Отчет по 9 работа

Задача 1

Используя утилиты hexdump и strings, вывести на экран содержимое одного из перечисленных ниже файлов из каталога /bin. Позиция файла для распечатки определяется номером бригады. Имена файлов для выполнения задания 1(TAR)

strings извлекает печатные строки из бинарных файлов, а -n10 означает, что будут показаны только строки длиной 10 символов и больше.

```
ubuntu@ubuntu:~$ strings -n10 /bin/tar
/lib64/ld-linux-x86-64.so.2
__libc_start_main
__cxa_finalize
__fprintf_chk
__stack_chk_fail
getpagesize
__errno_location
__isoc23_strtol
fflush_unlocked
__overflow
__vfprintf_chk
dcngettext
__readlinkat_chk
clock_gettime
fputs_unlocked
gettimeofday
__ctype_b_loc
__memcpy_chk
sigemptyset
__isoc23_strtoul
__strcpy_chk
```

hexdump -C /bin/tar покажет содержимое бинарного файла tar в шестнадцатеричном формате с ASCII-представлением

```
ubuntu@ubuntu:~$ hexdump -C /bin/tar
00040670  4c 63 2d 55 86 02 00 48  39 c6 0f 82 b0 05 00 00  |Lc-U...H9.......|
00040680  41 8b 44 24 08 4d 89 6f  18 85 c0 0f 84 85 f8 ff  |A.D$.M.o........|
00040690  ff 4c 8d 14 40 48 8d 45  a0 4c 89 a5 68 ff ff ff  |.L..@H.E.L..h...|
000406a0  4d 89 c4 49 c1 e2 04 48  89 45 88 48 8d 1d 45 4b  |M..I...H.E.H..EK|
000406b0  01 00 4f 8d 34 10 4d 89  f5 4d 89 c6 eb 0b 66 90  |..O.4.M..M....f.|
000406c0  49 83 c4 30 4d 39 ec 74  66 49 83 3c 24 00 74 f0  |I..0M9.tfI.<$.t.|
000406d0  41 f6 44 24 18 02 75 e8  48 8b 75 88 8b 3d ea 85  |A.D$..u.H.u..=..|
000406e0  02 00 e8 99 8a ff ff 49  8b 14 24 4c 89 ff 31 c0  |.......I..$L..1.|
000406f0  48 8d 35 f5 4a 01 00 e8  64 9f 00 00 48 8b 85 78  |H.5.J...d...H..x|
00040700  ff ff ff 31 c9 48 85 c0  74 07 48 8b 00 48 8b 48  |...1.H..t.H..H.H|
00040710  30 4d 89 f8 48 8d 15 d6  4a 01 00 48 89 de 4c 89  |0M..H...J..H..L.|
00040720  f7 e8 8a 8c ff ff 49 83  c4 30 4d 39 ec 75 9a 49  |......I..0M9.u.I|
00040730  8b 47 38 4c 8b a5 68 ff  ff ff 4d 89 f0 49 2b 47  |.G8L..h...M..I+G|
00040740  30 49 39 47 20 0f 83 dd  f7 ff ff 4c 89 ff 4c 89  |0I9G ......L..L.|
00040750  45 88 e8 49 98 00 00 4c  8b 45 88 e9 c8 f7 ff ff  |E..I...L.E......|
00040760  4c 89 f7 4d 63 ff 4d 63  e4 e8 32 98 00 00 49 8b  |L..Mc.Mc..2...I.|
```

Задача 2

Подсчитать общее количество файлов (каталогов) в одном из перечисленных ниже каталогов. Каталог для подсчета количества определяется номером бригады (bin)

Ls /bin выводит список файлов, вывод перенаправляется в wc (word count) и происходит подсчет:

```
ubuntu@ubuntu:~$ ls /bin | wc -l
1468
ubuntu@ubuntu:~$
```

Задача 3

Найти общее количество процессов, выполняющихся в системе в данный момент.

ps –aux выводит список процессов, вывод перенаправляется в wc (word count) и происходит подсчет:

```
ubuntu@ubuntu:~$ ps -aux | wc -l
337
ubuntu@ubuntu:~$
```

Задача 4

Вывести список выполняющихся процессов, в именах которых присутствует слово manager и отсутствует слово grep**:**

```
ubuntu@ubuntu:~$ ps -aux | grep manager | grep -v grep
root         128  0.0  0.0      0     0 ?        I<   02:05   0:00 [kworker/R-charger_manager]
```

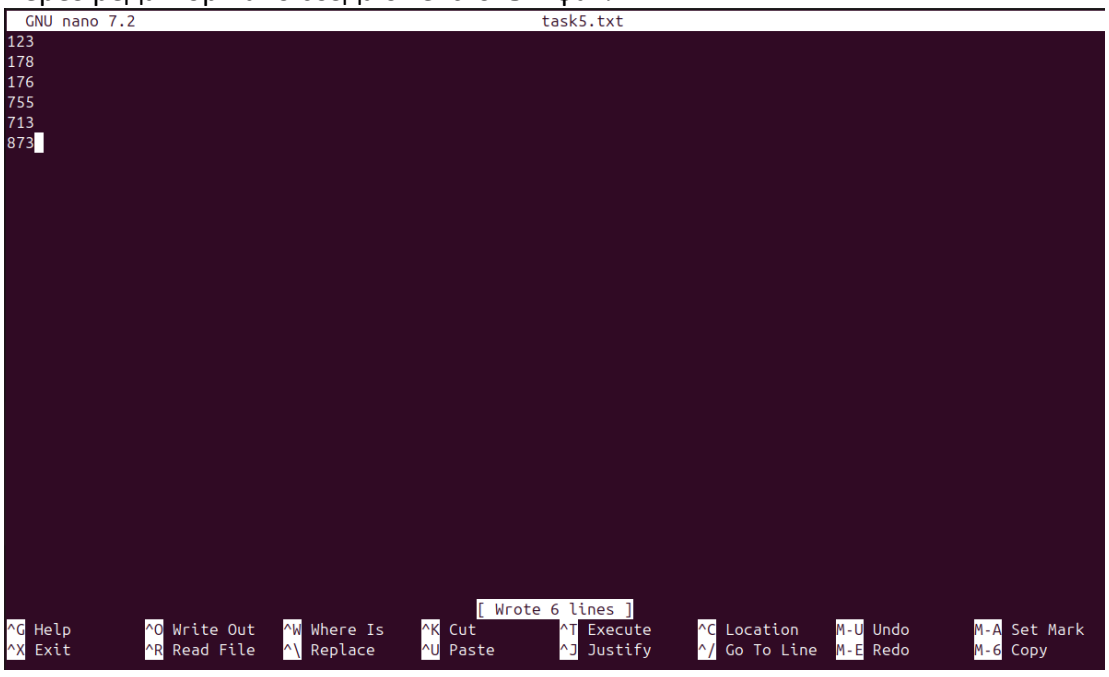
Задача 5

Создать текстовый файл, содержащий набор строк вида:

123
178
176
755
713
873

С помощью утилиты grep найти строки, в которых есть цифра 7, после которой находится одна из цифр —1, 3 или 5.

Через редактор nano создаю текстовый файл:



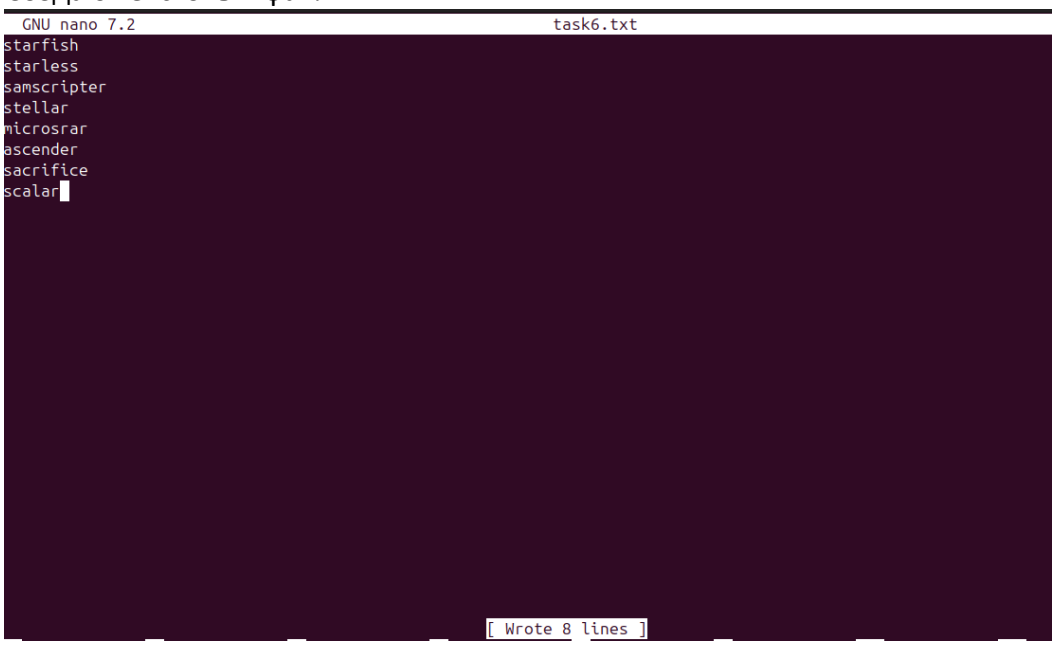
В grep передаю регулярное выражение 7[1|3|5] (на первом месте 7 обязательно, на втором любая из 3х заданных цифр):

```
ubuntu@ubuntu:~$ nano task5.txt
ubuntu@ubuntu:~$ grep '7[1|3|5]' task5.txt
755
713
873
ubuntu@ubuntu:~$
```

Задача 6

Создать текстовый файл, содержащий набор строк вида:
starfish
starless
samscripter
stellar
microsrar
ascender
sacrifice
scalar
С помощью утилиты grep найти строки, начинающиеся на букву s и заканчивающиеся на букву r.

Создаю текстовый файл:



В grep передаю выражение, в котором: ^ - начало строки, s - буква s в начале, .* - любое количество любых символов, r - буква r, $ - конец строки

```
ubuntu@ubuntu:~$ nano task6.txt
ubuntu@ubuntu:~$ grep '^s.*r$' task6.txt
samscripter
stellar
scalar
ubuntu@ubuntu:~$
```

Задача 7

Создать текстовый файл, содержащий простейшие адреса электронной почты вида username@website.com. С помощью утилиты grep найти строки, содержащие правильные простейшие адреса. Проверить возможность использования более сложного регулярного выражения для распознавания адресов, содержащих другие допустимые символы.

Через редактор nano создала файл с адресами и не только, вывела файл в терминал через cat и передала в grep регулярное выражение, чтобы выбрать из файла только возможные адреса эп:

```
ubuntu@ubuntu:~$ nano emails.txt
ubuntu@ubuntu:~$ cat emails.txt
ironmaiden@gmail.com
ahahhahahhah
helloween_band@gmail.com
idontknowwhoami
dasha@pochta.ru
konets
honkaistarrail@ya.ru
ubuntu@ubuntu:~$ grep -E '^[A-Za-z0-9._%=?^+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$' emails.txt
ironmaiden@gmail.com
helloween_band@gmail.com
dasha@pochta.ru
honkaistarrail@ya.ru
ubuntu@ubuntu:~$
```

В регулярном выражении: ^ - начало строки, [a-zA-Z0-9. %+-]+ - имя пользователя, символ @, [a-zA-Z0-9.-]+ - доменное имя, \. - точка перед доменной зоной ,[a-zA-Z]{2,} - доменная зона (2 и более букв), $ - конец строки.

Задача 8

Создать текстовый файл, содержащий допустимые и недопустимые IP-адреса, например
127.0.0.1
255.255.255.255
12.34.56
123.256.0.0
1.23.099.255
0.79.378.111
С помощью утилиты grep и руководства man найти строки, содержащие допустимые четырехбайтовые IP адреса.

Через редактор nano создаю текстовый файл и вывожу(cat) в терминал. Использую grep с ключом –E и регулярное выражение, позволяющее проверить корректность ip адреса (25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\. – проверяем, что каждое из четырех чисел адреса попадает в диапазон 250-255 или 200-249 или 0-199. И заканчивается «.» кроме 4 числа в ip адресе (в выражении символ \.)

```
ubuntu@ubuntu:~$ cat ipadress.txt
127.0.0.1
255.255.255.255
12.34.56
123.256.0.0
1.23.099.255
0.79.378.111
ubuntu@ubuntu:~$ grep -E '^(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4]
[0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0
-9]|[01]?[0-9][0-9]?)$' ipadress.txt
127.0.0.1
255.255.255.255
1.23.099.255
```

Пример использования tr (удаление повторяющегося символа из текста)

```
ubuntu@ubuntu:~$ echo daaaaaaaasha | tr -s 'a'
dasha
```

Задача 9

Создать текстовый файл, содержащий корректные и некорректные номера телефонов ведомственной АТС объемом 399 номеров, номера с 000 до 399 – корректные, 0, 400, 900 – некорректные. С помощью утилиты grep и руководства man найти строки, содержащие допустимые номера телефонов.

Создаю файл с номерами и проверяю их корректность используя следующее регулярное выражение: '(0[0-9][0-9]|[1-2][0-9][0-9]|3[0-9][0-9])'

```
ubuntu@ubuntu:~$ nano phones.txt
ubuntu@ubuntu:~$ cat phones.txt
199
00
29
300
399
400
401
900
0
5
ubuntu@ubuntu:~$ grep -E '(0[0-9][0-9]|[1-2][0-9][0-9]|3[0-9][0-9])' phones.txt
199
300
399
```

Для номеров в диапазоне 000-099 –первая цифра всегда 0, вторая и третья 0-9: 0[0-9][0-9]
Для номеров в диапазоне 100-299 –первая цифра 1 или 2, вторая и третья 0-9: [1-2][0-9][0-9]
Для номеров в диапазоне 300-399 –первая цифра 3, вторая и третья 0-9: [1-2][0-9][0-9]
Остальные не подходят.

Конец!