情報システム実験実習Ⅱ

Webプログラミング（1回目：bash）

2025年10月3日（金）1～2限

2025年10月17日（金）8限

（授業振替）

実験内容

- ガイダンス
- bash（「Linuxの教科書」の129～148ページが主な内容）
  - bashとは
  - エイリアス
  - bashのオプション
  - シェル変数
  - 環境変数
  - bashの設定ファイル
- レポートの説明

ガイダンス

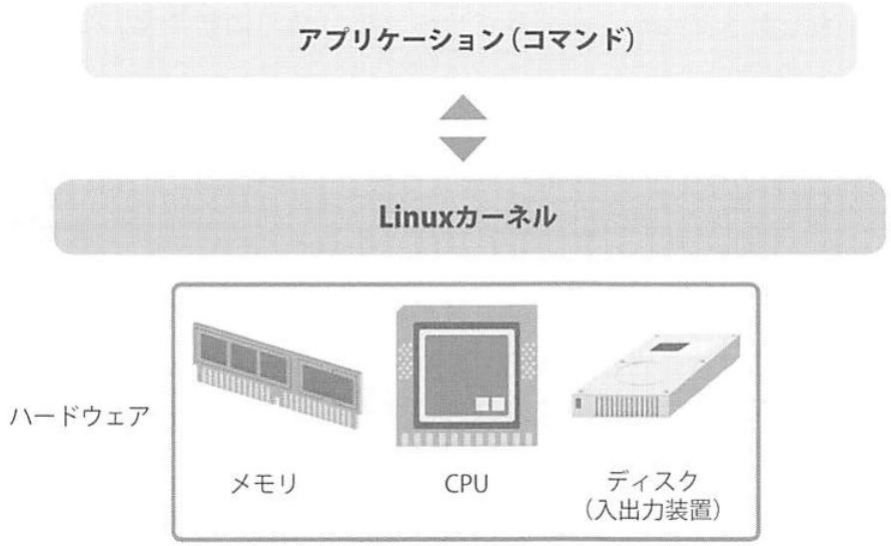
■実験テーマ：Webプログラミング

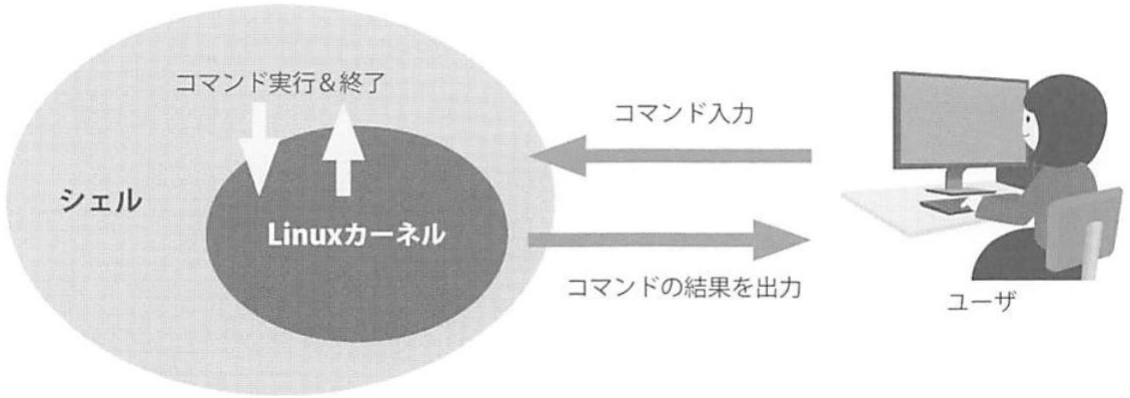
- 全3回でレポートは1本
  - 1回目（10/3（金））　：bash
  - 2回目（10/17（金））　：JavaScript
  - 3回目（10/29（金））　：PHP
- レポートの提出締切は11月21日（金）8時55分を予定

bashとは

■bash（Bourne Again Shell）

- カーネルのインタフェースを担うソフトウェアであるシェルの一種であり，sh（Bourne Shell）の機能を拡張するなどして置き換えたもの（bashはshに対して後方互換性がある）
- 多くのLinuxディストリビューションで標準のログインシェル





bashとは

■bash以外のシェル

- sh（Bourne Shell）
  - ➢ 最も古いシェルだが，現在でもシェルスクリプトの作成などで利用されている（冒頭で「#!/bin/sh」と書く）
- tcsh（T C Shell）
  - ➢ C言語に似た文法のcsh（C Shell）を機能拡張したシェル
- zch（Z Shell）
- fish（Friendly Interactive Shell）

エイリアス

■aliasコマンド

- コマンドに別名（エイリアス）を設定するコマンド

書　式　エイリアスを設定する

```
alias <名前>='<コマンド>'
```

▼-Fオプションでファイル種別を表示

```
$ ls -F /
bin@   dev/  home/  lib64@  mnt/  proc/  run/   srv/  tmp/  var/
boot/  etc/  lib@   media/  opt/  root/  sbin@  sys/  usr/
```

▼エイリアスの設定

```
$ alias ls='ls -F'
```

▼lsとだけ実行しているが、@や/などファイル種別を表示

```
$ ls /
bin@   dev/  home/  lib64@  mnt/  proc/  run/   srv/  tmp/  var/
boot/  etc/  lib@   media/  opt/  root/  sbin@  sys/  usr/
```

エイリアス

■便利なエイリアス

- lsコマンドにおける例

```
alias la='ls -a'
alias ll='ls -l'
```

「すべてのファイルを表示する」laコマンドを作る

「ファイルの詳細情報を表示する」llコマンドを作る

- 様々なコマンドにおける例

```
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

rmコマンドで削除前に確認する

cpコマンドで上書き前に確認する

mvコマンドで上書き前に確認する

エイリアス

■エイリアスの確認（typeコマンド）と削除（unaliasコマンド）

▼lsとcpのエイリアスを確認する

```
$ type ls
ls は `ls -F' のエイリアスです
$ type cp
cp は /bin/cp です
```

▼エイリアスの削除

```
$ unalias ls
```

■エイリアスの一時的な無効化

▼フルパスで指定する

```
$ /bin/ls
```

▼commandを使う

```
$ command ls
```

▼\を先頭に付ける

```
$ \ls
```

bashのオプション

■setコマンド

- bashのオプション（機能）の有効無効を切り替えるコマンド

書　式　bashのオプションを切り替える1

```
set -o/+o <オプション名>
```

「-o」で有効,
「+o」で無効

▼ignoreeofオプションをオンに設定

```
$ set -o ignoreeof          ← Ctrl + Dを無視するよう設定
$                           ← ここでCtrl + Dを入力
シェルから脱出するには "logout" を使用してください。  ← シェルが終了しない
$
```

オプション名	内容
ignoreeof	Ctrl + Dを押してもシェルを終了しない
noclobber	既に存在するファイルをリダイレクトで上書きしない
noglob	パス名展開を無効にする。*などはシェルに解釈されず、そのまま*となる

bashのオプション

■shoptコマンド

- bashのオプション（機能）の有効無効を切り替えるコマンド

書　式　bashのオプションを切り替える2

```
shopt -s/-u <オプション名>
```

「-s」で有効,
「-u」で無効

オプション名	内容
autocd	ディレクトリ名のコマンドを実行すると、それがcdコマンドの引き数に指定されたものとして実行される
dotglob	*や?を使ったパス名展開の結果に、.で始まるファイルも含める
cdspell	cdコマンド実行時、ディレクトリのちょっとしたミスタイプが自動修正される
globstar	パス名展開で**というパターンを使うと、サブディレクトリまで含めたすべてのファイルにマッチする
histappend	bashを終了するとき、履歴ファイルにコマンド履歴を追記し、上書きしない

bashのオプション

■shoptコマンド

- タイプミスを自動修正するcdspellの有効化と動作例

▼タイプミスしたcdコマンドを自動修正

```
$ shopt -s cdspell      ◀── cdコマンドの自動修正をオン
$ ls -F
media/
$ cd mediaa             ◀── mediaと間違えてmediaaと打ってしまった
media                   ◀── 自動修正されて「media」とみなされた
$ pwd
/home/osumi/work/media  ◀── タイプミスしたが修正されて正しく移動できている
$
```

シェル変数

■シェル変数とは

- シェルの内部で使用可能な変数のこと
- ここではbashにおけるシェル変数について取り扱う

■シェル変数の設定

- 値は「'」または「"」で囲む
- スペースを入れてはならない

書　式　シェル変数を設定する

```
<変数名>=<値>
```

▼シェル変数var1に値を設定

```
$ var1='test variable'
```

▼シェル変数var1を参照

```
$ echo $var1
test variable
```

▼変数を設定する際にスペースを入れるとエラー

```
$ var1 = 'test variable'
-bash: var1: コマンドが見つかりません
```

シェル変数

■シェルのプロンプトの変更

- シェル変数のPS1に設定した値がプロンプトとして表示される

▼プロンプトの変更

```
$ PS1='bash> '
bash>
```

▼プロンプトにユーザ名を表示

```
$ PS1='[\u]> '      ←― プロンプトの[]内にユーザ名を表示する
[osumi]>
```

▼プロンプトを見るだけでカレントディレクトリがわかる

```
$ PS1='[\u] \w \$ '      ←― \wをPS1に設定

[osumi] ~ $              ←― 現在はホームディレクトリ(~)にいる
[osumi] ~ $ cd /usr/local/   ←― /usr/localへ移動
[osumi] /usr/local $     ←― プロンプトを見るだけでカレントディレクトリがわかる
```

シェル変数

■シェルのプロンプトの変更

記号	内容
\d	「曜日 月 日」という形式の日付
\h	ホスト名のうち、最初の.までの部分
\H	ホスト名
\n	改行
\t	HH:MM:SS形式の現在時刻
\u	ユーザ名
\w	カレントディレクトリ
\W	カレントディレクトリの末尾のディレクトリ名
\$	rootユーザの場合は#、それ以外のユーザの場合は$
\\	\そのもの

▼改行の入ったプロンプト

```
$ PS1='[\u] \w (\d \t)\n\$ '      ←― ユーザ名、ディレクトリ、時刻などを表示後に改行する

[osumi] /usr/local (火 11月 18 09:48:23)
$      ←―― 改行してプロンプトが表示されている
```

シェル変数

■PATH

- コマンドの実行時に実体ファイルを探すディレクトリを指定するシェル変数

▼シェル変数PATHの内容

```
$ echo $PATH
/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/bin:/sbin
```

- コマンド実行時にそのコマンドが見つからないといったエラーメッセージが出る場合は，このPATHが適切に設定されていない可能性がある

シェル変数

■LANG

- 言語や国を識別するロケールを指定するシェル関数
- ロケールを示す文字列を代入すると表示言語などを変更できる

▼現在のロケールは日本語

```
$ echo $LANG
ja_JP.UTF-8
```

▼アメリカ英語環境でエラーメッセージを表示

```
$ LANG=en_US.UTF-8
$ cmd1
-bash: cmd1: command not found...
```

▼システムで利用可能なロケール一覧

```
$ locale -a
C
POSIX
aa_DJ
aa_DJ.iso88591
aa_DJ.utf8
aa_ER
aa_ER.utf8
```

シェル変数

■コマンドラインの履歴に関するシェル変数

シェル変数名	内容
HISTFILE	コマンドライン履歴を保存するファイル名。デフォルト値は~/.bash_history
HISTFILESIZE	履歴ファイルに保存するコマンドライン履歴の最大行数
HISTSIZE	コマンドライン履歴を保持する最大行数

■シェルの状態に関するシェル変数

シェル変数名	内容
HOME	ホームディレクトリ
SHELL	ログインシェルのパス
PWD	カレントディレクトリ

▼ホームディレクトリ下のディレクトリ指定

```
$ cd $HOME/report      ◀── これは cd ~/report と同じ
$ pwd
/home/osumi/report
```

環境変数

■組み込み（ビルトイン）コマンド

- シェルに組み込まれている（内部にある）コマンドのこと
  - シェルに組み込まれていない（外部にある）コマンドは外部コマンドという
- typeコマンドにより判別できる

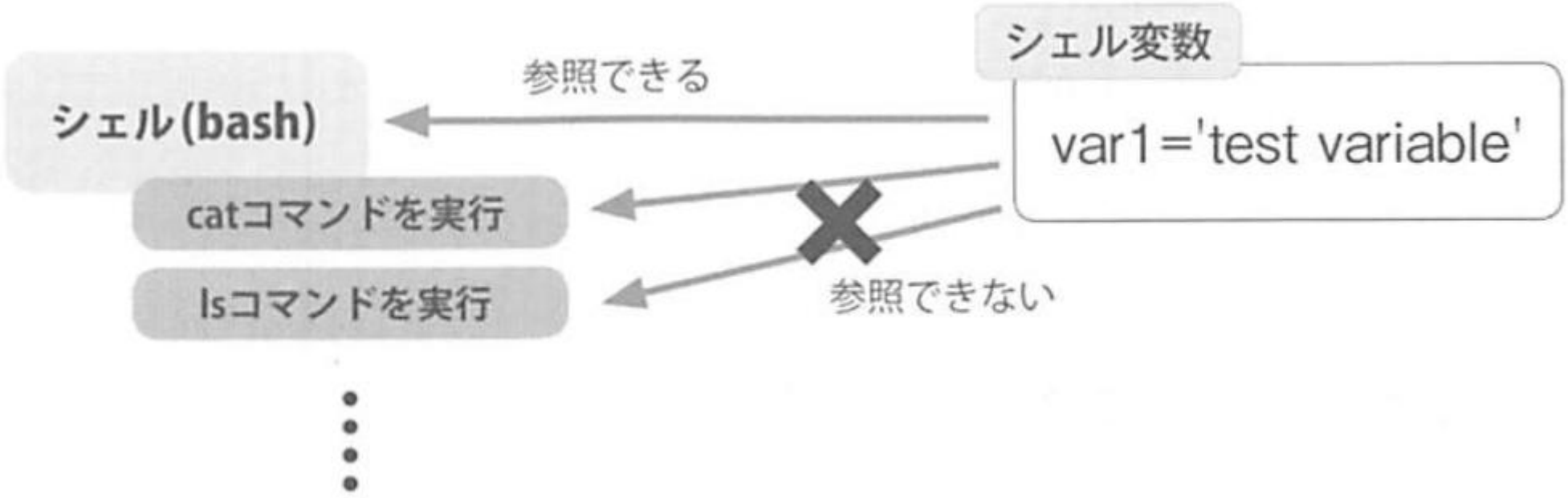
▼typeコマンドによる外部コマンドの判別

```
$ type set
set はシェル組み込み関数です
$ type cp
cp は /bin/cp です
```

環境変数

■シェル変数の利用

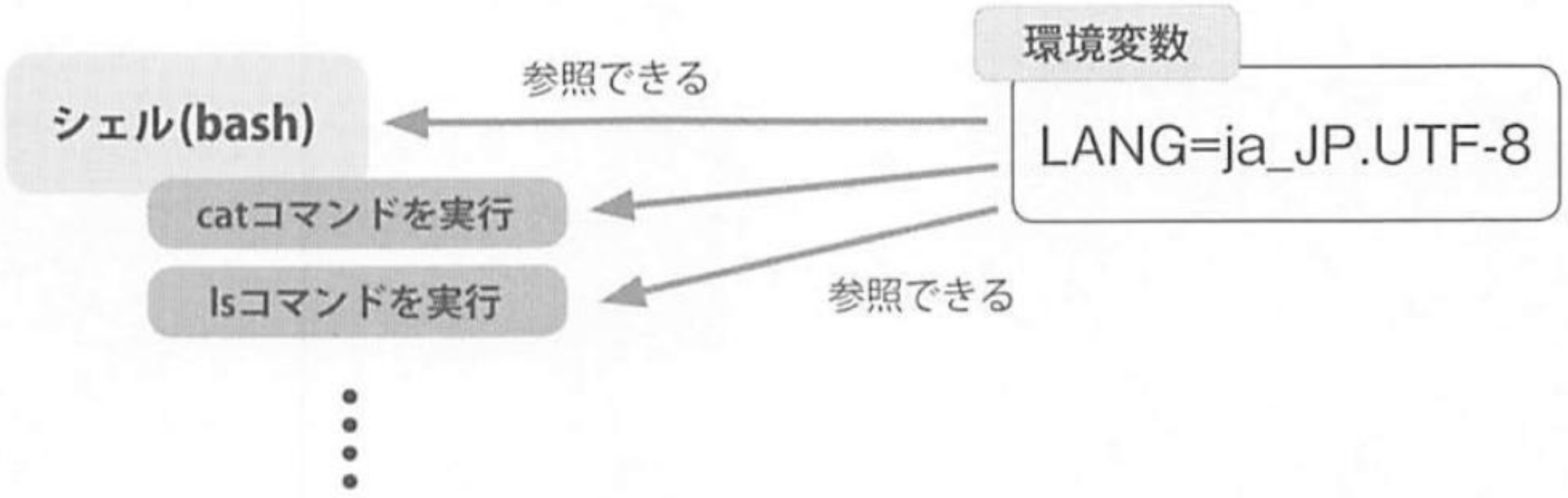
- 外部コマンドからはシェル変数の値を参照できない



環境変数

■環境変数とは

- 外部コマンドからも値を参照できる変数のこと
- LANGなど，環境変数として自動設定されるシェル変数もある



環境変数

■環境変数の一覧表示（printenvコマンド）

▼現在設定されている環境変数を表示

```
$ printenv
XDG_SESSION_ID=51
HOSTNAME=localhost.localdomain
SELINUX_ROLE_REQUESTED=
TERM=xterm
SHELL=/bin/bash
HISTSIZE=1000
SSH_CLIENT=192.168.2.44 59069 22
SELINUX_USE_CURRENT_RANGE=
SSH_TTY=/dev/pts/0
USER=osumi
....(省略)....
PWD=/home/osumi
LANG=ja_JP.UTF-8
```

環境変数

■環境変数の設定（exportコマンド）

書　式　指定したシェル変数を環境変数にする

```
export <シェル変数名>
```

▼シェル変数LESSを環境変数として設定

```
$ LESS='--no-init'
$ export LESS
```

▼値もいっしょに設定する場合

```
$ export LESS='--no-init'
```

LESSはlessコマンドが利用できる環境変数であり，
設定した値がオプションとして自動的に実行される

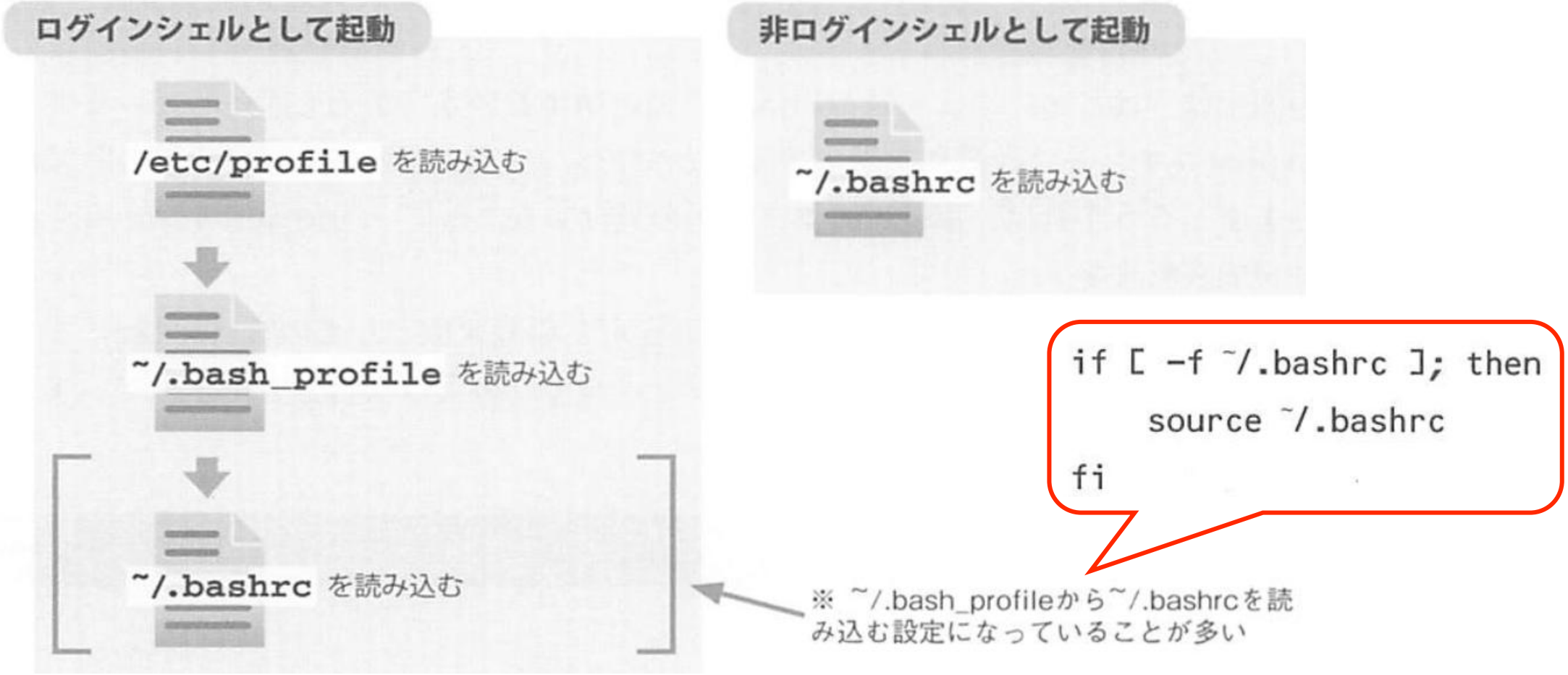
bashの設定ファイル

■bashの設定の恒久化

- ここまでの設定方法のみでは，bashを終了する（例えばターミナルを閉じる）と設定が消えてしまう
- 恒久化（自動設定する）には設定ファイルへの記述が必要
  - 個々のユーザに対してのみ設定する場合，基本的には「~/.bashrc」に記述すれば良い
  - 全てのユーザに対して設定する場合，基本的には「/etc/profile」に記述すれば良い

bashの設定ファイル

■bash起動時に設定ファイルが読み込まれる流れ



bashの設定ファイル

■設定ファイルを編集する際の注意

- 設定ファイルをバックアップした上で編集すると良い

▼.bashrcを別名でコピー

```
$ cp .bashrc .bashrc.org
```

➢ **bashの設定ファイルに限らず，重要であろうファイルやディレクトリはバックアップしてから作業すべき！**

- bashが起動しなくなる可能性があるので，複数のbashを起動しておく（例えば複数のターミナルを開いておく）と良い

bashの設定ファイル

■設定ファイルの反映（一方のみ行えば良い）

- bashを再起動する（例えばターミナルを閉じた上で再度開く）
- sourceコマンドを実行する

▼sourceコマンドの実行

```
$ source ~/.bashrc
```

■設定ファイルの編集例（プロンプトの設定）

```
PS1='[\u@\h] \W \t \$ '
```

▼プロンプトの設定後

```
[osumi@localhost] ~ 19:04:44 $
```

bashの設定ファイル

bashの設定ファイル（~/.bashrc）を，
以下の条件を満たすよう独自にカスタマイズする．

1. コマンドとしてディレクトリ名を実行すると，それがcdコマンドの引数として実行されるように設定する
2. プロンプトの標準の表示からホスト名を削除する
3. 環境変数のLESSで--no-init以外のオプションを指定する
4. Linuxの教科書で例示されていない設定を1つ以上含める
5. 適宜コメントを入れる

レポート評価の際に確認するので，きちんと取り組むこと！

レポートの説明

準拠していない場合は
大幅に減点

■レポートの書き方や内容

- **「レポートの書き方」に準拠すること！**
- 理路整然かつ明確に文章を書くこと
- 内容は表紙，実験結果，課題，感想，参考文献のみ
- 以下を「通し番号（2桁）-学籍番号-氏名-Webプログラミング」のファイル名のzipファイルにまとめ，WebClassから提出する
  - 「通し番号（2桁）-学籍番号-氏名-Webプログラミング」のファイル名のpdfファイル
  - 実験結果2のhtmlファイルと実験結果3のphpファイル（どちらもファイル名を変更しないこと）

レポートの説明

■pdfファイルの内容

- 表紙
- 実験結果1
  - 1ページに収まるようにし，最後で改ページする
- 実験結果2と実験結果3
  - 合わせて1ページに収まるようにし，最後で改ページする

レポートの説明

■pdfファイルの内容

- 課題1
  - 1ページに収まるようにし，最後で改ページする
- 課題2
  - 1ページに収まるようにし，最後で改ページする
- 課題3
  - 1ページに収まるようにし，最後で改ページする
- 感想
- 参考文献

レポートの説明

■実験結果1

- 独自にカスタマイズしたbashの設定ファイルについて
  - pdfファイルにて，設定ファイルの内容（修正や追記した部分のみ）をテキストで示した上で，**自らの独自で具体的な考えや経験を踏まえて**内容（どのような設定なのか）を簡潔に説明する

■実験結果1の主な評価基準

- 条件を満たすよう独自にカスタマイズされているか
- 内容が説明されているか
  - 自らの独自で具体的な考えや経験が踏まえられているか

レポートの説明

■課題1

- zchまたはfishについて，bashと比較しつつ説明する．また，**自らの独自で具体的な考えや経験を踏まえて**，どちらを利用すべきかを説明する．

■課題1の主な評価項目

- zchまたはfishについて正しく説明されているか
  - bashと正しく比較されているか
- どちらを利用すべきか説明されているか
  - 自らの独自で具体的な考えや経験を踏まえられているか