

Цель работы

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

Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-i` — прочитать данные из указанного файла; `-o` — вывести данные в указанный файл; `-r` — указать шаблон для поиска; `-C` — различать большие и малые буквы; `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-p`.
2. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции `exit(n)`, передавая информацию в код завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например `1.tmp`, `2.tmp`, `3.tmp`, `4.tmp` и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).
4. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`).

Выполнение лабораторной работы

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-i` — прочитать данные из указанного файла; `-o` — вывести данные в указанный файл; `-r` — указать шаблон для поиска; `-C` — различать большие и малые буквы; `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-p`.

```
artemagarkov@fedora:~ — /usr/libexec/vi search_s
#!/bin/bash

# Переменные для хранения значений по умолчанию
input_file=""
output_file=""
pattern=""
case_sensitive=false
line_numbers=false

# Обработка опций с помощью getopts
```

```

while getopts ":i:o:p:Cn" opt; do
  case $opt in
    i)
      input_file=$OPTARG
      ;;
    o)
      output_file=$OPTARG
      ;;
    p)
      pattern=$OPTARG
      ;;
    C)
      case_sensitive=true
      ;;
    n)
      line_numbers=true
      ;;
    \?)
      echo "Invalid option: -$OPTARG" >&2
      ;;
    esac
  done

  # Переменная для хранения команды поиска
  search_command="grep"

  # Добавляем опцию для различения регистра, если указано
  if [ "$case_sensitive" = true ]; then
    search_command+=" -E"
  fi

  # Добавляем опцию для вывода номеров строк, если указано
  if [ "$line_numbers" = true ]; then
    search_command+=" -n"
  fi

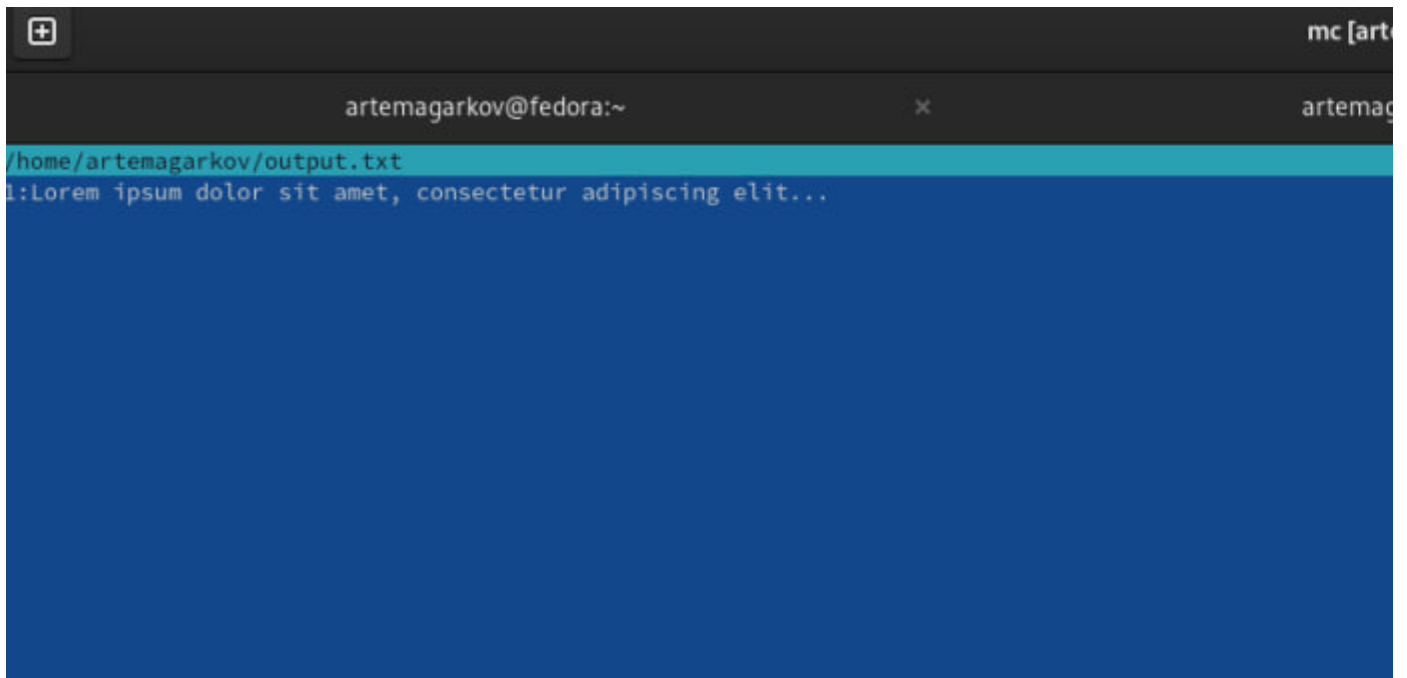
  # Добавляем шаблон для поиска
  search_command+=" \"\$pattern\""

  # Проверяем, указан ли файл для чтения
  if [ -n "$input_file" ]; then
    search_command+=" \"\$input_file\""
  else
    echo "Input file is not specified"
    exit 1
  fi

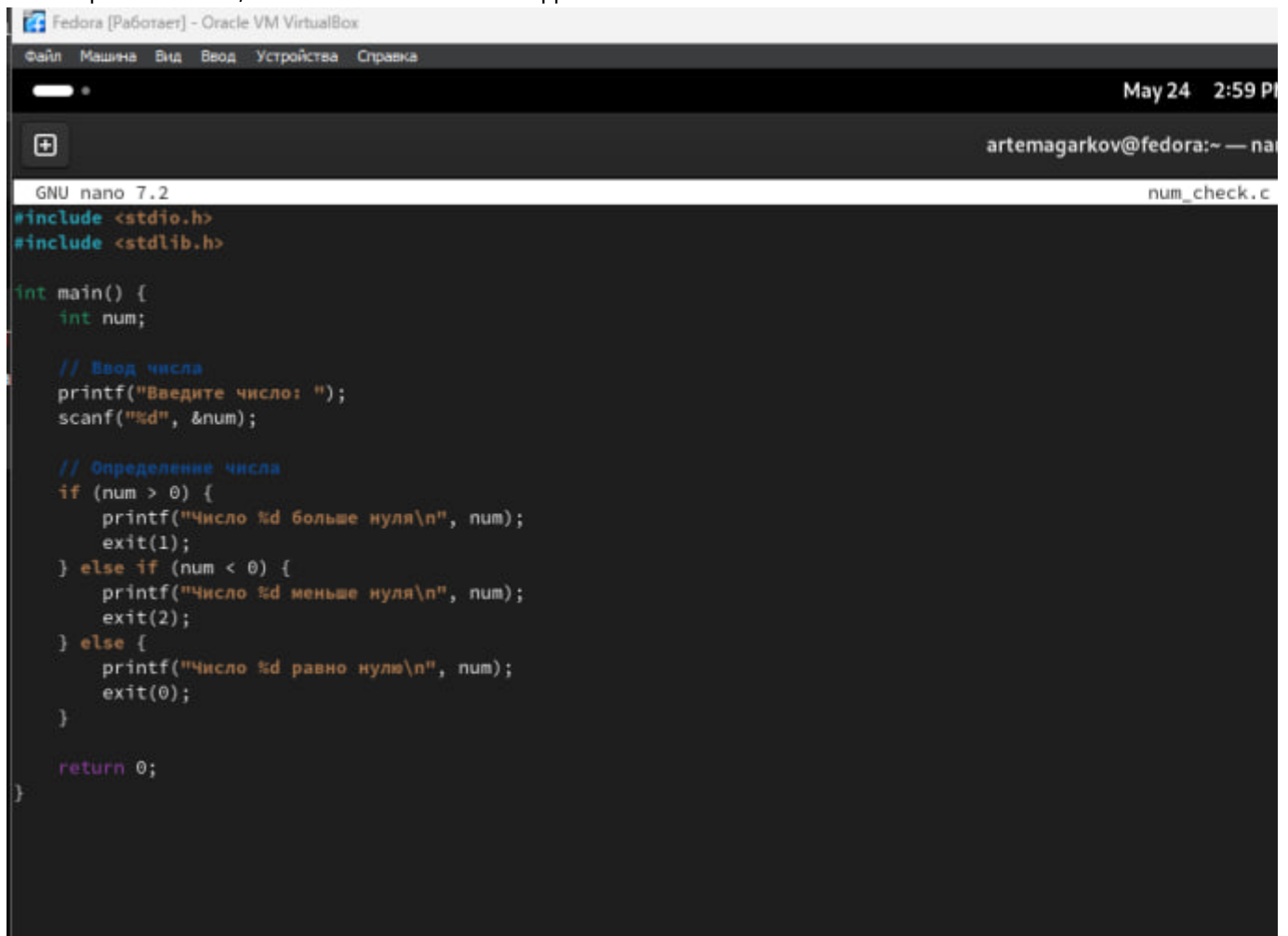
  # Проверяем, указан ли файл для записи
  if [ -n "$output_file" ]; then
    search_command+=" > \"\$output_file\""
  fi

  # Выполняем команду поиска
  eval $search_command

```



2. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции `exit(n)`, передавая информацию в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.



```
artemagarkov@fedora:~ — nano num_check.c
GNU nano 7.2
#!/bin/bash

# Вызов программы num_check и передача в нее аргумента
./num_check $1

# Проверка кода состояния
if [ $? -eq 0 ]; then
    echo "Был введен ноль"
elif [ $? -eq 1 ]; then
    echo "Было введено положительное число"
else
    echo "Было введено отрицательное число"
fi

artemagarkov@fedora:~$ ./run_num_check.sh 34123412
./run_num_check.sh: line 4: ./num_check: No such file or directory
Было введено положительное число
artemagarkov@fedora:~$
```

3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).



```
#!/bin/bash

# Проверяем количество аргументов
if [ "$#" -lt 2 ]; then
    echo "Usage: $0 <create|delete> <number_of_files>"
    exit 1
fi

action=$1
num_files=$2

if [ "$action" == "create" ]; then
    # Создаем указанное количество файлов
    for ((i = 1; i <= num_files; i++)); do
        touch "$i.tmp"
    done
    echo "Created $num_files files."
elif [ "$action" == "delete" ]; then
    # Удаляем файлы, если они существуют
    for ((i = 1; i <= num_files; i++)); do
        if [ -e "$i.tmp" ]; then
            rm "$i.tmp"
        fi
    done
    echo "Deleted $num_files files."
else
    echo "Invalid action: $action"
    echo "Usage: $0 <create|delete> <number_of_files>"
    exit 1
fi
```

^G Help
^X Exit

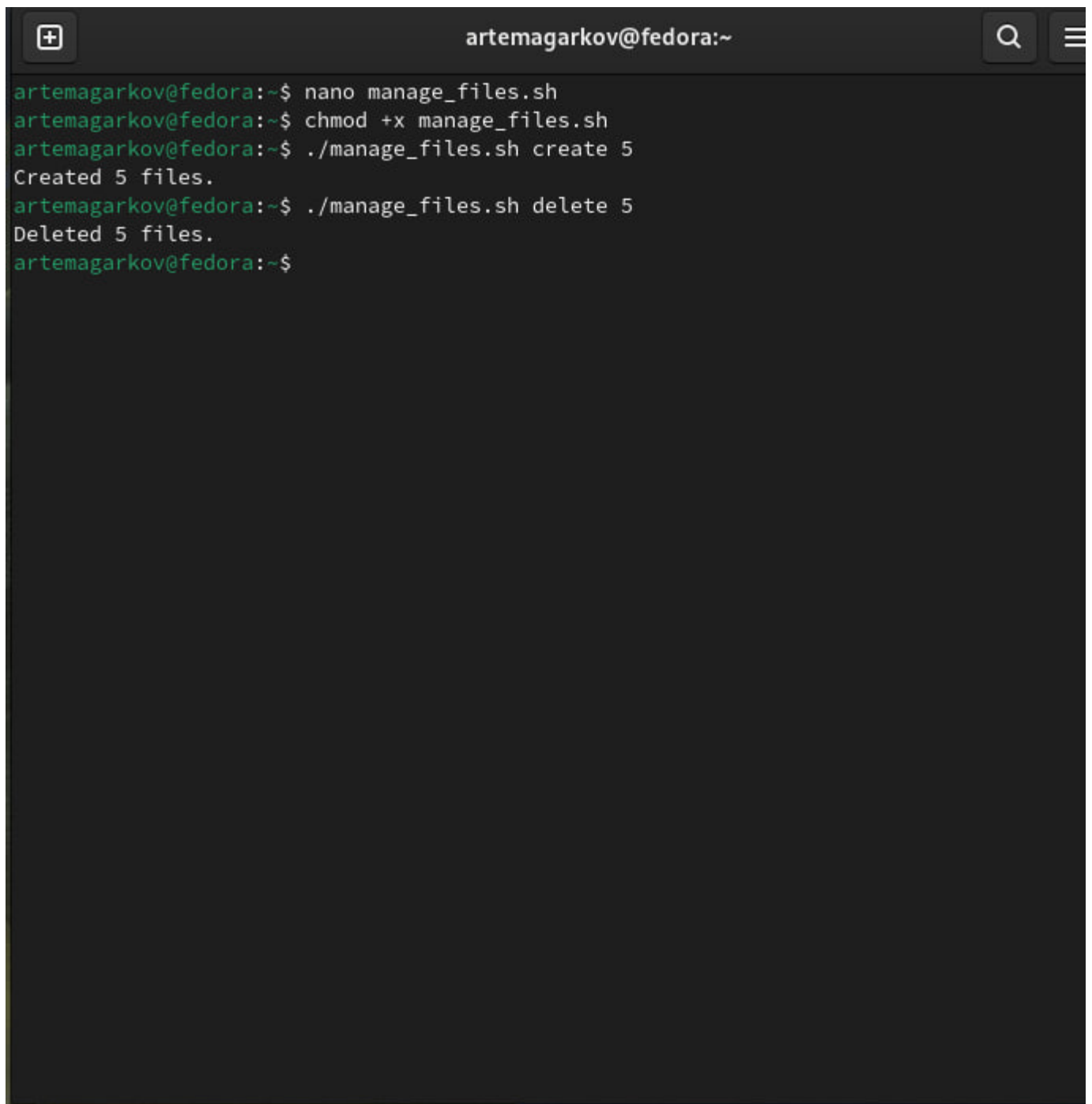
^O Write Out
^R Read File

^W Where Is
^_ Replace

^K Cut
^U Paste

^T Execute
^J Justify

^C Location
^/ Go To L

A terminal window titled 'artemagarkov@fedora:~' with a search icon and a menu icon in the top right. The terminal shows a series of commands and their outputs: 'nano manage_files.sh' is executed, followed by 'chmod +x manage_files.sh'. Then, './manage_files.sh create 5' is run, resulting in 'Created 5 files.'. Next, './manage_files.sh delete 5' is run, resulting in 'Deleted 5 files.'. The prompt returns to 'artemagarkov@fedora:~\$'.

4. Написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find).

```
artemagarkov@fedora:~ — nano backup_files.sh
GNU nano 7.2 backup_files.sh Modif
#!/bin/bash

# Проверяем количество аргументов
if [ "$#" -ne 1 ]; then
    echo "Usage: $0 <directory>"
    exit 1
fi

directory=$1
archive_name="backup_$(date +%Y%m%d%H%M%S).tar.gz"

# Запаковываем все файлы в указанной директории
tar -czf "$archive_name" -C "$directory" .

echo "Archive of all files created: $archive_name"

# Запаковываем только те файлы, которые были изменены менее недели тому назад
recent_archive_name="recent_backup_$(date +%Y%m%d%H%M%S).tar.gz"
find "$directory" -type f -mtime -7 | tar -czf "$recent_archive_name" -T -

echo "Archive of recent files created: $recent_archive_name"

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Li
artemagarkov@fedora:~$ ./backup_files.sh ~/Music
Archive of all files created: backup_20240525153239.tar.gz
Archive of recent files created: recent_backup_20240525153239.tar.gz
artemagarkov@fedora:~$
```

11.4. Контрольные вопросы и ответы

1. Каково предназначение команды getopt?

- getopt используется для разбора позиционных параметров в командных файлах (скриптах)

и обработки опций командной строки.

2. Какое отношение метасимволы имеют к генерации имён файлов?

- Метасимволы (например, *, ?, []) используются для шаблонного поиска файлов в командной строке, расширяя шаблоны в списки файлов, соответствующих этим шаблонам.

3. Какие операторы управления действиями вы знаете?

- Операторы управления действиями включают if, else, elif, case, for, while, until, а также операторы break и continue.

4. Какие операторы используются для прерывания цикла?

- Операторы break (для выхода из цикла) и continue (для перехода к следующей итерации цикла).

5. Для чего нужны команды false и true?

- false возвращает ненулевой статус выхода (обычно 1), означающий ложь или неуспех, а true возвращает нулевой статус выхода, означающий истину или успех. Они используются для управления потоками выполнения в скриптах.

6. Что означает строка if test -f man\$/i.\$s, встреченная в командном файле?

- Эта строка проверяет, существует ли файл с именем man\$/i.\$s (-f проверяет существование файла). Если файл существует, условие if выполняется.

7. Объясните различия между конструкциями while и until.

- while выполняет блок команд, пока условие истинно (true).
- until выполняет блок команд, пока условие ложно (false).

Выводы

Я познакомился с основами программирования в оболочке ОС UNIX и научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

