2 Day Special Workshops for Competitive Programming

Day II

Yongwhan Lim

9am CT, Sunday, April 9, 2023

Yongwhan Lim

Education

Stanford University

Part-time Jobs

Full-time Job

Google Research

TWO SIGMA

Workshops

Stanford ENGINEERING | Stanford Computer Forum

Coach/Judge

icpc International Collegiate Programming Contest

icpc.foundation

https://www.yongwhan.io

Yongwhan Lim

- Currently:
    - **CEO** (Co-Founder) in a Stealth Mode Startup;
    - **Co-Founder** in Christian and Grace Consulting;
    - ICPC **Internship Manager**;
    - ICPC **North America Leadership** Team;
    - Columbia ICPC **Head Coach**;
    - ICPC **Judge** for NAQ and Regionals;
    - **Lecturer** at MIT;
    - **Adjunct** (Associate in CS) at Columbia;

https://www.yongwhan.io

Overview

9am - 10:30am	**Dynamic Programming Part I**
10:30am - 12pm	**Dynamic Programming Part II**
12pm - 1pm	**Lunch**
1pm - 5pm	**Practice Contest**
	https://codeforces.com/gym/437880 (here)
5pm	**Debrief**

DP Part I

Dynamic Programming (DP)

- "DP is a general technique for solving optimization, search, and counting problems that can be decomposed into subproblems. You should consider using DP whenever you have to make choices to arrive at the solution, specifically, when the solution relates to subproblems."

DP: Question #1 out of 2

- Alice and Bob take turns playing a game, with Alice starting first. Initially, there is a number N on the chalkboard. On each player's turn, that player makes a move consisting of:
  - Choosing any proper divisor x of N.
  - Replacing the number N on the chalkboard with N - x.
- Also, if a player cannot make a move, they lose the game.
- Return true if and only if Alice wins the game, assuming both players play optimally.

DP: Answer #1 out of 2

- Time: $O(N^2)$
- Space: $O(N)$

```
const int mx=1007;
bool divisorGame(int N) {
    vector<bool> dp(mx,false);
    for (int n=2; n<mx; n++) {
        bool ok=false;
        for (int x=1; x<n; x++)
            if(n%x==0&&!dp[n-x])
                ok=true;
        dp[n]=ok;
    }
    return dp[N];
}
```

DP: Question #2 out of 2

- Given two strings s and t, return the length of their longest common subsequence.
- A subsequence of a string is a new string generated from the original string with some characters (can be none) deleted without changing the relative order of the remaining characters. (e.g., "ace" is a subsequence of "abcde" while "aec" is not).
- A common subsequence of two strings is a subsequence that is common to both strings. If there is no common subsequence, return 0.

DP: Answer #2 out of 2

- Time: O(nm)
- Space: O(nm)

```
int longestCommonSubsequence(string s, string t) {
    int n=s.size(), m=t.size();
    vector<vector<int>> dp(n+1, vector<int>(m+1,0));
    for (int i=0; i<n; i++)
        for (int j=0; j<m; j++)
            if(s[i]==t[j])
                dp[i+1][j+1]=dp[i][j]+1;
            else
                dp[i+1][j+1]=max(dp[i+1][j],dp[i][j+1]);
    return dp[n][m];
}
```

Lecture Exercise #1: Problem

- https://codeforces.com/problemset/problem/1470/A

Lecture Exercise #1: Solution

- https://codeforces.com/contest/1470/submission/121068197

Lecture Exercise #2: Problem

- https://codeforces.com/problemset/problem/858/C

Lecture Exercise #2: Solution

- https://codeforces.com/contest/858/submission/34159847

Lecture Exercise #3: Problem

- https://codeforces.com/problemset/problem/295/B

Lecture Exercise #3: Solution

- https://codeforces.com/contest/295/submission/15988207

Lecture Exercise #4: Problem

- https://codeforces.com/problemset/problem/459/E

Lecture Exercise #4: Solution

- https://codeforces.com/contest/459/submission/20755808

BREAK

DP Part II

1. dp optimization: Problem [CF 2600]

- https://codeforces.com/contest/321/problem/E

1. dp optimization: Solution [CF 2600]

- https://codeforces.com/blog/entry/8192 (321E)

1. dp optimization: Model Code [CF 2600]

- https://codeforces.com/contest/321/submission/29154603

1. dp optimization: Reference

- https://codeforces.com/blog/entry/8219
- https://usaco.guide/adv/dp-more?lang=cpp
- https://cp-algorithms.com/dynamic_programming/knuth-optimization.html

2. tree dp: Problem [CF 1800]

- https://codeforces.com/problemset/problem/161/D

2. tree dp: Solution [CF 1800]

- https://codeforces.com/blog/entry/4097 (161D)

2. tree dp: Model Code [CF 1800]

- https://codeforces.com/contest/161/submission/16033882

2. tree dp: Reference

- https://codeforces.com/blog/entry/20935
- https://usaco.guide/gold/dp-trees?lang=cpp

3. two pointers: Problem [CF 1600]

- https://codeforces.com/contest/616/problem/D

3. two pointers: Solution [CF 1600]

- https://codeforces.com/blog/entry/22712 (616D)

3. two pointers: Model Code [CF 1600]

- https://codeforces.com/contest/616/submission/15298635

3. two pointers: Reference

- https://codeforces.com/blog/entry/87248
- https://usaco.guide/gold/sliding-window?lang=cpp

4. greedy vs dp: Problem [CF 1500]

- https://codeforces.com/contest/1253/problem/C

4. greedy vs dp: Solution [CF 1500]

- https://codeforces.com/blog/entry/71489?locale=en (1253C)

4. greedy vs dp: Model Code [CF 1500]

- https://codeforces.com/contest/1253/submission/65218239

4. greedy vs dp: Reference

- https://stackoverflow.com/questions/16690249/what-is-the-difference-between-dynamic-programming-and-greedy-approach
- https://codeforces.com/blog/entry/20503

5. interval dp: Problem

- https://leetcode.com/problems/minimum-insertion-steps-to-make-a-string-palindrome/

5. interval dp: Solution

- https://leetcode.com/problems/minimum-insertion-steps-to-make-a-string-palindrome/discuss/470706/JavaC%2B%2BPython-Longest-Common-Sequence

5. interval dp: Model Code

```
int minInsertions(string s) {
    int n = s.size();
    vector<vector<int>> dp(n, vector<int>(n,0));
    for (int i = 1; i < n; i++)
        for (int j = 0, k = i; k < n; j++, k++)
            dp[j][k] = (s[j]==s[k]) ?
                            dp[j+1][k-1] :
                            min(dp[j][k-1],dp[j+1][k])+1;
    return dp[0][n-1];
}
```

6. coin exchange: Problem

- https://leetcode.com/problems/coin-change/

6. coin exchange: Solution

- https://leetcode.com/problems/coin-change/discuss/778548/C%2B%2B-DP-solution-explained-~100-Time-100-Space

6. coin exchange: Model Code

```
const int inf=1e6+777;
int coinChange(vector<int>& v, int n) {
    if(n<0) return -1;
    if(n==0) return 0;
    vector<int> dp(n+1,inf); dp[0]=0;
    sort(v.begin(),v.end());
    for (auto x : v)
        for (int i=0; i<=n-x; i++)
            dp[i+x]=min(dp[i+x],dp[i]+1);
    return dp[n]==inf?-1:dp[n];
}
```

7. numbers: Problem [CF 1900]

- https://codeforces.com/problemset/problem/9/D

7. numbers: Solution [CF 1900]

- https://codeforces.com/blog/entry/283 (D)

7. numbers: Model Code [CF 1900]

- [C++] https://codeforces.com/contest/9/submission/83587530
- [Python] https://codeforces.com/contest/9/submission/3014721

7. numbers: Reference

- [Catalan] https://codeforces.com/blog/entry/87585
- [generating function] https://codeforces.com/blog/entry/77468
- [generatingfunctionology] https://www2.math.upenn.edu/~wilf/gfologyLinked2.pdf

LUNCH

Practice Contest

https://codeforces.com/gym/437880

>>here<<<

Debrief

Closing

Where to go from here? (for ICPC)

- **Selection test** (locals in UCF)!
    - Make it to the team, to represent UCF in the regionals
- **Regional**s
    - Be the top few teams
- **North America Championship (NAC)**s
    - Make it to the world finals, with a goal of ‘medal’-ing
- **World Final (WF)**s
    - Make it to the top 12, for a medal!
- **GOAL**: You should make it a goal to reach rating of **2600+** in CodeForces (roughly **top 300**)

Where to go from here? (for training)

- Train, train, train, BUT only go so much to **NOT** burnout. **IT IS REAL!**
- Each and every one of you can do it, from what I observed last few days!!
- Register for **Universal Cup**: ask Arup, if interested!
- **CSES**: https://cses.fi/problemset/
- **Kattis**: https://open.kattis.com/ with its companion: https://cpbook.net/methodstosolve?oj=kattis&topic=all&quality=all
- **USACO Guide:** https://usaco.guide/ (especially Platinum and Advanced)
- **CP Algorithm:** https://cp-algorithms.com/
  - String Processing; Graphs; Linear Algebra; Data Structures; ...

International Collegiate Programming Contest (ICPC)

- If you would like to get involved in helping out as a volunteer or an official (unpaid) intern, please reach out to me at **yongwoods@icpc.global**.

Discord Servers: Please Join Us/Me!

- You are the most enthusiastic, driven group of students I have ever interacted with!!!

- You will be able to **continue** your **enthusiasm** through this group!

- So, please join the following discord servers, if you have not already!!!

  [ICPC CodeForces Zealots] https://discord.gg/7bvMnMyF6G

Practice makes PERFECT!

- Do as **many practice contests** as you can!
    - **Live Contests**
        - CodeForces: Division 1-4
        - AtCoder: Beginner; Regular; Grand;
        - LeetCode: Weekly/Biweekly
    - **ICPC North America Practice Contests** on:
        - **Sundays** from 1pm ET to 6pm ET
    - **Zealot Problem Sets**
        - **Everyday** (24 hours 7 days a week)!

1:1 Meeting Opportunity

- If you would like to meet in 1:1, please sign up using: https://calendly.com/yongwhan/one-on-one.

- **I'd love to help you landing your dream job!**

A Terse Guide on ICPC Contest Strategies

- We have Google Drive to Terse Guides, of course!
- These documents will be frequently expanded upon later.
- If you have any questions, please reach out to yongwhan@yongwhan.io!

Contact Information

- **Email**: yongwhan@yongwhan.io
- **Personal Website**: https://www.yongwhan.io/
- **LinkedIn Profile**: https://www.linkedin.com/in/yongwhan/
    - Feel free to send me a connection request!
    - Always happy to make connections with promising students!

Slide Decks

- You can find the slide decks from the presentations today and yesterday from:

https://github.com/yongwhan/yongwhan.github.io/tree/master/uta

THANK YOU