

atCoder 過去問練習帳

atCoder の過去問を解いていくろぐ。(atCoder Problems のユーザーページ^{*1})
わからんまま保留しているもののリスト→わからんもののリスト

2019-08

8/10 ABC137 初参加 → 灰色、パフォーマンス : 916

8/18,19 ABC138E [LUT を逆むきサーチで]

本番で 50 分ほどあったのに解けなかったやつ。テーブルを前もってつくることで高速化。アルファベットの個数 $\ll t$ の長さ \ll 答えの大きさ という大小関係をとらえる感覚が鈍かった。そこに気づけば、かなり重い前処理でも AC, のちにしーやん先生にならって逆にみていくのがミソの LUT 版を投稿: 7041951^{*2}

8/21 ABC054C [DFS]

N 小さかったので、すなおいに DFS で。next permutation 使えるなどおもったけど、未実施

8/21 ABC040D [UnionFind]

サイズを数える機能つき Union Find をその場で実装しちゃえとおもって、ソラで書いて無限にバグった。反省: 705691^{*3}

8/21 ABC131E [グラフ]

探索によらない、グラフの性質を考える問題。スター接続のときが、最も「距離 2 ペア」の数が多く、そのとき $\frac{(n-1)(n-2)}{2}$ 個ある、などから考えていく。

解説読むまでわからず、読んで「わかった」つもりでも、実装段階で数回間違った。要反省 : 7059102^{*4}

8/22 ABC049D [UnionFind + α]

ABC040D (二個まえにやった) の反省を生かして、手元ライブラリの UnionFind を使う。これは、配列を上手につかって、連結成分のサイズもとれるやつなので、これを使えばすごくいいではないか、道路で連結だった場合のみ Unite するようにしよう! と思ってよろこんで書いたやつは WA^{*5}

解説などをみつつ、あとからちゃんと数えるように作り直して AC になったが、上の版でだめになる反例をまだ思いつけていない。あとで→ わからんもののリスト へ : 7066761^{*6}

*1 <https://kenkoooo.com/atcoder/?user=unnohideyuki&kind=user>

*2 <https://atcoder.jp/contests/abc138/submissions/7041951>

*3 <https://atcoder.jp/contests/abc040/submissions/7057691>

*4 <https://atcoder.jp/contests/abc131/submissions/7059102>

*5 <https://atcoder.jp/contests/abc049/submissions/7066417>

*6 <https://atcoder.jp/contests/abc049/submissions/7066761>

8/23 ABC128C

解説によるとガロア体上の連立方程式と思えたいなことが書いてあった気がする。今回は、N がちっさいので全探索した。Hacker's delight に population count あったなあというので、ビット列を使った高速化：7089408*7

8/23 ABC129C

みんな大好きフィボナッチって感じで、書くだけだったけど、「入力例3」がなかったら、バグに気づかずWAしてたかも。壊れているのを特別に -1 とかで表現するか、実際投稿したやつのように二つの配列に分けるか悩んで後者にしたんだけど、broken[a] = true に設定するだけじゃなくて、stairs[a] = 0 にしないといけない（んだが、問題になるのは a = 1 のときのみ）てのにはまった：7091512*8

8/24 AGC019B

自力では解けなかった。解説参照：7097088*9

8/24 ABC067D

着想は自力で正解だったのだけど、牛刀を使おうとして失敗。AtCoder Problems で Moderate (50%) サジェストされるやつなので、二問にいかいくらいは自力で解きたい。

自力で解けそうだったんだけど、RE に悩まされる。原因は、Warshall-Floyed のための配列が大きくなりすぎたことのようにだった。（そこにたどり着くまでに30分ロス）任意の二点間の距離を求める必要はないので、Warshall-Floyed に頼る必要はなかった。

あとで実装しなおす。→した：7102655*10

8/24 第1回日本最強プログラマー学生選手権 予選 3回目の参加 → 茶色、パフォーマンス：927

8/27 CODE FESTIVAL 2016 qual C

わかったと思ってスラスラ書いたつもりが WA*11。簡単かなってときにも、矛盾するケースをちゃんと書き出すとか、丁寧にやるべきだったかな：7157750*12

8/27 Code Festival 2016 Grand Final B

わからず、解説みた。中学～高校数学って感じの問題だった。解説に書かれていなくて必要な知識は、以下の2つくらいだったかな：7160019*13

*7 <https://atcoder.jp/contests/abc128/submissions/7089408>

*8 <https://atcoder.jp/contests/abc129/submissions/7091512>

*9 <https://atcoder.jp/contests/agc019/submissions/7097088>

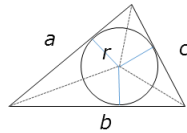
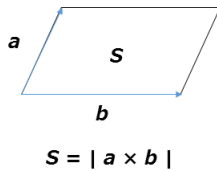
*10 <https://atcoder.jp/contests/abc067/submissions/7102655>

*11 <https://atcoder.jp/contests/code-festival-2016-qualc/submissions/7150641>

*12 <https://atcoder.jp/contests/code-festival-2016-qualc/submissions/7157750>

*13 <https://atcoder.jp/contests/cf16-exhibition-final-open/submissions/7160019>

2つのベクトルが張る平行四辺形の面積は
外積の大きさ



$$S = r(a + b + c) / 2$$

8/27 ABC021B

「AtCoder の問題を問題を分類しました」の 5.5 グラフより。7160319^{*14}

8/27 Nikkei 2019 予選 C

「ゼロサムゲームだ！」って感じでふつうにとけた 7160711^{*15}

ゼロサムちがう気もする。(ちがう)

8/27 AGC033A

最初の着想でそのまま解けた。7164832^{*16}

8/27 ABC051D

Difficult(20%) だけど解けそう!と思ったものの WA。くっ、やはり。

ワーシャルフロイドの過程で、距離を書き換えるケース(経路圧縮?)でついでに消していけばよからうと思ったものの、うまくいかなかった。なんでうまくいかないのか、まだ理解してない → わからんものリストへ。

ひとまず、解説通り、まずは最短経路を求めておいてから、最短経路の一部になっているかどうかを調べる方法を採用: 7165628^{*17}

8/27 ABC054D [DP]

Difficult(20%) で推薦されたやつ。わかんなかったのが、解説見て「うむむ、なるほど…?」ってなってる。あとで再挑戦。

あとで書こうとしてみたけど、結局わかんなくなつて写経。あとでもいっかい考える: 7168802^{*18}

つぎの日に、自分で書けるかやってみた。ちょっと間違い。そうか、0 個まで買った状態で次に買う商品は 0 番なので、商品番号はゼロ始まりにしなきゃ。だった。7183167^{*19}

^{*14} <https://atcoder.jp/contests/abc021/submissions/7160319>

^{*15} <https://atcoder.jp/contests/nikkei2019-qual/submissions/7160711>

^{*16} <https://atcoder.jp/contests/agc033/submissions/7164832>

^{*17} <https://atcoder.jp/contests/abc051/submissions/7165628>

^{*18} <https://atcoder.jp/contests/abc054/submissions/7168802>

^{*19} <https://atcoder.jp/contests/abc054/submissions/7183167>

8/30 ABC021D [Warshall-Floyd]

まず、Warshall-floyd を使うときの初期化で、自分へもどってくるのをゼロに初期化すべきだったところでバグ。そっちは自分で気づいたんだけど、Inf にあたる数が小さすぎて WA を連発してしまった。これは、初期化にも、最後の Min を求めるときにも使う：7208397^{*20}

8/30 ABC070D グラフ

おおまかな方針はすぐわかったのだけど、隣接行列でやろうとして失敗。そこから心を入れ替えて一応自力で解けたけど、時間かかりすぎ。はじめから、この実装にたどりつきたかった：7211329^{*21}

8/31 AGC032B グラフ

難しかった。補グラフを考えるという発想自体がなかった。けっか、できたプログラムはすごく簡潔：7224022^{*22}

8/31 EDPCA [DP]

Educational DP Contest A を解いてみた。普通にとけたけど、これは典型的な DP の書き方になってなかったな：7233600^{*23}

8/31 EDPCB [DP]

つづけて二問め。こんどは、典型的な「配る DP」風にしたつもりだったけど、Inf の大きさが甘くて WA。これ、よくやる。要注意。あと、dp 配列を $n+k$ で余分にとっておけば、ループ条件緩和できる。どっちでもいいような気もするけど：7233978^{*24}

8/31 EDPCC [DP]

なるほど 7234407^{*25}

でも、↓こういうループにした方がよかったかな

```
// ループ
for (int i = 0; i < N; ++i) {
    for (int j = 0; j < 3; ++j) {
        for (int k = 0; k < 3; ++k) {
            if (j == k) continue;
            chmax(dp[i + 1][k], dp[i][j] + a[i][k]);
        }
    }
}
```

^{*20} <https://atcoder.jp/contests/abc012/submissions/7208397>

^{*21} <https://atcoder.jp/contests/abc070/submissions/7211329>

^{*22} <https://atcoder.jp/contests/abc032/submissions/7224022>

^{*23} <https://atcoder.jp/contests/dp/submissions/7233600>

^{*24} <https://atcoder.jp/contests/dp/submissions/7233978>

^{*25} <https://atcoder.jp/contests/dp/submissions/7234407>

```
}  
}
```

2019-09

9/3 soundhound2018_summer_qual_c [さんすう]

かんたん、とかいいつつ、例2に助けられた(うかつ): 7318110^{*26}

9/3 ABC139E [グラフ]

本番では、ひとつだけ TLE のまま一時間くらい溶かしてしまったやつ。グラフの最長経路の長さを求める問題に帰着させる。あと、循環の検出。なかなか、ちゃんと書けなかった。7318341^{*27}

9/3 ARC073B [累積和、全探索]

あまり考えずに解説みてしまった。ナップサックだからといって DP ではない。条件を用いれば全探索可能になる。高速化のために累積和。そういえば、いちど、w の int あふれで WA だった。きをつけよう: 7322752^{*28}

9/3 ABC135D - Digits Parade [DP]

なるほど、DP って強力なんですね。解説みて、上の桁(左端)からやってくるのかというのが意外だったけど、なるほどきれいだ。7339526^{*29}

9/4 ARC059B

寝る前になんかやるかとおもて、Easy でレコメンドされたやつ。Easy っるのがヒントになっちゃったかもしれん。7342360^{*30}

9/5 ABC019D - 高橋くんと木の直径 [グラフ]

限られた質問回数で木の直径(二点間の距離の最大値)を求める: 7345767^{*31}

9/8 ABC140D - Face Produces Unhappiness

N や K の大きさから、探索めいたことをやっていたはダメだなあとは思いつつ、ちゃんと腰をすえて考えられなかった。(つぎの E を考えてみたり、F を考えてみたりと注意を分散させてしまった) 7420982^{*32}

^{*26} <https://atcoder.jp/contests/soundhound2018-summer-qual/submissions/7318110>

^{*27} <https://atcoder.jp/contests/abc139/submissions/7318341>

^{*28} <https://atcoder.jp/contests/arc073/submissions/7322752>

^{*29} <https://atcoder.jp/contests/abc135/submissions/7339526>

^{*30} <https://atcoder.jp/contests/arc059/submissions/7342360>

^{*31} <https://atcoder.jp/contests/abc019/submissions/7345767>

^{*32} <https://atcoder.jp/contests/abc140/submissions/7420982>