アルゴリズム・データ構造 I（再）第I回

コンピュータとプログラミングのしくみ

名城大学理工学部情報工学科

山本修身

この講義について

講義名： アルゴリズム・データ構造 I

担当者： 山本修身 (osami@meijo-u.ac.jp)

授業日：月曜日３時限（Bクラス）， 金曜日１時限（Aクラス）
水曜日6時限（再履修クラス）

教室： タワー7階 T-705

関連する科目： **プログラミング演習１，プログラミング演習２**

仮定する知識： プログラミングの基礎，プログラミングの経験

ホームページ： **http://osami.s280.xrea.com/Algo_Data2015/**

成績の評価： レポート（30%）と期末試験（70%）

Algorithm Data Structure

この講義の予定

- 講義と演習を交互に行う.

	回	金曜クラス	月曜クラス	再履修くらす	タイトル
	1	4/3	4/6	4/8	コンピュータとプログラミングのしくみ
*	2	4/10	4/13	4/18	JavaScript入門１ープログラムの入力と実行
	3	4/17	4/20	4/22	関数・再帰呼び出しと配列のしくみ
*	4	4/24	4/27	4/29	JavaScript入門２ー関数と配列を使ってみよう
	5	5/1	5/11	5/13	区分２分法とニュートン法ー欲しい数値の見つけ方
*	6	5/8	5/18	5/20	宝探しをしようー２分法とニュートン法による方程式の解法
	7	5/15	5/25	5/27	木構造と深さ優先探索，幅優先探索
*	8	5/22	6/1	6/3	順列と組み合せの生成
*	9	5/29	6/8	6/10	幅優先探索で8パズルを解く
	10	6/5	6/15	6/17	ハッシングの手法
*	11	6/12	6/22	6/24	色々なデータをハッシュで管理する
	12	6/19	6/29	7/1	単純な整列アルゴリズム
*	13	6/26	7/6	7/8	整列に要する計算時間
	14	7/3	7/13	7/15	計算量について
	15	7/10	7/20	7/22	まとめ

*印は演習形式の授業

アルゴリズム・データ構造で学ぶこと

- コンピュータ上でのデータの処理方法
- データ処理の高速化の手法
- 計算量の考え方
- アルゴリズムと現実のプログラミングとの接点

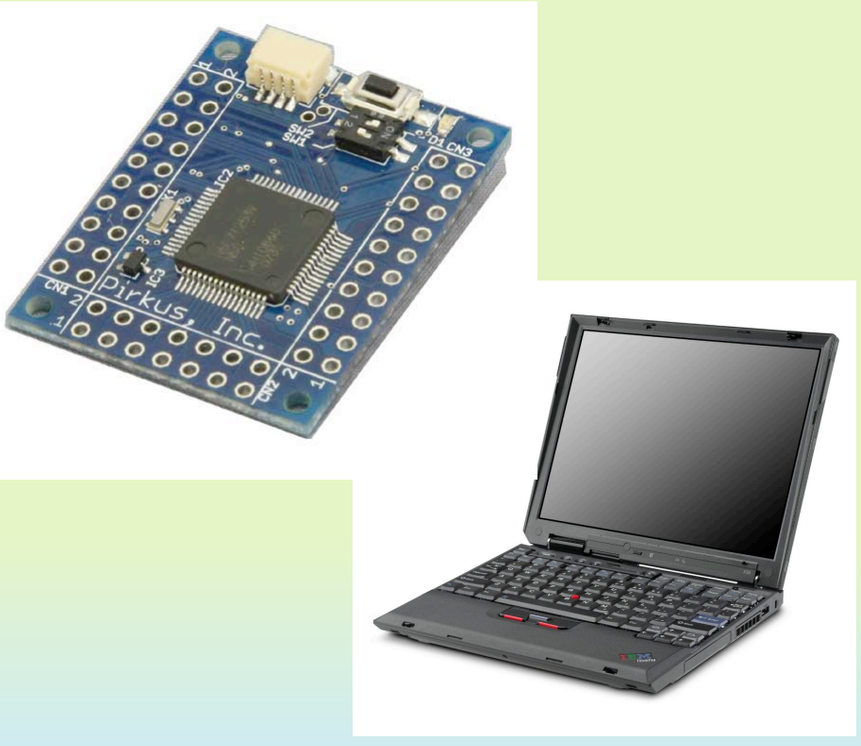
JavaScript

具体的には,

- バイナリーサーチ
- 木探索手法
- ソーティングアルゴリズム
- ハッシング

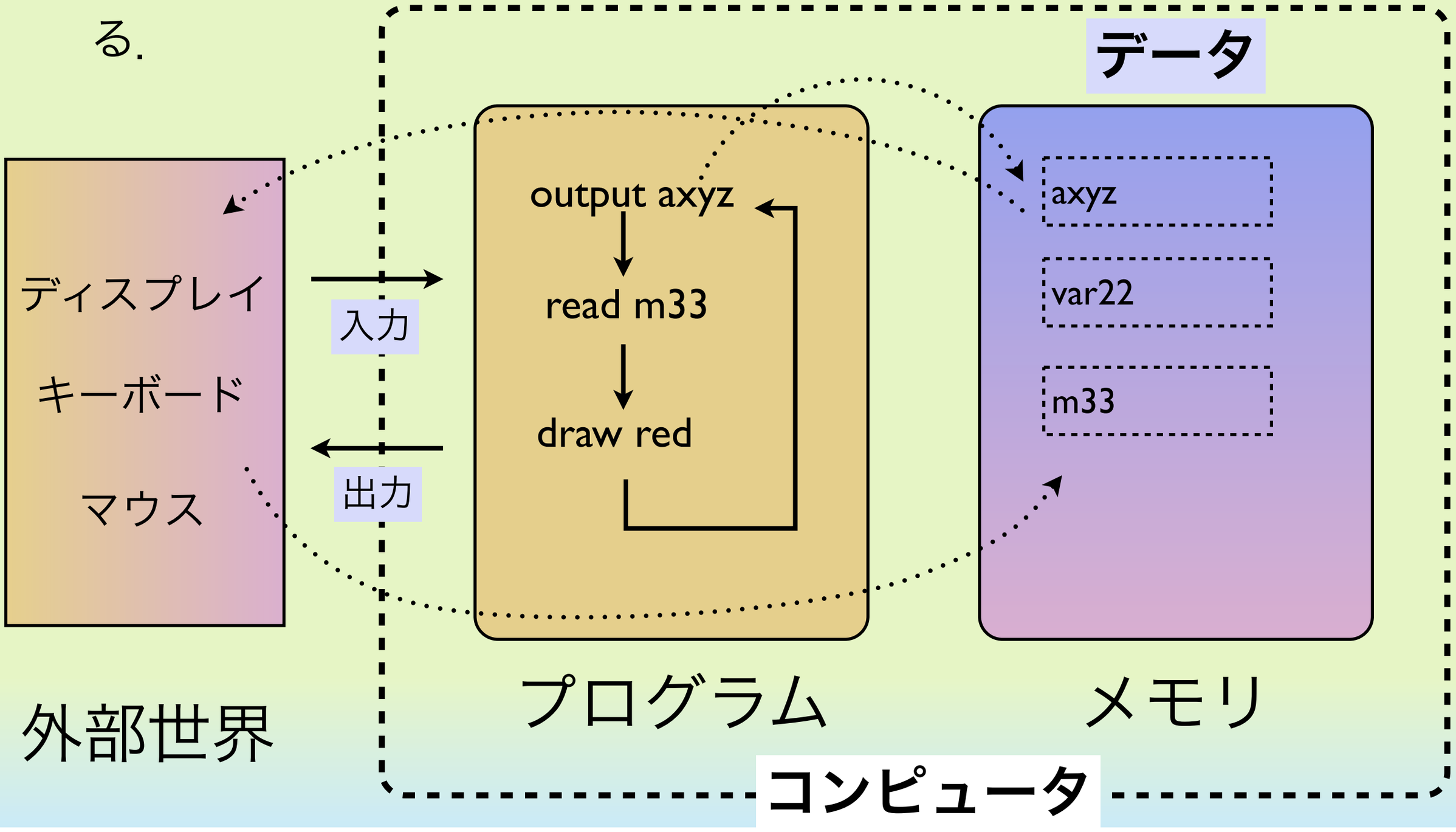
コンピュータとは何か

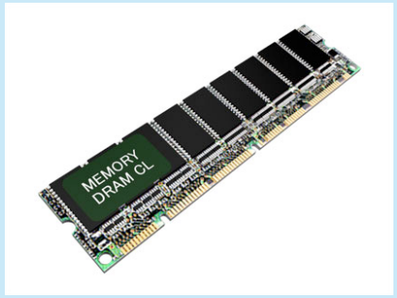
- コンピュータ（計算機）は計算するための機械である．ただし，「計算が何か」は簡単に定義できるものではない．
- 現実のパソコンは非常に複雑な機械である．アルゴリズムを学ぶ場合，複雑なコンピュータを仮定して話をすることは難しい．世の中のコンピュータは千差万別である．ティーポット中にもマイコンが入っている．自動車の中にはECUが入っているし，スマホにもCPUが入っている．我々はコンピュータに囲まれて生きている．それらのコンピュータには様々なタイプのものがある．
- この授業ではJavaScriptが動くコンピュータ（ブラウザ上にある）を決めて，その上で計算できるものが「計算可能」なものであると考える．これは標準的な**「計算機モデル」**となっている．実はCで計算できるものとJavaScriptで計算できるものに差はない．



この講義での計算機モデル

- コンピュータはメモリー（記憶）とプログラムから構成される.





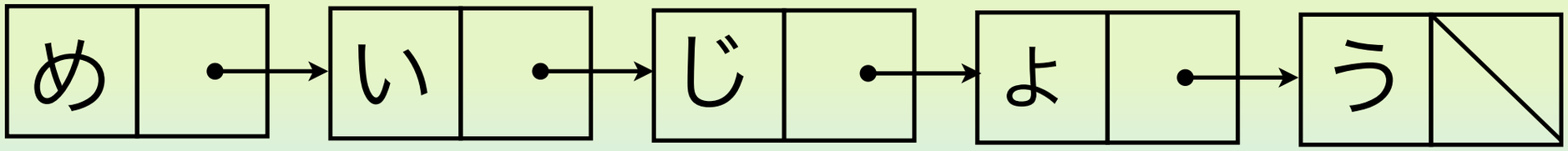
メモリとは

- メモリはデータの置き場所である．データは現実世界の「状態」を表現している．この講義ではこれらのデータの構造をどのように設計するのかについて学ぶ．
- 現実世界の複雑な状態を表現するためには，複雑なデータ構造が必要になる．データ構造をうまく設計できないと，現実世界の問題をうまく解くことができない．

現実世界

めいじょう

データ



プログラムとは

- プログラムとは「命令」の系列（並び）である．命令はCPUによって順次読み込み実行される．プログラムを作成することによって，コンピュータの動きを指定することができる．
- 「命令」はコンピュータの状態（メモリ）を変化させたり，状態を外部へ出力したり，逆に外部の状態を状態に反映させたりすることができる．
- 多くの計算機では，数や文字列（基本的なデータ）の演算を繰返し行うことで大きな計算を実行することができる．

```
int main(){
    int i;
    for (i = 0; i < 10; i++){
      printf(“%d\n”, i);
    }
    return 0;
}
```

◇◇ プログラミング環境について (1) ◇◇

この講義ではJavaScript言語を用いてアルゴリズムを実現する.
ここでは言語環境とプログラミング環境について解説する.

JavaScriptはもともと Netscape Communications によって1990年代前半に作られたwebブラウザ上で動作するプログラミング言語である. 現在では, Mozilla Firefoxや Google Chrome上で動く高速な処理系が提供されており, ブラウザ上でなくても動作する. webサーバとして動作する node.js などが有名である.

この講義では, ブラウザ上の処理系を利用してプログラムを作成し実行する.

◇◇ プログラミング環境について (2) ◇◇

- 主なブラウザに実装されているJavaスクリプトエンジンは以下のとおり.

Mozilla Firefox	Spidermonkey (Brendan Eich, Netscape Communications, 1995; C++)
Google Chrome	Google V8 Engine (Lars Bak, Google, 2008; C++)
Safari (MacOS, iOS)	Webkit JavaScriptCore (Apple, 1998; C++)
Internet Exprorer (JScript)	Chakra (Microsoft, 2011)

◇◇ プログラミング環境について (3) ◇◇

ホーム → JavaScript実行環境 → プログラミング環境（ACEエディタ）

◇◇簡単な計算機を使ってみよう（準備）◇◇

- Webブラウザー FireFoxを立ち上げる.
- 以下のURLを表示させる（ホームページからたどれる）.

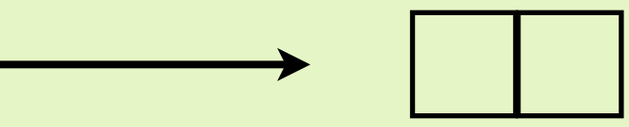
  `http://osami.s280.xrea.com/Algo_Data2015/Oekaki/index.html`

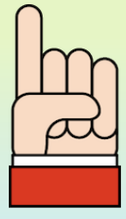
◇◇ 縦横線分描画計算機（1） ◇◇

- この計算機の命令は 「r」,「l」,「u」,「d」 の4文字のいずれかである. これらの命令を適当に並べることによってプログラムが書ける.
- この計算機の状態は描画面であり, 具体的にはこの描画面上に縦横の線分を描く

プログラム例 1

```
rdlur
rdlur
```



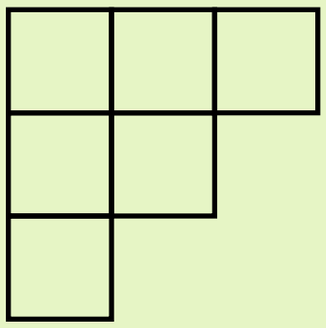


実際にやってみよう！

◇◇ 縦横線分描画計算機（２） ◇◇

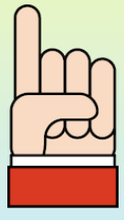
- さらに簡単な例を作ってみる.

プログラム例２



この図形を描くには
どうすればよいか

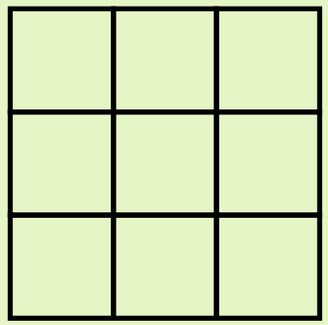
```
rrrdlllu
rdddluuu
rrddlluu
```



実際にやってみよう！

縦横線分描画計算機（３）

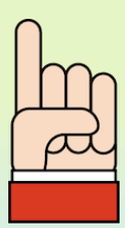
- ３x３の格子を描いてみよう

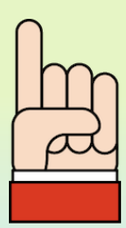


違うプログラムでも
同じ結果をもたらす
ことがある.

```
rrrddd lllluuu
rdddruuu
rdlllldrrr
```

```
rrrdllldrrrdlll
uuurdddruuurdd
```

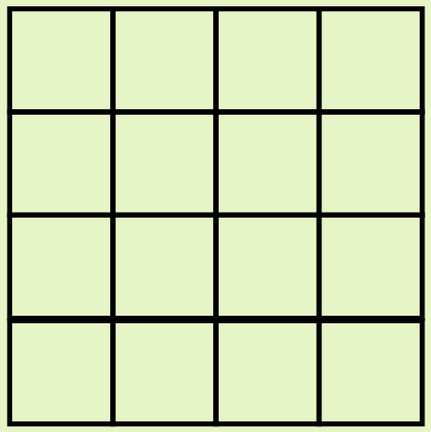




実際にやってみよう！

縦横線分描画計算機（４）

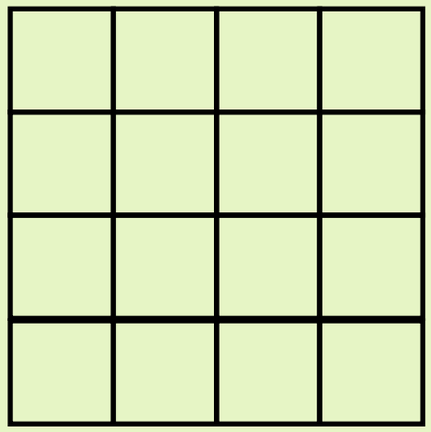
- ４x４の格子を描いてみよう



なるべく少ない命令で描くにはどうすればよいか？工夫してみよう.

◇◇ 縦横線分描画計算機（４） ◇◇

- ４x４の格子を描いてみよう



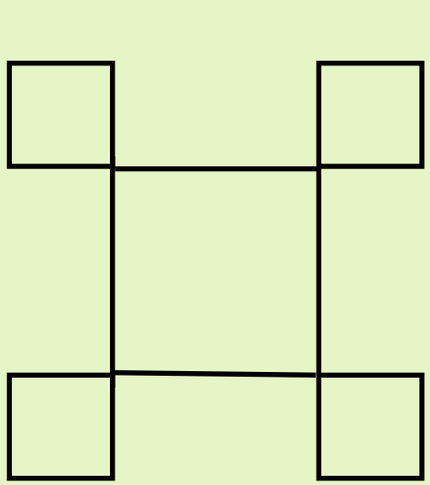
なるべく少ない命令で描くにはどうすればよいか？工夫してみよう.

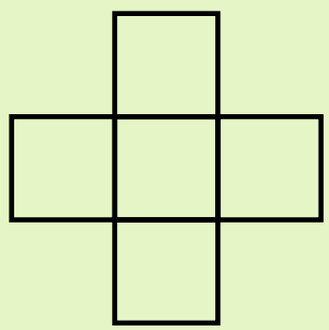
```
ddddruuuuurddddddruuuuurddddd
llllurrrrrullllllurrrrrullllll
```

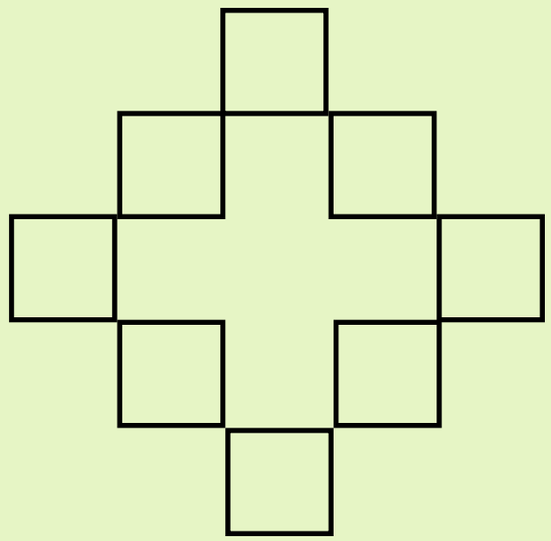
最小手順数になっているかどうかは不明

縦横線分描画計算機（５）

- いくつかの練習問題

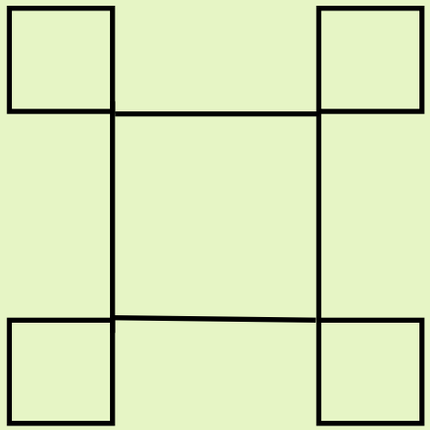




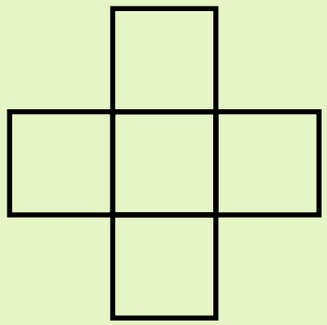


縦横線分描画計算機（５）

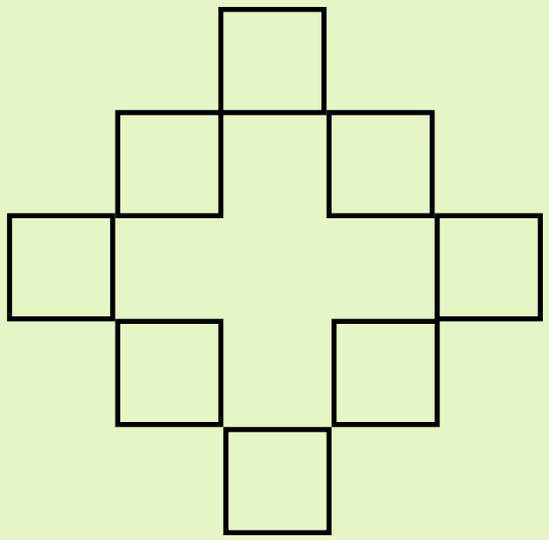
- いくつかの練習問題



```
rdddddl
urrrrrd
luuuuur
dllllllu
```



```
r
rrdl
llur
ddru
uuld
```



```
rrrddddllluuu
r
urddr
rdlld
dluul
lurru
```

命令の追加

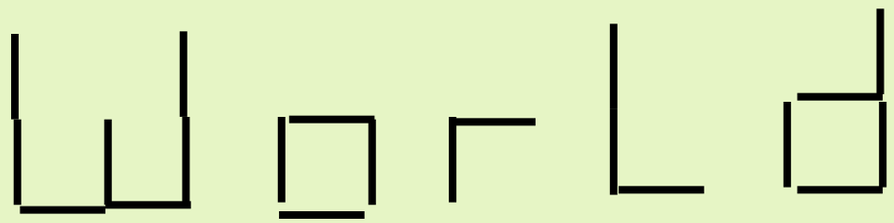
- r, l, u, dのみでは一筆書きの図形しか描くことができない．そこで，もう一つ命令を追加する．’z’によってペンの上げ下ろしができる．ペンが上がっているときはペンを動かしても線は描かれない．‘z’を一回行うとペンが上がり，もう一回おこなうとペンが下がる．最初はペンは下がっている．
- 以下の文字を書いてみよう．

この間は線を描かない

zrdurr...duz

命令の追加（空送り）

- r, l, u, dのみでは一筆書きの図形しか描くことができない．そこで，もう一つ命令を追加する．’z’によってペンの上げ下ろしができる．ペンが上がっているときはペンを動かしても線は描かれない．‘z’を一回行うとペンが上がり，もう一回おこなうとペンが下がる．最初はペンは下がっている．
- 以下の文字を書いてみよう．

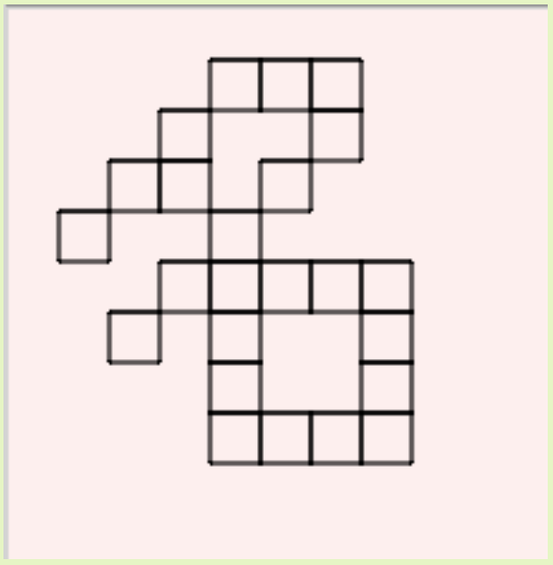


```
ddrudruu
zrdzrdlu
zrrdzur
zruzddr
zrurzldruu
```

命令を自動生成する

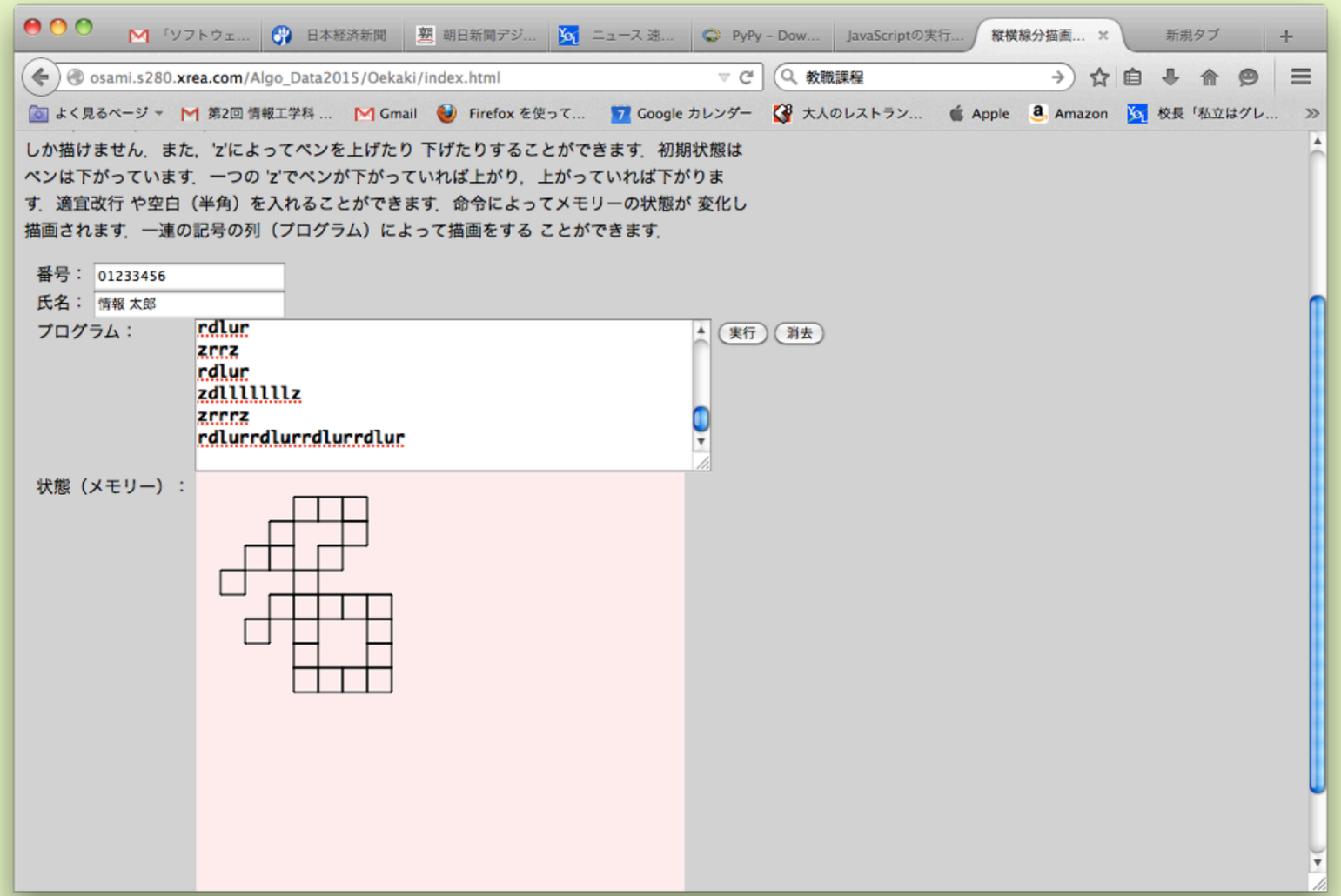
```
function square(n){
    var s = ""
    for (var i = 0; i < n; i++)
       s += "rdlur"
    puts(s)
}

function right(n){
    s = "z"
    for (var i = 0; i < n; i++)
        s += "r"
    puts(s + "z")
}

function down_left(n){
    s = "zd"
    for (var i = 0; i < n; i++) s += "l"
    puts(s + "z")
}

right(3); square(3);down_left(6)
right(2); square(1); right(2); square(1); down_left(6)
right(1); square(2); right(1); square(1); down_left(5)
square(1); right(2); square(1); down_left(4)
right(2); square(5); down_left(7)
right(1); square(1); right(1); square(1); right(2); square(1); down_left(7);
right(3); square(1); right(2); square(1); down_left(7);
right(3); square(4);
```

最適化を考えなければ，より複雑な図形を生成するための命令を自動生成することができる



◇◇ 本日の提出課題（出席の代わり） ◇◇

- この課題の実行結果をプリントアウトして，**学籍番号，氏名を紙の下部に書いて**最後に提出してください．



130043… 情報太郎

◇◇ 縦横線分描画計算機の問題点 ◇◇

- そもそもほとんど役に立たない.
- 計算できる事柄が少なすぎる.
- 一般の問題が解けない. たとえば足し算や掛け算ができない.
- 計算機への入力が存在しない. 出力のみ

それでもプログラミングを体験することはできる.

一人前のコンピュータと同等の計算能力を持っていること

縦横線分描画計算機は**チューリング完全**ではない！普通のコンピュータよりも能力が低い. これから我々が扱う計算機はチューリング完全（普通のコンピュータで計算できることは計算量や計算時間を気にしなければすべて計算できる）である. Cの処理系もJavaScript言語の処理系もチューリング完全である.

◇◇ プログラミング言語の構造 ◇◇

- 一般のプログラミング言語の構造は，本質的には縦横線分描画計算機のプログラミングと変わらないが，複雑な動作を表現するためのしくみや簡潔に書くためのしくみが含まれている.
- 特に，命令を繰り返し実行するためのしくみなどが含まれる.
- また，複雑なデータ構造を利用することができる.

```
int main(){
  int i;
  for (i = 0; i < 10; i++){
    printf("%5d", i);
  }
  putchar('\n');
  return 0;
}
```

アルゴリズムとは何か

- 「アルゴリズム」という言葉は9世紀のイラク バグダットの数学者 アル・フアリズミーから来ていると言われている．アルゴリズムは計算機上である特定の問題を解くための方法であり，プログラム自体もアルゴリズムと言えるが普通はプログラムよりも抽象的な方法論を指すことが多い．

- 計算機が与えられたとき，その上である問題を解決するためのアルゴリズムを考える場合，そのアルゴリズムを用いることによって必ず解けなければいけない．すなわち，計算機が停止しなくなったり（永遠に計算し続けること），エラーを吐いて止まってしまうことがあるとアルゴリズムとは言えない．

al-Khwārizmī

◇◇ 単純なアルゴリズムの例（1） ◇◇

ユークリッドの互除法

アルゴリズムの中身はわからなくても良い

二つの正整数a, bの最大公約数 (GCD) を計算するには以下のようにすれば良い.

```
while (b > 0)
    (a, b) = (b, a % b)
```

最後のaの値が元のa, bのGCDとなる.

この２点が言えないとアルゴリズムと呼べない.

正当性：止まれば必ず正しい答えを出す

停止性：必ず停止する（計算が終わること）

これを言うためには，多くの場合，数学的な方法が必要となる.

◇◇ 単純なアルゴリズムの例（2） ◇◇

Cでの実現

```
#include <stdio.h>
#include <stdlib.h>

int euclid(int a, int b){
  while (b > 0){
    int am = b;
    int bm = a % b;
    a = am;
    b = bm;
  } /* while */
  return a;
} /* euclid */

int main(){
  int a = 34;
  int b = 56;
  printf("GCD(%d, %d) == %d\n", a, b, euclid(a, b));
  return 0;
} /* main */
```

Cだと行数が多くなり本質的な部分が少々見えにくい

◇◇ 単純なアルゴリズムの例（3） ◇◇

JavaScriptでの実現（FireFox, version 1.7以上）

```
function euclid(a, b){
  while (b > 0)
    [a, b] = [b, a % b];
  return a;
} /* euclid */

var [a, b] = [35, 56]
puts(a + ', ' + b + ', ' + euclid(a, b))
```

JavaScriptでの実現（Chrome, ECMA-262レベル）

```
function euclid(a, b){
  while (b > 0){
    var c = b
    b = a % b
    a = c
  }
  return a;
} /* euclid */

var a = 35
var b = 56
puts(a + ', ' + b + ', ' + euclid(a, b))
```

◇◇ 単純なアルゴリズムの例（４） ◇◇

- **OCamlでの実現**

```
let rec euclid a b =
  if b > 0 then euclid b (a mod b)
  else a in
let a = 35 and b = 56 in
  Printf.printf "%d %d %d\n" a b (euclid a b)
```

- **Pythonでの実現**

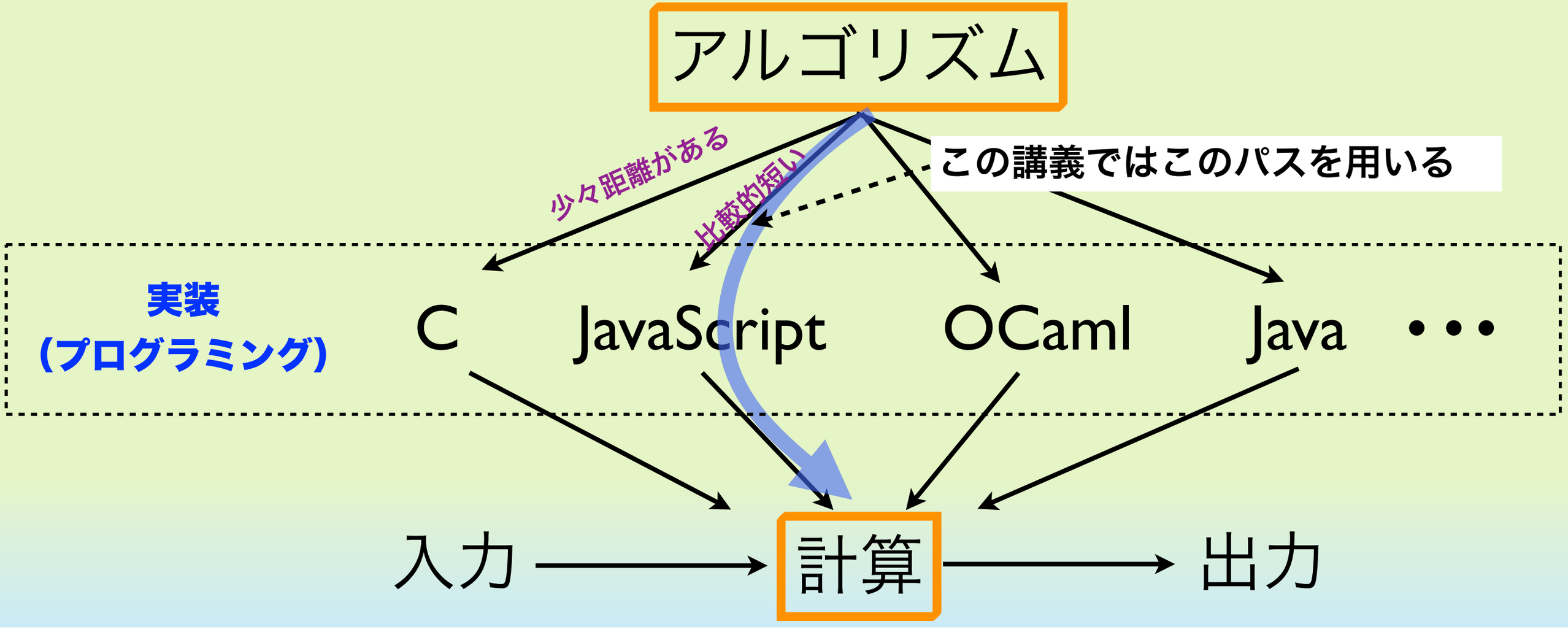
```
def euclid(a, b):
    while b > 0:
        a, b = b, a % b
    return a

a, b = 35, 56
print a, b, euclid(a, b)
```

- **Rubyでの実現**

```
def euclid a, b
  while b > 0 do
    a, b = b, a % b
  end
  return a
end

a, b = 35, 56
puts format("%d %d %d", a, b, euclid(a, b))
```

アルゴリズムと実装の関係

- 色々な実装に対応するために，アルゴリズムはある程度抽象的なものとなる.
- 現実の実装はその環境にしたがって，変える必要があるが，アルゴリズム自体の考え方が変わるわけではない.



◇◇ プログラミング言語 JavaScript について ◇◇

- JavaScriptの正式名称はEcmaScriptであるがここでは，JavaScriptと呼ぶ．JavaScriptはもともとMozillaによって作られた言語であるが，Ecma Internationalによって標準化されており，その仕様はECMA-262に定義されている．Javaという名前が使われているが，Java言語とは関係ない．FireFox, ChromeなどのWebブラウザでは直接JavaScriptを実行できるエンジンを備えている．特に「v8エンジン」(Google Chrome) と呼ばれる処理系や「spider monkey」(Mozilla Firefox) 処理系は，処理速度が極めて高速であることが知られている．

- この授業ではプログラミングにJavaScriptを用いる．JavaScriptは細かいところを気にしなくてもブラウザ上で普通に実行可能であり，C言語などに比べて柔軟性が高く，プログラムも書きやすい．本格的なソフトウェア開発の道具として用いるのではなく，アルゴリズムを記述するための道具として用いる．

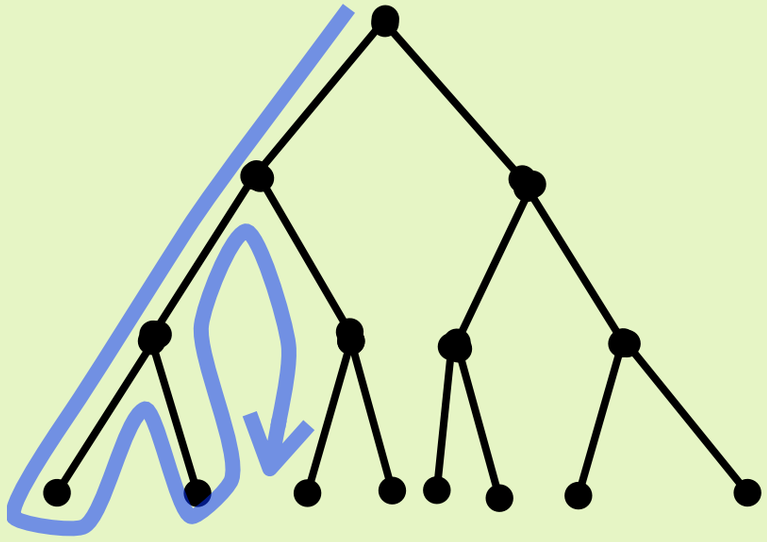
JavaScriptの特徴

- 文法がシンプルである.
- Webブラウザがあれば自宅でも大学でもどこでも動く. コンパイラ不要. インストール不要. もちろんタダ.
- とても高速である. Javaと同じくらいの速度（Cコンパイラの半分くらいの速度）で計算することができる.

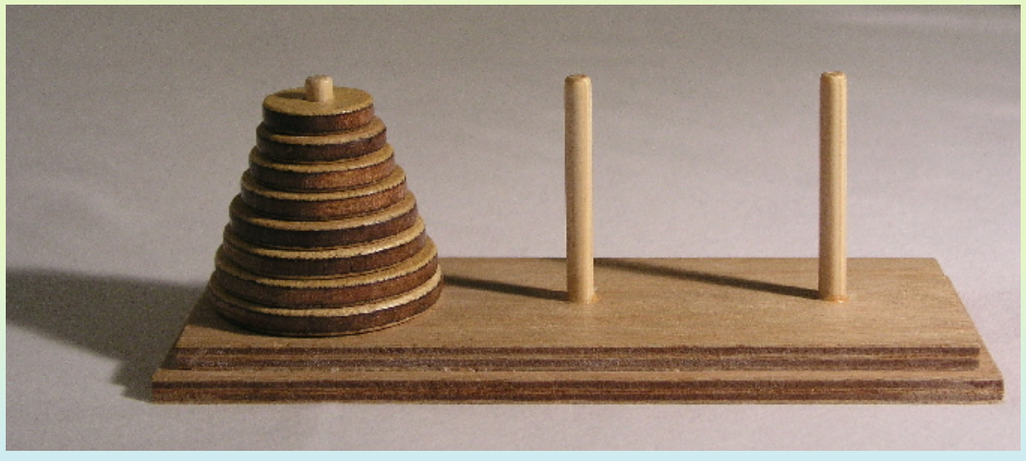
この授業の第2回～第4回にJavaScriptを用いてプログラミングの講義と演習を行う.

この講義で当面学ぶこと

- 欲しいものを見つけ出すこと．特に多くの可能性の中から本当の答えを見つけ出す方法について学ぶ.
- これを「探索」と呼ぶ.

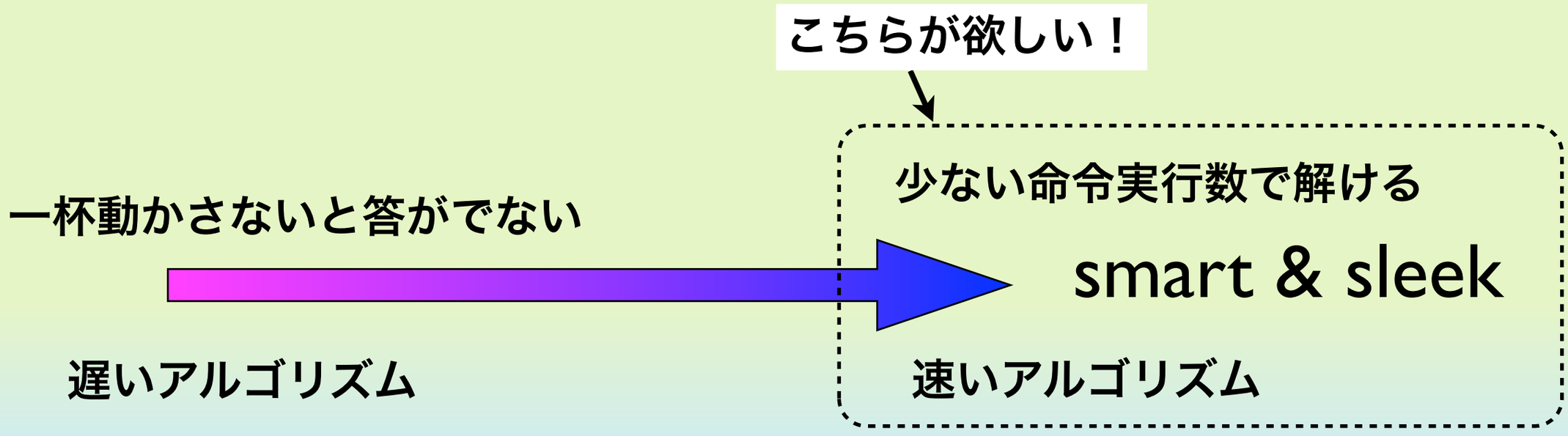


- 多くの場合，探索は木構造と呼ばれるデータ構造を辿る作業となる.
- 問題によっていくつかの辿り方を考える.



計算量について

- 色々なコンピュータ上でプログラムとして実現できるアルゴリズムはなるべく「速い」ものであることが望ましい.
- 実際の計算速度はプログラムの書き方やコンピュータそのものの速度などに依存するので一概に速度を決めることはできない.
- そのため, 「良いアルゴリズム」と「悪いアルゴリズム」を見極めることは, かなり難しい場合がある.
- この講義では最後の部分で計算量の標準的な考え方について学ぶ.



本日のまとめ

- コンピュータ（計算機）の基本的な考え方とどの動作原理について述べた．（もちろん，細かな動き方やプログラムの書き方については述べていない．）

- アルゴリズムは究極的にはプログラムと同義である．しかし，現実のプログラミングは言語仕様などによって色々に変化するので，アルゴリズムはそれよりは抽象的なレベルで議論する．

- より計算量の少ないアルゴリズムを設計することが目標であるが，抽象的なレベルではなかなか何が最適なのかを述べることは難しい．