知能情報システム工学実験1A
～シェルスクリプト～

山田 浩史
hiroshiy@cc.tuat.ac.jp

本実験の目的

- シェルスクリプトの理解を通じて、煩雑で面倒な処理を手軽に行う術を身につける
  - 既存コマンドの組み合わせ方を知ることで、生産性を向上させる(楽をする)
  - スクリプト言語，正規表現を組み合わせることでより高度な処理が簡単にできる

※ Linux 上で課題をこなすことを推奨します

- WSL(Windows Subsystem for Linux)やMinGW，Mac 上でもオッケーです

内容

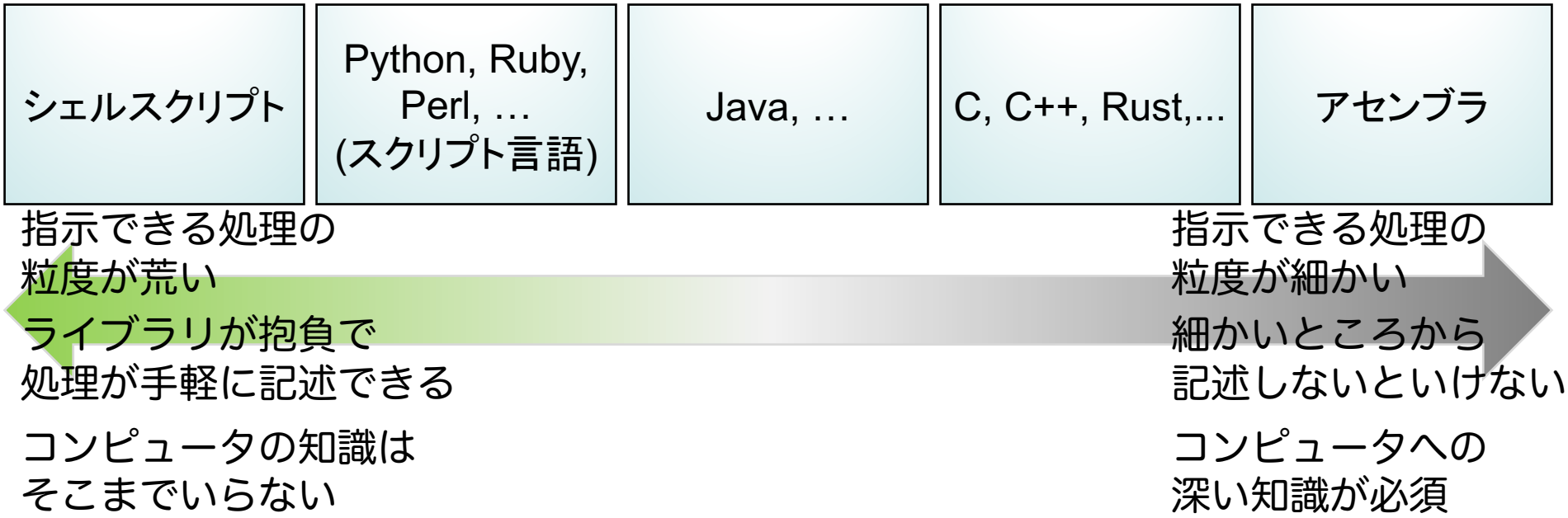
- シェルスクリプトを知る
  - シェルスクリプトはどう書けばよいのか?
- シェルスクリプトを使った例を見る
  - シェルスクリプトはどんなことができるのか？

Why シェルスクリプト?

- 体得するには十分な素養がある
  - C 言語が(ある程度)使えれば、容易に使うことができる
- 本実験で学んだことを最大限に活かすことができる
  - スクリプト言語と正規表現
    - これらのありがたみ・嬉しさ・位置づけを理解する
- 今後の作業効率に大きく影響する
  - 知っているのと知らないのとでは雲泥の差

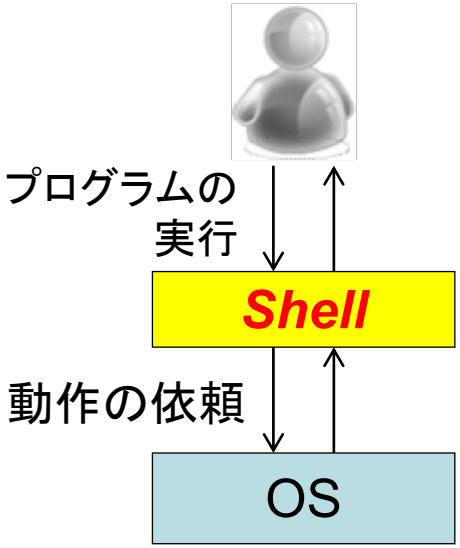
プログラミング言語の選択

- 用途に応じて使い分ける必要がある
  - トレードオフが存在する
    - 粗粒度 v.s. 細粒度，手軽さ v.s. 煩雑さ，敷居低 v.s. 敷居高，etc…
- 今日は左末端のシェルスクリプトに触れる

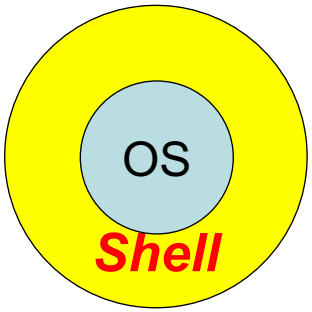


シェルスクリプトとは?

- シェル上でのコマンドをまとめて記述したもの
  - シェルとは: OS とユーザの仲介役
    - コマンドを通して OS に操作を対話的に依頼
  - ターミナル上にはシェルが起動している
    - シェルの種類も様々
    - 今回は bash と tcsh に注目する



- 入力していてはたるい処理をまとめることができる
  - 1,000 個のファイルの中から 1 KB 以上のものだけ CVS 形式に変換する
  - レポートとして提出されたプログラムをコンパイル・実行し、エディタが起動する
  - など



ユーザと OS との間に殻(shell)のように介在

シェルスクリプトを作るには

- 他のプログラミング言語同様、文法に従って記述すればよい
  - 変数や繰り返し文、if 文などがある
    - シェルの種類によって多少文法が異なる
    - e.g.) bash v.s. tcsh
  - 原則、スペースを入れずに記述する

```
#!/bin/bash

COLOR=purple
DATE=`date`
```

```
#!/bin/tcsh

set COLOR=purple
set DATE=`date`
```

- コマンドを扱うならではの機能がある
  - リダイレクト、パイプ、コマンドの制御

リダイレクトとパイプ

- リダイレクト
  - コマンドへの入出力にファイルを介在させる
    - cat –n test.c > test-n.txt
    - sort < word.txt > word-sorted.txt
  - コマンド自身がファイルへの入出力をサポートしなくてよい
- パイプ
  - コマンドの出力を別のコマンドへの入力にす
    - ls –al | grep txt
    - ps axu | grep yamada | grep test
  - コマンド同士を組み合わせて使うことが可能に

シェルがこれらの機能をどう実装しているかは 3 年生で

コマンドの制御

- コマンドの挙動に応じて、次に実行するコマンドを指定できる
  - mkdir foo && mkdir foo/bar
  - mkdir foo || mkdir bar
- コマンドを一つとして扱うことができる
  - (cd ${HOME}/work; ls –al | grep drw) > work_dir.txt
  - $HOME: ホームディレクトリを指す環境変数
    - 環境変数: 実行環境を表す変数

シェルスクリプトを使ってみよう

- シェルスクリプト: 人手ではたるい処理をまとめてやってくれる

- Case Study: プログラムの挙動解析
  - GOAL: grep の資源使用量を計測して、gnuplot でグラフ化する
  - 手でやるといかにもだるそう
    - grep を動作させて瞬間に監視プログラムを起動
    - 監視プログラムのログを gnuplot がわかる形に変換
    - もちろん失敗したら一からやり直し
      - 実験に失敗はつきもの
      - ときには数回とる必要もあり

シェルスクリプトを使えばいい！

- 手順としては・・・
  - 1. grep の資源使用率を計測する
    - CPU 使用率は top を，ディスクアクセス量は vmstat を使う
  - 2. 両者の出力から gnuplot 形式に変換する
  - 3. gnuplot に変換後のデータを与えて完了!

- 1., 2., をスクリプト化しておけばよい
  - スクリプト化すればほぼ全自動
  - 実験はコンピュータに任せればいい

というわけでスクリプト化

- まずは 1. をスクリプト化
  - コマンドの実行を伴うので
    シェルスクリプトを使う
  - 資源監視コマンドの出力はリダイレクトで
    ファイルへ
    - あとで gnuplot 用のデータに変換するために

```
1 #!/bin/bash
2
3 # main
4 top -b -d 1 > cpu_usage.log &
5 vmstat 1 > disk_usage.log &
6 grep -r linux /usr/src/linux/ > /dev/null 2>&1 && pkill top; pkill vmstat; echo "fin"
7
```

ログを gnuplot 形式にしよう

- Python スクリプトで記述することに

cpu2gnup.py

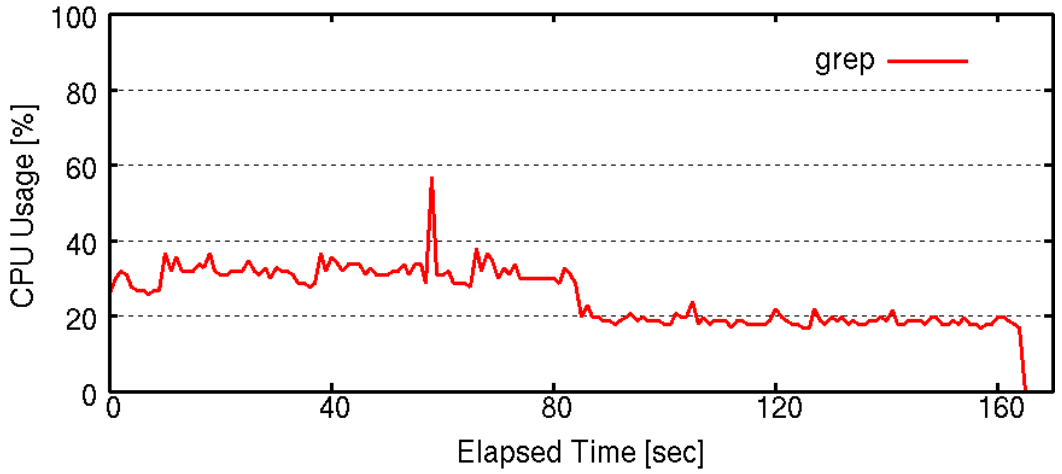
```
#!/usr/bin/python

import sys, re

cpu_pos = 8

pt = re.compile("grep")

def cpu2gnup(filename):
    f = open(filename)
    count = 0

    for line in f.readlines():
        obj = pt.search(line)
        if (obj):
            ln = line.split()
            print "%d\t%s" % (count, ln[cpu_pos])
            count = count + 1
    print "%d\t0" % count

# main
cpu2gnup(sys.argv[1])
```

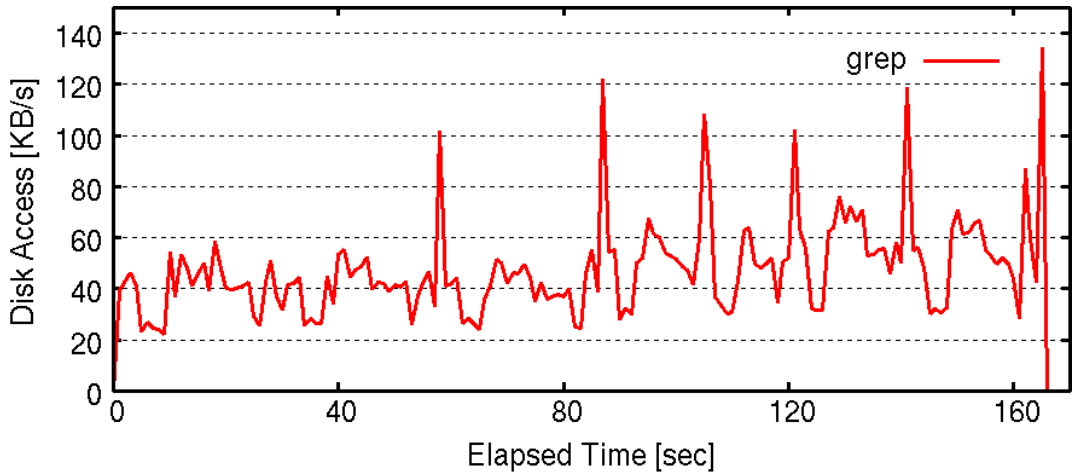
disk2gnup.py

```
#!/usr/bin/python

import sys

data_list_len = 17
diskin_pos = 8

def disk2gnup(filename):
    f = open(filename)
    count = 0

    for line in f.readlines():
        ln = line.split()
        if (len(ln) == data_list_len):
            if (ln[diskin_pos].isdigit()):
                print "%d\t%f" % \
                (count, float(ln[diskin_pos])/1024.0)
                count = count + 1
    print "%d\t0" % count

# main
disk2gnup(sys.argv[1])
```

完了!

- gnuplot に与えれば完了





もちろんスクリプト化は一通りではない

- 個人が利用しやすいコマンド,スクリプト言語を使うのがベスト
  - 先の例は Python を使わなくてもスクリプト化は可能
    - awk コマンドを利用するなど
      - 個人的に awk はあんまりオススメしないですが
- 大事なのは、少しの手間で、多くの作業をコンピュータにさせてしまうこと!

レポートについて(1/3)

- 課題1～4 に取り組んでレポートにまとめる
- 締切: 01/11(水) 0:00 （01/10(火) 24:00）
- 提出先: レポート投函システム
  - PDF 形式で提出してください
  - ファイル名は“学籍番号 8 桁数字_名字(ローマ字)”
    - 例: 10268039 の寺田くん => 10268039_terada

レポートについて(2/3)

- 課題の解答だけを書いてください
  - 問題文は書かなくてよいです
- 原理は不要です
- プログラムの設計の心，実行結果は記述すること
  - 設計の心: どうしてそのようにプログラムを作ったのか
- 無駄な考察は不要です．考察が必要な場合は明記します．

レポートについて(3/3)

- (希望者のみ)レポートフィードバックをします
  - レポートに対して山田がコメントを返します
  - レポートの書き方を洗練したい人,
    書き方を学びたい人にオススメ
    - 設計の書き方はこれでよいのか
    - 実行結果の示し方はこれでよいのか
    - そもそもレポートはこのような書き方でよいのか

- 対象: 課題3

- 希望者は Google Classroom のフォームにお名前をご登録ください
  - レポートで気になる点がありましたらレポート末尾にご記載ください．お答えします．