Algorithm Designing with Competitive Programming

MAC0214 - Extracurricular Activities

Undergraduate Vitor Santa Rosa Gomes
vitorssrg@usp.br, 10258862

Supervisor Nathan Benedetto Proença
nathan@ime.usp.br

Professor André Fujita
fujita@ime.usp.br

Professor Denis Deratani Mauá
ddm@ime.usp.br

Professor Kelly Rosa Braghetto
kellyrb@ime.usp.br

INTRODUCTION

Competitive programming has its roots in the scientific study of algorithms, however it focuses on implementing, submitting and debugging them. As a result, it stands out as an excellent tool to learn algorithms, because it encourages to design algorithms that really work, instead of sketching ideas that may work or not [AL].

Through practice of algorithm and data structure designing, this project aims to enrich one's Computer Science formation with a broad variety of carefully chosen real-world problems and their solutions. As a sideproduct, this project also offers to the community convenient and powerful tools to engage and encourage novice forthcomers to this highly specialized field.

METHODOLOGY

The great majority of the problems solved throughout the semester were from classical problem sets on Codeforces [CF], Sphere [SPOJ], HackerRank [HR] and LeetCode [LC] online judging platforms. Those problems were all either upvoted by the community or tailored by company interviewers as essential.

Nonetheless, competing on regular Codeforces contests and on the seasonal Google Kick Start [GKS], Google Code Jam [GCJ] and Facebook Hacker Cup [FBHC] also took place — lively whenever possible, however sometimes virtually.

Overall, more than 300 problems were studied, from which around 230 were sketched and around 70 had a working solution. Information about scheduling, the problems and their contests, some problems' tags and both the tools and compilations produced during this project is available at the <github.com/VitorSRG/MAC0214> github repository.

RESULTS

Initially intended as a personal productivity tool, a practical command line offline local judge application was created. It is not only capable of automatedly compiling, executing and testing any problem, but it also measures and limits the resources available for execution.

Working similarly to the online juges, it can be configured to run every time the solution is saved. In order to accomplish those, the local judge quickly parses special information contained on the programs' block comments, like in Figure 1. Upon execution, it produces an output similar to the one in Figure 2.

Figure 1: Example of problem tagging in C++ and Python

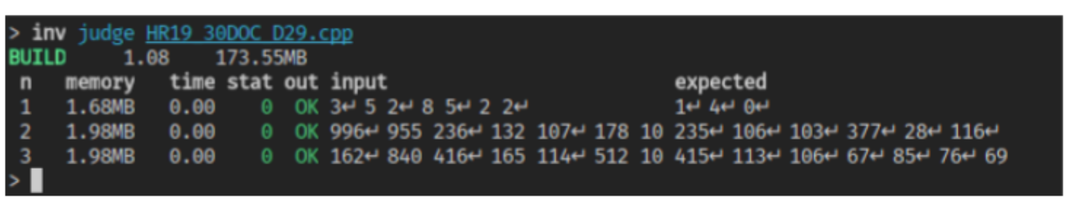

Figure 2: Example of local judge's output

The application can answer queries for problems, ordering the best matches given certain tags. It is important to point out that those tags are far more descriptive than usual problem tagging on the online judges, as they give away all the required insights.

Finally, every code snippet gathered from the reference books [CRLS], [SW] and [AL], alongside optimization details discussed on [CF], [CP] and [TC] forums that were used in any solution were compile under six groups: Numbers, Grouping, Ordering, Graphs, Dynamic Programming and Strings. Since no succinct, complete and optimized implementation of arbitrary-precision Integer, Decimal and Fraction were found, it is advised to use Python 3.5+ for the rare problems in which they are indeed required.

REFERENCES

[CRLS] CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford (2009). **Algorithms.** Third edition. Cambridge: MIT Press.

[SW] SEDGEWICK, Robert; WAYNE, Kevin (2011). **Introduction to Algorithms.** Fourth edition. Boston: Pearson Education.

[AL] Laaksonen, Antti. **Guide to Competitive Programming: Learning and Improving Algorithms Through Contests.** Helsinki: Springer.

[CF] **Codeforces.** Available at <codeforces.com>.

[SPOJ] **Sphere online judge.** Available at <www.spoj.com>.

[CP] **CP-Algorithms.** Available at <cp-algorithms.com/>.

[TC] **Topcoder.** Available at <www.topcoder.com/>.

[HR] **HackerRank.** Available at <www.hackerrank.com>.

[LC] **LeetCode.** Available at <leetcode.com>.

[GKS] **Google Kick Start.** Available at <codingcompetitions.withgoogle.com/kickstart>.

[GCJ] **Google Code Jam.** Available at <codingcompetitions.withgoogle.com/codejam>.

[FBHC] **Facebook Hacker Cup.** Available at <www.facebook.com/hackercup/contest>.