

# アルゴリズムクイック リファレンス

Algorithms in a Nutshell

第5章：探索  
お気持ちスライド

@rian\_tkb

2018/05/24

# 5章：探索

## ▶ この章の内容

### ▶ **線形探索、二分探索、ハッシュ、二分探索木**

▶ けっこう重い章で時間がやばいと思うので  
お気持ちスライドは必要最低限、**二分探索**だけにします



# お気持ちスライドの内容

3

- ▶ 二分探索をバグらせずに書けるようにしよう！
  - ▶ このスライドを理解すれば、一生二分探索がバグらなくなる！（ほんまか？）



# 二分探索

- ▶ この本の二分探索は閉区間でやっている

- ▶ 閉区間の説明は後でします

```
// 要素数 n のソート済み配列 A に t が存在すれば  
// その index を、存在しなければ -1 を返す
```

```
int search(int* A, int n, int t) {  
    int l = 0, r = n - 1;  
    while (l <= r) {  
        int m = (l + r) / 2;  
        if (A[m] == t) return m;  
        else if (A[m] > t) r = m - 1;  
        else l = m + 1;  
    }  
    return -1;  
}
```

# 二分探索

- ▶ じゃあこういう時どうする？
  - ▶ 要素数  $n$  のソート済み配列の中で、 **$t$  以上で最小の要素の index** がほしい！ ( $t$  以上の要素がなければ  $n$  を返す)
  - ▶ lower bound と呼ばれる処理



©A.H/YCP

2018/05/24

教科書輪講

# 二分探索

```

int search(int* A, int n, int t) {
    int l = 0, r = n;
    while (l < r) {
        int m = (l + r) / 2;
        if (A[m] >= t) r = m;
        else l = m + 1;
    }
    return l;
}

```

▶ .....ややこしい！

- ▶ 適当に書いたのでミスってるかも



©A.H/YCP

# 二分探索

- ▶ 他にもいくつか方法があります
  - ▶  $A[m] \geq t$  となる  $m$  を  $ans$  に格納しておいて、それを返す、など
  - ▶ どちらにせよややこしいし、解きたい問題に対していちいちこんなこと考えてたら無限にバグりますね、間違いない



# 二分探索

8

- ▶ そこで、二分探索を「**値を探すもの**」ではなく  
**「境界を探すもの」**だと考える

この人が探しているのは不可視境界線





りあん  
@rian\_tkb

9

クソみたいな輪講スライドが生成されている

## 二分探索

8

- ▶ そこで、二分探索を「値を探すもの」ではなく「境界を探すもの」と考える

この人が探しているのは不可視境界線



2018/05/24 教科書輪講

12:50 - 2018年5月23日

27件のリツイート 128件のいいね



1

27

128

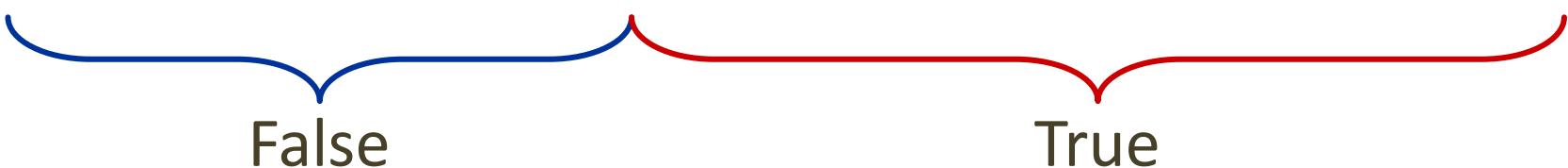
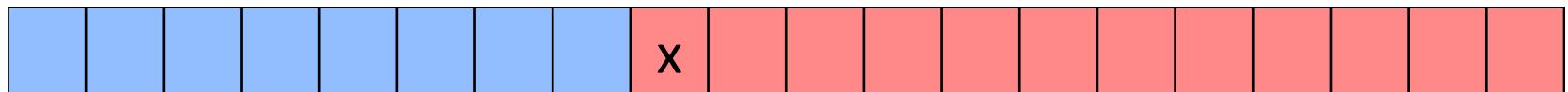
..

2018/05/24

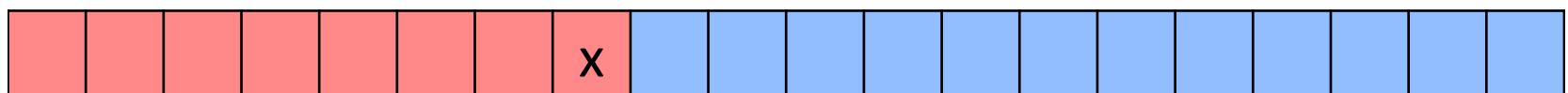
教科書輪講

# 二分探索

- ある真偽値を返す関数  $\text{pred}(m)$  が  $m < x$  において False、  
 $x \leq m$  において True を返すとする (T/F が逆でも可)
  - その境界  $x$  を知りたい
  - 今回の問題の場合、 $\text{pred}(m) = (\text{A}[m] \geq t)$ ;



逆の場合



# 二分探索

```

bool pred(int* A, int t, int m) {
    return A[m] >= t;
}

int search(int* A, int n, int t) {
    int ng = -1, ok = n;
    // ok - ng == 1 になるまで回る
    while (ok - ng > 1) {
        int m = (ng + ok) / 2;
        // m が pred(m) を満たすなら ok に
        // 満たさないなら ng に入れる
        if (pred(A, t, m)) ok = m;
        else ng = m;
    }
    return ok;
}

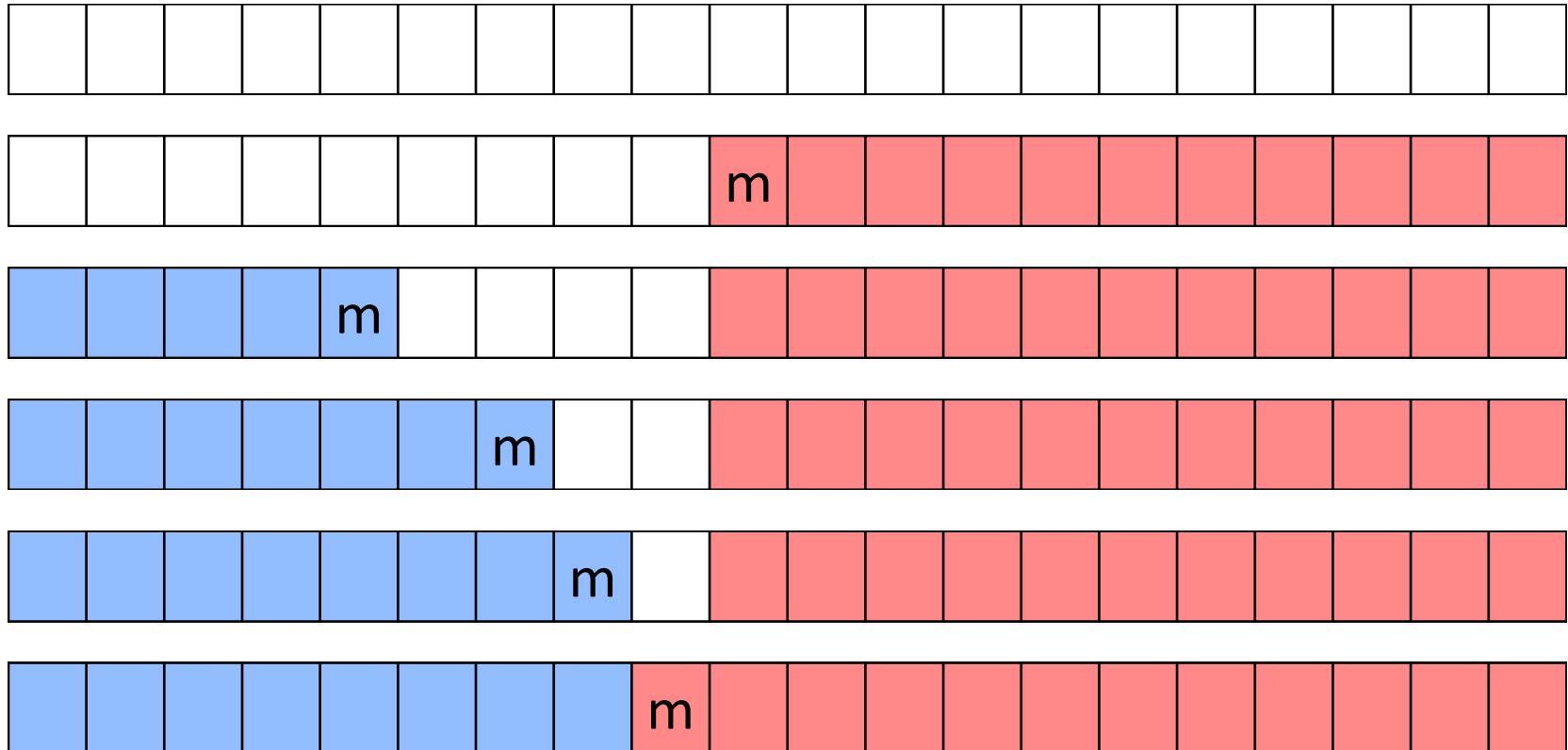
```

▶ こうなった



# 二分探索

- ▶ イメージとしては、ok の区間と ng の区間をだんだんと広げていくイメージ



# 二分探索

13

- ▶ `ok`, `ng` の初期値の設定が難しいが、定義域から 1 だけ外れた値を設定しても `m` が定義域から出ることはないので（大体の場合）大丈夫
- ▶ `m` が小さい方が `True` となる場合でも同じように回せる
  - ▶ 両方同じように書けるように `while (abs(ok - ng) > 1)` と書く人もいる



©A.H/YCP

# 二分探索

14

- ▶ これで、終了条件とか  $m + 1$  だけ  $m$  だけ  $m - 1$  だけとか、二分探索の中の面倒な条件を考えずに  $\text{pred}(m)$  さえ考えてしまえばもう二分探索が書ける！
  - ▶ これならバグらせにくい！



©A,H/YCP

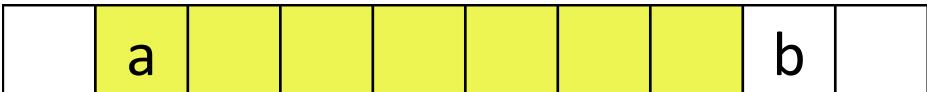
2018/05/24

教科書輪講

# 二分探索

- ▶ この本の二分探索は**閉区間**で書いてあったが、この二分探索は（解の範囲が  $[ok, ng)$  なので）**半開区間に**近い
  - ▶ 半開区間を意識してプログラミングをした方がスッキリしたコードが書けるケースが時々あったりなったりする
  
- ▶ 閉区間  $[a, b] = \{ a, a + 1, \dots, b - 1, b \}$ 

  
- ▶ 開区間  $(a, b) = \{ a + 1, a + 2, \dots, b - 2, b - 1 \}$ 

  
- ▶ **半開区間**  $[a, b) = \{ a, a + 1, \dots, b - 2, b - 1 \}$ 


# 二分探索

- ▶ 今回説明した二分探索は「めぐる式二分探索」と呼ばれることもあります
  - ▶ [https://twitter.com/meguru\\_comp/status/697008509376835584](https://twitter.com/meguru_comp/status/697008509376835584)
  - ▶ <http://chokudai.hatenablog.com/entry/2015/04/23/194909>
- ▶ 例題
  - ▶ [https://beta.atcoder.jp/contests/abc023/tasks/abc023\\_d](https://beta.atcoder.jp/contests/abc023/tasks/abc023_d)
  - ▶ [https://beta.atcoder.jp/contests/arc075/tasks/arc075\\_b](https://beta.atcoder.jp/contests/arc075/tasks/arc075_b)
  - ▶ [https://beta.atcoder.jp/contests/arc084/tasks/arc084\\_a](https://beta.atcoder.jp/contests/arc084/tasks/arc084_a)
  - ▶ [https://beta.atcoder.jp/contests/code-festival-2015-quala/tasks/codefestival\\_2015\\_qualA\\_d](https://beta.atcoder.jp/contests/code-festival-2015-quala/tasks/codefestival_2015_qualA_d)
  - ▶ <https://yukicoder.me/problems/no/448>

# まとめ

17

- ▶ 二分探索を書くのが難しい、という状態から、  
二分探索に落とすのが難しい、という状態にランクアップしよう！
  - ▶ 二分探索の経験をそこそこ積めば、二分探索に落とせるか  
どうかはわりとわかるようになるので（ほんまか？）、  
これであなたも二分探索マスター



©A.H/YCP

2018/05/24

教科書輪講