DISCS PRO 2018 Reflections

```
char author[] = "Carl Joshua Quines";
char date[] = "Mon Feb 19 12:40:27 2018";
```

This is a bit late, and I initially planned not to write reflections for this. But it's good to have everything on paper.

The DISCS is the Ateneo de Manila University Department of Information Systems and Computer Science, and the PrO stands for Programming Open, an ICPC-style contest they organize for both high school and college teams of three. It was held on Saturday, February 3, in Faura Hall.

Pre-round

The evening before the contest, we had some brief discussion on strategy. We generally agreed that our procedure would be to solve a problem theoretically, get someone to verify this solution, and get someone to code this problem. It was preferred that someone else programmed than the person who came up with the solution.

This came up after a lot of discussion. The main bottleneck, we felt, would be computer time, so a mistaken algorithm could be costly. This also implied it was good to have someone typing all the time. It would also be good to look for the giveaway as soon as possible, and to get someone to code it immediately.

We also had some "roles" assigned early on: Dan (Baterisna) was strong at classical, heavy-data structure problems, Kyle (Dulay) was strong at math and clever algorithms, and I would handle team management, aka, shouting at people to do stuff. We also handled preferences: go quickly at first, then go for accuracy.

Morning

On the morning of February 3, I went to Faura Hall. I was pretty early, so I waited for my teammates. Our team name was Kevin Fan Club,[1] KFC: members Dan Alden Baterisna, Kyle Patrick Dulay, Carl Joshua Quines.

Dan arrived first, and we talked about the difference of competitive programming and industry programming. Of course, the conversation went to Robin (Yu) and his abhorrence of `using namespace std;`. Dan agreed that it was sane to use `std`. But then I brought up how I used single-letter variable names, and then Dan was shocked. The guy across us, whom I forgot the name of, totally agreed with me. Called me his spirit animal, even.

There was a lot of waiting for Kyle to arrive. The people from Xavier arrived soon as well, team AXSCII: Alex (Go), Andrew (Ting), Ryan (Shao). We were really waiting for team Francois, who would be Leloy (Cesista), Joan (Francisco), and Nemu (Martinez).

[1] This is named in reference to Kevin (Atienza), a very pro competitive programmer.

KYLE WOULD ARRIVE MUCH LATER, and then we went up and registered our team, and paid the fees. We sat in the lounge-area-ish there, waiting for the round start. Dan printed a bunch of code as reference, since we were allowed some, and Kyle brought a whole stack of paper.

Dan realized that we didn't have a reference for UFDS and how to compile with g++. The latter was necessary since we were going to use PC2 for the competition. Anyway, he quickly wrote these down.

The most important reference we brought in, however, was a drawing of Kevin's avatar. We practiced doing this several times on some scrap paper and eventually managed to get it good enough, so we drew a huge one on a sheet of paper as our flag.

There was a lot of talking. The one I remember most easily is when the Pisay team came, and sir Edge (Angeles) was looking at Kyle with utter incredulity. How could he have betrayed Pisay and entered the contest as an independent? Ah well.

–0:10 – 0:00

Ten problems, three hours. We only had Notepad++, so that was that. Problems were printed on paper, as in actual ICPC, and were given to us facedown.

Vernon (Gutierrez) gave a few instructions. When the round was about the start, he asked everyone, "are you ready?" To which we (and some other people?) shouted "no!" Then Vernon said, "do you want the judge to say that to you? No?"

We took out our drawing of Kevin, and we were trying to attach it to the chair using a pen. We were having a bit of trouble with this, and the team behind us (Ateneo HS's team, if I remember correctly), lets us borrow a rubber band. It works.

We pay our respects to Kevin, and stay silent in supplication.

0:00 – 0:30

The time began. I got rid of the staple and divided the problems equally among everyone: ABC went to Kyle, DEF to Dan, GHIJ to me or something like that. It was quickly identified by Kyle that A was a giveaway, upon which he ran the obvious algorithm to me. (Find $\min\{a_2/a_1, b_2/b_1, c_2/c_1\}$.) I programmed it and it went through, and other teams were doing A as well.

The next problem we tried was F, an arithmetic sequence, which was also an obvious giveaway. I think Dan identified it as easy, and ran the algorithm through us, and we gave him the go to program it because it was not that hard either way. (Output $nx + n(n-1)/2$.)

But it was bugged – it got a WA. It failed even the sample cases, so we fixed that bug, and then it still got WA, and at this point I mentioned that this was a waste of attempts and we should do stringent testing in order not to get penalty. Kyle suggested the extreme cases, Dan fixed it, and then it got AC.

On the leaderboard, there was an early C solve from SandyBrown, and one other F solve from Vanilla Chocolate Thunder. I identified G as an easy problem, ran the algorithm by Kyle, and someone programmed it, and it got AC after three minutes. (Keep adding one and just check manually.)

0:30 – 1:00

That was the end of the easy problems. We take a few moments to recoup. I arrange all the finished problem statements in one pile, with a checklist for the finished problems. On the queue were problems B, which Kyle solved on paper, C, identified by Dan as a giveaway using `map`, and H, identified by me as an ad hoc giveaway.

We decided to do H first, because the other two were heavier. The algorithm for H was to just get the three largest cups and check if the sum of their capacities is at least the total volume. Kyle programmed this.

While we were both busy paper-solving other problems, I overlooked and saw that Kyle was doing a linear scan and storing the three largest so far. I asked him why, when sorting was easier to code, and he just said it was force of habit. Ah well.

Kyle asked us what to do for the vacuous cases $n < 3$. Dan and I pointed out the word "exactly" in the problem, which meant the answer should be no. This confirmed what he suspected, and we submitted and got AC.

Next up was C, and Dan ran the algorithm by me. It really was straightforward: `map` to keep a count of the number of mangoes of a certain color. While Dan was typing, I worked with Kyle to solve problem B.

For the fraction to be an integer, it's equivalent for $n|k^2$. Now the problem is reduced to counting the number of k less than n satisfying this. There exists a smallest k_0 such that if $k_0|k$ for any k, then $n|k^2$. Thus you just need to find the smallest k_0 such that $n|k_0^2$.

This can be done with some prime factorization. Kyle gave a formula in terms of the primes and the exponents, which I verified, it was a bunch of ceiling divisions by two. Dan submitted C and got AC, five problems down.

0:30 – 1:00

I told Dan about B, and he said that one of us should program it while he works on E. I was reluctant too, because I was afraid I might mess up an edge case or something, but Kyle encouraged me, as he was going to work on J.

So I did. I wrote some code to prime factorize n through trial division, then I implemented whatever Kyle wrote down. I checked everything, asked Dan if I should submit, and he said go, and it got WA.

I knew this would happen, I totally called it, but it was no time for panic. We do some unit testing, and I pulled Kyle in to tell me the answer for the cases of primes and semiprimes. The first mistake was that I did not use `long long`, and after that it was still wrong.

We did some more testing, and I wrote a Python program to bash the answer for B using $n|k^2$. And it was working for everything up to some really large number – around 10^6.

The square root. The square root of 10^{12}. The trial division failed because it didn't include any prime factors greater than 10^6. Anyway, we fixed this problem and got AC, great job us.

1:00 – 1:30

We took a bit of time to recoup. There was DEIJ left, and we knew, kind of, what to do for them. Dan told us D was an implementation problem, E was something none of

us could do yet, I was a recursion problem, and J was math.

Kyle said he had a formula for J, empirically, since a while ago, but he hasn't proved it yet. At this point... I kind of forgot what happened. So take the following events to have occurred in some unspecified order.

Dan shows me E and we agreed it was hard, unlike what the problem title suggested. It followed no one, really. I help Kyle prove J for some very limited cases, and we then decide it was probably best to just test it instead of trying to prove it, a proof by AC. It was probably Kyle who wrote the code for this.

On problem I, I found a recursion for the nth side and asked Dan to check it. It was the sum of the terms four and five indices before, I think, verified empirically. I showed this to Dan and asked him to verify it, and he swapped me for problem E, I think.

(Probably-)Kyle submits J and gets AC, seven problems down, DEI left. Dan finds the recursion for I, the sum of the last five terms, and starts programming the matrix exponentiation needed.

1:30 – 2:00

Around halfway through the contest, we had three problems left. Dan was on the computer, I think, and Kyle and I were both working on DE. I took some time verifying how to code D, which was already given in the problem, while Kyle tries to solve E.

Dan finishes the code for I after around ten more minutes or so, and he submits. It gets WA. I call Kyle and get Dan another pair of eyes to help debug his code for I.

The debug was actually quite time-costly. I remember we had to fix a *lot* of bugs. We had to backtrace an error all the way up to the matrix exponentiation function, which was around three levels deep. There were two different mistakes, which canceled out: one of them was not initializing in array, and the other was using an `int` when it should be LL.

Anyway, we did manage to find the bugs, verify with all the cases we could, verify the edge cases work, and then submit. It got AC, but by then the scoreboard was frozen, so no one would know but us. Then Kyle, I think, programs D, while Dan and I work on E.

2:00 – 2:30

Dan manages to think of an algorithm, and runs it by me while Kyle is programming. The problem needed to be solved in $O(N \lg N)$ time.

Dan suggested that to process the input, we use a sweep line algorithm, which was indeed $O(N \lg N)$. If there was a rectangle that wasn't connected to anything yet, add one to the answer. Kyle, I think, very quickly submits the solution for D, and then all three of us were working on the problem.

The sweep line worked for identifying adjacencies, and we work on some cases on paper. Kyle has not heard of sweep line before, so I give him a brief run down while Dan was doing something else, I think.

So, it was decided to start programming it. I was pointing to Dan, but Dan said he just did problem I, and then I suggested Kyle, but he said he didn't even know sweep line, and was not going to code it cold.

That left me. I did not want to do this. But what choice do I have? I had to do it for my teammates.

I started programming the rudiments for the sweep line while Dan and Kyle verified some more test cases. Dan was beside me, helping me code, a live debugger. It took around twenty minutes, I think, before we got something working, tested with a few cases, and it worked for everything we tried, so we hit submit.

wrong answer.

Oops. We go back through all our test cases. We even write a Python program to generate a bunch of test cases, and it gave us the expected results.

Then, Kyle helpfully points out a test case that I thought it would work for – a greater-than sign. It did not work for that case. Kyle saved the day by finding the minimal broken test case. But it was back to the drawing board for us.

It was Dan who made the suggestion of using a graph, but he pointed out we needed a way so that we didn't store all $O(N^2)$ edges. There was absolutely no way you could do a search on the graph without checking through all the edges, which was $O(N^2)$. So we had to find a way to change this.

2:30 – 2:45

We took out the Kevin chair, and I stared at the drawing. I say, loudly, "If we had Kevin, what would he have done?"

"He would have solved the problem by now," Kyle says. We share a laugh, but I think about what he said.

Think Kevin. The limit was the number of edges. Is it possible to not use all the edges? We tried using only one edge in one of the algorithms earlier, but that wouldn't work, if you placed the edge to the lowest rectangle before.

Why didn't it work? It failed the greater-than sign case. Because it didn't reach the rectangle above, it didn't detect it as part of the same component. But we can't just add an edge to the highest rectangle before, because then we would neglect the lowest rectangle.

Aha! What if you added more than one edge, but not all of them – just the lowest, and the highest? I say that aloud, and Kyle asks about the middle, and then I say just add an edge for the middle too.

Dan is on-board with this, but Kyle has doubts, like a good teammate. So we draw some test cases and test it out, and Kyle gave a go signal. And then there were only, like, fifteen minutes left.

2:45 – 3:00

As in, literally fifteen minutes left. Or ten, or some absurdly small number of minutes. Anyway, Dan quickly points out which parts of the code need to change, and then I change them.

I added an adjacency list, a vector of vectors, and Kyle says how to initialize them, and we add the three edges with three `lower_bound`s, and we checked the syntax, and then the graph was built at this point and then we needed to do a bfs.

And then I forget how to do a bfs. Stress does that.

So Dan begins dictating how to do a bfs, and then there were seven minutes left, and then I give the chair to Dan and ask him to finish the code. So Dan programs the bfs, and then he saves, we go to the command line, we compile the code.

Kyle suggests we try some test cases, so I quickly type in all the test cases we could think of, and it doesn't work for one of them I think and we quickly change the code and compiled and tested and tested and it worked and we opened PC^2 and selected the code and the problem and clicked submit.

With two minutes left on the clock.

We waited. The clock was ticking, hearts were beating really really loudly, and we waited, and waited, and then the judge responds.

ACCEPTED.

Now *that* was a team solve.

Lunch

We counted down to round end and applauded when it finished. We packed our things, and it was announced we would have lunch at Gonzaga, so we went. I mean, it was free food. Who could say no to free food?

On the way to Gonzaga, we talk to Aldrich (Asuncion). Dan asked him why he was staring at us a while ago during the round, and he said that he saw we were doing problem E, which apparently, he wrote. He asked us if we got AC in the end, and we said we did. He shares to us the intended solution, which was to do union-find while moving from left to right.

He also shared that, while they did have access to an unfrozen scoreboard, none of them chose to look at it, in order to share in the suspense that we had. Kyle asks for the intended solution in problem J, and Aldrich says he'll have to ask the guy who wrote the problem.

Anyway, we eat lunch, and sadly all the fish fillet was taken so I had to make do with *mechado*. We were one of the last teams to arrive, and we share a table with another team while analyzing our performance.

The guys at the table beside us were from the college division, so I tell Dan that we should try to swap problems with them, because they were still discussing theirs. Dan says I should do it, *pero nahihiya ako*, haha. Anyway, Dan managed to swap the problems, and we looked at them, and they were pretty similar, except they had slightly fewer problems.

After a brief-ish lunch, Aldrich leads us to the guy who wrote problem J, and Kyle asks for the proof. And so he explains Pascal's triangle and the binomial coefficient, yes, but the proof... eludes him as well. He says that we should get back to him, but we never did, oh well.

Awarding

We go back to Faura for the awarding ceremonies, which weren't that big or formal or flashy, typical comp prog. It was giving of certificates and taking pictures, and there wasn't even a photographer from DISCS, which is totally expected.

Several people have heard that we have solved all problems, like the Xavier team. And when they try to praise us, Kyle and I correctly point at Dan as the one who carried the team. Meanwhile, Dan would act bewildered, but deep inside he knows he actually did do the carrying and is just denying it in order to look cute. Kidding.

So we got awarded our certificates of participation, and then the winning teams were announced. Third was SandyBrown, second was AXSCII, and first was us. We

took a picture with the drawing of Kevin that we made, because after all, he is our team's namesake. We are, after all, his fans.

I invite Dan and Kyle to eat out afterward with the prize money, but it turns out they both had to go somewhere. Their parents were there soon after the awarding ceremonies ended, and I said goodbye to them.

Some live editorials for D and I were given out by Vernon, if I remember correctly. I spend this time talking to Aldrich, and we talk about problem-setting, the upcoming NOI, stuff like that. He shares that the hardest part about problem-setting is getting test cases done, which seems right. We talk about several things and then the editorials end, and everyone goes home.

Then I go home.

Figure 1: From left: Vernon, Dan, me, Kyle, Aldrich.

Reflections

Solving as a team is very fun. I initially thought I wouldn't enjoy it, but I actually did, a lot. We were able to solve problems that we wouldn't be able to solve individually, which was great. I also initially thought that I wouldn't want to join ICPC in college, but I think DISCS PRO changed that a little – it's now an option I can consider.

Of course, the team dynamics we had were excellent, and for that I want to acknowledge Dan and Kyle, for being great teammates. Kevin, for guiding us in a metaphysical manner. And of course, DISCS, for the good problems and the free food. It was a lot of fun and I really enjoyed it, so thank you for organizing the whole thing.

Problems

A. *Brian and Artisan Breads.* A recipe for bread requires a_1 cups of flour, b_1 packages of yeast, and c_1 bottles of milk per serving, however, there are only a_2, b_2, and c_2 of each, respectively. Output the maximum number of servings of bread that can be made. ($x_1 \leq 1000$, $x_2 \leq 2000$.)

B. *Brianomial Theorem.* Given n, determine the number of integers $0 \le k \le n$ such that $\frac{(n-k)k}{n}$ is an integer. ($n \le 10^{12}$.)

C. *Brian and the Colorful Mango Lootbox.* Given n mangoes of certain colors, and a list of $k \le n$ colors. On the ith day, a mango of the ith color on the list must be eaten. Output if this is possible for all k days, or the first day that this is not possible. ($n \le 5000$.)

D. *Brian and Dishes of Mushrooms and Mangoes.* A sequence of n numbers is given. For each i from 1 to n: a pointer starts at i. If the pointer is at j, and the entry at $\left\lfloor \frac{j}{2} \right\rfloor$ is smaller, they are swapped. The pointer is moved to $\left\lfloor \frac{j}{2} \right\rfloor$. Repeat until the pointer is at 1.

Each of q queries is an interval. Output the sum of the numbers in this interval after the process. ($n, q \le 10^5$.)

E. *Brian and Easy Non-Math Problem.* Given N rectangles of fixed length and height on the plane. Two rectangles are adjacent if they share a point in their interior or boundary. Output the number of connected groups. ($N \le 10^5$.)

F. *Brian and Fireflies.* The ith term of a sequence is one more than the $i-1$st term. For T test cases, given the first term x of the sequence, output the total until the nth term. ($T, x, n \le 10^5$)

G. *Brian and Gorgeous Numbers.* Given n, output the smallest number greater than or equal to n satisfying at least two of the following: its rightmost digit is 3, its rightmost digit is 8, it leaves a remainder of 3 when divided by 7. ($n \le 10^9$.)

H. *Brian and Hot Coffee.* Each of n cups of coffee have some volume and some capacity. Output if it is possible to pour the total volume into exactly three cups. ($n \le 100$.)

I. *Brian and Isogonal Modular Malls.* A spiral of equilateral triangles is drawn as follows: the first triangle has side length 1, and each subsequent triangle is drawn on the clockwise direction, with side length equal to the current perimeter of the figure in that direction.

For T test cases, output the perimeter after the nth triangle is drawn, modulo $10^9 + 7$. ($T \le 1000, n \le 10^{10}$.)

J. *Brian and the Joke.* For T test cases, output the number of $LOLOL \ldots LOL$ subsequences there are in $LOLOL \ldots LOL$, where the former has R characters and the latter has S characters. ($T \le 10^5$, $R \le 2 \times 10^3 - 1$, $S \le 4 \times 10^3 - 1$, $S - R \le 2 \times 10^3$.)

Results

A plus sign indicates an accepted solution with the number of attempts. So for example, +2 means two attempts. The negative sign is analogous, for solutions judged wrong. VCT is VANILLA CHOCOLATE THUNDER.

#	Name	A	B	C	D	E	F	G	H	I	J	Score	Time
1	Kevin Fan Club	+1	+3	+1	+1	+2	+3	+1	+1	+2	+1	10	724
2	AXSCII	+1	+2	+1	+1		+1	+1	+3	−2	−1	7	410
3	SandyBrown	+1	+2	+1	+5		+1	+1	+2	−1		7	509
4	EXP	+1	+2	+1	−2		+2	+2	+1			6	312
5	Francois	+1	+2	+1			+1	+2	+2			6	399
6	MistyRose	+1	−1	+1			+1	+1	+2	−2		5	228
7	VCT*	+1	−4	+2			+1	+2	+2	−4	−3	5	274
8	ACE	+1	−3	+1			+4	+1	−2			4	276
9	Kiss the Coder	+1	−3	+1	−1	−1	+2	+2	−7	−1	−1	4	299
10	PLDT T1	+1	−4	−3			+11	+2	+2			4	697
11	Strawberry Fries	+1	−6	−1			+2	+1	−1			3	198
12	Team B	+1	−26				+8	+1	−3	−3		3	331
13	One Woman Army	+1	−5	−1			−5	+1	−2			2	150
14	CRESENDO	+1		−3				+1	−3			2	167
15	Virtuoso	+3					−6	+1	−3			2	230
16	Team A	+2					−4	+2	−1			2	245
17	3 in 1	+1							−1			1	21
18	ABC	−12		−2			−1	−7			−1	0	0
18	Group 1	−2	−1				−1					0	0
18	Team 010		−1	−1			−7	−3				0	0