続・誰にでも使える csh 講座
第 1 回
「ASCII コードが知りたい」

安岡孝一

yasuoka ： root さん、 root さん。
root ： 何だい？
yasuoka ： ダブルクウォートの ASCII コードっていくらでしたっけ？
root ： いくらだっけ、そんなの覚えてないや。 `echo '"' | od -b` してごらん。
yasuoka ： `echo '"' | od -b` ですか？

```
~/bin% echo '"' | od -b （ぽこ）
0000000  042 012
0000002
~/bin% 
```

root ： 8 進数で 042ってことは 34 だな。
yasuoka ： 34 ですね、どうもありがとうございます。

```
~/bin% echo 34 | awk '{printf("%c\n",$1);}' （ぽこ）
"
~/bin% 
```

確かに。ところで、この `od -b`って、どんなコマンドなんですか？
root ： 入力の各バイトの ASCII コードを、 8 進数で出すんだよ。

> `od -b` ファイル名
> 入力の各バイトを 8 進数で出力する。
> `od -c` ファイル名
> 入力の各バイトを 8 進数で出力する。ただし文字として表現できるものは、文字で出力する。
> いずれも入力はファイル、出力は標準出力。ただしファイル名が省略された場合は、入力は標準入力。

root ： ちょっとやってみせようか。

```
~/bin% pwd （ぽこ）
~/bin
~/bin% pwd | od -b （ぽこ）
0000000  176 057 142 151 156 012
0000006
~/bin% 
```

~ は 8 進数で 176 つまり 126、 / は 057 つまり 47、 bin はそれぞれ 142、151、 156 で 98、 105、 110、最後に改行コードが 012 で 10、っていうのが出力されてる。
yasuoka ： 最初の 0000000 とか 0000006 とかは？
root ： そこまでで出力したバイト数。これも 8 進数だよ。あれ？ よく考えると、この pwd 変だな。
yasuoka ： あ、気づきました？

```
~/bin% alias pwd （ぽこ）
!! | sed "s/^\/home\/yasuoka/~/"
~/bin% 
```

前に「続・誰にでも使える Unix 講座」で、見せたはずですけど。
root ： そうだったっけ？ うーん、でもこれ、まずいことになるんじゃないかな。
yasuoka ： まずいことって？
root ： リダイレクトができなくなる。

```
~/bin% pwd > /tmp/tekito （ぽこ）
Ambiguous output redirect.
~/bin% 
```

ほらね。
yasuoka ： え？ どうしてですか？
root ： だってこれだと `pwd > /tmp/tekito` が

```
pwd > /tmp/tekito | sed "s/^\/home\/yasuoka/~/"
```

っていう風に展開されちゃうからね。 pwd の出力先が /tmp/tekito なのか sed なのかわからなくなる。
yasuoka ： sed してから /tmp/tekito に出力したいんですけど。
root ： それだったら、もうリダイレクトはあきらめる。
yasuoka ： じゃあ、 pwd の出力はファイルにはもう書けないんですか？
root ： いや、ファイルに書き出す時には tee を使う手があるよ。

yasuoka ： teeって？

tee ファイル名 　　標準入力からの入力を、ファイルと標準出力に出力する。 tee -a ファイル名 　　標準入力からの入力を、標準出力に出力し、ファイルに追加する。

root ： 標準入力を標準出力とファイルの両方に出すんだ。

```
~/bin% pwd | tee /tmp/tekito （ぽこ）
~/bin
~/bin% cat /tmp/tekito （ぽこ）
~/bin
~/bin% rm /tmp/tekito （ぽこ）
rm: remove /tmp/tekito? y （ぽこ）
~/bin% 
```

こんな感じ。

yasuoka ： どうしてそれだと sed の後で tee されるんですか？

root ： pwd | tee /tmp/tekito は

```
pwd | sed "s/^\/home\/yasuoka/~/" | tee /tmp/tekito
```

っていう風に展開されるからね。この辺が alias の！のわかりにくいところなんだけど。

yasuoka ： そうなんですか。

root ： それからもう一つの方法は、 pwd の alias を

```
pwd | sed "s/^\/home\/yasuoka/~/"
```

にして、！を使わないようにする。 pwd にはオプションがないから、この方がいいかもしれない。

yasuoka ： そうですね。少し考えてみます。

（間）

yasuoka ： root さん、 root さん。

root ： 何だい？

yasuoka ： pwd の alias なんですけど、結局

```
~/bin% alias pwd （ぽこ）
pwd | sed "s/^\/home\/yasuoka/~/"
~/bin% 
```

にしました。

root ： まあ、妥当な選択だな。

yasuoka ： それから、 od -b と tr を使ってこんなコマンドを作ったんですけど、見てくれますか？

```
~/bin% cat ascii （ぽこ）
#! /bin/sh
# "ascii" Version 1.0

case "$1" in
?) set "$1" `echo "$1" | od -b`
   echo $3 "$1" ;;
[0-3][0-7][0-7]) echo $1 x | tr x '\'$1 ;;
esac

exit 0
~/bin% 
```

root ： どうやって使うんだい？

yasuoka ： えっと

```
~/bin% ascii '"' （ぽこ）
042 "
~/bin% 
```

みたいに 1 文字指定すると、それの ASCII コードを返します。逆に

```
~/bin% ascii 043 （ぽこ）
043 #
~/bin% 
```

3 桁の 8 進数を指定すると、それに対応する文字を返します。

root ： なかなかやるな。

```
~/bin% ascii 176 （ぽこ）
176 ~
~/bin% !! | od -b （ぽこ）
```

```
ascii 176 | od -b
0000000  061 067 066 040 176 012
0000006
~/bin% ■
```

でも、使ってるのが tr だとすると、たぶん

```
~/bin% ascii 000（ぽこ）
000 x
~/bin% ■
```

やっぱりね。

```
~/bin% ascii x（ぽこ）
170 x
~/bin% ■
```

yasuoka ： え？ どういうことですか？

root ： tr は \ の後に 3 桁の 8 進数をおくことで、文字を ASCII コードで表せるけど、唯一 \000 だけはダメなんだ。だから 7 行目の echo で使われてる x が、そのまま出力されてる。

yasuoka ： えー？ それは困ったなぁ。じゃ、 awk の printf の %c を使おうかな。

root ： いや awk も、 ASCII コードの 0 つまりヌル文字は、出力できない。

```
~/bin% echo 0 | awk '{printf("%c",$1);}' | od -b（ぽこ）
0000000
~/bin% ^0^1^（ぽこ）
echo 1 | awk '{printf("%c",$1);}' | od -b
0000000  001
0000001
~/bin% ■
```

yasuoka ： うーん。どうすればいいんですか？

root ： glob を使うしかないんじゃないかな。

> glob 文字列
> C シェル内蔵。文字列を標準出力に出力する。改行コードは出力しない。文字列は複数書いてもよいが、その場合、文字列と文字列の間はヌル文字が出力される。

yasuoka ： globって？

root ： ちょっとやってみせようか。

```
~/bin% glob a b c（ぽこ）
abc~/bin% ■
```

yasuoka ： echo みたいなもんですか？

root ： まあそうだけど、ちょっと違う。

```
abc~/bin% !! | od -b（ぽこ）
glob a b c | od -b
0000000  141 000 142 000 143
0000005
~/bin% echo !*（ぽこ）
echo a b c | od -b
0000000  141 040 142 040 143 012
0000006
~/bin% ■
```

echo と違って glob は改行コードを出力しないし、文字列と文字列の間は 000 つまりヌル文字だ。

yasuoka ： ふーん。

root ： だからこれを使って

```
~/bin% glob '' ''（ぽこ）
~/bin% ■
```

としてやれば、ヌル文字を 1 文字出力できる。

```
~/bin% !! | od -b（ぽこ）
glob '' '' | od -b
0000000  000
0000001
~/bin% ■
```

yasuoka ： じゃあ、さっきの ascii で 000 を指定された時は、 glob を使えばいいんですね。

root ： まあ、そうなんだけど、でも…。

yasuoka ： じゃ、ちょっとやってみます。

（間）

yasuoka ： root さん、 root さん。

root : やっぱりダメだったかい？

yasuoka : はい。

```
~/bin% cat ascii （ぽこ）
#! /bin/sh
# "ascii" Version 1.1

case "$1" in
?) set "$1" `echo "$1" | od -b`
   echo $3 "$1" ;;
000) glob '000 ' ''
     echo '' ;;
[0-3][0-7][0-7]) echo $1 x | tr x '\'$1 ;;
esac

exit 0
~/bin% ascii 000 （ぽこ）
ascii: glob: not found

~/bin% 
```

この glob: not foundって、どういうことなんですか？

root : シェルを立ち上げてごらん。

yasuoka : え？ あ、はい。

```
~/bin% sh （ぽこ）
$ 
```

root : そこで glob を実行してごらん。

yasuoka : はい。

```
$ glob '' '' （ぽこ）
glob: not found
$ 
```

あれ？ どうして？

root : glob は C シェル内蔵のコマンドなんだよ。だから C シェルでしか使えないんだ。

yasuoka : C シェル内蔵って？

root : /bin とか~/bin とかにファイルとしてあるわけじゃなくって、C シェル自身が直接解釈して実行するコマンド。cd とか chdir とか pushd とか popd とかもそうだし、if とか switch とか while とか set とか、ああ alias や history、source もそうだ。

yasuoka : そういうコマンドだと、どうしてシェル・スクリプトでは使えないんですか？

root : シェルがそういうコマンドを内蔵してないから。シェルが内蔵してるのは cd とか if とか case とか while とか set とかで、glob や alias や popd は内蔵してないからね。

yasuoka : うーん、そういうことですか。でも、glob が C シェル内蔵コマンドだとしたら、シェル・スクリプトからは絶対使えないんですか？

root : いや、方法がないわけじゃない。csh -fc "glob '' ''"してごらん。

csh -fc 文字列 　　文字列をコマンドとみなし、C シェルを介して実行する。

yasuoka : はい、csh -fc "glob '' ''"ですね。

```
$ csh -fc "glob '' ''" （ぽこ）
$ 
```

エラーは出なかったけど、うまくいってるのかな。

```
$ csh -fc "glob '' ''" | od -b （ぽこ）
0000000  000
0000001
$ 
```

おお、完璧。これならいけそう。ちょっと待って下さいね。

```
$ exec true （ぽこ）
~/bin% cp ascii ascii~ （ぽこ）
~/bin% 
```

（間）

yasuoka : こんなもんでどうです？

```
~/bin% cat ascii （ぽこ）
#! /bin/sh
```

```
# "ascii" Version 1.2

case "$1" in
?) set "$1" `echo "$1" | od -b`
   echo $3 "$1" ;;
000) csh -fc "glob '000 ' ''"
     echo '' ;;
[0-3][0-7][0-7]) echo $1 x | tr x '\'$1 ;;
esac

exit 0
~/bin% 
```

root : 変えたのは 7 行目の csh -fc だけかい？

yasuoka : えっと

```
~/bin% diff ascii~ ascii （ぽこ）
2c2
< # "ascii" Version 1.1
---
> # "ascii" Version 1.2
7c7
< 000) glob '000 ' ''
---
> 000) csh -fc "glob '000 ' ''"
~/bin% 
```

2 行目のコメントもですけど。

root : うん。じゃ、やってみせてよ。

yasuoka : はい。

```
~/bin% ascii 000 （ぽこ）
000
~/bin% !! | od -b （ぽこ）
ascii 000 | od -b
0000000  060 060 060 040 000 012
0000006
~/bin% 
```

root : なかなか。でもいっそ、C シェルのスクリプトで書くっていう手もあるんじゃないかい？

```
#! /bin/csh
# "ascii" Version 2.0

switch ("$argv[1]")
case ?:
  set ascii=(`echo "$argv[1]" | od -b`)
  echo $ascii[2] "$argv[1]"
  breaksw
case 000:
  glob '000 ' ''
  echo ''
  breaksw
case [0-3][0-7][0-7]:
  echo $argv[1] x | tr x '\'$argv[1]
  breaksw
endsw

exit (0)
```

yasuoka : root さん、C シェルのスクリプトはきらいだったんじゃないんですか？

root : きらいだよ。けど C シェル内蔵コマンドを使いたい時は、しかたない。

yasuoka : こういった glob みたいな、シェルでは使えなくて C シェルでは使えるコマンドって、たくさんあるんですか？

root : うん。でも今日はもう時間がないから、それはまた次回にしてくれるかい？

yasuoka : はい、わかりました。じゃ、また次回に。