Beginner (Bronze/Silver) December Contest Prep

Srinidhi Krishnamurthy

December 2017

1 General Strategies

1. **Don't Quit**: Even if you are two hours in and you don't feel like you are getting anywhere, don't quit. Just try to keep making progress on the questions.
2. **Partial Credit**: USACO works off a partial credit system where the maximum score is 1000. Typically there are three problems worth 333.33 each.
3. **Brute-force first**: If you can't think of an algorithmic approach to the problem, then apply a brute-force solution to it. If you are looking to save time and know python, then implementing something quick is an option. Contest time is not the place to try out a new programming language.
4. **Solving it by hand**: You should solve the problem by hand first if you don't know how to do it immediately. If you can figure out how to do it, then most of the time you can tell the computer how to do it (at the most basic implementation level).
5. **Look at all the problems before you start solving**: It is a very good idea to see which ones are easiest so you can knock them out first.
6. **Bronze tips**: Just try to get some code going. I find that best for bronze. Working out the problems on paper really helps for this division. If you are getting a TLE that means you are probably doing something horribly wrong in terms of input/output or something lurking in your inefficient code.
7. **Silver tips**: This is where you are going to need to start being cognizant of algorithms and big O notation. I have put the cheat sheet below so that you can use it if needed during the contest.

2 Big-O Cheat Sheet (curated by Charles Zhao)

In USACO, for each test case, you are given 1 second for C++ and 2 seconds for Java. Your programs are run on machines that do approximately 10^8 operations per second. Based on the input size bounds given to you, here are around the complexities your programs should be:

- $N \leq 10 : O(N!)$
- $N \leq 25 : O(2^N)$
- $N \leq 50 : O(N^4)$
- $N \leq 500 : O(N^3)$
- $N \leq 5000 : O(N^2)$
- $N \leq 100000 : O(N \log N)$
- $N \leq 1000000 : O(N)$

3 Algorithm Types

Below are some of the major algorithm types (not comprehensive by any means) that you should be familiar with for the Bronze/Silver Contest. You may want to try to ID the problem type before you start working on the problem itself.:

1. **Implementation Problems**: These problems are generally straightforward and exist to test some basic problem solving/mathematical thinking and understanding of the programming language. If you have taken/are taking TJ APCS these problems will probably be the easiest for you.
2. **Binary Search**: You should be comfortable with the concept of binary search on an Array and know how to implement it relatively quickly and accurately (it is a lot harder than it seems for beginners so you probably shouldn't blow it off if you haven't practiced with it).
3. **Hash Maps / Other Fundamental Data Structures**: You should be comfortable using the different types of sets and maps in your programming language and know the complexities associated with them. Furthermore, you need to be able to see situations where one structure is more preferable.
4. **DFS/BFS**: You should be comfortable with Depth first search and breadth first search problems. This is probably the only "algorithm" that you will encounter.
5. **Recursion** You need to be familiar with the ideas associated with recursion.

6. **String searching/matching**: This is probably the least likely of the problem types but it could happen. Be comfortable with algorithms like KMP and Rabin-Karp. If you are not, you can look these up and come up with your own implementations.

4 Practice Problems / Cram Resources

While it may be pretty hard to cram for USACO, there are some strategies that I have found helpful in the past. Practice is key.

1. **USACO Training Pages**: If you haven't already, this is a good source of some beginner problems.
2. **SPOJ**: The Sphere Online Judge has a nice selection of problems that can be good for beginners (problems tend to be more fundamentally algorithmic here).
3. **a2oj**: https://a2oj.com/ladders has sets of problems for people of all different levels. Its good for getting practice sequentially.

Here are some problems from last year's first contest. If you want to promote, you should probably do it now because it gets a lot harder as the year goes on.

1. **Square Pasture (USACO Dec 2016 Bronze)**: Farmer John has decided to update his farm to simplify its geometry. Previously, his cows grazed in two rectangular fenced-in pastures. Farmer John would like to replace these with a single square fenced-in pasture of minimum size that still covers all the regions of his farm that were previously enclosed by the former two fences. Please help Farmer John figure out the minimum area he needs to make his new square pasture so that if he places it appropriately, it can still cover all the area formerly covered by the two older rectangular pastures. The square pasture should have its sides parallel to the x and y axes. $0 \leq x \leq 10$, $0 \leq y \leq 10$
2. **Counting Haybales (USACO Dec 2016 Silver)**: Farmer John has just arranged his N haybales $1 \leq N \leq 100,000$ at various points along the one-dimensional road running across his farm. To make sure they are spaced out appropriately, please help him answer Q queries $1 \leq Q \leq 100,000$, each asking for the number of haybales within a specific interval along the road.