



МИНОБРНАУКИ РОССИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**"МИРЭА - Российский технологический университет"**  
**РТУ МИРЭА**

---

Институт информационных технологий (ИТ)  
Кафедра Вычислительной техники (ВТ)

**Практическая работы №3**  
**по дисциплине**  
**«Архитектура вычислительных машин и систем»**

Выполнил студент группы \_\_\_\_ИКБО-15-22\_\_\_\_

Оганнисян Г.А.

Принял преподаватель \_\_\_\_\_

Рыжова А. А.

Практическая работа  
выполнена

« » ноября 2023 г.

## Содержание

Практическая работа №3.....	3
Вывод.....	6

## Практическая работа №3

### Введение

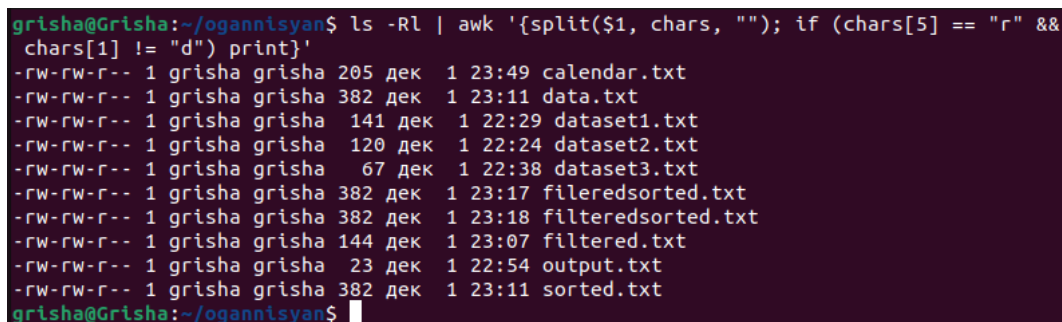
AWK — это интерпретируемый скриптовый C-подобный язык строчного разбора и обработки входного потока (например, текстового файла) по заданным шаблонам (регулярным выражениям). Используется в bash (SH) скриптах. Благодаря AWK в нашем распоряжении оказывается язык программирования, а не довольно скромный набор команд, отдаваемых редактору. С помощью языка программирования AWK можно выполнять следующие действия:

- объявлять переменные для хранения данных;
- использовать арифметические и строковые операторы для работы с данными;
- использовать структурные элементы и управляющие конструкции языка, такие, как условные операторы и циклы;
- реализовать сложные алгоритмы обработки данных;
- создавать форматированные отчёты.

AWK может запоминать контекст, делать сравнения, создавать форматированные отчёты, которые удобно читать и анализировать. Это оказывается очень кстати при работе с лог-файлами, которые могут содержать миллионы записей. При надлежащей сноровке, она может объединять множество строк. Awk – это инструмент, предоставляющий несколько очень удобных способов обработки текстовых данных, которые могут пригодиться в повседневной жизни.

### Выполнение работы

1. Вывод списка файлов, имеющих доступ для групп пользователей по чтению (Рисунок 1).



```
grisha@Grisha:~/ogannisyan$ ls -Rl | awk '{split($1, chars, " "); if (chars[5] == "r" && chars[1] != "d") print}'
-rw-rw-r-- 1 grisha grisha 205 дек  1 23:49 calendar.txt
-rw-rw-r-- 1 grisha grisha 382 дек  1 23:11 data.txt
-rw-rw-r-- 1 grisha grisha  141 дек  1 22:29 dataset1.txt
-rw-rw-r-- 1 grisha grisha  120 дек  1 22:24 dataset2.txt
-rw-rw-r-- 1 grisha grisha   67 дек  1 22:38 dataset3.txt
-rw-rw-r-- 1 grisha grisha 382 дек  1 23:17 filteredsorted.txt
-rw-rw-r-- 1 grisha grisha 382 дек  1 23:18 filteredsorted.txt
-rw-rw-r-- 1 grisha grisha  144 дек  1 23:07 filtered.txt
-rw-rw-r-- 1 grisha grisha   23 дек  1 22:54 output.txt
-rw-rw-r-- 1 grisha grisha 382 дек  1 23:11 sorted.txt
grisha@Grisha:~/ogannisyan$
```

Рисунок 1 - Файлы, имеющие доступ по чтению

2. Вывод списка каталогов, имена которых состоят из английских букв.

(См. Рис. 2)

```
grisha@Grisha:~$ ls -l | awk '$9~/[a-z]/ {print $9}'
Desktop
Documents
Downloads
Music
ogannisyan
Pictures
Public
snap
Templates
Videos
grisha@Grisha:~$
```

Рисунок 2 - Файлы с именами на английском

3. Определение количества байтов, занятых текстовыми файлами (txt)

(Рисунок 3).

```
4,0K  ./local/share/session_migration-ubuntu
4,0K  ./local/share/keyrings/user.keystore
4,0K  ./local/share/keyrings/login.keyring
4,0K  ./local/share/recently-used.xbel
4,0K  ./local/share/evolution/calendar/system/calendar.ics
4,0K  ./local/share/evolution/tasks/system/tasks.ics
84K   ./local/share/evolution/addressbook/system/contacts.db
32K   ./local/share/gvfs-metadata/home-0afa546b.log
4,0K  ./local/share/gvfs-metadata/home
32K   ./local/share/gvfs-metadata/root-48adfad3.log
4,0K  ./local/share/gvfs-metadata/root
4,0K  ./local/share/Trash/info/output.txt.trashinfo
4,0K  ./local/share/Trash/info/filtered.txt.trashinfo
4,0K  ./local/share/Trash/info/filtered.2.txt.trashinfo
4,0K  ./local/share/Trash/info/data.txt.trashinfo
4,0K  ./local/share/Trash/files/data.txt
4,0K  ./local/share/Trash/files/filtered.txt
4,0K  ./local/share/Trash/files/filtered.sorted.txt
4,0K  ./local/share/Trash/files/output.txt
4,0K  ./local/share/Trash/files/filtered.2.txt
63M   total
grisha@Grisha:~$ find . -type f -exec du -ch {} + -exec {} \;
```

Рисунок 3 - Количество байтов

4. Определение количества блоков, содержащих текущий каталог (Рисунок

4).

```
grisha@Grisha:~$ ls -l | awk '{sum+=$2} END {print sum}'
65
```

**Рисунок 4 - Количество блоков**

5. Сортировка списка файлов текущего каталога по возможностям доступа (Рисунок 5).

```
grisha@Grisha:~$ ls -l | sort -k 1
drwx----- 5 grisha grisha 4096 дек  1 21:32 snap
drwxrwxr-x 4 grisha grisha 4096 дек  1 21:31 ogannisyan
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Desktop
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Documents
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Downloads
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Music
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Pictures
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Public
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Templates
drwxr-xr-x 2 grisha grisha 4096 ноя 27 15:20 Videos
total 40
```

**Рисунок 5 - Сортировка по доступу**

6. Вывод списка каталогов, в которых обнаружены файлы с определенным именем (Рисунок 6).

```
grisha@Grisha:~$ find . -name report
./ogannisyan/temp/report
```

**Рисунок 6 - Поиск файла**

7. Подсчет количества вхождений пользователя в систему (Рисунок 7).

```
grisha@Grisha:~$ last | grep grisha | awk '{sum+=1} END {print sum}'
7
```

**Рисунок 7 – Количество зафиксированных входов в систему**

8. Вывод списка пользователей, отсортированного по времени входа в систему (Рисунок 8).

```
grisha@Grisha:~$ last | grep grisha | awk '{print $4, $5, $6, $7, $8, $9}' | sort
Fri Dec 1 21:26 - crash
Mon Dec 4 09:51 still logged
Mon Nov 27 15:20 - crash
Mon Nov 27 15:36 - 15:38
Mon Nov 27 15:39 - crash
Sun Dec 3 10:27 - crash
Sun Dec 3 13:24 - crash
```

**Рисунок 8 - Сортировка по времени входа**

## **Вывод**

В данной практической работы мы познакомились с возможностями программируемого фильтра `awk`. Фильтр широко применяется для обработки данных и формирования различного вида отчетов. Для более глубокого изучения всех возможностей фильтра рекомендуется изучить справочные страницы по команде `awk`.