БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра технологий программирования

#### ОТЧЕТ

#### по лабораторной работе №2

По курсу “Программирование мобильных и встраиваемых систем”

***Командная оболочка bash: перенаправление ввода-вывода, сценарии и другие возможности***

**Подготовил:**  
студент 3 курса 13 группы кафедры ТП  
Петров Андрей Александрович

## Минск, 2021

## ЦЕЛИ РАБОТЫ:

## Изучить синтаксис команд перенаправления ввода-вывода, утилиту awk и ее функциональные возможности и синтаксис, потоковый редактор sed и его возможности, изучить язык сценариев bash

## ЗАДАНИЕ 2.1. ПЕРЕНАПРАВЛЕНИЕ ВВОДА-ВЫВОДА

Изучите примеры задания 2.1 и выполните их в ОС Ubuntu и Raspberry PI (ранее Raspbian).

Включите ведение протокола командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — taskXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu. **Журнал меток** назвать по следующему шаблону — timelogXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu.



Примеры к заданию 2.1

**Выполнить примеры ниже в Raspberry PI**

1. Создайте в текстовом редакторе два файла для дальнейшего использования в лабораторной работе. Файл packages1.txt должен содержать следующих восемь строк:

*amanda galeon metacity mozilla postgresql procinfo rpmfind squid*

Файл packages2.txt должен содержать следующих шесть строк:

*amanda galeon metacity mozilla postgresql procinfo rpmfind squid*



1. Для просмотра содержимого файлов предназначена команда cat. В простейшем случае эта команда принимает ввод из файла или стандартного канала ввода STDIN и посылает его в стандартный канал вывода STDOUT.

*$ cat packages1.txt*

Изображение выглядит как текст

Автоматически созданное описание

1. Если команда cat не имеет аргументов, то ожидается, что она получает данные из стандартного канала ввода STDIN. Команда читает данные из канала стандартного ввода до нажатия на клавиатуре клавиш <Ctrl+d>:

*$ cat*

*Type some sample text, then press return.*

*<Ctrl+d>*



Изображение выглядит как текст

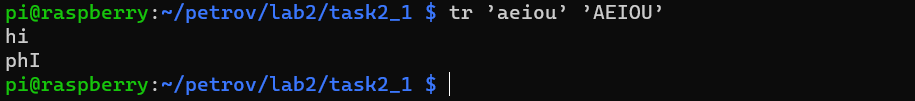
Автоматически созданное описание

1. Большинство команд Linux работают как фильтры, т. е. Получают исходные данные из STDIN, делают обработку этих данных и выводят результаты в STDOUT. Например, команда tr (translate):

*$ tr ’aeiou’ ’AEIOU’*

*Type some sample text, then press return.*

*<Ctrl+d>*



1. Используя директиву ’>’, перенаправим стандартный вывод из одного файла в другой файл:

*$ cat packages1.txt > packages1.catfile*

*$ cat packages1.catfile*

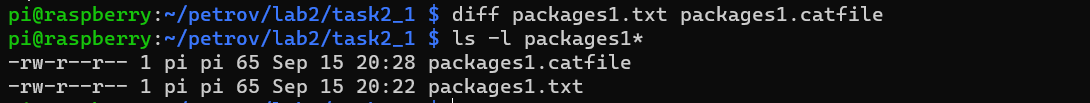
Изображение выглядит как текст

Автоматически созданное описание

Затем сверим исходный и полученный файлы:

*$ diff packages1.txt packages1.catfile*

*$ ls -l packages1\**



1. Для присоединения существующего файла к другому файлу предназначена директива ’>>’:

*$ cat packages2.txt >> packages1.catfile*

*$ cat packages1.catfile*

##### 

##### Подключиться по ssh из Raspberry PI и выполнить примеры ниже в Ubuntu.

##### 

##### 

1. Если команде cat не передается аргумент и стандартный ввод перенаправлен в файл, то весь ввод с клавиатуры до нажатия клавиш <Ctrl+d> будет перенаправлен в файл.

*$ cat > typedin.txt*

*This time, when text is typed at the keyboard, it is redirected to the file typedin.txt.*

*<Ctrl+d>*

*$ ls -l typedin.txt*

*$ cat typedin.txt*

Изображение выглядит как текст, экран, снимок экрана

Автоматически созданное описание

1. Повторите предыдущий шаг, подставив вместо команды cat команду tr.

*$ tr ’aeiou’ ’AEIOU’ > trfile.txt*

*This time, when text is typed at the keyboard,*

*it is not echoed back to the screen with the translations made.*

*Instead, it is redirected to the file trfile.txt.*

*<Ctrl+d>*

*$ ls -l trfile.txt*

*$ cat trfile.txt*

Изображение выглядит как текст

Автоматически созданное описание

1. Команда cat принимает в качестве аргумента имя файла или стандартный ввод, перенаправленный из файла. Проверьте это при помощи следующих двух команд:

*$ cat packages1.txt*

*$ cat < packages1.txt*

Изображение выглядит как текст

Автоматически созданное описание

1. Однако команда tr принимает ввод только из стандартного канала.

*$ tr ’aeiou’ ’AEIOU’ < packages1.txt*

Изображение выглядит как текст

Автоматически созданное описание

1. В следующем примере стандартный ввод и вывод одновременно перенаправляются

*$ tr ’aeiou’ ’AEIOU’ < packages1.txt > packages1.trfile.txt*

*$ ls -l packages1.txt packages1.trfile.txt*

*$ cat packages1.trfile.txt*

Изображение выглядит как текст

Автоматически созданное описание

## ЗАДАНИЕ 2.2. УТИЛИТА AWK ПРИМЕРЫ ДЛЯ ИЗУЧЕНИЯ

Изучите примеры задания 2.2 и выполните их в ОС Ubuntu и Raspberry PI. **Выполнить примеры 1-4 в Raspberry PI, подключиться по ssh к Ubuntu и выполнить примеры 5-14.**

Включите ведение протокола командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — taskXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu. **Журнал меток** назвать по следующему шаблону — timelogXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu.



### Примеры к заданию 2.2

#### Пример 1.

Предварительно создадим файл для обработки. Для этого выполним следующие действия:

* 1. Создаем каталог examples в домашнем каталоге пользователя.

*$ mkdir examples*



* 1. Убедитесь, что Вы находитесь в домашнем каталоге, выполнив команду

*$ pwd*

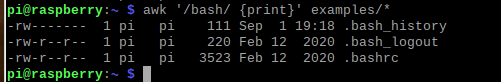


* 1. Перенаправим вывод команды ls расширенном формате в файл, например log.txt

*$ ls -la > examples/log.txt*



* 1. И выполним команду для поиска данных в созданном файле или наборе файлов. Будем искать строки, содержащие слово «bash»

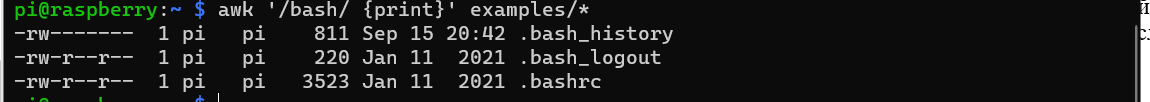
*$ awk '/bash/ {print}' examples/\**

Ищутся все файлы в каталоге ***examples***, который находится в домашнем каталоге пользователя и в которых встречается слово ***bash****.* Результаты направляются на стандартное устройство вывода — действие ***{print}***. В результатах выводятся строки, содержащие шаблон.

Действие направления на стандартное устройство вывода ***{print}*** обычно задается как действие по умолчанию, поэтому команду примера можно переписать так:

$ awk '/config/' examples/\*

Результат при этом не изменится.



#### Пример 2.

Найти по указанному пути файлы исходных текстов на языке С, включающие слова, например, QUIT или SETDATE:

*$ awk '/QUIT|SETDATE/' /usr/include/protocols/\**

Изображение выглядит как текст

Автоматически созданное описание

#### Пример 3.

1. Создайте в каталоге **examples** текстовый файл **myfile** следующего содержания:

This is a test.

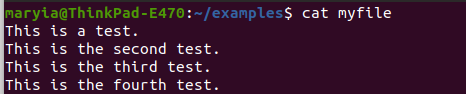
This is the second test. This is the third test. This is the fourth test.

Изображение выглядит как текст

Автоматически созданное описание

1. Просмотрите файл:

$ cat myfile



Одна из основных функций awk заключается в возможности манипулировать данными в текстовых файлах. Делается это путём автоматического назначения переменной каждому элементу в строке. По умолчанию awk назначает следующие переменные каждому полю данных, обнаруженному им в записи:

* + $0 — представляет всю строку текста (запись).
  + $1 — первое поле.
  + $2 — второе поле.
  + $n — n-ное поле.

Изображение выглядит как текст

Автоматически созданное описание

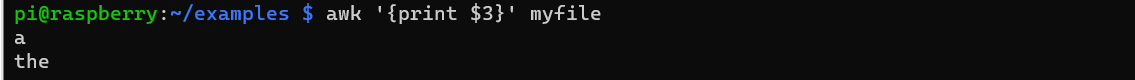
1. Выполните вывод первого элемента каждой строки:

*$ awk '{print $1}' myfile*



1. Выполните вывод третьего элемента каждой строки:

*$ awk '{print $3}' myfile*



#### Пример 4.

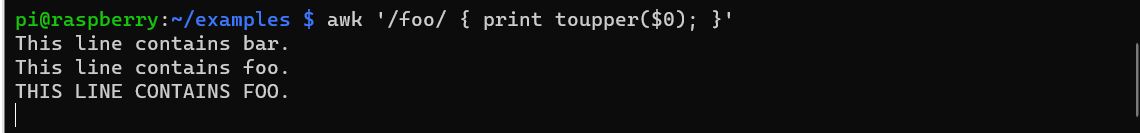
1. **awk** позволяет изменять регистр символов. В примере ниже необходимо ввести команду и две строки текста, одна из которых будет преобразована в строку в верхнем регистре:

*$ awk '/foo/ { print toupper($0); }'*

*This line contains bar.*

*This line contains foo.*

*THIS LINE CONTAINS FOO.*



1. Для завершения ввода нажмите комбинацию клавиш Ctrl+C.

Изображение выглядит как текст

Автоматически созданное описание

#### Пример 5.

* 1. Создать в каталоге **examples** текстовый файл list\_students, отсортированный по алфавиту, каждая строка которого включает: фамилию, имя, факультет, курс, рейтинговая оценка, и состоящий из следующих строк- записей:

Асташко Иван ПИ 1 6

Бузун Евгений КБ 1 9

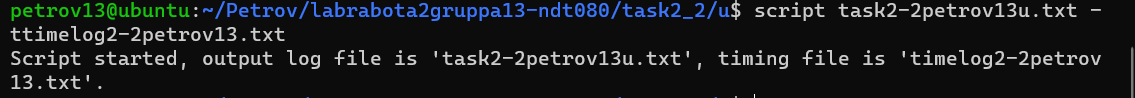
Кравченя Леонид ПМ 1 4

Кошкин Владимир ИН 1 7

Липин Федор ПИ 2 4

Пенов Николай ПМ 1 5

Яшин Петр КБ 4 8



Изображение выглядит как текст

Автоматически созданное описание

* 1. Вывести список, состоящий из фамилий, имен и рейтинговых оценок:

*$ awk '{print $1,$2,$5}' list\_students*

Изображение выглядит как текст

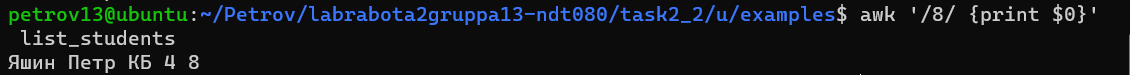
Автоматически созданное описание

#### Пример 6.

Выбрать из исходного файла *list\_students* студентов с рейтинговой оценкой 8 и распечатать все значения полей для выбранных записей:

*$ awk '/8/ {print $0}' list\_students*

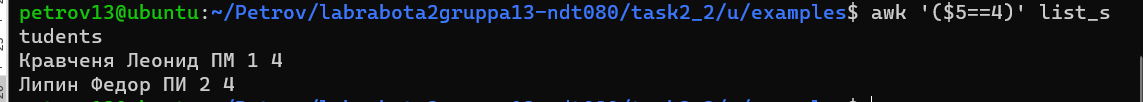
В случае, если оценка встречается в других полях, например, курс 4 и оценка 4, то условия выборки недостаточны.



#### Пример 7.

Выбрать из списка студентов, рейтинговая оценка которых 4:

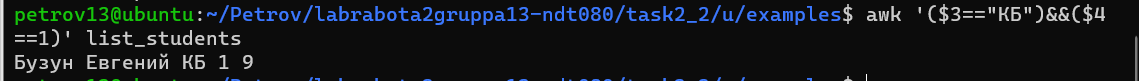
*$ awk '($5==4)' list\_students*



#### Пример 8.

Выбрать из списка студентов специальности КБ, 1 курса:

*$ awk '($3=="КБ")&&($4==1)' list\_students*



#### Пример 9.

Выбрать из файла со списком студентов все имена с длиной >5

*$ awk '(length($2)>5) {print}' list\_students*

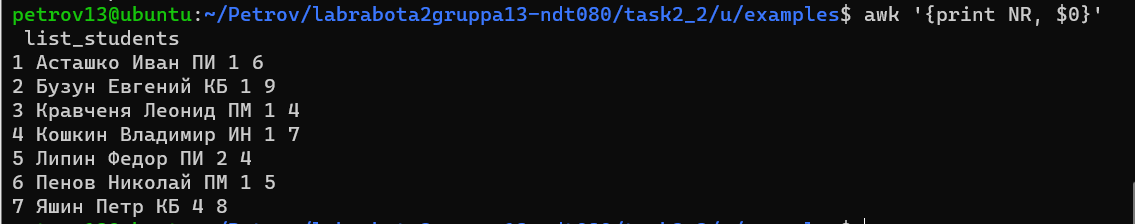
Изображение выглядит как текст

Автоматически созданное описание

#### Пример 10.

Вывести данные исходного файла в виде таблицы и пронумеровать строки:

*$ awk '{print NR, $0}' list\_students*



#### Пример 11.

Вывести все имена студентов в верхнем регистре:

*$ awk '{print toupper ($2)}' list\_students*

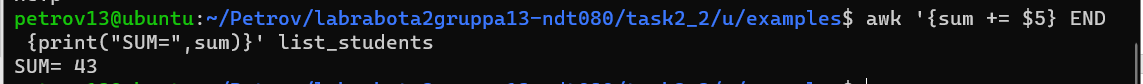
Изображение выглядит как текст

Автоматически созданное описание

#### Пример 12.

Посчитать суммарный балл оценок студентов:

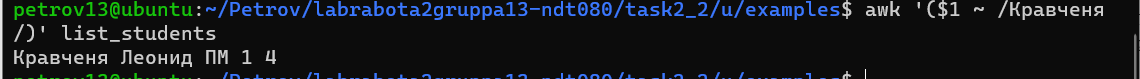
*awk '{sum += $5} END {print("SUM=",sum)}' list\_students*



#### Пример 13.

Присутствует ли фамилия Леонов в списке файла ***list\_students***?

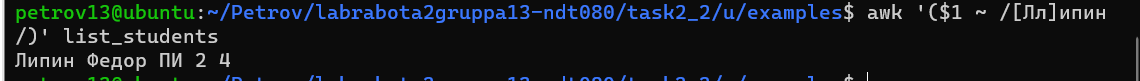
*$ awk '($1 ~ /Кравченя/)' list\_students*

******

#### Пример 14.

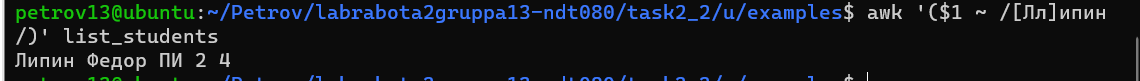
Проверить, есть ли запись студента Липина в списке файла ***list\_students****,* причем фамилия может быть написана с прописной или строчной буквы:

*$ awk '($1 ~ /[Лл]ипин/)' list\_students*

**

Вывести все записи кроме студента Липин:

*$ awk '($1 ~ /[Лл]ипин/)' list\_students*

**

#### Пример 15.

1. Создайте текстовый файл ***colours.csv*** вида:

*name,color,amount*

*apple,red,4*

*banana,yellow,6*

*strawberry,red,3*

*grape,purple,10*

*apple,green,8*

*plum,purple,2*

*kiwi,brown,4*

*potato,brown,9*

*pineapple,yellow,5*

Изображение выглядит как текст

Автоматически созданное описание

1. Выберите все записи из файла ***colours.csv***, количество которых больше 5, выполнив команду:

*$ awk -F”,” '$3>6 {print $1, $2}' colours.csv > output.txt*

В команде выше явно был указан разделитель данных, т.е. “,”. В результате выполнения команды будет создан файл, содержащий записи согласно условию.



## ЗАДАНИЕ 2.3. ОБРАБОТКА СТРОК ФАЙЛА С ПРИМЕНЕНИ-ЕМ ПОТОКОВОГО РЕДАКТОРА SED

Изучите примеры задания 2.3 и выполните их в ОС Ubuntu и Raspberry PI. **Выполнить примеры 1-4 в Raspberry PI, скопировать файл books в Ubuntu, подключиться по ssh к Ubuntu и выполнить примеры 5-8.**

Включите ведение протокола командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — taskXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu. **Журнал меток** назвать по следующему шаблону — timelogXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu.

### Примеры к заданию 2.3

1. Сохраните данные ниже в файл с именем books:

*Book one.*

*The second book. The third.*

*This is book four. Five.*

*This is the sixth. This is book seven. Eighth and last.*

Изображение выглядит как текст

Автоматически созданное описание

1. Чтобы вывести все строки файла и продублировать строки согласно шаблону, т. е. содержащие слово book выполните команду

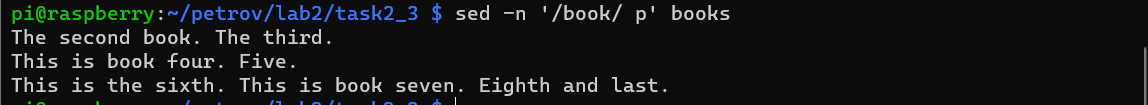
*$ sed '/book/ p' books*

*Изображение выглядит как текст

Автоматически созданное описание*

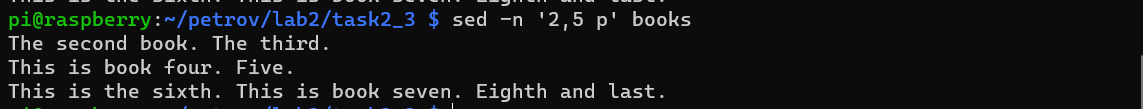
1. Чтобы выбрать определенные строки, например содержащие слово book, выполните команду

$ *sed -n '/book/ p' books*



1. Чтобы вывести часть файла, например, строки с 2 по 5, выполните команду

*$ sed -n '2,5 p' books*



Копируем файл books на OS Ubuntu и подключаемся к ней по ssh.

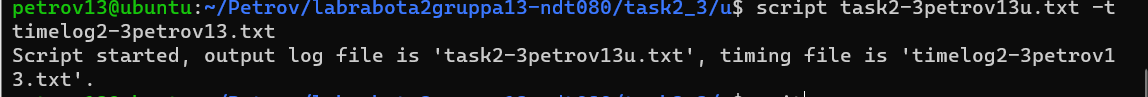
Изображение выглядит как текст

Автоматически созданное описание



Изображение выглядит как текст

Автоматически созданное описание



1. Для выполнения более сложных и длинных инструкций можем использовать файл программы для sed. Выполним команду из примера 4, указав параметры в файле records.

Содержимое файла records:

2,5 p

Пример команды из примера 4 с выполнением инструкций из файла:

*$ sed -n -f records books*



1. В данном примере выбираем строку 3 и используем инструкцию Добавить, чтобы добавить разделитель строк и текст «My favorite book.» к третьей строке:

Содержимое файла appends

3 a\

My favorite book.

Пример команды с добавлением строки:

*$ sed -f appends books*

Изображение выглядит как текст

Автоматически созданное описание

1. В данном примере требуется вставить разделитель строк и текст «SKARBONKA.» перед строками, в которых содержится слово «This»: Содержимое файла insert

/This/ i\ SKARBONKA.

Пример команды с со вставкой строк:

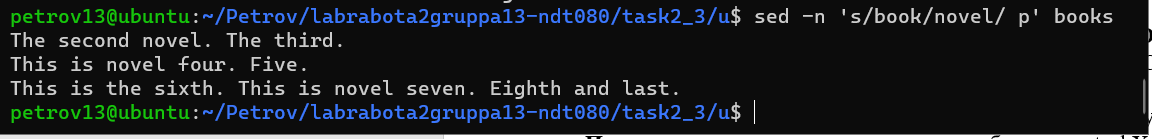
*$ sed -f insert books*

Изображение выглядит как текст

Автоматически созданное описание

1. В следующем примере выполняется замена слова book на novel

*$ sed -n 's/book/novel/ p' books*



## ЗАДАНИЕ 2.4. ИСПОЛНИТЕЛЬНАЯ СРЕДА ОБОЛОЧКИ BASH

Изучите примеры задания 2.4 и выполните их в ОС Ubuntu и Raspberry PI.

Имена скриптов могут быть любыми.

Включите ведение протокола командой script с журналом меток времени. **Протокол** назвать по следующему шаблону — taskXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu. **Журнал меток** назвать по следующему шаблону — timelogXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu.

### ПРИМЕРЫ ДЛЯ ИЗУЧЕНИЯ: ИСПОЛНИТЕЛЬНАЯ СРЕДА ОБОЛОЧКИ

##### Выполнить примеры 1-3 в Raspberry PI, подключиться по ssh к Ubuntu и выполнить примеры 4-12.

##### 

1. Войдите в систему под своей учетной записью.
2. Выполните следующие команды, которые используют переменные a, full\_name и short\_name командной оболочки Bash.

*$ a=879*

*$ echo "The value of \"a\" is $a."*

*$ export full\_name="John Lee"*

*$ echo $full\_name*

*$ export short\_name="John"*

*$ export short\_name*

*$ echo $ short\_name*

Изображение выглядит как текст

Автоматически созданное описание

1. Выполните следующие команды, которые иллюстрируют операции над переменными

*$ export foo=""*

*$ echo ${foo:-one}*

*$ echo $foo*

*$ echo ${foo:=one}*

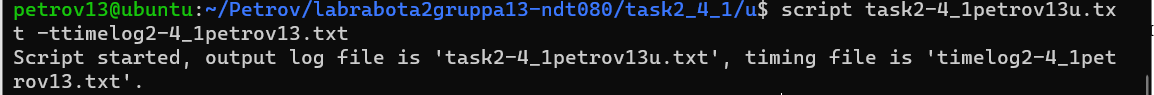
*$ export foo="this is a test"*

*$ echo $foo*

*$ echo ${foo:+bar}*

Изображение выглядит как текст

Автоматически созданное описание



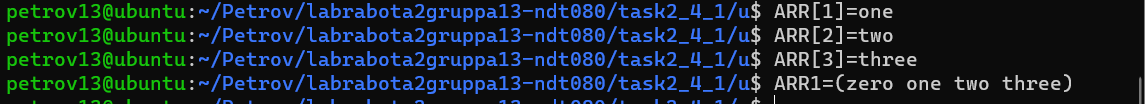
1. Выполнить следующие команды, которые создают переменные массивы.

*$ ARR[1]=one*

*$ ARR[2]=two*

*$ ARR[3]=three*

*$ ARR1=(zero one two three)*



1. Выполнить следующую команду, которая выводит на консоль значения элементов массива.

*$ echo ${ARR[0]} ${ARR[1]}*



1. Выполните следующие команды для удаления элемента ARR[1] массива ARR и для удаления всего массива ARR:

*$ unset ARR[1]*

*$ unset ARR*



1. Создайте псевдоним для команды clear, выполнив следующую команду:

*$ alias c=’clear’*

Просмотрите созданные псевдонимы.

*$ alias*

*$ c*

Изображение выглядит как текст

Автоматически созданное описание

1. Этот псевдоним будет утерян, если вы выйдите из командной оболочки, а затем войдете в нее вновь. Чтобы обеспечить сохранность псевдонима при каждом входе в командную оболочку пользователя student, нужно выполнить следующие действия. Откройте файл .bashrc в текстовом редакторе, например vim

*$ vim .bashrc*

*Изображение выглядит как текст, монитор, экран, черный

Автоматически созданное описание*

Найдите в файле bashrc строку, которая содержит текст:

*# User specific aliases and functions.*

Добавьте после этой строки следующую строку:

*alias c=’clear’*

Сохраните файл и выйдите из текстового редактора.



1. Чтобы проверить сохранение псевдонима, сначала выйдите из командной оболочки, а затем снова войдите в неё под именем student и выполните команду

*$ alias*

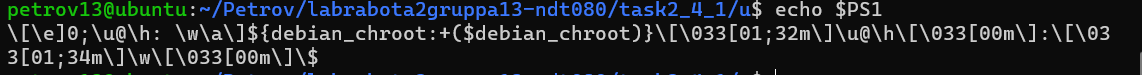
*$ c*

Изображение выглядит как текст

Автоматически созданное описание

1. Отобразите в терминале текущее значение вашей строки приглашения, выполнив команду

*$ echo $PS1*



1. Измените вашу строку приглашения, выполнив следующую команду:

*$ PS1=’Linux -> ’*



1. Восстановите традиционную строку приглашения, выполнив следующую команду:

*$ PS!=’\h $ ’*



### ПРИМЕРЫ ДЛЯ ИЗУЧЕНИЯ: ИСПОЛНЕНИЕ КОМАНД

##### Выполнить примеры 1-3 в Raspberry PI, подключиться по ssh к Ubuntu и выполнить примеры 4-7.

##### 

1. Выполните следующую команду, которая иллюстрирует расширение командным интерпретатором фигурных скобок.

*$ echo sp{el,il,al}l spell spill spall*



1. Выполните следующую команду, которая иллюстрирует расширение командным интерпретатором символа тильда на домашний каталог пользователя.

*$ cat ~/message.txt*

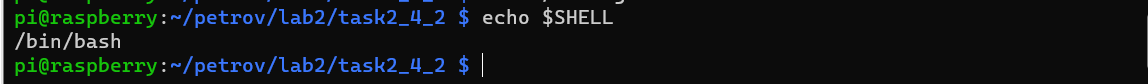
**

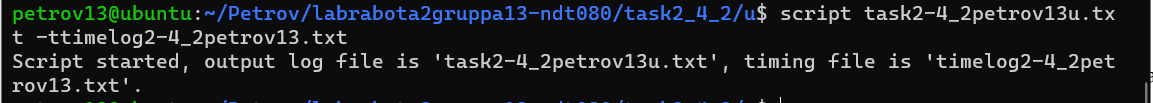
**

1. Выполните следующую команду, которая иллюстрирует расширение командным интерпретатором переменной.

*$ echo $SHELL*

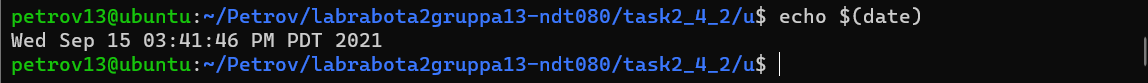
*/bin/bash*





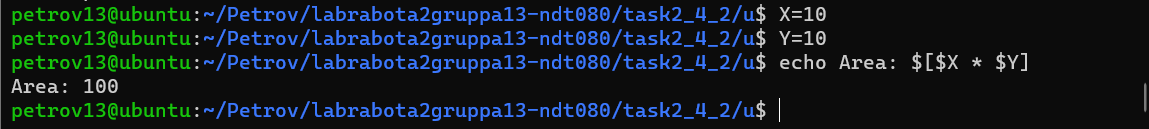
1. Выполните следующую команду, которая иллюстрирует расширение командным интерпретатором команд.

*$ echo $(date)*



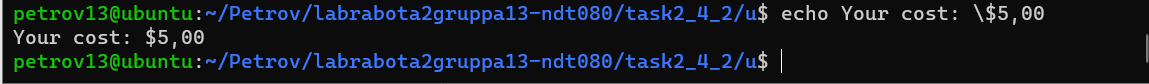
1. Объявите переменные X и Y и задайте им некоторые значения. Выполните следующую команду, которая иллюстрирует расширение командным интерпретатором арифметического выражения.

*$ echo Area: $[$X \* $Y]*



1. Выполните следующую команду, которая иллюстрирует запрещение расширения символа, который следует за символом обратный слэш.

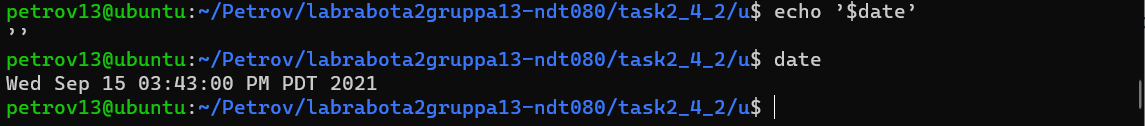
*$ echo Your cost: \$5,00*



1. Выполните следующую команду, которая иллюстрирует запрещение расширения символов, заключенных в одинарные кавычки.

$ echo ’$date’

$ date



### ПРИМЕРЫ ДЛЯ ИЗУЧЕНИЯ: СЦЕНАРИИ BASH

**Выполнить примеры 1-2 в Raspberry PI, подключиться по ssh к Ubuntu и выполнить примеры 3-12.**

****

***Пример 1.***

1. Создайте текстовый файл *ascript.sh*, который содержит следующий текст.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

1. Сохраните файл в текущем каталоге.

Изображение выглядит как текст

Автоматически созданное описание

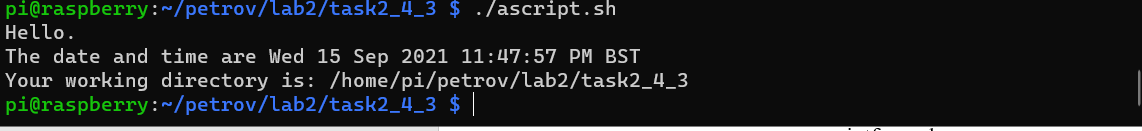
1. Сделайте файл *ascript.sh* исполняемым, выполнив команду

*# chmod u+x ascript.sh*



1. Запустите сценарий из командной строки, выполнив команду

*# ./ascript.sh*



#### Пример 2.

1. Создайте и выполните следующий скрипт, который поясняет использование команды read для ввода данных с терминала.

*#!/bin/bash*

*echo -n "Enter your name and press [ENTER]: " read var\_name*

*echo "Your name is: $var\_name"*

Изображение выглядит как текст

Автоматически созданное описание



1. Выполните следующие команды, которые иллюстрируют использование команды printf для форматированного вывода данных.

*# printf "%d\n" 5*

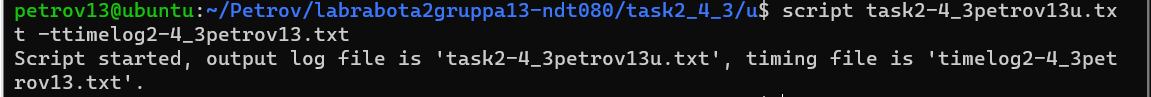
*# printf "Hello, $USER.\n\n" # distance=15*

*# printf "Distance is %5d Miles\n" $distance # printf "%d\n" 0xF*

*# printf "0x%X\n " 15*

Изображение выглядит как текст

Автоматически созданное описание



#### Пример 3.

При разработке скриптов можем создавать скрипты не только на языке bash, но и использовать инструкции awk. В пример рассматривается пример создания сценария awk.

При выполнении данного примера используйте файл ***colours.csv*** из в примера 15 задания 2.2.

* 1. Файл, содержащий awk-инструкции, может быть преобразован в файл сценария. Для этого создайте файл, например ***example.awk*** вида:

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

* 1. Назначьте права на выполнение:

*$ chmod u+x example.awk*

**

* 1. Выполните скрипт:

*$ ./example.awk colours.csv*

Изображение выглядит как текст

Автоматически созданное описание

* 1. В результате будет выведен список пронумерованных строк.

#### Пример 4.

1. Создайте и выполните следующий скрипт, который иллюстрирует использование условной инструкции if-then-else, сравнивая значения двух переменных.

#### Изображение выглядит как текст Автоматически созданное описание

#### Изображение выглядит как текст Автоматически созданное описание

#### 

#### Пример 5.

1. Создайте и выполните следующий скрипт, который объясняет использование условной инструкции case для выбора нужного варианта значения переменной.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

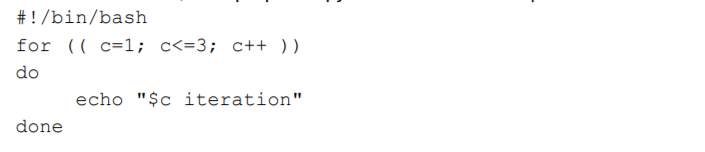
Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

#### Пример 6.

Создайте и выполните следующий скрипт, который иллюстрирует использование инструкции цикла for для числовых и символьных значений, а также показывает, как программируются бесконечные циклы for.



Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

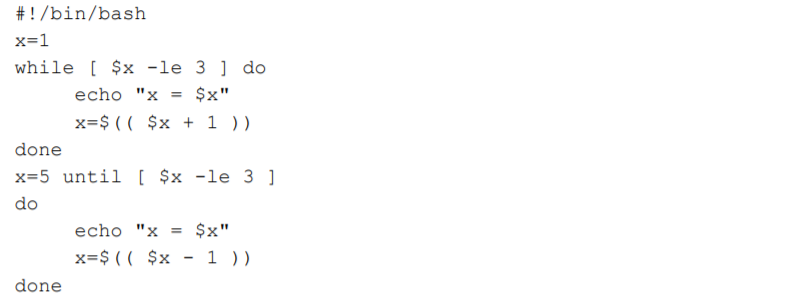
Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

#### Пример 7.

Создайте и выполните следующий скрипт, который иллюстрирует использование инструкций циклов while и until.



Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

#### Пример 8.

Создайте и выполните следующий скрипт, который иллюстрирует использование команды принудительного выхода из цикла continue.

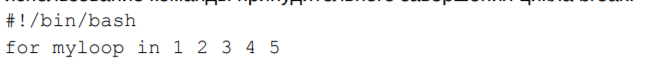
#### Изображение выглядит как текст Автоматически созданное описание

#### Изображение выглядит как текст Автоматически созданное описание

#### 

#### Пример 9.

Создайте и выполните следующий скрипт, который иллюстрирует использование команды принудительного завершения цикла break.

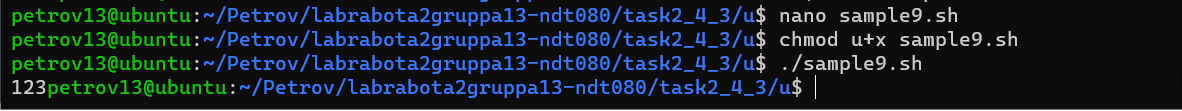


Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание



#### Пример 10.

Создайте и выполните следующий скрипт, который выводит свое имя, количество аргументов и значения первых трех аргументов, иллюстрируя тем самым обработку аргументов, которые передаются скрипту при вызове. При вызове этого скрипта передайте ему три произвольных аргумента.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

***Пример 11.***

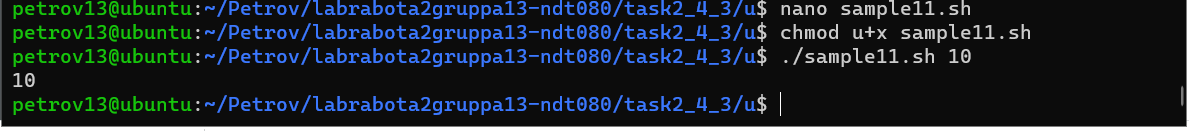
Создайте и выполните следующий скрипт, который выводит все свои аргументы, иллюстрируя тем самым обработку произвольного количества аргументов. Вызовите этот скрипт, передав ему произвольное количество аргументов.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

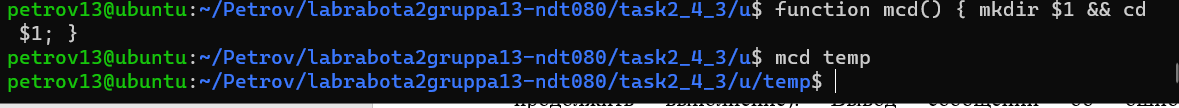
Автоматически созданное описание



#### Пример 12.

В командной строке объявите и вызовите функцию mcd, которая создает каталог и делает его текущим.





## ЗАДАНИЕ 2.5. ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Во всех заданиях должен быть контроль ошибок (если к какому-либо каталогу нет доступа, необходимо вывести соответствующее сообщение и продолжить выполнение). Вывод сообщений об ошибках должен производиться в стандартный поток вывода сообщений об ошибках (stderr) в следующем виде:

Имя\_модуля: текст\_сообщения.

Пример: ./script1 : error open file: 1.txt.

Имя модуля, имя файла берутся из аргументов командной строки.

В качестве ответа предоставить коды скриптов в репозитории и логи команды script, включая метки времени. **Протокол** назвать по следующему шаблону — taskXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu. **Журнал меток** назвать по следующему шаблону — timelogXФамилияNM, где X — номер выполняемого задания, Фамилия — заменить на вашу фамилию латиницей и строчными буквами, N — номер группы, например 12 или 13, M — **r** — для Raspberry PI, **u** — для Ubuntu.

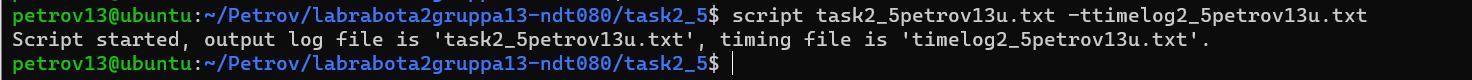
### Варианты

Для всех вариантов задач 4-5 проверить на Raspberry PI и подключившись по ssh проверить в Ubuntu, запустив их на выполнение в терминале, созданном мультиплексором терминала screen.

Для задач 4-5 проверить отложенный запуск скрипта, отсоединиться от терминала, созданного screen, подключиться к терминалу и проверить результаты работы скрипта.

Номер варианта индивидуального задания **K** равен числу букв **N1** вашей фамилии, умноженному на число букв **N2** Вашего имени (по паспорту), умноженному на число букв **N3** вашего отчества по модулю 7: k= (N1\*N2\*N3) mod 7

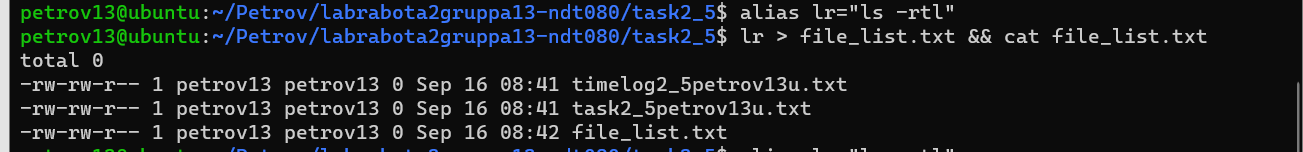
**K = (6\*6\*13) mod 7 = 6**

****

#### Вариант 6.

Задача 1. Создайте псевдоним lr, который вызывает команду ls со следующими возможностями: псевдоним перечисляет файлы в порядке времени их модификации. Перенаправьте вывод команды lr в текущем каталоге в файл file\_list.txt. Просмотрите последние 7 строк содержимого файла file\_list.txt.

Создадим псевдоним созданной нами команды:



Проверим нашу команду в корневом каталоге:

Изображение выглядит как текст

Автоматически созданное описание

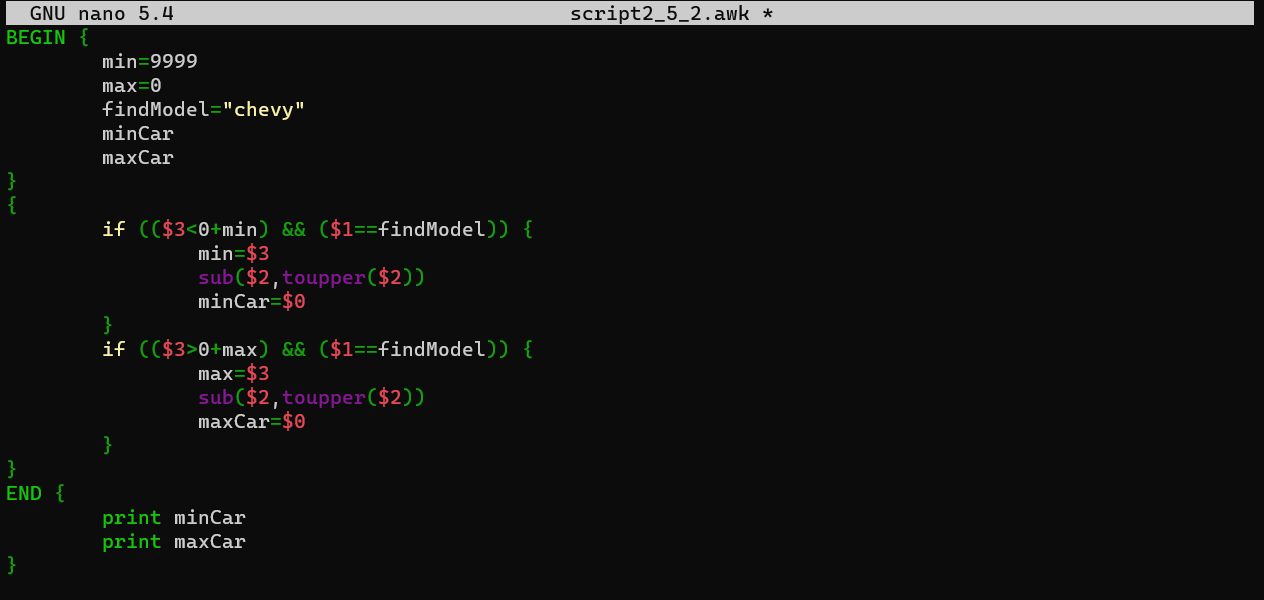
Задание 2. Напишите AWK-программу для вывода списка автомобилей из файла cars.txt (<https://edufpmi.bsu.by/mod/resource/view.php?id=464>), которая вывод самый старый и самый новый автомобиль chevy с выводом модели заглавными буквами.

Создадим файл cars.txt

Изображение выглядит как текст

Автоматически созданное описание

Создадим скрипт:



Выполним скрипт:

Изображение выглядит как текст

Автоматически созданное описание

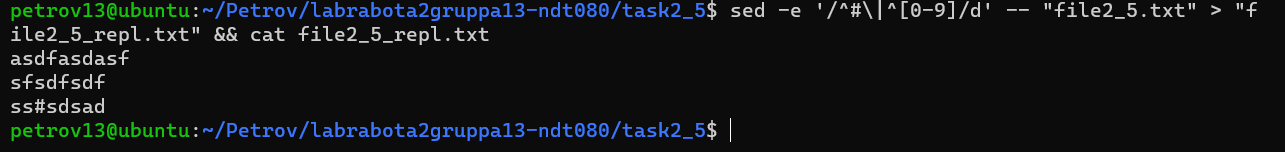
Задание 3. Напишите sed-команду, копирующую файл на стандартный вывод, удаляя при этом все строки, начинающиеся с символа # или цифры.

Создадим файл для тестирования команды:

Изображение выглядит как текст

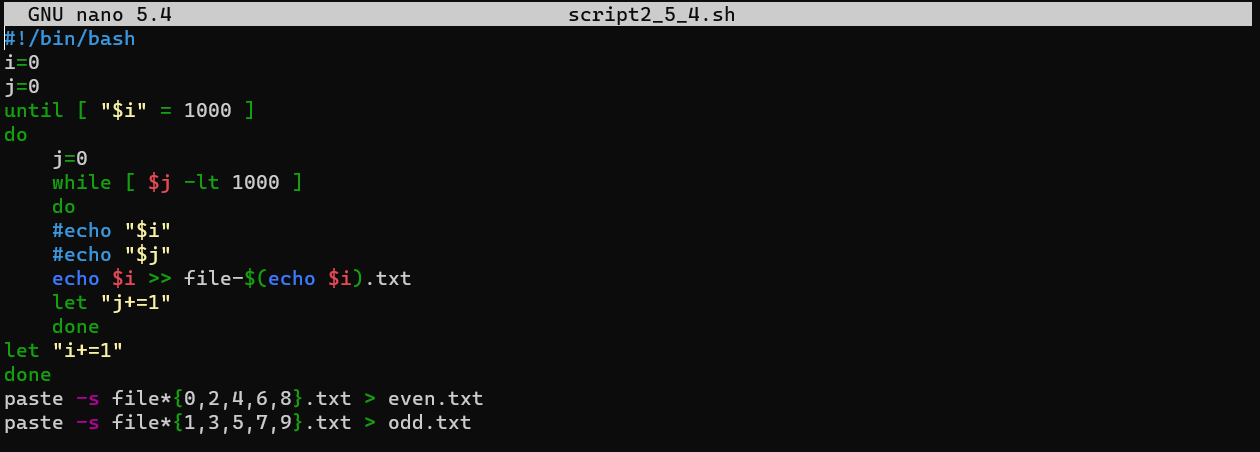
Автоматически созданное описание

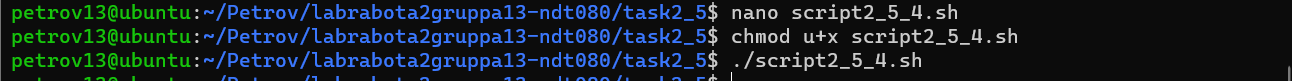
Выполним скрипт и отобразим файл с изменениями:

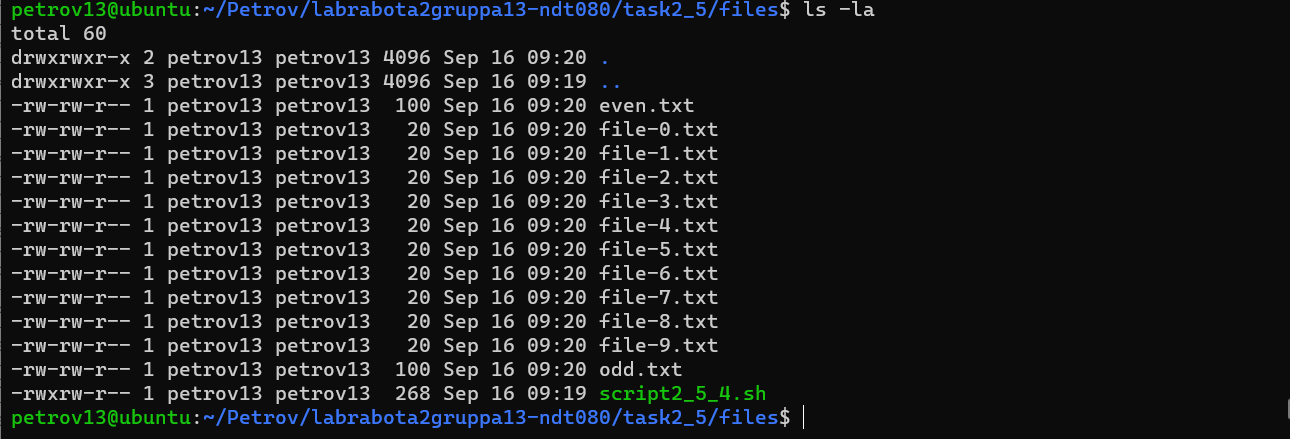


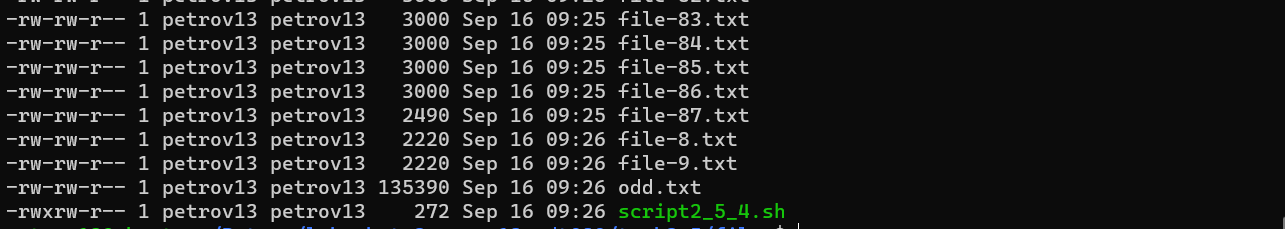
Задача 4. Напишите сценарий, который генерирует тысячу файлов 1.txt ....

1000.txt, и в каждый файл записывает подряд 100 чисел N, где N = порядковый номер файла. Затем скрипт должен соединить в один файл все файлы с четными номерами (even.txt) и в другой файл — все файлы с нечетными номерами (odd.txt).

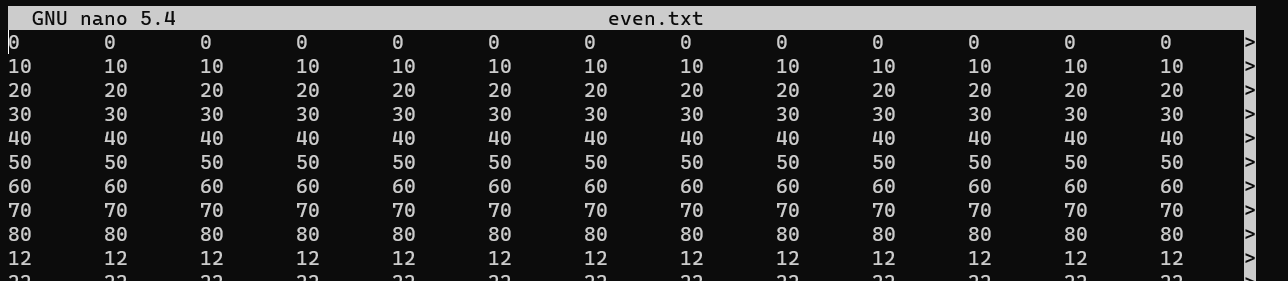


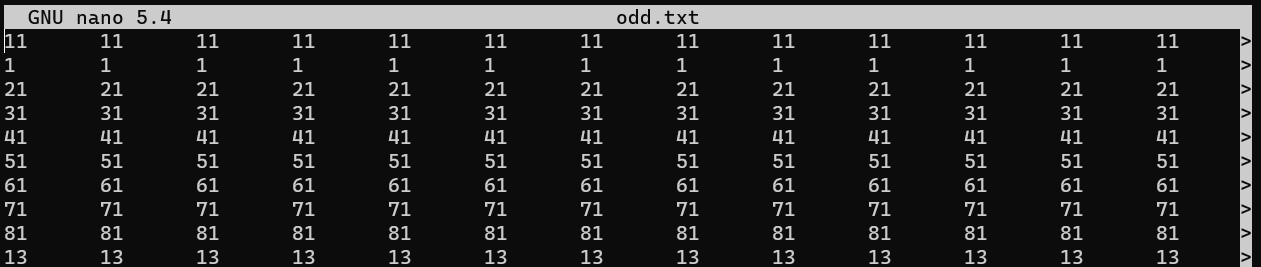






Файлы even.txt и odd.txt





Задача 5.

Написать скрипт, подсчитывающий суммарный размер файлов в заданном каталоге и во всех его подкаталогах (имя каталога задаётся пользователем в качестве аргумента командной строки). Скрипт выводит результаты подсчёта в файл (второй аргумент командной строки) в виде: каталог (полный путь), суммарный размер файлов, число просмотренных файлов. Проверить работу программы для каталога /usr.



