

ソフトウェア実習

4. 簡易版 cat コマンドの作成

大木

目的

- Unixコマンドの復習
 - コマンドとは
 - リダイレクトにも慣れる
- 自作 cat コマンドを作成
 - 実行ファイル a.out に名前を付ける

復習：Unixコマンド - 基本

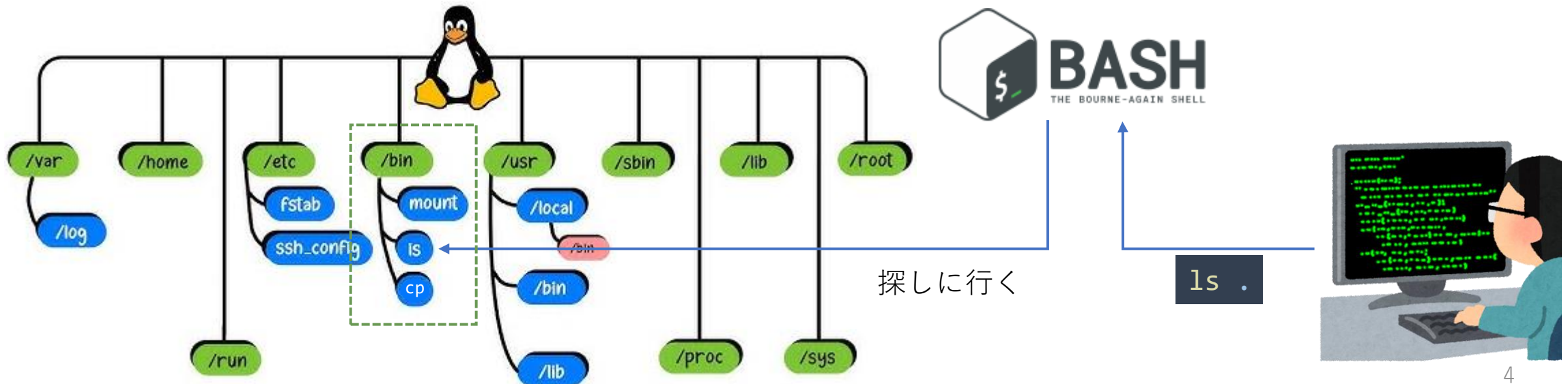
- Unixコマンド
 - 様々なタスクを実行するプログラム
 - 特にファイル操作や検索を覚えると作業効率が大幅に向上する
 - コマンドの実行方法
 - ターミナル(コマンドライン)上でコマンド名を入力

```
# 作業ディレクトリをfolder1に変更
cd folder1

# 現在のディレクトリの内容を一覧表示
ls .
> file1.txt file2.txt folder1 folder2
```

復習：Unixコマンド - 実行プロセス

- コマンドが実行されるまで
 - コマンド本体は実行ファイルとしてOSのどこかに配置されている
 - シェルが本体を探してくれる
 - シェルはターミナルと同時に立ち上がる
 - シェルには bash や zsh など色々な種類がある(名前の由来など調べてみると面白いです)



復習：Unixコマンド - フォーマット

- 基本のフォーマット
 - **[コマンド名]+[オプション]+[引数]**
 - オプションで動作を変更できる
 - オプションは --version や省略形として -v のように指定できる
 - 引数はコマンドの対象となるファイルやディレクトリ
 - 例：
 - `ls` コマンドに `-l` オプションを付けると詳細表示になる

```
ls -l /home/user
> total 0
-rw-r--r--  1 user  staff  1024 May  7 10:00 file1.txt
-rw-r--r--  1 user  staff  2048 May  7 10:30 file2.txt
drwxr-xr-x  5 user  staff   160 May  7 09:00 folder1
drwxr-xr-x  3 user  staff    96 May  7 09:30 folder2
```

復習：Unixコマンド - 実践例

- 自分が良く使うコマンドランキング
 - お仕事でも cd, cat などよく使っていました

	コマンド	回数
1	git	1332
2	kubectl	571
3	docker	503
4	cd	491
5	less	335

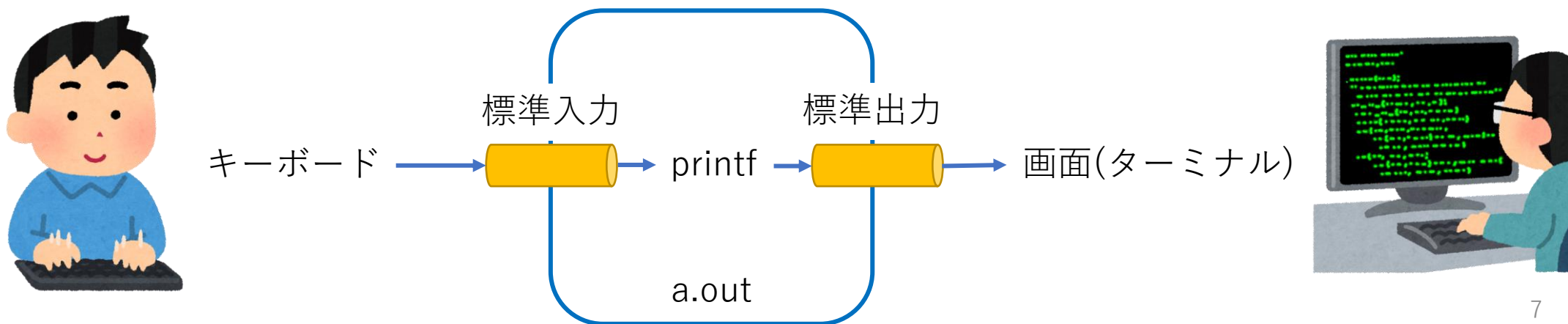
	コマンド	回数
6	cat	297
7	curl	288
8	cp	197
9	history	173
10	pip	151

```
history | awk '{CMD[$2]++} END {for (x in CMD) print CMD[x] "¥t" x}' | sort -nr | head -n 10
```

Unixコマンド - リダイレクト

- リダイレクトとは
 - 通常は標準入力（キーボード）、標準出力（画面）に繋がっている
 - 例：キー入力を受け取って**printf**するプログラム
 - キーボード→毎回手入力するのは面倒 & 画面→出力結果をログとして残したい

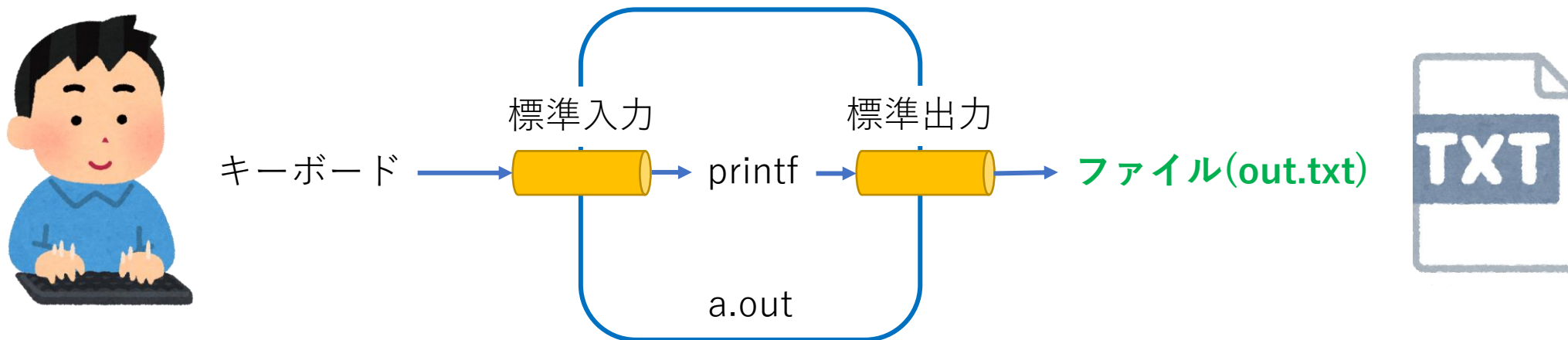
```
./a.out  
This is a pen.  
> This is a pen.
```



Unixコマンド - リダイレクト

- 出力リダイレクト
 - 画面 → **ファイル**に標準出力の行き先を切り替えて実行できる

```
./a.out > out.txt  
This is a pen.  
> 何も表示されない...  
cat out.txt  
> This is a pen.
```



Unixコマンド - リダイレクト

- 簡単に試せる例：

```
# echo [文字列] で [文字列] を標準出力に出力する
echo "Hello world."
> Hello world.
```

```
# echo の結果は画面には表示されず out.txt というファイルが作成される
echo "Hello world." > out.txt
> 何も出力されない...
ls
> out.txt
```

```
# out.txt の中身を見ると, Hello world. と書き込まれている
less out.txt
> Hello world.
out.txt (END)
```

Unixコマンド - リダイレクト

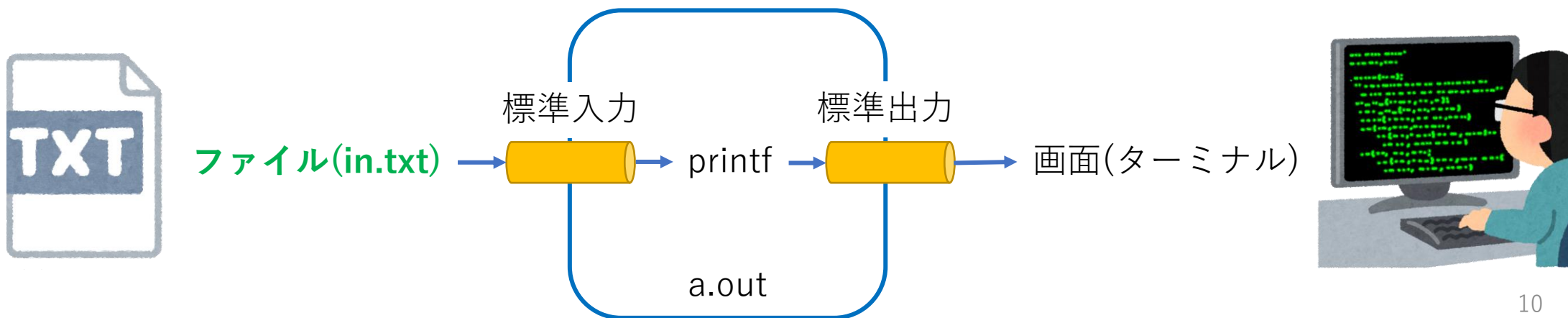
- 入力リダイレクト

- 標準入力の入力元をキーボード → **ファイル**に切り替えて実行できる

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

キーボードからの入力待ちにならない

```
> This is a pen.
```



Unixコマンド - リダイレクト

- 簡単に試せる例：

in.txt には Hello world. と書かれている

less in.txt

> Hello world.

in.txt (END)

何も引数を与えずに cat を実行すると標準入力からの入力を待つ

cat

> 入力待ち(文字をタイプ & ctrl-dでタイプした文字が出力される)

in.txt の内容を手動で入力した時と同じ動作になる

cat < in.txt

> Hello world.

❌ ctrl-d はUnixにおいてEOF(入力の終了)を表す

Unixコマンド - リダイレクト

- リダイレクトの種類

- 基本:

- 入出力

- [コマンド名][オプション][コマンド引数]>[出力ファイル名]

- [コマンド名][オプション][コマンド引数]<[入力ファイル名]

- 複数

- [コマンド名][オプション][コマンド引数]<[入力ファイル名]>[出力ファイル名]

- 切り替えは左側から順番に設定される

- 応用:

- 追記

- [コマンド名][オプション][コマンド引数]>>[出力ファイル名]

- 標準エラー

- [コマンド名][オプション][コマンド引数]&>[出力ファイル名]

- 標準エラー出力もファイルに切り替える(シェルによって書き方が異なるかも)

Unixコマンド - リダイレクト

- 出力リダイレクトの注意
 - 意図しない上書き
 - 書き込み先として空ファイルを作成→コマンドが実行される

```
# in.txt には Hello world. と書かれている
```

```
less in.txt
```

```
> Hello world.
```

```
in.txt (END)
```

```
# in.txt の内容を同一のファイルにリダイレクトして出力する
```

```
cat in.txt > in.txt
```

```
> 何も表示されない
```

```
# in.txt の内容が空になってしまう(シェルの仕様なので仕方ない)
```

```
less in.txt
```

```
> in.txt (END)
```

課題：内容

- 基本的な **cat** の機能を持ったコマンドを自作する
 1. 引数としてファイル名を受け取り内容を表示
 2. 引数が複数なら連結して表示
 - 引数を与えられなかった場合の挙動はどうなるでしょうか？
 3. 行番号表示などのオプションが指定できる
- 実行例：
 - 行番号を付与(-n オプション)しながら複数ファイルの内容を表示

```
cat -n file1.txt file2.txt
>      1 foo
        2 bar
```

課題：内容

- 実装時に気を付けること
 - コマンド引数は [オプション]→[ファイル名] の順番を仮定してOK
 - オプションは全てハイフン(-)で始まる

```
# 必ずこの順番で実行される
./mycat [オプション] [ファイル名...]

# 実行例
./mycat -n -h3 -t12 file1.txt file2.txt
```

課題：内容

- 実行ファイルに a.out 以外の名前を付けましょう
 - コンパイル時に `-o` オプションで指定するだけ
 - どのようなコマンドなのか一目でわかるような名前を考えてみる

```
# コンパイル時に '-o' オプション + [実行プログラム名]
```

```
gcc -o mycat prog.c
```

```
# a.out ではなく mycat という実行ファイルが作成される
```

```
ls
```

```
> prog.c mycat
```


課題：やること

0. 全般

- 課題1より順番に機能を追加していく
- 入力に依存しないプログラムになるように意識
 - ファイルの行数や1行あたりの文字数に制限がない方が好ましい
 - OOM(Out Of Memory) には十分注意する
 - 単純なテストデータだけでは気が付きにくい
 - 一度に全て読み込むのではなく、少量ずつ読み込んで逐次的に処理する
- テストデータは各自で適宜作成する
 - 必要かつ十分なデータを用意する

課題：やること

1. 基本仕様の実装

- 引数のタイプ {オプション, ファイル名} を正しく判断
 - `-n` はオプション, `file1`, `file2` はファイル名として解釈できるように
- 引数が 0 個でも動作すること
 - テキスト 4.7 のサンプルコードを参考に取り組むことを推奨

課題：やること

2, 3, 5(オプション). オプション機能の追加

- 仕様に迷ったら適宜 cat コマンドを叩いて確認してみる

※ 課題3以外

- コマンド引数の解析とファイル出力は分離してしまうと楽
 - 引数の解析をしてから、改めてファイルの出力を行う

課題：やること

4. エラー処理の実装

- 未実装オプションが入力された場合のエラー処理は必須
- どのようなエラー処理があるとユーザに優しいか考えること