

MODUL PRAKTIKUM PERTEMUAN 3
SISTEM OPERASI
IF-2231

OPERASI INPUT OUTPUT



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI PRODUKSI DAN INDUSTRI
INSTITUT TEKNOLOGI SUMATERA
2019

Tujuan dan Pokok Bahasan

Setelah mempelajari materi yang ada pada modul praktikum ini, praktikan diharapkan mampu:

1. Memahami konsep proses I/O dan *redirection*
2. Memahami standar *input*, *output* dan *error*
3. Menggunakan notasi *output*, *append* dan *here document*
4. Memahami konsep PIPE dan *filter*

Teori Dasar

1. Operasi Input Output

Proses adalah program yang sedang dieksekusi atau sedang berjalan. Perintah atau *command* yang diberikan melalui Shell disebut sebagai eksekusi program yang selanjutnya disebut proses. Setiap kali instruksi diberikan, maka *kernel* akan menciptakan sebuah proses dengan memberikan nomor PID/*Process Identity*. Proses dalam Linux selalu membutuhkan Input dan menghasilkan suatu Output. Dalam konteks Linux input/output (I/O) adalah :

- Keyboard
- Layar monitor
- Files
- Struktur data kernel
- Perangkat input/output lainnya



2. File Descriptor

File Descriptor adalah indikator abstrak yang digunakan untuk mengakses *file* atau sumber input/output lainnya, seperti *pipe* atau soket jaringan. *File descriptor* direpresentasikan melalui angka. Tiga buah file descriptor standar yang lalu diciptakan oleh proses adalah :

- 0 = standar *input*
- 1 = standar *output*
- 2 = standar *error*



Linux tidak membedakan antara peralatan hardware dan file, Linux memanipulasi peralatan hardware sama dengan file.

3. Redirection

Redirection atau pembelokan dilakukan untuk standar *input*, *output* dan *error*, yaitu untuk mengalihkan *file descriptor* dari 0, 1 dan 2. Simbol untuk pembelokan adalah :

- 0< atau < pengganti standar *input*
- 1> atau > pengganti standar *output*
- 2> pengganti standar *error*

4. PipeLine

Mekanisme pipa digunakan sebagai alat komunikasi antar proses.

Proses 1 menghasilkan output yang selanjutnya digunakan sebagai input oleh Proses 2. Hubungan output input ini dinamakan pipa, yang menghubungkan Proses 1 dengan Proses 2 dan dinyatakan dengan simbol |.

Proses 1 | Proses 2

5. Filter

Filter adalah utilitas Linux yang dapat memproses standar *input* (dari keyboard) dan menampilkan hasilnya pada standard *output* (layar). Contoh filter adalah **awk**, **cat**, **sed**, **sort**, **grep**, **head**, **tail**, **sort**, **uniq** dan lainnya.

Jika pada sebuah susunan rangkaian PipeLine :



Maka Proses 2 sampai dengan Proses n-1 harus utilitas Linux yang berfungsi sebagai *filter*. P1 (awal) dan Pn (terakhir) boleh tidak filter. Utilitas yang bukan filter misalnya **who**, **ls**, **ps**, dan lainnya.

Beberapa perintah Linux yang digunakan untuk proses penyaringan antara lain:

- **cat**

Digunakan untuk membuat 1 atau banyak *file*, melihat isi *file*, menggabungkan dan membelokkan / *redirect* output di *terminal* atau *file*

- **sort**

Digunakan untuk mengurutkan masukannya berdasarkan urutan nomor ASCII dari karakter.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	:	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

Source: www.LookupTables.com

- **grep**

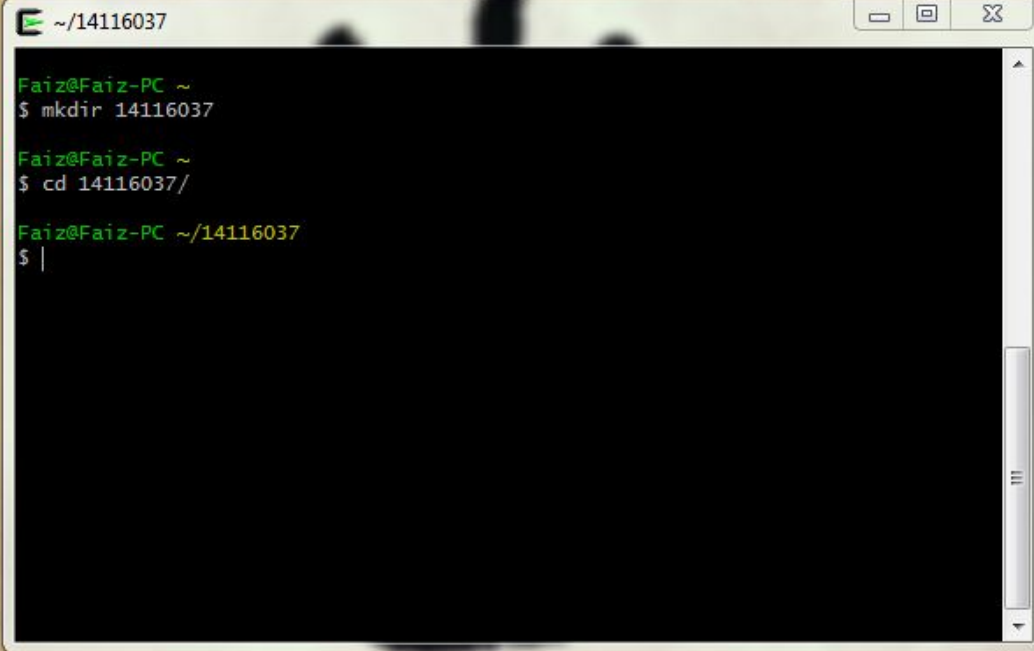
Digunakan untuk menyaring *input* dan menampilkan baris-baris yang hanya mengandung pola yang ditentukan. Pola ini disebut regular expression.

- **uniq**

uniq merupakan utilitas untuk melaporkan atau memfilter baris yang diulang di dalam *file*.

Percobaan

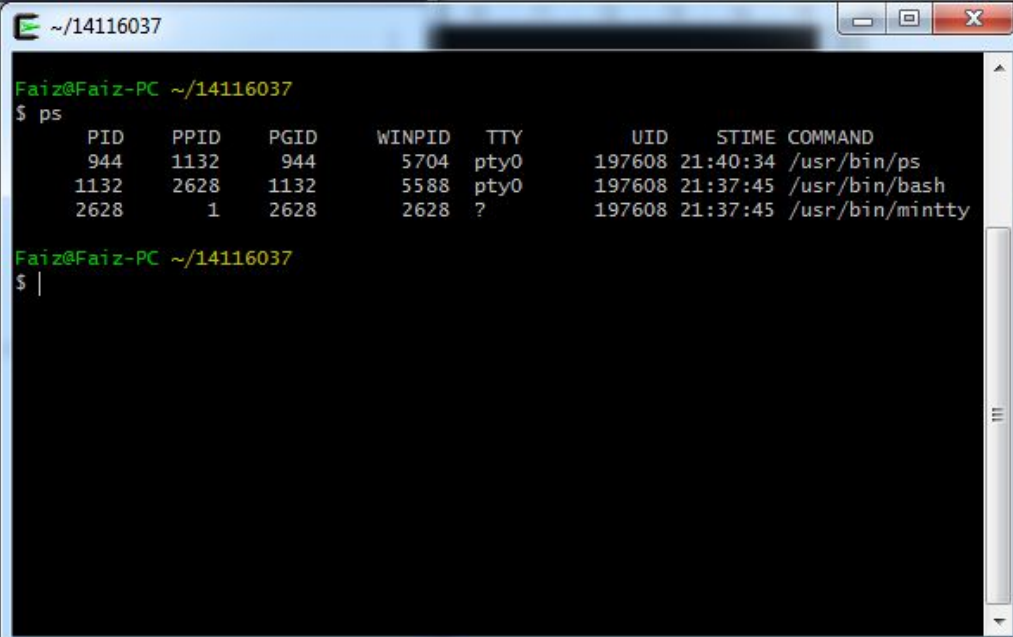
Dalam percobaan, sebaiknya terlebih dahulu membuat direktori dengan NIM praktikan. Kemudian gunakan folder tersebut untuk praktikum.



```
~/14116037
Faiz@Faiz-PC ~
$ mkdir 14116037
Faiz@Faiz-PC ~
$ cd 14116037/
Faiz@Faiz-PC ~/14116037
$ |
```

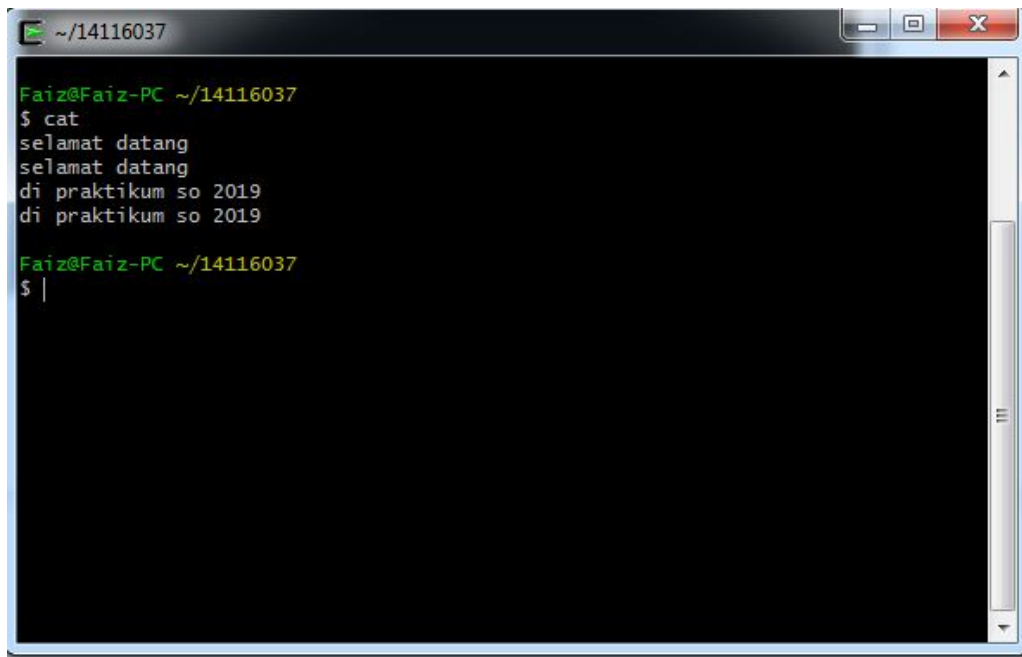
Percobaan 1: File Descriptor

1. **ps**, *input* dari sistem (kernel), *output* ke layar (standar *output*)



```
~/14116037
Faiz@Faiz-PC ~/14116037
$ ps
  PID   PPID  PGID   WINPID  TTY      UID    STIME  COMMAND
   944    1132   944     5704   pty0     197608 21:40:34 /usr/bin/ps
  1132   2628  1132     5588   pty0     197608 21:37:45 /usr/bin/bash
  2628     1   2628     2628   ?        197608 21:37:45 /usr/bin/mintty
Faiz@Faiz-PC ~/14116037
$ |
```

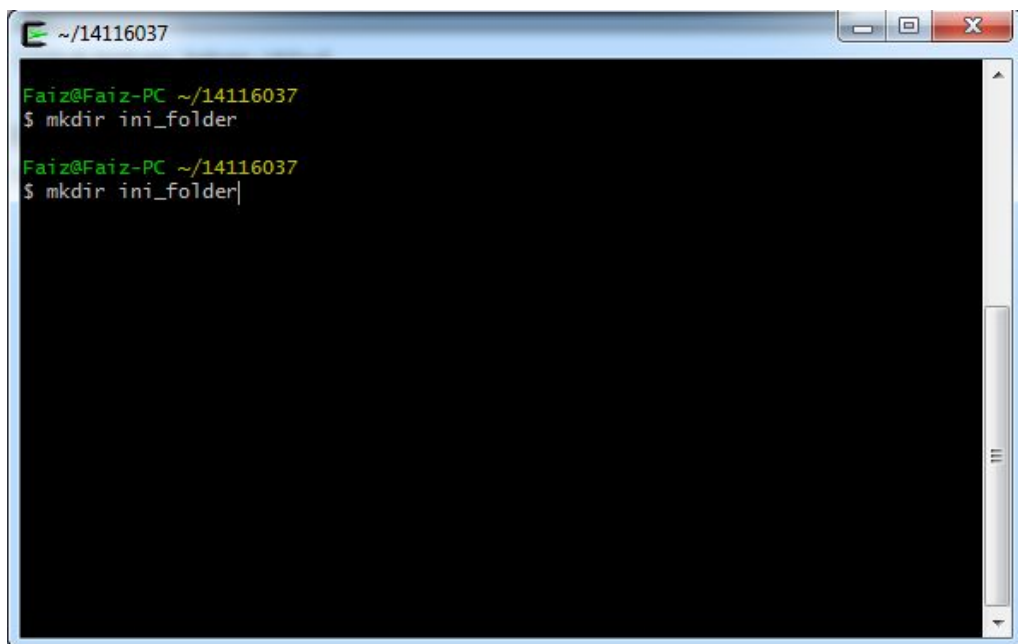
2. **cat**, *input* dari keyboard (standar *input*), *output* ke layar (standar *output*).
ketik **cat**, lalu tekan *enter*.
Ketik kalimat apa saja, lalu tekan *enter*.
Kalimat yang diketik tadi akan muncul di bawahnya.



```
~/14116037
Faiz@Faiz-PC ~/14116037
$ cat
selamat datang
selamat datang
di praktikum so 2019
di praktikum so 2019
Faiz@Faiz-PC ~/14116037
$ |
```

Untuk keluar, tekan *ctrl+d*.

3. Buat folder dengan sembarang nama, lalu buat folder dengan nama yang sama. Perhatikan apa yang terjadi.



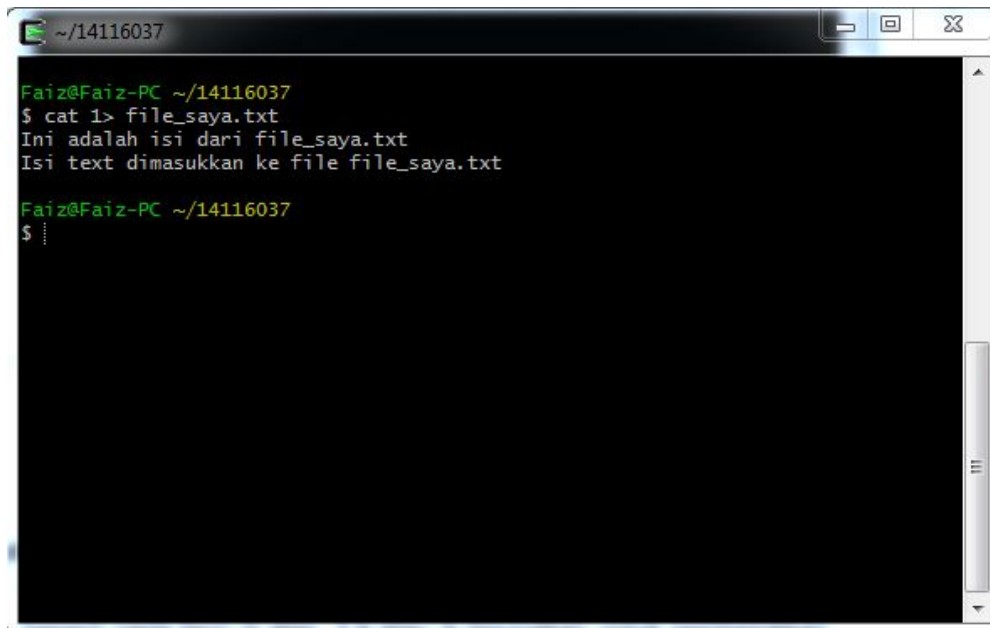
```
~/14116037
Faiz@Faiz-PC ~/14116037
$ mkdir ini_folder
Faiz@Faiz-PC ~/14116037
$ mkdir ini_folder|
```

bila terjadi error, maka akan ada tampilan *error* pada layar (standar *error*)

Percobaan 2: Redirection

1. Pembelokkan standar *output* dan standar *input*.

Seperti pada teori di atas, **1>** atau **>** digunakan untuk membelokkan pengganti ke standar *input*.

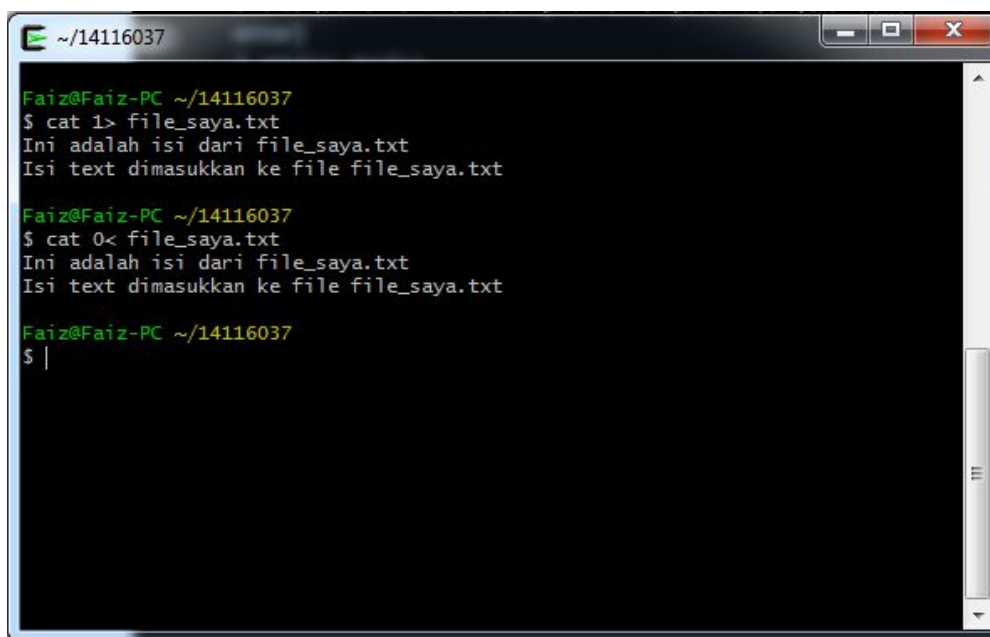


```
~/14116037
Faiz@Faiz-PC ~/14116037
$ cat 1> file_saya.txt
Ini adalah isi dari file_saya.txt
Isi text dimasukkan ke file file_saya.txt

Faiz@Faiz-PC ~/14116037
$
```

Pada percobaan di atas, *input* dibelokkan ke sebuah *file* bernama 'file_saya.txt'. *File* 'file_saya.txt' sekarang sudah berisi seperti 2 kalimat di atas.

Seperti pada teori di atas, **1>** atau **>** digunakan untuk membelokkan pengganti ke standar *input*.



```
~/14116037
Faiz@Faiz-PC ~/14116037
$ cat 1> file_saya.txt
Ini adalah isi dari file_saya.txt
Isi text dimasukkan ke file file_saya.txt

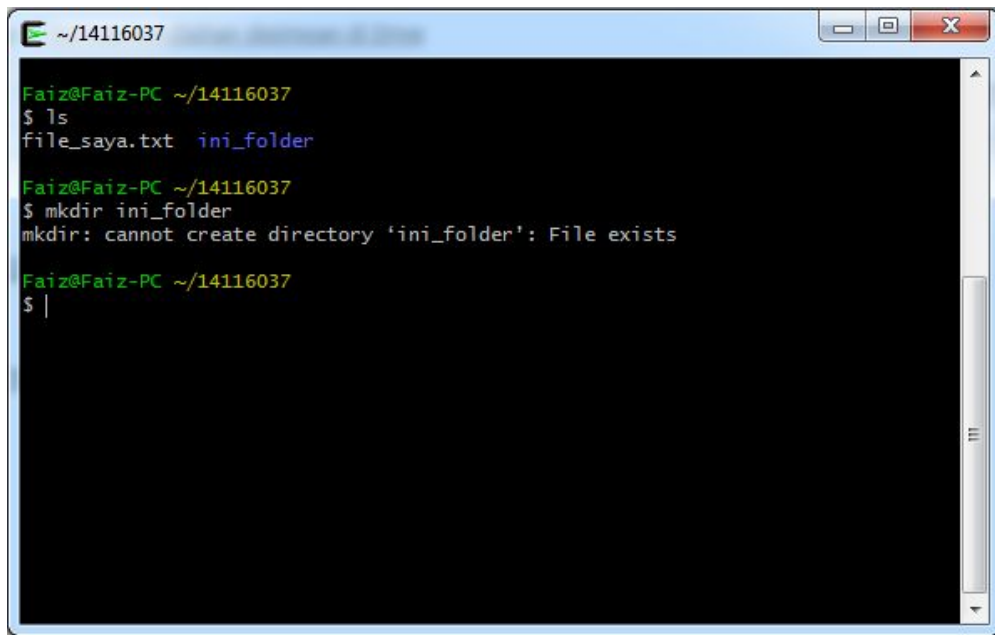
Faiz@Faiz-PC ~/14116037
$ cat 0< file_saya.txt
Ini adalah isi dari file_saya.txt
Isi text dimasukkan ke file file_saya.txt

Faiz@Faiz-PC ~/14116037
$ |
```


Untuk menampilkan isi file menggunakan pemblokkan standar *output* **cat**, bisa menggunakan perintah seperti diatas.

2. Pemblokkan standar *error*

Pada percobaan *file descriptor*, salah satu *error* disebabkan oleh pembuatan direktori dengan nama yang sama.

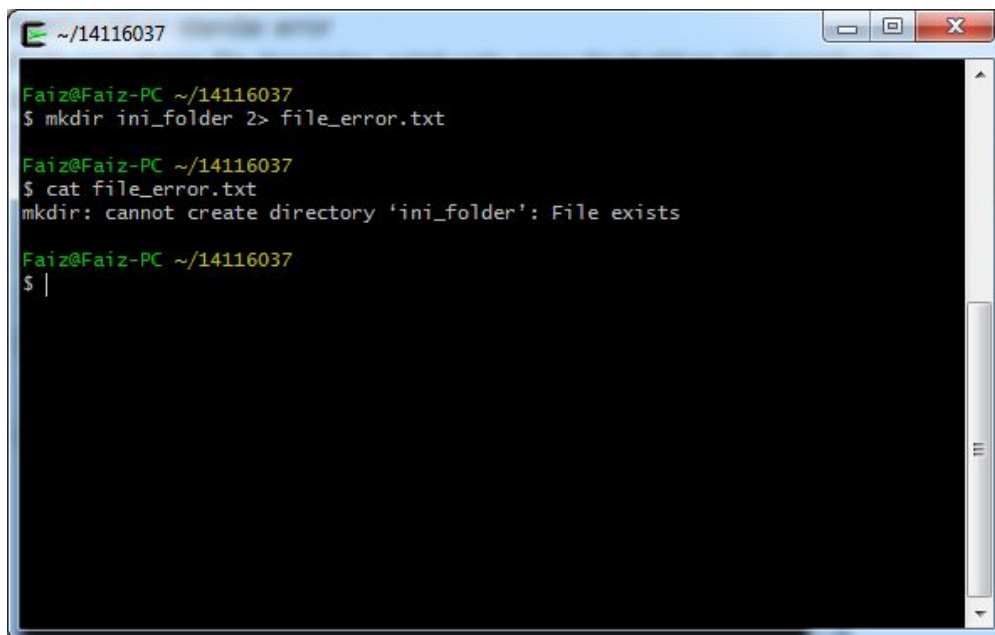
A terminal window titled '~ /14116037' showing a series of commands and their outputs. The user first lists files, then attempts to create a directory named 'ini_folder', which fails because it already exists.

```
~/14116037
Faiz@Faiz-PC ~/14116037
$ ls
file_saya.txt  ini_folder

Faiz@Faiz-PC ~/14116037
$ mkdir ini_folder
mkdir: cannot create directory 'ini_folder': File exists

Faiz@Faiz-PC ~/14116037
$ |
```

penulisan *error* tersebut dapat diblokkan ke dalam sebuah file.

A terminal window titled '~ /14116037' showing the same 'mkdir' command as before, but with the error output redirected to a file named 'file_error.txt'. The user then uses 'cat' to display the contents of 'file_error.txt', which shows the error message.

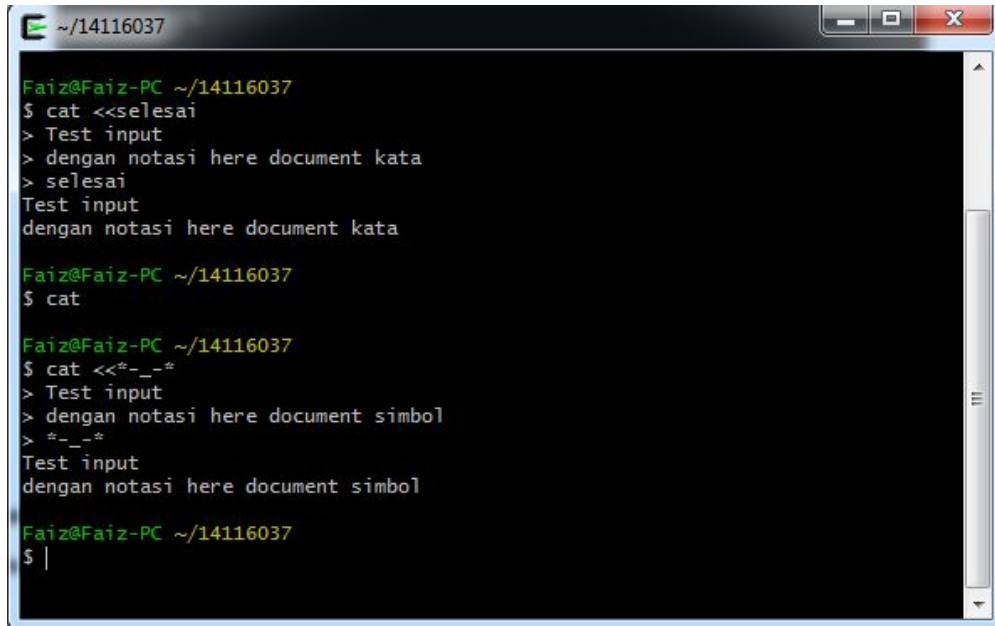
```
~/14116037
Faiz@Faiz-PC ~/14116037
$ mkdir ini_folder 2> file_error.txt

Faiz@Faiz-PC ~/14116037
$ cat file_error.txt
mkdir: cannot create directory 'ini_folder': File exists

Faiz@Faiz-PC ~/14116037
$ |
```

3. Notasi here document

Digunakan sebagai pembatas *input* dari keyboard. Tanda pembatas bisa apa saja, namun tanda harus sama awal dan akhir

A terminal window titled '~/.14116037' with standard window controls. It shows three examples of using here documents with the 'cat' command. The first example uses '<<selesai' and '>selesai' as delimiters. The second example uses '<<*_-*' and '>*_-*' as delimiters. The third example shows the start of a here document with '<<*_-*'.

```
Faiz@Faiz-PC ~/.14116037
$ cat <<selesai
> Test input
> dengan notasi here document kata
> selesai
Test input
dengan notasi here document kata

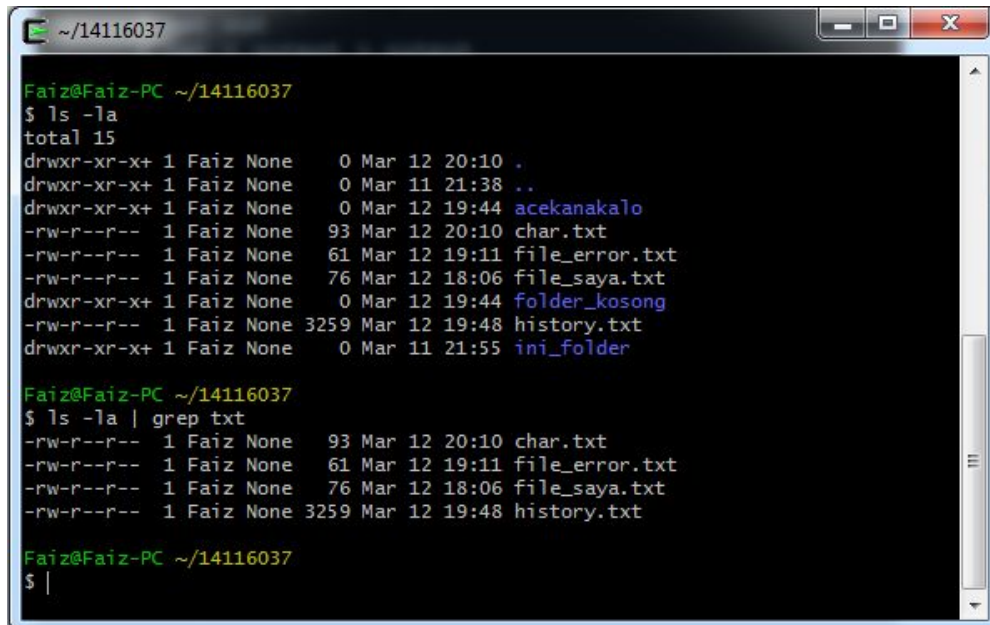
Faiz@Faiz-PC ~/.14116037
$ cat

Faiz@Faiz-PC ~/.14116037
$ cat <<*_-*
> Test input
> dengan notasi here document simbol
> *_-*
Test input
dengan notasi here document simbol

Faiz@Faiz-PC ~/.14116037
$ |
```

Percobaan 3: Pipeline dan Filter

Pipeline digunakan untuk membuat eksekusi proses dengan melewati data langsung ke data lainnya.

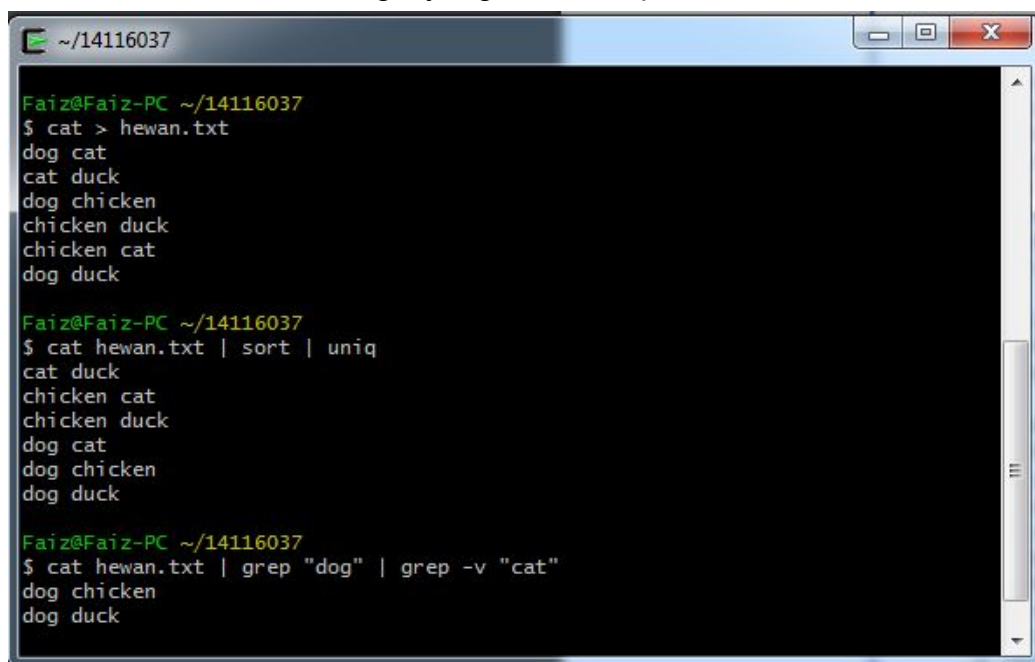


```
~/14116037
Faiz@Faiz-PC ~/14116037
$ ls -la
total 15
drwxr-xr-x+ 1 Faiz None 0 Mar 12 20:10 .
drwxr-xr-x+ 1 Faiz None 0 Mar 11 21:38 ..
drwxr-xr-x+ 1 Faiz None 0 Mar 12 19:44 acekanakalo
-rw-r--r-- 1 Faiz None 93 Mar 12 20:10 char.txt
-rw-r--r-- 1 Faiz None 61 Mar 12 19:11 file_error.txt
-rw-r--r-- 1 Faiz None 76 Mar 12 18:06 file_saya.txt
drwxr-xr-x+ 1 Faiz None 0 Mar 12 19:44 folder_kosong
-rw-r--r-- 1 Faiz None 3259 Mar 12 19:48 history.txt
drwxr-xr-x+ 1 Faiz None 0 Mar 11 21:55 ini_folder

Faiz@Faiz-PC ~/14116037
$ ls -la | grep txt
-rw-r--r-- 1 Faiz None 93 Mar 12 20:10 char.txt
-rw-r--r-- 1 Faiz None 61 Mar 12 19:11 file_error.txt
-rw-r--r-- 1 Faiz None 76 Mar 12 18:06 file_saya.txt
-rw-r--r-- 1 Faiz None 3259 Mar 12 19:48 history.txt

Faiz@Faiz-PC ~/14116037
$ |
```

Pipeline juga digunakan untuk mengkombinasi utilitas sistem untuk membuat fungsi yang lebih kompleks.



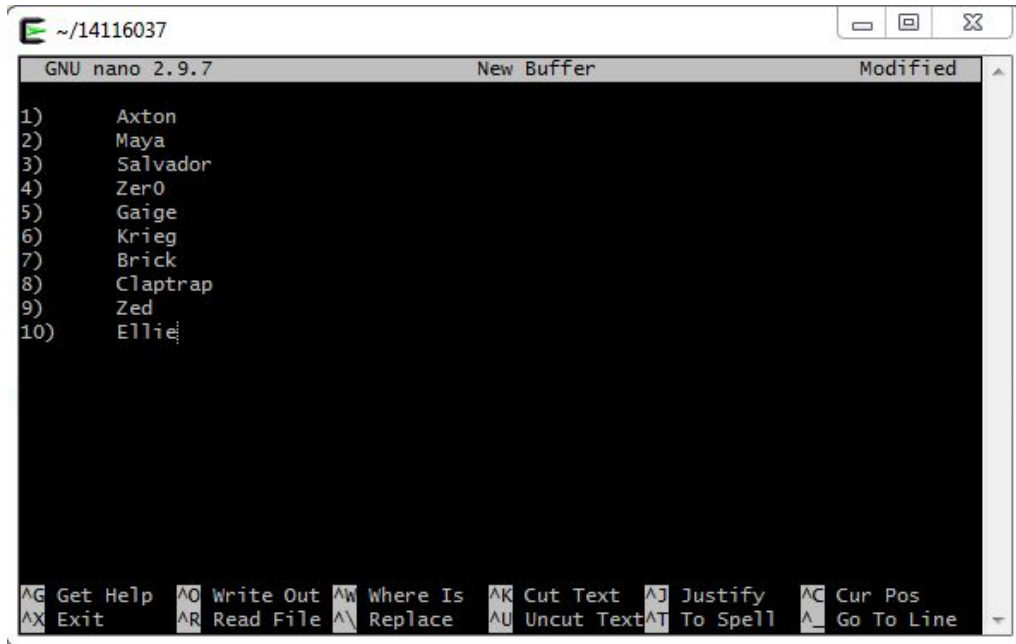
```
~/14116037
Faiz@Faiz-PC ~/14116037
$ cat > hewan.txt
dog cat
cat duck
dog chicken
chicken duck
chicken cat
dog duck

Faiz@Faiz-PC ~/14116037
$ cat hewan.txt | sort | uniq
cat duck
chicken cat
chicken duck
dog cat
dog chicken
dog duck

Faiz@Faiz-PC ~/14116037
$ cat hewan.txt | grep "dog" | grep -v "cat"
dog chicken
dog duck
```

Tugas

1. Jelaskan perbedaan antara command **more** dan **less**, serta berikan contoh dengan *screenshot*.
2. Buat isi file dengan isi seperti berikut (bisa menggunakan nano, vim, ataupun pembelokkan, akan tetapi jika menggunakan pembelokkan mendapat nilai +)



The screenshot shows a terminal window with the GNU nano 2.9.7 text editor. The editor is open to a file named 'New Buffer'. The content of the file is a list of names, each preceded by a number from 1 to 10. The names are: Axton, Maya, Salvador, Zer0, Gaige, Krieg, Brick, Claptrap, Zed, and Ellie. The editor's status bar at the bottom shows various keyboard shortcuts for editing and navigation.

```
GNU nano 2.9.7 New Buffer Modified
1) Axton
2) Maya
3) Salvador
4) Zer0
5) Gaige
6) Krieg
7) Brick
8) Claptrap
9) Zed
10) Ellie

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

simpan dengan nama char.txt. Dengan menggunakan **awk**, tampilkan isi file tersebut, namun hanya nama-nama yang ditampilkan. Setelah ditampilkan, urutkan berdasarkan alphabet. Screenshot proses pembuatan *file*, isi *file* sebelum dan sesudah pengurutan.

3. Buatlah file bernama pesan_kernel.rxt dan belokan hasil keluaran perintah **dmesg** kedalam pesan_kernel.txt.
4. Jelaskan apa itu regular expresion dan pengunanya menggunakan grep.
5. Hitung isi direktory / pada linux kalian menggunakan pembelokkan.