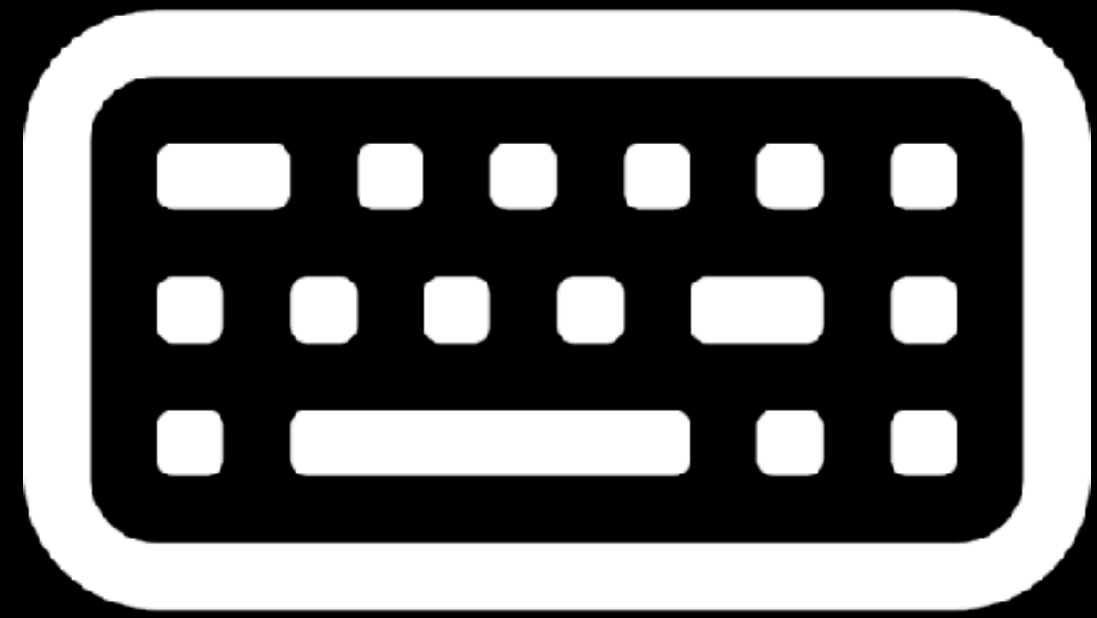


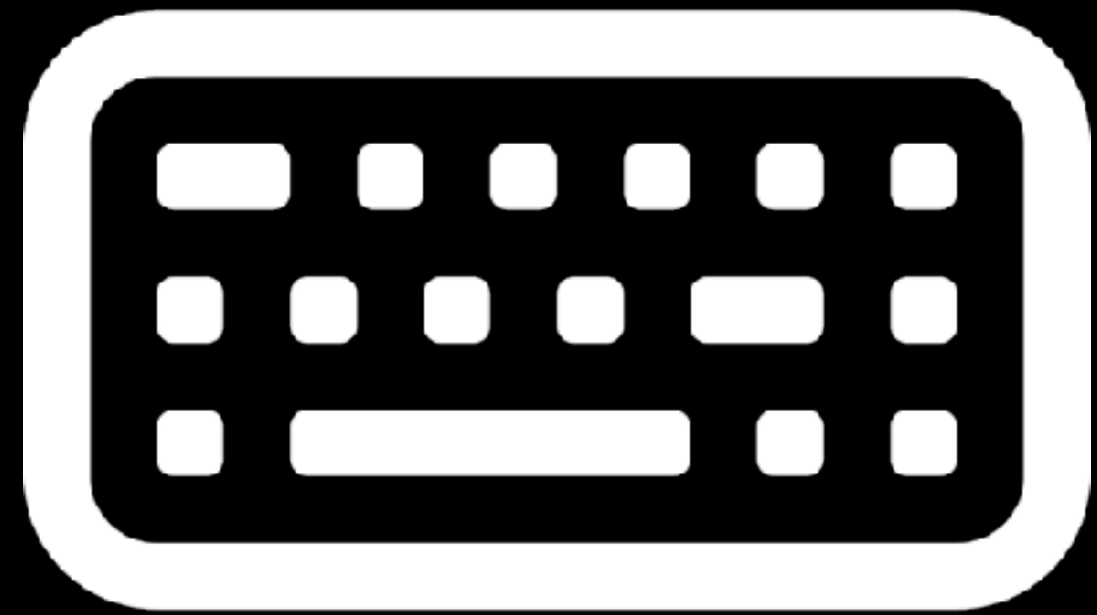
Shayan Hosseini - Feb 2023

Competitive Programming

aka Sport Programming



Programming



Programming



Time

How much time?

How much time?

- Short:
 - Up to 5 hours

How much time?

- Short:
 - Up to 5 hours
- Long:
 - More than 5 hours, typically at least a day
 - More sort of a hackathon

How much time?

- **Short:**
 - **Up to 5 hours**
- Long:
 - More than 5 hours, typically at least a day
 - More sort of a hackathon

What sort of programming?

- Algorithmic puzzle solving
- Capture the flag
- Ad hoc

What sort of programming?

- **Algorithmic puzzle solving**
- Capture the flag
- Ad hoc















Algorithmic puzzle solving

Famous competitions

- College students:
 - ICPC — International Collegiate Programming Contest
- High school students:
 - IOI — International Olympiad in Informatics
- Open division:
 - Google Code Jam
 - Meta Hacker Cup
- Websites:
 - Codeforces
 - Top Coder

ICPC World Finals Dhaka














final standings

RANK	TEAM	SCORE	A	B	C	D	E	F	G	H	I	J	K	L
1	 Massachusetts Institute of Technology <small>North America</small>	11 1339	50 1 try	184 2 tries	25 2 tries	19 19 tries	196 1 try	122 1 try	166 3 tries	43 2 tries	106 1 try	69 1 try	248 1 try	30 1 try
2	 Peking University <small>Asia East</small>	10 1711	71 1 try	133 4 tries	104 3 tries	5 5 tries	274 3 tries	207 1 try	154 4 tries	48 4 tries	204 1 try	69 2 tries		167 1 try
3	 The University of Tokyo <small>Asia Pacific</small>	9 1036	128 1 try	150 1 try	47 1 try	2 2 tries		64 1 try	85 1 try	17 1 try	271 3 tries	37 1 try	5 5 tries	177 2 tries
4	 Seoul National University	9 1303	113 1 try	76 3 tries	73 2 tries	1 1 try	286 2 tries	159 1 try	228 6 tries	11 1 try	4 4 tries	47 1 try	1 1 try	130 1 try
5	 ETH Zürich <small>Europe</small>	9 1541	157 1 try	274 3 tries	86 3 tries	1 1 try	1 1 try	238 2 tries	293 2 tries	22 1 try	61 1 try	92 1 try		158 3 tries
6	 École Normale Supérieure de Paris	9 1706	136 2 tries	293 2 tries	162 5 tries			208 1 try	256 1 try	29 2 tries	287 1 try	43 1 try		132 2 tries
7	 Carnegie Mellon University	9 1741	258 1 try	2 2 tries	64 3 tries		222 1 try	245 1 try	183 2 tries	118 3 tries	125 1 try	148 1 try		278 1 try
8	 University of Warsaw	8 946	87 1 try	3 3 tries	43 1 try			118 1 try	218 3 tries	24 1 try	201 2 tries	48 1 try		127 2 tries
9	 National Research University Higher School of Economics <small>Northern Eurasia</small>	8 1092	217 1 try	91 2 tries	63 1 try			111 1 try	189 1 try	18 1 try	3 3 tries	98 2 tries		185 5 tries
10	 St. Petersburg State University	8 1217	191 1 try	9 9 tries	38 3 tries			197 1 try	110 3 tries	16 1 try	255 1 try	126 4 tries		144 1 try
11	 University of Oxford	8 1269	56 1 try	165 2 tries	189 4 tries			264 1 try	211 1 try	19 1 try	2 2 tries	127 2 tries		118 2 tries
12	 University of Engineering and Technology - VNU	8 1430	70 1 try	168 1 try	35 2 tries			207 1 try	147 1 try	160 3 tries	2 2 tries	198 3 tries		285 4 tries
13	 KTH - Royal Institute of Technology	8 1583	285 3 tries	288 4 tries	91 2 tries			178 1 try	243 1 try	21 1 try		140 1 try		197 2 tries
14	 Shanghai Jiao Tong University	8 1586	221 1 try	277 6 tries	118 1 try			169 1 try	237 2 tries	55 4 tries		108 1 try		181 3 tries

Hello 2023

Final standings

You may double click into cells (or ctrl+click) to view the submissions history or hack the solution

Standings 												
#	Who	=	*	A 500	B 750	C 1250	D 1500	E 2250	F 2250	G 2750	H 4000	
1	 Benq	12382		497 00:02	686 00:06	1214 00:09	1428 00:15	2070 00:25	1926 00:45	2108 01:13	2453 01:57	
2	 tourist	12286		491 00:06	691 00:04	1214 00:09	1409 00:19	2070 00:25	1934 00:44	2064 01:18	2413 02:04	
3	 fantasy	11731		492 00:05	676 00:10	1190 00:15	1371 00:27	1898 00:49	1811 01:01	1738 01:55	2555 01:49	
4	 hitonanode	11543		497 00:02	684 00:07	1148 00:13	1395 00:22	1998 00:35	1775 01:06	1976 01:28	2070 02:23	
5	 jiangly_fan	10590		496 00:03	729 00:09	1024 00:19	1380 00:25	1970 00:39	1618 01:07	1730 01:56	1643 02:29	
6	 jiangly	9781		494 00:04	681 00:08	1202 00:12	1380 00:25	2027 00:31	1898 00:49	2099 01:14		
7	 duality	9715		497 00:02	736 00:06	1202 00:12	1404 00:20	2056 00:27	1862 00:54	1958 01:30	-1	
8	 maroonrk	9609		497 00:02	731 00:08	1198 00:13	1400 00:21	2013 00:33	1812 00:54	1958 01:30	-4	
9	 Maksim1744	9510		497 00:02	731 00:08	1190 00:15	1404 00:20	2027 00:31	1761 01:08	1900 01:31	-6	
10	 -0.5	9453		497 00:02	738 00:05	1206 00:11	1321 00:27	1998 00:35	1682 01:12	2011 01:24	-3	
11	inaFSTream	9340		497 00:02	734 00:07	1210 00:10	1390 00:23	1984 00:37	1898 00:49	1627 02:02		
12	 Radewoosh	9293		494 00:04	619 00:13	1178 00:18	1380 00:25	1912 00:40	1804 01:02	1906 01:36		

How it works?

1. Problem

Job Search

Time Limit: 2 seconds

Memory Limit: 256 megabytes

Shayan is looking for a job in tech. There are n companies hiring software engineers, but Shayan is extremely concerned with the culture of tech companies. Shayan exactly knows how toxic the culture of every company is. We denote the toxicity of company i 's culture by a_i ($1 \leq i \leq n$).

From the set of all hiring companies, Shayan wants to apply to a non-empty set of companies, such that the geometric mean of their toxicity is minimum. Help Shayan find the minimum geometric mean for a non-empty set of companies.

For a set of numbers a_1, a_2, \dots, a_n the geometric mean is defined as:

$$\left(\prod_{i=1}^n a_i \right)^{\frac{1}{n}} = \sqrt[n]{a_1 a_2 \dots a_n}$$

$$1 \leq n \leq 10^6$$

$$1 \leq a_i \leq 10^9 \quad (1 \leq i \leq n)$$

$O(n)$

$$1 \leq n \leq 10^6$$

$$1 \leq a_i \leq 10^9 \quad (1 \leq i \leq n)$$

$$1 \leq n \leq 10^6$$
$$1 \leq a_i \leq 10^9 \quad (1 \leq i \leq n)$$

$$O(n)$$
$$O(n \log n)$$

$$1 \leq n \leq 10^6$$

$$1 \leq a_i \leq 10^9 \quad (1 \leq i \leq n)$$

$$O(n)$$

$$O(n \log n)$$

Sample Input	Sample Output
5 MindGeek 9 TechBros 15 Web100 8 BestTech 12 qwerty 6	6.00

2. Coming up with an idea

**You have some intuition
about why your idea works**

You have some intuition about why your idea works

- You have two options:

You have some intuition about why your idea works

- You have two options:
 - Prove it!

You have some intuition about why your idea works

- You have two options:
 - Prove it!
 - Stick with intuition and proceed to the next step

You have some intuition about why your idea works

- You have two options:
 - Prove it!
 - Stick with intuition and proceed to the next step

Goldbach's conjecture is one of the oldest and best-known **unsolved problems** in **number theory** and all of **mathematics**. It states that every **even natural number** greater than 2 is the sum of two **prime numbers**.

3. Write down the code

Which programming language?

Which programming language?

- Requirements:

Which programming language?

- Requirements:
 - Being fast

Which programming language?

- Requirements:
 - Being fast
 - Has a rich set of utilities in the standard library:

Which programming language?

- Requirements:
 - Being fast
 - Has a rich set of utilities in the standard library:
 - Common algorithms and data structure

Which programming language?

- Requirements:
 - Being fast
 - Has a rich set of utilities in the standard library:
 - Common algorithms and data structure
 - Being able to quickly write the code

Which programming language?

- Requirements:
 - Being fast
 - Has a rich set of utilities in the standard library:
 - Common algorithms and data structure
 - Being able to quickly write the code
- C++

```
#include <iostream>
using namespace std;
```

```
int main() {
    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
int main() {
    vector<int> a;
    return 0;
}
```

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
    vector<int> a;
    return 0;
}
```

```
#include <iostream>
#include <vector>
using namespace std;

int main() {
    vector<int> a;
    string s;
    return 0;
}
```

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;

int main() {
    vector<int> a;
    string s;
    return 0;
}
```

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <map>
#include <vector>
#include <list>
#include <set>
#include <queue>
#include <deque>
#include <algorithm>
#include <bitset>
#include <complex>
#include <stack>
```

```
#include <cstring>
#include <cstdio>
#include <cstdlib>
#include <cctype>
#include <cmath>
#include <climits>
#include <ctime>
using namespace std;

int main() {
    return 0;
}
```


One line to rule them all

One line to rule them all

- Around 2014, a genius person found `bits/stdc++.h` in the standard library

One line to rule them all

- Around 2014, a genius person found `bits/stdc++.h` in the standard library
- `bits/stdc++.h` has already included all the other header files

One line to rule them all

- Around 2014, a genius person found `bits/stdc++.h` in the standard library
- `bits/stdc++.h` has already included all the other header files

```
#include <bits/stdc++.h>  
using namespace std;
```

```
int main() {  
    return 0;  
}
```

```
#include <bits/stdc++.h>
using namespace std;

const int N = 10;

int main() {
    int x, y;
    vector<pair<int, int>> v;
    for (int i = 0; i < N; i++) {
        cin >> x >> y;
        v.push_back(make_pair(x, y));
    }

    vector<int> s;
    for (int i = 0; i < N-1; i++) {
        s.push_back(v[i].first + v[i+1].second);
    }
    return 0;
}
```

```
#include <bits/stdc++.h>
using namespace std;

#define pb push_back
#define mp make_pair
#define X first
#define Y second
#define REP(i,n) for(int(i)=0;(i)<(int)(n);(i)++)
typedef pair<int, int> pii;

const int N = 10;

int main() {
    int x, y;
    vector<pii> v;
    REP(i, N) cin >> x >> y, v.pb(mp(x, y));
    vector<int> s;
    REP(i, N-1) s.pb(v[i].X + v[i+1].Y);
    return 0;
}
```

```

#include <bits/stdc++.h>
using namespace std;

#define pb push_back
#define mp make_pair
#define X
#define Y
#define R
typedef p
const int

int main(
int x
vector
REP(i, N) cin >> x >> y, v.pb(mp(x, y));
vector<int> s;
REP(i, N-1) s.pb(v[i].X + v[i+1].Y);
return 0;
}

```



pushback

;(i)++)



pb

```

using pi = pair<ll, ll>;
using vi = vector<ll>;
template <class T>
using vc = vector<T>;
template <class T>
using vvc = vector<vc<T>>;
template <class T>
using vvvc = vector<vvvc<T>>;
template <class T>
using vvvvc = vector<vvvvc<T>>;
template <class T>
using vvvvvc = vector<vvvvvc<T>>;
template <class T>
using pq = priority_queue<T>;
template <class T>
using pqg = priority_queue<T, vector<T>, greater<T>>;

#define vv(type, name, h, ...) \
    vector<vector<type>> name(h, vector<type>(__VA_ARGS__))
#define vvv(type, name, h, w, ...) \
    vector<vector<vector<type>>> name( \
        h, vector<vector<type>>(w, vector<type>(__VA_ARGS__)))
#define vvvv(type, name, a, b, c, ...) \
    vector<vector<vector<vector<type>>>> name( \
        a, vector<vector<vector<type>>>( \
            b, vector<vector<type>>(c, vector<type>(__VA_ARGS__)))

// https://trap.jp/post/1224/
#define FOR1(a) for (ll _ = 0; _ < ll(a); ++_)
#define FOR2(i, a) for (ll i = 0; i < ll(a); ++i)
#define FOR3(i, a, b) for (ll i = a; i < ll(b); ++i)
#define FOR4(i, a, b, c) for (ll i = a; i < ll(b); i += (c))
#define FOR1_R(a) for (ll i = (a)-1; i >= ll(0); --i)
#define FOR2_R(i, a) for (ll i = (a)-1; i >= ll(0); --i)
#define FOR3_R(i, a, b) for (ll i = (b)-1; i >= ll(a); --i)
#define overload4(a, b, c, d, e, ...) e
#define overload3(a, b, c, d, ...) d
#define FOR(...) overload4(__VA_ARGS__, FOR4, FOR3, FOR2, FOR1)(__VA_ARGS__)
#define FOR_R(...) overload3(__VA_ARGS__, FOR3_R, FOR2_R, FOR1_R)(__VA_ARGS__)

#define FOR_subset(t, s) \
    for (ll t = (s); t >= 0; t = (t == 0 ? -1 : (t - 1) & (s)))
#define all(x) x.begin(), x.end()
#define len(x) ll(x.size())
#define elif else if

#define eb emplace_back
#define mp make_pair
#define mt make_tuple
#define fi first
#define se second

#define stoi stoll

int popcnt(int x) { return __builtin_popcount(x); }
int popcnt(u32 x) { return __builtin_popcount(x); }
int popcnt(ll x) { return __builtin_popcountll(x); }
int popcnt(u64 x) { return __builtin_popcountll(x); }
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 1, 2)
int topbit(int x) { return (x == 0 ? -1 : 31 - __builtin_clz(x)); }
int topbit(u32 x) { return (x == 0 ? -1 : 31 - __builtin_clz(x)); }
int topbit(ll x) { return (x == 0 ? -1 : 63 - __builtin_clzll(x)); }
int topbit(u64 x) { return (x == 0 ? -1 : 63 - __builtin_clzll(x)); }
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 0, 2)
int lowbit(int x) { return (x == 0 ? -1 : __builtin_ctz(x)); }
int lowbit(u32 x) { return (x == 0 ? -1 : __builtin_ctz(x)); }
int lowbit(ll x) { return (x == 0 ? -1 : __builtin_ctzll(x)); }
int lowbit(u64 x) { return (x == 0 ? -1 : __builtin_ctzll(x)); }

```

```
#include <bits/stdc++.h>
```

```

#define mp make_pair
#define mt make_tuple
#define fi first
#define se second
#define pb push_back
#define sz(x) ((int)((x).size())
#define all(x) (x).begin(), (x).end()
#define rall(x) (x).rbegin(), (x).rend()
#define forn(i, n) for (int i = 0; i < (int)(n); ++i)
#define forl(i, n) for (int i = 1; i <= (int)(n); ++i)
#define ford(i, n) for (int i = (int)(n) - 1; i >= 0; --i)
#define fore(i, a, b) for (int i = (int)(a); i <= (int)(b); ++i)

```

```
using namespace std;
```

```

typedef pair<int, int> pii;
typedef vector<int> vi;
typedef vector<pii> vpi;
typedef vector<vi> vvi;
typedef long long i64;
typedef vector<i64> vi64;
typedef vector<vi64> vvi64;
typedef pair<i64, i64> pi64;
typedef double ld;

```

```

template<class T> bool uin(T &a, T b) { return a > b ? (a = b, true) : false; }
template<class T> bool uax(T &a, T b) { return a < b ? (a = b, true) : false; }

```

<https://codeforces.com/contest/1785/submission/192337551>


```
#define int long long
```

Community

Community

- Around websites such as Codeforces

Community

- Around websites such as Codeforces
- ~ Weekly contests

Community

- Around websites such as Codeforces
- ~ Weekly contests
- How it works?
 - Contests are authored and prepared by community
 - Authors get paid by the website
 - Website itself gets supported by different tech companies
 - Companies sponsor contests to get exposure and hire talented programmers

Legendary Grandmaster

jiangly ★

Lingyu Jiang, [Chongqing, China](#)
From [Jiangly Fan Club](#)

📊 Contest rating: **3474** (max. **legendary grandmaster, 3754**)

★ Contribution: **+119**

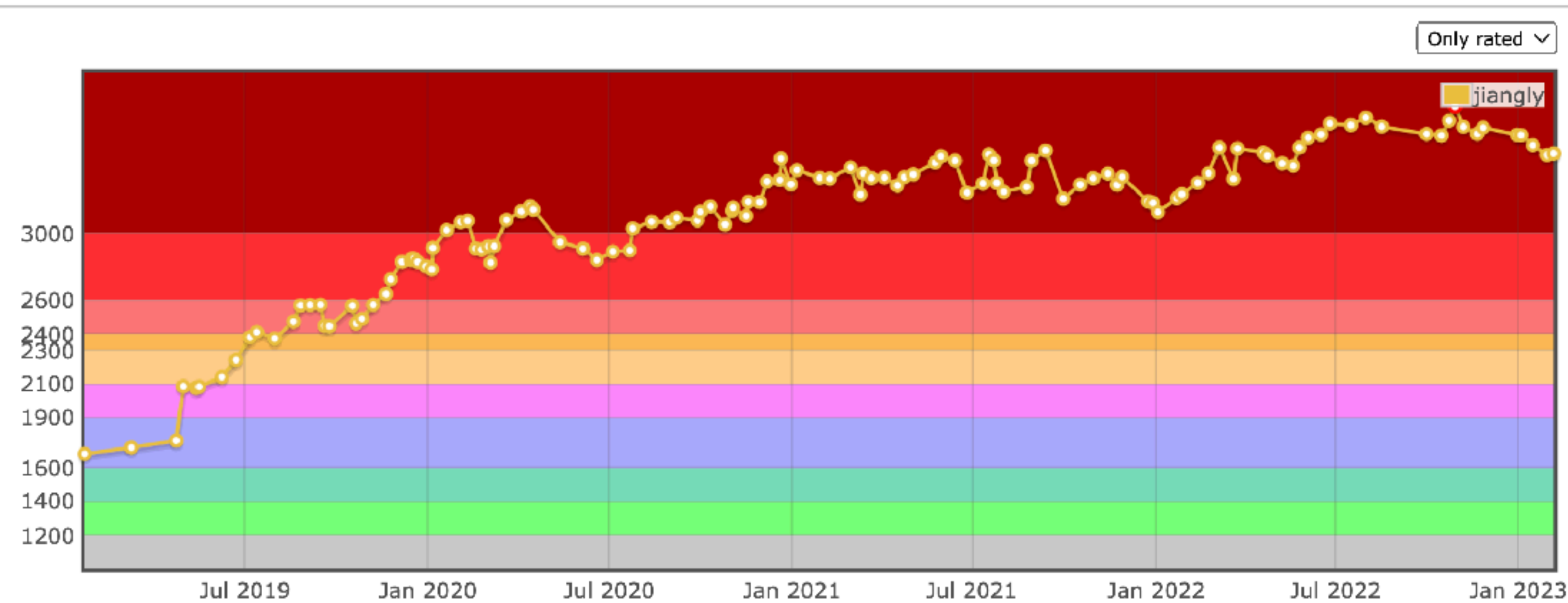
★ Friend of: 12,968 users

Last visit: 10 hours ago

Registered: 4 years ago

📖 [Blog entries \(2\)](#), [comments](#)












✉ [Talks](#) | [Send message](#)



Elo-MMR	Title	Division	Number	Percentile	CF at same rank (spread)
3000+	Legendary Grandmaster	1	8	99.99	3382+
2700-2999	International Grandmaster	1	37	99.95	3010-3329 (372)
2400-2699	Grandmaster	1	255	99.7	2565-3010 (445)
2200-2399	International Master	1	560	99.1	2317-2565 (248)
2000-2199	Master	1	2089	97	2088-2317 (229)
1800-1999	Candidate Master	2	3968	93	1804-2088 (284)
1600-1799	Expert	2	7103	86	1564-1804 (240)
1400-1599	Specialist	3	11003	75	1328-1564 (236)
1200-1399	Apprentice	3	16909	58	1104-1328 (224)
1000-1199	Pupil	4	23977	34	818-1104 (286)
Up to 999	Newbie	4	33923	0	Up to 818

Top Participants by Performance in Major Contests

By Major Contests By Prize Money

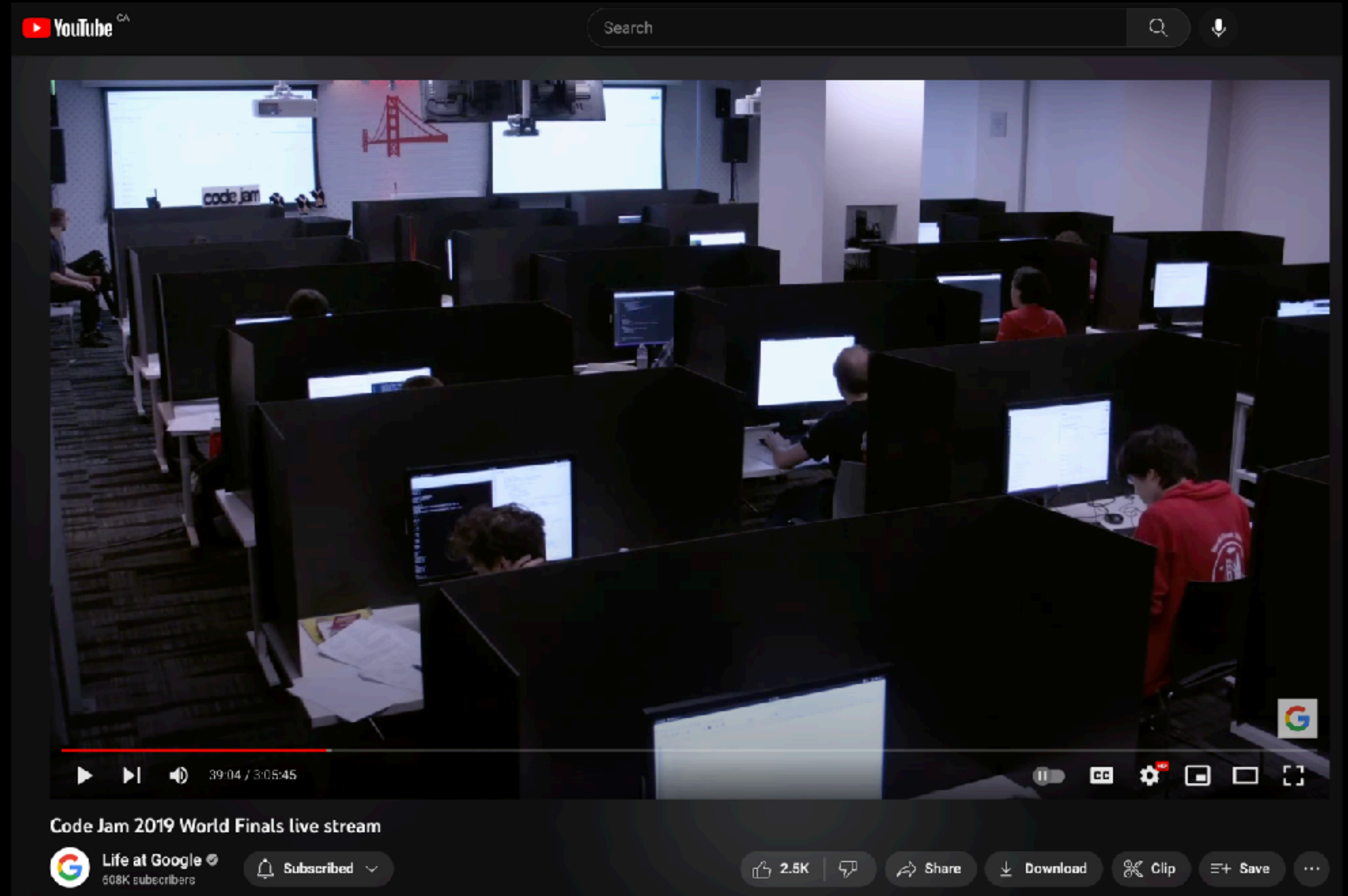
	Name	Major
1	 Gennady Korotkevich (tourist)	38 + Q×3 G×21 S×1 B×1
2	 Petr Mitrichev (Petr)	31 G×9 S×4 B×3
3	 Przemyslaw Debiak (Psyho)	17 + Q×2 G×7 S×3 B×1
4	 Makoto Soejima (rng_58)	23 G×5 S×4 B×3
5	 Tiancheng Lou (ACRush)	29 + Q×1 G×3 S×3 B×2
6	 Tomasz Czajka (tomek)	16 G×3 S×3 B×1
7	 Andrew He (ecnerwal)	16 + Q×2 G×2 S×1 B×5
8	 Bruce Merry (bmerry)	19 G×2 S×1 B×2
9	 Yuhao Du (xudyh)	10 + Q×5 G×2 S×1
10	 Egor Kulikov (Egor)	22 G×2 B×3
11	 Jakub Pachocki (meret)	9 G×1 S×2 B×1

Streaming

YouTube, Twitch, ...

Streaming

YouTube, Twitch, ...




The image shows a YouTube video player interface. At the top, the YouTube logo and a search bar are visible. The main video area displays a live stream of a coding competition, with participants seated at desks with computers in a large room. A red 'code jam' sign is visible in the background. Below the video, the title 'Code Jam 2019 World Finals live stream' is shown, along with the channel name 'Life at Google' and '508K subscribers'. The video progress bar indicates 39:04 / 3:05:45. The bottom right corner features interaction buttons: Like (2.5K), Comment, Share, Download, Clip, Save, and a menu icon.

YouTube ^{CA} Search

code jam

39:04 / 3:05:45

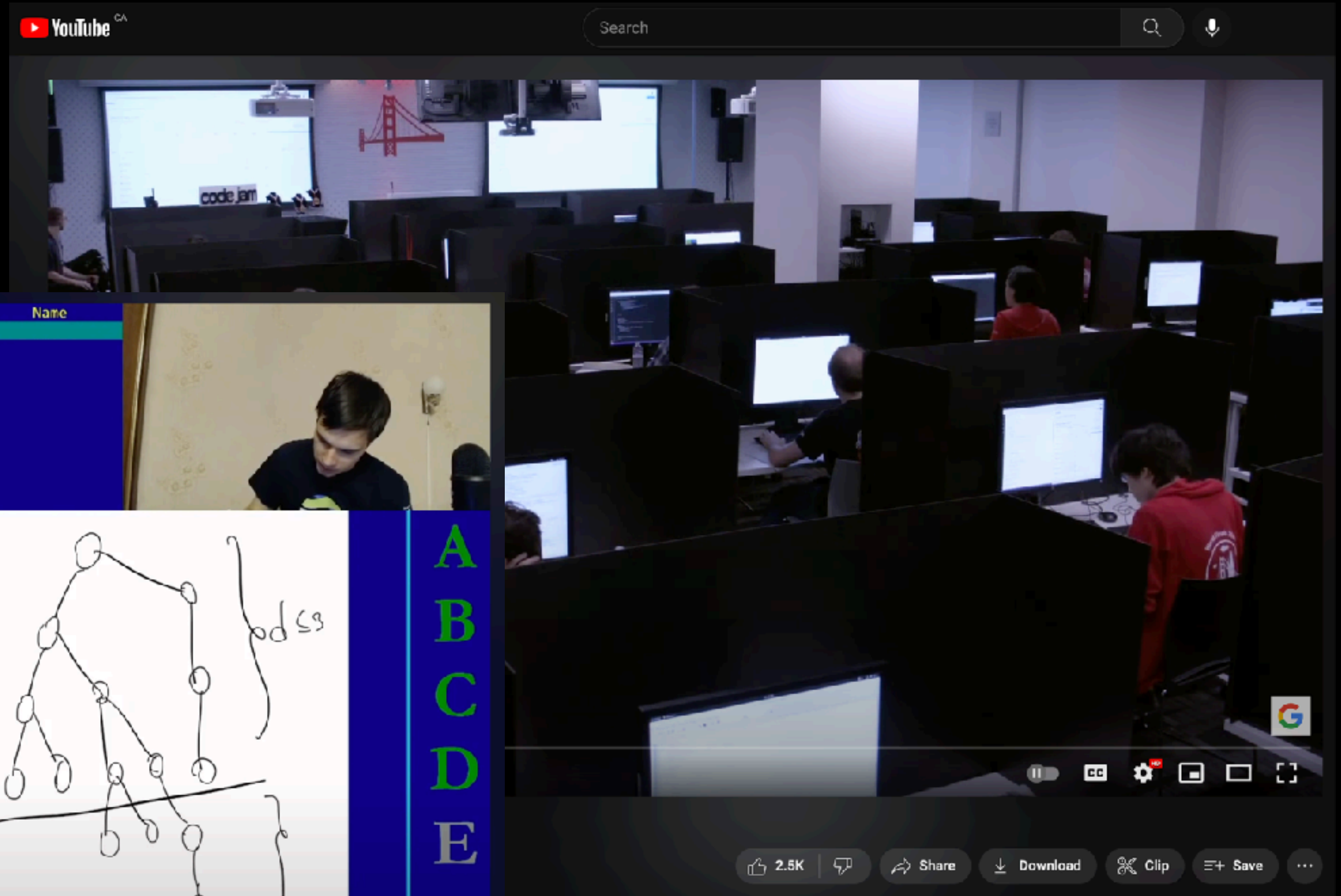
Code Jam 2019 World Finals live stream

Life at Google  508K subscribers

Subscribed

2.5K | Comment | Share | Download | Clip | Save | ...

Streaming YouTube, Twitch, ...



time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

This version of the problem differs from the next one only in the constraint on n .

A tree is a connected undirected graph without cycles. A weighted tree has a weight assigned to each edge. The distance between two vertices is the minimum sum of weights on the path connecting them.

You are given a weighted tree with n vertices, each edge has a weight of 1. Denote $d(v)$ as the distance between vertex 1 and vertex v .

Let $f(x)$ be the minimum possible value of $\max_{1 \leq v \leq n} d(v)$ if you can temporarily add an edge with weight x between any two vertices a and b ($1 \leq a, b \leq n$). Note that after this operation, the graph is no longer a tree.

For each integer x from 1 to n , find $f(x)$.

Input
The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases.
The first line of each test case contains a single integer n ($2 \leq n \leq 3000$).
Each of the next $n - 1$ lines contains two integers u and v ($1 \leq u, v \leq n$) indicating that there is an edge between vertices u and v . It is guaranteed that the given edges form a tree.
It is guaranteed that the sum of n over all test cases doesn't exceed 3000.

Output
For each test case, print n integers in a single line, x -th of which is equal to $f(x)$ for all x from 1 to n .

Example

```
input
3
4
1 2
2 3
1 4
```

Name

$d=3$

A
B
C
D
E

touriststream 018: Codeforces Round 769 (Div. 2)

Gennady Korotkevich
37.8K subscribers

Subscribe

733



Share

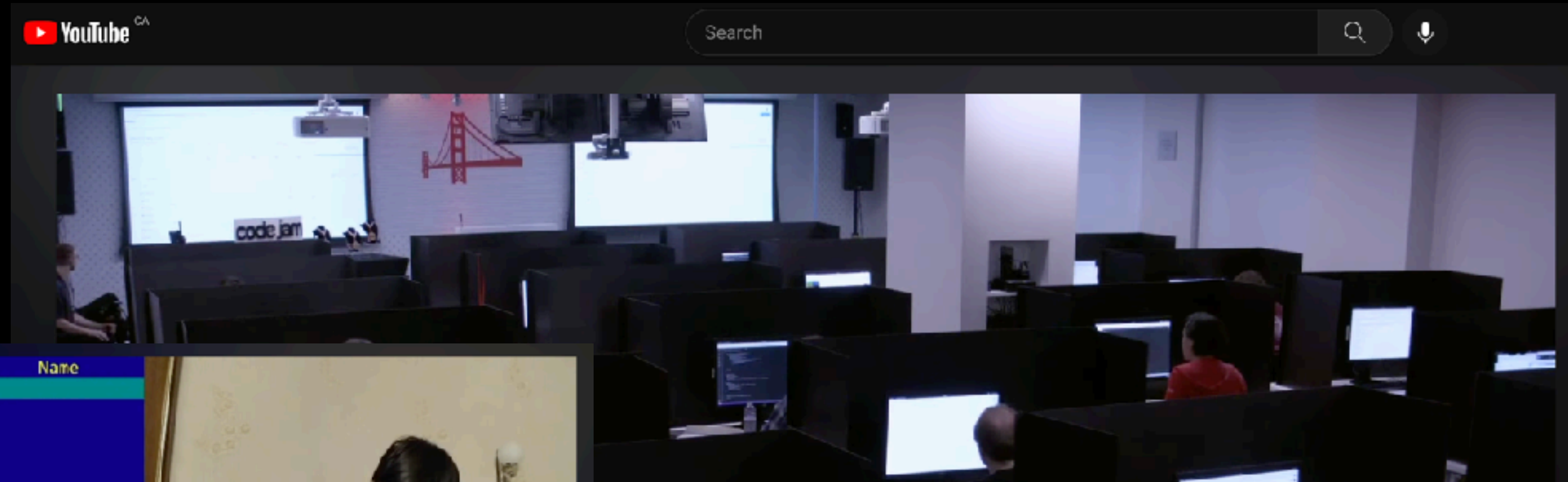
Download

Clip

Save



Streaming YouTube, Twitch, ...



time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

This version of the problem differs from the next one only in the constraint on n .

A tree is a connected undirected graph without cycles. A weighted tree has a weight assigned to each edge. The distance between two vertices is the minimum sum of weights on the path connecting them.

You are given a weighted tree with n vertices, each edge has a weight of 1. Denote $d(v)$ as the distance between vertex 1 and vertex v .

Let $f(x)$ be the minimum possible value of $\max_{1 \leq v \leq n} d(v)$ if you can temporarily add an edge with weight x between any two vertices a and b ($1 \leq a, b \leq n$). Note that after this operation, the graph is no longer a tree.

For each integer x from 1 to n , find $f(x)$.

Input
The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases.

The first line of each test case contains a single integer n ($2 \leq n \leq 3000$).

Each of the next $n - 1$ lines contains two integers u and v ($1 \leq u, v \leq n$) indicating that there is an edge between vertices u and v . It is guaranteed that the given edges form a tree.

It is guaranteed that the sum of n over all test cases doesn't exceed 3000.

Output
For each test case, print n integers in a single line, x -th of which is equal to $f(x)$ for all x from 1 to n .

Example

```
input
3
4
1 2
2 3
3 4
1 4
```

touristream 018: Codeforces Round 769 (Div. 2)

Gennady Korotkevich
37.8K subscribers

Subscribe

733

Qualification Round - Hash Code 2022

Practice mode

Problem description

Contributors

There are N contributors. Each contributor has a name and one or more skills at a specific level (0, 1, 2, ...). Not possessing a skill at level 0.

For example, three contributors could have the following skills:

- Anna: Python level 3
- Bob: C++ level 3
- Maria: HTML level 4, CSS level 6

Projects

There are M projects. Each project is described by:

- its name
- the duration of the project in days (how long it takes to complete a project once it is started)
- the score awarded for completing the project
- the "best before" time in days — if the project's last day of work is strictly before the indicated day, it earns the full score. If it is still worked on during or after its "best before day", it gets one point less for each day it is late, but no less than zero points in the "Assignments" section below.
- a list of roles for contributors working on the project

Each project has one or more roles that need to be filled by contributors. Each role requires one skill at a specific level, and each contributor can fill at most one role on a single project.

I Lost In Google Hash Code 2022

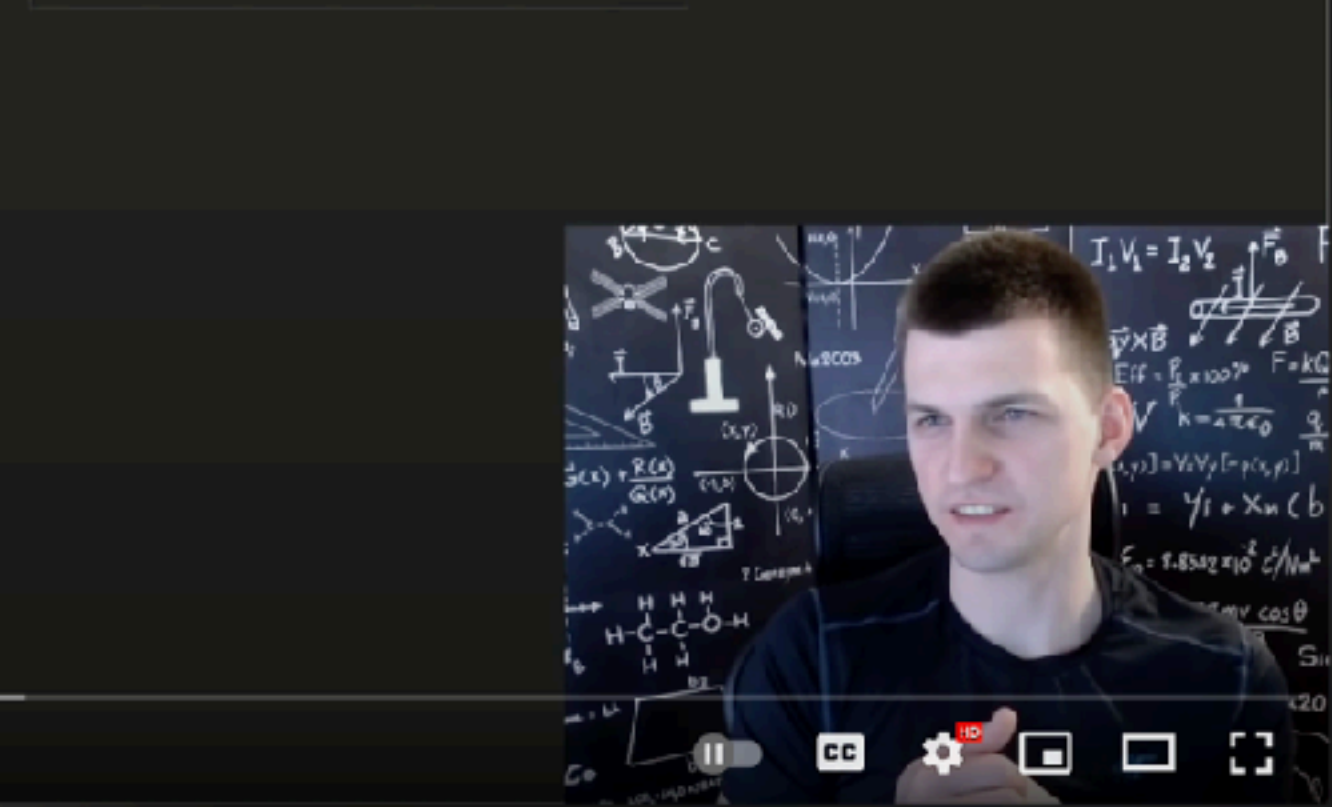
Errichto
285K subscribers

Subscribe

What we tried:

- 1) greedy
- 2) greedy
- 3) Gurobi solver
- 4) solution for test F

What we didn't try:



1K

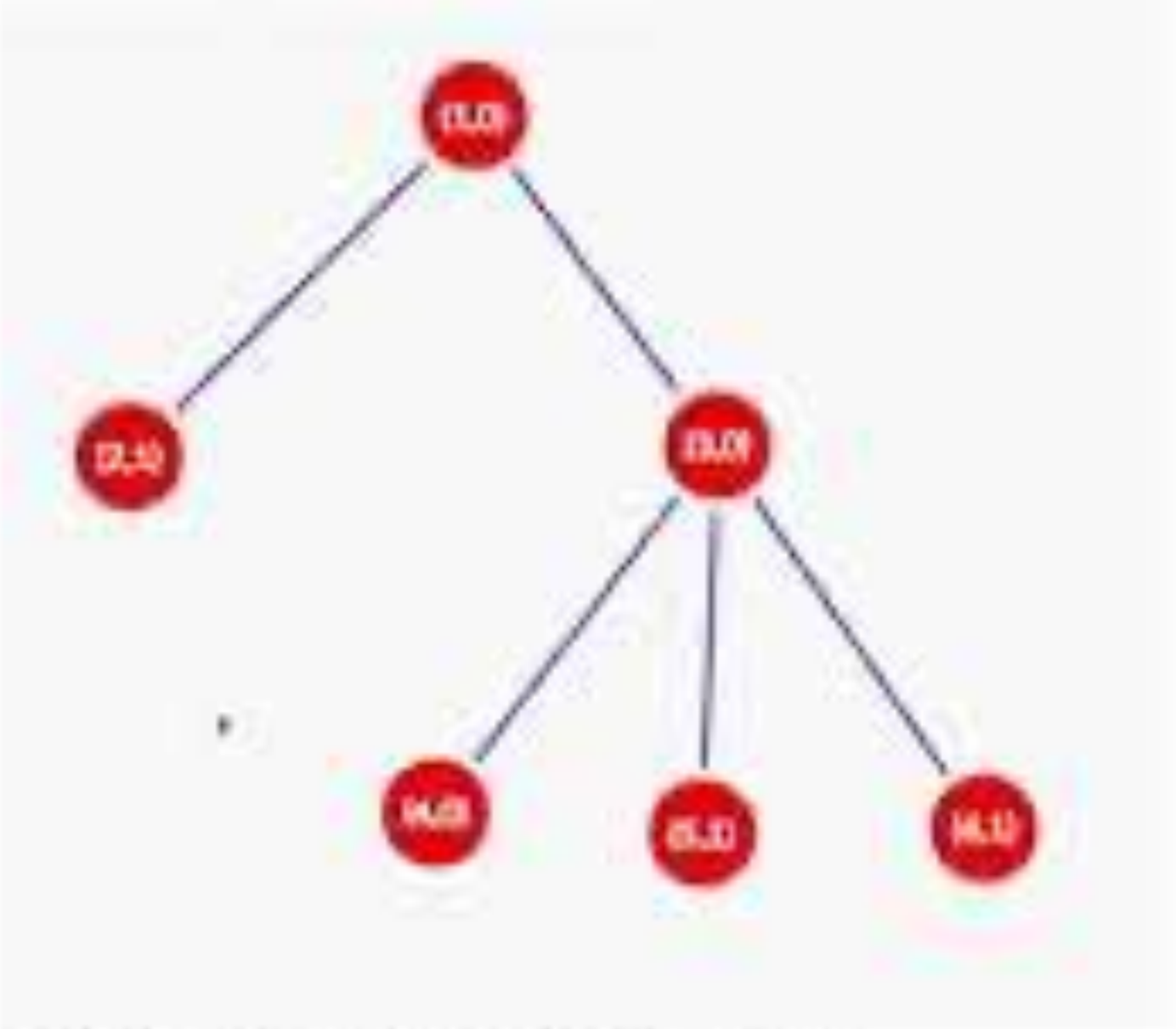
Share

Download

Clip

Save

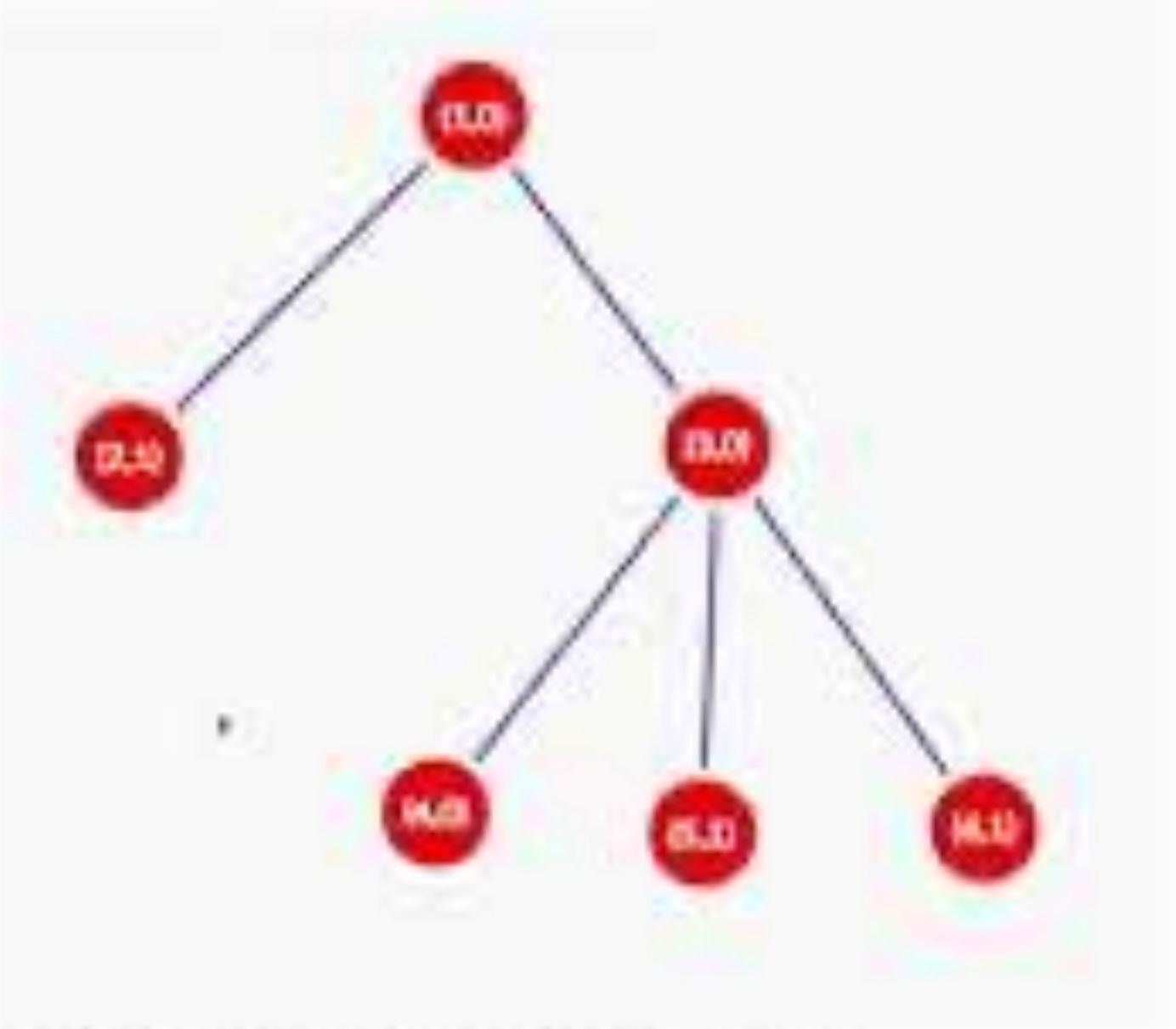
Let us find $f(1,0)$ for the configuration $A = (0, 1, 0, 0, 1, 1, 1, 1)$ (inserted the value of each cell initially set $f = 0$ for each cell as shown in the picture below. If each node, its value is stored the number and the value of this node. $f(1,0)$ for this configuration is 3



$f(1,0) = 3$ for configuration changed to $(1, 0, 0, 0, 0, 0, 0)$. The tree looks as shown below: $f(1,0) = 1$



Let us find $F(1,0)$ for the configuration $A = (0, 1, 0, 0, 1, 1, 1, 1)$ (inserted the value of each cell initially set $F = 0$ for each cell as shown in the picture below. If each node, its value is stored the number and the value of this node. $F(1,0)$ for this configuration is 3



$F(1,0) = 3$ for configuration changed to $(1, 0, 0, 0, 0, 0, 0)$. The tree looks as shown below: $F(1,0) = 1$



What sort of programming?

- Algorithmic puzzle solving
- Capture the flag
- Ad hoc

What sort of programming?

- Algorithmic puzzle solving
- **Capture the flag**
- Ad hoc

Capture The Flag

CTF

Capture The Flag

CTF

- Security / hacking

Capture The Flag

CTF

- Security / hacking
- You have to find the secretly hidden “flags”

Capture The Flag

CTF

- Security / hacking
- You have to find the secretly hidden “flags”
- Typically involves the following categories:
 - Binary Exploitation / Reverse Engineering
 - Cryptography
 - Web application security

What sort of programming?

- Algorithmic puzzle solving
- Capture the flag
- Ad hoc

What sort of programming?

- Algorithmic puzzle solving
- Capture the flag
- **Ad hoc**

Advent of Code

Ad hoc

- An annual set of Christmas-themed computer programming challenges that follow an Advent calendar.
- A new problem is released every night at 9pm PST, for 25 days
- First 100 people who solve a problem get points for that problem
 - 1st person: 100 pts
 - 2nd person: 99 pts
 - ...
 - 100 person: 1 pts
- More than 1 million registered users

Day 1 - Advent of Code 2022 x +

adventofcode.com/2022/day/1

Advent of Code [About] [Events] [Shop] [Settings] [Log Out] shayanh 35*
/*2022*/ [Calendar] [AoC++] [Sponsors] [Leaderboard] [Stats]

--- Day 1: Calorie Counting ---

Santa's reindeer typically eat regular reindeer food, but they need a lot of **magical energy** to deliver presents on Christmas. For that, their favorite snack is a special type of **star** fruit that only grows deep in the jungle. The Elves have brought you on their annual expedition to the grove where the fruit grows.

To supply enough magical energy, the expedition needs to retrieve a minimum of **fifty stars** by December 25th. Although the Elves assure you that the grove has plenty of fruit, you decide to grab any fruit you see along the way, just in case.

Collect stars by solving puzzles. Two puzzles will be made available on each day in the Advent calendar; the second puzzle is unlocked when you complete the first. Each puzzle grants **one star**. Good luck!

The jungle must be too overgrown and difficult to navigate in vehicles or access from the air; the Elves' expedition traditionally goes on foot. As your boats approach land, the Elves begin taking inventory of their supplies. One important consideration is food - in particular, the number of **Calories** each Elf is carrying (your puzzle input).

The Elves take turns writing down the number of Calories contained by the various meals, snacks, rations, etc. that they've brought with them, one item per line. Each Elf separates their own inventory from the previous Elf's inventory (if any) by a blank line.

For example, suppose the Elves finish writing their items' Calories and end up with the following list:

```
1000
2000
3000

4000

5000
6000

7000
```

Our **sponsors** help make Advent of Code possible:

Ahrefs - Work on the next general purpose search engine, a world-class crawler, and real big data. Leveraging bleeding-edge hardware and advanced programming technologies. From anywhere in the world. OCaml, ReasonML, Dlang, C++

--- Day 1: Calorie Counting ---

Santa's reindeer typically eat reindeer food, but they also have a special favorite snack that is reserved for St. Nicholas. Each year, the elves work with the reindeer to make special "reindeer food" that is safe to eat. The Elves have brought you a list of the favorite snacks that each Elf likes. For example, the first Elf likes:

To supply enough magical energy, you need to collect a certain number of **fifty stars** by December 25th. The grove has plenty of fruit, but you need to find a way to get it, just in case.

Collect stars by solving puzzles. Each puzzle is a little arithmetic problem; complete the first. Each puzzle gives you a list of numbers, and you need to find a way to combine them to get the target number.



















The jungle must be too overgrown and difficult to access from the air; the Elves' flying sleds don't fly high. However, as you look at the map and compare it to the list of supplies, you realize that you've been given an elf's list. One important consideration: the list only contains **Calories** each Elf is carrying.

The Elves take turns writing down the number of Calories carried by each item in their inventory (one item per line). Each Elf separates their own inventory items by a blank line. For example, suppose the first Elf's inventory is:

For example, suppose the first Elf's inventory is as follows:

- 1000
- 2000
- 3000
- 4000
- 5000
- 6000
- 7000

First hundred users to get both stars on Day 1:

1)	Dec 01	00:00:53		nim-ka
2)	Dec 01	00:00:55		David Robinson
3)	Dec 01	00:00:57		Robert Xiao
4)	Dec 01	00:01:00		sciyoshi
5)	Dec 01	00:01:00		petertseng
6)	Dec 01	00:01:05		Max Jäger (AoC++)
7)	Dec 01	00:01:06		Tris Emmy Wilson (AoC++)
8)	Dec 01	00:01:07		nthistle (AoC++)
9)	Dec 01	00:01:07		leijurv (AoC++)
10)	Dec 01	00:01:09		(anonymous user #1510407) (AoC++)
11)	Dec 01	00:01:09		Tim Vermeulen (AoC++)
12)	Dec 01	00:01:10		dan-simon
13)	Dec 01	00:01:16		hughcoleman
14)	Dec 01	00:01:16		Anish Singhani
15)	Dec 01	00:01:16		Kroppeb (AoC++)
16)	Dec 01	00:01:19		dawninganchors
17)	Dec 01	00:01:21		jhawthorn (AoC++)
18)	Dec 01	00:01:23		mserrano
19)	Dec 01	00:01:25		betaveros (AoC++)

Our [sponsors](#) help make Advent of Code possible:

[Ahrefs](#) - Work on the next general purpose search engine, a world-class crawler, and real big data. Leveraging bleeding-edge hardware and advanced programming technologies. From anywhere in the world. OCaml, ReasonML, Dlang, C++



















Crazy stuff

Advent of Code

- A person solved all 2022 problems in AWK!
 - AWK typically considered as a utility in shell scripts
 - `cat output.log | grep RESULT | awk '{ total += $2 } END { print total/NR }'`
 - Even solved the problem that required an efficient implementation of travelling salesman problem
 - My C++ implementation wasn't fast enough




















Overall Leaderboard — 2022

Advent of Code

1)	3693	 betaveros (AoC++)
2)	3119	 dan-simon
3)	3042	(anonymous user #1510407) (AoC++)
4)	2860	 Antonio Molina
5)	2804	 tckmn
6)	2754	 jonathanpaulson (AoC++)
7)	2707	 leijurv (AoC++)
8)	2704	 5space (AoC++)
9)	2663	 D. Salgado
10)	2609	 hyper-neutrino (AoC++)
11)	2535	 Nathan Fenner (AoC++)
12)	2517	 boboquack
13)	2459	 David Wahler (AoC++)
14)	2447	 bluepichu
15)	2386	 nthistle (AoC++)
16)	2358	 Carl Schildkraut
17)	2320	 Ian DeHaan
18)	2169	 nim-ka
19)	2110	 Robert Xiao

Overall Leaderboard — 2022

Advent of Code

1)	3693	 betaveros (AoC++)	
2)	3119	 dan-simon	
3)	3042	(anonymous user #1510407) (AoC++)	
4)	2860	 Antonio Molina	
5)	2804	 tckmn	
6)	2754	 jonathanpaulson (AoC++)	
7)	2707	 leijurv (AoC++)	
8)	2704	 5space (AoC++)	
9)	2663	 D. Salgado	
10)	2609	 hyper-neutrino (AoC++)	
11)	2535	 Nathan Fenner (AoC++)	
12)	2517	 boboquack	
13)	2459	 David Wahler (AoC++)	
14)	2447	 bluepichu	
15)	2386	 nthistle (AoC++)	
16)	2358	 Carl Schildkraut	
17)	2320	 Ian DeHaan	
18)	2169	 nim-ka	
19)	2110	 Robert Xiao	

betaveros/advent-of-code-2022 | betaveros/noulith: *slaps roof c x | +

github.com/betaveros/advent-of-code-2022

betaveros / **advent-of-code-2022** Public

Watch 7 | Fork 5 | Star 192

Code | Issues | Pull requests | Actions | Projects | Security | Insights

main | 1 branch | 0 tags

Go to file | Add file | Code

betaveros 25 | 91165bf on Dec 24, 2022 | 36 commits

.gitignore	initial commit	3 months ago
22.jpg	22.jpg	2 months ago
README.markdown	initial commit	3 months ago
advent-prelude.noul	oops you need this precedence for stuff	2 months ago
advent.noul	infra improvements	2 months ago
env.fish	infra improvements	2 months ago
p1.noul	initial commit	3 months ago
p10.noul	10	2 months ago
p11.noul	11	2 months ago
p12.noul	12	2 months ago
p13.noul	13	2 months ago
p14-fast.noul	14	2 months ago
p14.noul	14	2 months ago
p15.noul	15	2 months ago
p16.noul	16: fix bug pointed out by /u/Basermannen	2 months ago

About
actually publishing my solutions now that they're not redundant...
Readme
192 stars
7 watching
5 forks

Releases
No releases published

Packages
No packages published

Languages
Shell 100.0%

betaveros/advent-of-code-202 x betaveros/noulith: *slaps roof c x +

github.com/betaveros/noulith

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

betaveros / noulith Public Watch 8 Fork 17 Star 948

Code Issues 4 Pull requests 2 Actions Projects Security Insights

main 2 branches 0 tags Go to file Add file Code

max-sixty Add only (#12) 1fb7338 on Jan 6 192 commits

src	compress/decompress + it's wordin' time	last month
tests	fix ord	2 months ago
vim	why did I name these things unreasonably	2 months ago
.gitignore	fizzbuzz	7 months ago
Cargo.lock	compress/decompress + it's wordin' time	last month
Cargo.toml	compress/decompress + it's wordin' time	last month
README.md	Add only (#12)	last month
index.html	actually wasm, pass envs to functions	7 months ago
worker.js	actually wasm, pass envs to functions	7 months ago

README.md

An attempt to give myself a new Pareto-optimal choice for quick-and-dirty scripts, particularly when I'm not on a dev computer, and to practice writing a more realistic programming language instead of the [overengineered stack-based nonsense](#) I spend too much time on. ([Crafting Interpreters](#) is such a good book, I have no excuses.)

About

slaps roof of [programming language] this bad boy can fit so much [syntax sugar] into it

Readme 948 stars 8 watching 17 forks


Releases

No releases published

Packages

No packages published

Contributors 5





Internet Problem Solving Contest

[IPSC 2018](#)[Archive](#)[Hall of Fame](#)[Training Area](#)[Organizers](#)[Rules](#)[Announcements](#)[Problem Set](#)[My Submissions](#)[Final Results](#)[Registration](#)[Teams](#)

Welcome to IPSC

Internet Problem Solving Contest pushes the boundary of what is possible in programming competitions. The problem set has a wide mix of problems that includes both challenging algorithmic problems and various unusual kinds of problems which will test your outside the box thinking. Every year, thousands of contestants gather to compare their skills, learn something new, and have fun. Will you join us too?

IPSC 2018 is over

IPSC 2018 took place from **6 October 2018, 15:00 UTC** to **6 October 2018, 20:00 UTC**.

Congratulations to **the winners**, and thanks to everyone who participated! We hope you had fun, and we're looking forward to seeing you again next year.

Open Division

Team **♪♪♪**
shik, step5, peter50216 from Taiwan, representing Google / UT Austin / 🐰
🏠

Open Division (individuals)

Team **usagi**
Yui Hosaka from Japan, representing The University of Tokyo

Secondary School Division

Team **FAILED TSTST Day 3**
Benjamin Qi, David Hu, Walden Yan from United States, representing
Princeton HS, Lynbrook HS, Avon HS

Secondary School Division (individuals)

Team **saba2000**
Nikoloz Birkadze (16) from Georgia

IPSC 2016

Problem I – Intelligence report

- **Easy input data set – I1**
- **Hard input data set – I2**

TOP SECRET

Secret assignment number 32250:

Agent,

we have obtained several files from a computer of the person of interest. We believe they may contain stolen passwords for our military control systems. Find out what these files are and recover the passwords from them, so that we may verify whether our passwords were stolen.

BURN AFTER READING

Problem specification

You are given a file. Find out what this file is and recover passwords from it.

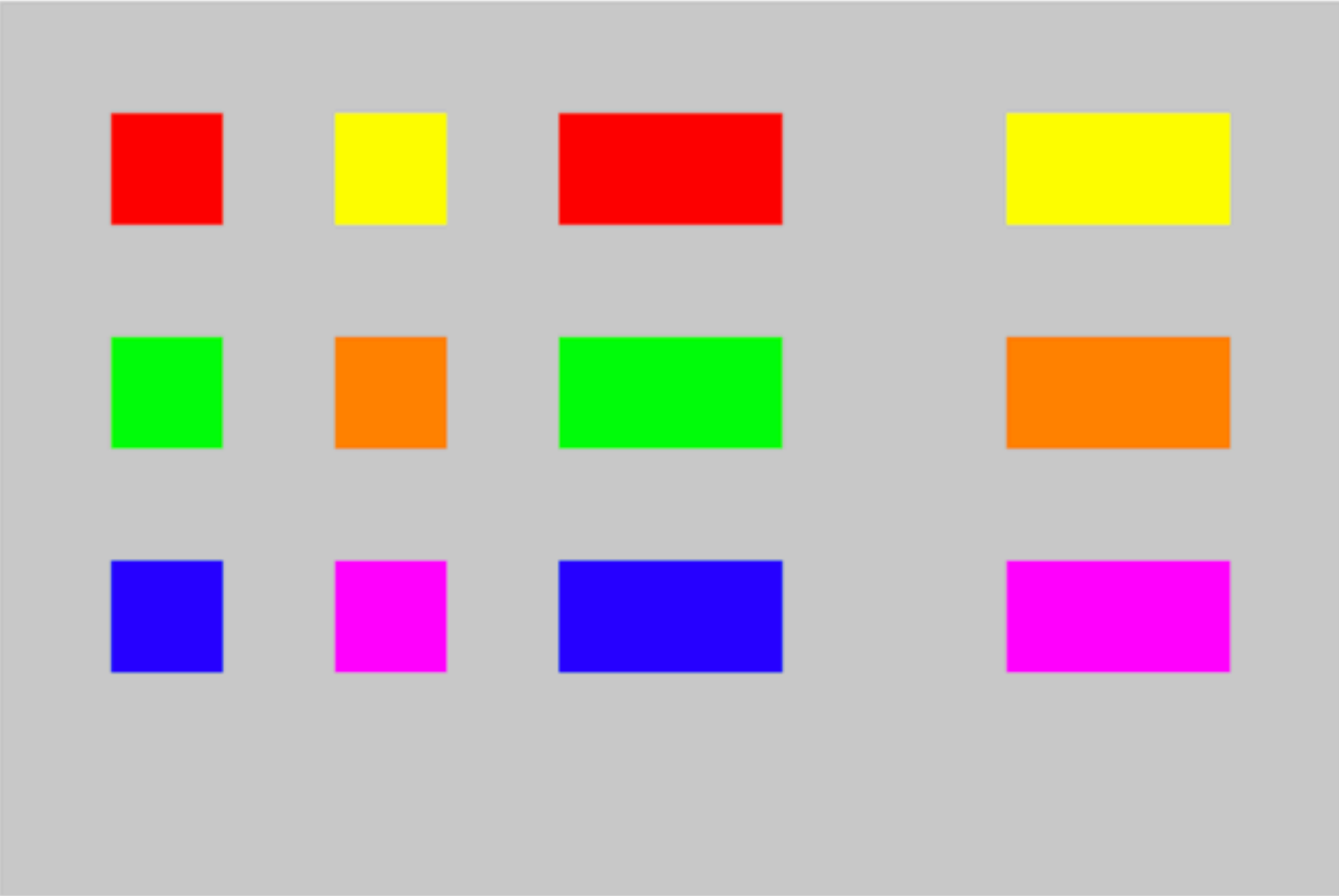
It might be easier to solve this problem using Linux. If you don't have Linux installed, you might want to download and run an arbitrary live distribution. Or you may choose to do something completely different. Your choice of tools is completely up to you. Whatever works.

Output specification

The file for the **easy subproblem I1** contains 10 passwords for test cases numbered from 0 to 9, each one is a 32-character string containing only alphanumeric characters. Submit one file with one password per line. Passwords should be in the correct order.

The file for the **hard subproblem I2** contains a single password. The password consists of 38 alphanumeric characters. Please make sure that your submission contains a single 38-character string, without any whitespace between the characters of the string.

Level 4: Click on the orange square. Then click on the Next button.



Next

Action log:

```
278 69  
next  
483 215  
487 156  
next  
163 263  
next
```

How to be good at competitive programming?

How to be good at competitive programming?

- You don't have to come up with a brilliant idea every time

How to be good at competitive programming?

- You don't have to come up with a brilliant idea every time
- You should know A LOT OF IDEAS
 - be able to quickly correspond a problem with the ideas you know

How to be good at competitive programming?

- You don't have to come up with a brilliant idea every time
- You should know A LOT OF IDEAS
 - be able to quickly correspond a problem with the ideas you know
- Being "in shape"

bitmasks
dfs and similar
probabilities
games
string suffix structures
brute force greedy
fft
expression parsing
ternary search
trees
data structures
chinese remainder theorem
shortest paths
matrices
schedules
strings
constructive algorithms
divide and conquer
math
combinatorics
two pointers
*special
graphs
2-sat
number theory
meet-in-the-middle
flows
dp
binary search
geometry
dsu
hashing
interactive
graph matchings
sortings

How to get involved?

- I'm interested in any sort of programming competition, send me a message and we can team up!
- UBC Clubs:
 - [Algorithmic Contest Monkeys](#)
 - [Maple Bacon](#)

CONTEST TIME



<https://play.typingracer.com?rt=2m6q21kp1c>