工学基礎実験実習
ファイル操作とシェル 第 2 回レポート

氏名: 重近 大智 (SHIGECHIKA, Daichi)
学生番号: 09501527

出題日: 2019 年 5 月 21 日
提出日: 2019 年 5 月 24 日
締切日: 2019 年 5 月 28 日

1　はじめに

ファイルリダイレクション，パイプ，シェルスクリプトやコマンドについて記述する. 機能及び具体例についても記述する.

2　ファイルリダイレクションとパイプ

2.1　リダイレクション

2.1.1　標準出力のリダイレクション

機能 一般的にコマンドは実行すると端末上に結果が表示されるが，>記号を使うことで，ファイルに結果を出力することができる. 具体的には以下のようにする.

具体例 cat コマンドによって出力される「solarsystem1」の表示結果を「datalist」に書き込む場合を考える.

```
$ cat solarsystem1 > datalist
$ cat datalist
Sun      0.000  1304000      1.41
Mercury  0.579        0.056  5.43
Venus    1.082        0.857  5.24
Earth    1.496        1.000  5.52
Mars     2.279        0.151  3.93
$
```

上記のようにすると，cat コマンドによって出力された「solarsystem1」の内容が「datalist」に出力される.

2.1.2 標準入力のリダイレクション

機能 逆に、<記号を用いることで，データなどをファイルから入力することができる．具体的には以下のようにする．

具体例 sort コマンドの並びかえの対象を「solarsystem1」から入力し，2 番目の項目を降順に並べる場合を考える．

```
$ sort -k 2 -nr < solarsystem1
Mars     2.279        0.151  3.93
Earth    1.496        1.000  5.52
Venus    1.082        0.857  5.24
Mercury  0.579        0.056  5.43
Sun      0.000  1304000      1.41
$
```

上記のようにすると，sort コマンドの並びかえの対象を「solarsystem1」から入力することができる．

2.2 パイプ

機能 あるコマンドの出力結果を別のコマンドに入力する場合，別のファイルを経由せず直接入力することができる．この時にパイプと呼ばれる機能を利用する．

具体例 「solarsystem1」というファイルの表示結果を sort コマンドの入力にする場合を考える．

```
$ cat solarsystem1 | sort -k 4 -nr
Earth    1.496        1.000  5.52
Mercury  0.579        0.056  5.43
Venus    1.082        0.857  5.24
Mars     2.279        0.151  3.93
Sun      0.000  1304000      1.41
$
```

上記のようにすると，cat コマンドによって出力された「solarsystem1」というファイルの表示結果が，sort コマンドに入力される．

3 コマンドと実行可能ファイルの関係

3.1 ソースファイルの作成

「Windows7」と表示する C 言語のプログラムのソースファイルを作ることを考える．ファイル名は「Windows7.c」とし，内容は以下の通りである．

```
1 #include<stdio.h>
  2 int main(void)
  3 {
  4     printf ("Windows7\n");
  5     return 0;
  6 }
```

3.2 ソースファイルのコンパイルと実行可能ファイルの実行

gcc コマンドを用いてコンパイルすると，実行可能ファイル「a.out」が作成され，これを端末上で実行すると以下のようになる．

```
$ gcc Windows7.c
$ ./a.out
Windows7
```

./a.out を実行すると，結果として「Windows7」が表示される．

3.3 シェルスクリプト

実行可能形式のファイルは，C 言語のソースファイルをコンパイルするだけでなく，シェルスクリプトによって作成することもできる．ただしシェルスクリプトを実行するには，事前に chmod コマンドを用いて実行パーミッションを付加する必要がある．

ここでは実行すると「Windows10」と表示するシェルスクリプト「Windows10.sh」を作成する．スクリプトは以下の通りである．

```
#!/bin/sh
echo "Windows10"
```

実行パーミッションを付加するには，次のように入力する．

```
$ chmod 755 Windows10.sh
```

C 言語の実行可能ファイル「a.out」と同様にシェルスクリプトを実行する．実行するには，次のように入力する．

```
$ ./Windows10.sh
Windows10
```

./Windows10.sh を実行すると，結果として「Windows10」が表示される．

4 その他のコマンド

ここでは前回のレポートに記述しなかったいくつかのコマンドを紹介する．今回はオプションの確認に man コマンドも用いた．

4.1 ps コマンド

ps コマンドの概要は，以下の通りである．

機能 実行中のプロセスを表示する．

形式 ps (option)

オプション
- -e： システム上の全てのプロセスを表示する．
- -u： root として実行されている全てのプロセスを表示する．
- -r： 実行中のプロセスのみ表示する．

使用例 $ ps

```
  PID TTY          TIME CMD
 4064 pts/0    00:00:41 emacs
24252 pts/0    00:00:00 ps
31768 pts/0    00:00:00 bash
32556 pts/0    00:00:22 evince
$
```

ps コマンドを実行すると，上記のような結果が得られた．この結果より，エディタ (emacs)，ps コマンド，evince コマンドなどが実行されていることが分かる．ps -l とすると，次のような詳細な表示になった．

```
$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  3327  4064 31768  0  80   0 - 288841 poll_s pts/0    00:00:42 emacs
4 R  3327 31162 31768  0  80   0 - 38304 -      pts/0    00:00:00 ps
0 S  3327 31768 31756  0  80   0 - 28758 do_wai pts/0    00:00:00 bash
0 S  3327 32556 31768  0  80   0 - 359568 poll_s pts/0   00:00:23 evince
$
```

4.2 which コマンド

which コマンドの概要は，以下の通りである．

機能 指定したコマンドに対応する実行可能形式のファイルのシステム上の場所を表示する．

形式 which (option) [directory name]

オプション
- -a： パスで合致するものをすべて表示する．

使用例 $ which -a emacs

```
/usr/bin/emacs
/bin/emacs
```

4.3 ＆コマンド

&コマンドの概要は，以下の通りである．

機能 プロセスやジョブをバックグラウンンドタスクとして実行する．

形式 [commmand] &

オプション オプションは見つからなかった．

使用例 $ emacs&
[2] 10263

4.4 ^Z コマンド

^Z コマンドの概要は，以下の通りである．

機能 フォアグラウンンドのプロセスやジョブを中断する．

形式 ^Z

オプション オプションは見つからなかった．

使用例 $ emacs
^Z
[3]+ 停止 emacs

4.5 bg コマンド

bg コマンドの概要は，以下の通りである．

機能 指定したジョブをバックグラウンドタスクに移行する．

形式 bg %[jobnumber]

オプション オプションは見つからなかった．

使用例 $ emacs
^Z
[1]+ 停止 emacs
$ bg %1
[1]+ emacs &

4.6 fg コマンド

fg コマンドの概要は，以下の通りである．

機能 指定したジョブをフォアグラウンドタスクに移行する．

形式 fg %[jobnumber]

オプション オプションは見つからなかった．

使用例 $ emacs

```
^Z[3]   終了                  emacs

[4]+  停止                  emacs
$ bg %4
[4]+ emacs &
$ fg %4
emacs
```

4.7 kill コマンド

kill コマンドの概要は，以下の通りである．

機能 指定したプロセスやジョブに指定されたシグナルを送る．

形式 kill [-signal] %[jobnumber or processID]

なお，シグナルを指定しなければ「SIGKILL」というシグナルが送られる．

オプション オプションは下記の通りである．

- -1： プロセス番号が 1 より大きい全てのプロセスに指定されたシグナルを送る．
- 0： 現在のプロセスグループの全てのプロセスにシグナルを送る．

使用例 $ emacs

```
^Z
[3]+  停止                  emacs
$ kill %3
```

5 alias の使用

機能 コマンドの別名を登録する．

形式 alias [alias name]="[program name]"

オプション

- -p： 登録されている alias の一覧を表示する．

使用例 fire と打つと firefox が起動するように alias を登録するには次のようにする．

```
$ alias fire="firefox"
```

alias 登録後，fire と打つと下記のように firefox が実行される．

```
$ fire
$ jobs
[1]-  実行中                 firefox &
[2]+  実行中                 evince report2_09501527.pdf &
$
```

alias を定義することで，firefox を簡単に起動することができた．

6　考察

本レポートの作成を通して，ファイルリダイレクション，パイプやシェルスクリプトなどのコマンドの使い方，またそのオプションについて理解することができた．C 言語プログラムのソースファイルやシェルスクリプトファイルの中身を一部書き換えることで，どのように動作が変わるのかを検証した．alias を用いることで一部のコマンドを全て打つ必要をなくすことができて，効率化につなげられると思った．

7　まとめ

本レポートでは，ファイルリダイレクション，パイプ，シェルスクリプトやコマンドをまとめた。またいくつかのオプションについては，端末から`--help`と`man`コマンドを用いて確認した．