情報科学演習 資料 4
シェルの入力支援機能とメタキャラクタ

令和 7 年 5 月 8 日

目 次

1　コマンド入力の支援機能

1.1　シェル

シェル (shell) は，ユーザーがコマンド行に打ち込んだコマンド等を解釈し，コンピュータに実行させる役割をもつプログラムです[1]。ユーザーがコンピュータにログインすると，そのユーザーのためにシェルが動き出します。これをログインシェル (login shell) といいます。ユーザーがコマンド行にコマンドをタイプして実行できるのは，シェルが動いているからです。

シェルには幾つかの種類があり，ユーザーがログインシェルとして何というシェルを用いるかは，予め管理者が設定しています。シェルが持つ機能や使い方はシェル毎に違いますので，`echo $SHELL` を実行して，利用中のシェルを確認しましょう[2]。/bin/bash と出力されますね。皆さんが使っているシェルは Bash です。

1.2　コマンド入力の支援機能

Bash が持つ機能の一例として，コマンド入力を助ける機能を幾つか紹介します。なお，ここで紹介する機能は，他の最近よく使われるシェルでも利用できます。

1.2.1　コマンド行の編集

プロンプトの存在する，コマンド入力のための行をコマンド行 (command line) と呼びます。Bash では，コマンド行でカーソルを移動したり，コマンド行に打ち込んだ文字を削除するなど，コマンド行の内容を編集することができます。コマンド行に何か適当な文字列を打ち込んだ後で以下を試してください。

カーソル移動　カーソルの左右移動には，矢印キーが使えます。カーソルがコマンド行の途中にある状態で `<ENTER>` を押すと，コマンド行の全体が実行されますので，コマンド実行の前にカーソルを行末まで移動する必要はありません。

消去　コマンド行の文字を消去するときには，`<BS>` や `<DEL>` が普通に使えます。また，現在編集中のコマンド行を実行せずに，新しいプロンプトを出すには，

CTRL-c

を打ちます。

1.2.2　ヒストリの一覧と利用

過去に実行したコマンド行の履歴をヒストリ (history) といいます。上向きや下向きの矢印キーを使えば，過去に実行したコマンド行を再度使うことができます。

次のコマンドはヒストリ一覧を表示します。

---

[1] これは，シェル利用の一形態です。他の形態 (シェルスクリプトのためのコマンドインタプリタ) でシェルを利用することもあります。

[2] echo は引数をそのまま出力するコマンドですが，コマンド行における $ はシェルにとって特別な意味があるため，画面に $SHELL とは表示されません。

```
history
```

ヒストリ一覧から，過去に実行したコマンド行を選んで再実行するには

```
!n
```

に続いて <ENTER> を押します[3]。ここで n はヒストリ一覧中のヒストリ番号です。

1.2.3 コマンド行の補完

コマンド名やファイル名の一部をタイプしてから <TAB> を押すことで，残りを自動的に補う (補完) ことができます。この機能を使うとコマンドやファイル名のタイプミスを防ぐこともできますので，是非，活用してください。

1. [コマンド名の補完] コマンド行に his とタイプして <TAB> を押してみましょう。続いて <ENTER> を押しましょう。
2. [補完候補の表示] コマンド行に hi と打ってから <TAB> を押してみましょう。
   hi で始まるコマンドは複数ありますので，まだ補完はされません。
3. 続いて s をタイプしてコマンド行を his としてから <TAB> を押してみましょう。続いて <ENTER> を押しましょう。
4. [ファイル名の補完] まず，カレントディレクトリに存在するファイルの名前を ls コマンドを実行して確認してください。続いて，cat の引数に，既存のファイル名の一部をタイプしてから <TAB> を押してみましょう。ファイル名が補完されたら <ENTER> を押しましょう。
5. [パス名の補完] まず，ls /usr/bin を実行して，ディレクトリ /usr/bin （ルートディレクトリ / の下の usr の下の bin) に存在するファイル一覧を表示してみましょう。続いて，ls /u<TAB>b<TAB> と打って <ENTER> を押してみましょう。

2 コマンド行における特殊文字の扱い

コマンドを入力する行 (コマンド行) の内容を解釈して，コマンドを実行するのはシェルという種類のプログラムでした。ここでは，コマンドに引数を与えて実行する際に必要となる，シェルにとって特別な意味のある文字の取り扱いを学びます。また，検索等でも必要となる，文字列のマッチングに関する考え方の基礎を習得します。

2.1 メタキャラクタ

カレントディレクトリに，仮に a1，a2, . . . ，a5 という 5 つの通常のファイルと distdir というディレクトリのみが存在するとし，a1，. . . ， a5 を distdir に移動したいとします。ファイル移動のコマンドは mv であり，mv の引数には複数のファイルを指定できますから，

[3]知っていて便利なヒストリ機能には次のものもあります。!-n (n だけ前に実行したコマンド行), !! (一つ前に実行したコマンド行 (!-1 と同じ)), !*str* (*str* で始まる最も最近のコマンド行)。ここで n には正の整数を，*str* には適当な文字列 (通常はコマンド名またはその始めの一部) を記述します。これらに続いて <ENTER> を押すと，当該コマンド行の内容が実行されます。

```
mv a1 a2 a3 a4 a5 distdir
```

を実行すれば希望は叶えられますが，移動するファイルをすべて列挙するのは面倒です。

この例のように，一度のコマンド操作で複数のファイルを処理したいとき，**メタキャラクタ** *(meta-character)* という，シェルにとって特殊な意味のある文字を利用すると便利です。メタキャラクタを用いると，上記の操作は

```
mv a? distdir
```

や

```
mv a* distdir
```

で済みます。a? と a* は，シェルによって共に a1 a2 a3 a4 a5 に展開 (置き換え) されてから，引数として mv コマンドに渡されます。

2.1.1 ? と *

? と * は，一言でいえば，

? : 任意の 1 文字
* : 任意の文字列

にマッチする[4]メタキャラクタです。その具体的な使い方を次の例で見てみます。

1. 準備

   (a) カレントディレクトリを /pub/eis/metawork に変更してください。うまくいったかどうか，pwd コマンドで確認してください。

   (b) ls コマンドを使ってカレントディレクトリに存在するファイルを確認してください。次のファイルがあります。

   A A1 B C a a1 a11 a12 a13 a2 a3 b b1 b11

2. ? の使い方

   ls コマンドは，引数にディレクトリ名やファイル名を与えられると，それらの情報を出力します。例えば，

   ```
   ls a b c
   ```

   を実行すると，カレントディレクトリにファイル c はないので，画面に `a b` と表示されます。また，

   ```
   ls -l a b
   ```

   では，ファイル a と b の詳細情報だけが出力されます。

   (a) カレントディレクトリに存在するファイルを，再度 ls コマンドで確認してください。

   (b) 次のコマンドを順番に実行してください。

[4]後に説明しますが，? と * ともに，ファイル名先頭の. (ドット) や / にはマッチしません。

```
ls ?
ls ??
ls ???
ls ????
```

ls の引数に ? を与えても，? とは表示されませんね。

シェルは ? をカレントディレクトリに存在する 1 文字からなるファイル名のリストに展開し，引数として ls コマンドに渡します。その結果として 1 文字からなるファイル名がすべて表示されます。

同様に，?? は 2 文字の，??? は 3 文字のファイル名に展開されます。4 文字のファイル名をもつファイルはカレントディレクトリに存在しないので，最後の例では，? がファイル名に展開されず，その旨のメッセージが出ます。

(c) ? を他の文字と組み合わせて使うこともできます。

```
ls a?
ls ?1?
```

最初の例では，ファイル名の先頭が a であって，その後に何という文字でも構わないから，1 文字が続くものに展開されます。その次の例は，1 の前後に任意の 1 文字が存在するファイル名に展開されます。

(d) ? はパス名の一部として使うことができます。例えば，

```
ls /bin/????
```

だと，/bin というディレクトリに存在するファイルのうち，4 文字のファイル名をもつものが該当します。

なお，? はファイル名の先頭にある . や，パス名の区切り記号 / にはマッチしません。

3. * の使い方

* は ? に似ていますが，1 文字のみではなく，任意の長さの文字列にマッチする点が異なります。

次の例は，各々 a と a1 で始まるファイル名に展開されます。

```
ls a*
ls a1*
```

最初の例で注意すべきは，展開されたファイル名の中に a が含まれていることです。すなわち，任意の長さには，0 文字の長さも含みます。同じ理由から，2 番目の例でも，a1 が結果に含まれます。

次の結果も，容易に予想できるでしょう。

```
ls *3
ls *
ls /bin/*
```

なお，? の場合と同様に，* はファイル名の先頭にある . や，パス名の区切り記号 / にはマッチしませんので，これらは明示的に記述する必要があります。例えば，カレントディレクトリに存在する . で始まるファイル名をすべて扱うには .* とする必要があります。

以上，？や * がどのようなファイル名に展開されるのかを調べるために ls コマンドのみを使ってきましたが，メタキャラクタはシェルが解釈する特殊文字なので，コマンドの引数一般に利用することができます[5]。例えば，`echo a*` では何が出力されるかを考えてから，実行してみましょう。

2.1.2 その他のメタキャラクタ

ファイル名やディレクトリ名に展開される，* や？以外のメタキャラクタとして，次のものもあります。

[*c1c2...*]	[ ] 内の 1 文字（[*c1*-*c2*] を用いて文字範囲を指定することも可能）
{*string1,string2, ...*}	{ } 内の文字列

これらは次のように使います。

```
ls [abc]
ls [A-Z]*
ls /{home,usr,var}
```

`[abc]` は a, b, c のいずれかのファイル名に，`[A-Z]*` は大文字で始まるファイル名に展開されます。`/{home,usr,var}` は `/home /usr /var` に展開されます。

また，~ は自分のホームディレクトリの絶対パス名に展開される特殊文字であり，ほとんどのシェルで利用できます。次のように使えます。

```
ls ~
ls ~/a*
```

2.2 特殊文字の無効化

echo コマンドを使って，画面に 2 * 3 = 6 と表示するにはどうすればいいでしょう。

```
echo 2 * 3 = 6
```

を，結果を予想してから実行してみましょう。

メタキャラクタである * を，普通の文字として echo コマンドで出力するには，シェルによる * の展開を抑止（エスケープ）してから，echo コマンドの引数に与える必要があります。

また，シェルが特殊な意味を持つ記号として解釈する文字には，$ や "，空白等，これまでに紹介した以外にも存在します。コマンド行において，それらを普通の文字として扱う場合にも，次のようにして特殊文字の持つ意味を無効にする必要があります。

1. 単一の特殊文字 *c* に対する特殊な意味の無効化

   $\backslash c$

2. 文字列 *string* 内の特殊文字に対する特殊な意味の無効化

[5] Windows では * や ？ はワイルドカードと呼ばれます。Windows では，スタートボタンからコマンドプロンプトを起動すれば，キーボードからのコマンド入力が可能であり，そこでワイルドカードを利用できます。ただし，コマンドプロンプトでのワイルドカードは，シェルではなく，各コマンドが解釈するものなので，ワイルドカードが利用できるかどうかは実行するコマンドに依存します。

'string'

- *string* 内に ' を含む場合，この方法は使えません。
- 空白を含む文字列を，空白で区切られた複数の引数としてではなく，単一の引数としてコマンドに与えたい場合，この方法は便利です。(引数の区切り文字としての空白文字の意味を無効にします)

2.2.1 例

次の例で * のエスケープを試してください。

```
echo 2 \* 3 = 6
echo '2 * 3 = 6'
```

前者の場合，echo の引数が 4 個であるのに対し，後者では一つです。

なお，シェルには様々な機能があり，man コマンドによるオンライマニュアルも用意されています。

3 問題

1. カレントディレクトリを，ホームディレクトリ内の eis25 に変更した上で，ディレクトリ /pub/eis/metawork に存在する b で始まる名前のファイルを，一度の操作で全てカレントディレクトリに複写してください。

2. ホームディレクトリにファイルがたくさんある人はホームディレクトリ内を整理してください。ディレクトリを作ってから，メタキャラクタを使ってファイルを移動したり，不要なファイルを削除しましょう。

   注意：コマンドの実行結果を取り消すことは基本的にはできませんので，ファイル操作は慎重に行ってください。特にメタキャラクタを使ってファイル操作を行う際には，メタキャラクタの展開結果を事前に確認する習慣を身につけましょう（例えば `ls pp*.c` をまず実行してみるなど)。

3. echo コマンドを使って画面に What's your name? と表示する方法を考え，試してください。

4. `cp a b c` と `cp 'a b' c` は，各々，何をするための操作かを考えてください。

   実際に操作を試したければ，これらを実際に行うために必要なファイル等を echo コマンド等で作成して行ってください。

   ヒント：空白文字は引数を区切る役割をする特殊文字ですが，ファイル名に空白文字を含めることも可能です。ただし，この問を解く以外の場面では推奨しません。