第2回のキーワード

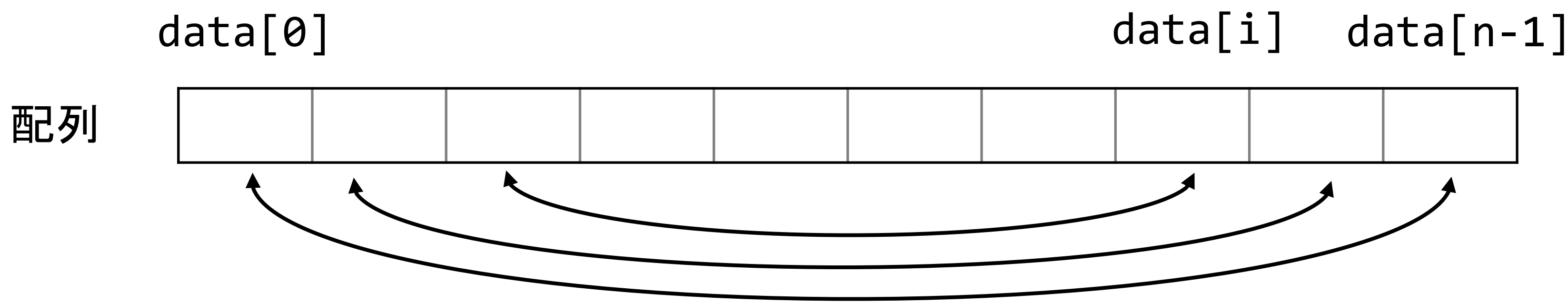
アルゴリズム関係

- アルゴリズム
- 確率，期待値
- 再帰
- 線形探索(linear sarch)
- 計算量
- 時間計算量
- 空間計算量
- 最大(最悪)計算量
- 平均計算量
- O記法(オーダー記法)
- $O(n)$

Java関係(プロIIの復習)

- 配列
- クラスの配列
- ポリモーフィズム(多態性)
- equals

配列の反転のアルゴリズム



データ数が n 個の場合，交換回数は $\lfloor n/2 \rfloor$ 回

```
n = data.length;
for (int i = 0; i < n / 2; i++) {
    double t = data[i];
    data[i] = data[n - 1 - i];
    data[n - 1 - i] = t;
}
```

ソースコードを分析すると，
データ数に依存するのは，
ここのループ回数だけ

筆算のアルゴリズム

- 筆算と時間の計算は共通
  - 下の位（桁）から順に計算し，繰り上がり（繰り下がり）があったら，1つ上の位に加算（減算）していく
  - これを応用すれば，非常に桁数が多い整数の演算も可能

	1	7	4
+		5	9
	2	3	3

下の位から上の位へ計算

	18時間	34分	48秒
+)	9時間	53分	35秒
1日	4時間	28分	23秒

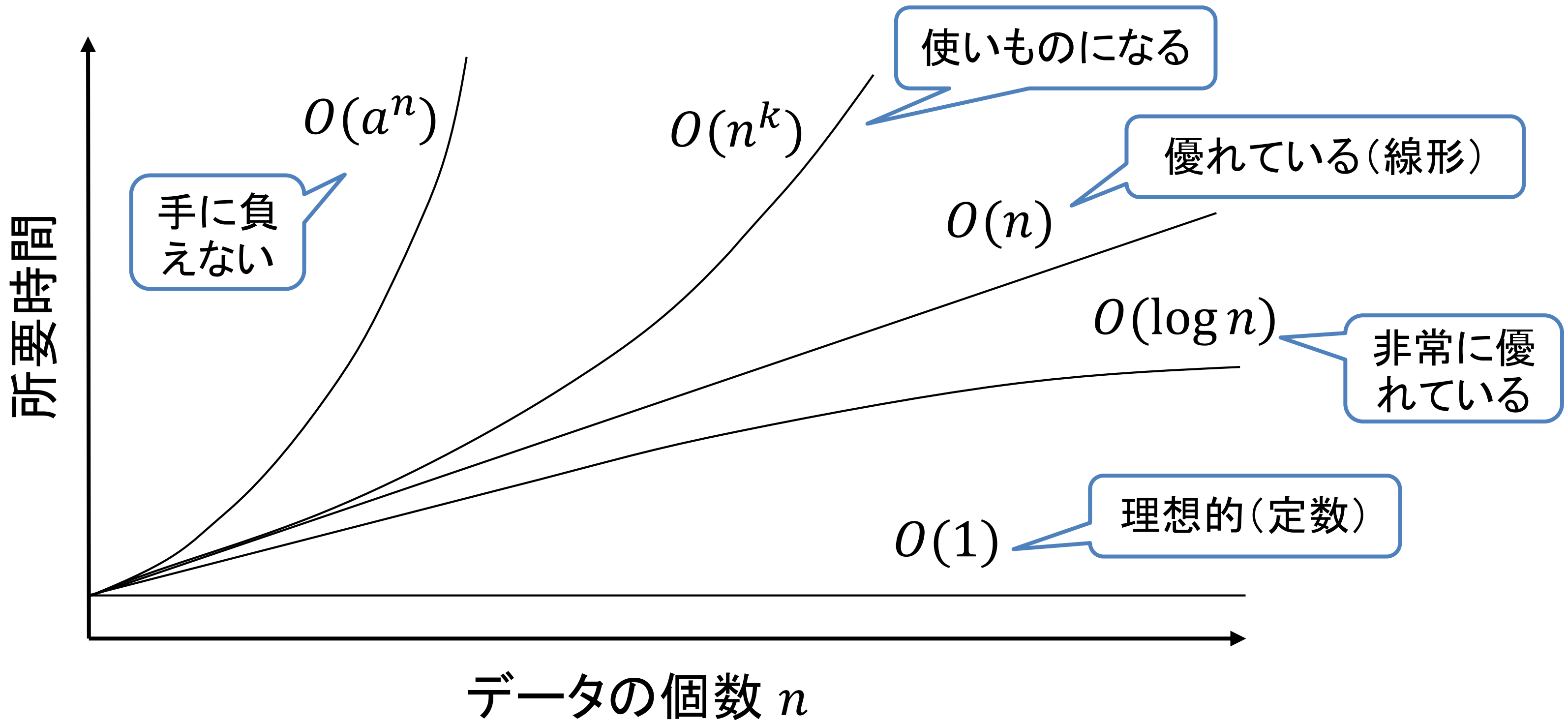
下の位から上の位へ計算

アルゴリズムの性能評価

- よいアルゴリズムとは？
  - 速い（所要時間） ⇒「時間計算量」が少ない
  - 小さい（使用メモリ） ⇒「空間計算量」が少ない
  - これらは，両立が難しいことが多い
- 計算量の指標
  - アルゴリズムの種類によって，データの並び方や構造に対する得意・不得意がある
  - 最大計算量：そのアルゴリズムが最も不得意なデータセットに対する計算量
    - 例：小さい順に並んでいるデータセットから最大値を線形探索
  - 平均計算量：あらゆるデータセットに対する平均の計算量

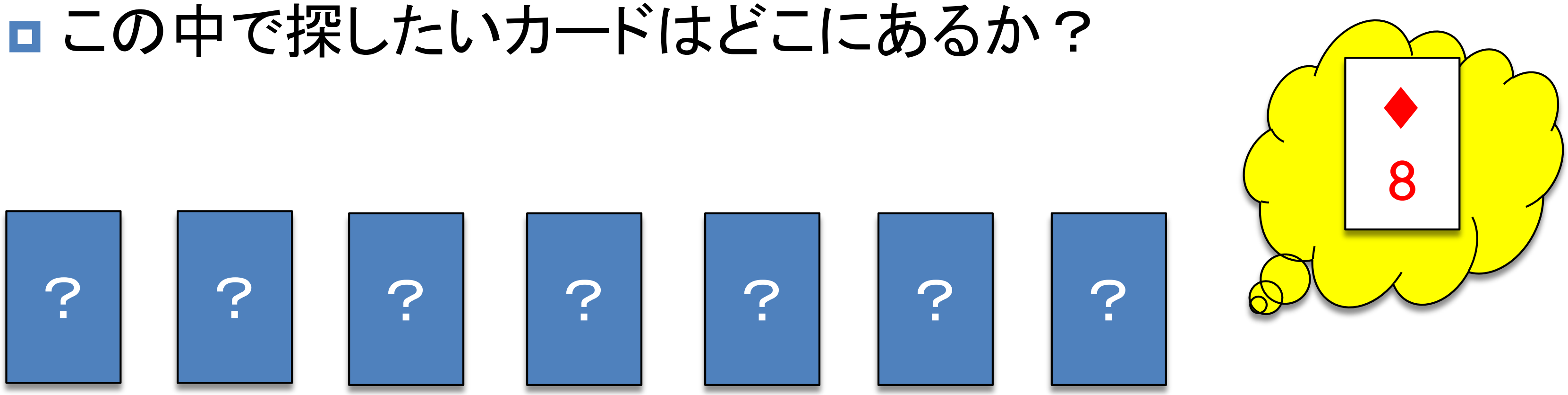
オーダー記法

- 計算量の数学的表現
  - ほとんどアルゴリズムは，データの個数(n)が増えれば増えるほど，計算に時間がかかるようになり，メモリも必要になる
  - nに対する計算量の増加を“関数の種類”で分類し，O記法で表す



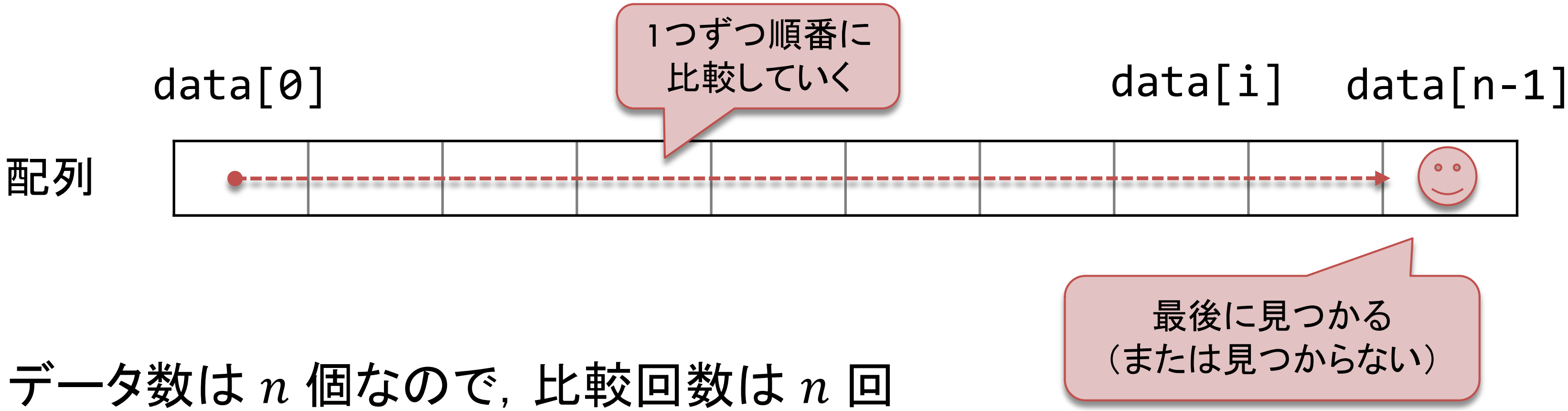
線形探索の考え方

- トランプにたとえると…
  - カードが裏返しに並んでいる（特に順番はない）
  - この中で探したいカードはどこにあるか？



- どうやって探すか？
  - とりあえず，左から（または右から）順にめくってみよう

線形探索の最大計算量



データ数は n 個なので，比較回数は n 回

比較以外にかかる時間は，n に関わらずほぼ一定（定数）

n が十分大きい場合（$n \to \infty$）には比較部分の影響だけ考えればよい

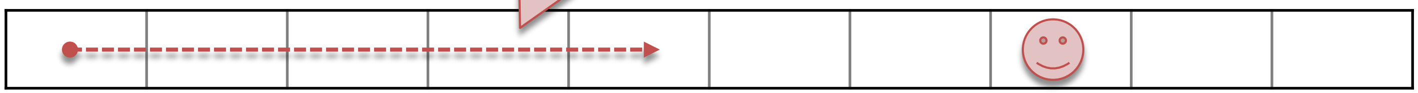
時間計算量は $O(n)$

線形探索の平均計算量

1つずつ順番に
比較していく

`data[0]` `data[i]` `data[n-1]`

配列



ここにkeyがある確率	$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	…	$\frac{1}{n}$	…	$\frac{1}{n}$
	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	…	$\times$	…	$\times$
発見までの比較回数	1	2	3	4	5	6	…	$i+1$	…	n
	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	…	$\parallel$	…	$\parallel$

比較回数の期待値

$$\frac{1}{n} + \frac{2}{n} + \frac{3}{n} + \frac{4}{n} + \frac{5}{n} + \frac{6}{n} + \cdots + \frac{i+1}{n} + \cdots + \frac{n}{n}$$

平均計算量

$$= \frac{1}{n}(1 + 2 + \cdots + n) = \frac{1}{n} \cdot \frac{n(n+1)}{2} = \frac{n+1}{2} = \frac{1}{2}n + \frac{1}{2}$$

$n \to \infty$ を考え
$O(n)$