アルゴリズムとデータ構造

第3回　2分探索法と計算量

第3回のキーワード

アルゴリズム関係

- 2分探索（binary sarch）
- $O(\log n)$
- 再帰による2分探索

Java関係

- compareTo

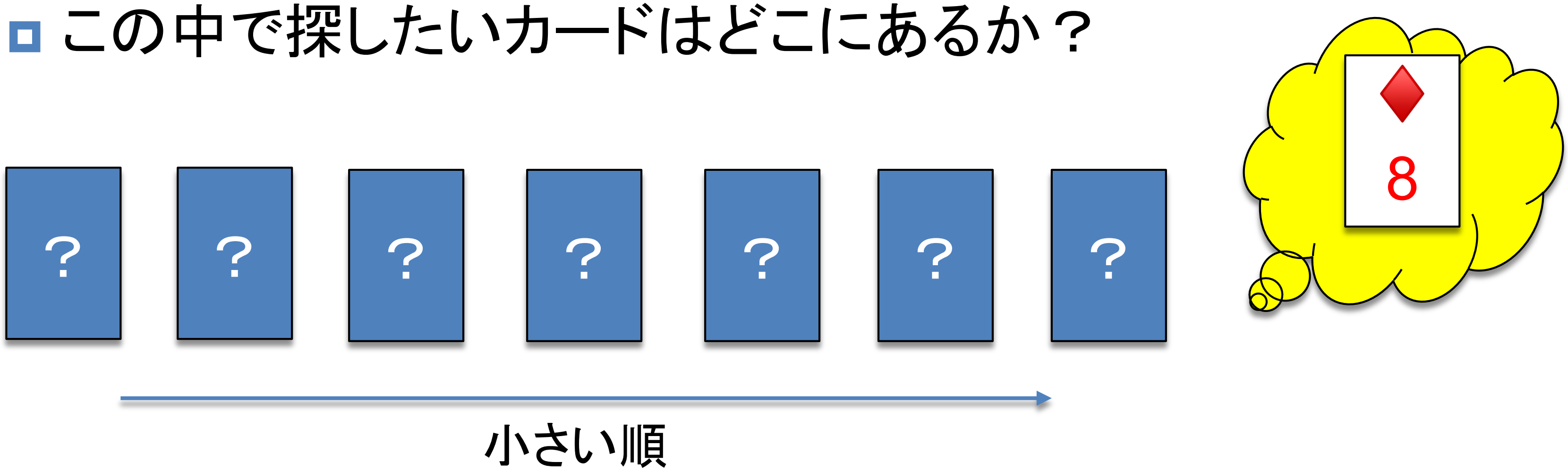
この資料は図解が中心なので，文章は説明不足の部分もあります。
講義内容をノートに取り、演習課題や参考書の説明もよく読みましょう。

もっと速い探索方法はないの？

- 前回紹介「線形探索」
  - 先頭から順番に調べていくしかないのか？
  - コンピュータなら，1000人の名前がバラバラに並んだ名簿からでも一瞬で検索できるけど…
- データ構造を工夫する
  - もっと効率的に探し出すためには，どうすればいいか？
  - 世の中の情報は，どのように整理してあるだろうか？
- 探索の高速化戦略
  - 整列しておく ⇒ 今回説明
  - 分類しておく ⇒ 似たようなことは次回以降説明

2分探索の考え方

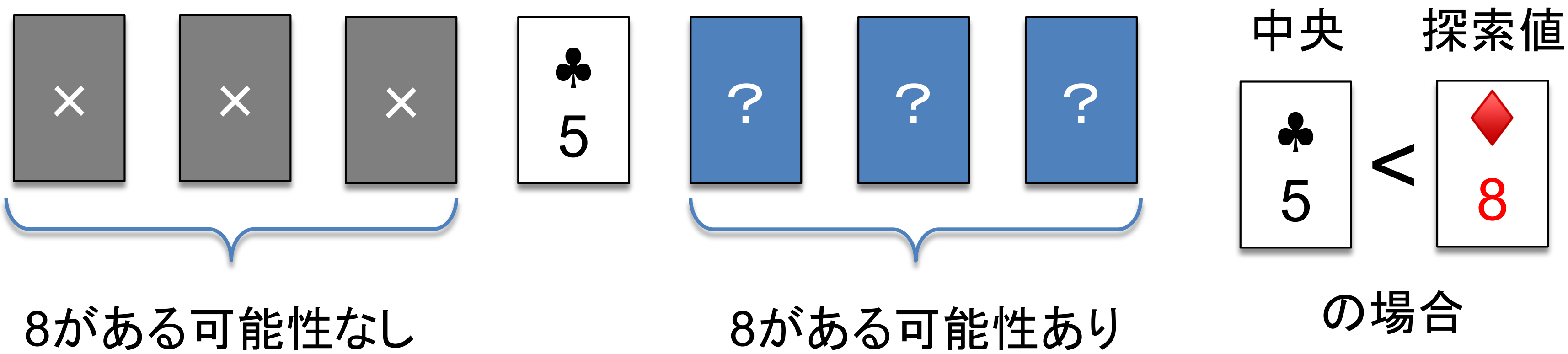
- トランプにたとえると…
  - 裏返しのカードが，数の**小さい順**に並んでいる
  - この中で探したいカードはどこにあるか？

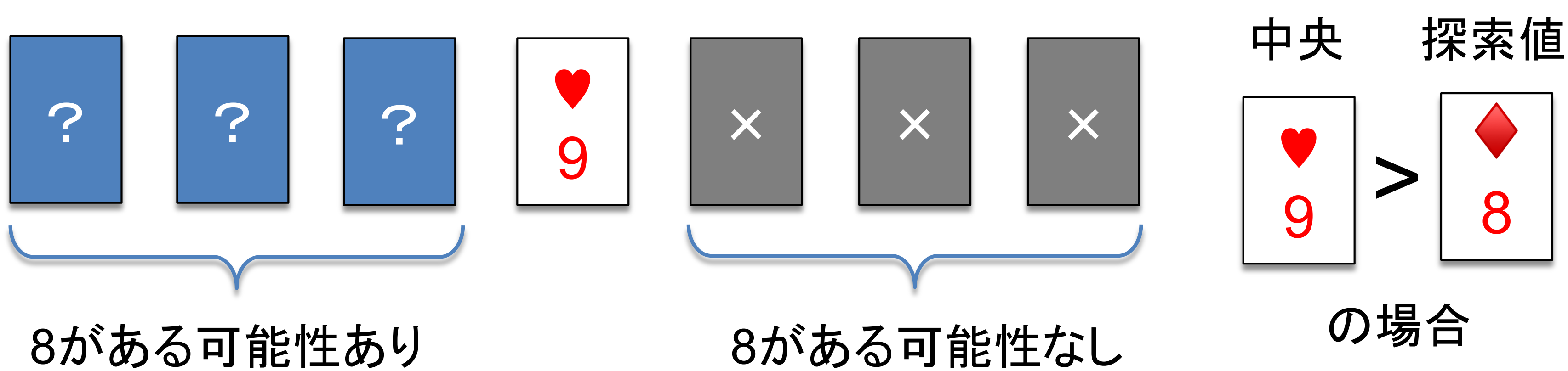


- 戦略を考えてみよう
  - まず，真ん中のカードを開けると何がわかるか？
  - 候補が半分ずつに減っていく（半分→4分の1→8分の1…）

2分探索の考え方

- 中央を開けると探索範囲を半分に絞り込める





2分探索のアルゴリズム

- データ
  - 配列の要素は，小さい順（または大きい順）に並べておく
  - 探索する値をkeyとする

この準備が高速化のポイント！
（タダで高速化できているわけではない）

- アルゴリズム
  - 配列（探索範囲）の中央にある値とkeyを比較する
  - もし両者が等しければ，発見したのでその位置を返す
  - もしkeyの方が小さければ，探索範囲を前半分にせばめる
  - もしkeyの方が大きければ，探索範囲を後半分にせばめる
  - 以上の手順を，探索範囲に要素がなくなるまで繰り返す
  - 探索範囲に要素がなくなったら，keyは含まれていない

確認問題

- 2分探索の理解
  - 下記は線形探索の確認問題で使用した配列である

a									
6	8	2	3	5	9	1	7	4	0

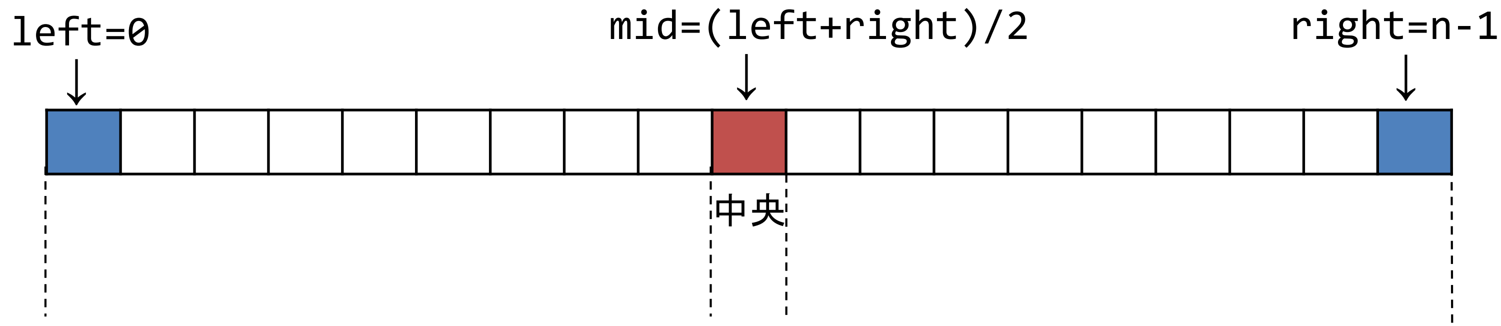
  - 配列aの内容を「2分探索」が適用できるように変更せよ

a									

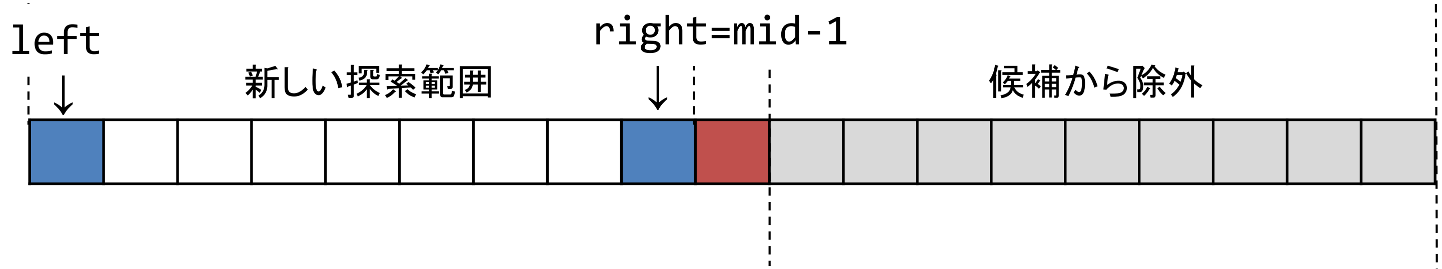
  - 変更された配列から2分探索で「8」を探すとき，発見するまでに比較される要素の値を順に示せ
    - 探索範囲が偶数個の場合，中央として先頭に近い方の要素を使う
  - 変更された配列の全要素について，2分探索の何回目の比較で発見できるか示せ

探索範囲のせばめ方

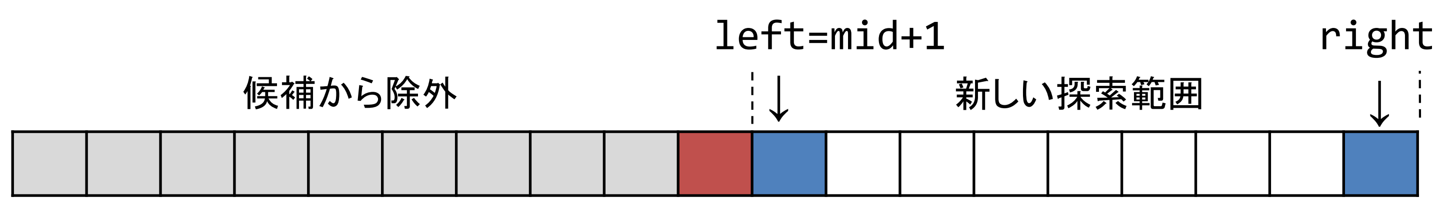
探索範囲の左端と右端を表す変数を使う



左半分にせばめるときは，右端を動かす



右半分にせばめるときは，左端を動かす



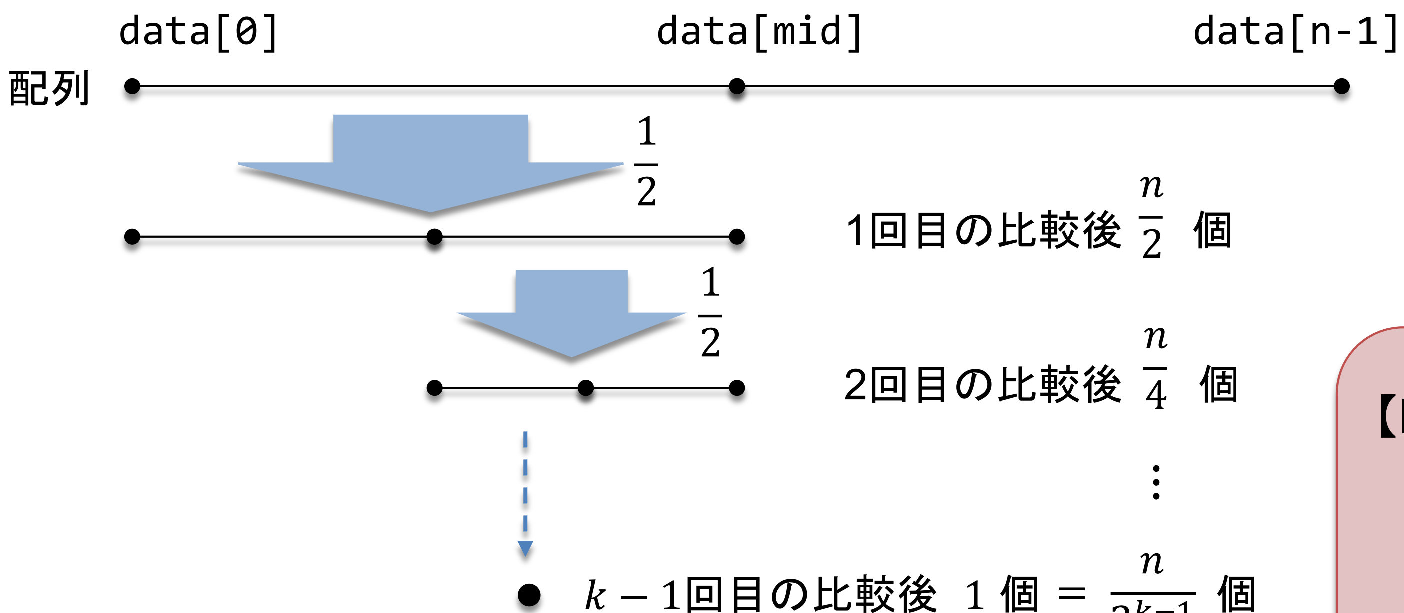
確認問題

- 必要な処理を追加してプログラムを完成させよ

```
public static int binarySearch(int key, int[] data) {

    int left = 0;   // 探索範囲の左端
    int right = data.length - 1;   // 探索範囲の右端

    while (left <= right) { // 探索範囲が1個以上である間繰り返す
        int mid =

        if (key == data[mid]) {

        } else if (key < data[mid]) {

        } else {

        }
    }
    return -1;   // 発見できなかった場合は-1を返す
}
```

2分探索の最大計算量

総比較回数を仮にk回とおき，kを求める



1回目の比較後 $\frac{n}{2}$ 個

2回目の比較後 $\frac{n}{4}$ 個

⋮

$k-1$回目の比較後 1 個 $= \frac{n}{2^{k-1}}$ 個

残りが1個になるのは，最後の比較をする直前だから，$k-1$回目の比較後である

$$\frac{n}{2^{k-1}} = 1$$

$$n = 2^{k-1}$$

$$k = \log_2 n + 1$$

$n \to \infty$ を考え
$O(\log n)$

【logの定義】

$y = a^x$

のとき

$x = \log_a y$

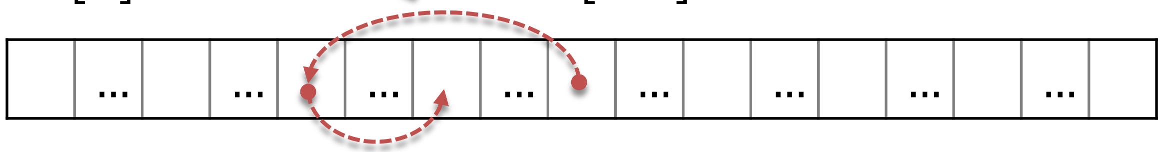
2分探索の平均計算量

簡単のため $n = 2^k - 1$ とする

中点→“四分”点→“八分”点と比較

data[0] data[mid] data[n-1]

配列



そこにkeyがある確率 $\frac{1}{n}\ \frac{1}{n}\ \frac{1}{n}\ \frac{1}{n}\ \cdots\ \frac{1}{n}\ \cdots\ \frac{1}{n}\ \cdots$

比較回数の期待値

$\times\ \times\ \times\ \times\ \cdots\ \times\ \cdots\ \times\ \cdots$

1回目で発見 1個 1 $= \frac{1}{n} \times 1 \times 1$

2回目で発見 2個 $2 \quad 2$ $= \frac{1}{n} \times 2 \times 2$

3回目で発見 4個 $3 \quad 3 \quad 3 \quad 3$ $= \frac{1}{n} \times 3 \times 4$

⋮ … … … … … … … … ⋮

k回目で発見 2^{k-1}個 $k\ k\ k\ k\ k\ k\ k\ k\ \ldots$ $= \frac{1}{n} \times k \times 2^{k-1}$ +

$$\frac{1}{n}\left(1 \cdot 1 + 2 \cdot 2 + 3 \cdot 4 + \cdots + k \cdot 2^{k-1}\right)$$

全部で $\frac{1-2^k}{1-2}$ 個

さらに $n = 2^k - 1$ を使って n の式にせよ