Competitions
Overview
Copilot Opportunities
Design
Development
UI Development
QA and Maintenance
CloudSpokes
Algorithm
High School
The Digital Run
Submit & Review

TopCoder Networks
Events
Statistics
Tutorials
Forums
Surveys
My TopCoder
Help Center
About TopCoder

Dashboard > TopCoder Competitions > ... > Algorithm Problem Set Analysis > SRM 608

Search

TopCoder Competitions

SRM 608

View | Attachments (22) | Info

Browse Space

Added by vexorian , last edited by vexorian on Feb 14, 2014 (view change)
Labels: (None) EDIT

Single Round Match 608
Friday, February 7th, 2014

Archive
Match Overview
Discuss this match

Match summary

This problem set was brought to you by algorithm director **rng_58**. A tough match that put emphasis in mathematical thought. Division 1 coders began having to solve a problem that required some good analysis to reveal the greedy solution. **Petr** solved this problem in seemingly supernatural 4 minutes. The mathematical reasoning was not missing from the graph theory division 1 medium either, **Petr** also had a stunning speed in this problem solving it correctly in under 12 minutes. Many great coders solved the division 1 hard, but **WJMZBMR** did it in glorious 17 minutes. Some similar speed records (second fastest submission in medium, fourth fastest submission in easy), allowed **WJMZBMR** to score a significant division win over **Petr** (2nd place), **ainu7** (3rd), **tomek** (4th) and **mR.ilchii** (5th). Division 2 actually faced similar themes and complexities, with slightly simpler versions of the division 1 easy and medium. The fastest submission in their division 2 problem gave **andyooo** the division 2 win.

The Problems

OneDimensionalRobotEasy | MysticAndCandiesEasy | BigOEasy | MysticAndCandies | BigO | OneDimensionalRobot

BigO

Rate It Discuss it

Used as: Division One - Level Two:

Value	600
Submission Rate	102 / 743 (13.73%)
Success Rate	59 / 102 (57.84%)
High Score	**Petr** for 516.31 points (11 mins 50 secs)
Average Score	305.32 (for 59 correct submissions)

The explanation for the division 2 version explains how to identify unbounded graphs (those in which we need to return -1) and also a proof that also finds a method to calculate the degree of the polynomial in bounded graphs.

Code

The specific division 1 code:

```
static const int MAX_N = 50;
int m;
bool compressed[MAX_N];
int degree[MAX_N];
int g[MAX_N][MAX_N];
int dp[MAX_N];

// finds the path with the maximum number of compressed nodes starting at
// node x in the SCC graph.
int rec(int x)
{
    int res = dp[x];
    if (res == -1) {
        res = 0;
        for (int i=0; i < degree[x]; i++) {
            res = std::max(res, rec(g[x][i]) );
        }
        if ( compressed[x] ) {
            res++;
        }
    }
    dp[x] = res;
    return res;
}

int minK(vector<string> graph)
{
    // Find the SCCs in the graph:
    int n = graph.size();
    int reach[n][n];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            reach[i][j] = ( (graph[i][j] == 'Y') ? 1 : 0 );
```

Alternative solutions and additional comments.

<Place your comments here>

Next problem: OneDimensionalRobot

Author

By **vexorian**

TopCoder Member

Editorial feedback Part 1	**Results:** (42 total votes)
I liked it.	(29 votes, 69%)
I didn't like it.	(13 votes, 30%)

Editorial feedback division 1 hard	**Choose**
I liked it.	
I didn't like it.	

Comments (Hide Comments)

As it's in the Wiki, there's a possibility to improve it. It can be language correction, wording improvement or additional explanation in some parts, your additional comments, description of alternative solutions, etc. If you want to improve the wording of editorial writer or correct some language error, please feel free to put your change over the original text. And if you wish to add a comment or describe another approach, there's a section for this at the bottom of each problem.

Before editing, please be sure to check the guidelines.

You can also add a comment in this comment section. When posting a comment thread, make sure to specify the problem you are talking about.

Posted by vexorian at Feb 13, 2014 15:09Updated by **vexorian** | Reply To This

(j != k) && reach[j][k] && (graph[j][k] == 'Y')
the "reach[j][k]" maybe unnecessary ?

Posted by cepin at Feb 20, 2014 01:49Updated by **cepin** | Reply To This

In bigOEasy, there is a sentence "For each walk of length L−6 that includes vertex 1"..It doesn't make sense to me. Do you mean for each walk such that L-6 >=0? Also, there are a few typos and repeated words in the editorial for this problem. Could you proof read and correct them please?

Posted by arviman at Mar 09, 2014 06:29 | Reply To This

Add Comment