**LAPORAN RESMI**

**PRAKTIKUM SISTEM OPERASI**

**“OPERASI INPUT OUTPUT”**



**NAMA : Muh. Arga Swara Iskandar**

 **NRP : 3120600021**

**KELAS : 1 D4 IT A**

1. Tugas Pendahuluan
2. Apa yang dimaksud *redirection* ?

Redirection adalah pembelokan yang dilakukan untuk standard input, output dan *error*, yaitu untuk mengalihkan file descriptor dari 0, 1 dan 2.

1. Apa yang dimaksud *pipeline* ?

Pipeline adalah mekanisme untuk komunikasi antar proses, digunakan untuk perintah yang saling berurut, dimana output dari perintah awal akan digunakan oleh perintah berikutnya.

1. Apa yang dimaksud perintah di bawah ini :

echo, cat, more, sort, grep, wc, cut, uniq

* echo

*echo* akan mencetak string yang akan diproses sebagai argumen pada output standard.

* cat

*cat* merupakan perintah yang digunakan untuk melihat isi dari suatu file, bahkan kita juga dapat melihat isi dari beberapa file sekaligus.

* more

*more* merupakan perintah yang digunakan untuk menampilkan isi dari suatu file per satu layar penuh, apabila layar sudah penuh tetapi semua file belum ditampilkan, maka kita dapat menekan *spasi* untuk melihat isi file selanjutnya.

* sort

*sort* digunakan untuk mengurutkan keluaran pada terminal, yang mana urutan tersebut berdasarkan urutan ASCII.

* grep

*grep* merupakan salah satu perintah untuk melakukan filter, *grep* akan menampilkan output berupa baris - baris yang mengandung kata kunci yang dicari pada suatu file.

* wc

perintah *wc* digunakan untuk menghitung jumlah baris, kata dan karaketer pada suatu file. *Wc* memiliki beberapa argument yang dapat ditambahkan seperti *-l* untuk menghitung baris, ­*-w­* untuk menghitung kata, dan ­*-c­* untuk menghitung karakter.

* cut

cut digunakan untuk mengambil kolom tertentu dari baris-baris masukannya, yang ditentukan pada option –c.

* uniq

perintah *uniq* merupakan perintah filter yang akan menghilangkan baris berurutan yang mengalami duplikasi, biasanya digabungkan dengan perintah *sort* menggunakan *pipeline*.

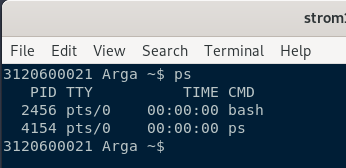
1. Percobaan

* Percobaan 1 : File descriptor

1. Output ke layar (standar output), input dari system (kernel)
2. Command

*$ ps*

1. Capture Output



1. Analisa

*$ ps* akan menampilkan aktif proses yang berjalan pada sistem

1. Output ke layar (standar output), input dari keyboard (standard input)
2. Command

$ cat

hallo, apa khabar

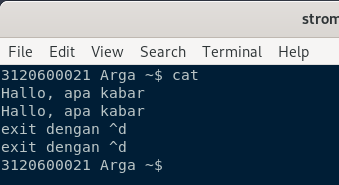
hallo, apa khabar

exit dengan ^d

exit dengan ^d

[Ctrl-d]

1. Capture Output



1. Analisa

*$ cat* akan memungkinkan kita untuk memberikan input dan setelah itu akan mengembalikan output ke layar berupa input yang kita tuliskan. Untuk keluar dari *cat,* kita dapat menggunakan ctrl + d.

1. Input dari keyboard dan output ke alamat internet
2. Command

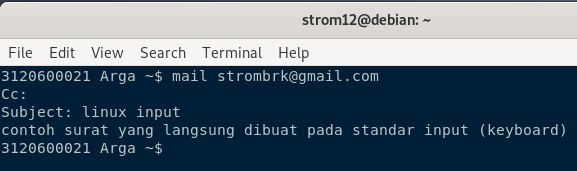
$ mail arna@eepis-its.edu

contoh surat yang langsung

dibuat pada standard input (keyboard)

[Ctrl-d]

1. Capture Output



1. Analisa

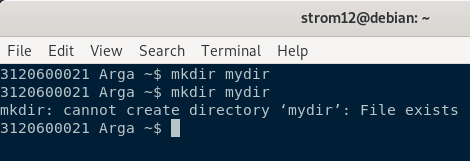
Dengan menggunakan *$ mail* yang diikuti dengan alamat email seseorang, maka kita dapat mengirimkan email ke orang tersebut, kiat dapat memberikan input pesan dengan keyboard dan pesan akan ditampilkan sebagai output pada alamat email yang dipilih (internet)

1. Input nama direktori, output tidak ada (membuat direktori baru), bila terjadi *error* maka tampilan *error* pada layar (standard *error*)
2. Command

$ mkdir mydir

$ mkdir mydir (Terdapat pesan *error*)

1. Capture Output



1. Analisa

Pada percobaan d iatas kita membuat suatu direktori baru dengan nama *mydir* dan berhasil dilakukan, untuk seanjutnya kita membuat direktori lagi dengan nama yang sama dan diberikan *output* *error* karena direktori yang ingin dibuat sudah ada.

* Percobaan 2 : Pembelokan (redirection)

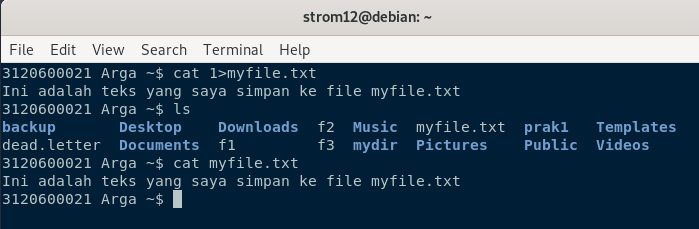
1. Pembelokan standart output
2. Command

$ cat 1> myfile.txt

Ini adalah teks yang saya simpan

Ke file myfile.txt

1. Capture Output



1. Analisa

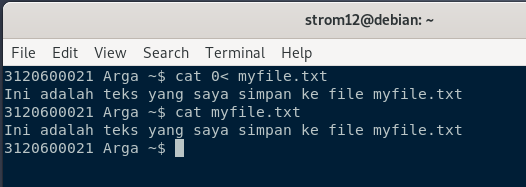
Denngan menggunakan

1. Pembelokan standar input, yaitu input dibelokkan dari keyboard menjadi dari file
2. Command

$ cat 0< myfile.txt

$ cat myfile.txt

1. Capture Output



1. Analisa

0< merupakan perintah untuk membelokkan standard input dari keyboard menjadi dari file

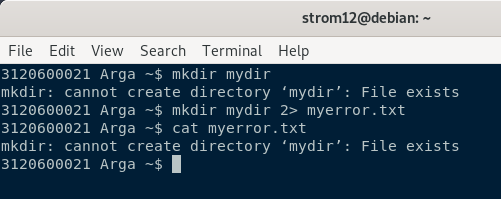
1. Pembelokan standar *error* untuk disimpan di file
2. Command

$ mkdir mydir (Terdapat pesan *error*)

$ mkdir mydir 2> my*error*.txt

$ cat my*error*.txt

1. Capture Output



1. Analisa

Pada percobaan di atas, kita mengalihkan pesan *error* ke dalam file *myerror.txt* dengan menggunakan 2> *myerror.txt.* Pertama kita membuat direktori mydir yang akan menghasilkan pesan *error* dan *error* akan tampil di layer. Kedua kita membuat direktori yang sama, akan tetapi kita alihkan pesan *error* ke dalam file *myerror.txt* sehingga pesan *error* tidak langsung muncul di layer melainkan disimpan di dalam file *myerro.txt.* Ketika kita membuka file *myerror.txt*, maka pesan *error* dari perintah sebelumnya akan ditampilkan.

1. Notasi 2>&1 : pembelokan standar *error* (2>) adalah identic dengan file descriptor 1
2. Command

$ ls filebaru (Terdapat pesan error)

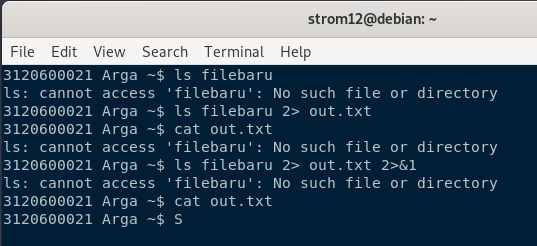
$ ls filebaru 2> out.txt

$ cat out.txt

$ ls filebaru 2> out.txt 2>&1

$ cat out.txt

1. Capture Output



1. Analisa

Pada percobaan di atas, pertama kita menyimpan pesan *error* ke dalam file *out.txt* dengan menggunakan 2> *out.txt*, sehingga pesan error tidak langsung tampil ke layar, melainkan disimpan di dalam file *out.txt*. Ketika kita membuka isi file *out.txt* dengan perintah *$ cat*, maka pesan *error* pada perintah sebelumnya akan ditampilkan di layar.

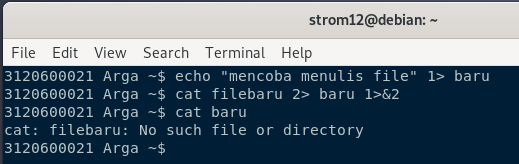
1. Notasi 1>&2 (atau >&2) : pembelokan standar output adalah sama dengan file descriptor 2 yaitu standar *error*
2. Command

$ echo “mencoba menulis file” 1> baru

$ cat filebaru 2> baru 1>&2

$ cat baru

1. Capture Output



1. Analisa

Pertama memasukkan text “mencoba menulis file” ke file *baru*, kemudian kita lakukan instruksi “cat *filebaru* 2> *baru* 1>&2”, maksudnya kita memasukkan hasil *error* di file *baru* dengan mengganti isi file *baru* semula. Sehingga ketika kita melihat ulang isi file *baru* setelah melakukan perintah diatas, yang muncul bukan tulisan “mencoba menulis file” melainkan pesan error akibat pembelokan output tadi.

1. Notasi >> (append)
2. Command

$ echo “kata pertama” > surat

$ echo “kata kedua” >> surat

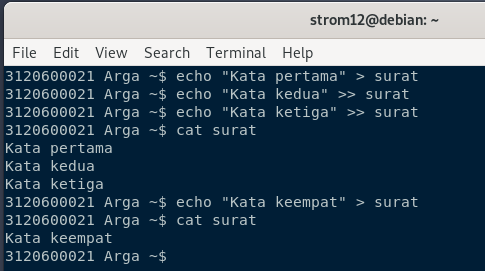
$ echo “kata ketiga” >> surat

$ cat surat

$ echo “kata keempat” > surat

$ cat surat

1. Capture Output



1. Analisa

Perintah *echo* digunakan untuk menuliskan sesuatu ke dalam file. Pada percobaan di atas, pertam kita menuliskan “Kata pertama” ke dalam file surat dengan menggunakan perintah *echo* untuk menuliskan dan perintah “*>”* untuk menyimpan tulisan ke dalam file surat. Untuk kata ke-dua dan ke-tiga, kita akan menambahkannya ke dalam file surat dengan mneggunakan tanda “*>>*” (*append*) yang akan menambahkan apa yang kita tuliskan dengan perintah *echo* ke dalam file surat. Untuk kata ke-empat, kita menambahkannya dengan tanda “*>”*, sehingga kata ke-empat akan menimpa kata yang telah ditulis sebelumnya.

1. Notasi here document (<<++….++) digunakan sebagai pembatas input dari keyboard. Perhatikan bahwa tanda pembatas dapat digantikan dengan tanda apa saja, namun harus sama dan tanda penutup harus diberikan pada awal baris.
2. Command

$ cat <<++

Hallo, apa kabar ?

Baik-baik saja ?

Ok!

++

$ cat <<%%%

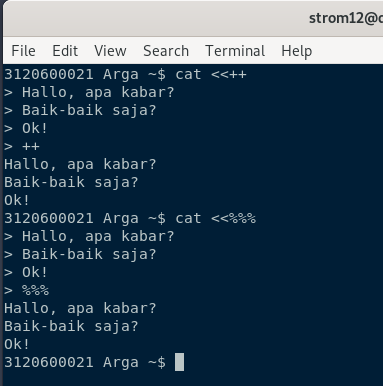
Hallo, apa kabar ?

Baik-baik saja ?

Ok!

%%%

1. Capture Output



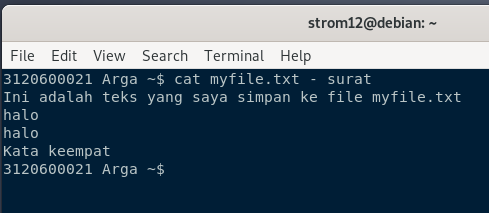
1. Analisa

Pada percobaan di atas, kita dapat mengambil beberap baris input dengan menggunakan perintah *& cat <<* yang diikuti dengan tanda pembuka (ex: ++, %%%, dll). Dengan perintah <<++ kita dapat menulsikan kalimat yang kita inginakan, jika kalimat sudah cukup, kita hanya perlu memberi tanda penutup ++ (karena pembuka juga menggunakan ++), dimana tanda penutup ini harus sama dengan tanda pembuka, setelah itu, maka kalimat yang kita inputkan akan langsung tampil di layar sebagai output.

1. Notasi – (input keyboard) adalah representan input dari keyboard. Artinya menampilkan file 1, kemudian menampilkan input dari keyboard dan menampilkan file 2. Perhatikan bahwa notasi “-“ berarti menyelipkan input dari keyboard
2. Command

$ cat myfile.txt – surat

1. Capture Output



1. Analisa

Pada percobaan di atas, kita akan menampilkan isi dari file *myfile.txt* dan file surat, tetapi sebelum menampilkan isi dari file surat, kita ingin mengambil input dari keyboard terlebih dahulu, maka kita dapat menggunakan tanda “ - “ untuk menyelipkan input dari keyboard dan kata yang dituliskan akan langsung dtampilkan sebagai output. Saat kita menekan *ctrl+d* (menghentikan input), maka isi file surat akan ditampilkan.

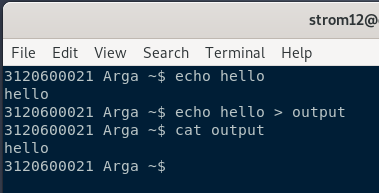
1. Untuk membelokkan standart output ke file, digunakan operator >
2. Command

$ echo hello

$ echo hello > output

$ cat output

1. Capture Output



1. Analisa

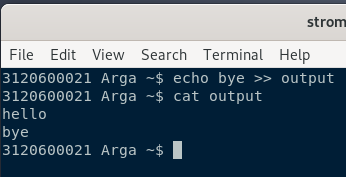
Pada percobaan di atas, kita akan membelokkan standart output ke dalam file ‘*output’*. Saat kita coba menggunakan perintah *$ echo*, maka apa yang kita tuliskan akan langsung ditampilkan sebagai output, tetapi jika kita ingin membelokkan outputnya ke dalam file *‘output’*, maka kita dapat menggunakan tanda *“>”* diikuti dengan nama file, sehingga outpunya tidak langsung ditampilkan di layar. Saat kita membuka file *‘output’* maka output dari perintah sebelumnya akan tampil.

1. Untuk menambahkan output ke file digunakan operator >>
2. Command

$ echo bye >> output

$ cat output

1. Capture Output



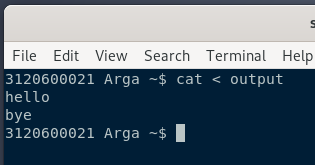
1. Analisa

Untuk menambahkan output ke dalam suatu file, kita dapat menggunakan tanda *“>>”* (*append*) yang akan menambah hasil output ke dalam file yang diinginakan.

1. Untuk membelokkan standart input digunakan operator <
2. Command

$ cat < output

1. Capture Output



1. Analisa

Operator “<” digunakan untuk membelokkan standard input, pada percobaan di atas kita membelokkan standard input yang awalnya dari keyboard menjadi dari file.

1. Pembelokan standart input dan standart output dapat dikombinasikan tetapi tidak boleh menggunakan nama file yang sama sebagai standart input dan output
2. Command

$ cat < output > out

$ cat out

$ cat < output >> out

$ cat out

$ cat < output > output

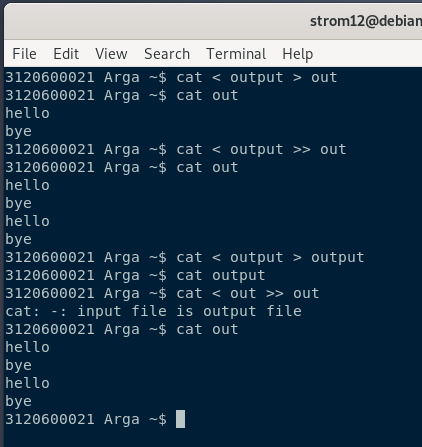
$ cat output

$ cat < out >> out (Proses tidak berhenti)

[Ctrl-c]

$ cat out

1. Capture Output



1. Analisa

* $ cat < output > out : menjadikan file output sebagai pembelok standar input sekaligus sebagai input dari file out
* $ cat out : menampilkan isi dari file out
* $ cat < output >> out : menjadikan file output sebagai pembelok standar input sekaligus menambahkan isi file output ke file out
* $ cat < output > output : menjadikan file output sebagai pembelok standar input sekaligus sebagai input dari file output. Karena isi dari file output kosong (sudah dipindahkan ke file out), maka ketika ditampilkan tidak menghasilkan apa-apa.
* Ketika kita menggunakan “$ cat < out >> out” ada error bahwa input merupakan output dari file.
* Percobaan 3 : Pipa (*pipeline*)

1. Operator pipa (|) digunakan untuk membuat eksekusi proses dengan melewati data langsung ke data lainnya.
2. Command

$ who

$ who | sort

$ who | sort -r

$ who > tmp

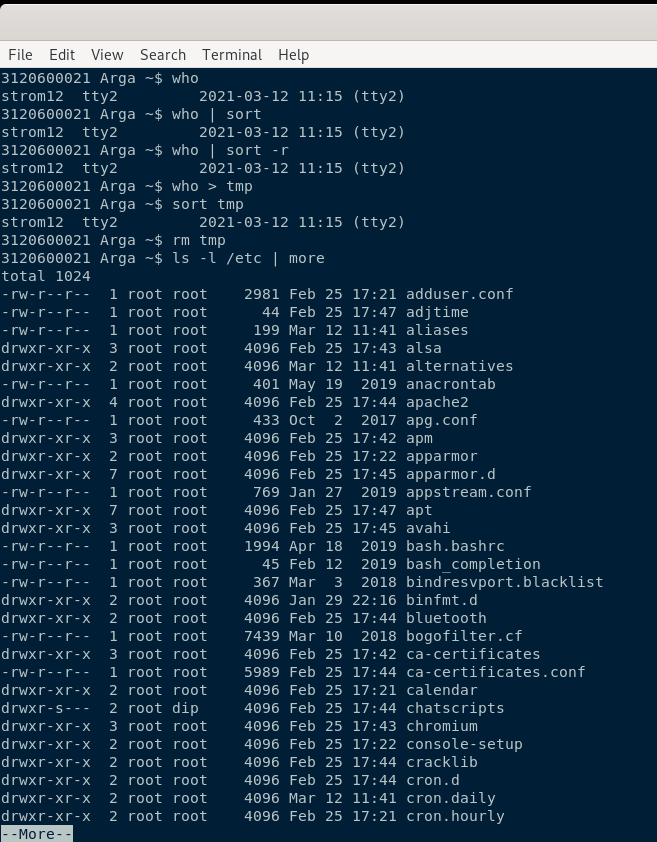
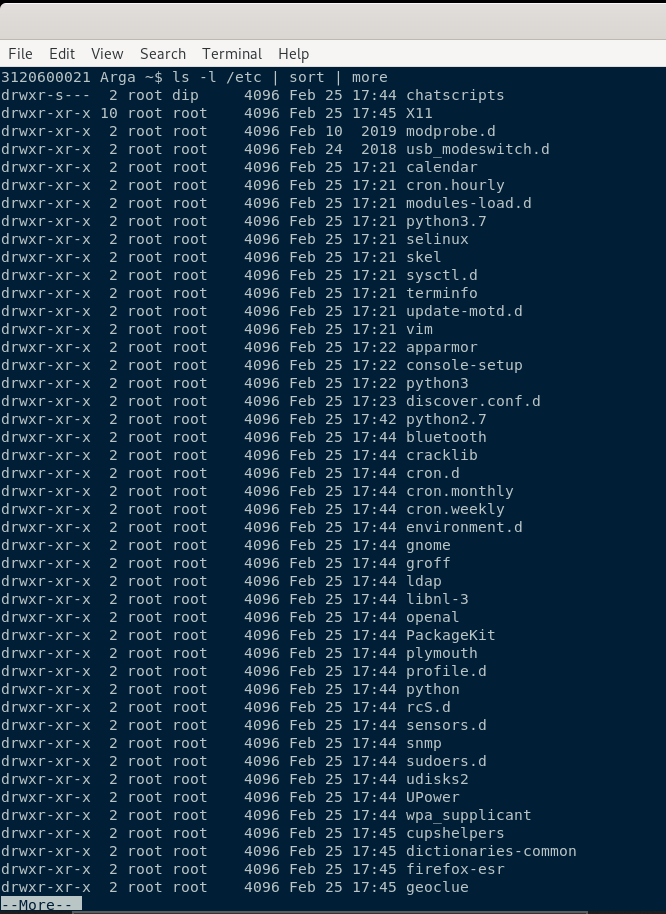
$ sort tmp

$ rm tmp

$ ls -l /etc | more

$ ls -l /etc | sort | more

1. Capture Output



1. Analisa

* $ who | sort digunakan untuk mengurutkan keluaran dari perintah *who* secara *ascending*
* $ who | sort -r digunakan untuk mengurutkan keluaran dari perintah *who* secara *descending*
* $ who > tmp memindahkan output ke file *tmp*
* $ *sort tmp* untuk mengurutkan isi file *tmp*
* $ *rm tmp* untuk menghapus file *tmp*
* $ *ls -l /etc | more* untuk menampilkan isi dari /etc per satu layar penuh
* $ *ls -l /etc | sort | more* untuk menampilkan isi dari /etc secara terurut per satu layar penuh.
* Percobaan 4 : Filter

1. Pipa juga digunakan untuk mengkombinasikan utilitas system untuk membentuk fungsi yang lebih kompleks
2. Command

fungsi yang lebih kompleks

$ w –h | grep <user>

$ grep <user> /etc/passwd

$ ls /etc | wc

$ ls /etc | wc –l

$ cat > kelas1.txt

Badu

Zulkifli

Yulizir

Yudi

Ade

[Ctrl-d]

$ cat > kelas2.txt

Budi

Gama

Asep

Muchlis

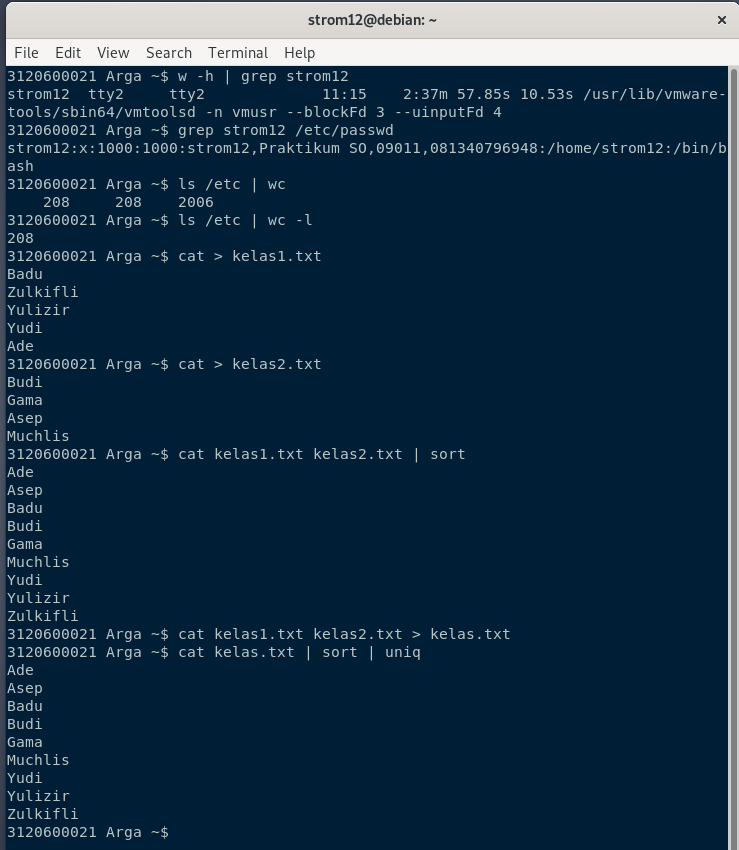
[Ctrl-d]

$ cat kelas1.txt kelas2.txt | sort

$ cat kelas1.txt kelas2.txt > kelas.txt

$ cat kelas.txt | sort | uniq

1. Capture Output



1. Analisa

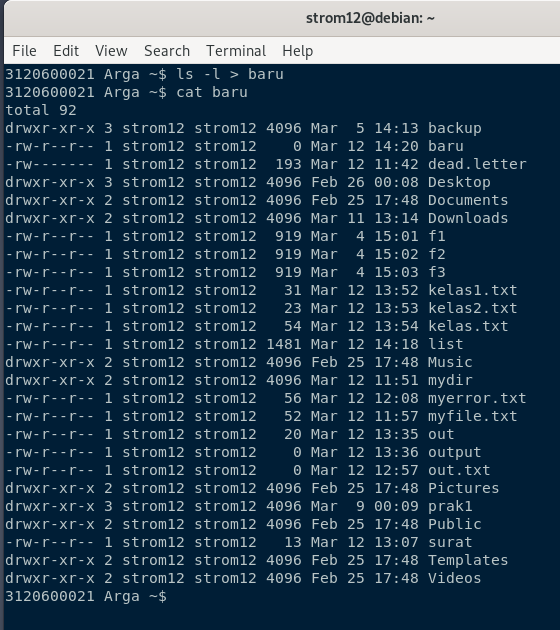
Pada percobaan di atas, pertama kita akan menampilkan informasi dari user yang mengandung kata “strom12” dengan perintah *w -h | grep strom12.* Lalu, mencari dalam file */etc/passwd* yang mangandung kata “strom12” dengan perintah *grep strom12 /etc/passwd*. Setelah itu, kita akan menghitung berapa jumlah baris, kata dan karakter dalam suatu file dengan melakukan perintah filter yaitu *wc*, jika kita hanya menghitung jumlah barisnya dapat menambahkan parameter *-l* pada perintah *wc*. Selanjutnya, kita akan membelokkan standar output ke file *kelas1.txt* dan *kelas2.txt* dengan menggunakan perintah *cat > kleas1.txt* dan *cat > kelas2.txt,* maka outuput akan disimpan kedalam file tersebut. Kita akan mengurutkan isi dari *kelas1.txt* dan *kelas2.txt* dengan perintah sort. Terakhir, kita akan mebelokkan standard output dari *kelas1.txt*  dan *kelas2.txt* ke dalam *kelas.txt* dengan perintah *cat kelas1.txt kelas2.txt > kelas.txt*, setelah itu, kita akan mengurutkan dan melakukan filter agar tidak terjadi duplikasi dengan perintah *cat kelas.txt | sort | uniq*, maka isi dari file *kelas.txt* akan tampil dengan terurut dan tidak ada yang didupliaksi.

1. Latihan
2. Lihat daftar secara lengkap pada direktori aktif, belokkan tampilan standard output ke file baru.
3. Command

$ ls -l > baru

$cat baru

1. Capture Output



1. Analisa

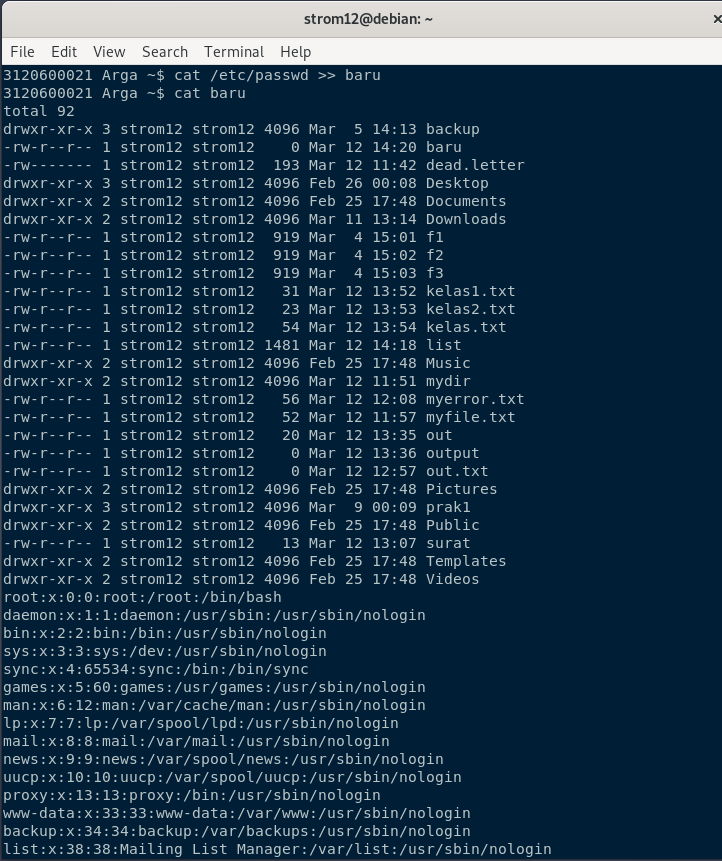
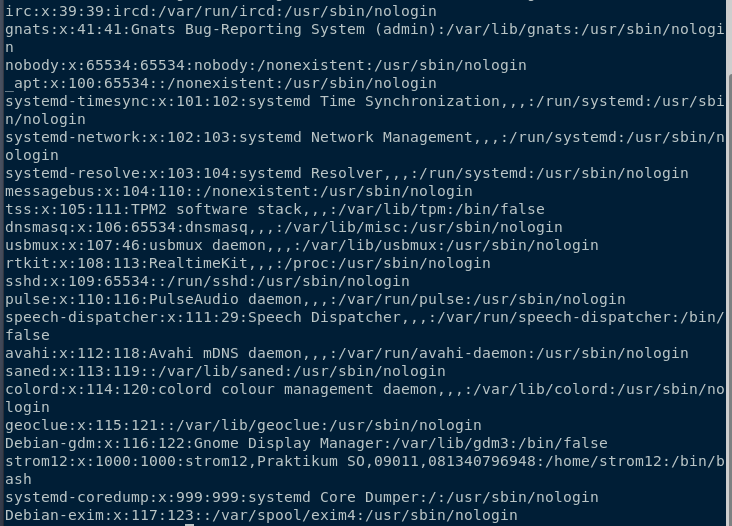
Untuk membelokkan standart tampilan output kita dapat menggunakan tanda *“>”* diikuti dengan nama file, sehingga output akan disimpan ke dalam file yang diinginkan, saat file dibuka, maka output dari perintah sebelumnya akan ditampilkan.

1. Lihat daftar secara lengkap pada direktori /etc/passwd, belokkan tampilan standard output ke file baru tanpa menghapus file baru sebelumnya.
2. Command

$ cat /etc/passwd >> baru

$cat baru

1. Capture Output



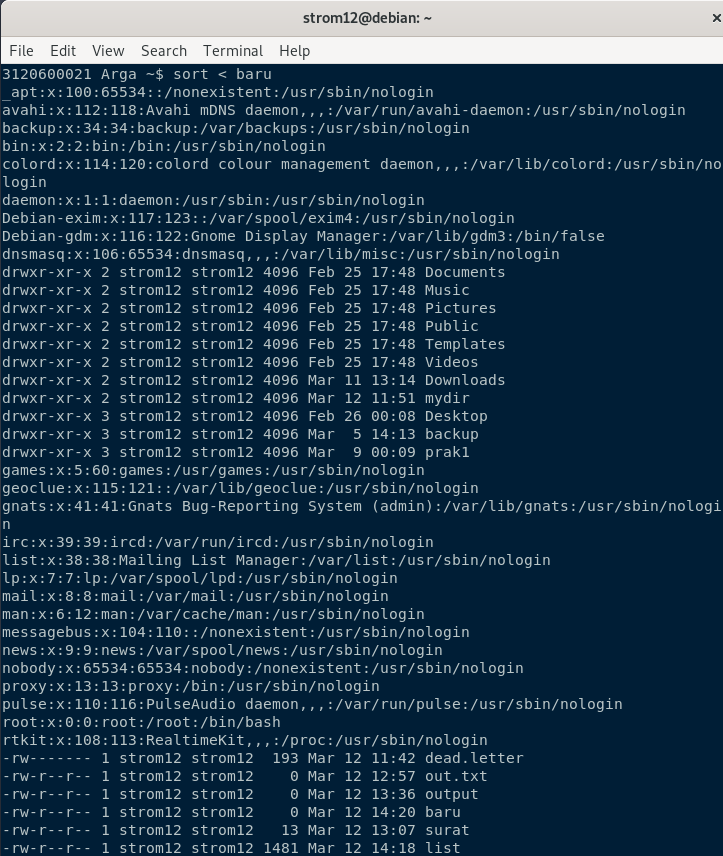
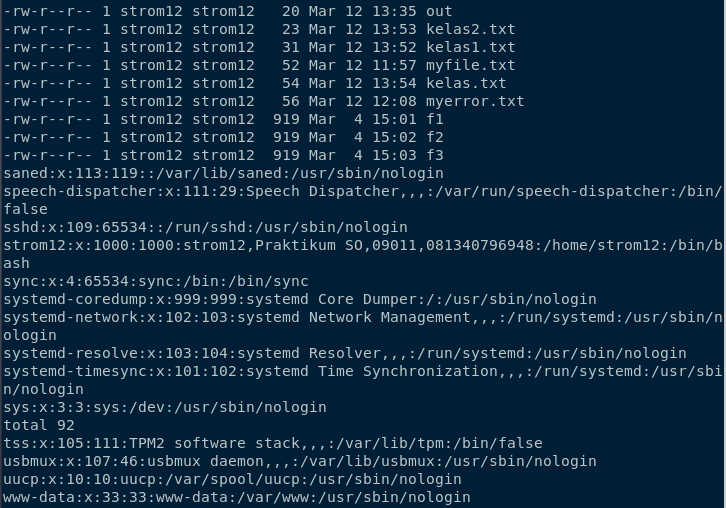
1. Analisa

Kita dapat menggunakan “>>” untuk membelokkan tampilan standar output tanpa mengahapus isi sebelumnya dari file. Tanda “>>” akan menambah isi dari file.

1. Urutkan file baru dengan cara membelokkan standard input.
2. Command

$ sort < baru

1. Capture Output



1. Analisa

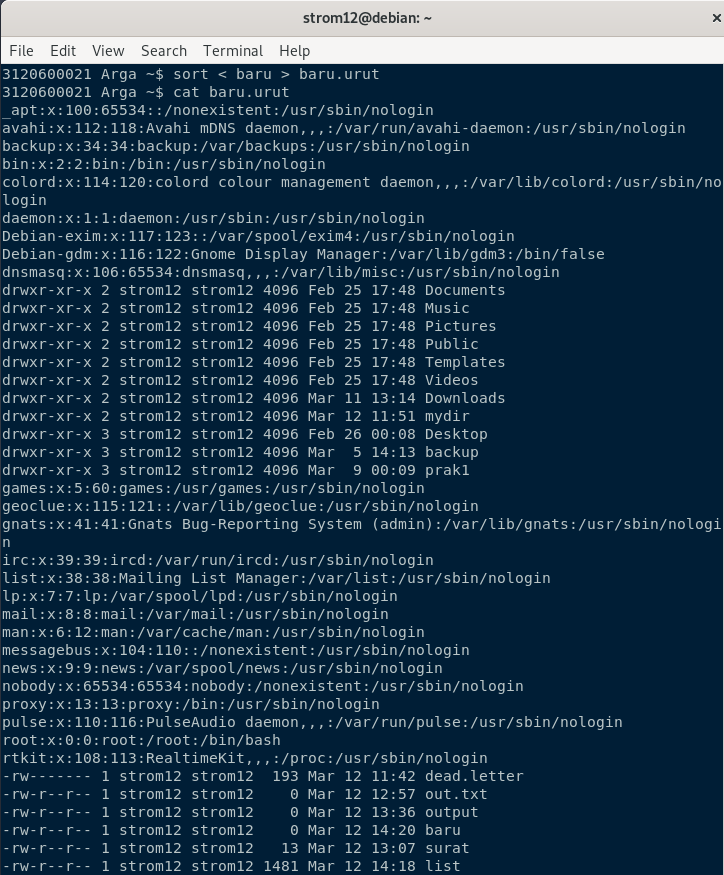
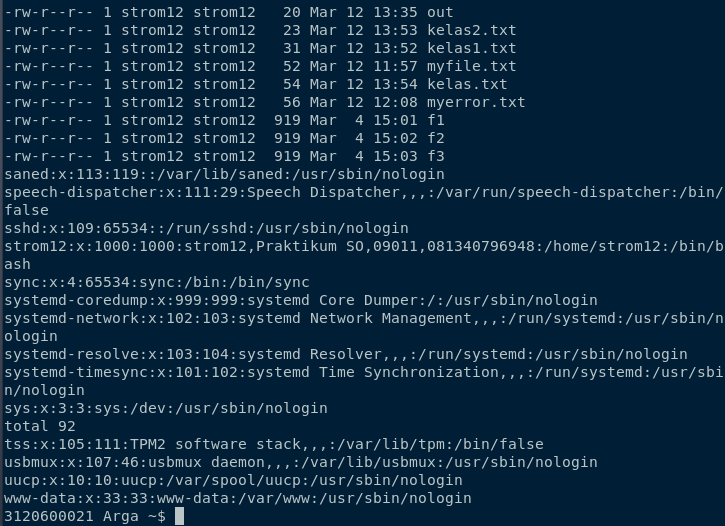
Untuk mengurutkan file, kita dapat menggunakan perintah *sort* dan jika ingin membelokkan standar input, kita dapat menggunakan tanda “<”

1. Urutkan file baru dengan cara membelokkan standard input dan standard output ke file baru.urut .
2. Command

$ sort < baru > baru.urut

$ cat baru.urut

1. Capture Output



1. Analisa

Pertama kita mengurutkan file *baru* dengan menggunakan perintah *sort* dan tanda “<” untuk membelokkan standart input, setelah itu, kita belokkan standar output ke dalam file *baru.urut* dengan menggunakan tanda “>”, maka isi dari file *baru* yang sudah terurut akan disimpan di dalam file baru.urut

1. Buatlah direktori latihan2 sebanyak 2 kali dan belokkan standart *error* ke file rmdir*error*.txt .
2. Command

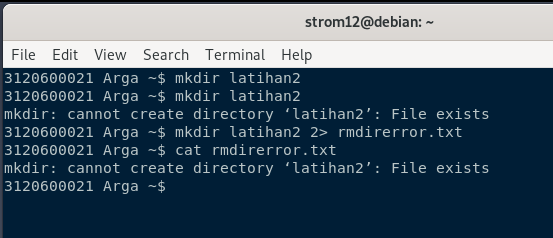
$ mkdir latihan2

$ mkdir latihan2 (error)

$ mkdir latihan2 2> rmdirerror.txt

$cat rmdirerror.txt

1. Capture Output



1. Analisa

Kita dapat membelokkan standart *error* dengan menggunakan tanda “2>” diikuti nama file untuk menyimpan error. Pada latihan di atas kita akan membelokkan standart *error* ke dalam file *rmdirerror.txt* sehingga pesan *error* tidak langsung tampil di layar. Ketik kita membuka file *rmdirerror.txt,* maka pesan *error* untuk perintah sebelumnya akan ditampilkan.

1. Urutkan kalimat berikut :

Jakarta

Bandung

Surabaya

Padang

Palembang

Lampung

Dengan menggunakan notasi here document (<@@@...@@@)

1. Command

$ sort <<@@@

> Jakarta

> Bandung

> Surabaya

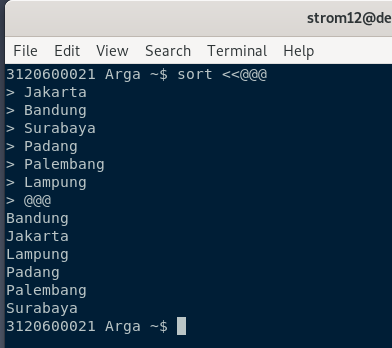
> Padang

> Palembang

> Lampung

> @@@

1. Capture Output



1. Analisa

Kita dapat mengurutkan dengan meggunakan perintah *sort* dan untuk menulis beberap kalimat/kata, kita dpaat menggunakan notasi *here document,*

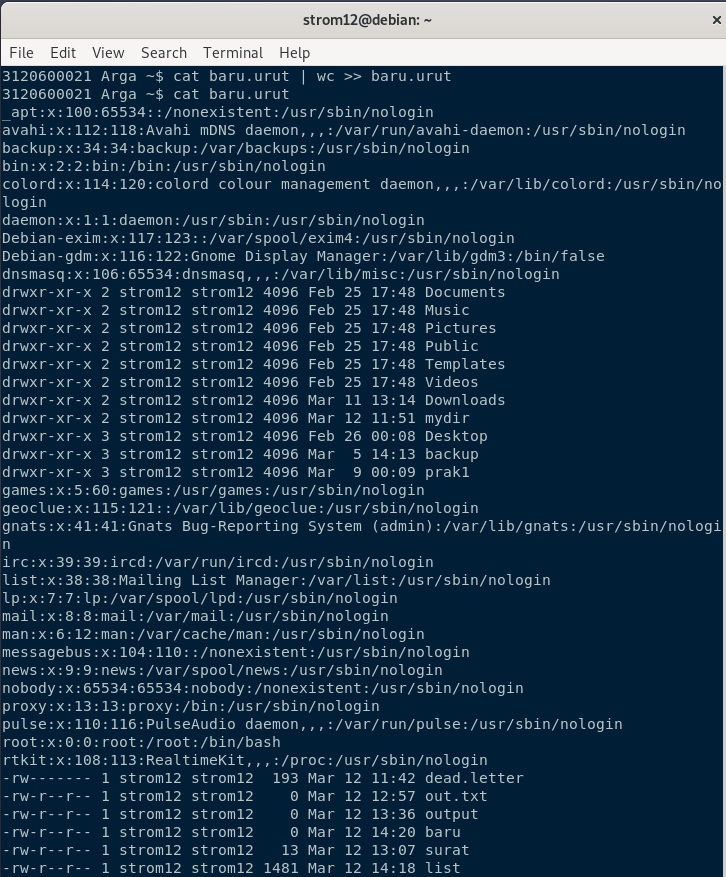
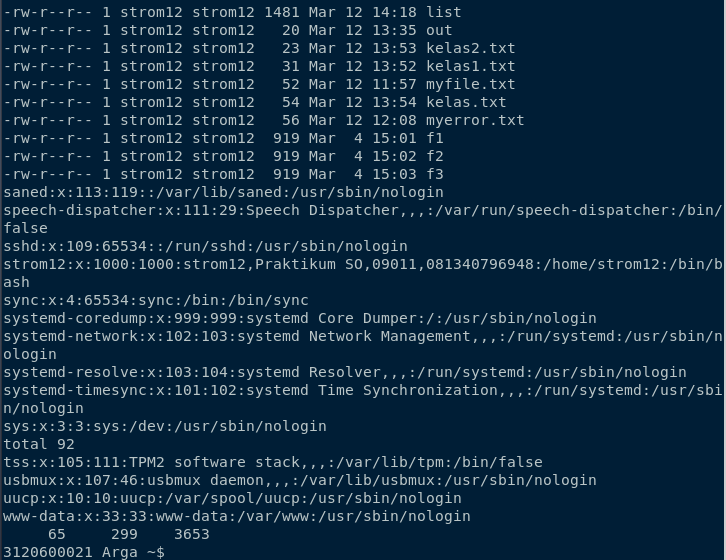
disini kita menggunakan “<<@@@....@@@”, maka kata yang kita tuliskan akan langsung ditampilkan dan diurut setelah kita mengetikkan @@@ sebagai penutup

1. Hitung jumlah baris, kata dan karakter dari file baru.urut dengan menggunakan filter dan tambahkan data terseubt ke file baru .
2. Command

$ cat baru.urut | wc >> baru.urut

$ cat baru.urut

1. Capture Output



1. Analisa

Untuk menghitung jumlah baris, kata dan karakter, kita dapat menggunakan perintah filter yaitu *wc*, dan untuk menambah hasilnya kedalam file *baru.urut*, kita dapat menggunakan tanda “>>”, maka hasil dari perhitungan tadi akan disimpan ke dalam file *baru.urut*.

1. Gunakan perintah di bawah ini dan perhatikan hasilnya.

$ cat > hello.txt

Dog cat

Cat duck

Dog chicken

Chicken duck

Chicken cat

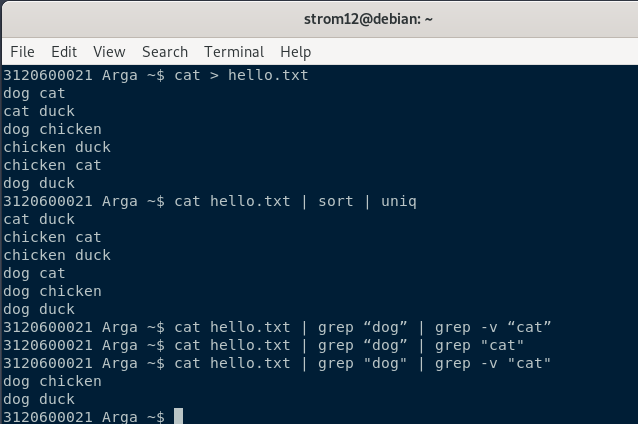
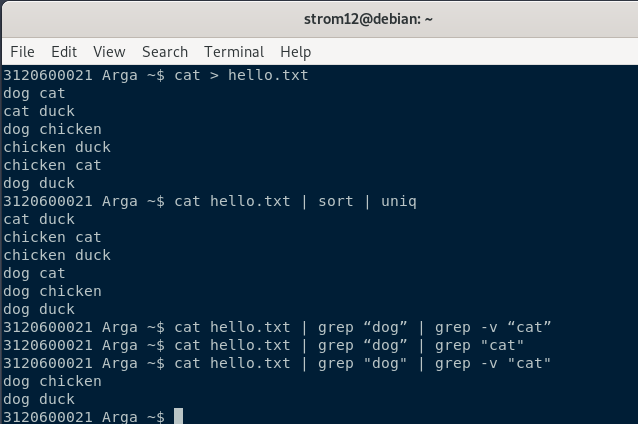
Dog duck

*[Ctrl-d]*

$ cat hello.txt | sort | uniq

$ cat hello.txt | grep “dog” | grep -v “cat”

1. Capture Output



1. Analisa

Untuk perintah di atas, pertama kita akan melakukan penmbelokan standart output ke dalam file *hello.txt*, setelah itu kita akan mengurutkan isi dari filenya dan mengecek apakah ada yang sama/didupliaksi dengan menggunakan *sort* | *uniq*. Terakhir, kita akan menampilkan isi dari file dengan melakukan filter, dimana kita hanya ingin menampilkan teks yang mengandung “*dog*” dan tidak mengandung “*cat*” dengan cara *grep “dog” | grep -v “cat”.*

1. Kesimpulan

Kita dapat mebelokkan standard input yang awalnya berasal dari keyboard menjadi dari file, begitu pula dengan output, yang awalnya pada layar menjadi pada file. Kita dapat menggunakan perintah $cat untuk melihat isi dari suatu file. Kita juga dapat melakukan filter file dengan menggunakan *sort, uniq, grep*, dll.

Untuk melakukan beberapa perintah secara berurutan kita dapat menggunakan *pipeline* dimana output dari perintah pertama akan menjadi input pada perintah selanjutnya. Kita juga dapat melakukan pembelokan (*redirection*) dengan mengalihkan file *descriptor*, symbol yang dapat digunakan yaitu:

* 0< untuk standard input
* 1< untuk standard output
* 2< untuk standard error