

入出力の話

@leno3s

leno3s.net

July 1, 2019

もくじ

- 1 CUI 使ってますか？
- 2 標準入出力って？
- 3 /dev/pts の話
 - 補足
- 4 発展：シェル芸
- 5 競プロへの応用

今日の目標

- 標準入出力を理解する.
- リダイレクトを用いたファイルの入出力ができる.(重要)
- シェル芸への思想を理解する.(発展)

対象

- 簡単な CLI での操作 (ls, cd, cat, echo, ...) はわかる
- 競技プログラミングの簡単な問題が解ける

CUI使ってますか？

使ってる人👉

使っていない人👉

わからない人 

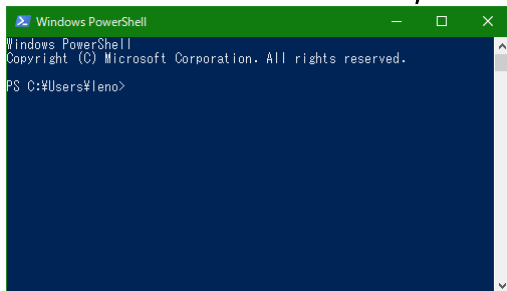
はい

というわけで CUI の話をします.

そもそも CUI とは？

- Character User Interface
- グラフィカルでない, 文字を基本としたインターフェース

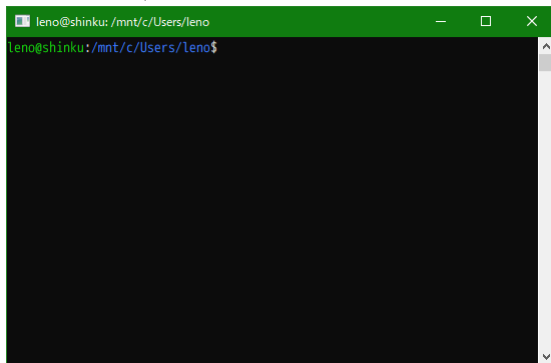
↓ これとか,



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\lino>
```

↓ これとか.

A terminal window with a green title bar. The title bar contains the text 'leno@shinku: /mnt/c/Users/leno' and standard window control buttons (minimize, maximize, close). The terminal area has a black background with a green prompt 'leno@shinku:/mnt/c/Users/leno\$'. A vertical scrollbar is on the right side of the terminal window.

```
leno@shinku: /mnt/c/Users/leno
leno@shinku:/mnt/c/Users/leno$
```

普段からこんな感じで使っていると思います.
ここに画像

競プロではあるある風景ですね. (ほんまか?)

—ところでそろそろ ICPC 予選ですね.

ICPC では, テストケースがテキストファイルで与えられます.
(提出も, 出力結果のテキストファイルです.)

この形式のコンテストに参加した
人がある人👉

fopen() とか使うのめんどくさいよね...
(普段みたいに cin/cout, scanf/printf でやりたいよね)

というわけで

標準入出力について, お話します.

標準入出力についてご存知の人



標準ストリーム (英: *standard streams*) とは、UNIX や Unix 系オペレーティングシステム (OS) において、プログラムの活動実体であるプロセスとその実行環境 (通常は端末) の間の接続として、(プロセスから見ると) あらかじめ確立されている入出力チャネル (パイプ (コンピュータ)) である。OS のカーネルではなくシェルで実装されている機能だが、広く使われているため標準化されている。UNIX や Unix 系 OS では 3 つの入出力があり、標準入力 (英: *standard input*)、標準出力 (英: *standard output*)、標準エラー出力 (英: *standard error*) である。

標準ストリーム - Wikipedia¹ より.

¹<https://ja.wikipedia.org/wiki/標準ストリーム>

(' ω ') ? ? ? ? ? ? ? ?

標準ストリームとは、*UNIX*や *Unix* 系オペレーティングシステム (OS) において、プログラムの活動実体であるプロセスとその実行環境 (通常は端末) の間の接続として、(プロセスから見ると) あらかじめ確立されている入出力チャネル (パイプ (コンピュータ)) である。

OS のカーネルではなくシェルで実装されている機能だが、広く使われているため標準化されている。*UNIX* や *Unix* 系 OS では 3 つの入出力があり、標準入力、標準出力、標準エラー出力である。

重要ポイント

- なんかプロセス間の通信に使ってるらしい
- 標準入力, 出力, エラー出力がある

/dev/pts/ の話

ここから Linux/Unix 基準の話をします

環境の無い人は WSL や MSYS2 でも大丈夫, Cygwin は確認していない
コマンド交えつつ話すので, 手元で実行してね

/dev/って？

/以下のディレクトリの1つ

ここに ls /dev の画像

/dev/って？

- **device** の dev
- 装置の意味
- Linux/Unix で、デバイスのための疑似ファイルが置かれる

疑似ファイルの例

- /dev/stdin
- /dev/stdout
- /dev/stderr
- /dev/null
- /dev/zero
- /dev/(u)random

など 実際はもっといっぱいある²

²\$ ls /dev/ して見れる 見てね

/dev/stdin を追ってみる

\$ ls -al /dev/stdin で stdin の情報が見れる

ここに画像

どうやら /proc/self/fd/0 へのリンクらしい

/dev/stdin を追ってみる

同様に `$ ls -al /proc/self/fd/0` する

ここに画像

`/dev/pts/1` へのリンクらしい (数字部は人によって違います)

同様に `$ ls -l /dev/pts/1` する
ここに画像
これが本体, 数字部は... 後で説明します.

tty コマンド

実はこの番号を調べるコマンドがある

```
$ tty  
/dev/pts/1
```

tty コマンド

もう一つウィンドウを開いて\$ tty してみる

```
$ tty  
/dev/pts/2
```

tty コマンド

!!! 違う番号になった !!!

tty コマンド

ポイント

- この番号によってウィンドウを識別している
- なんかすごいぎじゅつでウィンドウごとに `stdin/out/err` の番号が変わる

補足

Q. tty って何よ

A. Tele **TY**pewriter

補足

印刷電信機, 昔はこれで通信をしていた. → 通信端末
ここにテレタイプ端末の画像

補足

ユーザーがコマンドを打ち込み, 結果を出力する様子がこれと似ている
→ 端末, Terminal

補足

しかし, 今あるウィンドウは実際の端末ではない

→ Pseudo TTY(仮想端末)

→ pts

補足

↓ 端末 ↓

補足

ウィンドウの表示や標準出力を表示, キーボード入力を取得したりする
コマンドの解釈はしない
→ これはシェルの機能

補足

シェル

- bash, dash, tcsh, fish, ...
- 入力されたコマンドを解釈して実行する
- カーネルを直接操作から保護する → 殻っぽい → シェル

補足 2

/proc/self/fd/[0-9] の fd ってなによ

→ File Descriptor

→ システムを介してファイルを扱うための鍵

補足 2

stdin/out/err に 0/1/2 が必ず割り当てられる
他のファイルをプログラムが open 等すると fd が生成される
→ syscall の open() の返り値がこの int 値

補足終わり

ここで疑問

違う番号の/dev/pts/[0-9]*に書き込みをしたらどうなるのか？

実際にやってみる

- vim や nano 等での書き込みはできない
- `$ echo hoge` してもその端末の標準入力へ流れる

リダイレクト

出力先を変えるコマンド (嘘)³ がある

```
$ echo hoge > ./out.txt
```

で./out.txt へと出力することができる

!!!! 出力先に指定したファイルは上書きされるので注意 !!!!

³これはシェルの機能であってコマンドではない

リダイレクト

```
$ echo hoge >> ./out.txt
```

>> で追記になる


(ファイルの末尾へ書き込む)

やってみよう！

```
$ echo hoge > /dev/pts/2
```

(端末の番号は各自変えてね)

やってみよう！

できましたか？どうになりましたか？ 

やってみよう！

\$ command > to/file/path

が標準出力先を to/file/path へ向けている
→ ファイルへの保存ができる！

標準入力

入力のリダイレクトもできる

```
$ grep regex < hoge.txt
```

普段はキーボードから入力を待つところを, ファイルの内容を入力とすることができる.

→ !!! ファイルからの標準入力に使えるのでは !!!

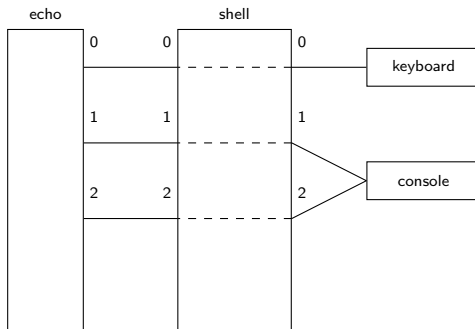
ちょっと補足

リダイレクトには File Descriptor を指定できる
e.g.)

- `echo hoge 0>&1`
- `curl example.com > file.txt 2>&1`

図解 1

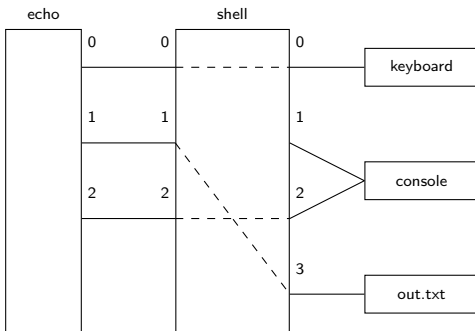
\$ echo hoge



KB が標準入力に, 標準 (エラー) 出力がコンソールに繋がる

図解 2

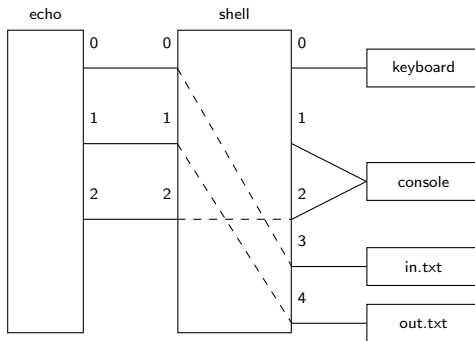
\$ echo hoge > out.txt



echo の標準出力が fd=3(out.txt) へ向かう

図解 3

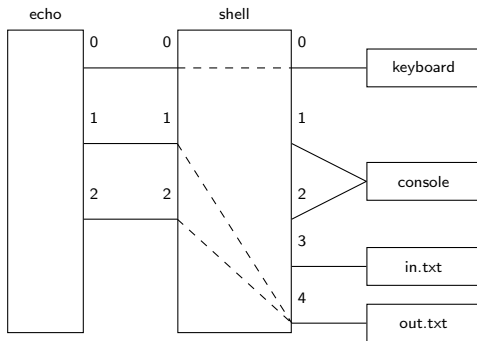
```
$ ./a.out < in.txt > out.txt
```



echo への標準入力に in.txt が渡される

図解 4

```
$ ./a.out in.txt > out.txt 2>&1
```



出力, エラー出力共に out.txt へ書き込まれる

まとめ

\$ command > to/file/path
→ 標準出力リダイレクト

\$ command < to/file/path
→ 標準入力リダイレクト

まとめ

\$ command < input.txt > output.txt

→ 入力を input.txt とし, 出力を output.txt へ保存する.

→ fopen/fwrite 使わなくていい! cin/cout, scanf/printf でいい!

発展：シェル芸

パイプ

あるコマンドの出力を次のコマンドの入力として渡す事ができる

```
$ echo foge | sed s/f/h/
```

hoge

```
# cat file | grep regex
```

いわゆるシェル芸で頻出

シェル芸とは？

“シェル芸 とは、主に UNIX 系オペレーティングシステムにおいて「マウスも使わず、ソースコードも残さず、GUI ツールを立ち上げる間もなく、あらゆる調査・計算・テキスト処理を CLI 端末へのコマンド入力 一撃で終わらせること」である。この技術を持つ人物を シェル芸人 という。”⁴

⁴USP 友の会会長・上田隆一による定義



<https://twitter.com/minyoruminyon>

興味のある人は過去のシェル芸勉強会の問題集⁵を参照するとよい.

⁵<https://b.ueda.tech/?page=00684>

簡単な例

サイコロ

```
$ seq 6 | shuf | head -1
```

連番ディレクトリの作成

```
$ mkdir $(seq -w 14)
```

FizzBuzz

```
$ seq 31 | sed 5~5cBuzz | sed 3~3s/[0-9]*/Fizz/
```

OS のユーザー数を求める

```
$ cat /etc/passwd | awk -F: '{print $1}' | wc -l
```

競プロへの応用

実際に ICPC を想定してリダイレクトの演習をします.
けどテストデータ作るのは面倒くさいので, AtCoder の例題でも解いてもらいます.

ここから@_Bactpus_のターン

おわり.
ご清聴ありがとうございました.
ご意見やマサカリ, 質問は twitter.com/leno3s へ.