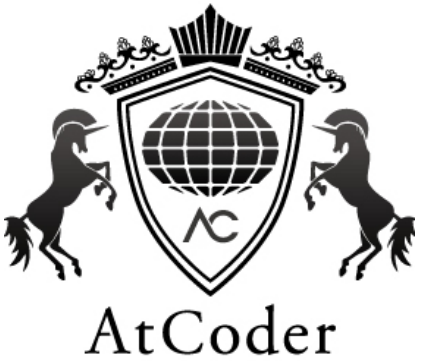
実践・最強最速のアルゴリズム勉強会
第五回　講義資料



AtCoder株式会社 代表取締役
高橋 直大

はじめに！

AtCoder

- 本講義では、ソースコードを扱います。
- 前面の資料だけでは見えづらいかもしれないので、手元で閲覧できるようにしましょう。
- URLはこちらから
  - http://www.slideshare.net/chokudai/wap-atcoder5
  - URLが打ちづらい場合は、Twitter: @chokudaiの最新発言から飛べるようにしておきます。
    - フォローもしてね！！！
  - はじっこに座ると真ん中に入れなくなるので、真ん中の方に座ってください！

目次

1. 勉強会の流れ
2. 難問に挑戦！
3. 本日のまとめ

勉強会の流れ

1. 勉強会の日程
2. 1日の流れ

勉強会の日程

AtCoder

1日目　シミュレーションと全探索

2日目　色々な全探索

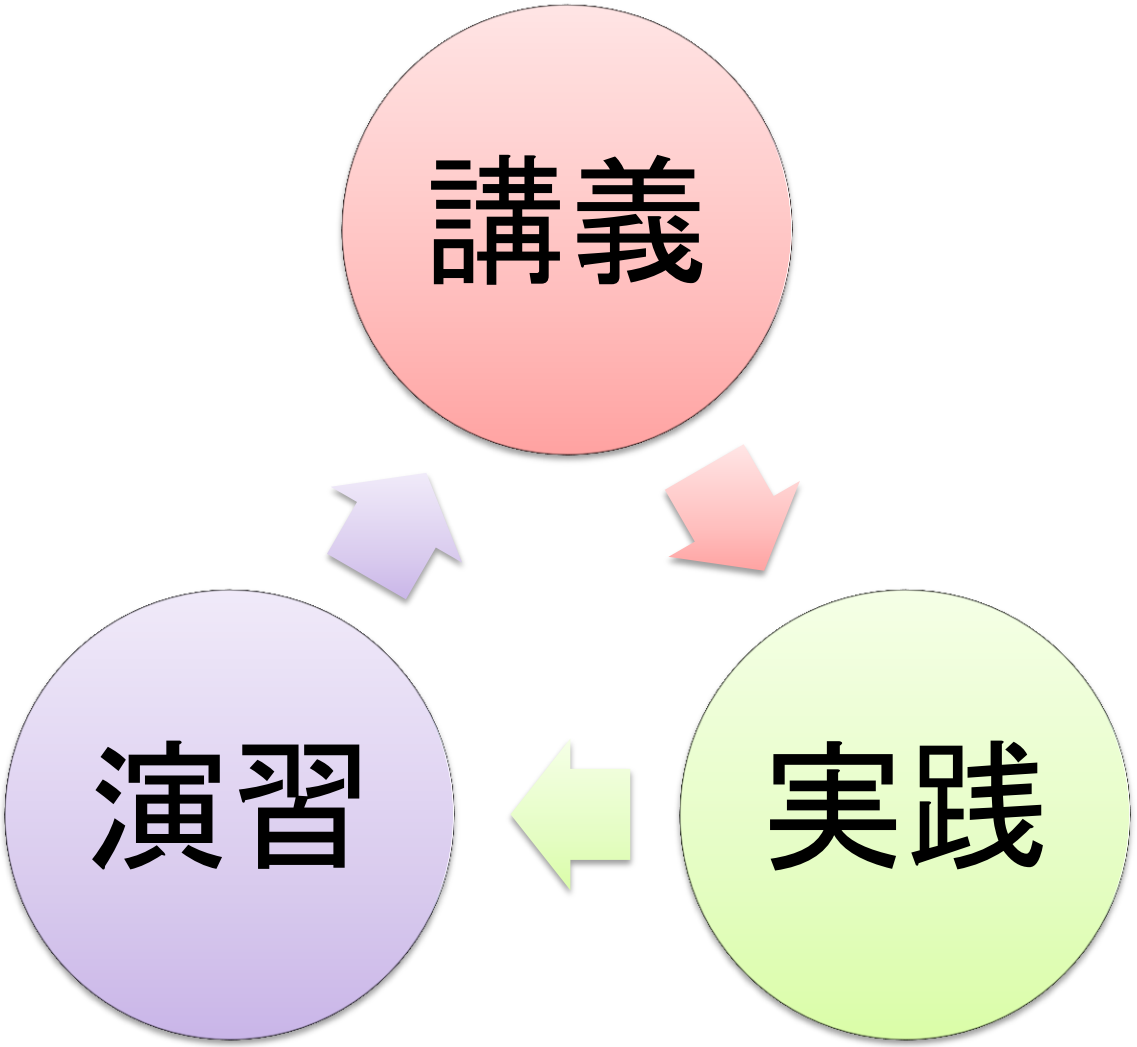
3日目　動的計画法とメモ化再帰

4日目　動的計画法と計算量を減らす工夫

5日目　難問に挑戦！

１日の流れ

AtCoder



１日の流れ

AtCoder

講義

- 基礎的なアルゴリズムを学ぶ
- 必要な知識を補う

実践

- 実際に問題例を見る
- コードでの表現方法を覚える

演習

- 自分で問題を解いてみる！
- コードを書いて理解を深める

１日の流れ

AtCoder

- 演習について
  - 実力差があると思うので、暇な時間が出来る人、ついていけない人、出ると思います。
    - どうしようもないので、早い人は支援に回ってもらえると嬉しいです！
  - 解らないことがあったら、#WAP_AtCoderでTwitterに投稿！
    - 多分早く終わった人が質問回答してくれます。
    - 具体例はこんな感じ
      - 「今やってる問題どれですか！」
      - 「この解答のWAが取れません！　http:// ~~」
      - 「コンパイルエラー出るよーなんでー＞＜　http:// ~~」
      - とりあえずコードが書けてたら、間違ってても提出してURLを貼りつけよう
  - もちろん、手を上げて質問してくれてもOK!
    - 回りきれる範囲では聞きに行きます。

今日の流れ

1. 前回までの復習
2. 今回やること

前回までの復習

AtCoder

- 全探索
  - 枝分かれの数が定数の場合 （第一回）
    - N重のforループによる全探索
  - 枝分かれの回数が不定の場合 （第二回）
    - 深さ優先探索
    - 幅優先探索
    - Bitを利用した深さ優先探索
      - 枝分かれが二股の時限定

前回までの復習

AtCoder

- 計算が間に合わない場合
  - 計算量の概念を用いた計算時間の予測（第３回）
  - メモ化再帰（第３回）
    - 深さ優先探索をメモするようにしたもの
  - 動的計画法（第３・４回）
    - メモ化再帰の向きを逆転し、forループのみで書けるように！
  - 貪欲法（第４回）
    - 探索せずに、正しい方針を決め打ちできるように！

今日やること

AtCoder

- 難問に挑戦！
  - これまで習ったツールのみで解くことが出来る、最高峰の問題に挑戦してみよう！

今日の目次

1. 難しい問題がなぜ難しいか？
2. 今日取り組む問題
3. 全探索で難しい問題
4. 動的計画法で難しい問題

難しい問題がなぜ難しいか？

AtCoder

- よくある誤解
  - 「全探索」の問題はかんたん
  - 「動的計画法」の問題は難しい
  - 「最小費用流」の問題はものすごく難しい

- こういうむずかしさの分類は、全然あてにならない！
  - 難しさを表す1要因でしかない

難しい問題がなぜ難しいか？

AtCoder

- 競技プログラミングにおけるむずかしさって？
  - スカラー値で表すのであれば、「解けた人数」で表せる
  - しかし、問題によって、難しさの質が違う
    - なぜ難しいのかを考えよう！

難しい問題がなぜ難しいか？

AtCoder

- 競技プログラミングにおける、三大要素
  - 実装力
    - 解き方が分かった時に、そのプログラムをミスなく書くことが出来るか？
    - どれだけコード長が長いか？どれだけ複雑なプログラムになるか
  - 知識力
    - 問題で要求されている、典型アルゴリズムを知っているか否か
  - 発想力
    - 問題で要求されている回答にたどり着くまでに、どれだけの論理展開が必要か？
    - 知識が十分にあると仮定した上で、どれだけ回答に近づきやすいか？
- 三つの要素は完全に独立ではない
  - 知識をアイデアで補ったり、発想を他の問題の知識で補ったり

難しい問題がなぜ難しいか？

AtCoder

- 実装力
  - コーディングを、早く正確に出来るか？
  - 短くまとまったコードに落とし込めるか？
  - 複雑なコードでも手を止めずにコードを書けるか？

- 攻略方法
  - バグなく素早く実装するのは練習あるのみ！
    - 人によっては、タイピング練習もした方が良い場合も？

難しい問題がなぜ難しいか？

AtCoder

- 実装力が問われる問題
  - 天下一プログラマーコンテスト2013 決勝C問題
    - http://tenka1-2013-final.contest.atcoder.jp/tasks/tenka1_2013_final_c
    - ジグソーパズルの探索。実装量が非常に多い

難しい問題がなぜ難しいか？

AtCoder

- 知識力とそれに対応するアルゴリズム
  - データ構造も含む　　ここに書いたのは一部！

Forのみの全列挙　ダイクストラ　半分全列挙　Euler Tour

深さ優先探索　ワーシャルフロイド　凸包　Nim

幅優先探索　Union-Find　2-SAT　Grundy数

貪欲法　二分探索　Fenwick Tree　最小費用流　フロンティア法

ソート　動的計画法　座標圧縮　最大流　高速フーリエ変換

二部マッチング　強連結成分分解

優先度付キュー　しゃくとり法　行列累乗　Link-Cut Tree

平衡二分木　ダブリング　Segment Tree

難しい問題がなぜ難しいか？

AtCoder

- 知識力　攻略方法
  - さまざまな方法で知識をつけよう！
    - 本を読む
      - 最強最速アルゴリズマー養成講座
      - プログラミングコンテストチャレンジブック
    - コンテストに出る
      - 解けなかった問題を復習することで新たなアルゴリズムを発見
    - 研究をする
      - 得意分野のアルゴリズムは自動的に詳しくなる！

難しい問題がなぜ難しいか？

AtCoder

- 知識力が問われる問題
  - QUPC2014 G 立ち入り禁止区域
    - http://qupc2014.contest.atcoder.jp/tasks/qupc2014_g
    - 凸包を求めるアルゴリズムを知っているかどうかが問われる
  - AtCoder上ではあまり多くありません。

難しい問題はなぜ難しいか？

AtCoder

- 発想力
  - 最終的にやることが同じでも、問題によって解き方が全く違う。
  - 問題文を見てから、その回答に辿り着くまで、どれだけのステップがあるか？

難しい問題はなぜ難しいか？

AtCoder

- 同じ問題でも、発想力の要求度合いは変えられる

- 具体例
  - ABC006 スフィンクスのなぞなぞ
    - 「この街には人間が N 人いる。人間は、大人、老人、赤ちゃんの 3 通りだ。
      この街にいる人間の、足の数の合計は M 本で、大人の足は 2 本、老人の足は 3 本、赤ちゃんの足は 4 本と仮定した場合、存在する人間の組み合わせとしてあり得るものを 1 つ答えよ。」

難しい問題はなぜ難しいか？

AtCoder

- 同じ問題でも、発想力の要求度合いは変えられる
  - 「ある規則性」を見つける必要がある問題である時
    - その規則性を見つけやすい入出力例を用意するか否か
    - そもそもその規則性を問題文中に書いてしまえば、同じ問題でも難易度はガクッと下がる
  - コーナーケースが存在する時
    - そのコーナーケースを入出力例で用意してしまえば、嵌る人は減る
      - これも大きな難易度減少となる

難しい問題はなぜ難しいか？

AtCoder

- 発想力　攻略方法
  - 問題に対する慣れが一番大切！
    - 「知識力」は、問題に対する大きな括りでの知識
    - 「発想力」は、小さな問題に対して、「この問題はあの問題と似てる！」といったような関連付けを行い、発想を生み出す手助けとなる。
  - 数学力なども相関が大きい？
    - このためだけに数学を学ぶのは少し非効率かも

難しい問題はなぜ難しいか？

AtCoder

- 発想力が問われる問題
  - 貪欲法の大抵の問題
    - AtCoder Regular Contest 014 C問題 魂の還る場所
      - http://arc014.contest.atcoder.jp/tasks/arc014_3
  - 動的計画法の大抵の問題
  - 全探索で間に合わない大抵の問題

今日取り組む問題

AtCoder

- 知識力
  - これまでの4日間で、AtCoderで戦う上で、十分な概念の学習は出来ました！
- 実装力
  - これからAtCoderのコンテストに出場していくことで、高めていけます！
    - あんまり勉強会で教えるものでもないです
- 発想力
  - ここを考えるのが一番楽しいです！
    - たまに思いつかない自分に絶望して辛くなることも。「解けないのが普通」「解けたら自分すごい！」と思うようにしましょう。
  - 今回は、ここを問われる問題を重点的に！

全探索の難しい問題

全探索で難しい問題

AtCoder

- 全探索で難しい問題って？
  - 全探索は、「全て調べるだけ」の簡単な問題なはず？
    - それでも、難しいパターンが存在する！
  - 難しいパターンは、おおよそ以下の2通り
    - 実装力が求められるパターン
      - 全探索を実装すること自体が難しい！
      - 全探索でやることが複雑！
    - 発想力が求められるパターン
      - 全探索をどう使っていいかわからない！
      - 全探索に全く見えない！

全探索で難しい問題　実践

AtCoder

- 全探索　発想力要求レベル0
- AtCoder Regular Contest 001 センター採点
  - http://arc001.contest.atcoder.jp/tasks/arc001_1

- 問題概要
  - 4択クイズと、その回答が与えられる。
  - 高橋君は、1～4の全てに同じ答えを付けたが、どれを選んだか忘れてしまった。
  - 高橋君の正解数の最小値と最大値を求めなさい。

全探索で難しい問題　実践

AtCoder

- 入力例
  - 9　←文字数
  - 131142143　←解答
- 出力例
  - 4 1
  - 選択肢1を選ぶと4問正解となり、これが最高
  - 選択肢2を選ぶと1問正解となり、これが最低

全探索で難しい問題　実践

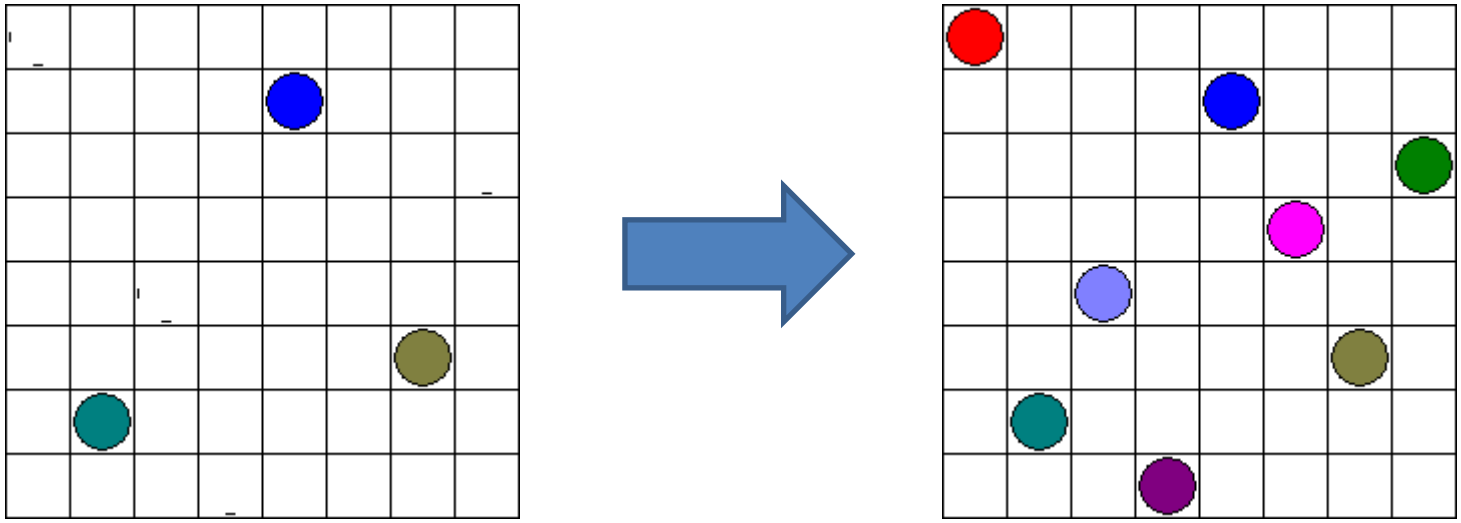
AtCoder

- 解説
  - 最早、問題が、「全探索をしなさい」と語りかけてきている
    - 1～4について全通り調べてみればいいんだね、と絶対思う
  - だったら素直に全探索してあげれば良い
  - 自明かつ第１回でも取り上げたので、コードは省略

全探索で難しい問題　実践

AtCoder

- ARC001 C問題　パズルのお手伝い
  - http://arc001.contest.atcoder.jp/tasks/arc001_3
  - 問題概要
    - 8*8のマス目が与えられます。8個の駒を、「たて」「よこ」「ななめ」で同じ列に存在しないように配置したいです。最初に3個配置されているので、残りのマスを埋めなさい。無理ならNo Answerと出力



全探索で難しい問題　実践

AtCoder

- 解説
  - 「全てのパターンを調べなさい」と問題文には書かれていない。
    - 何か工夫して置くようなパターンも考えられる
  - だが、ボードも小さいので、クイーンの置き方を全通り試せることが十分想像可能
    - 実装は結構大変
  - 深さ優先探索などで、クイーンの置き方の全パターンを試してあげれば良い。
    - ソースコードは省略

全探索で難しい問題　実践

AtCoder

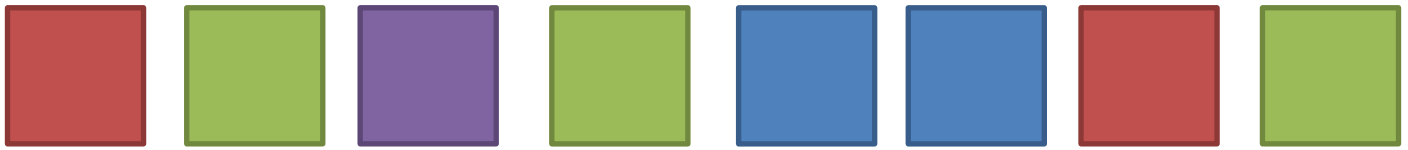
- 全探索　発想要求レベル2
  - AtCoder Regular Contest 020 縞模様
    - http://arc020.contest.atcoder.jp/tasks/arc020_2

- 問題概要
  - 色つき画用紙がn枚与えられる
  - これを縞模様（2種類の色が交互にくる模様）になるように、画用紙を塗り替えたい。
  - 塗り替える必要のある最小数を求めなさい。
    - 実際は絵具の費用があるので、これにcを掛ける。
  - n≦100、色の種類C≦10

全探索で難しい問題　実践

AtCoder

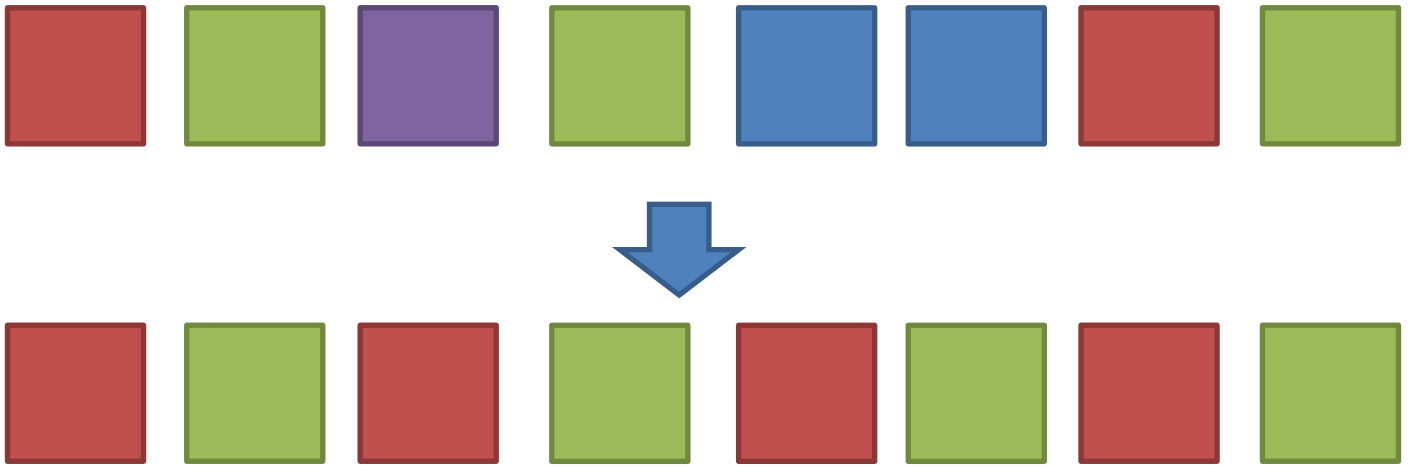
- 問題のイメージ
  - こんな感じで画用紙が与えられる
    - 入力例を示すのであれば、
      - 7 1
      - 1 2 3 2 4 4 1 2の様な感じ。本当はスペース区切りでなく改行



全探索で難しい問題　実践

AtCoder

- 問題のイメージ
  - こんな感じで画用紙が与えられる
    - こんな感じで塗り替える！
    - 今回の場合は3枚！



全探索で難しい問題　実践

AtCoder

- 全探索をしよう！
  - まずは最初の画用紙の塗り替えパターンが10通り！

全探索で難しい問題　実践

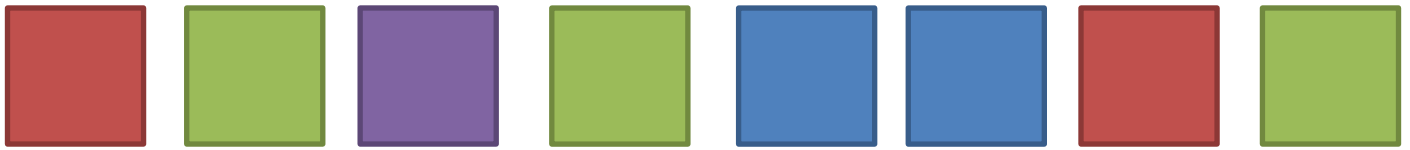
AtCoder

- 全探索をしよう！
  - まずは最初の画用紙の塗り替えパターンがC通り！
  - 次の画用紙もC通り！
    - こうやっていくと、計算量はO(C^n)
    - C = 10, n=100で、凄い大きな数になってしまう！　全探索出来ない！

全探索で難しい問題　実践

AtCoder

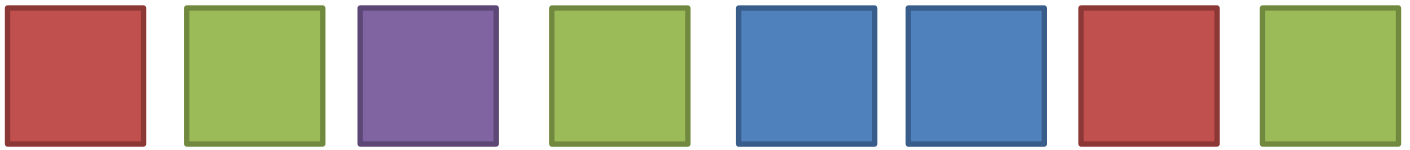
- 全探索出来ないの？
  - 「画用紙の全ての塗り替えパターン」を全探索することは出来ない！
  - だからと言って、別の全探索が出来ないわけではない



全探索で難しい問題　実践

AtCoder

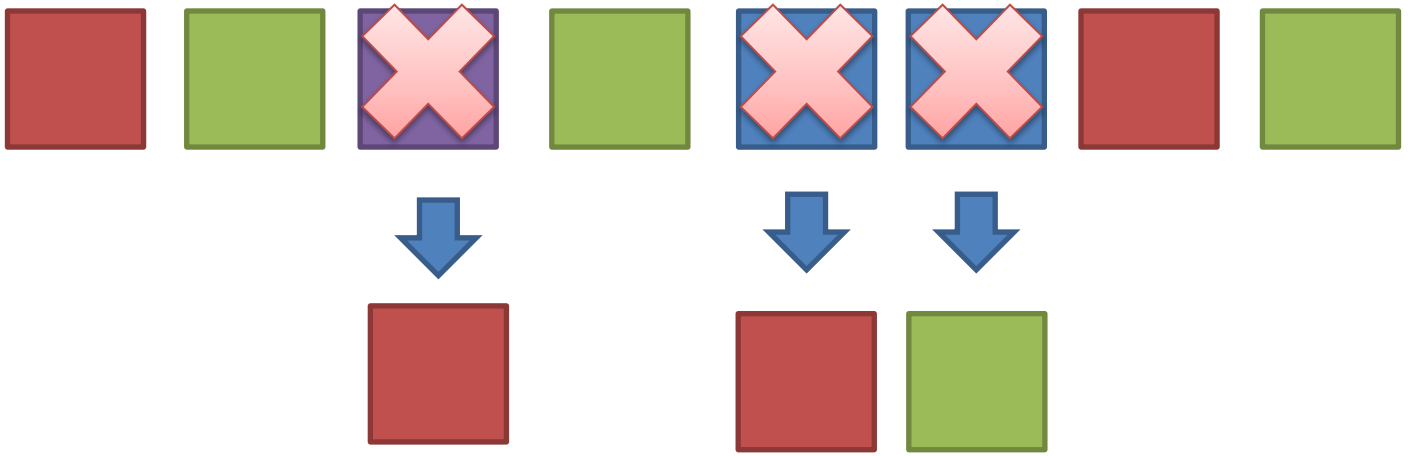
- 全探索出来ないの？
  - 「画用紙の全ての塗り替えパターン」を全探索することは出来ない！
  - だからと言って、別の全探索が出来ないわけではない



全探索で難しい問題　実践

AtCoder

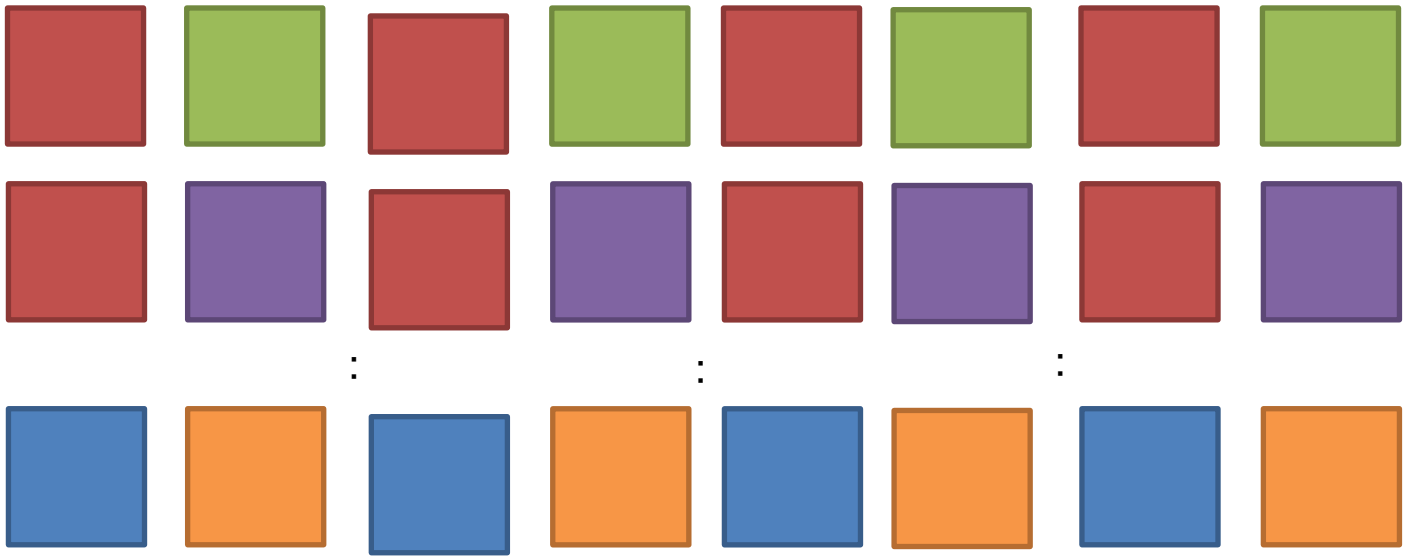
- 他の全探索って？
  - 例えば、塗り替える画用紙の全探索
  - 以下のように、塗り替える画用紙を全探索してあげる
    - 塗り替える色は周りに合わせる。同じ模様になるべき場所に2色以上あったら失敗
    - この計算量はO(2^n) n=100の時これでも遅い



全探索で難しい問題　実践

AtCoder

- さらに他の全探索
  - 最終的な縞模様の全探索
    - これは、O(C^2)通りしか存在しない！
  - この状態から、塗り替える必要のある画用紙の数を、全通り調べれば良い



全探索で難しい問題　実践

AtCoder

- ソースコード
  - http://arc020.contest.atcoder.jp/submissions/157049
    - 入力部

```
void run()
{
    Scanner cin = new Scanner(System.in);
    int n = cin.nextInt();
    int c = cin.nextInt();

    int[] paper = new int[n];
    for(int i=0;i<n;i++){
        paper[i] = cin.nextInt();
    }
```

全探索で難しい問題　実践

AtCoder

- ソースコード
  - http://arc020.contest.atcoder.jp/submissions/157028

```
int ret = n;
for(int i=0;i<c;i++){ //1番目の画用紙の色
    for(int j=0;j<c;j++){ //2番目の画用紙の色
        if(i==j) continue;
        int count = 0;
        for(int k=0;k<n;k++){
            if(k%2==0){
                //偶数番目 違ったらcount++
                if(paper[k]!=i) count++;
            }
            else{
                //奇数番目 違ったらcount++
                if(paper[k]!=j) count++;
            }
        }
        //最小値の更新
        ret = Math.min(ret, count);
    }
}
System.out.println(ret * c);
```

全探索で難しい問題　実践

AtCoder

- 全探索は、問題に対して1通りとは限らない！
  - 何を全探索するのか？
  - どう全探索するのか？

- 問題によって、「何が全探索出来るか」を考えなければならない！

全探索で難しい問題　実践

AtCoder

- 全探索　発想要求レベル2　改題
  - AtCoder Regular Contest 020 縞模様
    - http://arc020.contest.atcoder.jp/tasks/arc020_2

- 問題概要
  - 色つき画用紙がn枚与えられる
  - これを縞模様（2種類の色が交互にくる模様）になるように、画用紙を塗り替えたい。
  - 塗り替える必要のある最小数を求めなさい。
    - 実際は絵具の費用があるので、これにcを掛ける。
  - n≦10、色の種類C≦100000

全探索で難しい問題　実践

AtCoder

- この改題の時、
  - 塗り分けパターンの全探索
    - 計算量はO(C^n)
    - C=100000, n=10で、計算回数は10^50 地球爆発
  - 使う画用紙の全探索
    - 計算量はO(2^n)
    - n=10で、計算回数は1024 一瞬！
  - 塗り分けパターンの全探索
    - 計算量はO(C^2)
    - C=100000で、計算回数は10^10 100秒くらいかかる！
- 今度は、2番目に紹介した全探索が有利になった！

全探索で難しい問題　実践

AtCoder

- 全探索を考えるコツ
  - 「何が少ない」「何が多い」を把握する
  - 今回の問題で言えば、
    - 「色の種類が非常に少ない」
      - 累乗の肩にも乗れる
    - 「カードの枚数は少な目」
      - n^3くらいまでなら大丈夫そう
      - 累乗の肩には乗れない
  - という意識を持てば、余計な全探索は考えず、適切な全探索を考えることが可能

休憩！

全探索で難しい問題　演習

AtCoder

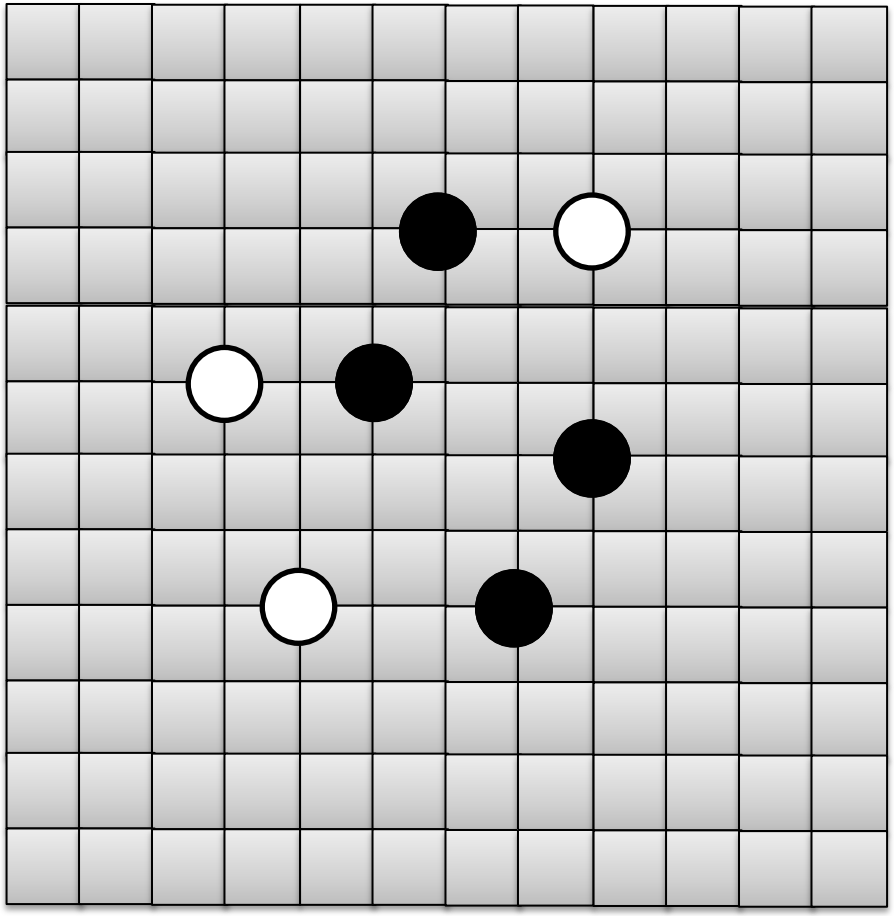
- 全探索　発想要求レベル3
  - AtCoder Regular Contest 012 C問題 五目並べチェッカー
    - http://arc012.contest.atcoder.jp/tasks/arc012_3

- 問題概要
  - 19*19の、五目並べの盤面が与えられる
  - 五目並べのルールは、以下のようなものである
    - 石をボード上に交互に打つ
    - 自分の石を5つ以上連続で、たて・よこ・ななめのいずれかの方向で揃えたら勝ち。その時点でゲームが終了する
  - 五目並べを進行した上で、登場し得る局面であればYES、そうでなければNOと出力しなさい

全探索で難しい問題　演習

AtCoder

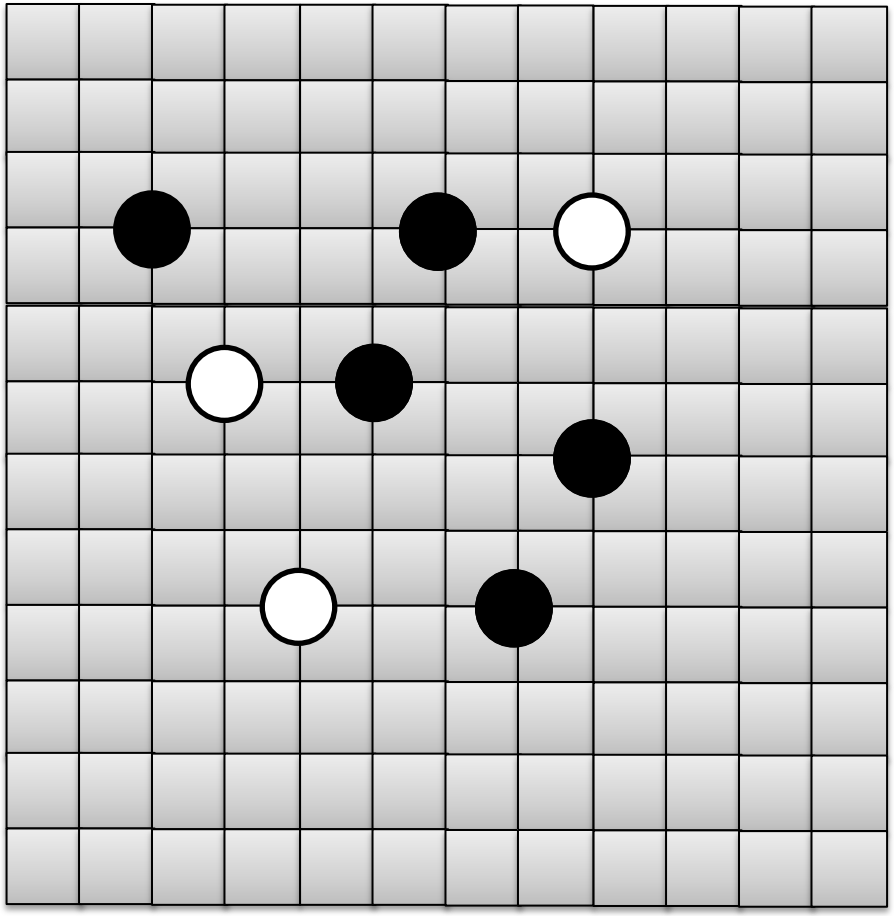
- 問題例
  - 先手の黒が4回、白が3回打ったところ
  - 黒が先行なので、普通に進行してあり得る状態なので、YES



全探索で難しい問題　演習

AtCoder

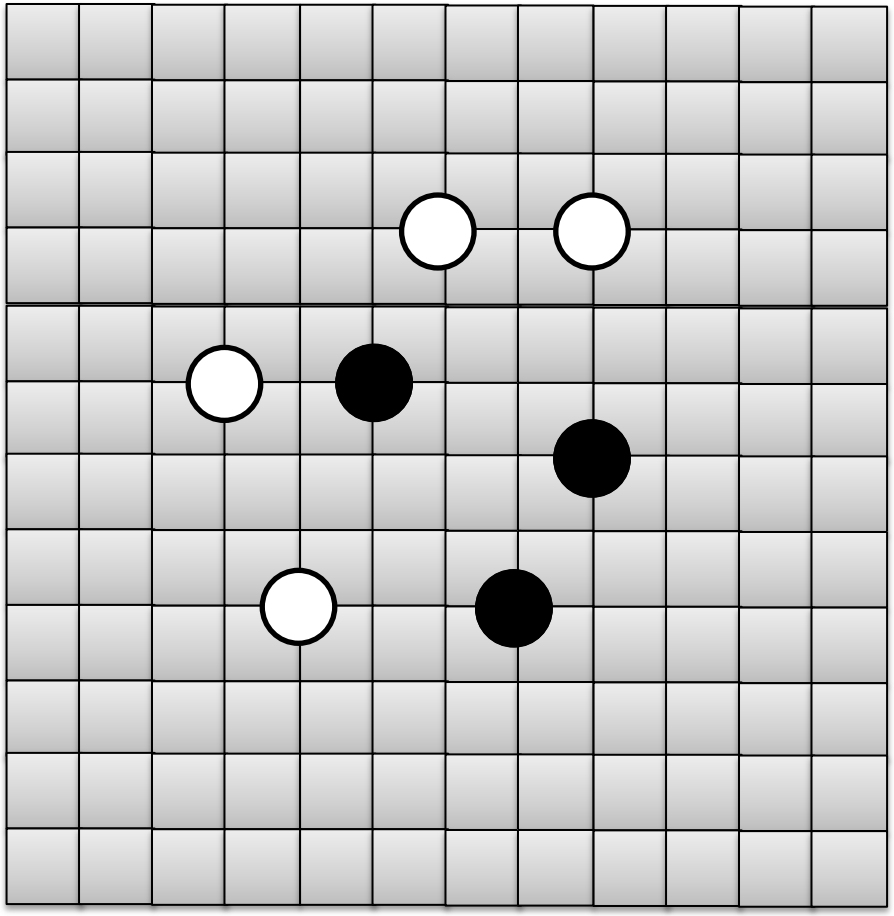
- 問題例
  - 先手の黒が5回、白が3回打っている
  - これは、黒が2連打していないとあり得ないのでNO



全探索で難しい問題　演習

AtCoder

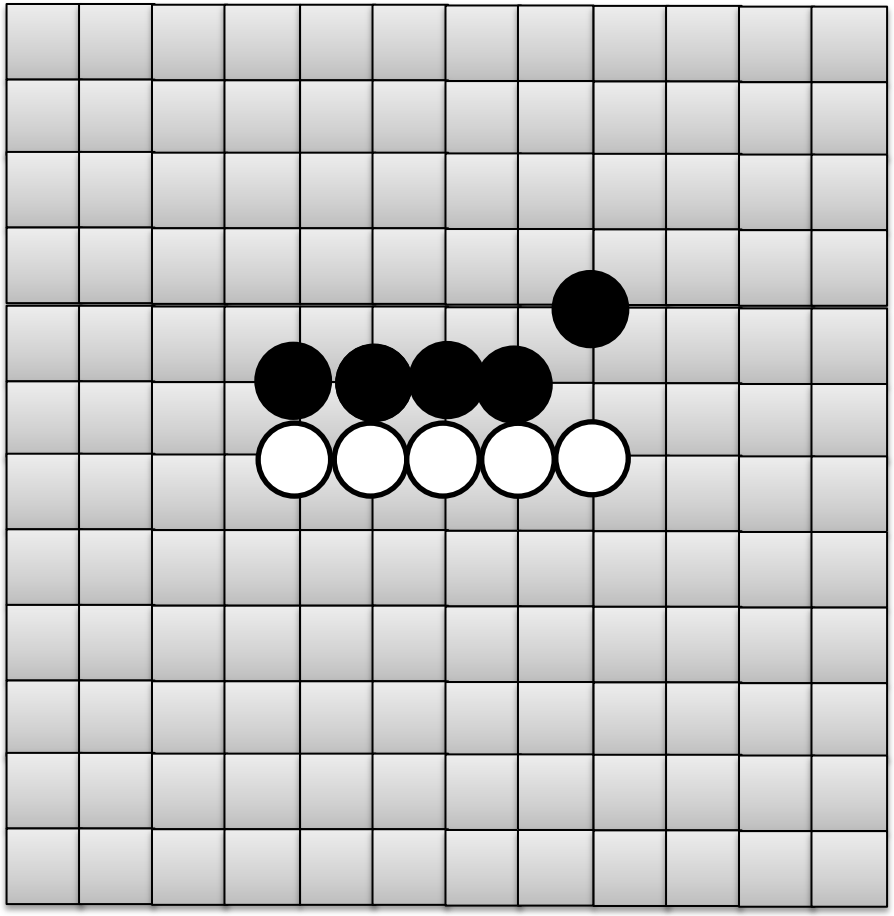
- 問題例
  - 先手の黒が3回、白が4回打っている
  - これもあり得ない。NO



全探索で難しい問題　演習

AtCoder

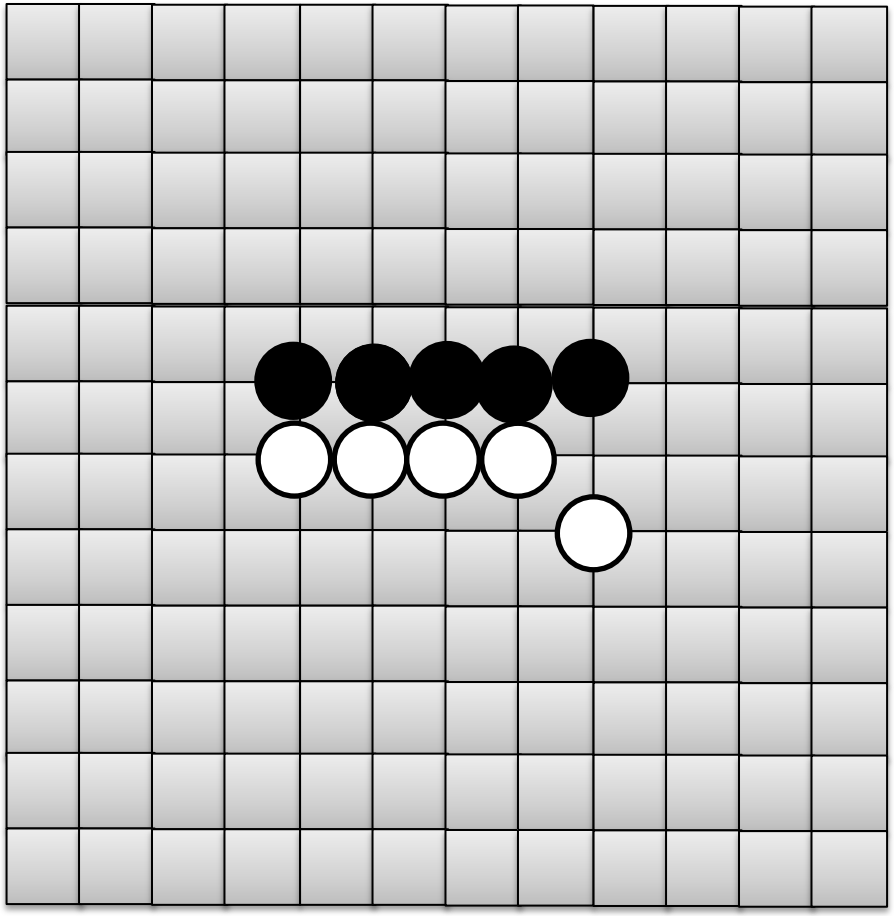
- 問題例
  - 先手の黒が5回、白が5回打っている
  - 白がちょうど勝ったところ。YES



全探索で難しい問題　演習

AtCoder

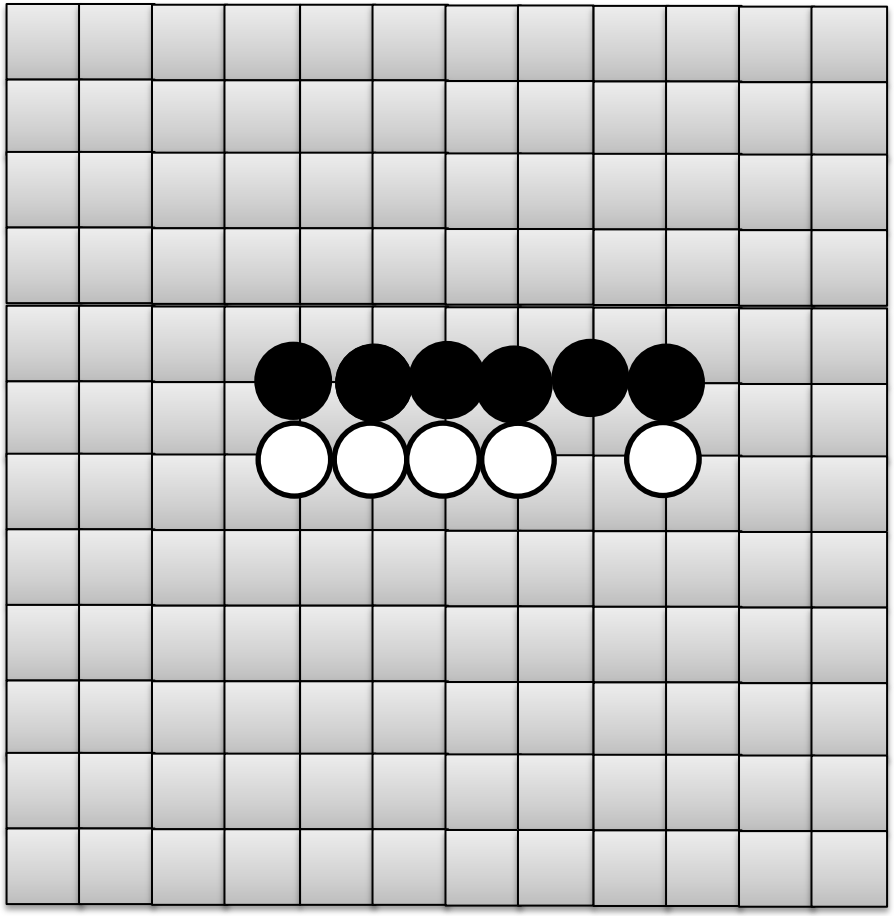
- 問題例
  - 先手の黒が5回、白が5回打っている
  - 黒が前のターンで確実に勝っているはずなのに、白が5手目を打っている
  - これもNO



全探索で難しい問題　演習

AtCoder

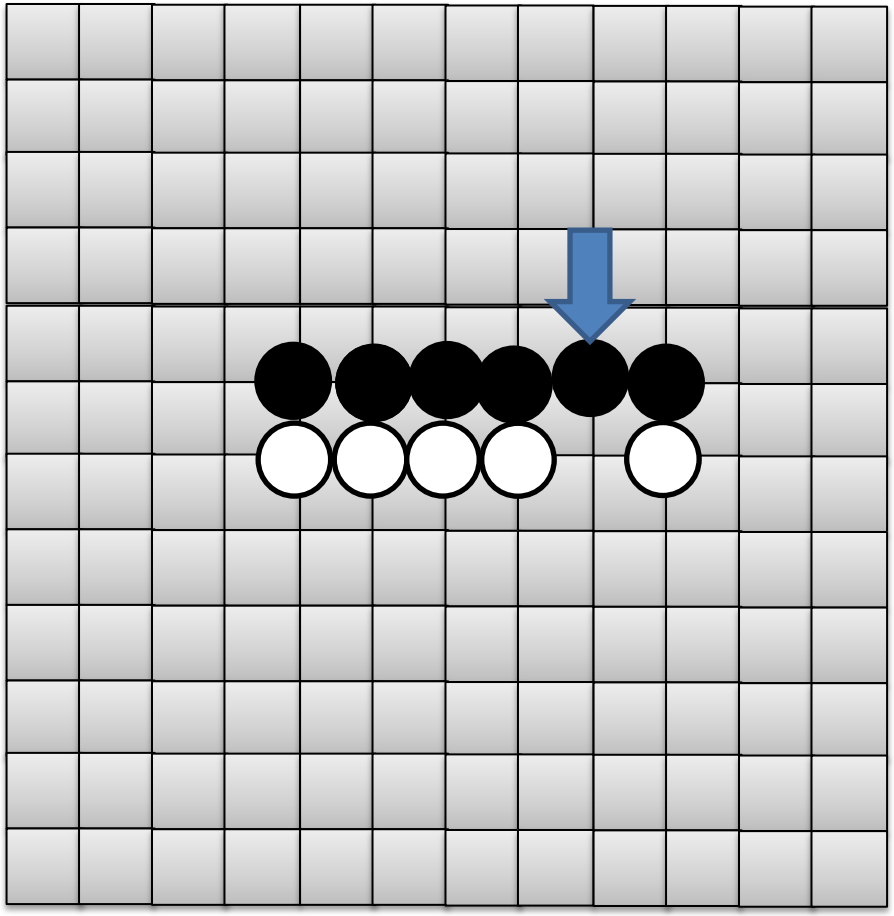
- 問題例
  - 先手の黒が6回、白が5回打っている
  - 黒がちょうど勝ったところ？その前のターンで5つ並んでいるはず？



全探索で難しい問題　演習

AtCoder

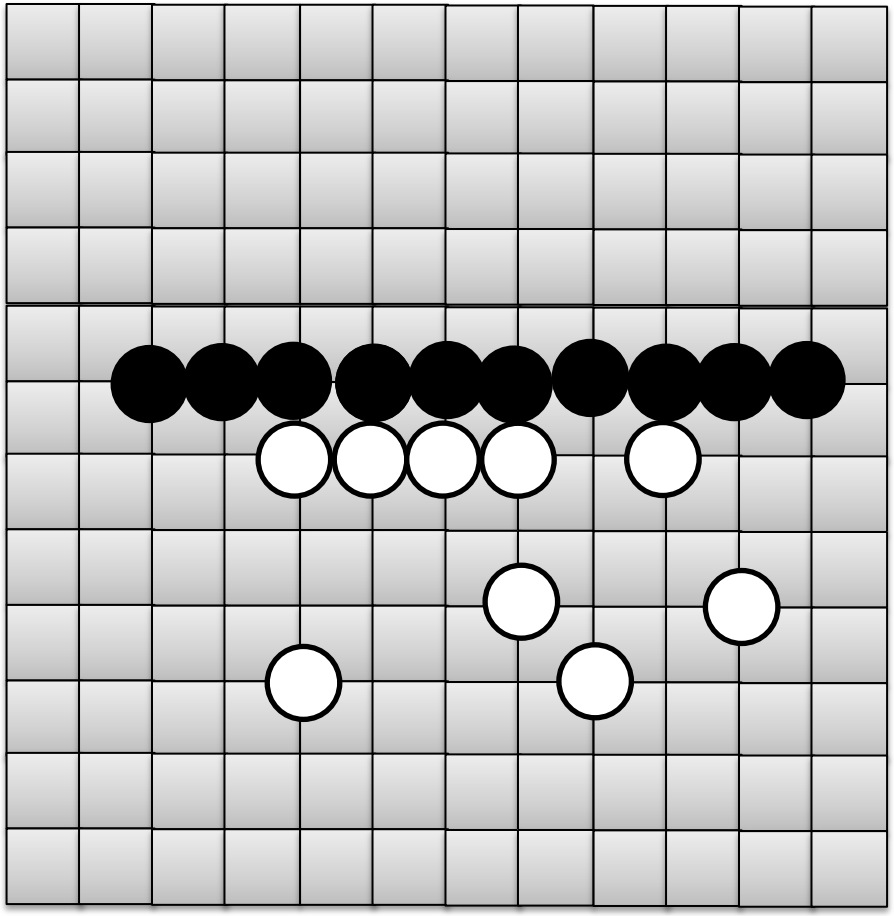
- 問題例
  - 先手の黒が6回、白が5回打っている
  - 黒がちょうど勝ったところ？その前のターンで5つ並んでいるはず？
  - ここに最後に打ったのであれば、この瞬間に勝ちになったところなのでYES



全探索で難しい問題　演習

AtCoder

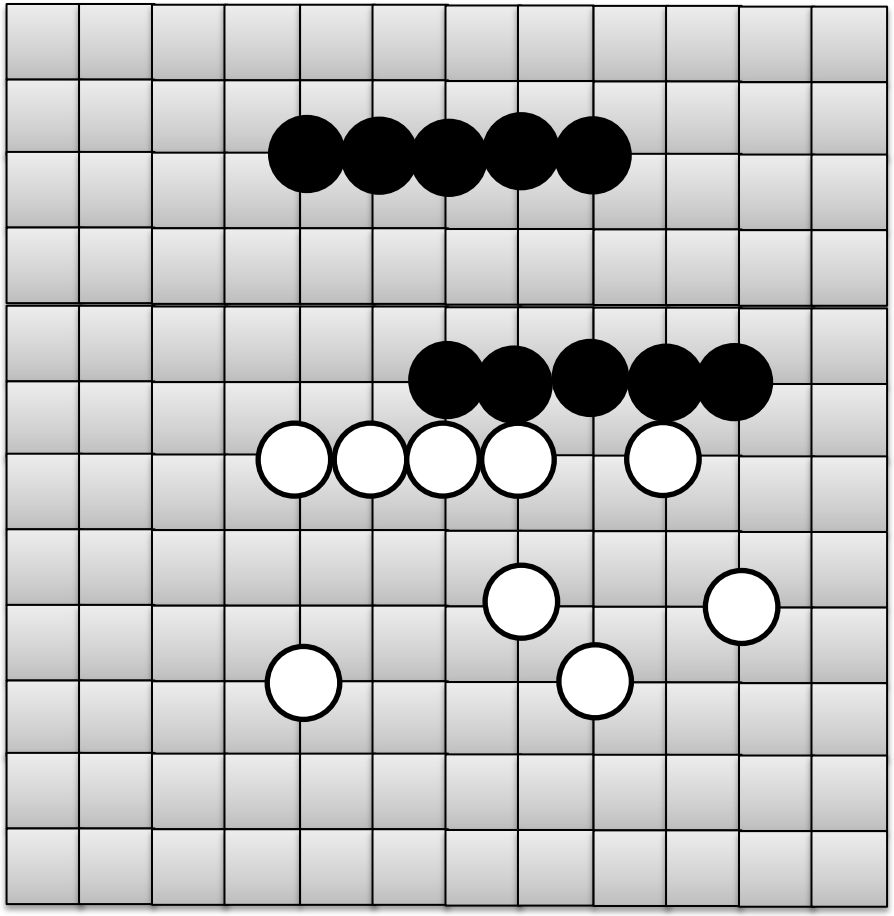
- 問題例
  - 先手の黒が10回、白が9回打っている
  - 黒がちょうど勝ったところ？その前のターンで5つ並んでいるはず？
  - どうやっても前のターンで5つ以上並んでいるためYES



全探索で難しい問題　演習

AtCoder

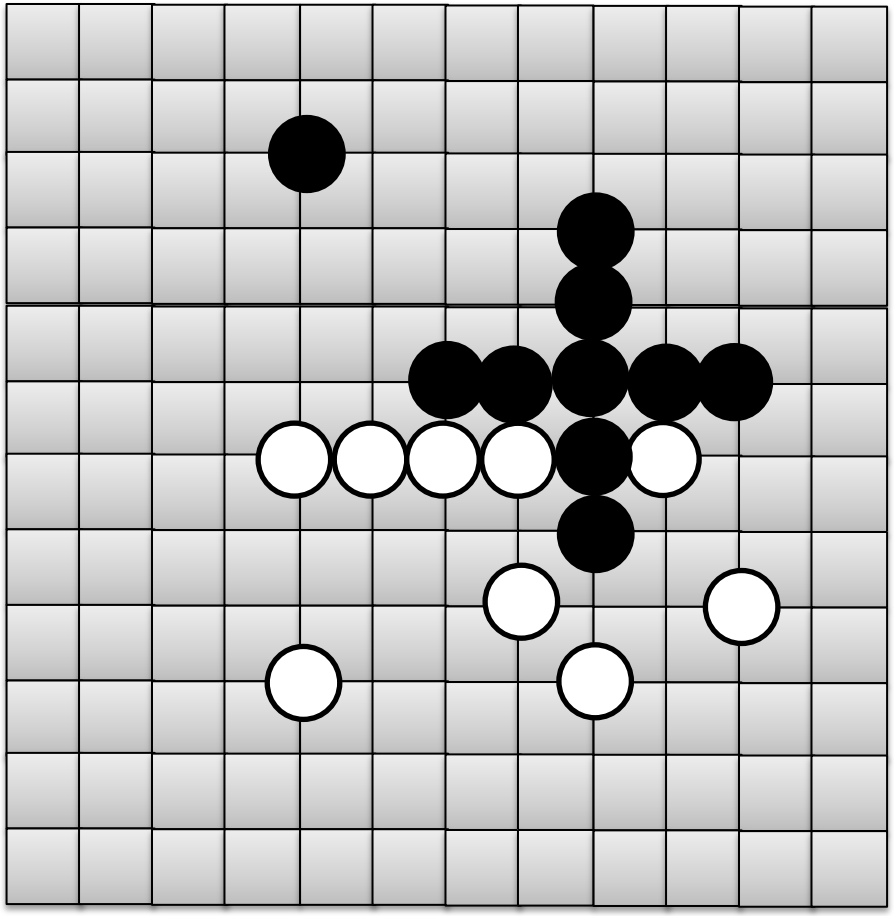
- その他のコーナー
  - 先手の黒が10回、白が9回打っている
  - 黒がちょうど勝ったところ？その前のターンで5つ並んでいるはず？
  - 2つの場所で勝ってるのはおかしい。NO



全探索で難しい問題　演習

AtCoder

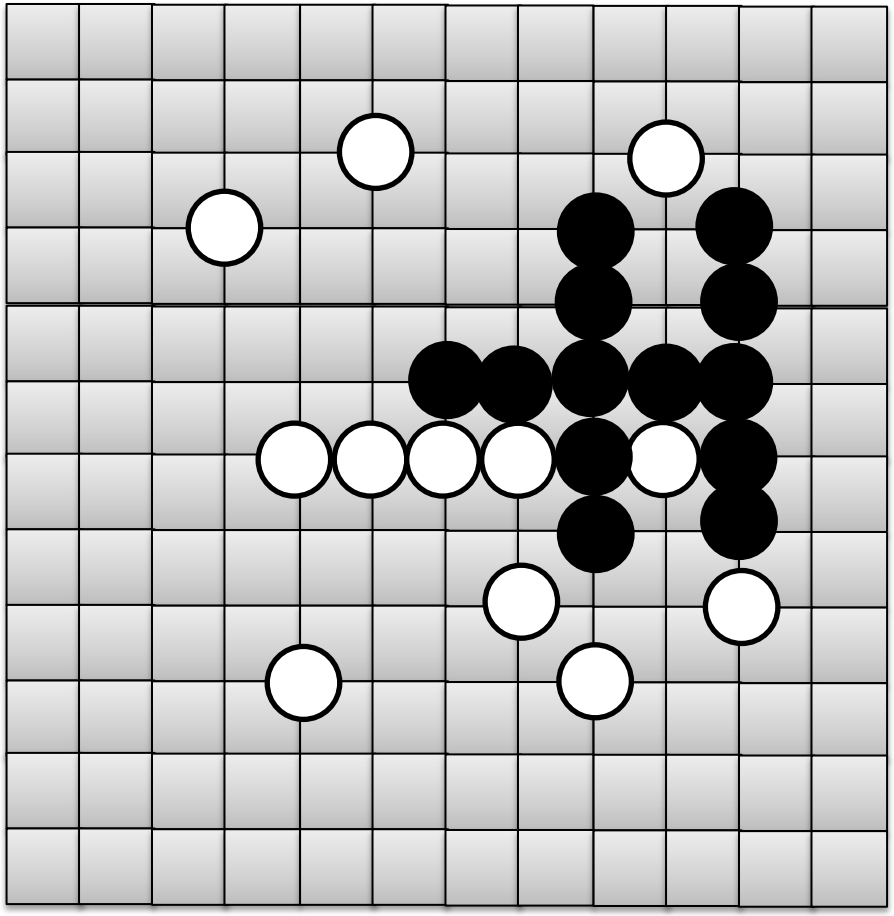
- その他のコーナー
  - 先手の黒が10回、白が9回打っている
  - 黒がちょうど勝ったところ？その前のターンで5つ並んでいるはず？
  - 2つの場所で勝ってても、中央で打てばおかしくない。YES



全探索で難しい問題　演習

AtCoder

- その他のコーナー
  - 先手の黒が13回、白が12回打っている
  - 黒がちょうど勝ったところ？その前のターンで5つ並んでいるはず？
  - これはおかしい NO



全探索で難しい問題　演習

AtCoder

- 解き終わった人は以下の問題にチャレンジ！
  - AtCoder Beginner Contest 004 D問題　マーブル
    - http://abc004.contest.atcoder.jp/tasks/abc004_4
  - KUPC2013 B問題　ライオン
    - http://kupc2013.contest.atcoder.jp/tasks/kupc2013_b
  - ほそながいところ
    - http://judge.u-aizu.ac.jp/onlinejudge/description.jsp?id=2427
      - » AtCoderじゃないですが＞＜

全探索で難しい問題　演習

AtCoder

- ヒント1
  - 全探索で解きましょう！
    - こんなの場合分けで解いてたら死にます。絶対バグります。
  - 何を全探索するのかな？
    - 暫くは、ここを考えてみましょう！
      - 次ページにヒントがあります。

全探索で難しい問題　演習

AtCoder

- ヒント2
  - 「どういう状態だったらNOになるのか考えよう！」
    - 問題がなければYESなので、NOになる条件を考える！
    - 白石と黒石の総和は解りやすいので、それ以外で！

    - 次のページにヒント3があります。考えてからめくってください。

全探索で難しい問題　演習

AtCoder

- ヒント3
  - 1ターン前がどうなっていたらNO？
    - 1ターン前→終わってない　今のターン　終わってない
      - 勝負がついていないためYES
    - 1ターン前→終わってない　今のターン　終わってる
      - 今回のターンで勝負がついた　YES
    - 1ターン前→終わってる　今のターン　終わってない
      - このパターンはあり得ない。碁石は増えるしかないため。
    - 1ターン前→終わってる　今のターン　終わってる
      - このパターンだけNO!
  - よって、「1ターン前に終わっていないパターンが存在するかどうか」が判定出来れば良いよね？
    - 次のページでヒント4になります。考えてからめくります。

全探索で難しい問題　演習

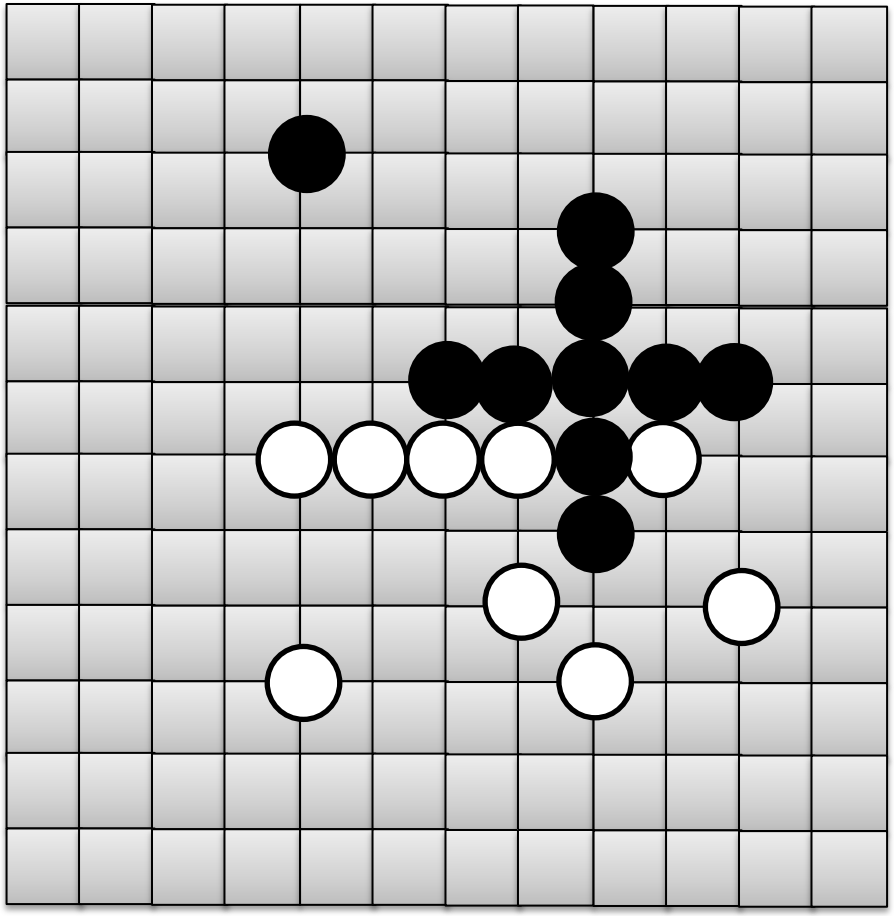
AtCoder

- ヒント4
  - 1ターン前に終了条件を満たしているか調べるのはどうすればいい？
    - 1ターン前に置いた可能性のある石を全探索！
      - 直前に置いた石の色は、白石と黒石の個数から判断出来る
    - これによって、1ターン前の全局面を列挙できる！
    - これで、1つでも終わっていない状態があればYES、そうでなければNO

全探索で難しい問題　演習

AtCoder

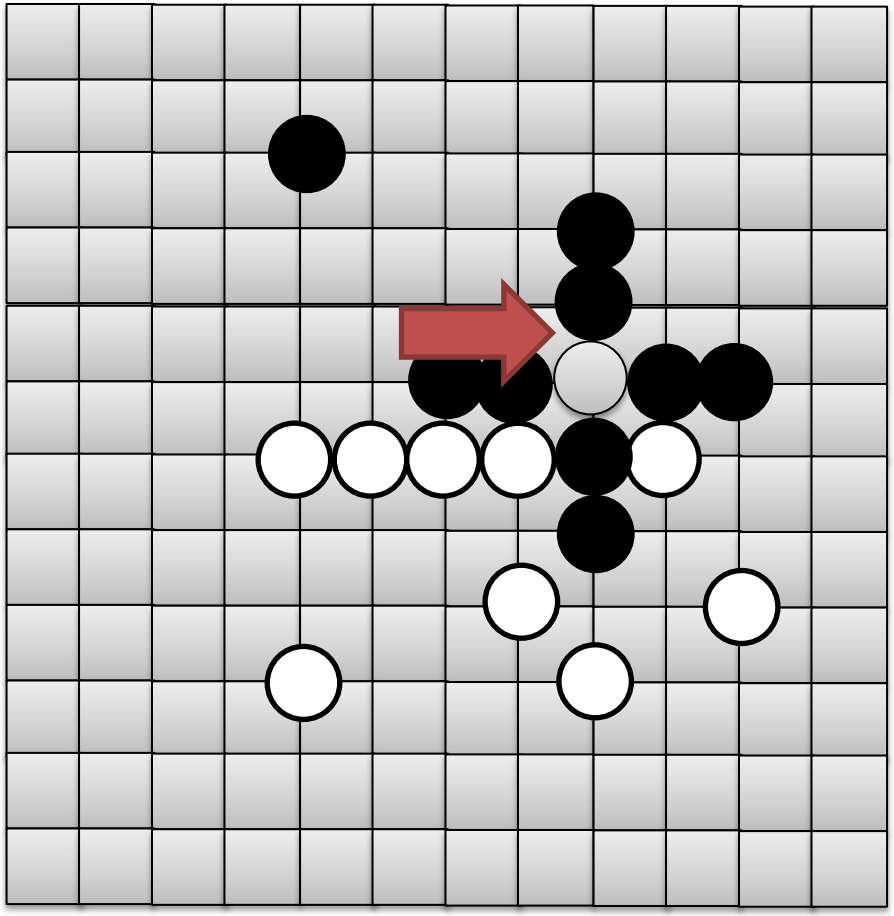
- 黒が最終手
  - この状態がなぜOKか？



全探索で難しい問題　演習

AtCoder

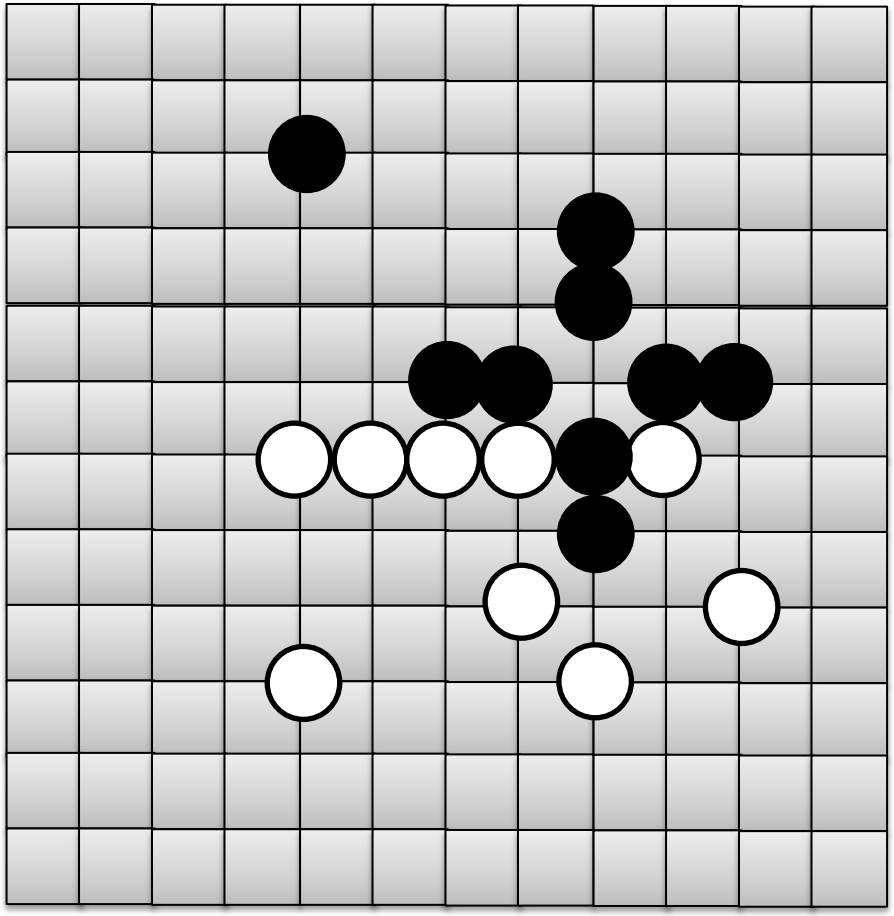
- 黒が最終手
  - この状態がなぜOKか？
  - 最後に置いた石が、この石である可能性がある！



全探索で難しい問題　演習

AtCoder

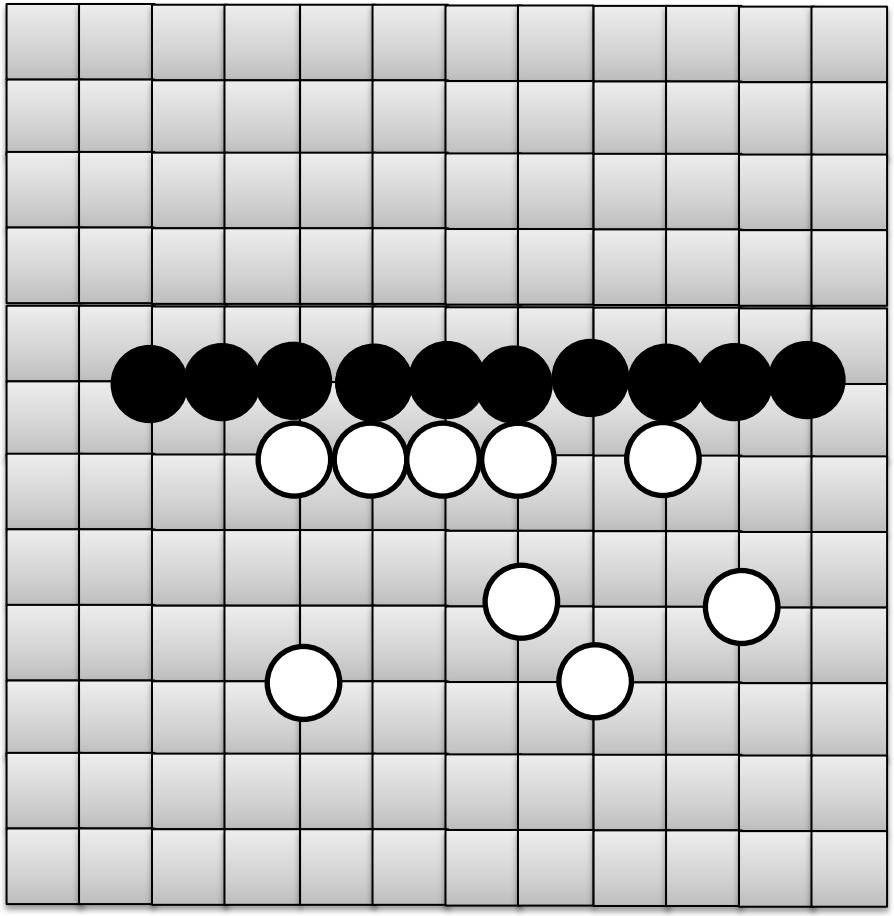
- 黒が最終手
  - この状態がなぜOKか？
  - 最後に置いた石が、この石である可能性がある！
  - この石を取り除くと、どちらも5つ揃えられていない。
  - よって、ここで黒石を置くのは問題のない動作だと言える



全探索で難しい問題　演習

AtCoder

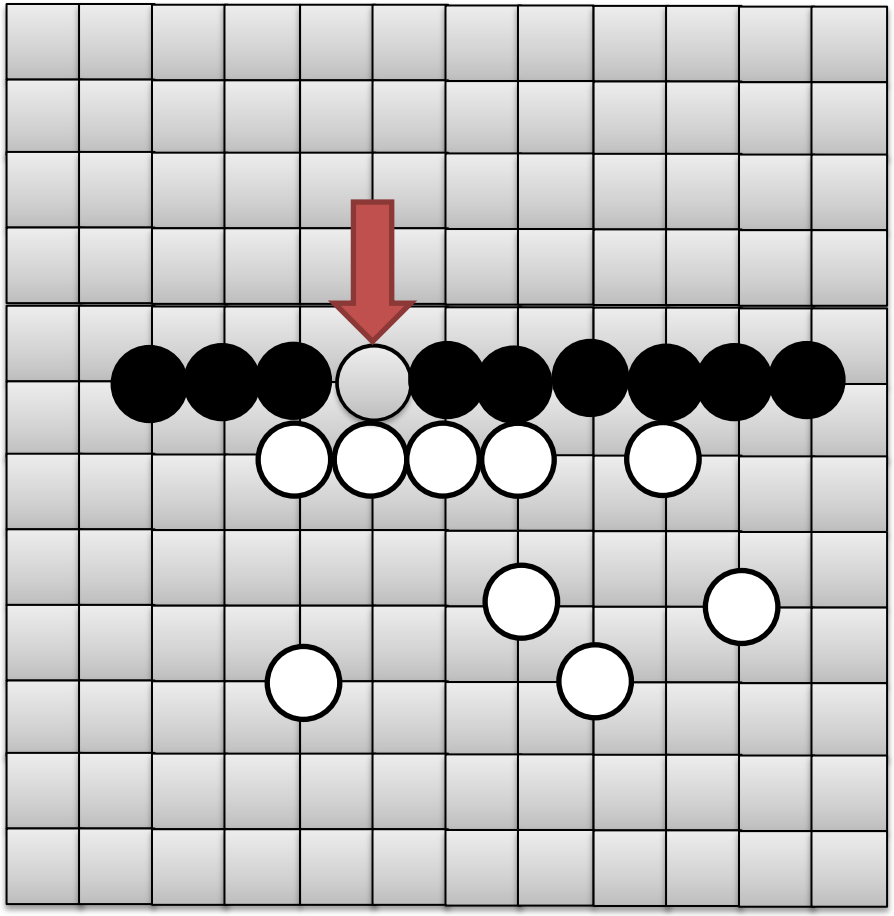
- 解説
  - 黒が最終手
  - この状態がなぜNGか？



全探索で難しい問題　演習

AtCoder

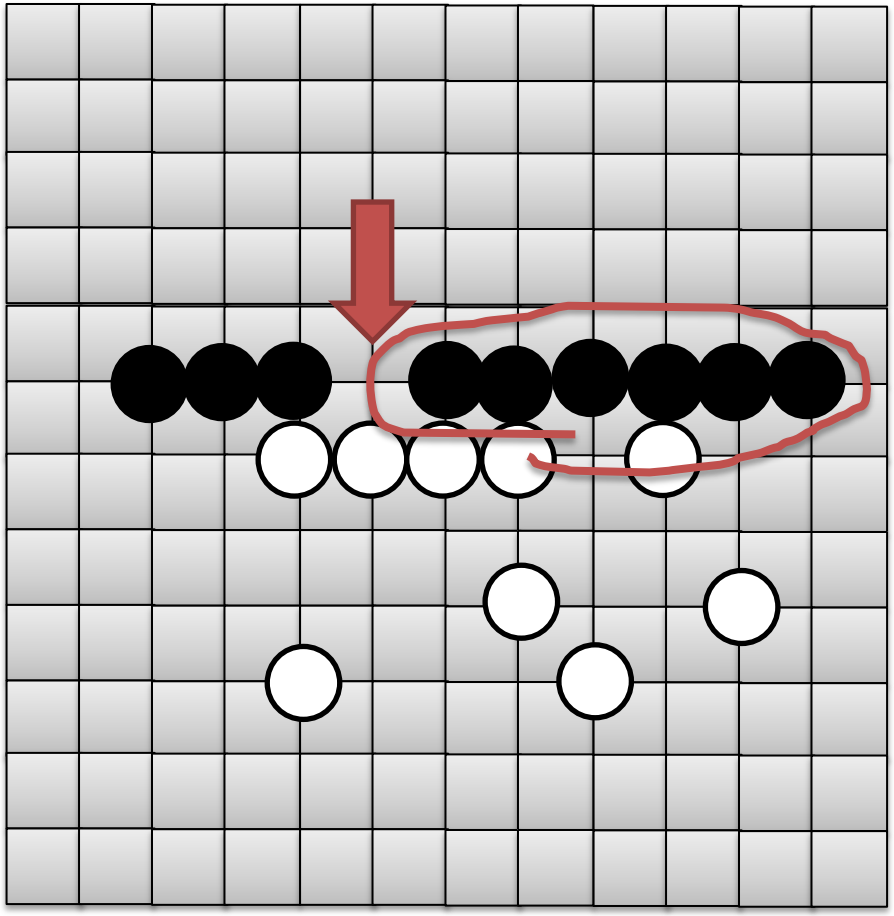
- 解説
  - 黒が最終手
  - この状態がなぜNGか？
  - 例えば、最終手がこの石だとする



全探索で難しい問題　演習

AtCoder

- 解説
  - 黒が最終手
  - この状態がなぜNGか？
  - 例えば、最終手がこの石だとする
  - これを取り除いても、既に黒が勝っている！
    - ということは、ゲームが終わっているので、黒が打つのはおかしい



全探索で難しい問題　演習

AtCoder

- 以下のような手順で解くことが出来る
  - 黒石と白石を数える。黒白どちらが最終手か求める
    - この時点で矛盾がある場合はすぐにNOを出力して終了
  - 最終手の可能性のある石を、1個取り除いた盤面を全列挙する
    - 各盤面に対して、「勝利条件を満たしているか」を、盤面の各石から、全8方向に、5つ同じ石が並んでいるかの全探索を行う
    - もし勝利条件を満たしていなければ、正常な状態なので、YESを出力して終了
  - 条件を満たす盤面が存在しなかった場合は、不正な状態なので、NOと出力する

全探索で難しい問題　演習

AtCoder

- ソースコード
  - http://arc012.contest.atcoder.jp/submissions/157045

```
int N = 19;
int[][] board;
void run()
{
    Scanner cin = new Scanner(System.in);
    board = new int[N][N];

    //入力。String型は扱いにくいのでint型の2次元配列へ
    //ついでにoとxの数を数える
    int ocount = 0;
    int xcount = 0;
    for(int i=0;i<N;i++){
        String st = cin.next();
        for(int j=0;j<N;j++){
            if(st.charAt(j) == 'o'){
                board[i][j] = 1;
                ocount++;
            }
            if(st.charAt(j) == 'x'){
                board[i][j] = 2;
                xcount++;
            }
        }
    }
```

全探索で難しい問題　演習

AtCoder

- ソースコード

```
//どちらが最後の手であったかを調べる
int prev = 0;
if(ocount - 1 == xcount) prev = 1;
if(ocount == xcount) prev = 2;

//矛盾がある場合はNOを出力
if(prev == 0){
    System.out.println("NO");
    return;
}

//!!コーナーケース！まだ何も置かれていないときはYES
if(ocount==0){
    System.out.println("YES");
    return;
}
```

全探索で難しい問題　演習

AtCoder

- ソースコード

```
    //最終手候補となる石を列挙し、それぞれの直前の局面を調べる
    for(int i=0;i<N;i++){
        for(int j=0;j<N;j++){
            if(board[i][j] == prev){
                board[i][j] = 0;
                if(checkboard()){
                    System.out.println("YES");
                    return;
                }
                board[i][j] = prev;
            }
        }
    }
    //条件に合う盤面が見つからなかったらNOを出力
    System.out.println("NO");
}
```

全探索で難しい問題　演習

AtCoder

- ソースコード

```
boolean checkboard(){
    //8方向に移動する時の変数
    int[] vy = new int[]{1,1,1,0,0,-1,-1,-1};
    int[] vx = new int[]{1,0,-1,1,-1,1,0,-1};
    //全ての盤面から、石の置いてある場所を探す
    for(int i=0;i<N;i++){
        for(int j=0;j<N;j++){
            //石を見つけたら、その石を起点とした5連結の石を調べる
            if(board[i][j] != 0){
                int now = board[i][j];
                //8方向に対して調べる
                for(int k=0;k<8;k++){
```

全探索で難しい問題　演習

AtCoder

- ソースコード

```
                boolean flag = true;
                for(int l=0;l<5;l++){
                    int y = i + vy[k] * l;
                    int x = j + vx[k] * l;
                    if(!ok(y,x) || board[y][x] != now){
                        flag = false;
                        break;
                    }
                }
                //もし5連続の石を見つけたら離脱
                if(flag) return false;
            }
        }
    }
    return true;
}

boolean ok(int y, int x){
    return y>=0 && x>=0 && y<N && x<N;
}
```

全探索で難しい問題　まとめ

AtCoder

- 全探索には、色々なパターンがある
  - 「問題文にしろと書いてある全探索」
  - 見えやすい全探索
  - 見えにくい全探索

- 素直な全探索が通用しないなら、何か別のものを全探索出来ないか考える習慣をつけよう！
  - 例えば以下のようなパターン
    - 途中の状態を無視して、最終状態だけ考える
    - 「どう使うか」は置いといて「使う」「使わない」だけ
    - 「実際の値」は無視して、「差」や「和」だけに注目する

動的計画法の難しい問題

動的計画法の難しい問題

AtCoder

- 動的計画法は、全探索よりも大変！
  - まぁ解りきったことだとは思いますが・・・。

  - 特定の問題の「全探索」をするにも、様々なものに対する「全探索」があったため、様々な全探索を考えなければいけなかった。
  - 動的計画法は、さらに、その全探索を「まとめる」作業が必要となる
    - これは、同じものを全探索する場合も、「どういう順番で」「どういう情報を持って」全探索するかによって、纏められるかどうかが変わってくる！
  - 「何を」「どうやって」探索するかの両方を考えないと、動的計画法は適切に使えない！

動的計画法の難しい問題

AtCoder

- 動的計画法の難しい問題はどう解くの？

- 今回の講座では説明しきれません！！！
  - 時間的に絶対に無理です！
  - 5回全部を動的計画法で講義することも可能なくらい、奥が深いです。

- また別の機会に動的計画法講座があれば、その時またよろしくお願いします。
  - それまでに教え方も考えておきます。

動的計画法の難しい問題　演習

AtCoder

- それでも難しい問題を解きたい人へ
  - » 解いて体調を崩しても、責任は取りません。本当に難しいです。
  - ARC020 D問題　お菓子の国の旅行
    - http://arc020.contest.atcoder.jp/tasks/arc020_4
  - ARC005 D問題　連射王高橋君
    - http://arc005.contest.atcoder.jp/tasks/arc005_4
  - ARC001 D問題　レースゲーム
    - http://arc001.contest.atcoder.jp/tasks/arc001_4
  - 天下一プログラマーコンテスト2012予選C D問題　ゆうびんやさんのお花畑
    - http://tenka1-2012-qualc.contest.atcoder.jp/tasks/tenka1_2012_12
  - 天下一プログラマーコンテスト2012予選A D問題 アリの巣
    - http://tenka1-2012-quala.contest.atcoder.jp/tasks/tenka1_2012_qualA_4
  - Typical DP Contest P問題 うなぎ
    - http://tdpc.contest.atcoder.jp/tasks/tdpc_eel

動的計画法の難しい問題　演習

AtCoder

- ちょっと難しい、くらいの問題を解きたい人へ
  - TDPCを上から順番に解いてみることをお勧めします
    - http://tdpc.contest.atcoder.jp/
    - 典型DPコンテスト、という名前ですが、結構難しいです

本日のまとめ

本日のまとめ

AtCoder

- 難しい問題に触れた！
  - 解けるわけではないが、難しい問題がどうして難しいかが分かった！
- 世界の広さを知った！
  - 難しい問題は本当に難しいです。
- 競技プログラミングの楽しさが分かった！
  - わかってくれていると嬉しいです。
    - 解けないと辛いけど、解けると楽しいよ！

これまでの講義のまとめ！

AtCoder

- 今回触れたのは、これくらいの範囲です！

Forのみの全列挙
ダイクストラ
半分全列挙
Euler Tour
深さ優先探索
ワーシャルフロイド
凸包
Nim
幅優先探索
Union-Find
2-SAT
Grundy数
貪欲法
二分探索
Fenwick Tree
最小費用流
フロンティア法
ソート
動的計画法
座標圧縮
最大流
高速フーリエ変換
二部マッチング
強連結成分分解
優先度付キュー
しゃくとり法
行列累乗
Link-Cut Tree
平衡二分木
ダブリング
Segment Tree

これまでの講義のまとめ！

AtCoder

- 今回触れたのは、これくらいの範囲です！
  - さらに、標準ライブラリについてるものを追加

Forのみの全列挙　ダイクストラ　半分全列挙　Euler Tour

深さ優先探索　ワーシャルフロイド　凸包　Nim

幅優先探索　Union-Find　2-SAT　Grundy数

貪欲法　二分探索　Fenwick Tree　最小費用流　フロンティア法

ソート　動的計画法　座標圧縮　最大流　高速フーリエ変換

二部マッチング　強連結成分分解

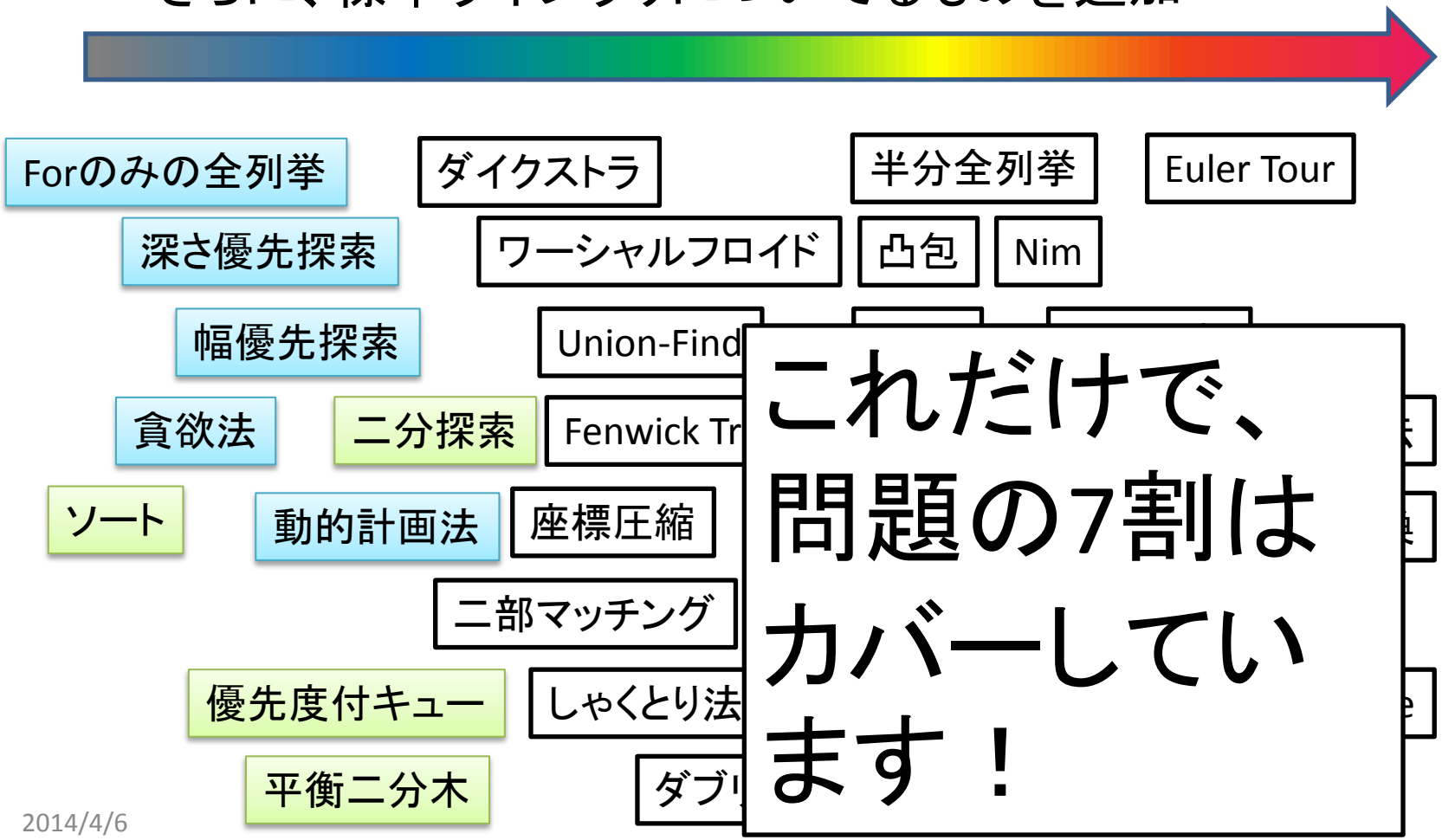
優先度付キュー　しゃくとり法　行列累乗　Link-Cut Tree

平衡二分木　ダブリング　Segment Tree

これまでの講義のまとめ！

AtCoder

- 今回触れたのは、これくらいの範囲です！
  - さらに、標準ライブラリについてるものを追加



これまでの講義のまとめ！

AtCoder

- 今回触れたのは、これくらいの範囲です！
  - さらに、標準ライブラリについてるものを追加

Forのみの全列挙
ダイクストラ
半分全列挙
Euler Tour
深さ優先探索
ワーシャルフロイド
凸包
Nim
幅優先探索
Union-Find
貪欲法
二分探索
Fenwick Tree
ソート
動的計画法
座標圧縮
二部マッチング
優先度付キュー
しゃくとり法
平衡二分木
ダブリング

ここまで身につければ、知識は殆ど問題なし！

これまでの講義のまとめ

AtCoder

- 今回の講義に全て出た人は、既に「コンテストを楽しむのに十分な知識」を身に着けています！
  - ここからは、自分で定期的に問題を解いていきましょう！

- AtCoderでは、2つのコンテストを、毎週土曜午後9時から、交互に開催しています。
  - AtCoder Beginner Contest
    - 初心者・初級者向けコンテスト。必ず終了後に解説配信があります。
  - AtCoder Regular Contest
    - 初級者～上級者向けコンテスト。歯応えのある問題が解けます。

これまでの講義のまとめ

AtCoder

- 今回の講座で覚えて言って欲しいもの
  - 概念が分かったところで、なかなか実装は出来ない。
  - でも概念が解ると、解説されればイメージは出来る。
  - 解説を聞いても、実装するのは結構難しい。
  - 概念だけ知っていても、どう解けば良いか解らない問題はたくさん存在する。

- 要するに、「知識」「実装力」「発想力」全てが必要！
  - これらをバランス良く伸ばしていきましょう！
    - 頭でっかちにならない！実装ばっかりに偏らない！
    - すぐに思いつかない問題を嫌って避けたりしない！

これまでの講義のまとめ

AtCoder

- 今回の講座で不明点があれば、下記の連絡先にお願いします。
  - Twitter: @chokudai
  - メール: chokudai@gmail.com
    - どちらも反応するとは限りません。
- 5週間ありがとうございました！