с помощью командного файла

1. Написал командный файл, создающий строку длиной 25 со случайной последовательностью букв латинского алфавита (Различая заглавные и строчные буквы), при этом используя встроенную переменную $RANDOM.

 - Командный файл, выводящий случайную последовательность латинских букв

 - Вывод командного файла

## Контрольные вопросы

1. В этой строке надо квадратные скобки заменить на круглые ( while ($1 != “exit”) ).
2. Можно объединить с помощью знаков “>”, “|” или используя подобные выражения “expr3=expr2” (Использовалось в командном файле с рандомными числами).
3. Команда “seq” выводит последовательность целых или действительных чисел с указанным шагом, можно использовать с циклом for, подставляя команды.
4. Результатом будет число 3 (дробная часть отбрасывается).
5. В zsh возможно свободно настраивать сочетания клавиш; встроенная команда zmv позволяет массово переименововать файлы или директории; в zsh поддерживаются числа с плавающей точкой; в zsh нумерация начинается с 1.
6. Да, синтаксис этой конструкции верен (“for((a=1;a <= LIMIT;a++))”).
7. Перечислим некоторые отличия:
   * Скорость работы программ на ассемблере на 50% медленее оптимизированных на C/C++;
   * Скорость работы виртуальной ява-машины с байт-кодом обычно превосходит скорость аппаратуры с кодами (уступает только ассемблеру и лучшим оптимизирующим трансляторам), получаемыми трансляторами с языков высокого уровня;
   * Скорость компиляции и исполнения программ на яваскрипт в популярных браузерах лишь в 2-3 раза уступает лучшим трансляторам и даже превосходит некоторые качественные компиляторы (намного при этом обгоняя в скорости исполнения программ);
   * Скорость кодов, генерируемых компилятором C оказалась меньшей, чем у GNU;
   * Стек большинства тестируемых языков поддерживает только очень ограниченное число рекурсивных вызовов, некоторые же трансляторы позволяют увеличить размер стека;
   * В версиях gawk, php, perl и bash реализован динамических стек, который позволяет использовать всю ОП компьютера. Но bash использует стек слишком экстенсивно.

# Выводы

Закрепил основы программирования в оболочке ОС UNIX/Linux, научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.