5. 基本的なコマンド 2

5.1 ファイルのタイムスタンプの変更 (touch)

ファイルにはタイムスタンプ (最終更新日) が存在する． `touch` コマンドは，最終更新時刻を更新する．

書式:

```
touch [option] ファイル名
```

- `touch` コマンドを実行すると，ファイルのタイムスタンプが現在日時に更新される．
  - option により，新しいタイムスタンプとなる日時を指定できる．
- ファイルが存在しない場合， `touch` コマンドによって中身が空である0バイトのファイルを作成する．

実行例:

```
ai@ai-VirtualBox:~$ sudo cp /etc/hosts /etc/hosts.bk
ai@ai-VirtualBox:~$ ls -l /etc/hosts.bk
-rw-r--r-- 1 root root 228 11月 16 19:57 /etc/hosts.bk
ai@ai-VirtualBox:~$ date
2021年 11月 16日 火曜日 19:58:24 JST
ai@ai-VirtualBox:~$ sudo touch /etc/hosts.bk
ai@ai-VirtualBox:~$ ls -l /etc/hosts.bk
-rw-r--r-- 1 root root 228 11月 16 19:58 /etc/hosts.bk
```

実行例 (`touch` コマンドによるファイル作成):

```
ai@ai-VirtualBox:~/Documents/5_touch_workspace$ ls
ai@ai-VirtualBox:~/Documents/5_touch_workspace$ touch touched-file
ai@ai-VirtualBox:~/Documents/5_touch_workspace$ ls -l
total 0
-rw-rw-r-- 1 ai ai    0 11月 16 19:59 touched-file
```

5.2 ファイルの一部の取得 (head , tail)

ファイルの先頭や末尾など，一部分のみを見る場合は `head` コマンドや `tail` コマンドが使える．

`head` コマンド

書式:

```
head [option] ファイル名
```

option:

```
-n 行
先頭から指定した行を標準出力する

-c バイト
先頭から指定したバイト分を標準出力する
```

- オプションを付けない場合は，先頭から10行を標準出力する．
- ファイル名には対象のファイル名を入力する．
    - ファイル名の部分を空白にした場合，もしくは `-` を指定した場合は，標準入力からのデータに対して処理を行う．

実行例:

```
ai@ai-VirtualBox:~$ head ~/.bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac
```

ここで，以下の3つのコマンドの実行結果は同一になる．

```
head FILE
cat FILE | head -
cat FILE | head
```

`tail` コマンド

書式:

```
tail [option] ファイル名
```

option:

```
-n 行
末尾から指定した行を標準出力する

-c バイト
末尾から指定したバイト分を標準出力する
```

- オプションを付けない場合は末尾から10行を標準出力する
- ファイル名には対象のファイル名を入力する
    - ファイル名の部分を空白にした場合，もしくは `-` を指定した場合，標準入力からのデータに対して処理を行う．

したがって，以下の3つのコマンドの実行結果は同一になる．

```
tail FILE
cat FILE | tail -
cat FILE | tail
```

特別なオプション `-f`

`tail` は本来，ファイルの末尾を表示するコマンド．ファイルによっては，末尾が随時変わることがある．

`tail` には `-f` というオプションがある．このオプションを付けることで，変更をリアルタイムにモニタすることが可能．
(e.g. ログファイルのモニタ)

書式:

```
tail -f ファイル名
```

実行例:

```
ai@ai-VirtualBox:~/Documents/5_tail_workspace$ man less > ~/manual-less
ai@ai-VirtualBox:~/Documents/5_tail_workspace$ tail -n 5 manual-less
       Report bugs at https://github.com/gwsw/less/issues.
       For more information, see the less homepage at
       http://www.greenwoodsoftware.com/less.

                         Version 551: 11 Jun 2019                LESS(1)


ai@ai-VirtualBox:~/Documents/5_tail_workspace$ tail -f -n 5 manual-less
       Report bugs at https://github.com/gwsw/less/issues.
       For more information, see the less homepage at
       http://www.greenwoodsoftware.com/less.

                         Version 551: 11 Jun 2019                LESS(1)
```

ここで，新しい端末で

```
ai@ai-VirtualBox:~/Documents/5_tail_workspace$ echo 'Hello' >> manual-less
```

とすると，manual-less にデータが追加され， `tail-f` を実行している端末は以下のようになる．

```
ai@ai-VirtualBox:~/Documents/5_tail_workspace$ tail -f -n 5 manual-less
       Report bugs at https://github.com/gwsw/less/issues.
       For more information, see the less homepage at
       http://www.greenwoodsoftware.com/less.

                         Version 551: 11 Jun 2019                LESS(1)
Hello
```

このファイルの読み込みは，ctrl + c が押され，処理が中断されるまで継続される．
つまり， `tail-f` を用いると，ファイルを動的に読み込むことが可能であり，Web サーバのアクセスログやエラーログを見るのに便利である．

5.3 テキストファイルのソート (sort)

テキストファイルの中身をソートするには `sort` コマンドが使える．オプションでソートの順序を指定できる．

書式:

```
sort [option] ファイル名
```

option:

```
-r
逆順でソートする

-k n
n 列目のデータをソートする

-n
数値としてソートする
```

ファイルの準備

ソートの機能を確認するために，サンプルとなるファイルを作成する．

実行例:

```
ai@ai-VirtualBox:~/Documents/5_sort_workspace$ cat > score
yoshinori kawazu        85
keiichi oka     70
toru minemura   100
```

`sort` の実行

まず，オプションを付けずにソートする．

実行例 (オプションを付けずにソート):

```
ai@ai-VirtualBox:~/Documents/5_sort_workspace$ sort score
keiichi oka     70
toru minemura   100
yoshinori kawazu        85
```

オプションを付けない場合，各行の1文字目をアルファベット順にソートしている．

実行例 (`-r` オプションを付けた逆順によるソート):

```
ai@ai-VirtualBox:~/Documents/5_sort_workspace$ sort -r score
yoshinori kawazu        85
toru minemura   100
keiichi oka     70
```

アルファベットの逆順にソートされる.

n 列目のデータソート (-k)

`sort` では, `-k` オプションを付けると，ソートの鍵として利用する列の番号を指定する.

実行例 (2列目のデータソート)

```
ai@ai-VirtualBox:~/Documents/5_sort_workspace$ sort -k 2 score
yoshinori kawazu        85
toru minemura   100
keiichi oka     70
```

数値式でのソート

```
ai@ai-VirtualBox:~/Documents/5_sort_workspace$ sort -k 3 score
toru minemura   100
keiichi oka     70
yoshinori kawazu        85
```

- キーとして 3 を指定すると，3列目が "100 -> 70 -> 85" としてソートされている.
- これは，数字が文字として認識されているため (辞書式).
- 数値式ソートにするためには, `sort` に `-n` オプションをつける.

```
ai@ai-VirtualBox:~/Documents/5_sort_workspace$ sort -n -k 3 score
keiichi oka     70
yoshinori kawazu        85
toru minemura   100
```

5.4 行の重複の消去 (uniq)

uniq コマンドを使うことで，直前の行と同じ内容があった場合，対象行を出力しない．つまり，連続している同じ内容の行を1行にまとめられる．

書式:

```
uniq ファイル名
```

実行例 (ファイル作成):

```
ai@ai-VirtualBox:~/Documents/5_uniq_workspace$ cat > uniq-sample
AAA
BBB
AAA
CCC
DDD
```

実行例 (uniq コマンドの実行)

```
ai@ai-VirtualBox:~/Documents/5_uniq_workspace$ uniq uniq-sample
AAA
BBB
AAA
CCC
DDD
```

2行連続していた "CCC" が1行にまとめられた．

5.5 文字列の置き換え (tr)

`tr` コマンドを使って，標準入力からのデータを文字毎に置き換える (TRanslate) ことができる.

書式:

```
tr 文字列1 文字列2
```

実行例:

```
cat FILE | tr abc ABC
```

- `cat` コマンドで FILE を開く
- `tr` コマンドで a,b,c をそれぞれ A,B,C に置き換える.

実行例 (ファイルの作成):

```
ai@ai-VirtualBox:~/Documents/5_tr_workspace$ cat > translate
Android
iPhone
Windows Phone
```

実行例 (`tr` コマンドによる置き換え):

```
ai@ai-VirtualBox:~/Documents/5_tr_workspace$ cat translate | tr on ON
ANdrOid
iPhONe
WiNdOws PhONe
```

実行例 (`tr` の結果をリダイレクトに使ったファイル出力):

```
ai@ai-VirtualBox:~/Documents/5_tr_workspace$ cat translate | tr on ON > translate2
ai@ai-VirtualBox:~/Documents/5_tr_workspace$ cat translate2
ANdrOid
iPhONe
WiNdOws PhONe
```

5.6 ファイルの比較 (diff)

変更の有無を調べる場合に用いられる． diff の結果は標準出力されるが，リダイレクトすればファイルに出力することもできる．

書式:

```
diff [option] ファイル1 ファイル2
```

option:

```
-c
context diff 形式で差分を出力する

-u
unified diff 形式で差分を出力する
```

実行例:

```
diff file1 file2
diff -c file1 file2
diff -u file1 file2
```

実行例 (ファイルの作成):

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ echo 'test text' > file1
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ echo 'test text' > file2
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ echo 'new line' >> file2
```

実行例 (diff コマンドによる比較):

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ diff file1 file2
1a2
> new line
```

実行例 (diff -u コマンドによる比較):

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ diff -u file1 file2
--- file1       2021-11-16 21:41:09.231759717 +0900
+++ file2       2021-11-16 21:41:26.607370568 +0900
@@ -1 +1,2 @@
 test text
+new line
```

実行例 (`diff -c` コマンドによる比較):

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ diff -c file1 file2
*** file1       2021-11-16 21:41:09.231759717 +0900
--- file2       2021-11-16 21:41:26.607370568 +0900
***************
*** 1 ****
--- 1,2 ----
  test text
+ new line
```

- オプションを付けずに実行すると，異なる部分を標準出力する．
- オプションを付けると，ファイルの更新時刻を含めた多くの差分情報が出力される．

次に，file2 を以下のように置き換えて `diff` コマンドを実行する．

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ echo 'overwrite text' > file2
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ echo 'new line' >> file2
```

実行例 (`diff` コマンドによる比較):

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ diff file1 file2
1c1,2
< test text
---
> overwrite text
> new line
```

実行例 (`diff -u` コマンドによる比較):

```
ai@ai-VirtualBox:~/Documents/5_diff_workspace$ diff file1 file2
1c1,2
< test text
---
> overwrite text
> new line
```

- 削除された行の頭にマイナス記号，追加された行にプラス記号が付加されて標準出力される．
    - この例では，file1 と file2 の共通していた文字列 'test text' が削除され，新たに 'overwrite text' と 'new line' という行が追加されたことがわかる．
- オプション `-u` は文字列の増減が激しい，文書やプログラムのソースコードを比較する際に有用である．