Competitive Programming
aka Sport Programming
Programming
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
<table>
<thead>
<tr>
<th>RANK</th>
<th>TEAM</th>
<th>SCORE</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Massachusetts Institute of Technology</td>
<td>11 1339</td>
<td>50</td>
<td>184</td>
<td>25</td>
<td>196</td>
<td>122</td>
<td>166</td>
<td>43</td>
<td>106</td>
<td>69</td>
<td>248</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Peking University</td>
<td>10 1711</td>
<td>71</td>
<td>133</td>
<td>104</td>
<td>274</td>
<td>207</td>
<td>154</td>
<td>48</td>
<td>204</td>
<td>69</td>
<td>167</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>The University of Tokyo</td>
<td>9 1036</td>
<td>128</td>
<td>150</td>
<td>47</td>
<td>64</td>
<td>85</td>
<td>17</td>
<td>166</td>
<td>271</td>
<td>37</td>
<td>177</td>
<td>2 tr</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>Seoul National University</td>
<td>9 1303</td>
<td>113</td>
<td>76</td>
<td>73</td>
<td>286</td>
<td>159</td>
<td>228</td>
<td>11</td>
<td>47</td>
<td>37</td>
<td>130</td>
<td>1 try</td>
<td>1 try</td>
</tr>
<tr>
<td>5</td>
<td>ETH Zürich</td>
<td>9 1541</td>
<td>157</td>
<td>274</td>
<td>86</td>
<td>238</td>
<td>293</td>
<td>22</td>
<td>61</td>
<td>92</td>
<td>158</td>
<td>3 try</td>
<td>3 try</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>École Normale Supérieure de Paris</td>
<td>9 1706</td>
<td>136</td>
<td>293</td>
<td>162</td>
<td>208</td>
<td>256</td>
<td>29</td>
<td>287</td>
<td>43</td>
<td>132</td>
<td>2 tr</td>
<td>2 tr</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Carnegie Mellon University</td>
<td>9 1741</td>
<td>258</td>
<td>64</td>
<td>222</td>
<td>245</td>
<td>183</td>
<td>118</td>
<td>125</td>
<td>148</td>
<td>287</td>
<td>1 tr</td>
<td>1 tr</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>University of Warsaw</td>
<td>8 946</td>
<td>87</td>
<td>43</td>
<td>118</td>
<td>218</td>
<td>24</td>
<td>201</td>
<td>48</td>
<td>127</td>
<td>1 tr</td>
<td>3 trs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>National Research University Higher School of Economics</td>
<td>8 1082</td>
<td>217</td>
<td>91</td>
<td>63</td>
<td>111</td>
<td>189</td>
<td>18</td>
<td>98</td>
<td>185</td>
<td>5 tr</td>
<td>1 tr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>St. Petersburg State University</td>
<td>8 1217</td>
<td>191</td>
<td>38</td>
<td>197</td>
<td>110</td>
<td>18</td>
<td>255</td>
<td>126</td>
<td>144</td>
<td>1 tr</td>
<td>4 tr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>University of Oxford</td>
<td>8 1269</td>
<td>56</td>
<td>165</td>
<td>189</td>
<td>264</td>
<td>211</td>
<td>19</td>
<td>127</td>
<td>118</td>
<td>2 tr</td>
<td>2 tr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>University of Engineering and Technology - VNU</td>
<td>8 1430</td>
<td>70</td>
<td>168</td>
<td>35</td>
<td>207</td>
<td>147</td>
<td>160</td>
<td>2 tr</td>
<td>198</td>
<td>285</td>
<td>4 tr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>KTH - Royal Institute of Technology</td>
<td>8 1583</td>
<td>286</td>
<td>288</td>
<td>91</td>
<td>178</td>
<td>243</td>
<td>21</td>
<td>140</td>
<td>197</td>
<td>197</td>
<td>2 tr</td>
<td>2 tr</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Shanghai Jiao Tong University</td>
<td>8 1566</td>
<td>221</td>
<td>277</td>
<td>118</td>
<td>169</td>
<td>237</td>
<td>55</td>
<td>108</td>
<td>181</td>
<td>3 tr</td>
<td>3 tr</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Final standings

<table>
<thead>
<tr>
<th>#</th>
<th>Who</th>
<th>Points</th>
<th>A 500</th>
<th>B 750</th>
<th>C 1250</th>
<th>D 1500</th>
<th>E 2250</th>
<th>F 2250</th>
<th>G 2750</th>
<th>H 4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Benq</td>
<td>12382</td>
<td>497</td>
<td>686</td>
<td>1214</td>
<td>1428</td>
<td>2070</td>
<td>1926</td>
<td>2108</td>
<td>2453</td>
</tr>
<tr>
<td>2</td>
<td>tourist</td>
<td>12286</td>
<td>491</td>
<td>691</td>
<td>1214</td>
<td>1409</td>
<td>2070</td>
<td>1934</td>
<td>2064</td>
<td>2413</td>
</tr>
<tr>
<td>3</td>
<td>fantasy</td>
<td>11731</td>
<td>492</td>
<td>676</td>
<td>1190</td>
<td>1371</td>
<td>1898</td>
<td>1811</td>
<td>1738</td>
<td>2555</td>
</tr>
<tr>
<td>4</td>
<td>hitonanode</td>
<td>11543</td>
<td>497</td>
<td>684</td>
<td>1148</td>
<td>1395</td>
<td>1998</td>
<td>1775</td>
<td>1976</td>
<td>2070</td>
</tr>
<tr>
<td>5</td>
<td>jiangly_fan</td>
<td>10590</td>
<td>496</td>
<td>729</td>
<td>1024</td>
<td>1380</td>
<td>1970</td>
<td>1618</td>
<td>1730</td>
<td>1643</td>
</tr>
<tr>
<td>6</td>
<td>jiangly</td>
<td>9781</td>
<td>494</td>
<td>681</td>
<td>1202</td>
<td>1380</td>
<td>2027</td>
<td>1898</td>
<td>2099</td>
<td>2070</td>
</tr>
<tr>
<td>7</td>
<td>duality</td>
<td>9715</td>
<td>497</td>
<td>736</td>
<td>1202</td>
<td>1404</td>
<td>2056</td>
<td>1862</td>
<td>1958</td>
<td>1958</td>
</tr>
<tr>
<td>8</td>
<td>maroonrk</td>
<td>9609</td>
<td>497</td>
<td>731</td>
<td>1198</td>
<td>1400</td>
<td>2013</td>
<td>1812</td>
<td>1958</td>
<td>1958</td>
</tr>
<tr>
<td>9</td>
<td>Maksim1744</td>
<td>9510</td>
<td>497</td>
<td>731</td>
<td>1190</td>
<td>1404</td>
<td>2027</td>
<td>1761</td>
<td>1900</td>
<td>1900</td>
</tr>
<tr>
<td>10</td>
<td>-0.5</td>
<td>9453</td>
<td>497</td>
<td>738</td>
<td>1206</td>
<td>1321</td>
<td>1998</td>
<td>1682</td>
<td>2011</td>
<td>2011</td>
</tr>
<tr>
<td>11</td>
<td>inaFSTstream</td>
<td>9340</td>
<td>497</td>
<td>734</td>
<td>1210</td>
<td>1390</td>
<td>1984</td>
<td>1898</td>
<td>1627</td>
<td>1627</td>
</tr>
<tr>
<td>12</td>
<td>Radewoosh</td>
<td>9293</td>
<td>494</td>
<td>619</td>
<td>1178</td>
<td>1380</td>
<td>1912</td>
<td>1804</td>
<td>1906</td>
<td>1906</td>
</tr>
</tbody>
</table>
How it works?
1. Problem
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 \( a_1, a_2, \ldots, a_n \) the geometric mean is defined as:

\[
\left( \prod_{i=1}^{n} a_i \right)^{\frac{1}{n}} = \sqrt[n]{a_1 a_2 \ldots a_n}
\]
\[1 \leq n \leq 10^6\]

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

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

\[ O(n) \]
\[ 1 \leq n \leq 10^6 \]
\[ 1 \leq a_i \leq 10^9 \quad (1 \leq i \leq n) \]
\[ O(n) \quad O(n \log n) \]
\[1 \leq n \leq 10^6\]
\[1 \leq a_i \leq 10^9 \quad (1 \leq i \leq n)\]

\[
O(n) \\
O(n \log n)
\]

<table>
<thead>
<tr>
<th>Sample Input</th>
<th>Sample Output</th>
</tr>
</thead>
</table>
| 5
MindGeek 9
TechBros 15
Web100 8
BestTech 12
gwerty 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;
}
```cpp
#include <iostream>
#include <vector>
using namespace std;

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

```cpp
#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 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>

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

typedef pair<int, int> pii;
typedef vector<int> vi;
typedef vector<pii> vpi;
typedef vector<vi> vvi;
typedef vector<vvi> vvvi;

typed_pair pair164, pair164;
typed_vector164 vector164;
typed_double double164;

typedef struct T

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; }

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<vvc<T> >;

template<class T> T serve(T *v) { return v[0]; }
template<class T> T serve(T *v) { return v[1]; }
using vv = vector<vector<T> >;

using queue = priority_queue<T, vector<T>, greater<T> >;
using pq = priority_queue<T, vector<T>, greater<T> >;

template<class T>
using vpp = vector<pair<T, T> >;

using vector<T> &v = serve(v);
using queue& q = serve(q);

using vector<T> &v = serve(v);
using queue& q = serve(q);

int main()
{
    return 0;
}

using namespace std;

typedef pair<int, int> pii;
typedef vector<int> vi;
typedef vector<pii> vpi;
typedef vector<vi> vvi;
typedef vector<vvi> vvvi;
typedef vector<pair<int, int>> typed_pair;
typedef vector<vector<pair<int, int>>> typed_vector;
typedef vector<vector<vector<pair<int, int>>>> typed_vector_vector;
typedef vector<vector<vector<vector<pair<int, int>>>>> typed_vector_vector_vector;

int main()
{
    return 0;
}
#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
<table>
<thead>
<tr>
<th>elo-mmr</th>
<th>title</th>
<th>division</th>
<th>number</th>
<th>percentile</th>
<th>cf at same rank (spread)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3000+</td>
<td>legendary grandmaster</td>
<td>1</td>
<td>8</td>
<td>99.99</td>
<td>3382+</td>
</tr>
<tr>
<td>2700-2999</td>
<td>international grandmaster</td>
<td>1</td>
<td>37</td>
<td>99.95</td>
<td>3010-3329 (372)</td>
</tr>
<tr>
<td>2400-2699</td>
<td>grandmaster</td>
<td>1</td>
<td>255</td>
<td>99.7</td>
<td>2565-3010 (445)</td>
</tr>
<tr>
<td>2200-2399</td>
<td>international master</td>
<td>1</td>
<td>560</td>
<td>99.1</td>
<td>2317-2565 (248)</td>
</tr>
<tr>
<td>2000-2199</td>
<td>master</td>
<td>1</td>
<td>2089</td>
<td>97</td>
<td>2088-2317 (229)</td>
</tr>
<tr>
<td>1800-1999</td>
<td>candidate master</td>
<td>2</td>
<td>3968</td>
<td>93</td>
<td>1804-2088 (284)</td>
</tr>
<tr>
<td>1600-1799</td>
<td>expert</td>
<td>2</td>
<td>7103</td>
<td>86</td>
<td>1564-1804 (240)</td>
</tr>
<tr>
<td>1400-1599</td>
<td>specialist</td>
<td>3</td>
<td>11003</td>
<td>75</td>
<td>1328-1564 (236)</td>
</tr>
<tr>
<td>1200-1399</td>
<td>apprentice</td>
<td>3</td>
<td>16909</td>
<td>58</td>
<td>1104-1328 (224)</td>
</tr>
<tr>
<td>1000-1199</td>
<td>pupil</td>
<td>4</td>
<td>23977</td>
<td>34</td>
<td>818-1104 (286)</td>
</tr>
<tr>
<td>up to 999</td>
<td>newbie</td>
<td>4</td>
<td>33923</td>
<td>0</td>
<td>up to 818</td>
</tr>
<tr>
<td>Name</td>
<td>Major</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gennady Korotkevich (tourist)</td>
<td>38 + Q×1 10×1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Petr Mitrinov (Petr)</td>
<td>31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Przemyslaw Debias (Psyho)</td>
<td>17 + Q×2 10×1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Makoto Sato (masato)</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tiancheng Lou (ACRush)</td>
<td>29 + Q×1 10×1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tomasz Czajka (tomek)</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Andrew Hu (ecnawma)</td>
<td>16 + Q×2 10×1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bruce Merry (bmerry)</td>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wuhao Du (wdu)</td>
<td>10 + Q×3 10×1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Egor Kulikov (Egor)</td>
<td>72</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jakub Pachocki (meret)</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Streaming
YouTube, Twitch, ...
Streaming
YouTube, Twitch, …
Streaming

YouTube, Twitch, …
Streaming
YouTube, Twitch, …
Let us find $f(L; I)$ for the configuration $X = 3, 5, 5, 7, 8, 8, 8$. Insert the value of each to the left of the corresponding node. For $f = 1$, no line is to be drawn to the left of each node. If each node has a value, insert the number and the value of the node. $f(3, 5) = 3$.

$A f = 3$: The configuration changes to $5, 5, 5, 8, 8, 8$. The line from $3$ to $5$ is closer than $3(2) = 3$.
Let us find $F[i,j]$ for the configuration $i = (1,2,3,4)$. Instead of writing it explicitly, let $F[i,j]$ denote the value of node $x_i$. Initially, let $F[i,j] = 0$ for all $i > j$.

In each node, two values are shown: the number and the value of the node. For this configuration, $F[2,1] = 2$.

For $F[2,1]$, this configuration changes to $(1,2,4,3)$. The base case is shown below: $F[2,1] = 3$. 

Let's consider the subconfiguration $(2,3,4,5)$. We have $F[3,1] = 3$.
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: 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\(^1\) 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\(^1\). 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 boots 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:

```
3000
3000
4000
5000
6000
```

\(^1\)In reality, you’ll probably get much more than enough stars to eat. Not that you’re planning on eating any of these - hrmph.
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 01:00 sciyoshi
5. Dec 01 01:00 petertseng
6. Dec 01 01:05 Max Jäger (AoC++)
7. Dec 01 01:06 Tris Emmy Wilson (AoC++)
8. Dec 01 01:07 nthistle (AoC++)
9. Dec 01 01:07 leijury (AoC++)
10. Dec 01 01:09 (anonymous user #1510407) (AoC++)
11. Dec 01 01:09 Tim Vermeulen (AoC++)
12. Dec 01 01:10 dan-simon
13. Dec 01 01:16 hughcoleman
14. Dec 01 01:16 Anksh Singhani
15. Dec 01 01:16 Kroppeb (AoC++)
16. Dec 01 01:19 dawn inganchors
17. Dec 01 01:21 jhawthorn (AoC++)
18. Dec 01 01:23 mserrano
19. Dec 01 01:25 betaveros (AoC++)
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
<table>
<thead>
<tr>
<th>Rank</th>
<th>Score</th>
<th>Username</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3693</td>
<td>betaveros (AoC++)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3119</td>
<td>dan-simon</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3042</td>
<td>(anonymous user #1510407) (AoC++)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2860</td>
<td>Antonio Molina</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2804</td>
<td>tckmn</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2754</td>
<td>jonathanpaulson (AoC++)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2707</td>
<td>leijurv (AoC++)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2704</td>
<td>5space (AoC++)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2663</td>
<td>D. Salgado</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2609</td>
<td>hyper-neutrino (AoC++)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2535</td>
<td>Nathan Fenner (AoC++)</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2517</td>
<td>boboquack</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>2459</td>
<td>David Wahler (AoC++)</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2447</td>
<td>bluepichu</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>2386</td>
<td>nthistle (AoC++)</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>2358</td>
<td>Carl Schildkraut</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>2320</td>
<td>Ian DeHaan</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>2169</td>
<td>nim-ka</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>2110</td>
<td>Robert Xiao</td>
<td></td>
</tr>
</tbody>
</table>
Overall Leaderboard — 2022
Advent of Code

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

<table>
<thead>
<tr>
<th>Open Division</th>
<th>Secondary School Division</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team shik, step5, peters0216 from Taiwan, representing Google / UT Austin / 🧸</td>
<td>Team FAILED TSTST Day 3 Benjamín Qi, David Hu, Walden Yan from United States, representing Princeton HS, Lynbrook HS, Avon HS</td>
</tr>
<tr>
<td><strong>Open Division (individuals)</strong></td>
<td><strong>Secondary School Division (individuals)</strong></td>
</tr>
<tr>
<td>Team usagi Yui Hosaka from Japan, representing The University of Tokyo</td>
<td>Team saba2000 Nikołoz Birkadze (16) from Georgia</td>
</tr>
</tbody>
</table>
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.

**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”
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