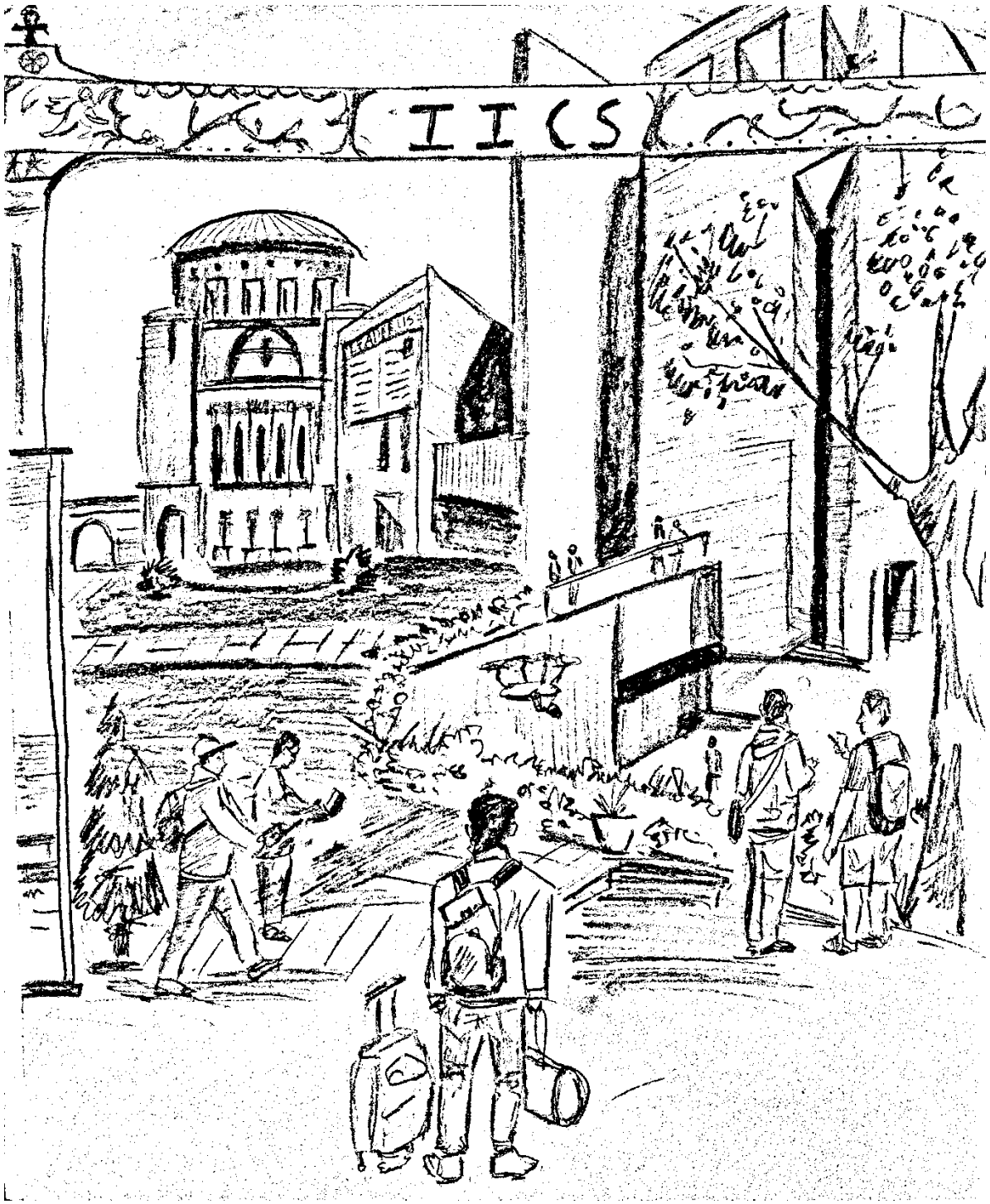

IICS

Introduction

So, it's finally that day, I'm finally joining college.

"Did you pack your books?"

"Yes, Ma"

"Shoes?"

"Done"

"Laptop?"

"Oh!", I forgot my newly bought laptop," I'll get it"

"International Institute of *Computer* Science, you know that means you will need the computer right",

"I know, I know", I said, rolling eyes at my mother's ever-present sarcasm. She pulled me closer and hugged me, tears of pride lurking in her eyes.

"I'm proud of everything you've achieved Jai, and I know you'll achieve much more now, but remember to always stay grounded ok?", she said, still holding me.

"I know, I know", I said, but this time with a smirk on my face. I always value Mother's wisdom but its hard not to smile when she suddenly switches from sarcasm to *hithopadesh.*

I got my laptop bag and came downstairs to find my dad waiting near the car.

"Ready champ!", he said, full of enthusiasm and sparkle in his eyes. Dad always does this when there is a big event about to happen in my life, he always tries to hype everybody up and fill them with self-confidence. We can't blame him though, if the whole household had the energy levels that mother and I have, it would be pretty boring. (No offence Ma, if you're reading this even when I told you not to)

We all got in the car, set for a 3-hour car ride to IICS. After a couple of minutes, Dad started calling all the relatives, to tell them that we had started our journey, and that I had grown up too fast, and some other emotional stuff. He did this so many times by now that I don't even feel embarrassed anymore.

I still remember the last time he did it, when we just got the results of my Checkpoints. Checkpoints are the common exam every high school graduate in the country writes to get into their dream colleges, the only thing I ever thought about for the past 4 years, the only thing everybody in the family talked to me about for the past 2 years. Maybe that's the

reason I remember result day so clearly, otherwise my memory is pretty foggy. I remember some of my cousins came over that day, to celebrate my result. I already had an expected rank in my mind based on my marks, the reason why all the cousins where there. If I really did get those many marks that meant the I had a solid shot at getting into IICS, the one and only International Institute of Computer Science, the first one to do it in the family. The website kept delaying the results, and my cousins took this opportunity to gas me up.

“You know their top student always gets an automatic placement in every big-name company you can think of!”

“You know they have a 100 percent placement record for the past 50 years!”

“Their prescribed laptop is more advanced than what I use for my job!”

“You have to take me to their campus after you join, I want to see the leader board, I heard it’s the size of displays you find in cricket stadiums.”

It almost felt as if they all had researched the college, just for that day. Though, it seemed that for everybody in the outside world (people not part of the checkpoint rat race) IICS was like Hogwarts, too perfect to exist. Even after all that talk about IICS, I was only concerned about not getting less than my predicted marks, which I ended up getting 8 hours after the actual release time.

Surely enough, my cousins made sure to make it a big deal before they left for the day, and I knew that day that IICS was going to be my future college, it almost felt as if there was no other option any sane man would choose. Though, after that day I never really thought about college. I was too focussed on enjoying my new-found freedom during the holidays. Which meant playing football with my mates (yes, I feel British when talking about football, and no its not called soccer) and playing / watching streams of all the video games I missed in the last 4 years.

“We are here!”, announced dad, pulling me back to my surroundings. The college entrance was filled with parents and students unloading luggage from their vehicles, most of them seemed to be freshers like me. It was hard to comprehend what was happening everywhere, but one family stood out. They seemed very orthodox, but that wasn’t the funny part, they had all their luggage unpacked in front of the main gate, and it looked like they were repacking each item after checking it off what looked like a checklist. The boy seemed embarrassed with all the attention he was getting from his would be batchmates but too scared to protest. I couldn’t help but feel a bit sorry for him but things seemed to

die down around him as their repacking seemed to be complete by the time, we had unloaded our luggage from the car.

“It looks like we will have to split up Jaishnav”, said dad, pointing towards a sign that showed two different directions for parents and freshers.

“We’ll take the luggage; you go on ahead”

I nodded in agreement and made my way towards the rapidly thickening stream of students at the entrance gate. We were taken to a crossroad where students divided into batches were led in different directions. The guide for our batch started introducing herself by the time I made it to the front.

“So, once again I would like to welcome all of you to IISC. I know all of you are very excited to explore your future campus. We will be starting with the hostels, your future homes for the next 4-5 years, and then we will be moving on to different parts of the campus”

The hostel tour was good, each room was for two students, which was to encourage interactions (according to the guide) but there were separate think spaces designed for individual working, which didn’t make sense to me. At this point one of the students asked about something called the codespace.

“Patience my friend, we saved the best for the last”, responded the guide lady, smiling ear to ear, as if it was a question she was expecting from the day she knew she was going to be a guide.

After the hostels, we were shown the building filled with all the classrooms and seminar halls. To be fair some of the seminar halls were the biggest rooms I had ever seen, a complete contrast to the tin case classrooms I spent most of my last 4 years in. This building, very uniquely, is apparently called “Class Block”. Apart from the impotent building naming, another worrying thing was the distance between the hostel and the class block. I would like to inform in advance that, given my completely deteriorated fitness thanks to 4 years of zero physical activity, even 500 meters is a significant distance. Regaining my fitness was one of the few things I set myself to do during college. Therefore, before we reached the codespace, the only things that I consciously registered were the sports facilities, especially the football ground. There was nothing special about the ground, the special thing was it was the only ground which was being used at that point of time, all the other sports facilities where empty. This was great! I knew that this didn’t mean that people only played football, but at the very least, it meant that football spirit in the college was strong. Finally, there was one thing in this college that perfectly fitted with my image of what college was.

Then, we reached the ‘codespace’, and to be honest, I really did not get what all the hype was about. The codespace, to summarize, looked like a well-maintained workspace, with some fancy chairs, a coffee machine and a 50-inch TV displaying some kind of points table. Although, the guy who previously asked about codespace and most of the other students were elated by what they were seeing.

“Is that the leader board?”, asked the same guy, pointing towards the 50-inch TV.

“Yes and No, that’s not THE leader board, but it displays what the leader board displays for the occupants of codespace. Luckily enough, our next and last stop is THE actual leader board”

Contrary to the codespace the *actual* leader board lived up to its hype and my cousin’s descriptions. It was the size of a stadium display, made up of multiple large LED panels fixed onto a wall. It was about 50 ft long and displayed 100 names and their role numbers, and finally their points. The marking is 5*2 +8*5+5*10 =100 points. The top 30 ranks seemed pretty spread out, in contrast to the bottom 30 where most of them had the same number of points.

“Just watch, I’ll top this board before the first semester ends”, said the guy standing beside me.” It was the same guy who was asking questions about the leader board and codespace. I forced a smile and walked away; nothing is more repulsive than a person full of himself. Conveniently enough, the introduction meet was in the building behind the leader board. From the looks of it, this whole building was taken up by just the one room, which looked like the seminar rooms I had seen in the class block but much bigger almost three times. Thankfully, the name was Catalyst and not big room or announcement hall. All the students seated themselves randomly, while the dean was on the stage conversing with people in the front rows. Eventually I found myself a seat farther back in the hall, which was my natural habitat.

“Settle down please, we will be starting shortly”, she said, “We will start with professor introductions first”.

One after another the professors came onto the stage from the first row to deliver their speeches. The professors’ introduction speeches, as expected, were repetitive and full of inside jokes most people in the hall didn’t understand. The few people who did laugh either secretly lived with the profs or was just doing it for the sake of it, I was part of the second batch. Though, some profs had good inspiring speeches if the internal jokes were to be ignored. After all the profs, the dean came back on stage,

“Thank you, professors, for those lovely words of inspiration, now I would like to take the remaining time to explain some integral aspects of being a member of the IICS family”, she

paused as though to create suspense and then resumed after a couple of seconds scanning everybody in the room.

“We are a premiere institute, the best of the best, and that means all of you are the also the best of the best, and I believe it is no surprise to you when I say that it takes humongous effort and commitment to reach that level and a lot more to stay there…”

“Does it though”, said the guy in front of me, whose face by now had sadly become familiar, it was the same overconfident dude from the leader board. For some reason, however hard I try, he always seems to end up boasting near me.

“… At IISC we have multiple mechanisms in place to help you stay motivated and focussed to stay at this high level. Though, many outsiders *misinterpret* these mechanisms as measures to ensure competition and not knowledge, but was there ever a better way of learning than the motivation to solve a problem. Has there ever been a stronger push than the hunger to stay at the top, these are all blatant truths that some find hard to accept”

“I don’t, I just find it hard to accept that she is giving an explanation for these things, she should just release the questions and watch as I rocket to the top of the board”, it was him again, this dude was getting on my nerves. Just as I was about to ignore him, I heard a voice beside me,

“Wow, he looks like a pro-coder”

“Pro-coder, more like a pro-retard”, said the girl to my right, I guess she was overhearing our conversation “, even if he is as great as he says, which I really doubt, he wouldn’t ramble about it to everybody in a 500-metre radius of him”.

“I guess she has a point”, I said with a smile on my face, because I just realised that the guy who first spoke was the same embarrassed guy I had seen at the main entrance and his expression was hilarious. He looked flabbergasted by what he had just heard, as if he had just witnessed an unforgivable crime.

“Umm umm maybe, bu… but that was rude”, said the embarrassed guy.

“I didn’t say it to his face so there is nothing rude about it, anyways I’m Sucheta and you both are?”, she asked.

“I’m Jaishnav”, I said

“I’m Rudra”, said the embarrassed guy.

“Nice to meet you both”, she said, and then we all diverted our attention back to the dean’s speech.

“Speech continued… Finally, I would like to put an end to the wait that all of you have had to endure and tell you the release time of the prelim leader board questions is… now”, the screen behind her showed a huge QR code which everybody started to scan with their phones, so did I, to find a website open up with 5 questions. They could have just sent the website to our e-mail, but I guess the leader board was too important for it to be that simple.

“All these theatrics for a bunch of questions … pft”, said the girl.

“Do you know how important the leader board is, it’s what makes this college so special and successful, even you’re CG doesn’t matter if you’re in the top 20 of the leader board…”, said the guy, only to be cut off in them middle, which I’m guessing he has gotten used to by now.

“Exactly my point, how can a bunch of questions about some algorithms or something define knowledge acquired through completing full-fledged courses.”, she argued.

“Umm… can we cut out the leader board talk for now and figure out how to exit this place without bumping into Mr. Boastman over there”, I said, pointing to him standing near the exit.

“Ohh no!”, said the guy.

“So even you don’t like him, I knew it”, she said, and we both laughed at once, leaving him embarrassed, that was a first.

After barely scraping out of the hall, I headed to the hostels. My room was on the topmost floor, and I did the mistake of taking the stairs instead of the elevator. After what seemed like an eternity, I reached my room. The room was nothing special, two beds on the opposite ends, a study table adjoining each bed and some storage beside the entrance. Though, the study table was quite large and the natural lighting seemed great, the main attraction was my roommate. He looked like the stereotype for nerd, just like in the movies. Neatly combed hair, round glasses the size of dinner plates, and a dressing sense even my mother would find old. As if appearance wasn’t enough, he was already on his laptop coding, classes weren’t even starting for at least a week. No, I was being too judgemental, I should probably get to know him before all of this.

“Hi, I’m Jaishnav”, I said hoping he wasn’t everything I had judged him to be.

“Hi, I know who you are, I read the room allocation list on the notice board, and I’m Suhas. If we’re done, I would like to continue trying the leader board questions”, said my new roommate. Any hope that I had about him had vanished, he was exactly what I had judged him to be, a complete nerd.

I didn't have any expectations for my roommate but, if I'm being honest this was a disappointment. Though, not a major one, with football and the other activities that I want to participate in, I didn't think I would spend a lot of time in my room. On the other hand, what the dean said started replaying in my mind

We will provide motivation to remain best of the best.

I silently swore to myself, if this leader board was going to make even college life like the past four years of checkpoint days, then I'm not going to have anything to do with it. There has to be another way to stay at the top and focussed, without having a target rank constantly pressurising my brain. Besides, the way courses are going to be taught by the professors and greater focus on the practical side of subjects should be enough motivation to consistently maintain focus on coursework.

Phase-1

----------ALGO INTRO NOTES----------

Algorithm: An algorithm is a specific procedure for solving a well-defined computational problem.

There are many algorithms to solve the problem, we should choose the best algorithm among the available options.

What is a better algorithm?

- Assuming there are two algorithms, both work in every case and are always correct, Now the best algorithm is the one which takes less time to complete and less space. There are time complexity and space complexity analysis for the algorithm so we can compare algorithms, and choose the better one among those two.
- This works every time assuming that the worst case is given as an input, if the inputs are not random then we should analyze the algorithms for those specific situations and choose and it might be counter intuitive some times.

Types of Algorithms?

1. **Simple recursive algorithms**
2. **Greedy algorithms**
3. **Divide-and-conquer algorithms**
4. **Dynamic programming algorithms**
5. **Backtracking algorithms**
6. **Branch-and-bound algorithms**
7. **Linear programming algorithms**

Others

- **Quantum Algorithms**
- **Brute Force algorithms**
- **Heuristic Algorithms**
- **Approximation Algorithms and many others.**

----------NOTES 1----------

Greedy Notes:

- *Greedy is something which we use in real life without realizing it. Basically, greedy is about choosing an option every time which makes the choice as the best choice in the current.*
- *It is not considered from the point of view of the overall optimization to consider, but every time the choice is only in a sense of the local optimal choice.*
- *So according to greedy locally best choice gives the best globally this is not always true but it's true many times*
- *Like for **example** assume there are 5 shops and at each shop there are 4 choices to take a box of chocolates and you can choose one and you've to maximize number of chocolates you get so if you choose best in every shop then the problem is solved but assume your decision here affects all your options in other shops then choosing best at every shop doesn't mean you made the right choices.*
- *This is intuitive and in most of the cases we don't realize it's greedy because it looks obvious.*
- *But the problem with this is we tend to apply greedy when it won't work, so one of the main things which we need to check is choosing best options locally doesn't mean best option or to put it properly if the problem has independent sub structures then we can use greedy so best options in each of these sub structures means best option overall.*
- *So when it comes to applying it is intuitive we make best options so that we get the right answer one of the most interesting applications of greedy is in graphs an algorithm called **Kruskal's algorithm** for Minimum spanning tree for a graph.*
- *There are a lot more real-life applications and I probably think you will use it in this college many times unknowingly.*

Kruskal's algorithm:

- *There are so many practical applications for MST in real life in various branches. So Minimum spanning tree for a graph is basically a tree whose sum of edge weight is minimum. There can be multiple MST's for a single graph in that case any path is fine.*
- *So according to Kruskal's algorithm take an empty graph and keep on adding edges with small weight and don't add the edges which might form a cycle so adding N-1 edges like that means it's the MST.*
- *To prove correctness, it looks correct intuitively, but we should be able to prove it there are many ways like proof by contradiction or proving the cut property.*
- ***Proving by contradiction:** Assume the MST given by Kruskal's algorithm T formed from graph G is not a Minimum spanning tree but there is R which is the MST.*
- *Starting from the light weight edges assume at an edge E which connects p,q one of R,T disagree (as they are different they must disagree at some point)*
- *Assume T rejects it then the only reason would be it forms a cycle or else it wouldn't so this can't be the case.*
- *Only possibility is T has E and R doesn't have so for R to be a MST there must be an unique path from p to q and there is no E so there must be E1 which is from p to q and has weight more than E (if it's weight was less than it would've been in T) so here we get a contradiction that with E1 as an edge R can't be a MST so we can say MST which we get through Kruskal's is MST.*

Dijkstra's Algorithm:

- *The problem here is to find the shortest distance from source to all the other nodes. So we take an array and store the distance from source s to each node in it. Initially we initialise it to a large value (the reason will follow shortly) and Dist[s]=0.*
- *The idea here is from the source to all the adjacent nodes we can say the shortest distance is the edge weight, now from the shortest distant vertex if we find the distance to its adjacent vertices and update it so if we do this to all edges and update it if the new distance to it is less than the initial one that's the reason why we initialised with large values so we can get minimum quickly.*
- *This process is called relaxing and we do it like this:*

```
if (dist[v] > dist[u] + weight){
        dist[v] = dist[u] + weight;
```

```
            pq.push(make_pair(dist[v], v));
        }
```

- *Here pq is priority queue which is defined in a way so at the top we have lowest value at top so for each smallest length we find we go through it's adjacency list and update if we have chance if it's lesser*
- ***Proof** can be easily done by induction.*

Every assumption I had was heartbreakingly wrong. Motivation was somewhat of a scarce resource, after 1 month of college life this was one of the most widely accepted facts. It seems many of my fellow batchmates were shocked like me at the current state of events, and its huge contrast to the picture painted in our minds about college. Out of this pain obviously came the analysis of the situation, and many reasons were found. To list some, college treats us like adults when it comes to managing studies, which coincidentally is the only area of life where many of us lack maturity. The professors do not use divine methods to teach their courses, and their assignments are the only thing close to a cosmic entity, namely hell. Coming to assignments, they are the single most quoted reason for lack of motivation and self-driven learning, if not the difficulty of the assignments, the quantity itself is quite demotivating. I don't remember a day when we had no assignments pending, I think the minimum was two assignments. With this never-ending sea of assignments, it is difficult enough stay afloat let alone try to try to ride the waves and explore the courses and get motivated by their scope.

In all this grimness, the only good part of college life was the part that had nothing to do with academics, it was friends, games and movies. Another common phenomenon in college was the formation of friend groups. Most groups formed are quite natural, like ours, while some were formed based on rooms. Though, none of the so-called friend groups are official, it is a known fact. While there are always inter-group interactions, most social interactions that take place are intra-group ones. For example, my group, Rudra, Sucheta and I, almost hang out together the whole day except for the times when Sucheta goes to her hostel or one of us goes to play or for some club activity.

One of the only things that have gone as expected in college was my participation in games. I have played football almost every day and tried out new sports like table tennis and volleyball. Sport is also one of the most regular things that I have done in college, apart from maybe hanging out with Rudra and sucheta. Which makes any sport event on campus one of the most looked forward to for me. There is one major thing to confess here, compared

to most of the top players on campus, my fitness levels are still puny. Though, I have made a lot of progress, which gives me hope for getting selected in the try outs coming up for the college team.

Though our schedule is tightly packed with deadlines, we somehow always manage to make time for movies. I have watched more movies in the theatres in this one month than in the past 4 years put together!

While, this side of campus life was running smoothly, the listening to classes part of it wasn't doing great. Losing interest in classes in just 1 month sounds much worse than it feels, and like every other disappointment in the academic department there is an analysis for this phenomenon. Over the past four years, every teacher we've had had hyped up the 'elite' college way of teaching whenever the chance arose. After all that, we ended up getting points read out of slides and the same old syllabus based material coverage. Though, there was one professor who stood out for most of the batch due to the tone in which he taught the course. It was almost as if he were teaching a bunch of kindergartners about the space, stars and beyond. He always taught even the simplest of things with an air of suspense and curiosity. It would be fair to say that h was the only one that came close to the hyped version of the professors we were brainwashed about. I have just realised that I have written notes for his class in this book itself, well seeing that his is the only class I right notes for, there will be more notes scattered around the book, I guess.

Now, finally coming back to the reason I have started writing after one month, I have realised that academics have not been going as panned and I flipped some pages to read what I had written when I joined. Doing that made me realise that, I joined this college with many rigid thoughts and ideologies and maybe that led me to be a lot more judgemental about things that I had seen. Though, I wouldn't say that all my previous conclusions were completely foolish, some of them needed changing for me to rescue my academics. Therefore, I have resolved to give the first phase of leader board a try, not for the points or the rank, but just for the additional interest the questions may generate to help me perform better in my academics.

"I though you didn't want to try the leader board and explore different methods to stay motivated?", asked Rudra when I told him about my decision.

"I know, but you know how that's been working out for us so far, I won't do it for the rank, I'll just do it to check how the questions, maybe I'll get interested in one of them and get a spark, who knows", I replied.

"True, better to try know than to regret later right, if you find a spark do share it with us Optimus prime", said Sucheta smiling almost in a mocking fashion.

“Stop mocking us, Rudra you were right man, she is always mocking us because her courses are going well”, I said, lighting a fire in real time.

Sucheta jumped and looked towards Rudra, with a completely serious expression,

“Did you now?”, she asked.

“No, no I never said anything like that, he’s lying don’t take him seriously. Jaishnav man why do you do this to me.”, he said, looking all pale, but I wasn’t done yet.

“What did I do, didn’t you say that she was mocking us when you visited my room for the first time?”, I asked.

“I didn’t say it like this, no, no he’s lying!”, said Rudra, already up and ready make a run for it.

“Sooo you did say it, death, this is the sentence”, she said with her best impression of a psychopath and started chasing after him.

Meanwhile, I started laughing, pitching them against each other is something I specialize in and the entertainment value never seems to deteriorate. Ever since the three of us first met at the introduction meet, Rudra was always scared of Sucheta because of how spontaneous and direct she was, and once Sucheta got to know that (through me 😉) she started using that to her advantage to toy with him. Normally their chases lead to a compromise where Rudra buys some food for Sucheta and eventually Rudra ends up in his room sulking at the money loss. So, I got up after I finished my maggi and headed towards his room. As expected he was sulking on his bed, pretending to read notes.

“Are those Algo notes?”, I asked, sitting beside him.

“Yeah, you know that is already 200 rupees spent on compromises with Sucheta.”, he said, setting aside his notes.

“Really, and how much did you spend this whole month 10 20 rupees?”, I asked.

“50 rupees”, he said.

“And how much is your monthly pocket money?”

“2000 rupees but…”, he said slowly understanding where I was heading.

“No but, if you don’t even spend 10 percent of your pocket money, don’t you think your parents will think something is wrong.?”

“Why would they think something was wrong?”, asked Rudra, genuinely confused.

“You won’t get it, let us change topic”, I said, picking up his notes, “Why do you write notes for every class when all they do is read from their slides which are already shared?”, I asked.

“Because I am supposed to, why do you write notes for the algo class, you could just use my notes right”, said Rudra.

“Because I like his class and he barely ever gives actual notes, most classes are spent on discussions on can we do better and also I don’t want to waste time deciphering your handwriting”, I said grinning as I handed his notebook back to him.

“Mine is better than yours, you know that”

“Sure”, I said, grinning with sarcasm as I left his room. Though, I doubt he understood sarcasm.

I headed back to my room; it was time to try out the leader board questions.

----------QUESTION 1----------

Q1) We have a fixed denomination of notes available. They are Indian currency 1,2,5,10,20,50,100,500,2000 we get money. We have to exchange it such that the number of notes involved here are very less i.e minimize the number of notes and return the number of notes exchanged and print the number of notes of each type used space separated starting from 1 to 2000 if you didn’t use it then print 0.

So basically we have to give the change with less numbers. If he asks change for 501 we have to give 500 and 1 instead of all other possibilities.
I started reading the test cases. The first one was 1201 and the answer was 5 and on the newline it’s 1 0 0 0 0 0 2 2 0.
It looked easy so I have to start to consider right from highest denomination and see if I could use it, If I can then I must or else it can’t be the minimum so doing it normally, just the way we count money If I keep on subtracting the amount as I pick a note then it should work, the moment where I can’t pick anymore notes of that denomination it means I must use them and print them so this should do. I quickly coded it and it looked like this:
*Here note is an array consisting {**1,2,5,10,20,50,100,500,2000**}. Money is the input.*

```
for (int i=8;i>=0;i--){
    if(money>=note[i]){
        int freq=0;
        while(money>=note[i]){
            freq++;
            money-=note[i];
        }
        cout << freq << " ";
        ans+=freq;
    }
}
```

It was working after fixing the order but then it struck to me that we subtract generally because we want to be sure after dividing, here there is no point to do that here on a system instead of subtracting if I divide it then it would be better, so I edited the part where I used a while loop to subtract every time to this:

```
int freq=money/note[i];
money-= note[i]*freq;
cout << freq << " ";
ans+=freq;
```

Or I could even write money as money = money % note[i] as money will be less than note[i] anyway here. So it would make this as the final one and it was accepted

---

----------NOTES 2----------

Divide and Conquer:

- *The whole idea of divide and conquer is breaking the question into two halves and solving the parts. This gives us new perspective always like assume searching a word in a dictionary we start from middle or first or last we don't start from first always right.*

- *There are a lot of algorithms using this concept and they are fast generally compared to the original naive method just like searching from different places to find meaning in a dictionary instead of starting from the first page.*

- *There is an interesting question to understand this concept in assuming you are on an island. There are 8 people and except one everyone weighs*

the same. We have to find the person using a see-saw with as few trails as possible?

The first answer would be 7 trails like fixing one and trying with all other 7 if 6 are equal one is heavy then that person is odd if all are unequal and light then the chosen person is heavy so in worst case we may feel we need 7 trails but we can do it even quickly

- *here's how we can solve the problem quickly:*
  - *Divide them into 2 groups of 4 each and weigh each group with other one of them will be heavy then choose the heavy group and divide them into groups of 2.*
  - *weigh them one of them will be heavy again then of the two if we weigh them against each other than we are done one of them will be heavy and he is the answer .*
  - *Basically in 3 trails we found the answer which is big improvement compared to 7 now if there were 16 people and one is odd among them then using divide and conquer approach we can do it in 4 trails compared to 15 .*
  - *Basically for very large numbers we can appreciate this because when there are 1,048,576 people in the island you just need 20 trails to find the odd!*

    One of the most famous algorithm which uses divide and conquer is Binary search. This we use when searching a word in a dictionary, we just don't do it with discipline and do it unknowingly. So when the question is finding an element from a sorted array.

Binary Search:

- *The naive method would be starting from one side and breaking from the loop once we found the value so in worst case it takes O(n) time complexity but we can do better as we did in the island question .*
- *The main idea being as the array is sorted we can say for sure it belongs to one half not the other so searching the other half for the element is pointless so we can extend this idea so we compare the value we need to find let it be X.*
- *We compare X with the middle element of array if it is less than or equal to the middle value then we may find it in the left side of the array if it's more than we may find it in the right side so we repeat this to those halves too just like the island problem we do it until we find one so if it's same we found it or else we didn't. We used this many times without recognising it.*

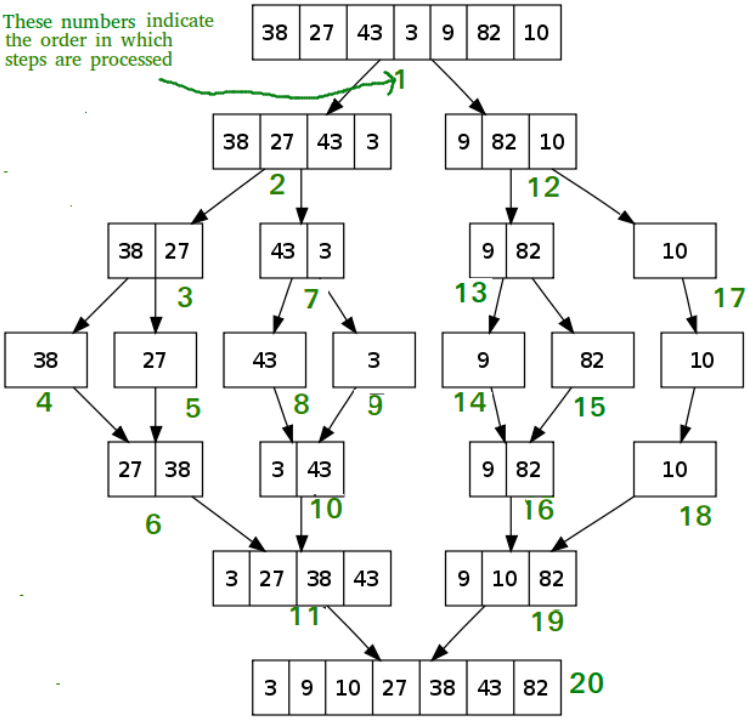
- *The major advantage of this is same as island problem in minimal trails we found the odd one here also with minimal operations we found X.*
- *Initially there would be a for loop and we check at every instant if we found X or not this takes O(n). Now we just check with middle elements of each part so if there are 8 elements 3 searches would do for 16 4 so it takes O(logn) here base of the logarithm is 2.*

- *Again we made it an O(n) algorithm into O(logn). As we already saw how small logn is compared to n when n is high so this is so fast compared to naive algorithm and this solved many problems binary search is used in so many places today.*
- *Other interesting algorithms which use divide and conquer for sorting are merge sort and quicksort.*

Merge Sort:

- *The idea of merge sort is two divide arrays sort them then combine them by checking order.*
- *There are many sorting algorithms bubble sort, insertion sort and others most of them take O(n*n) time in the worst case so using divide and conquer approach we can do better.*
- *According to idea,*
  - *we take an array divide it into halves sort the halves*
  - *combine them so we extend this idea until we get single element*
  - *Then we have arrays of size 2 then we combine 2 2's we get 4*
  - *We go on combining like that assume merge function merges two sorted arrays such that it's still sorted then the function looks like:*

```
void mergeSort(int arr[], int l, int r)
{
    if (l < r)
    {
        int mid = l + (r - l) / 2;
        mergeSort(arr, l, mid);
        mergeSort(arr, mid + 1, r);
        merge(arr, l, mid, r);
    }
}
```



This image shows how it happens

- *Analysing this merge function is O(n) as it checks compares and adds linearly to array such that it's sorted from the two sorted arrays so we can write it as*
  *T(n) = 2*T(n/2) + O(n)*
- *This falls into case 2 of master method so it's best case worst case and average case time complexities all are O(n*logn) which is way better than initial O(n*n).*
- *This was also so innovative because of the lesser time it takes it is also used widely to sort as it's a guarantee that you would get a sorted array after c*n*logn operations.*

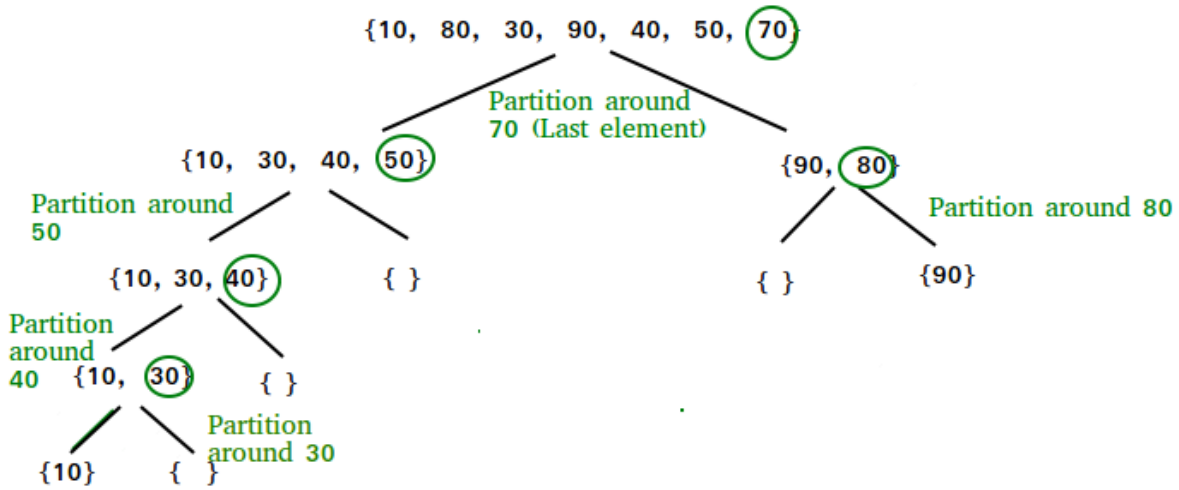
Quick Sort:

- *The idea here is basically choosing a pivot for the array and making sure that on the left all elements are less than the pivot and on right all elements are more than the pivot so effectively we found the correct place for pivot like it will be at the same position in the sorted array so we keep on quick sorting the arrays by fixing pivots position every time.*
- *There is a function which we write called partition which returns the correct position of pivot so we quicksort the left and right partitions of those recursively.*

- *So at last we get an array where every element is at it's position such that all elements at left are lesser than it and all elements to right are more than it. Basically that is called as a sorted array.*
- *Here we have options to choose pivot. There are many like choosing one in random or choosing the median as pivot or choosing the last element as pivot or choosing the first element.*
- *To choose median as pivot we need to find median first which is again extra calculation so random or last or first can be used anything works the code looks like this here partition function fixes the pivot at correct position with left elements less than X and right elements more than X and returns the position of pivot.*

```
void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);  //left half
        quickSort(arr, pi + 1, high);  //right half
    }
}
```

- *Time complexity analysis for quick sort each time we find pivot and call the same function twice with halves of different sizes and the partition function takes O(n).*
  So we can write T(n) = T(n-c) + T(c) + O(n)
- *If the pivot is always smallest or greatest then the case would be T(n)=T(n-1)+0+O(n) so here it would be O(n*n) if the array is in descending order.*
- *If the partition function chooses middle element as pivot always then T(n)=2*T(n/2) + O(n) so it would be O(n*logn) as it's similar to merge sort.*
- *In an average case it would be O(n*logn) which can be done using finding time for each possible permutation of the array which takes a lot of time.*



Fast Fourier Transformation (FFT):

- *This is used for polynomial multiplication, which is common and takes O(n*m) time generally where n and m are degrees of the polynomial so we can write it as O(n*n). Can we do better? Yes there is a way.*
  - *So each n degree polynomial can be uniquely represented by n+1 points on the 2D plane.*
  - *Let A(x),B(x) be the polynomials we need to find C(x)=A(x)*B(x). So A(x) can be found if we know $A(x_0),A(x_1),A(x_2),\ldots.A(x_n)$*
  - *Then we can find A(x) it's called interpolation. Using this we can say $C(x_0) = A(x_0)*B(x_0)$ so if we know 2n+1 points we can interpolate C so we need to find A(x)*B(x) values at 2n+1 points*
  - *And now interpolate but still this also takes O(n*n) but we have an advantage here if we choose the points in some better way then we are done so that it won't take O(n*n).*
  - *If we choose complex roots of unity as the points to find C(x) at those now to find the product of A(x)*B(x) in a quick way.*

$A_0(x) = a_0 + a_2x + a_4x^2+\ldots..$
$A_1(x) = a_1 + a_3x +\ldots..$
$A(x) = A_0(x^2) + A_1(x^2)$
And the next are solved in the same way so we find product of A(x) and B(x) at 2n points in nlogn time
Then from that we interpolate and find coefficients of C.

“So, how was the first question?”, asked Rudra.

He seemed more focused on the leader board questions than I was.

“It was ok, easier than I expected, given all the hype about it on campus”, I replied.

“And you say I’m the smart one! Liar”, said Sucheta, she never leaves a chance to have a go at anybody.

“I know why you won’t attempt it, you’re scared that once you start you’ll top the board and the world will get to know about you’re genius”, I said.

“Yeah yeah totally. By the way, why aren’t you trying the questions if you are so interested?”, she asked Rudra.

“I’m scared”, replied Rudra.

“Scared! What are you scared about!”, I asked, completely confused.

“I don’t want to get stuck halfway and make a fool out of myself, one of the seniors told me it is better to build my resume rather than waste time on the leader board because only the displayed ranks have significance”

“Wow, if you put so much thought into actually solving the questions, top 100 wouldn’t be too far”, remarked Sucheta.

“Very funny”, grunted Rudra, using what was evidently sarcasm! I might have underestimated his social skills after all.

“Are you mocking me Mr. Rudra”, said Sucheta, and then started laughing, unable to hold her interrogation face. I joined in, leaving Rudra at his natural state of embarrassment.

“Anyway, the debate and quiz club event is at 5 pm today, do you want to come rura?”, I knew Sucheta wasn’t into debate so didn’t bother asking.

“If he’s coming I’ll come too”, said Sucheta to our amazement.

“But you don’t like that stuff right?”, asked Rudra.

“It’s ok, are you going or not?”

“If it means all of us going to an event together, then sure”, said Rudra.

“Aww, our very own senti”, said Sucheta, and we inevitably resumed laughing, this time Rudra joined in a bit too.

The event debate and quiz club were organizing was called Survival Mode. It was basically a team event, each team consists of three students, and the teams compete in a quiz of

different topics each round, where the team with minimum points in each round will get eliminated. The process continues until there is only one team left. Since it was 3 members per team, the three of us would obviously form one, Sucheta's general knowledge wasn't something she was very proud of, but her movie knowledge might come in handy, Rudra might be of some help if all the topics were from the textbooks he read since he was born. That left rest of the topics to me, which was a scary thought. My knowledge of sports and world politics was decent, other than those any information I had of the other topics was foggy. Just as I was thinking that we would have a decent chance, Sucheta tugged at my shirt and pointed towards the board in the room.

"What?", I asked.

"Read the third point in the rules thing", she said.

I started reading it aloud,

"Teams will be chosen randomly, What! That's stupid", I exclaimed, I had no idea about that rule.

"So much for participating together, I'm leaving", said Sucheta turning towards the exit.

I stopped her,

"Hey, you never know we might fall in the same team, you've come this far, wait till they announce the teams"

"Ok", she said.

After a while they started announcing the teams. They were calling up each team's members onto the stage and sending them to different rooms. This meant that once called up, leaving was not an option. Sucheta seemed to have realised that too.

"If we don't fall in the same team, you will regret it jai…",

"Sucheta!", her name was announced.

Sucheta glared at me and left for the stage. If she really didn't end up with the both of us, I pity the team that got her, because I doubt, she will be able to hold her anger throughout all the rounds.

"Rhuthik J", the second guy in her team was announced, this was bad.

"and Rohan P", that was the third guy. I could clearly see her glaring at us from the stage, I couldn't help but smile back. Though it was unfortunate that we didn't fall in the same team, her expression was priceless.

Though, Rudra and I ended up in the same team with a new guy called Jeeshanth, which I knew would only result in a bigger scolding from Sucheta after the event but that was secondary. This Jeeshanth guy looked familiar, but he wasn't someone I met on the football field or during other club meets. As we were walking towards our designated room, Rudra whispered,

"It's the same guy we saw on the first day during intro meet", that was it. This was the same guy that was boasting about his coding proves. That was irrelevant now, I moved towards him to introduce myself,

"Hi I'm Jaishnav"

"Hi, I'm Jeeshanth, and you are", he asked, turning towards Rudra

"I I'm Rudra", stuttered Rudra, his social anxiety seemed to be kicking in again.

"We'll have to be much faster than that if we want to win this", he said. There was still that hint of overconfidence in his tone, but it was much lower than what we heard on the first day. I was still a bit irritated, but keeping the competition in mind I hid it with a fake smile.

As the quiz progresses, after each round we were transferred to another room and pitched against another team. This meant that we never knew how close we were to the final because the total number of participants was never disclosed and also a lot of travelling because we were progressing a lot more than expected. The reasons being, most of the rounds were, luckily, on world politics which was maybe because of the fact that the same club also held the MUNs (Model United Nations). Then any other famous statistics about the Indian government was answered by Sucheta, as expected. The Jeeshanth guy was also crucial in rounds where the topics were coding related. Judging by his confidence while answering, it seemed as though coding was the only thing he did in his life. He also made it a point that his contribution was acknowledged, giving the both of us pompous looks after any such round.

While we were travelling again, for what may have been the 100th time, to another room one of the organizers intercepted us,

"Congratulations, you've made it to the finals! This round will be in front of a crowd so get ready.", he said.

I was happy and anxious at the same time, that was a first, making it to the finals was great but quizzes are memory heavy competitions and my memory doesn't function very well when I'm in front of a crowd.

We entered one of the seminar halls, to a large audience of students, eagerly clapping and smiling, as if waiting for one of us to mess up and give them a chance to laugh their hearts out.

“Let’s go!”, said Rudra, to my astonishment. I assumed that he was even more petrified than me, but it seemed he was too pumped about reaching the finals to acknowledge the crowd. His energy helped calm myself as we took our seats on the stage.

“The final round is going to be an amalgamation of all the rounds you have been through so far. Each question will be from different topics you have faced so far and both the teams will have to answer the same question with the help of a buzzer. But be careful about the buzzer, if you press the buzzer first and don’t come up with any answer, that team will be penalized with a point deduction. There will be 15 questions in total, with one point for each correct answer. Finally, you have 20 seconds to decide to press the buzzer and 10 seconds to answer after the buzzer is pressed.”, announced the organizer.

The final round was close to some of those TV shows where the participants play as if their life depended on it. At the beginning our score and the oppositions were pretty close where the lead kept changing sides after 2 questions. Though after 10 questions or so, when both our scores where even, the opposition managed to get 2 consecutive questions right. This meant that we had to get all the remaining questions right to win.

I could see Rudra ready to press the buzzer for the next question, I tried to tell him to be cautious, but he wasn’t paying attention to anyone except the question. As soon as the question was displayed Sucheta pressed the buzzer. The question was neither in my domain nor a coding related question, which meant Sucheta should have the answer. But he didn’t know the answer for the question,

“Umm, umm .. “, stuttered Sucheta, he had no clue, I was going to urge him to say any random answer to avoid the point deduction, but I was too late.

“Time is up! We have a winner! “, announced the organizer.

We were congratulated for making it to the finals and then we left. As we were leaving the hall, Jeeshanth intercepted us, I though he left early but he was waiting for us near the exit.

“We lost because of you stutter guy”, he said moving towards Rudra,

“Hey man back off, it’s just a game, relax”, I said putting myself between him and Rudra.

“Whatever… I don’t like losing like you lot”, he said and as he was turning to walk away he murmured,” Bloody losers”.

I was furious, and about to confront him, but Rudra held me down and Jeeshanth disappeared into the crowd.

----------NOTES 3----------

Dynamic Programming Notes
DP is not a specific algorithm, but a technique like Divide-and-Conquer. DP was developed back in the day when 'Programming' meant 'Tabular Method' like Linear Programming.
DP is a technique for problems with optimal substructure and overlapping subproblems: Optimal solution to a problem contains optimal solutions to subproblems. This does not necessarily mean that every optimal solution to a subproblem will contribute to the main solution.

- *In Divide and Conquer, subproblems are independent so we can solve them in any order.*
- *For Greedy Algorithms, we can choose the right subproblem by a greedy choice.*
- *In DP, we solve many subproblems and store the results (all of them won't contribute to solving the larger problem). Because of optimal substructure and overlapping subproblems, we can be sure that at least some of the subproblems will be useful.*

There are a lot of algorithms using this concept and they are fast generally compared to Recursion, Linear Programming and Naive methods.
Dynamic programming helps us in solving this problem.
Techniques to identify a DP problem
It is quite important to identify if a given problem could have a Dynamic Programming solution, otherwise, we won't be able to practice the advantages of this approach. Two major techniques are -

1. *Overlapping Subproblems - Overlapping subproblems must satisfy two conditions that problem can be broken down into subproblems and the result of each subproblem could be of use at later stage.*
2. *Optimal Substructure - If the optimal solution to a problem has resulted from the optimal solutions of its subproblem, then the given problem has Optimal Substructure characteristics too. Most of the classical DP problems are attributed to this principle.*

My favourite part of the DP lecture was this:

Sir asked, “How many of you love cricket?”, Indians love cricket. I am not an exception here, Everyone answered yes with enthusiasm. Sir asked us again, “How many of you watched the movie Moneyball starring Brad Pitt?” Now the number was less but I watched it and I love it. few said yes, as he referenced this movie I thought he was going to talk about statistics or Machine learning. Sir replied, “Oh just few people, well it’s a movie about baseball and it’s not a movie where the hero loses at the first and wins at the end.”
Sir started giving idea about the movie, “Well just to give you an idea I am comparing they have baseball tournaments every year it’s not same as IPL but it’s way different so one stark difference is the team management can spend any amount for players they can trade remove and do a lot of things, Brad Pitt plays the role of General manager for Oakland Athletics (aka Oakland A’s) their owner spends 35 million dollars for the team which is very low where others spend over a 100 million, they never won the championship and every good player who was groomed at Oakland A’s left it as they can’t pay the player more so all the best players at the end of the tournament will go to other clubs now when this movie takes place they’ve lost 3 best players. Brad Pitt should build a good team and he meets Jonan Hill in the movie and he is a MIT graduate who loves baseball. Now they look baseball completely in a new way they believe numbers they take many of old and wasted players but they order them also train them using the previous data, this team performs poorly initially for different reasons then once they get the momentum and they won 21 games continuously which was the world record and no team till today broke that record, however they didn’t win championship that year and the GM faced lot of criticism saying baseball is not like math it’s even complicated but there was a team called Boston Red Sox liked his approach and their owner gave a job offer worth 25 million dollars to him to be GM of the Boston red sox which he rejected for other reasons if he would’ve accepted it he would be the highest paid GM of the world, now this is a true story and many teams be it IPL or any rely on data so heavily.”
I knew this story but still listened it wondering where will he get to, Sir started, “2 Phd graduates from New Zealand after extensive research for 4 years who were asked initially to come up with an efficient system other than Duckworth-Lewis method to continue games which are stopped due to rain and sadly there is no good one till now”, Everyone laughed, Sir continued“ they came up with WASP (Winning and score predictor tool) this was first used in 2014 in a local game the core of this prediction and calculation is DP, based on results of all past matches they predict the current score and based on this they can predict the winner I’ll explain how they did it”, Sir moved towards board and started writing this:
Let $V(b,w)$ be the expected additional runs for the rest of the innings when b balls (not considering extras) have been bowled and w wickets have been lost, and let $r(b,w)$ and $p(b,w)$ be, respectively, the estimated expected runs and the probability of a wicket on the next ball in that situation.
We can then write,

$$V(b,w) = r(b,w) + p(b,w)V(b+1,w+1) + (1-p(b,w))V(b+1,w)$$

Since $V(b^*,w)=0$ where b^* equals the maximum number of legitimate deliveries allowed in the innings (300 in a 50 over game), we can solve the model backwards.

This means that the estimates for $V(b,w)$ in rare situations depends only slightly on the estimated runs and probability of a wicket on that ball, and mostly on the values of $V(b+1,w)$ and $V(b+1,w+1)$, which will be mostly determined by thick data points.
Sir continued, “This has many drawbacks like assuming the batting order is changed

and a good batsman comes at 10 now he could turn the game around this is a simple model they updated with time and technology".

----------QUESTION 2----------

Q)Given a perfect binary tree in the form of an array, find the maximum sum of nodes out of all paths from root to leaves?

At the start found the problem easy as I recently learnt Greedy algorithm and thought to solve it using Greedy approach just after seeing the question without hesitation. Started coding in the Sublime editor and logic goes on like this:
If we are at a node other than the leaves , then move in onto the node which has more value.If we are at ith node then we compare the values its children nodes i.e.. (2*i)th node and (2*i + 1)th node . So, we start at the root where the value of answer is the value of the root node. By comparing it's children's values we move to the children that has more value and add this value to the answer. We stop this process once we reach one of the leaves and return the answer. Got the code correct for a basic test case where the array is {1,2,3}.
The tree looks like this.

1
2 3

Here initially the value of answer is 1 and as 3>2 the value of the final answer becomes 1+3 which is 4.I thought that the code was right and was shocked when I saw that the 2nd test case itself fails. The second test case is an array = {7,3,12,1,10,6,5,99,4,8,13,2,9,11,14}. For this test case using greedy approach we would get the path 7->12->6->9 whose sum is 34 .But the answer that is given was 110.I was shocked by seeing the possible and tried to calculate it manually as it is a small test case . After calculating manually I got to know that the correct path is 7->3->1->99 . Now I got to know where I was wrong and understood where my approach . Choosing better way at a point will not always lead us to the correct solution . Got to know that greedy algorithm does not work in every situation . Now I got stuck and do not know how to continue . I was sitting alone in the room and there were no people around me to give any ideas . I thought I would get an idea if I revise notes and started revising notes that were thought after greedy algorithm . After studying the notes of DP I got thought I could use it to solve the question . In DP we build the solution of larger problems using the solutions of smaller problems . I thought this type of approach would help me to solve the given problem . Thinking of how to start , I started to think about the base case.To get to know about the base case we need to know how to solve the problem . For that I created an array named "Dp" and initialised it to zero . In this array Dp[i] would indicate the maximum sum of nodes from ith node to any of the leaves . So, for all the leaves their value in the Dp array will be the value of the respective nodes.Yes , I got the base cases and thought that I was moving in the proper direction . For a node "i" its children are "2*i" and "2*i + 1" . So, the value of Dp[i] is as follows

Dp[i] = max(Dp[2*i],Dp[2*i + 1]) + value of ith node.

The final answer would be the value of the Dp array at the root , that is Dp[0] which is the Dp value at root.We can find if a node "i" is leaf node or not by checking whether 2*i or 2*i + 1 is greater than the size of the given array or not . Now I thought I builded up the whole logic and started coding it . Now I started from the first node i.e.. the root and calculated the value of Dp[i] in each iteration and finished coding . Now I submitted the code and was shocked by it a wrong answer . I was into a dielama whether the logic was wrong or I was implementing

it in a wrong way. I was terrified by seeing that the code failed in the first test case itself . I was so confused as it failed in the first test case itself. So, I checked the values the Dp array for the first test case and got their values as {1,2,3} which is same as the given initial array . But I was confident that my logic is correct and there was some mistake in the implementation. So, I opened the code again and started checking it . For the first time saw it , it seemed quite well . So, I wanted to check it once again but wanted to look much more keen into it. While I was going through the loop , I just got angry on myself for making such a silly mistake. I did not initialize the base case and started calculating from the root which has to be done in the reverse direction . So, started from last node , checked if it was a leaf node or not ,if it was a leaf node then initialize it with the value of the node. Else calculate it Dp value using the method given above . In this method first we will all the leaf nodes i.e.. Initializing all base cases and then moving to their parent . Now submitted the code and got it right for all test cases.

----------NOTES 4----------

Branch and Bound notes:

Branch and bound (BB, B&B, or BnB) is an algorithm design paradigm for discrete and combinatorial optimization problems, as well as mathematical optimization. A branch-and-bound algorithm consists of a systematic enumeration of candidate solutions by means of state space search: the set of candidate solutions is thought of as forming a rooted tree with the full set at the root.

The algorithm explores branches of this tree, which represent subsets of the solution set. Before enumerating the candidate solutions of a branch, the branch is checked against upper and lower estimated bounds on the optimal solution, and is discarded if it cannot produce a better solution than the best one found so far by the algorithm.

The term branch-and-bound refers to all state space search methods in which all children of the £-node are generated before any other live node can become the £-node.

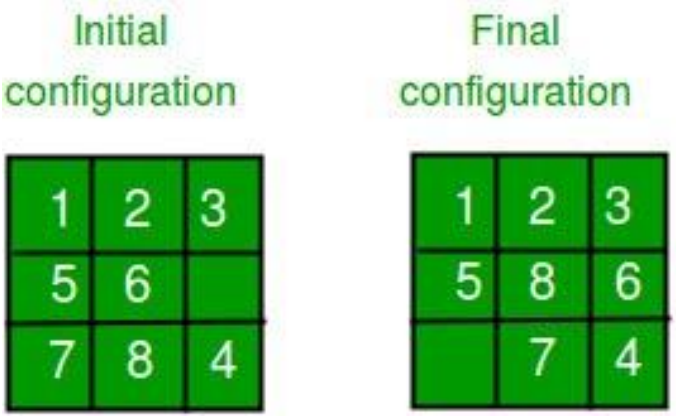
We have already seen two graph search strategies, BFS and D-search, in which the exploration of a new node cannot begin until the node currently being explored is fully explored.

Both of these generalize to branch-and-bound strategies.

- *In branch-and-bound terminology, a BFS-like state space search will be called FIFO (First In First Out) search as the list of live nodes is a first-in-first-out list (or queue).*
- *A D-search-like state space search will be called LIFO (Last In First Out) search as the list of live nodes is a last-in-first-out list (or stack)*

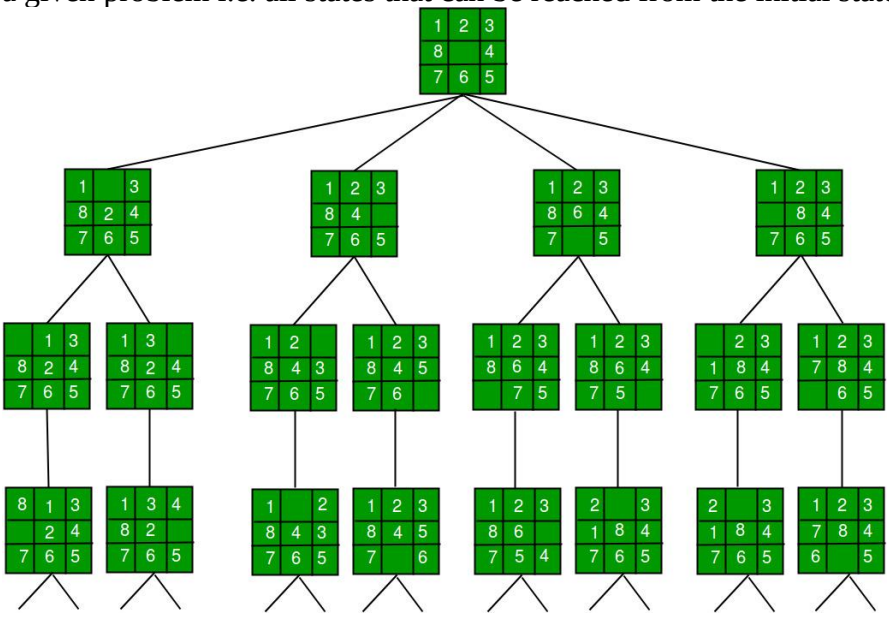
----------QUESTION 3----------

Given a 3×3 board with 8 tiles (every tile has one number from 1 to 8) and one empty space. The objective is to place the numbers on tiles to match final configuration using the empty space. We can slide four adjacent (left, right, above and below) tiles into the empty space.
Ex:



Rightnow I am thinking of applying DFS or BFS or Branch and Bound. so I will start with analysing DFS and then BFS and then Branch and Bound.

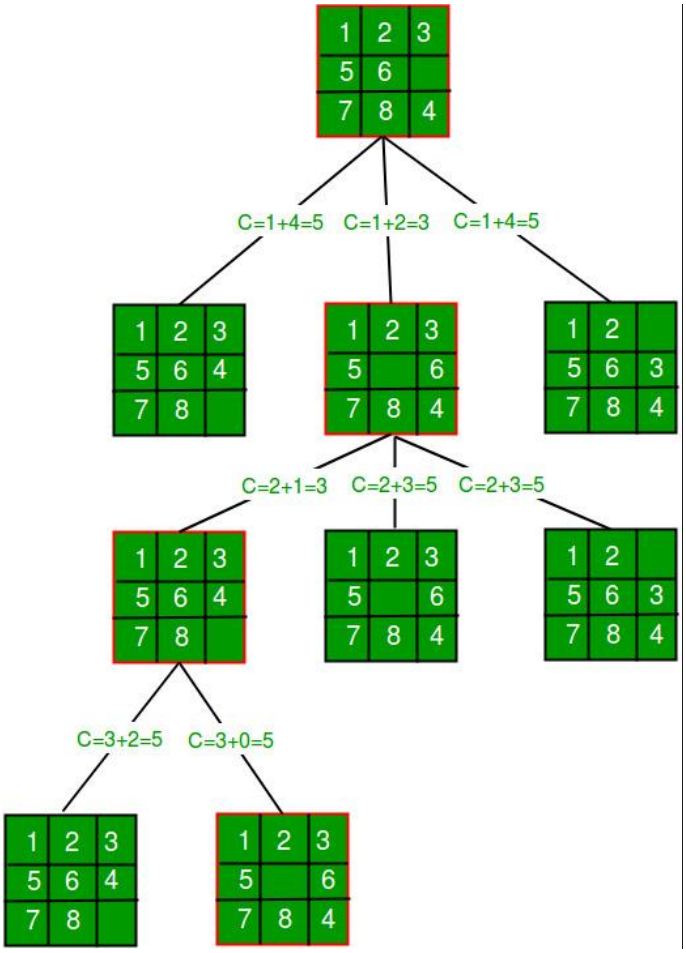
DFS is a brute force method, I can perform a DFS on state space (Set of all configurations of a given problem i.e. all states that can be reached from the initial state) tree.



In this solution, successive moves can take us away from the goal rather than bringing closer. So the search of the state space tree follows the leftmost path from the root regardless of the initial state. An answer node may never be found in this approach.
Since this approach takes a lot of steps and space and there is no definitive proof that an answer will be found so I don't think doing this is a better idea so I think I should start thinking about BFS.

Even BFS is also a brute force method like DFS. Here we can perform a BFS on the state space tree. This always finds a goal state nearest to the root. But no matter what the initial state is, the algorithm attempts the same sequence of moves like DFS. So this is also just a waste method for this solution.

So I'm going with branch and bound would be better.The search for an answer node can often be speeded by using an “intelligent” ranking function, also called an approximate cost function to avoid searching in sub-trees that do not contain an answer node.
It is similar to the backtracking technique but uses BFS-like search.We assume that moving one tile in any direction will have 1 unit cost. Keeping that in mind, I defined a cost function for the 8-puzzle algorithm. Now I have an algorithm for getting an approximation of the cost of reaching an answer node from the initial node which is an unknown value. Then I completed the algorithm.



This is the path followed by the final algorithm. This is the best path because only the nodes having the least value of cost function are expanded.

For the first time since I joined college, it felt like the academic side of life was going more smoothly than the rest. It wasn't something to be completely sad about, but I couldn't help but feel sad. Ever since that incident at the Survival mode event, I reduced my participation in such events and started spending more time with Rudra and Sucheta. That wasn't really a bad thing but most of the time we spent together was spent in cheering up Rudra, who was understandably very sad. Even Sucheta stopped being here usual bossy self, to help cheer up Rudra. He was making progress towards going back to his past self, but an incident like that would leave scars on anybody and for someone like Rudra they will inevitably take longer to heal.

A long weekend at home seemed to do the trick for Rudra. He looked much livelier, and much more like the Rudra we had grown used to over most part of the past 2 months. This was the last break before the semester ended. Most students would start preparing for endsems at this point. Though I still had 2 questions left on the leader board phase 1 list. Instead of juggling both of those, I decided to get done with the leader board stuff and then fully focus on the exams.

----------NOTES 5----------

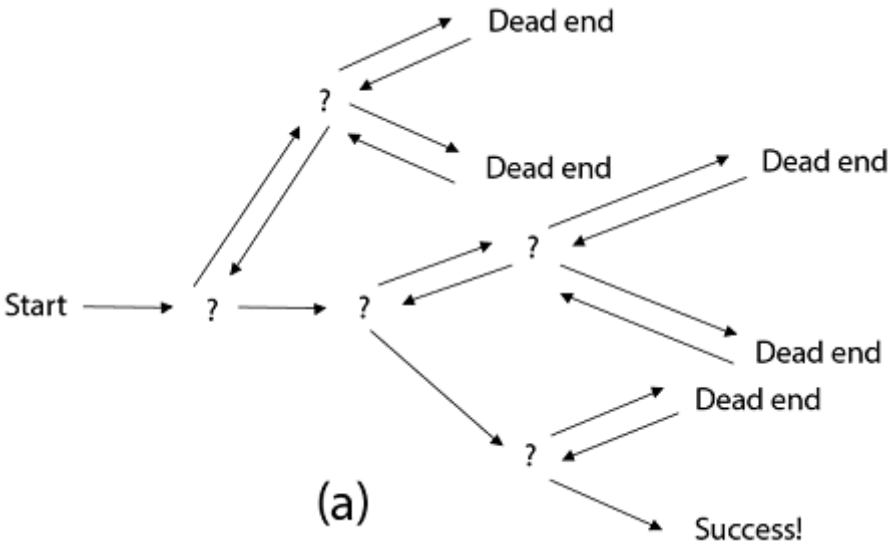
BACK TRACKING NOTES:

A backtracking algorithm is a problem-solving algorithm that uses a brute force for finding the desired output. If the current solution is not correct go back(backtrack) and check other solutions.

State Space tree: A tree is an abstract model of the possible consequences of choices we could make.

Its construction of the solution from partial solution starting with the root with no component solution (…). A node is said to be good if the partial solution is still feasible. If not, the algorithm ***backtracks*** *to the first promising node and explores the other branches of the state-space tree. Basically, Back tracking follows Depth for search (DFS) approach.*

A Visual representation of process looks like:



Main parts of the algo are:

- *The recursive function: Same as DFS, which takes a node and checks is it safe to add to Ans set. If so, adds it and iterates on the children. If not removes the present node from Ans set and goes back to last safe node and iterates on the next children of it. also includes terminating condition.*
- *The function that checks is it safe to add the present node to answer set. This checks all the conditions are satisfied or not if so, returns true else false.*

Pseudo code for algorithm:

```
boolean solve (node n){
    if n is a leaf node{
        if the leaf is a goal node, return true
        else return false
    }
    else{
       for each child c of n{
       if solve(c) succeeds, return true
       }
       return false
    }
}
```

A common example is path finding:

Robot can for example plan its path in a maze by recurring over the paths and backtracking from the ones that lead nowhere. This of course requires us to represent the maze in a way that the algorithm is compatible with. A common method is to use a 2-d matrix and values within it to represent obstacles or paths. Below is a simplified version of the maze solving problem that should help clarify the backtracking algorithm.

The Simplified Path Finding Problem

Given a N×N matrix of blocks with a source upper left block, we want to find a path from the source to the destination (the lower right block). We can only move downwards and to the left. Also, a path is given by 1 and a wall is given by 0.

Example input:

{ 1 , 0 , 0 , 0 }

{ 1 , 1 , 0 , 0 }

{ 0 , 1 , 0 , 0 }

{ 0 , 1 , 1 , 1 }

Sol:

The pseudo code goes like this:

If we have reached the destination point

return an array containing only the position of the destination

else

a,move in the forwards direction and check if this leads to a solution

b,if option a does not work, then move down

c, if either work, add the current position to the solution obtained at either a or b

```
def solveMaze( Maze , position , N ):
    # returns a list of the paths taken
    if position == ( N - 1 , N - 1 ):
        return [ ( N - 1 , N - 1 ) ]
    x , y = position
    if x + 1 < N and Maze[x+1][y] == 1:
        a = solveMaze( Maze , ( x + 1 , y ) , N )
        if a != None:
            return [ (x , y ) ] + a

    if y + 1 < N and Maze[x][y+1] == 1:
        b = solveMaze( Maze , (x , y + 1) , N )
        if  b != None:
            return [ ( x , y ) ] + b


Maze = [[ 1 , 0 , 1, 0 , 0],
        [ 1 , 1 , 0, 1 , 0],
        [ 0 , 1 , 0, 1 , 0],
        [ 0 , 1 , 0, 0 , 0],
        [ 1 , 1 , 1, 1 , 1]
        ]

print solveMaze(Maze,(0,0),5)
```

Fun fact: *If we check all possible combinations the total number for some famous problems the numbers look like these:*

- *N-Queens Problem: n^2 C n ways to place n X n board. For 8 X 8 there will be 4,426,165,368 possible arrangements.*
- *Sudoku: 9^n ways to fill n blank squares. for sudoku with 61 empty cells it will be 59-digit number.*
- *Travelling Salesman person: n! Ways for 200 nodded path it is 375-digit number.*
- *There are $(15!)^7$ ways to arrange the walkers in a 7-day Kirkman Schoolgirls Problem. That is an 85-digit number.*

- *When k-coloring maps of n regions, there are k^n possible colorings. For example, there are 1, 267, 650, 600, 228, 229, 401, 496, 703, 205, 376 ways to color a 50-region map with 4 color.*

After the class some guy made this meme showing his reaction towards leader board questions, I laughed at it.

----------QUESTION 4----------

Problem 3: Generate all sub sequences of given array

As always I took a paper and took a 4-letter string. ABCD now started permuting letter between them. After some time, I managed to list out all the 24 permutations. I understood that simple listing like this doesn't work. Started searching some pattern in the way he did it.as a common way is to jumble two letters we get a new combination.

But this jumbling (swapping) needs to be done in order way. So, thinks a little and gets the way.

I figured out swapping needs to be done in orderly way like this:

Pick an element (ex i =4) and swap with each right element of it (i =5 ,6, 7,) each time we swap we get a string now on the result string swap the element at i+1 (i=5) with elements on the right (i =6,7,)

Doing this recursively till we reach I =n-1 (end), Then print it.

Now for the recursive nature to work, swapping an element at j we create many branches to cover all those we need to get back after working on one of them...and do next one

This is looking like DFS....

So immediately coded in Vs code. checks for few cases and it works ...

```
void permute (int left, int right)
{
    // reached the max possible depth in dfs tree
    if (left == right)
    {  print_fn(arr);
       return ;
    }

    // iterating through each child node
    // created after swapping an element
    for (int i = left; i <= right; i++)
    {
        // Swapping
        swap(arr[left], arr[i]);

        permute(left+1, right);

        //backtrack, going back to parent ,
        //doing this as we are using same string
        swap(a[left], a[i]);
        }

}
```

----------NOTES 6----------

Linear Programming Notes:

Linear programming (LP) is a mathematical technique that allows the generation of optimal solutions that satisfy several constraints at once. Linear programming is one of the simplest ways to perform optimisation. Problems ranging from sharing a bar of chocolate between siblings to devising inventory and warehousing strategy for an e-tailer whilst considering millions of SKUs with

different popularity in different regions to be delivered in defined time and resources can be solved using linear programming.

Linear programming is a simple technique where we ***depict*** *complex relationships through linear functions and then find the optimum points. An interesting problem of optimisation is the "Diet Problem"(the search of a low-cost diet that would meet the nutritional needs of a US Army soldier). A diet had to be planned while taking into consideration nutritional quality, economic viability, and environmental sustainability. The mathematicians employed during thought LP to be "the ideal tool to rigorously convert precise nutrient constraints into food combinations". Linear Programming (LP) was used to solve questions on matching diets to nutritional and other additional constraints with a minimum amount of changes. Interestingly or weirdly during the earlier trials, the final diet that resulted was 200 bouillon cubes per day! This was because no upper bounds were set for salt content. So, for the first time upper bounds were added to LP.*

The basic assumption in this method is that the various relationships between demand and availability are linear. To obtain the solution, it is necessary to find the solution of the system of linear inequalities (that is, the set of n-values of the variables x_i that simultaneously satisfies all the inequalities). The objective function is then evaluated by substituting the values of x_i in the equation that defines f .

Common terminologies used in Linear Programming

- ***Decision Variables:*** *The decision variables are the variables that will decide my output. They represent my ultimate solution. To solve any problem, we first need to identify the decision variables.*
- ***Objective Function:*** *It is defined as the objective of making decisions. For example, if a company wishes to increase the total profit, profit is the objective function.*
- ***Constraints:*** *The constraints are the restrictions or limitations on the decision variables. They usually limit the value of the decision variables.*
- ***Non-negativity restriction:*** *For all linear programs, the decision variables should always take non-negative values. This means the values for decision variables should be greater than or equal to 0.*

The process to formulate a Linear Programming problem
Let us look at the steps of defining a Linear Programming problem generically:

1. *Identify the decision variables*
2. *Write the objective function*
3. *Mention the constraints*
4. *Explicitly state the non-negativity restriction*

For a problem to be a linear programming problem, the decision variables, objective function and constraints all have to be linear functions.

If all the three conditions are satisfied, it is called a ***Linear Programming Problem****.*

I have delayed my endsem preparation much more than expected. I need to finish this last question today by all means.

----------QUESTION 5----------

Q)Given an array in ascending order you have to find the position of the asked number if it's repeated more than once then the smallest one should be returned?

And then there are few constraints and this is the first question this is too lame just a for loop and checking if it's the element or not everytime will give the output it's just basic. Opened Sublime text editor and coded it quickly and it worked for a given test case where the size of array was 7 and the elements are 1,99,100,121,122,123,125 and we are supposed to find the position of 100 and 101. The question statement says if not found print not found so just return -1.

To solve for the -1 I have to just add a flag and change its value if we found the number or else don't change so if the value of flag didn't change it means we didn't find it. This is so basic so let me code it and done let's check for 100 it says 3 which is right and for 101 it says -1 so yes then it's done, Submitting and done it's running and it said TLE in red.

So this isn't that straight forward obviously why would they ask this simple questions let me read the question again to see if I can do better, is there any other way this is O(n) in worst case can we do better if we can then it's probably O(logn) that's only possible and the constraints say size of array can go till 10 power 18 so it's O(logn) or nothing.

"Oh wait I think I got it, We've to use binary search it's O(logn) we are done then", I shouted others in the room reacted like "I guess archimedes didn't shout Eureka that loud we all know we have to use binary search but still getting it wrong".

"I guess you've implemented it wrong way I'll do it", Then it's simple so basically I have to compare with middle each time and change the range each time if I find it then I am done

basically I have to break and return the answer if I don't find it and the range becomes 0 then it means there is no such element then I have to return -1.
Okay so two integers which have start and end of the array we have to search the condition to end is when out of those two integers low,high when low crosses high then it means there is no answer so the condition for the while loop is low<=high and based on mid the low and high change and the moment A[mid]=X we have to print mid and break but we need to return -1 if we don't find so for that if we add a flag and change value of flag if we find the value so if the flag isn't changed then we return -1.
Okay yeah I am done lets run it for the given case for 100 it's 3 for 101 it's -1. Good it's working and the response is "WA" in red.
What could be the problem here? Let me see if it's giving the correct position for all values and it is correct and it's giving -1 when it's different for the given array.
Let me read the question again ascending order it is and if the number is repeated then return the smallest position wait the numbers can be repeated so let me try 1,2,2,2,2,2,2 and position of 2 I got 4 oh I shouldn't break right after finding that was the mistake, so removing the break statement and the next search should be in range (0,mid) if it <= A[mid] in other half now after changing let me see for the 1,2,... case and got 2 it's working okay "I think it's done I'm submitting" Others didn't consider the case that it could be same so everyone was eager to see the result and it said WA again. "How? Again I feel like the test cases are wrong at this point probably", others said " The toppers submitted dude".

"Are you still doing the last question", whispered Rudra.

The three of us were studying in one of the workspaces, where silence was mandatory.

"Yeah, it's an easy question but I can't find the bug", I whispered back.

"You should ask someone for help", said Sucheta," If you delay any further, there will no time to cover everything"

"I know, but who do I ask?"

"Ask your favourite prof", replied Sucheta

"I can't ask him such a small thing, that'll be really stupid", I was sure that it was some very small mistake, but even after verifying each step multiple times I wasn't able to find any faults, all the test cases I tried worked out.

"Ask Jeeshanth", said Rudra.

“What!”, my reaction was too spontaneous to control my volume. Everyone in the workspace glared at our side of the room.

“Are you mad?”, I asked, reverting back to whispers.

“No, if you won’t ask, I will. It’s the fastest and best option you have right now”, he said.

“No stutter, you aren’t bluffing are you Rudra”, asked Sucheta, she looked proud of him for some reason.

“No, I’m not, there’s no point wasting time over a small bug, just ask him and get it over with”

“I’ll just leave the question”, I said.

“We all know that won’t happen”, said Sucheta, “It’s ok, for once Rudra is right, just get it over with, we need you to explain a lot of topics after you’re done with the leader board business”

“Damn bug, and could you stop with the explaining thing.”

After a lot of arguing in whispers, and a lot of back and forth, I finally decided to ask Jeeshanth. For me, this was a step towards forgiving people faster and trying to understand their side of the story. That was the one of the main reasons I used to convince myself to do it.

“What’s his room number”, I asked, hoping Rudra might know.

“I don’t know”, replied Rudra.

“You don’t need his room number he’s right there”, said Sucheta, pointing to the other end of the workspace. Jeeshanth was there, watching something on his laptop, was he already done with his preparation, that wasn’t important.

“Hey”, I went there with my laptop to show him my code.

“What”, he grunted.

“Are you free right now?”, I asked, just to be nice.

“Obviously”, he said pointing at his laptop screen where some sort of anime was playing.

“Umm.. I have a doubt in the last question of leader board”, he interjected me before I could complete.

“The binary search one, that’s the easiest of all”, he said, his tone was the same as before, full of arrogance.

“Yes, I did everything as explained in class, all my test cases are working, I’m not able to find the bug. I was hoping you could help.”

That statement took a lot of determination and patience. I was on the verge of saying never mind and leaving, but that would defeat the purpose of everything and make me look like an idiot.

“If you did everything correctly, you wouldn’t be asking me for help would you”, he said talking my laptop to check my code.

“This is wrong”, he said pointing to a line in my code.

mid = (min+max)/2;

“It should be ….”, he typed something in its place.

mid = min+(max – min)/2;

Weren’t they both the same? But I was too irritated to extend the conversation any longer.

“You don’t know the difference, do you? Oh, you don’t even want to know the difference, I get it no problem”, he said.

“I didn’t say that …”, my volume started shooting up again. I don’t know if that was the reason or it was because I didn’t want to talk to him anymore, I just grabbed my laptop and left the workspace. I was just too angry to think properly. I could see that Rudra and Sucheta were packing their stuff to follow me out of the workspace, but I didn’t wait for them, let them catch up. This was their stupid idea after all.

Eventually both of them caught up with me in the middle of one of the hallways.

“What happened”, asked Sucheta.

“Nothing”, I said, moving faster now, just to make it harder for them to maintain their stride.

“Why do boys always have to get their ego in the way for everything”, she said.

“You are talking about ego, wow, if you were in my position you would have punched him in the face.”

“We can’t hear you”, she said, struggling with the books.

“Good”

I started walking faster, I didn’t want to talk about it, they wouldn’t be able to catch up with all the books in their hands. I’ll tell them later.

Phase-2

Last semester was a mixed bag, but I don't think I have any major regrets. Endsems went well, taking into account my very late arrival to the revision scene. My SG, I felt reflected the sem perfectly, 8.4, slightly above average. This was a first for most of us, being in the average section of the group. Over the past four years, anything below the top was unacceptable, irrespective of what parents or peers think, it was an innate feeling. Though, none of those feelings seem to have resurfaced after the sight of my SG. I could sense that my parents did expect me to maintain the same level as I did during checkpoint, but they obviously acted happy about it when I was around. I don't know whether to treat this as a good thing or a bad thing. The will to always be the best was my biggest driving factor until college, now that I seem to have lost it, I should be able to find more meaningful factors to drive me forwards. Though, there still this feeling of dread, am I telling myself to find other motivating factors just to slack off from academics?

I had hoped that completing phase-1 of the leader board questions would have helped me resolve my dilemma, but that wasn't the case. Most of the phase-1 questions just tested whether we were able to apply what was taught in class, without any twists or turns or interesting new information. Finishing it because of Jeeshanth help took away any sense of satisfaction on completing those questions. I did understand my mistake after I had cooled down that day, but that didn't help with the feeling that I wouldn't have found the bug if not for him, and that feeling was infuriating. After that day, I decided whatever the consequences, I will not ask him for help regarding anything.

Leaving last sem behind, most of the courses in the new sem were continuations of the courses taught in the last sem. This meant that most of the professors that taught us last sem also taught their continuation courses. This semester, I also decided to maintain a separate book for notes, but I will still be using this one for the leader board questions. (No specific reason)

Speaking of the leader board, we were sent an email one week into the sem, containing the phase-2 set of questions for the leader board. As expected, only those students who completed the first phase received that mail. Even though there was no real reason for me to try out the new questions, there wasn't any reason to not try them out, so I decided to do what I did last sem, and go one question at a time.

----------Medium Q1----------

The advertising alternatives for a company include television, newspaper and radio advertisements. The cost for each medium with its audience coverage is given below.

	Television	**Newspaper**	**Radio**
Cost per advertisement ($)	**2000**	**600**	**300**
Audience per advertisement	**100,000**	**40,000**	**18,000**

The local newspaper limits the number of advertisements from a single company to ten. Moreover, in order to balance the advertising among the three types of media, no more than half of the total number of advertisements should occur on the radio. And at least 10% should occur on television. The weekly advertising budget is $18,200. How many advertisements should be run in each of the three types of media to maximize the total audience?

First, I am going to formulate my problem for a clear understanding. The final goal is optimisation, so this is a problem that can be solved using LP.

I read the notes to see the approach and the approach was:

Simplex Method

Simplex Method is one of the most powerful & popular methods for linear programming. The simplex method is an iterative procedure for getting the most feasible solution. In this method, we keep transforming the value of basic variables to get maximum value for the objective function.

*A linear programming function is in its **standard form** if it seeks to maximize the objective function.*

$Z = c_1x_1 + c_2x_2 + ... + c_nx_n$

subject to constraints,

$a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \leq b_1$

$a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \leq b_2$

. . . .

. . . .

$a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \leq b_m$

where, $x_i \geq 0$ and $b_i \geq 0$. After adding slack variables, the corresponding system of constraint equation is,

$a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n + s_1 \leq b_1$

$a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n + s_2 \leq b_2$

. . . .

. . . .

$a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n + s_m \leq b_m$

where, $s_i \geq 0$

The variables, $s_1, s_2, ..., s_m$ are called slack variables. They are non-negative numbers that are added to remove the inequalities from an equation.

I used this to solve the question following the given steps

First, Identify Decision Variables

Let X_1, X_2, X_3 represent the total number of ads for television, newspaper, and radio respectively.

Then, Objective Function

The objective of the company is to maximize the audience. The objective function is given by:

$Z = 100{,}000X_1 + 40{,}000X_2 + 18{,}000X_3$

Finally, Write down the constraints

Now, I will mention each constraint one by one.

It is clearly given that we have a budget constraint. The total budget which can be allocated is $18,200. And the individual costs per television, newspaper and radio advertisement is $2000, $600 and $300 respectively. This can be represented by the equation,

$2000X_1 + 600X_2 + 300X_3 \leq 18{,}200$

For a newspaper advertisement, there is an upper cap on the number of advertisements to 10. My first constraints are,

$X_2 \leq 10$

The next constraint is the number of advertisements on television. The company wants at least 10% of the total advertisements to be on television. So, it can be represented as:

$X_1 \geq 0.10(X_1 + X_2 + X_3)$

The last constraint is the number of advertisements on the radio cannot be more than half of the total number of advertisements. It can be represented as

$X_3 \leq 0.5(X_1 + X_2 + X_3)$

Now, I have formulated my linear programming problem. We are using the simplex method to solve this.

To reiterate all the constraints are as follows. I have simplified the last two equations to bring them in standard form.

$2000X_1 + 600X_2 + 300X_3 \leq 18{,}200$

$X_2 \leq 10$

$-9X_1 + X_2 + X_3 \leq 0$

$-X_1 - X_2 + X_3 \leq 0$

We have a total of 4 equations. To balance out each equation, I am introducing 4 slack variables,

S_1, S_2, S_3 and S_4.

So our equations are as follows:

$2000X_1 + 600X_2 + 300X_3 + S_1 = 18{,}200$

$X_2 + S_2 = 10$

$-9X_1 + X_2 + X_3 + S_3 = 0$

$-X_1 - X_2 + X_3 + S_4 = 0$

I hope now you are available to make sense of the entire advertising problem. All the above equations are only for your better understanding. Now if you solve these equations, you will get the values for $X_1 = 4$, $X_2 = 10$ and $X_3 = 14$.

On solving the objective function, we will get the maximum weekly audience as 1,052,000.

If all questions in this phase were going to follow a similar pattern, there wouldn't be much trouble. Though it is more work than the first phase, the difficulty is only in reading up on the actual implementation of stuff theoretically taught in class. In other breaking news, Suhas has friends! Most of them have been hanging out in our room recently, trying to crack the new phase of leader board questions. While I was the only one caring about the leader board in out trio, my room mates friends seem to be caring only about that, we might be the weird ones after all.

----------Medium Q2----------

Q2) Given an array cell[] of N elements, which represent the positions of the cells in a prison. Also, given an integer P which is the number of prisoners, the task is to place all the prisoners in the cells in an ordered manner such that the minimum distance between any two prisoners is as large as possible. Finally, print the maximized distance.

It's asking maximized distance so It looks like If I use greedy this would work, based on the way this sentence is framed, "This looks easy I'll try this", I said out loud in the room everyone looked at me very surprised, they probably tried and failed. It doesn't matter, this looks simple, so it's a 2D coordinate system and I have to choose N points such that the minimum distance between two prisoners is maximum, it asked distance it can be irrational also right I thought, I asked,

"What if the distance is irrational? They should be asking us the square of distance right", I asked.

Everyone made odd faces and one of them said,

"How come subtracting two integers will be irrational, did you understand the question?", I replied yes and told them what I understood. Everyone giggled, I decided I didn't like them any better than my roommate, and Suhas said,

"No, the array cell will give you all the available positions in the prison you have to choose P from those N such that the smallest difference between all of them is as big as possible. For

example see the example test case they gave cell= {10,12,20,100} and P=3 there are 4C3 possibilities they are {10,12,20}, {10,12,100},{10,20,100},{12,20,100} minimum distance of these are 2,2,10,8 respectively so it's maximum in {10,20,100} so you have to return 10."

Why didn't I see test cases and make sense out of them, anyway the explanation was clear so I thanked him and moved on to solve it. One way would be the naive method checking the minimum distance for all permutations and returning the minimum, I could use the next perm function and breaking the vector let me code this, it's simple storing minimum of all permutations in an array Minarr and returning maximum of Minarr would do it, I have coded it let me submit and it was quick and It said TLE in red. I looked at constraints, for N it is 3<N<106 for P its 1<P<N so now if I check for every permutation the number of times I check it is a polynomial of n and the worst case would be nCn/2 would be large so this is not the right way and I wondered if there is any other way, One thing I can be sure of is two can't go to same prison then the minimum distance would be 0. Greedy should do I think, If I start right from the left and keep maximising distance then I would be getting the maximum of minimum distances, I looked at Suhas and asked, "Greedy should do this I guess if we keep each prisoner as far as possible then we will get the maximum of minimums right?". He replied, "No we can't as choosing an option here will affect other options you have, Do you remember that class where sir told when greedy won't work well that is the case here". Then, I realised my mistake, greedy won't work here, "So which approach are you trying now?", I asked him, he replied, "Dynamic programming, that's what we could do right?", I replied, "Well DP will have the time complexity O(N*P) this also won't work in our case."

Why do I always jump to stupid assumptions that fast? This question was nothing like the first one, it wasn't some complex algorithm implementation. On the face of it, the question doesn't look that complicated, but every method I have tried for finding a solution has been futile. For the first time, I don't have any idea about how to solve the question and it is frightening. Though, I shouldn't jump to conclusions and repeat my mistake. There is every chance that the next question is something I will be able to do. Though, that would mean that my plan of sequentially completing the questions would have to be disturbed. Doing it sequentially wasn't a big deal but, it always helps with confidence building, and my confidence is pretty low after this question.

Though, the non-academic part of college was much more exciting at this point. After monitoring us throughout the first semester, the senior members of the football team have decided to take trials for the first years. It seems the tradition is that the 4^{th} years who will be leaving the team will conduct trials and pick one person from the juniors based on their

observations. I have always looked up to the fourth year players , throughout the first sem, whenever I made mistakes, one of them would be there to teach me what the right way was. It's almost like they were choosing their successors, and it would be great if any of them picked me but if the captain picked me it would be unforgettable. The concept was so cinematic, it might help Sucheta

overcome her hatred for sports where there is running involved.

"I just don't like it", she would snap back, whenever I brought up her hatred for football. She was so great at sports like table tennis, volleyball and badminton, that I deduced her hatred for football was because of the running part. Sadly, I was never able to confirm that hypothesis, but the idea of using the successor thing is still on the table.

Rudra on the other hand was too clean for sports. Even during the compulsory PE slots, he would be the only one to come out with neatly parted hair, clean glasses and the least amount of sweat. In the four months that we had, Sucheta and I were able to slightly alter Rudra's 90s fashion sense, but his obsession with cleanliness was still a problem to be rectified.

----------Medium Q3----------

Q)Let there be N workers and N jobs. Any worker can be assigned to perform any job, incurring some cost that may vary depending on the work-job assignment. It is required to perform all jobs by assigning exactly one worker to each job and exactly one job to each agent in such a way that the total cost of the assignment is minimized.

In this case I can think of a brute force way, by Hungarian algorithm, DFS/BFS on a state space tree, Finding Optimal Solution using Branch and Bound. First I'll go through analysing brute force method and then Hungarian algorithm and then DFS/BFS on a state space tree then Finding Optimal Solution using Branch and Bound.

In the brute force method first I generate all the n! possible job assignments and for each such assignment, then I computed its total cost and returned the less expensive assignment. Since the solution is a permutation of the n jobs, its complexity is O(n!).

Next onto the Hungarian algorithm. The optimal assignment can be found using the Hungarian algorithm. The Hungarian algorithm has the worst case run-time complexity of O(n^3). So it's no good too so I moved onto the DFS/BFS on a state space tree.

I performed depth-first search on the state space tree but successive moves can take us away from the goal rather than bringing closer. The search of the state space tree follows the leftmost path from the root regardless of initial state. An answer node may never be found in this approach. I also performed a Breadth-first search on the state space tree. But

no matter what the initial state is, the algorithm attempts the same sequence of moves like DFS. So this method is also of no use.

This is a disaster, I have the football trials tomorrow, and I successfully screwed up my mood. This is so weird, I'm not even able to get a basic idea that will work for me to build on. I guess now I understand why some of the seniors advised against attempting the leader board completely. Not many people actually completed the whole of phase-1, and every time I feel like giving up on the leader board, that thought keeps stopping me from doing it. I know even this train of thought comes under jumping to conclusions, but two in a row is too much. I can't help but feel that all I am able to do is easy questions or questions that just need me to implement something that already exists. What if I really end up stuck in this part of the leader board, what if I ended up doing the exact thing I was warned against doing by my seniors? This is bad, I need to sleep, it's getting late, but I don't know if I can. I must, I can't go half asleep to the trials tomorrow.

I regulated my food intake during breakfast and lunch to ensure a good performance in the trials, and it was finally time. The football ground was divided into three parts using cones, half of the court was set aside for what looked like a mini match, and the other half was divided into two more halves. Everybody started lining up in front of the seniors that were already there, standing with bibs and some balls in the centre of the field. In the middle of all the seniors was the team captain, Arjun Desai,

"Ok, line up fast everybody, based on the forms you filled last week, we have a list of your names, preferred positions and preferred foot. I will be taking attendance to check if everyone has showed up. Then you will be divided into four separate groups and sent to different parts of the field to take part in the four different tests, stamina, passing, dribbling and shooting and mini-match.", he announced.

I expected only three tests based on the field division, but the fourth and unexpected test was on my weakness, stamina. Could my luck get any worse? I hope not. After the attendance, we were randomly split up and our group was taken to the field entrance.

"We will be doing the stamina test first, 10 rounds", said one of the seniors in charge of our group. Arjun wasn't in the group of seniors that were evaluating our group so that completely ruled out the possibility of being his successor. Though, having the stamina test first was helpful, because I might not able to do it if it was the last test. Contrary to our assumption the ten rounds were not around half the court but the total court, this was double the distance I was prepared for, but I made it in the end, barely able to jog during the last round. Anybody not able to complete the ten rounds was asked to leave and try

next year, that greatly helped push myself to the finish. After the stamina test we had a 5 minute break and then continued on to the passing and dribbling and shooting tests. Finally we played a 20 minute 7 a side match for the final test, in which I played in my favoured role as defender. Our side conceded just one goal as compared to the oppositions 4, so I hoped my performance was good enough to make it to the final cut.

Though Arjun came over right after the end of our mini-match, with the results on his phone. They must have decided their pick mid-way through the match itself.

“Great performances from everybody, unfortunately, we wont be able to select all of you. I have three names here chosen by my fellow classmates who have monitored each of you closely during the trials. If you don’t hear your names, you may leave the field, but remember there is always next year and never give up,”

“Rishabh Yadav”

“Saranga Pani”, one attacker and one midfielder already picked, it was now or never.

“Jaishnav”, this was maybe the most joy I had ever felt since joining college. I had been selected into the team! Though I had always hoped for this, I was mentally preparing myself for rejection.

“Okay you may leave, Jaishnav”, called Arjun sir, pointing towards a vacant spot on the field. I walked there behind him.

“Congratulations Jaishnav”, he said, but his expression was serious something was wrong.

“Thanks sir”, I replied, trying to sound as upbeat as possible.

“So, how is your leader board phase 2 going”, he asked.

That was completely random. Why would he be asking about the leader board? Did see my points stagnate. Maybe he is willing to help, I shouldn’t lie about it then.

“Not so great sir”

“As expected, if you try to skip the basics and go straight for higher level questions you will obviously get stuck”, I was confused. When did I skip easy questions? I did all the phase-1 questions, I wouldn’t be able to do any of the phase-2 questions if I didn’t.

“I didn’t skip any of the phase-1 questions sir”, I interjected, making my confusion obvious with my tone.

“But you used another student’s work to finish it didn’t you”, he said.

“No sir! I did all of them by myself”, I was completely thrown off by the accusation.

“There is no point denying it Jaishnav, Jeeshanth told me that you used his code to finish the last question in phase-1. I trust him because he’s from my school, and I don’t think he would lie about such things.”

This man was my idol, and he was falsely accusing me of plagiarism. This shouldn’t be happening.

“I didn’t use his code sir, I just…”

“How you did it is irrelevant now. Just remember that in a few months you will be a senior too, and a new batch will be looking up to you as a role model. From what I have seen on the field, I know you’re a good kid, but what you did was wrong. This whole talk was to make sure that you never repeat such a thing. I hope a warning will suffice; I’m not penalizing you because I trust you. Can I trust you?”

“Yes sir”, I answered, too many emotions where flowing through me, I felt like running away.

“Good, now go, you must be tired”, he said and left the field.

I stood rooted to the same spot for what seemed like an eternity. Then my mind cleared, and I had only one dominant emotion in my mind, anger. I was setup by Jeeshanth, he somehow managed to convince Arjun that I cheated in the last question. Why would he do something that horrible? I had to find out, he has to pay for this.

I ran towards the hostel; the notice board in the ground floor has a list of all students and their room numbers. Normally, it would be pretty hard to find someone by their name in such a huge list, but his name stood out for me.

Jeeshanth Room No 204

He must pay. I reached his room and kicked the door open. He was there, standing in the middle of his room, his roommate wasn’t there, which was good. I closed the door behind me.

“What do you want”, he asked, as if nothing had happened.

“Why? Why did you lie to Arjun sir?”, I asked, my whole body was shaking, barely able to control myself.

“Ohh that, I didn’t lie, I just told him information which might be helpful while selecting people for the team.”

That was it, I could no longer restrain myself. I grabbed him by the neck and shoved him onto the wall behind him.

“Tell me why you did it, I wont ask again”, I still had my arm around his neck. All that arrogance and overconfidence had vanished from his face, he looked scared.

“I just thought that someone better deserved that spot on the team”

“Someone better?”, then it dawned on me, the used football kit on the floor, it all made sense,” You didn’t get selected so you tried to take my spot by telling on me”

He didn’t respond, he still had that same scared expression. I let go of his neck.

“All that show business, all that act you put up, as if you are above everyone. This is what you actually are, a deceiving piece of shit.”

“I didn’t lie”, he repeated, but this time there was barely any volume.

“You won’t change, will you?”, I knew scum like him, always want to be the best by stamping down the rest of their competition, “ I will show you your place, I will show it to you in that leader board that you love so much. You want to be the best, beat me and then I will accept you didn’t lie and leave this damn college.”

He was still whimpering, and I didn’t care if he registered what I said. I knew what to do next. I slammed the door behind me and headed towards my room.

I started to go back to the questions and try to think those again so I went back to the second question.

Q2) Given an array cell[] of N elements, which represent the positions of the cells in a prison. Also, given an integer P which is the number of prisoners, the task is to place all the prisoners in the cells in an ordered manner such that the minimum distance between any two prisoners is as large as possible. Finally, print the maximized distance.

All my previous approaches were futile so, I thought for a while and realized. I should code it in a way such that time complexity is O(N) or O(N*logn) I feel O(n) is impossible so probably it’s divide and conquer approach.”

If there are two then I have to place them at the two end points and now the third should be near to the middle element of those two so maximum of minimum distance is maximum if we follow this approach for all then we will get the answer it looks like binary search, so the first requirement here is the cell array should be sorted so we have to sort it initially then,
Now we if we search for maximum distance in all distances possible then we are done so

now the left and right or high or low of the binary search represent total distance between to ends and 0 and the mid represents the current distance for which we are checking now at each mid we have to check whether it's possible to arrange elements such that the minimum distance between those is at least mid, and follow normal binary search approach so we could get the highest mid.

```
sort(cell, cell + n);
int start = 0;
int end = cell[n - 1] - cell[0];
int mid = 0;
while (start <= end) {
        mid = start + ((end - start) / 2);
        if (canPlace(cell, n, p, mid)) {
                start = mid + 1;
        }
        else {
                end = mid - 1;
        }
}
return mid;
```

Here mid has the answer and the function canPlace will check if we can place prisoners such that the minimum distance is mid among them.
This should work. I didn't make any mistakes in binary search as before and I submitted it and it was not accepted. I got WA in red. Then I realised that the mid keeps on changing and it's not correct to return it every time so storing mid value in another variable whenever it's possible to have mid as the answer and returning that variable would solve it.
I submitted it after making that change and it was accepted finally,
With this confidence I went back to the third question to solve

Q3)Let there be N workers and N jobs. Any worker can be assigned to perform any job, incurring some cost that may vary depending on the work-job assignment. It is required to perform all jobs by assigning exactly one worker to each job and exactly one job to each agent in such a way that the total cost of the assignment is minimized.

So I moved onto Branch and Bound. I think the selection rule for the next node in BFS and DFS is “blind”. So the selection rule does not give any preference to a node that has a very good chance of getting the search to an answer node quickly. The search for an optimal solution can be speeded by using an “intelligent” ranking function, also called an approximate cost function to avoid searching in sub-trees that do not contain an optimal solution.Since it is similar to BFS-like search but with one major optimization. Instead of following FIFO order, we choose a live node with least cost. I don’t think I can get the optimal solution by following node with least promising cost, but it will provide a very good chance of getting the search to an answer node quickly.

Now for each worker, we choose a job with minimum cost from the list of unassigned jobs (take minimum entry from each row).

I will now take an example to visualize the algo.

Ex: calculating the promising cost when Job 2 is assigned to worker A.

	Job 1	Job 2	Job 3	Job 4
A	9	2	7	8
B	6	4	3	7
C	5	8	1	8
D	7	6	9	4

Since Job 2 is assigned to worker A (marked in green), cost becomes 2 and Job 2 and worker A becomes unavailable (marked in red).

	Job 1	Job 2	Job 3	Job 4
A	9	**2**	7	8
B	6	4	3	7
C	5	8	1	8
D	7	6	9	4

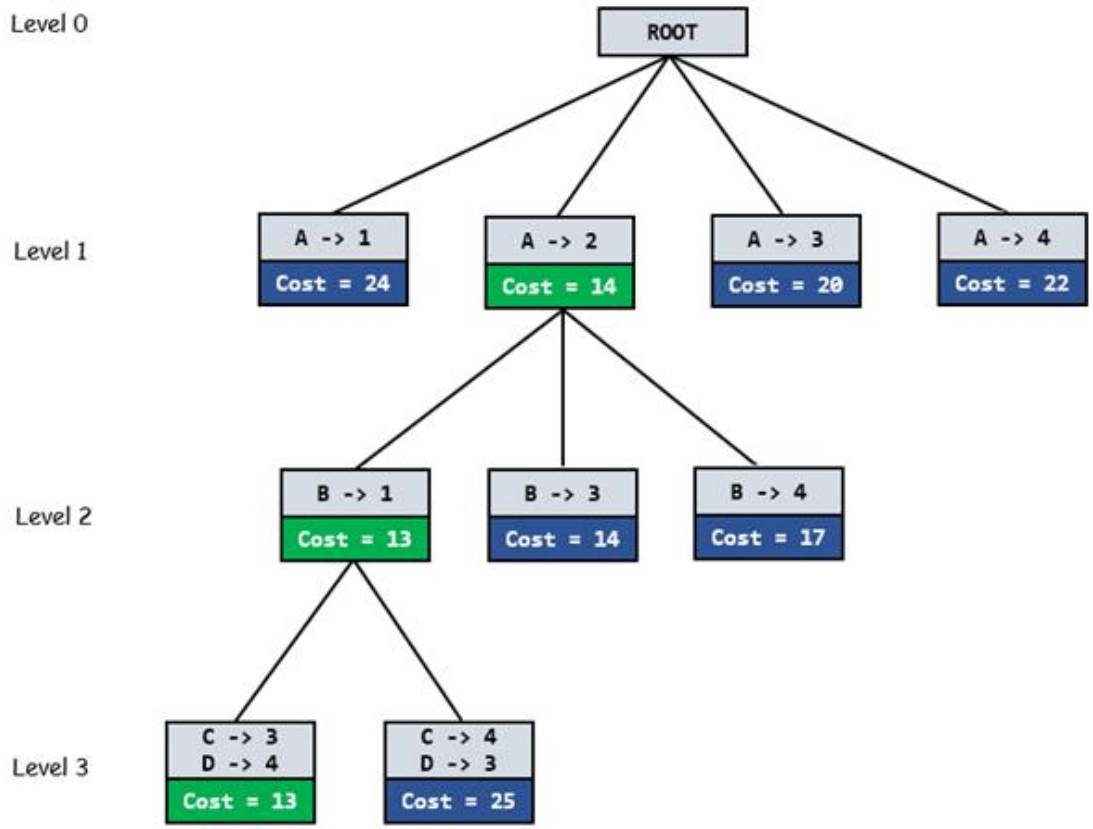
Now we assign job 3 to worker B as it has minimum cost from list of unassigned jobs. Cost becomes 2 + 3 = 5 and Job 3 and worker B also becomes unavailable.

	Job 1	Job 2	Job 3	Job 4
A	9	**2**	7	8
B	6	4	**3**	7
C	5	8	1	8
D	7	6	9	4

Finally, job 1 gets assigned to worker C as it has minimum cost among unassigned jobs and job 4 gets assigned to worker C as it is only Job left. Total cost becomes 2 + 3 + 5 + 4 = 14.

	Job 1	Job 2	Job 3	Job 4
A	9	**2**	7	8
B	6	4	**3**	7
C	**5**	8	1	8
D	7	6	9	**4**

complete search space diagram showing optimal solution path in green:



This algorithm was efficient, and it worked. I was so happy as I solved the ones which I was about to skip. I moved onto all questions from here with enthusiasm.

A farmer has recently acquired a 110 hectares piece of land. He has decided to grow Wheat and barley on that land. Due to the quality of the sun and the region's excellent climate, the entire production of Wheat and Barley can be sold. He wants to know how to plant each variety in the 110 hectares, given the costs, net profits and labor requirements according to the data shown below:

Variety	Cost (Price/Hec)	Net Profit (Price/Hec)	Man-days/Hec
Wheat	**100**	**50**	**10**
Barley	**200**	**120**	**30**

The farmer has a budget of US$10,000 and availability of 1,200 man-days during the planning horizon. Find the optimal solution and the optimal value.

I should use LP to solve this problem. I wanted to learn the graph approach so I could use it some time again. There were notes for this

Graphical Method

This method is used to solve a two-variable linear program. If you have only two decision variables, you should use the graphical method to find the optimal solution.

A graphical method involves formulating a set of linear inequalities subject to the constraints. Then the inequalities are plotted on an X-Y plane. Once we have plotted all the inequalities on a graph the intersecting region gives us a feasible region. The feasible region explains what all values our model can take. And it also gives us the optimal solution.

To solve this problem, first we gonna formulate our linear program. The final goal is to optimise the profits, so this problem can be solved using LP.

Formulation of Linear Problem

Step 1: Identify the decision variables

The total area for growing Wheat = X (in hectares)

The total area for growing Barley = Y (in hectares)

X and Y are my decision variables.

Step 2: Write the objective function

Since the production from the entire land can be sold in the market. The farmer would want to maximize the profit for his total produce. We are given net profit for both Wheat and Barley. The farmer earns a net profit of US$50 for each hectare of Wheat and US$120 for each Barley.

Our objective function (given by Z) is, **Max Z = 50X + 120Y**

Step 3: Writing the constraints

1. It is given that the farmer has a total budget of US$10,000. The cost of producing Wheat and Barley per hectare is also given to us. We have an upper cap on the total cost spent by the farmer. So our equation becomes:

100X + 200Y ≤ 10,000

2. The next constraint is the upper cap on the availability of the total number of man-days for the planning horizon. The total number of man-days available is 1200. As per the table, we are given the man-days per hectare for Wheat and Barley.

10X + 30Y ≤ 1200

3. The third constraint is the total area present for plantation. The total available area is 110 hectares. So the equation becomes,

$X + Y \leq 110$

Step 4: The non-negativity restriction

The values of X and Y will be greater than or equal to 0. This goes without saying.

$X \geq 0, Y \geq 0$

We have formulated our linear program. It's time to solve it.

Solving an LP through Graphical method

Since we know that $X, Y \geq 0$. We will consider only the first quadrant.

To plot for the graph for the above equations, first I will simplify all the equations.

$100X + 200Y \leq 10{,}000$ can be simplified to $X + 2Y \leq 100$ by dividing by 100.

$10X + 30Y \leq 1200$ can be simplified to $X + 3Y \leq 120$ by dividing by 10.

The third equation is in its simplified form, $X + Y \leq 110$.

Plot the first 2 lines on a graph in the first quadrant (like shown below)

The optimal feasible solution is achieved at the point of intersection where the budget & man-days constraints are active. This means the point at which the equations $X + 2Y \leq 100$ and $X + 3Y \leq 120$ intersect gives us the optimal solution.

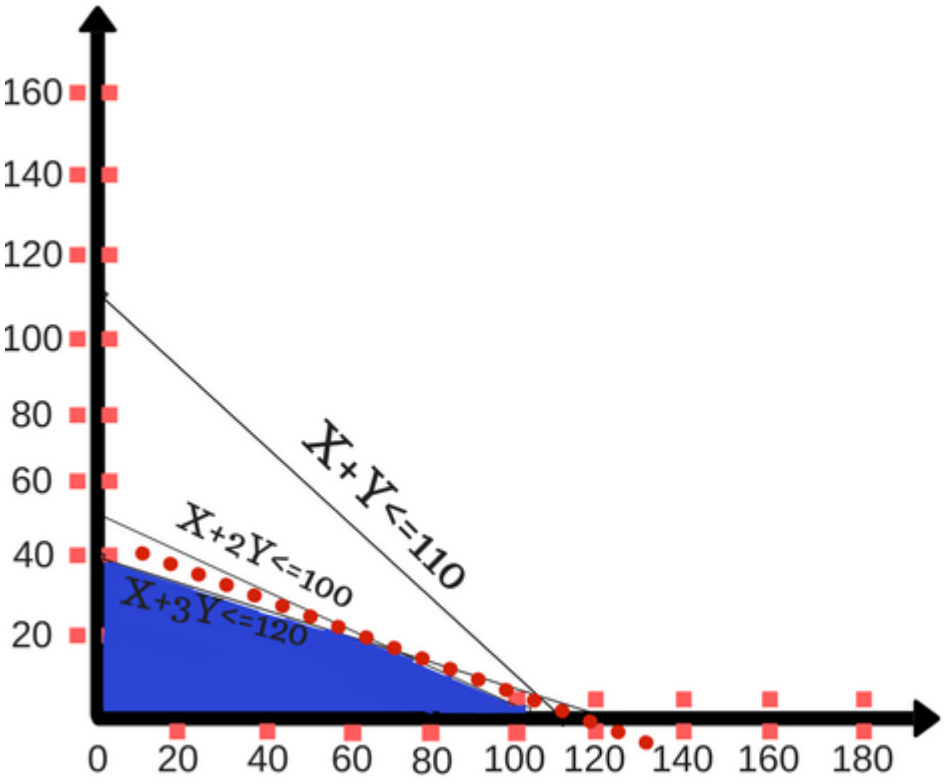
The values for X and Y which gives the optimal solution is at (60,20).

To maximize profit the farmer should produce Wheat and Barley in 60 hectares and 20 hectares of land respectively.

The maximum profit the company will gain is,

Max Z = 50 * (60) + 120 * (20)

= US$5400



I moved onto the next question

Q) **You will be given two polynomials (their order and coefficients in order) return the product of those coefficients.**

Oh! Looks like pretty easy one! Let me do this smaller one and score the minimum marks and lets move forward .Just multiplying and adding the corresponding coefficients should do fine. A for loop inside other one will solve this for me I guess, Opened my terminal and wrote a code and compiled it . I tested it for a sample test case and it is fine .And with a lot of enthusiasm submitted the code and was waiting for the result .It showed a red and seems to be time limit error. Then I was thrown into hell. From childhood I am doing the same method and now what should I do! Put my laptop aside and thought for ways to do it. And I remembered that while learning divide and conquer there was something related to polynomials so I searched for what it could be and found FFT (Fast Fourier Transform) this was exactly what I was searching for let me check about this in the textbook I think this is the solution as Divide and conquer generally involves a factor of logn in worst case time complexity We were told divide and conquer in class I think that should help me .Let us list out known things about polynomials so that they might help me in getting any clue to solve the problem.

Point Value Representation of a polynomial

A(x)=x^3-2x+1

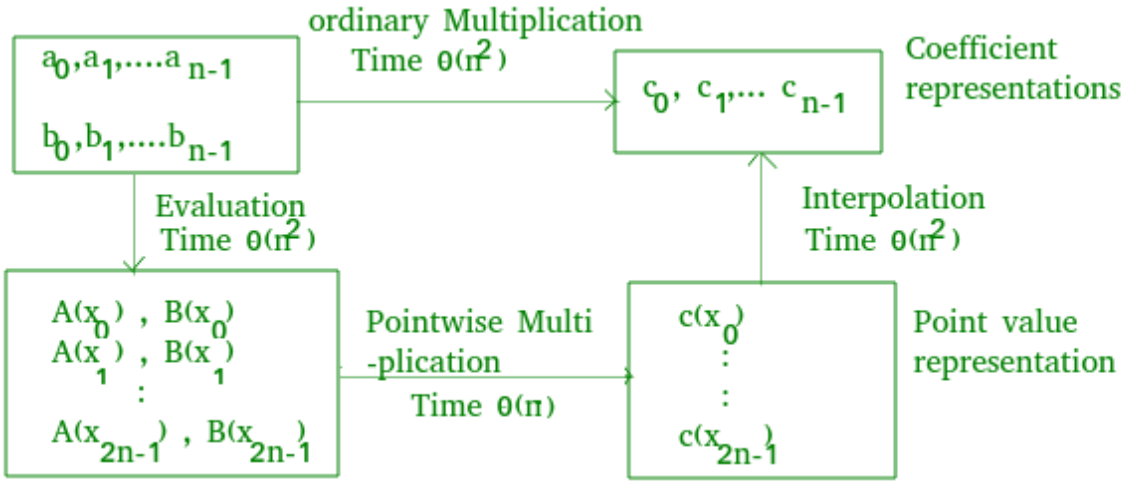
Xi -- 0,1,2,3

A(xi) – 1,0,5,22

Point-value representation of above polynomial is { (0, 1), (1, 0), (2, 5), (3, 22) }

It's just calculation of values of A(x) at some x for n different points, so time complexity is

O(n^2). Now that the polynomial is converted into point value, it can be easily calculated C(x) = A(x)*B(x) again using horner's method. This takes O(n) time. An important point here is C(x) has degree bound 2n, then n points will give only n points of C(x), so for that case we need 2n different values of x to calculate 2n different values of y. Now that the product is calculated, the answer can to be converted back into coefficient vector form. To get back to coefficient vector form we use inverse of this evaluation. The inverse of evaluation is called interpolation. This picture helped me:



This idea still solves the problem in O(n^2) time complexity. We can use any points we want as evaluation points, but by choosing the evaluation points carefully, we can convert between representations in only O(n log n) time. If we choose "complex roots of unity" as the evaluation points, we can produce a point-value representation by taking the discrete Fourier transform (DFT) of a coefficient vector. We can perform the inverse operation, interpolation, by taking the "inverse DFT" of point-value pairs, yielding a coefficient vector. Fast Fourier Transform (FFT) can perform DFT and inverse DFT in time O(nlogn).

DFT

DFT is evaluating values of polynomial at n complex nth roots of unity . So, for ω_n^0, ω_n^1, ω_n^2…… ω_n^{n-1} so for $y_k = \omega_n^k$ k= 0, 1, 2, …, n-1, y = (y0, y1, y2, …, yn-1) is Discrete fourier Transformation (DFT) of given polynomial.

The product of two polynomials of degree-bound n is a polynomial of degree-bound 2n. Before evaluating the input polynomials A and B, therefore, we first double their degree-

bounds to 2n by adding n high-order coefficients of 0. Because the vectors have 2n elements, we use “complex 2nth roots of unity, ” which are denoted by the W2n (omega 2n). We assume that n is a power of 2; we can always meet this requirement by adding high-order zero coefficients.

FFT

Here is the Divide-and-conquer strategy to solve this problem.

Define two new polynomials of degree-bound n/2, using even-index and odd-index coefficients of A(x) separately

$A0(x)=a0+a2x+a4x^2+....an-2x^n/2-1.$

$A1(x)= a1+a3x+a5x^2+....an-1x^n/2-1.$

$A(x)=A0(x^2)+xA1(x^2)$

The problem of evaluating A(x) at $\omega_n^0, \omega_n^1, \omega_n^2 \ldots\ldots \omega_n^{n-1}$ reduces to evaluating the degree-bound n/2 polynomials A0(x) and A1(x) at the points

$(\omega_n^0)^2, (\omega_n^1)^2, (\omega_n^2)^2 \ldots\ldots (\omega_n^{n-1})^2$

Now combining the results by

$A(x)=a0(x^2)+xA1(x^2)$

The list $(\omega_n^0)^2, (\omega_n^1)^2, (\omega_n^2)^2 \ldots\ldots (\omega_n^{n-1})^2$

does not contain n distinct values, but n/2 complex n/2th roots of unity. Polynomials A0 and A1 are recursively evaluated at the n complex nth roots of unity. Subproblems have exactly the same form as the original problem, but are half the size. So recurrence formed is T(n) = 2T(n/2) + O(n), i.e complexity O(nlogn).

And I coded it using this logic it took a while to get submitted and I had no hope because the logic itself is complicated but I was happy that I’m at least in the right direction and to my surprise it worked it said AC in green.

I’m in a good mood now and I went on to do even more

Question: Implement Hamiltonian cycle of a given graph using back tracking. Given graph[N][N] adjacency matrix (has value 1 if edge exists else 0)

Recalls what's Hamiltonian cycle is: It’s a Hamiltonian path with first and last nodes joined (edge exists), which means start at a node and visits every node of the graph exactly once and returns to start node.

As its already given in problem as back tracking, so I will start what are required for backtrack algo. The state space tree, the constraints, terminating conditions.

State space tree is the given graph itself that has undirected edges

Terminating condition or whole goal is to find path that start from 0^{th} node and visits every node and reaches back.

Like in backtracking after adding a node to my answer set, I need to check some conditions right... Those I think are

. It should not be already in our answer set

. It should be our adjacent node to the last node. As the given structure is of adjacency matrix.

….

Then declares the graph as G[n][n], hamiltonian path(ans) as ans vector of size N as I know in prior that the answer will have n vertices

and writes code in main function to take input. And prints the graph again to verify and it succeeds. Now before writing the actual backtracking recursive function let me figure out the check function then

writes the check function like this:

For checking adjacency of new node to last added, I would maintain the no of vertexes in answer set and notes as last_pos so now adjacency checking is just a if loop. And for checking already included or not is also easy: just iterating through answer set.

Pseudo code:

```
bool issafe()
{
   // check if this is adjacent to last added vertex
    if(graph[ans[last_pos-1]][v]==0)
        return false

    // checking if its already included
    for loop on answer set :i
        if(ans[i]== v)
            return false

// if both conds are satisfied then its safe to add the vertex
return true;

}
```

I need to instantiate some things in before

Each element of ans array is declared to some neg integer (as an indicator that ith element is not yet found)

Push 0^{th} node to ans vector as it is the starting point.

Now the recursive loop for Bt.

Loop (last_pos):

For loop for iterating through vertices:v

```
  If ( Issafe(v,last_pos))
  {    adding vertex to path I.e., ans[last_pos] = v;
       If(loop(last_pos+1))
return true;
Else
       Remove the vertex I.e., ans[last_pos] = -1;
 }
  Print( ans vector)
```

Codes this up and tests with few test cases but code doesn't show anything on terminal. Includes some printf statements as to check where he is getting wrong. And observes that his print function at last doesn't show up.

Gets to know it's going to infinite loop as its printing repeatedly, then checks the last_pos value by printf statements, gets the mistake that his code was printing even after finding solution. So adds extra if in start of Loop. That if last_pos = n return true.

But also reminds that he needs to find Hamiltonian cycle not path. So, add one more if to check if there exists an edge between last vertex (ans[last_pos or n]) and start vertex. If present return true in the Loop else false.....

I continued this, I am loving this streak, it everything feels so interesting

Q)Given a character matrix where every cell has one of the following values.

(i)"C" --> The cell has a coin.

(ii)"#"--> The cell is a blocking cell . We cannot move from here.

(iii)"E" --> The cell is empty . We don't get a coin but we can move from here.

The initial position is (0,0) and the initial direction is right.

Following are rules for movements across cells.

If the face is right , then we can move to below cells

(i) Move one step ahead, i.e., cell (i, j+1) and direction remains right

(ii)Move one step down and face left, i.e., cell (i+1, j) and direction becomes left.

If face is Left, then we can move to below cells

(i)Move one step ahead, i.e., cell (i, j-1) and direction remains left.

(ii)Move one step down and face right, i.e., cell (i+1, j) and direction becomes right.

Final position can be anywhere and final direction can also be anything. The target is to collect maximum coins and find the maximum number of coins.

Ans) First of all I am so confused by seeing the question . Didn't understand it when I read it for the first time . So, I read it two to three times to get a clarity about the question . Didn't have an idea as to how to start the solution . I thought I could divide it into small problems and use the solutions of them to solve bigger ones . So, I thought to use Dynamic Programming over recursion as we have no restriction on space and solving it in less is always a better solution . Started thinking how to get the DP expression and what the base cases could be . The basic thought anybody could get after some DP problems is that to create a 2-D array where each element indicates the number of max coins we could get if we start from that element . I thought this approach was correct and started coding it . After initializing some variables I got doubt that "What will be the role of direction in this approach" . I haven't the direction element anywhere in this approach . Now I felt that this approach was incomplete . To get an idea I started reading the question once more . In the middle of question I read a line "**The initial position is (0,0) and the initial direction is right."**. After reading this point keenly I understood a point that the direction in which we start will also matter . So , I thought that if we create a 3-D array where the first two elements indicate the co-ordinates and the third element indicates the direction in which we start . Here the third element will be 1 if the direction is right and 0 if the direction is left.So, now the array looks like Dp[R][C][2] where R is the number of rows and C is the number of columns.

Here Dp[i][j][0] would be an int value which represents the maximum number of coins we can collect if we start from the co-ordinate (i,j) in the left direction . Dp[i][j][1] would be an

int value which represents the maximum number of coins we can collect if we start from co-ordinate (i,j) in the right direction .Let the 2-D array given as input be named as Input[R][C] . So, if Input[i][j] is equal to “#” then Dp[i][j][0] and Dp[i][j][1] both should be zero as we cannot collect coins once we start from that cell as it is a blocking cell .If Input[i][j] is equal to “C” then we initialize the values of both Dp[i][j][0] and Dp[i][j][1] as 1 because we could find a coin int he first slot itself . If Input[i][j] is equal to “E” then we would initialize the values of Dp[i][j][0] and Dp[i][j][1] as 0 because we did not find a coin in that spot but we could move forward .Yes , now I think I have the proper base cases . From the above given instructions we can write the following equations .

Dp[i][j][1] += max(Dp[i+1][j][0],Dp[i][j+1][1])

Dp[i][j][0] += max(Dp[i+1][j][1],Dp[i][j-1][0])

Now the big confusion I got is how to initialize the base cases . This part seemed like a big trouble for me . Suddenly a doubt struck my mind that why didn’t I use recursion . When I thought about it again I understood that the problem was in the time complexity which was exponential . This time complexity was because we were a value many times . Now , an idea struck my mind that why don’t I pass the array to the recursive function , so I could check that if it was already calculated or not . If it was already calculated we would the value that was calculated before . If not calculated we can calculate it now . In this way I could store the values using recursion . After finished the code , I submitted the code and was so happy AC in the first attempt . I felt much excited when I got the idea to merge two concepts which I thought were totally different but I got to know that they were actually very well inter related to each other .

QUESTION:

Consider a chocolate manufacturing company that produces only two types of chocolate – A and B. Both the chocolates require Milk and Choco only. To manufacture each unit of A and B, the following quantities are required:

- **Each unit of A requires 1 unit of Milk and 3 units of Choco**
- **Each unit of B requires 1 unit of Milk and 2 units of Choco**

The company kitchen has a total of 5 units of Milk and 12 units of Choco. On each sale, the company makes a profit of

- **Rs 6 per unit A sold**
- **Rs 5 per unit B sold.**

Now, the company wishes to maximize its profit. How many units of A and B should it produce respectively?

Solution: The first thing I did was represent the problem in a tabular form for better understanding.

	Milk	Choco	Profit per unit
A	1	3	Rs 6
B	1	2	Rs 5
Total	5	12	

Let the total number of units produced by A be = X

Let the total number of units produced by B be = Y

Now, the total profit is represented by Z

The total profit the company makes is given by the total number of units of A and B produced multiplied by its per-unit profit of Rs 6 and Rs 5 respectively.

Profit: Max Z = 6X+5Y

which means we have to maximize Z.

The company will try to produce as many units of A and B to maximize the profit. But the resources Milk and Choco are available in a limited amount.

As per the above table, each unit of A and B requires 1 unit of Milk. The total amount of Milk available is 5 units. To represent this mathematically,

$X+Y \leq 5$

Also, each unit of A and B requires 3 units & 2 units of Choco respectively. The total amount of Choco available is 12 units. To represent this mathematically,

$3X+2Y \leq 12$

Also, the values for units of A can only be integers.

So we have two more constraints, **$X \geq 0$ & $Y \geq 0$**

For the company to make maximum profit, the above inequalities have to be satisfied. Solving the inequalities by finding the intersection of

X+Y = 5

3X+2Y = 12

The intersection is at (2,3). Now all i needed to was solve for the value of the maximum profit that could be obtained which is Z = 6X+5Y. By solving we get that the maximum profit that can be made is Rs 27.

At this point LP was so easy that one of our classmates made this meme, this is true all are same effectively.

This phase had some ups and downs but overall it’s productive and I am in the top 15 in leader board which is an improvement, 14 more I thought.

Phase-3

Only a few people reached to the third level of leaderboard everyone say this is the hard level, it's kinda obvious as this the last level questions *will* be hard, but the plus point is we can take help of prof which sounded cool to me because, his classes are really good and of course he is my favorite prof in the campus.

After classes I went to my room and tried to attempt hard questions, I am in a better total 15 students entered hard mode and I was the 6th student, I want to be on top of leaderboard I have to complete all of these to get there, I told to myself.

Q1: Given a recursive relation $a_n = m*a_{n-1} + k*a_{n-2} + o*a_{n-3}$ you will be given a_0, a_1, a_2 and m,k,o and n you have to find a_n
Constraints are $0<n<10^{18}$

This looks easy all I have to do is find a_n so if I use recursion I am done with this I guess but this won't be that simple I don't think recursion would work I have to think another way did I see anything like this before because it looks quite common oh yes it's similar to Fibonacci series as Fibonacci is also recursion where $a_n = a_{n-1} + a_{n-2}$ recursion didn't work there so it won't work here too so I have to use DP so basically I have to store values of all a_n's and store them in an array so I can find it for the n they asked it's easy then let me code it.

Defining an array and taking m,k,o as inputs then n then a_0,a_1,a_2 as inputs so, Array[0]=a_0 , Array[1]=a_1, Array[2]=a_2. Now a for loop with Array[i] = m*Array[i-1] + k*Array[i-2]+o*Array[i-3] would do at the end we have to print Array[n] and it's done submitting. It's taking more time than usual, it said RE in red.

I took my phone and messaged Rudra asking what RE meant he replied Run Error this occurs when there is a segmentation fault it happens when you try to access unallocated memory or there is a memory overflow.

Ok I am not accessing unallocated memory there could be a memory overflow let me see the constraints it's 10^{18} in the worst case this won't work I think I can't declare an array of size 10^{18} as the limit is 10^8 so that is the problem so I can't do anything with arrays so DP fails here.

No problem let me write the naive recursion one there won't be any arrays there so it's basically returning except when n is 0 1 or 2 that's it let me submit it, still got RE. I started to think for a while how did I get RE then I just realized how dumb I am I use more memory in

naive recursive way than DP approach as to know a_3 it does a set of operations and when a_4 needs a_3 to find it does the same thing again so I end up using so much space this is exponential at least in DP I just used an array. I couldn't find a single way which is shorter than O(n) but the solution demands O(logn).

I knew that this had something to do with divide and conquer but I couldn't get any idea, I was on this problem for a day there was no proper way to calculate it exactly, there is an approximate method but the problem was at $10^{>10}$ these approximations are no longer approximations these are way wrong.

After wasting few more hours, I gathered my courage and decided to ask professor for a hint, I was a bit shy but the professor was welcoming. I introduced myself, "Hi I am Jaishnav, student in the algorithm's course.".

Actually sir was in a hurry but he replied politely, "Hello, I remember your face you always sit in that corner right", as he pointed towards left.

I smiled and nodded, Sir continued, "Why did you come?, any question from class or leaderboard problems".

By looking at sir at this point I thought, Sir won't answer leaderboard problems he is always interested about real life questions and others based on the way he teach but anyway I gathered the courage again and replied, "leaderboard".

Sir looked jovial and he was smiling and said, "Those are also algorithms, Don't worry I'll help you out", he probably understood what I thought, sir continued "Which level?"

"Level 3 sir", I replied

Sir looked happy and asked, "Which question?"

"Finding a_n in the given recursion question or the first one sir", I replied.

Sir thought for a while and looked at me and asked, "there is a problem and you aren't able to solve it what would you do?"

I started to think why is he asking me this, I shouldn't have asked I just came here to embarrass myself, So I politely replied, "Try again."

"You are not in some self-help workshop", Sir replied and laughed and I smiled sir continued, "It failed now you have to do it anyway, you have unlimited resources but you couldn't find the solution for the given question. What would you do?"

I thought for a while and replied, “I will break that question and using the resources I have I’ll see if I can do it in parts.”

Sir laughed and replied, “Did you honestly think that or just because I teach algorithms, you thought of divide and conquer approach?”.

I smiled and replied, “This was intuitive sir, Honestly I realized it was divide and conquer only after you said.”

Sir smiled and asked, “Well that didn’t work, what now?”

I thought for a while and replied, “I’ll try to search solutions for similar problems and understand them and try to solve this.”

Sir replied, “Yes that’s what you have to do try to solve some similar question quickly.” And he left saying bye.

I honestly didn’t think it would happen like that the conversation was so funny and so interesting I wanted to have more.

I shifted my focus back to solving the problem and went to workspace I got the idea so whatever this O(logn) way we should be able to find nth fibonacci number too in this way let's search textbook or internet we might find a way then I went on to google fastest way to find nth fibonacci number. Then I saw some link with something to do with 2D matrices and I was excited and clicked it.
Rudra shouted “I found it, we have to do it by matrix exponentiation” I replied “Yeah I am also near something with matrices I’ll read this you read that lets discuss later.” the link opened it gave the first two methods which I did and they fail in this case so the 3rd one was taking a 2D matrix there was this picture

$$\begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} * \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

And similarly:

$$\begin{pmatrix} F_2 \\ F_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} * \begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^2 * \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

Which in general turns into:

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n * \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

We know that F_0 and F_1 = 0, 1 so we can put the whole thing into python and see how it fairs compared to our polynomial algorithm

So they wrote it as a product of two matrices and each can be written in that way so for the n^{th} fibonacci number we need to find that 2X2 matrix raised to n and multiplied to a_0 a_1 . So the problem is reduced to finding the nth power of that matrix. By this time Rudra said exactly the same thing I asked him "It's the same anyway right finding the n^{th} power will also take O(n) right?" He replied casually "We can do it in O(logn) using divide and conquer. Sir didn't cover this in class but it was in a textbook I read it. It's simple, So the idea is basically finding x^4 you can multiply x 4 times or find x^2 multiplying x and x then find x^4 by multiplying x^2 and x^2 the code is something like this" he took a paper and started showing the code.
It looked like this.

For x^y

```
int res = 1;
while (y > 0)
{
    if (y % 2)
        res = res*x;
    y = y/2;
    x = x*x;
}
return res;
```

He started explaining assume x^7 as 7 is odd res will be x, 7 becomes 3 x becomes x^2 now 3 is also odd res is $x*x^2$ which is x^3 y becomes 1 x becomes $x^2 * x^2$ which is x^4 now as y is odd res becomes x^3*x^4 which is x^7 and y becomes 0 so the loop breaks after x becomes x^8.
Now res has the answer we need. We got x^7 as $x^1*x^2*x^4$ applying this to a matrix we will get $A^{10\text{^}18}$ in O(logn) itself this is a very big improvement. we have to do it, all that is left is finding A for our question as the question is not about fibonacci.

$$\begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \end{bmatrix} = \begin{bmatrix} m & k & o \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ a_{n-3} \end{bmatrix}$$

$$\begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \end{bmatrix} = \begin{bmatrix} m & k & o \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}^2 \times \begin{bmatrix} a_{n-2} \\ a_{n-3} \\ a_{n-4} \end{bmatrix}$$

$$\begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \end{bmatrix} = \begin{bmatrix} m & k & o \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}^{n-2} \times \begin{bmatrix} a_2 \\ a_1 \\ a_0 \end{bmatrix}$$

So we found our A we have to just raise it to n-2 using the new exponentiation method we are done it's easy let me code and it's done working for the given cases "I am submitting it and its running and it got accepted". I wondered now I can know 2^20 th fibonacci number in such less steps this was really interesting.

Then I moved onto the next question with some positivity because, I and Rudhra solved it first among the 15 members.

Q2:

N-Queen's: Place N queens on NxN chessboard such that no two queens attack each other.

How could I do this.... is it by checking each combination possible? I would try it for small numbers. (Then he grabs a book and pen). Started drawing 4*4 matrix and tries to place queens. Marks few boxes as queen and after marking 2 or 3, I can't place the next one.... starts again. And after few trails I came to know that this randomness doesn't work so starts by placing the first queen at (0,0) and tries if it fails, I will place in (0,1).(iterating through first column). After each wrong combination I observed changing the before placed queen saves time instead of going to start … It reminded me of backtracking concept. And I was sure that it's back tracking problem. And saw that it was mentioned in notes that checking all combinations would lead to 10-digit number for even 8*8 chess board. And as mentioned in the book I was trying to modulate the problem as Fixing of tree structure, checking of state.

First let me fix the tree structure. While trying on paper I was iterating in first column for 1st queen placement. So, root node is empty, first level is (0,0), (1,0), (2,0) … and from each node going to boxes in second column. (0,1), (1,1) …. And then next level is next column....

For checking of state: I need to check whole matrix that no two queens attack each other. first check on no two queens in same row then same column and then same diagonal. And after some rough work on paper came up with Pseudo code for check function that goes like this:

```
bool checkfn(row,col)
{    //checking in row of suggested(input) cell
     For loop (i is 0 to col)
          if (arr[row][i])
               return false
    //checking in column of suggested(input) cell
     For loop (j is 0 to row)
          if (arr[j][row])
               return false
    // checking through left diagonal
     For loop ( i=n-1, j=n-1;...;i--, j--)
      if(board[i][j])
       return false
    //checking through right diagonal
     For loop ( i=0, j=n-1;...;i++,j--)
      if(board[i][j])
       return false

 return true
 }
```

Now for main part iterating through the tree.

As already thought of solving level by level (column by column). Searches for ways of iterating through matrices and finally decides to iterate through cells of column pick one and do check fn on it if it's true go for next column. If no cell in present column returns true for check fn , the recursive loop goes for changing position of last placed queen and continues the process. Gets the idea of (back tracking).

So, coded everything up and tests for few test cases and it works (hurray!!!). and submitted the problem, but got TLE, what could be wrong: am I iterating through cells more than required, I wondered. And checked my code but found no mistake.

I wasted almost 4 hours debugging but couldn’t find a single mistake nothing was unnecessary I felt before asking for hint I decided to check it once more and then I found my mistake and screamed thinking how was I that dumb.

I was checking for not filled columns also in check function. Like my code was wasting half of its time in checking in levels that are below by present level.....

So I modified the check function so it checks only the cells that are below present column....

Changes was made to diagonal checking loops those are:

```
// checking through left diagonal
 For loop ( i=row , j=col; i>=0 && j>=0;i--, j--)
  if(board[i][j])
   return false
//checking through right diagonal
 For loop ( i=row, j=col ; i<n && j>=0;i++,j--)
  if(board[i][j])
   return false
```

And finally, this was accepted. I am relaxed now.

I looked at the leaderboard and currently I am second if I solve the next question I would be the topper so I put everything aside and I am determined to solve the next question so I can be on top.

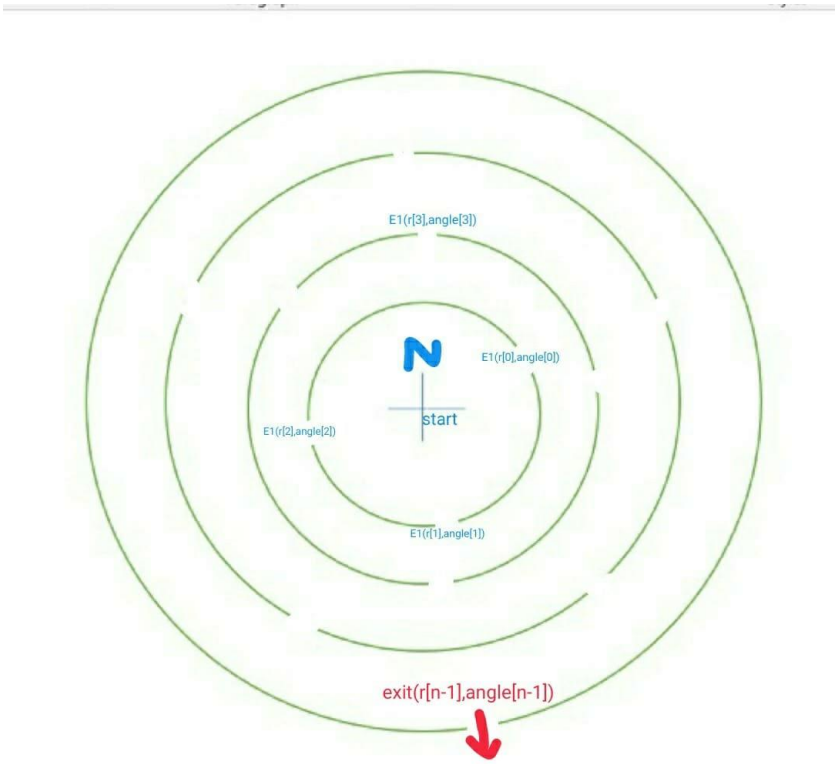
The next question was:

Q3: You are in the middle of a desert with very less amount of water where you found the clue of the next mystery with the clue some walls rise which are circular and concurrent with increasing radius linearly and each has some exits. Whose locations are given to you in terms of radians due north in the map as you have very less water need to find fastest exit so as to survive. Let d be the distance travelled by him during exit return d.

Total number of exits is N, and position of the exits are given by their radius array and angle with the north r[N],angle[N], where for each i , 0<=i<N,r[i],angle[i],corresponds to the position of the exits 2<N<10^5 0<=angle[i]<=2*pi.

“This looks way different and odd, did you understand it?”, I asked, Sucheta replied, “Well yes it looks like we can solve it using greedy choosing the quickest exit every time then we are done I guess.”, I asked her to explain it clearly as I wasn’t understanding it she said, “You are in a desert with less water and you have to go out but here there are gates for exit at random, there are multiple walls, like the ones in attack on titan except their exits are not on same line now assume you went along north, and there is an exit for the next wall there won’t be any exit for it now assume there is next exit at a radians to the north, we have to cover r[i]*a length and then exit here the constraint is the water is limited so assuming he walks at some speed all we have to do is reduce his walking time.”, I was about to say

something she cut me in the middle and said, “Wait a minute I’ll draw this it would be clear” she said while she was drawing and she showed me this:



I said, “Oh thanks that was so clear”. I was convinced choosing quickest exit every time would mean going fast so just coded it used a variable called distance and added it every time as I choose the nearest exit to go out and returned it then all of a sudden I realized that choosing exits here would affect my next chances like choosing the farther one now might be useful for future, I realized with simple example assuming 3 walls and the first one having in front and the next two are at 179 and 181 degrees from north if he had travelled and took the south door initially then he covered small distance so choosing best every time is not true, I turned to her to say that it’s wrong and she said, “Yeah just got WA and I realized? What could we do? Does this have anything to do with DP?”
I said, “I don’t think that anyways you try it in that way, I’ll look for another way.” She left.

I tried DP approach it was giving correct answers but there was time limit so got TLE.

I sat there for long time didn’t get any other idea this looked so strange and odd, I felt like I was wasting more time so I decided to go and take Professor’s help as I didn’t have a better idea.

I walked towards the faculty room, I am not nervous now as it's second time but somewhere deep down I had a feeling like this is my second time what would he say but I met him anyway as I have to solve it.

I looked at him and stood there politely, sir looked me and waved his hand, I was a bit shocked that's so cool, I never expected that, I waved back smiling.

Sir came towards me and said, "Hey Jaishnav, How's everything?"

"Fine Sir, just struck on a question."

"Then use your resource, I am here to help", Sir said.

Sir was so friendly if we didn't have age barrier we could become good friends, however if we don't have age difference Sir couldn't be the prof. "Sir I am stuck on the desert question, It sounded odd for me and I know greedy fails, DP takes longer time I don't think there is any other way sir".

Sir smiled and replied, "Of course there is a way, first answer this question take a paper and pen I'll tell you the question."

I quickly took the nearby paper, there was a pen in my pocket, and I was ready I hoped he would ask me some similar question so I could solve this.

Sir opened his mobile, and read the question out loud, "The German city of Königsberg (present-day Kaliningrad, Russia) is situated on the Pregolya river. The geographical layout is composed of four main bodies of land connected by a total of seven bridges. is it possible to take a walk through the town in such a way as to cross over every bridge once, and only once?"

This question sounded way odd and it's a logical question I didn't understand why he asked this question, So to answer it I tried all possibilities and then I replied, "No sir it's not possible."

Sir smiled and replied, "Yes you are right show me your paper." Sir said and moved his hand to ask the paper I gave it to him.

He looked at it and said, "So you checked possibilities and said no right?".

I replied, "Yes sir, why did you ask this question, what does this have to do with algorithms"

Sir smiled and replied, "Technically this isn't hard question the way it's formulated is hard."

I thought, He is probably talking about circular coordinates instead of normal coordinates and replied, "Does the circular co-ordinates make this hard?"

Sir looked at me and he expected this question and said, “No.”

Sir gave the paper back to me and said, “Try this in other way, better mathematically so that you could code it.”

Why should I code it, anyway I decided to follow sir and thought for a while and then it struck to me the lands are nodes and the bridges are edges of a graph. I replied, “Well the lands can be considered as vertices and bridges as edges.”

Sir nodded his head in agreement and said, “Well that’s it, this was a famous riddle in 1700s and Euler tried to solve this and apparently and this was the foundation of Graph Theory. See we can use graph theory to solve real world problems like this one not the other way round.”

I was shocked by this, Now I can see the question in a completely new perspective, I thought. And replied, “So here all exits are nodes and the edge weights is the distance between those” as I said this the answer was in my mind I could apply Dijkstra and it would be O(E*log(V)) and that will be accepted definitely.

Sir smiled and replied, “Yes that’s correct you got it.”

Sir continued, “I gave this question to you so you can appreciate graph theory and how it can be used to solve many real world problems, there is something called networks explore if you are interested by this, they are so interesting there are so many interesting algorithms and questions which you can learn with the basics of graph theory which you know.”, Sir asked, “for example have you used LinkedIn”?
I replied, “Yes sir”,

Sir replied, “Well not just LinkedIn any social media uses graphs or you can call it networks here and help us. It’s not AI or ML which does everything graphs are integral part of it. That’s the basis of how they are run.”

This was fascinating to me, “With my current knowledge could I explore?”, I asked.

“Of course, you can”, Sir replied laughing.

At this point I am more interested towards graph theory than the leaderboard problems but then it struck to me that I have to finish the question so I could top the leaderboard.

I went back and coded it using Dijkstra from the logic that it’s a graph and it got accepted now I am first and no one did the second question yet, I did the third question too, I was happy now.

I didn’t feel like starting another question instead tried to make sense of what sir said, and I decided to start to learn something about networks/graphs.

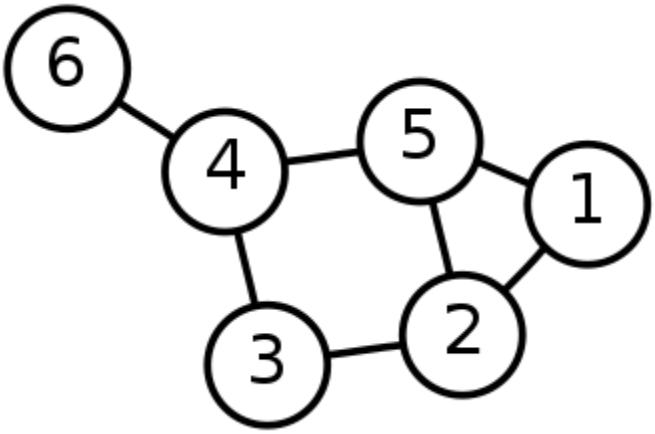
The conversation I had with prof when he told me that what we learnt in graphs is just basic, currently it's so advanced and sophisticated as many problems are simple only after you realize how we can form a graph based on the information we have, we start seeing things differently so I went on to google and searched about it, most networks can be described as graphs and we can apply algorithms to get answer, there was one thing which caught my eye it was called networkx which is used at many places along with graphs, I was curious to see what it was and also how sophisticated it is as it was used at almost every place.

After reading it's wiki and some of it's documentation I got the idea of what it was, it is a python library for studying graphs and networks, "Graphs and Networks?", I said it out loud, what is the difference between those aren't they the same, I thought.

I started digging deep into these and there are two arguments, one saying both are the same others saying no they aren't. I started to go by definition, expecting it would clear things.

Definition of graph: A **graph** (sometimes called *undirected graph* for distinguishing from a directed graph, or *simple graph* for distinguishing from a multigraph) is a pair $G = (V, E)$, where V is a set whose elements are called *vertices* (singular: vertex), and E is a set of paired vertices, whose elements are called *edges* (sometimes *links* or *lines*).

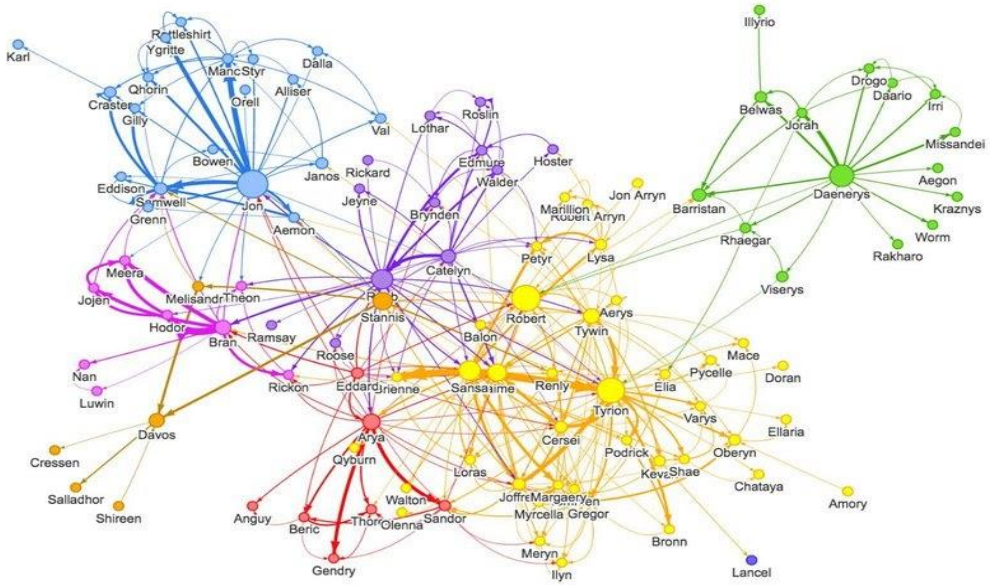
The vertices x and y of an edge $\{x, y\}$ are called the *endpoints* of the edge. The edge is said to *join* x and y and to be *incident* on x and y. A vertex may not belong to any edge. (According to wikipedia)



(this is a graph)

Networks is a vast topic apparently and there is a very generalized definition for networks but the place where I am interested is called Network theory or Network analysis to be more specific.

Definition of Network (according to Network theory): **Network theory** is the study of graphs as a representation of either symmetric relations or asymmetric relations between discrete objects. In computer science and network science, network theory is a part of graph theory: a network can be defined as a graph in which nodes and/or edges have attributes.



(all nodes are not identical here and they can have different attributes)
I realized all of a sudden that weight of an edge is indeed an attribute of the edge,So they are essentially the same; we just use graphs to analyse networks here like, breaking a network topology into graphs so we can solve problems more efficiently also using all the advancements of graph theory to our advantage.
Albert-László Barabási, one of the most famous network scientist, said, "**In the scientific literature the terms network and graph are used interchangeably:**

NETWORK SCIENCE	GRAPH THEORY
Network	**Graph**

Node	**Vertex**
Link	**Edge**

Yet, there is a subtle distinction between the two terminologies: the {network, node, link} combination often refers to real systems: The WWW is a network of web documents linked by URLs; society is a network of individuals linked by family, friendship or professional ties; the metabolic network is the sum of all chemical reactions that take place in a cell. In contrast, we use the terms {graph, vertex, edge} when we discuss the mathematical representation of these networks: We talk about the web graph, the social graph (a term made popular by Facebook), or the metabolic graph. Yet, this distinction is rarely made, so these two terminologies are often synonyms of each other."

As I was reading more I found this, "Graph theorists are interested in arbitrary questions about graphs, whereas the network theorists are more interested in questions that are relevant to the situations that are modeled by networks or real life things.". I am not completely sure if I understand this because the question was about the difference between graphs and networks, not the difference between what they do with those or probably the only difference could be what they do with those, I wondered.

I went back to the networkx documentation to try to make some sense out of it and then I found it was like all other python libraries simple, easy to use, most of the instructions are pre coded so we can call it, one interesting thing about those was most of the time it assumed the graph is stored as list of edges, to be specific it was ordered pair of vertices of edges not adjacency list or adjacency matrix, however there are inbuilt commands which can convert it into those anyway. I probably should explore these little more, there are definitely so many advancements here, Sir was right.

As I was reading further about the Network theory, I found this, "Network Theory has been used to analyse functional MRI data: after segmenting the brain in a set of smaller regions acting as nodes, and using BOLD signal of those regions as their attribute, edges are created dynamically with a strength attribute equal to the correlation coefficient between two node activities.", there can potentially be many advancements we can make as we have more data and more efficient algorithms to solve for the problem.

Well the question of whether they are the same or not is in some way or another like a difference between a filmmaker and director, the former one sounds cool but honestly I don't think there should be a difference just like here but there is.

This was so interesting and I am excited, but I prioritized leaderboard questions because sir thought a lot before giving questions and I want to learn more so, I decided to solve the leaderboard question initially.

Q3:Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible tour that visits every city exactly once and returns to starting point.

For example, consider the graph shown in the figure on the right side. A TSP tour in the graph is 0-1-3-2-0. The cost of the tour is 10+25+30+15 which is 80.

The methods I can think of are Naive and dynamic programming, Approximate solution using MST, Branch and Bound.

In the Naive and dynamic programming method:

Naive Solution:

1) Consider city 1 as the starting and ending point.

2) Generate all (n-1)! permutations of cities.

3) Calculate cost of every permutation and keep track of minimum cost permutation.

4) Return the permutation with minimum cost.

Time Complexity: O(n!)

So naive solution isn't good.

Let the given set of vertices be {1, 2, 3, 4,….n}. Let us consider 1 as the starting and ending point of output. For every other vertex i (other than 1), I found the minimum cost path with 1 as the starting point, i as the ending point and all vertices appearing exactly once. Let the cost of this path be cost(i), the cost of corresponding Cycle would be cost(i) + dist(i, 1) where dist(i, 1) is the distance from i to 1. Finally, we return the minimum of all [cost(i) + dist(i, 1)] values.

To calculate cost(i) using Dynamic Programming, I thought of having some recursive relation in terms of sub-problems. Let us define a term C(S, i) to be the cost of the minimum cost path visiting each vertex in set S exactly once, starting at 1 and ending at i.

I started with all subsets of size 2 and calculated C(S, i) for all subsets where S is the subset, then I calculated C(S, i) for all subsets S of size 3 and so on. I also confirmed that 1 must be present in every subset.

For a set of size n, I considered n-2 subsets each of size n-1 such that all subsets don't have nth in them.

Using the above recurrence relation, we can write dynamic programming based solution. There are at most O(n*2n) subproblems, and each one takes linear time to solve. The total running time is therefore O(n2*2n). The time complexity is much less than O(n!), but still exponential. Space required is also exponential. So this approach is also infeasible even for a slightly higher number of vertices.

So I think I should try an approximate solution using MST. But for approximate algorithms to work the problem instance should hold triangle equality.

Triangle-Inequality: The least distant path to reach a vertex j from i is always to reach j directly from i, rather than through some other vertex k (or vertices), i.e., dis(i, j) is always less than or equal to dis(i, k) + dist(k, j). The Triangle-Inequality holds in many practical situations.

When the cost function satisfies the triangle inequality, we can design an approximate algorithm for TSP that returns a tour whose cost is never more than twice the cost of an optimal tour. The idea is to use a Minimum Spanning Tree (MST).

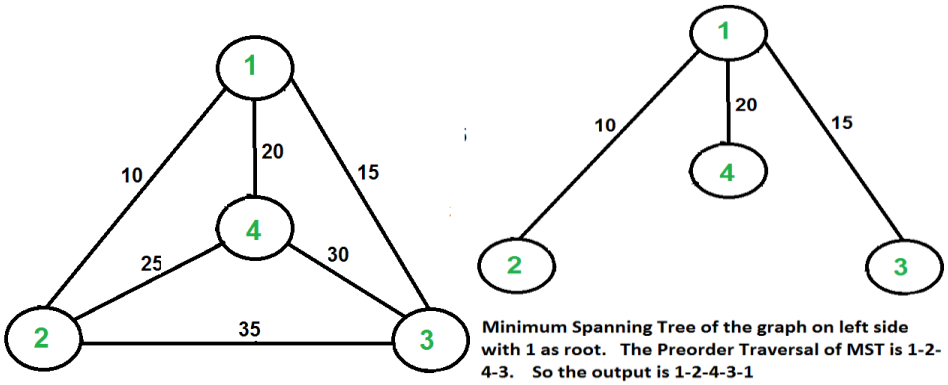
Algorithm:

1) Let 1 be the starting and ending point for salesman.

2) Construct MST from with 1 as root using Prim's Algorithm.

3) List vertices visited in preorder walk of the constructed MST and add 1 at the end.

Let us consider the following example. The first diagram is the given graph. The second diagram shows MST constructed with 1 as root. The preorder traversal of MST is 1-2-4-3. Adding 1 at the end gives 1-2-4-3-1 which is the output of this algorithm.



Minimum Spanning Tree of the graph on left side with 1 as root. The Preorder Traversal of MST is 1-2-4-3. So the output is 1-2-4-3-1

In this case, the approximate algorithm produces the optimal tour, but it may not produce optimal tour in all cases. So this is also not a very good method.

So I moved onto the branch and bound. In the Branch and Bound method, for the current node in the tree, I computed a bound on the best possible solution that I could get if I down this node. If the bound on the best possible solution itself is worse than the current best (best computed so far), then I ignored the subtree rooted with the node.

The cost through a node includes two costs:

1) Cost of reaching the node from the root (When I reach a node, I have this cost computed).

2) Cost of reaching an answer from the current node to a leaf (I compute a bound on this cost to decide whether to ignore subtree with this node or not).

Now I have an idea about computation of lower bound. So now i have to think about applying it for a state space search tree. So I started enumerating all possible nodes.

1. The Root Node: Without loss of generality, I assumed that I would start at vertex “0” for which the lower bound has been calculated.

Dealing with Level 2: The next level enumerates all possible vertices I can go to (vertex has to occur only once in any path) which are, 1, 2, 3… n. Consider that i’m calculating for vertex 1, Since I moved from 0 to 1, our tour has now included the edge 0-1. This allows me to make necessary changes in the lower bound of the root.

Lower Bound for vertex 1 =

Old lower bound - ((minimum edge cost of 0 + minimum edge cost of 1) / 2)+ (edge cost 0-1)

As we move on to the next level, we again enumerate all possible vertices. For the above case going further after 1, we check out for 2, 3, 4, …n.

Consider lower bound for 2 as we moved from 1 to 1, we include the edge 1-2 to the tour and alter the new lower bound for this node.

Lower bound(2) =

Old lower bound-((second minimum edge cost of 1+minimum edge cost of 2)/2)+ edge cost 1-2)

The only change in the formula is that this time I included second minimum edge cost for 1, because the minimum edge cost has already been subtracted in previous level.

This works fine and got accepted.

I thought of continuing graph/networks as there is only one question left and that is so interesting and I am not able to stop thinking about it.

I sat down to learn network theory so that I can appreciate the advancements of graph theory and how they helped us and the honest reason is because it's so interesting,

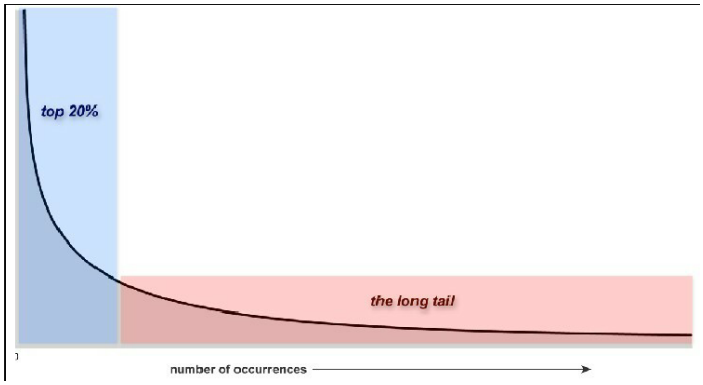
In the graph theory which I learnt we could represent the graph using adjacency list, adjacency matrix, list of edges. Here also it's the same effectively however nodes and edges all need not be numbers so we have to hash them for the list of edges. Just like graph theory there is no one good thing as such to do here, based on the use and situation we have to change the representation so that we can solve the problem efficiently.

To my surprise here there are Network Structures these characterize the network, so we can have an empirical understanding of them they are:

1. Number of Edges
2. Degree distributions
3. diameter
4. Clustering coefficient

The first one **Number of edges** says whether the graph is sparse or dense, if the edges are less it's a sparse graph if they are more then it's a dense graph this is just to get idea of the network while solving for any problems or applying algorithms as higher complexity algorithms can be applied to sparse graphs efficient ones are applied for dense graphs.

Degree distribution: it's basically a graph whose x axis is number of neighbours and Y axis is the number of vertices with that degree, plotting it is simple initially we have to find degree of each vertex and then creating an array H and for each vertex we have to increase H[dgrofV]++. Now we have to plot 0 to n-1 (degree) as the X coordinate and Y coordinate everywhere is H[i]. It is generally plotted on a regular log-log scale. There are two possibilities here it can be a straight line and it can't be a straight line. One interesting property is called long/heavy tail which looks like this



If the graph has a long tail then it will obey power law as in slight changes would reflect the changes as proportional to power of the initial value, this Degree distribution would help us in knowing how changes in few things might affect the overall state of the network

Diameter: It's defined on a network it's the maximum of all the minimum distances between any two nodes that is after calculating all pairs shortest path the path with maximum length in all those is the diameter, so the diameter gives the idea of how near or far two nodes in a graph, that is if the diameter is high it means the nodes are scattered if it's low all are nearby.

Clustering Coefficient: The idea of clustering is to group things together clustering coefficient talks about the extent to which this can be a cluster based on the idea that, if there are more triplet nodes (or technically triangles formed by nodes) compared to number of paths with 2 edges then the more this ratio is the greater is the clustering coefficient there is no proof of how they arrived but based on the set of examples there I am convinced that it's true and also it's not the only way to be sure but the other way is correct higher clustering coefficient means higher chance that it could be a cluster as there is higher node density.

C = 3* No.of_traingles / number_of_2_edgepaths

Finding the denominator is easy here because if there are paths with 2 edges it means one of the 3 nodes is at the centre so it's basically finding the number of paths which exist with each of the vertices as the centre. Number of paths with length 2 edges having vertex v at the middle are, given deg(v) is it's degree ${}^{deg(v)}C_2$. Sum of all this for every V belongs to a set of vertices is the denominator.

And nothing was explicitly said about the numerator but the question was interesting finding number of triangles in the graph and I took my notebook and pen out and tried to figure it out how to find them, the first thought was the naive method to solve, this was pretty basic, for every triplet a,b,c in the given graph we have to check if they form a triangle that is are they adjacent to each other if they are then we have to increase the count if they form a triangle this is naive and the time complexity is $O(n^4)$. As there will be 4 for loops one two move along the graph and other 3 are just to check adjacencies of a,b.c, I thought for a while, I was sure there is a better algorithm but I couldn't get any idea to optimize it so, it was back on my mind as I moved further while learning.

Now there are even more definitions I started noting them down,

Distance: length of shortest path between two nodes is called distance.

Eccentricity: length of the longest shortest path from a vertex v is called its eccentricity.

Then I realised diameter is nothing but maximum eccentricity, and I started to think how to find diameter.

If we just go by definition finding diameter is basically finding shortest from the output of all paths shortest pairs output so the initial idea was to run a floyd warshall algorithm which returns all pairs shortest paths and finding the smallest of these would work so it's $O(n^3)$ for floyd warshall followed by $O(n^2)$ for finding the minimum of the returned 2D array so effectively it's $O(n^3)$ this might not be the best approach as we can definitely optimize this.

I started reading to see if there is a better way and the given approach was also brute force technically the idea was just slightly different here rather than finding all pairs shortest path here it was to find, eccentricity of each vertex and return maximum of them, there are n vertices and finding eccentricity we have to do bfs and find the max so it takes $O(n^2)$ we have to do this for each so the worst case complexity is $O(n^3)$ which is better in no way so both Floyd Warshall and this approach won't help us in any way.

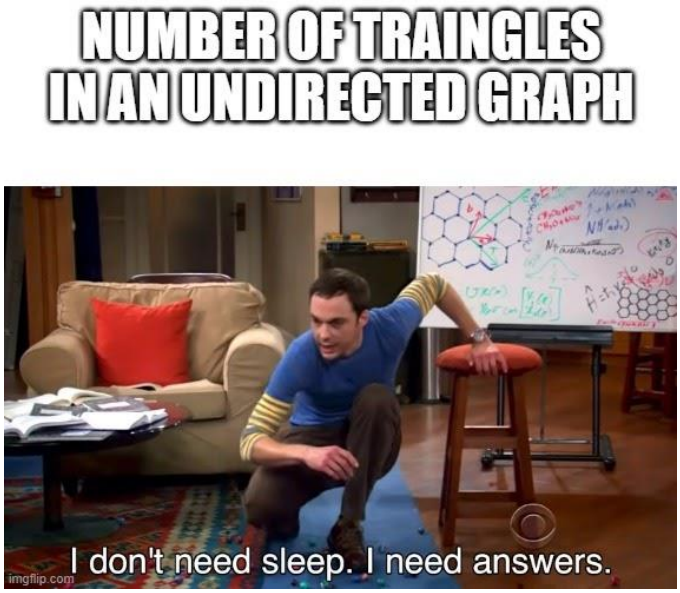
Luckily there was a better approach which followed, it was starting from a random vertex and setting the value of diameter to be 0 and start from random vertex a, let b be the farthest vertex with distance l, if distance from a to b that is l is more than the current value of diameter then update the value of diameter to l and start dfs from b and repeat the process if it's less then we can break and that would be the diameter, this is a faster approach comparatively. If the diameter is small then the network is said to be exhibiting small world phenomenon which literally means how the world is small as everything is nearby.

I didn't learn about most of these, I thought while something was at the back of my mind thinking how to find the number of triangles in the given graph.

The end of the year is approaching so I decided to finish the last question to see what else sir has added and do it.

As I entered workspace to solve everyone was congratulating me, I didn't understand why and Manideep called me and said, "Congrats you are the topper of the leaderboard, I think you didn't shout at the start saying everywhere that you would top the leaderboard", as he

looked at the Jeeshanth, he was embarrassed and feeling sad, but he somehow still believes he could top the leaderboard at this point that leaderboard didn't fascinate me the only thing on my mind is finding the number of triangles in a given graph, Nothing more.



I sat down to solve the final question it's on graphs but the odd thing was, I already understood it's on graphs the question gave this away, I wondered what sir want us to make out of this and started

Q4) Given a weighted and directed graph , find the shortest path between every pair of nodes in the graph . Each edge will be given as a line to the input .

Ans) I know how to use the greedy approach to find shortest path but that only finds shortest paths from a single source node . That greedy approach is also known as Dijkstra algorithm.

So, I thought of applying The greedy approach from aa nodes so I could find the shortest path between all pairs of nodes . But I think that will become a naive approach and ther might be a better way to find a solution . I don't know how to start the solution and where to start from . I only know a few techniques now . So, I started thinking if any of the technique that I learned will be helpful or not . I did not get any idea how to start and what to use. I think I need to ask professor for a clue.

I am struck here basically there is no other option for me but to ask sir, but somewhere I felt that attempting it again with fresh mind would help so, I thought that to be true so stopped doing the question and left the space.

I reached my room only to continue studying so I could find the clustering coefficients numerator which is basically finding number of triangles in a given graph.

I continued learning network theory, that counting triangles can be done better I just don't know how as of now, I thought as I started, and I found that the current topic is clustering coefficient algorithm, which effectively means finding the numerator of clustering coefficient which I am struck at as I couldn't find a better way.

As expected the first algorithm was the naive $O(n^4)$ algorithm which is quite intuitive and that's what I expected too, then there was this second algorithm which is putting every edge of the graph (v,w) into a hash table H, so we can check directly if there exists an edge between (v,w) in O(1) time.

One of the main advantage here is initially we choose a vertex and in the adjacency list we choose any 2 vertices out of those adjacency lists so it's ${}^{n}c_2$ in worst case so of all these we check if there is edge if there is then we increase the counter, there isn't any great jump but $O(n^3)$ is better than $O(n^4)$ anyway.

The other way uses adjacency matrix representation it is this:

If we compute An for an adjacency matrix representation of graph, then a value An[i][j] represents number of distinct walks of length n between the vertex i to j in graph, so finding A^3 and finding sum of all elements is the denominator only, so all that is needed is finding A^3 which can be done in $O(n^3)$ and finding sum of elements is $O(n^2)$ so effectively this algorithm also takes $O(n^3)$ if we use Strassen's Matrix multiplication algo it would become $O(n^{2.84})$.

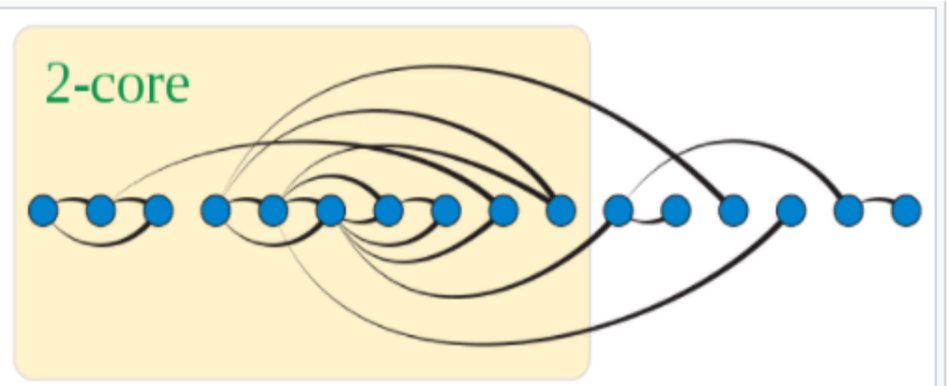
One thing that puzzled me was, how does $A^3[i][j]$ represent the number of distinct walks between i,j. Of course A[i][j] does represent distinct walks of length 1 which are basically edges the proof is probably by induction I guess let me check,

For example, the number of walks of length 2 is the number of vertices k such that there is an arc from i to k and an arc from k to j. And this is exactly the i, j entry in A2 , by the definition of matrix multiplication. So the definition of matrix multiplication itself does this, I didn't know this before, How did I not know this? I never thought adjacency matrix representation would help but this is so interesting.

The problem wasn't solved however but the power of adjacency matrix was so interesting to me I still don't understand why am I that surprised.

To my surprise it was followed by something called graph degeneracy it followed like this:

Graph Degeneracy: The degeneracy of a graph is the smallest value of d for which every subgraph has a vertex of degree at most d. This is also called as a d-degenerate graph.



A 2-degenerate graph: each vertex has at most two neighbors to its left, so the rightmost vertex of any subgraph has degree at most two. Its 2-core, the subgraph remaining after repeatedly deleting vertices of degree less than two, is shaded.

If a graph has degeneracy d, then there exists an ordering of the vertices of G in which each vertex has at most d neighbors that are earlier in the ordering.

I was thinking how to find the ordering here and I had an idea if I use greedy and repeatedly find and remove the vertex of smallest degree, adding it to the end of the list would give the d-degeneracy ordering, and the given algorithm was also same after all it's linear time so that's probably the best the given algorithm was this except this is a bit formal:

1. Initialize an output list, L, to be empty.
2. Compute a number, d_v, for each vertex v in G, which is the number of neighbors of v that are not already in L. Initially, d_v *is* just the degrees of v.
3. Initialize an array D such that $D[i]$ contains a list of the vertices v that are not already in L for which $d_v = i$.
4. Let N_v be a list of the neighbors of v that come before v in L. Initially, N_v is empty for every vertex v.
5. Initialize k to 0.
6. Repeat n times:
   - Let i be the smallest index such that $D[i]$ is nonempty.
   - Set k to $\max(k, i)$.
   - Select a vertex v from $D[i]$. Add v to the beginning of L and remove it from $D[i]$. Mark v as being in L (e.g., using a hash table, H_L).
   - For each neighbor w of v not already in L (you can check this using H_L):
     - Subtract one from d_w
     - Move w to the cell of D corresponding to the new value of d_w, i.e., $D[d_w]$
     - Add w to N_v

And now it went back to the main question which was finding the number of triangles so we they probably used this ordering algorithm I thought as I saw the algorithm was to initially

compute a d-degeneracy ordering using the earlier algorithm, and also hashing all the edges so we can check in O(1) time if there is an edge.

Now L has the d-degeneracy ordering (from previous algo) now, For each vertex, v
For each pair of vertices, u and w, adjacent to v and earlier in the ordering, i.e., u and v are in the list Nv from the degeneracy algorithm:

If (u,w) is an edge in the graph, then add one to the triangle count.

This runs in just O(N*E) which can be written as O(N*N) which is a big improvement from $O(N^4)$ and $O(N^3)$.

Now that I finally found a better way to find the number of triangles possible, I started to think now, I know that I can find clustering coefficients but how to form clusters, I mean social media, internet everywhere they form clusters so we can see more things we can relate unlike random.

I thought of customer segmentation I read about this in some article saying how google makes money through targeted ads, It's obvious here they form clusters and target clusters, I am sure because of some ads which I see feel so odd, it's not completely personalized but there are clusters and similar people are grouped so we feel that it's "personalized". I slept but the thoughts in my mind are basically animations of dense networks, clustering trying to make sense of diameter and many more I planned to complete the leaderboard question the next day.

I woke up and my head and mind were clear so, I started working on it again, I didn't make any progress I was struck at the same place. I decided to meet prof for this question, there are two reasons one is why did he say that it's a graph when he can complicate it into a real life scenario and the second is I was struck.

I went to the faculty block. Sir was near his car using his mobile, I went and said, "Hi" waving my hands as he was friendly with me in the past two interactions.

"Hello Jaishnav you look good today on which question are you stuck the queens one?", Sir asked.

"No I did that sir, I am unable to solve the last question.", I replied.

"Last, that's good you did the other 5, oh sorry 4 that's good, the graph question right?", Sir asked.

"Yes sir, I can't think of an efficient method.", I replied.

"What are you using to store your graph?", Sir asked.

"Store? Oh, as in taking input and pushing it to the adjacency list sir.", I replied.

“Is it the only way to store a graph?”, Sir asked.

“No sir, we can store the edges in an array too sir.”, I replied.

“So you know two ways to store a graph or do you know more?”, Sir asked.

I thought for a while and I realized that we can store it in adjacency matrix too. I replied, “No sir 3 we can store it in adjacency matrix also sir, I just remembered it now.”

Sir smiled and said, “Yeah adjacency list is not the only way to store a graph there are other ways too.”

Then I got the hint that I have to use adjacency matrix as adding edges into a list won’t do any change. I asked my first question, “Why did you tell it’s a graph sir when you could complicate it so it takes time to figure out graphs.”

Sir smiled and replied, “That was my initial thought but here I wanted to make a point that adjacency list is not the only way to store the graph. And I didn’t want the last question to be more tough, I am against that idea and hype basically.”

I smiled, thanked him and left. Went to the space and started working on it again

I think Adjacency matrix will be better as we can update the same matrix . So, I have to start with storing the graph in the form of an adjacency matrix . In an adjacency matrix named Adj the values will be as follows

Adj[i][i] will be equal to zero as the shortest distance from i to i is zero.

Adj[i][j] will be weight of the edge from i to j if a direct edge exists from i to j . If such edge does not exist its value will be infinity . But while coding we will use a large integer value for infinity . But how do I do it from here . I am stuck once again . In many such situations viewing the bigger problem in terms of smaller problem might help a lot . So , I started to think of the adjacency matrix as of what might it exactly represent . It only has shortest paths between nodes that are connected directly by edges . There might be a shorter path than the directly connected edge path through some other node . From this I started to extend my thought process . Let the initial adjacency matrix represent the shortest paths that do not include any node . Now , if we have the shortest path from i to j as some x , we may have another path passing through some node k that has the shortest path as y . So , now lets check if there is shortest path between every i and j that passes through the node k . If we can do this one k , then we can do this for every node and eventually find the answer . Till now the logic looks fine and I am confident that I could implement it . But how to implement it . Lets do the above process for one value of k . Then we can do it for all nodes

just by a for loop . Now we want to check if there a shorter path from i to j through k . This can be done by comparing the values of Adj[i][j] and Adj[i][k] + Adj[k][j] . We will update the value of Adj[i][j] with the minimum of above two compared values .The equation goes something like this

Adj[i][j] = min(Adj[i][j],Adj[i][k]+Adj[k][j]);

We will apply this equation to a pair of i,j only if both Adj[i][k],Adj[k][j] are not equal to infinity . There will be no problem if we apply the equation if one of Adj[i][k],Adj[k][j] is equal to infinity . There will be no change if we apply that equation if Adj[i][k] or Adj[k][j] =infinity because if Adj[i][k] or Adj[j][k] = infinity then Adj[i][j] will always be less than Adj[i][k] + Adj[k][j] as far as there are no negative cycles . As there is no change by applying the equation if Adj[i][k]=infinity or Adj[k][j]=infinity we do not apply it to decrease the time complexity and make the code more efficient . The code snippet for only one value of k looks something like this .

```
for(i=0;i<n;i++)
{
	for(j=0;j<n;j++)
	{
		if(Adj[i][k] != INF && Adj[k][j] !=INF)
		{
			Adj[i][j] = min(Adj[i][j],Adj[i][k]+Adj[k][j]);
		}
	}
}
```

To run this code snippet for every node , we just have to run a loop from k=0 to k<n . The code looks like this

```
for(k=0;k<n;k++)
{
	for(i=0;i<n;i++)
	{
		for(j=0;j<n;j++)
		{
			if(Adj[i][k] != INF && Adj[k][j] !=INF)
			{
				Adj[i][j] = min(Adj[i][j],Adj[i][k],Adj[k][j]);
			}
		}
	}
}
```

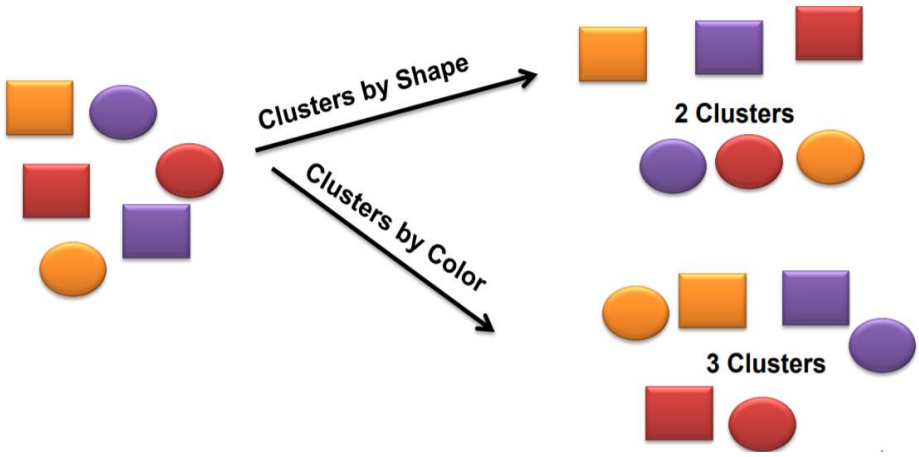
I was just shocked that I got it in one go . I realized that through practice we can become more fast accurate . Now I submitted the code and get it all correct in the first attempt .

It accepted, I am done with leaderboard I felt relieved, now I went back to the network algorithms (which is indeed more graph theory) to continue learning.

After learning how to find clustering coefficient, or more specifically finding the number of triangles in a given graph, I decided to learn about clustering graphs because it is obvious, customer segmentation is nothing but clustering nodes and giving them a name, this are used for ads, social media recommendations and many other real life applications.

It also provides a macro-level view of the data-set to study.

The process of dividing a set of input data into possibly overlapping, subsets, where elements in each subset are related, this was given as an example:



For example, the number of walks of length 2 is the number of vertices k such that there is an arc from i to k and an arc from k to j. And this is exactly the i, j entry in A2 , by the definition of matrix multiplication. So the definition of matrix multiplication itself does this, I didn't know this before, How did I not know this? I never thought adjacency matrix representation would help but this is so interesting.

The problem wasn't solved however but the power of adjacency matrix was so interesting to me I still don't understand why am I that surprised.

It was obvious here, there can be many types of similarities so we can divide them in many ways, I wondered how to cluster a graph, what are the algorithms present but I realized I need to know more and one of the interesting thing was why should we cluster a single graph, we can cluster multiple graphs too they are called by different names:

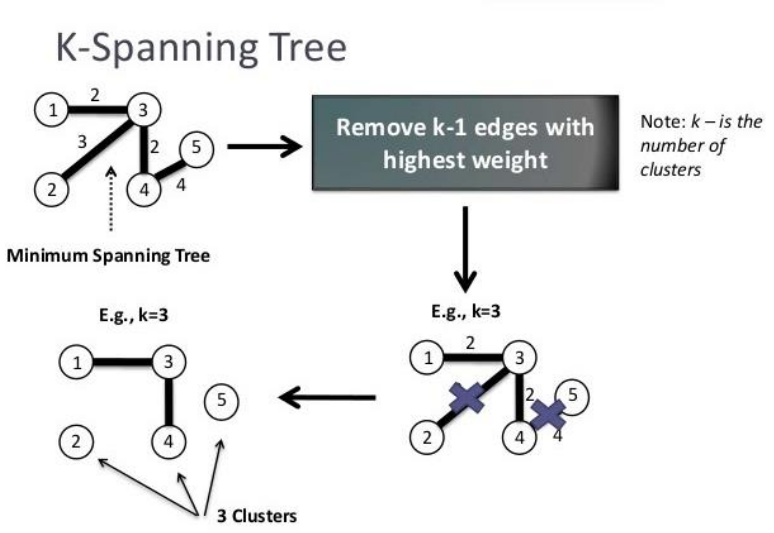
Between Graph Clustering: Clustering a set of graphs.
Within Graph Clustering: Clustering the nodes/edges of a graph.

Both looked so important and interesting in the field of data, we can know a lot.

After completing the differences, the article now started to go into the different algorithms to do between-graph clustering, the first one was

K-Spanning Tree, I know about Minimum spanning tree (MST) what does this mean I thought as I moved onto learn this, the idea here was to form a minimum spanning tree first, we can do it with Kruskal algorithm or Prim will also work I thought, then if we remove k-1 edges from it we get k parts of a graph these are not clusters.

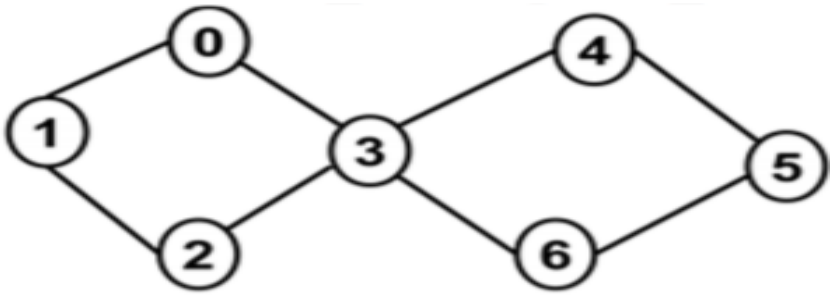
So if we remove edges with higher edge weights, then all nearby one's will become separate which is indeed a cluster, It is so interesting and also so simple, I thought. Removing k-1 edge high weight edges would result in k clusters. This image summarizes it:



However we here we should have an idea of how many clusters we need, if we ask for more there would be more clusters but with very less nodes so which may not be efficient for us, but in other cases when we are sure we need some k clusters this would work and almost all MST algorithms take O(Elogv) time this also takes the same time.

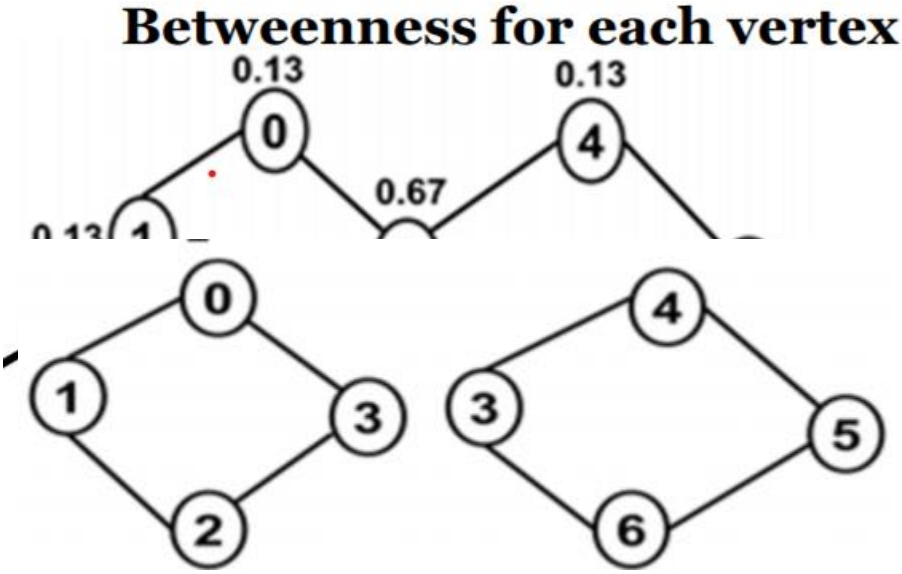
There are so many things which I didn't know about graph theory but with the basics, I had it is indeed interesting.

The second algorithm was betweenness centrality based and it said find the node or edge with highest betweenness centrality and break it into two graphs separately, followed by these pictures:



This looked really good. The only problem was what does betweenness centrality even mean?, I thought for a while and googled it and found an article with title, "**The 5 graph algorithms that you should know"**

The codes are written in python using networks, I expected it because it's so simple to do



any operation so people who are working can do their job properly without worrying about implementation.

I looked at the clock it was late and thought I would continue it next day and I dozed off. And planned to read more tomorrow then focus on final exams.

The algorithms were Connected Components, Shortest paths, Minimum spanning tree, Page rank, centrality. There were also interesting real-life examples and applications for all these so I started reading those again.

Connected Components: it finds all the connected components of the graph, so dividing the connected components would help us seeing things differently, the given example was if a graph was given with roads as edges and cities as nodes, it is possible that there is no way on road to reach from one city to another if we can see the connected components then technically we have separated cities with water in between, there can be many applications here.

The code was in python so basically a single command however I realized that we can solve it using DFS quickly anyway.

Shortest Paths: This was obvious, shortest path calculation has a lot of advantages, used in finding routes for google maps, also linkedin shows 1st degree connections, 2nd degree connections this are also done by this.

Again code was in python so it's a command basically however we can use Dijkstra for finding this and there was this Dijkstra talking about his own words:

> *What is the shortest way to travel from Rotterdam to Groningen, in general: from given city to given city. It is the algorithm for the shortest path, which I designed in about twenty minutes. One morning I was shopping in Amsterdam with my young fiancée, and tired, we sat down on the café terrace to drink a cup of coffee and I was just thinking about whether I could do this, and I then designed the algorithm for the shortest path. As I said, it was a twenty-minute invention. In fact, it was published in '59, three years later. The publication is still readable, it is, in fact, quite nice. One of the reasons that it is so nice was that I designed it without pencil and paper. I learned later that one of the advantages of designing without pencil and paper is that you are almost forced to avoid all avoidable complexities. Eventually that algorithm became, to my great amazement, one of the cornerstones of my fame.*
>
> *— Edsger Dijkstra, in an interview with Philip L. Frana, Communications of the ACM, 2001 [3]*

A 20-minute invention is used by google maps and many places every day, I thought.

Minimum Spanning Tree: there is nothing new here it's used for clustering, network design and many, the idea started from how to connect all nodes by travelling less distance that's the motivation here.

Page Rank: This is used by all search engines so the results are sorted in the order of page rank, initially they are assigned by quality of page, number of quality incoming and outgoing links to the page. All recommendation engines use this for ordering, this won't be simple as google's algorithm itself is very complicated and it's made better every time.

And I reached to the end the important one for which I came here initially centrality, specifically betweenness centrality to my surprise there are many types of centralities Degree Centrality, Betweenness centrality, Closeness centrality, EigenCentrality and many more.

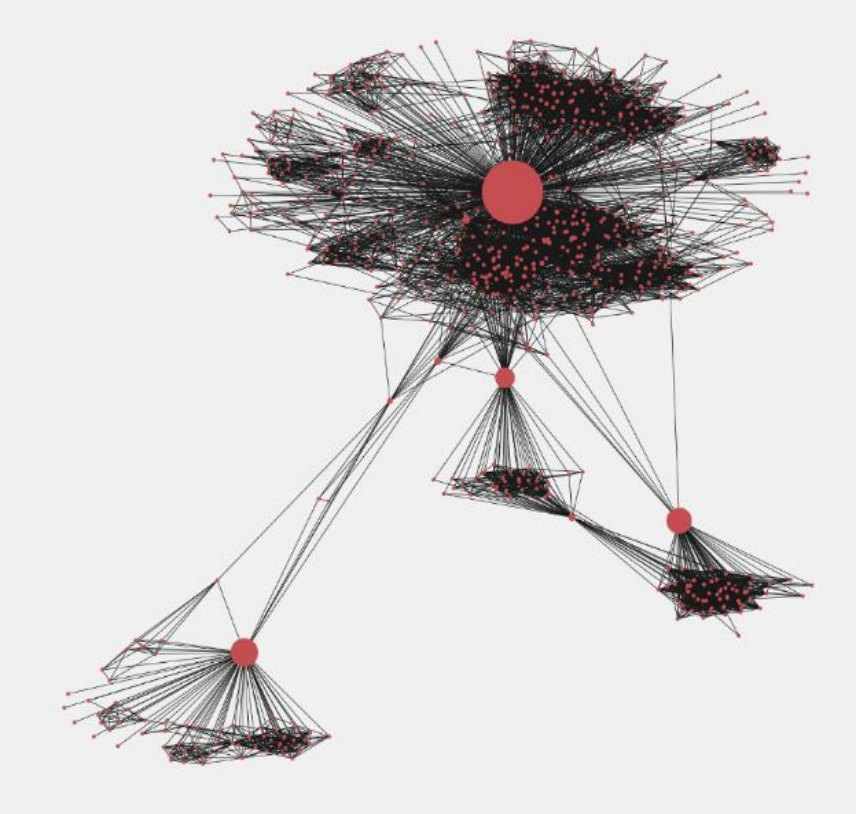
(the one in italics means the definitions that I read)

Degree Centrality: *Degree centrality assigns an importance score based simply on the number of links held by each node. This can be used to tell which node is more connected, holds more information.* So, losing nodes with high degree centrality may lead to loss of much information generally, I thought.

Closeness Centrality: *Closeness centrality scores each node based on their 'closeness' to all other nodes in the network.Closeness centrality scores each node based on their 'closeness' to all other nodes in the network*. So, nodes with high closeness centrality can influence the network as they are more connected.

Eigen Centrality: *Like degree centrality, EigenCentrality measures a node's influence based on the number of links it has to other nodes in the network. EigenCentrality then goes a step further by also taking into account how well connected a node is, and how many links their connections have, and so on through the network.By calculating the extended connections of a node, EigenCentrality can identify nodes with influence over the whole network, not just those directly connected to it.* So, this gives us more information about nodes as it goes and takes extra steps while assigning some score.

Betweenness Centrality: *Betweenness centrality measures the number of times a node lies on the shortest path between other nodes.This measure shows which nodes are 'bridges' between nodes in a network. It does this by identifying all the shortest paths and then counting how many times each node falls on one.* This can have many uses, I thought almost the entire social media uses this. They used a picture to represent it



The centrality part of the article was so good, and Now I can't wait to learn how to calculate the scores. What are the formulas, these help in so many ways, I initially thought this was something related to AI or ML but this is graph theory in a sense.

Mutual friends on Facebook, degree connections on LinkedIn, Railway junctions and many more are basically how betweenness centrality work. There are many more applications to these types of point and they are the basis of social media, Sir was right. Network has maybe one of the widely used applications in today's world, and I haven't even scratched the surface of the implementation of everything. I've always felt in awe at the way the computer science world shot up in such a short span of time, and networking is only a part of it. Though significant, there are many other areas like quantum computing, AI and Machine learning, even database management and analysis that have many wide range applications in todays world and the foreseeable future.

Inception

Another assumption shattered; it looks like every assumption I make in this book turns out to be false almost immediately. I woke up today to a mail of a familiar format, it was a mail consisting of a link, to yet another phase of questions for the leader board. I couldn't believe it, all this time it was believed that 100 was the maximum marks possible, but this would contradict that completely, and nobody knew about this on campus! Well, the marks were no longer a concern, but I was too curious to neglect the mail completely.

The link opened up to a single question. The number of questions was interestingly low, but the question itself was much more peculiar. It was a yes/no question! After all those paragraph size questions, with solutions with codes spanning hundreds of lines, the last question was a simple yes or no question!

----------Q1----------

Consider a hypothetical experiment. There exists a man, whose every reaction since birth is monitored and the event-reaction pairs are fed into an intelligent machine. After many years, both the man and machine are given a situation, will the man's reaction and the machine's prediction be the same?

On a superficial look this looked like a simple question, if the machine knows reactions to everything the man has encountered so far, it shouldn't be very hard for it to guess how the man would react to a new situation. Results from previous such events can be taken into account and used to predict the reaction of the man. Though, the other side of the argument also needs to be considered. Every action of a man is not predictable, this is largely due to limitations of today's AI but also due to randomness in the behaviour of humans or any living creature. Though, wouldn't those random events also start forming a noticeable pattern if information is being recorded from those many years. Since everything is hypothetical, there is no concrete way of validating all these claims. I don't think valid claims are expected, given that the question is a thought experiment. Does a right answer even exist for such type of questions? There was only one way to find out. I chose the yes option.

The screen went blank for second and then another question appeared. It was again a yes or no type question. By the looks of it this seemed like an endless chain, I heard of something like this before, but I wasn't able to remember exactly what that was.

----------Q2----------

This question is based on your answer to the previous question. Given that the machine was able to predict the man's reaction based purely on the previous actions of the man, which is the only entity it interacted with. By analogy, can we say that the man was also similarly influenced by the only entity that he interacted with, which is nature?

This question is basically asking if the situation between the machine and the man is analogous to the situation between the man and nature. That is an interesting statement, and very intuitive, but there are many large assumptions in that statement. The way a machine analyzes situations and the way a human being analyzes a situation cannot be the same, however intelligent the machine is. Though, if we go on to assume that the machine has exactly human level intelligence, then the nature-human and human-machine interactions seem similar. Except for the fact that human-nature interactions are more random, the human chooses what to interact with and what not to interact with, this creates a smaller set of cases for the machine to analyze, and the machine does not have the power of choice. While, choice seemed like a big factor to discard this analogy, the question doesn't seem to focus on that aspect of the hypothetical situation. Continuing my trend from the previous question, I went with my gut and chose yes again.

The screen blanked out again and another question appeared. It was again a yes or no type of question. I finally remembered why this structure of questions was so familiar. Initially I thought these questions were forming a chain, hence I was not able to remember. These questions were actually forming a tree, a decision tree. I was pretty sure there were different questions if I had answered no in the previous questions.

----------Q3----------

This question is based on your answer to the previous question. Given that interactions between two entities govern how one of the entities behaves. On extrapolating to the level of universe, can we say that if we have information about every interaction can we predict all further events. Moving forward, with information can we go on to say that information is enough to create universes, i.e. is information fundamental?

Unlike the previous two questions, I don't think I can make any advancements in this question. Firstly, the question is very vague with terms like information and creation of universes. While the analogy from machine-human to human-nature seemed intuitive, this analogy is far too big.

The consequences of such an analogy, if true would be very huge. If information is all that is needed to manipulate every entity in the universe, most of today's physics would be redundant. Though, to completely disregard the given statement is also something I don't have enough information about. I need to consult with prof about this, such vague and expansive topics are normally quite interesting when he drifts on n class. I'll answer this after the discussion.

Judging by the vagueness of the previous few questions and their contrast to all the other questions part of the leader board phases, I felt this might be ungraded and was only included in the leaderboard to ensure that whoever finished everything would for sure attempt it. Whatever the intention of the institute, those questions were far too deep and the answers they were implying far too unconventional for me to ignore.

At this point I was completely comfortable talking to sir and he addresses me with my name, I went to the faculty block, sir was sitting on one of the benches in the garden and pondering about something, I approached him he realized and smiled and told to come.

"Yes Jaishnav, how may I help you?", he said, pointing me onto a path to walk on.

Now walking with him, I asked, "Sir I have a doubt."

Sir smiled and said, "Before that, why didn't you answer the third question when you attempted the first two?"

I was shocked I didn't expect sir would track my responses, I replied, "Sir, well, I am in a dilemma, answering it would change a lot of things, and it could potentially change the way we see the world. Also, I felt that it was a decision tree with no end, so I would have to consult you, so I thought the sooner the better."

Sir looked a bit perplexed and asked, "The endless decision tree part is correct but you just made a big statement before that, could you elaborate."

"Umm, let's assume a software is a universe, it's made by 0's and 1's which are on top of registers, which are stored in hard disk and main memory and there is another layer of code which runs the software now if I somehow managed to hack he software and alter the 0's and 1's permanently this won't mean anything here, but if we extrapolate this and assume

universe is also made of information which is bits which might be different compared to 1's and 0's now it should be possible that I can disprove many famous laws like conservation of momentum, or I can disprove the 2nd Law of thermodynamics which says entropy of a process can be constant or it increases it will never decrease. Now If I can make that happen a lot of phenomena will be unexplainable everything changes, I attempted the first two questions because I was confident that information plays an important role but the idea that it can be fundamental is something I can't be sure of, as the word information itself is quite vague."

Sir listened attentively to everything I said and replied, "Well there is classical physics because Newtonian mechanics couldn't do everything, Now there is quantum physics too which contradicts with both classical and Newtonian mechanics, what you are saying might end up being the biggest jump mankind ever made. Just because it possibly disproves many of the current beliefs doesn't mean that it shouldn't be true right."

Sir continued, "Yes sir, but if we can alter the information which technically means we know information, if we know information, we can analyze it and answer almost every question which technically means job of physics is done."

Sir gave me some time to digest and continued, "Your whole idea is based on the fact that you could alter the fundamental thing, Assume if electron was the fundamental thing and it had positive charge, a lot of things would change so because of this idea. But it's based on the fact that you could alter it which is not simple. The question is can you and the important question is are you ready for the consequences.

This could make a good plot for a Sci-fi movie.

I was convinced how my idea has faults however, I asked, "Well I agree with it, Assuming universe started from Big Bang, and we have all the information from that time about everything until now. Can we use that to create another universe with the information probably no, we might need a particle, so I believe information can't be it."

Sir replied, "Well what if you had information to make it."

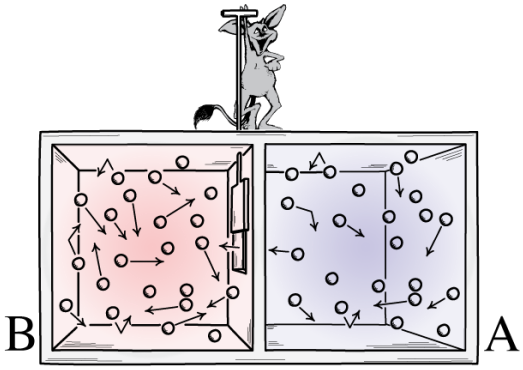
"Well still we need some particle to do it."

"What if the particle is nothing but information?."

At this point I felt the word information is being used more often and I need to know what it exactly means and then I had the thought how vague it is.

I asked, "Well the meaning of information itself is not completely clear, and what advantage can we possibly have by knowing extra information."

Sir smiled and started, “Well there is an interesting thought experiment named Maxwell’s demon. In that, there is an insulated container, and there is a wall in the container with a massless gate which is controlled by a wise owl. This owl has the complete positions of atoms and molecules at every instant along with their velocities, the owl moves the gate whenever some molecule is heading towards right and is near the gate so it quickly opens and closes such that a molecule or atom moved to the other side, after infinite time there is a situation such that there is difference in number of molecules in each side which means there is a temperature difference between those to sides now we can use this temperature difference to create energy”. Sir opened his mobile to show the picture of the system.



I never expected this, “Wait, What about conservation of energy?”.

“To explain it you should understand about information here we have to feed the owl with information every time apparently energy used in that is the energy generated.”

Information isn’t that vague I thought, So I looked at sir, and asked “So information is fundamental?”. Sir laughed and said, “Many people are working on it and they didn’t come to a conclusion.

Sir continued, “The idea was first proposed by Physicist John Wheeler. And he called it “It from bit”, it being the universe and bit being information.”

“But I can confidently say, information is not something random which is just in the universe it does have some higher meaning and purpose. The Universe makes no sense, and it’s boring interesting theories like “it from bit” help me to appreciate everything”. He stopped walking and turned towards me.

believe that consciousness is not an accidental by-product of the physical realm but is in some sense the primary purpose of reality. Without us to ponder it, the universe makes no sense; worse, it's boring. What do you think Jaishnav?”, he asked.

“I think so too sir, the theory is really interesting but yet quite vague. I think because of our present understanding of the world, this “it from bit” theory seems as vague as the vagueness it is trying to solve.”

“That is true, we are very far from verifying such theories. Don’t worry I won’t ask that much of you in the lab, but you should aim to reach there, if not in college, by the time you retire from this field.”

“Lab sir?”, I asked, confused by the sudden change in direction of the discussion.

“Your research lab Jaishnav, I look forward to working with you”, said sir, his arm outstretched.

“Thank you, sir”, I said, shaking his hand, a smile on both of our faces. This was how it all began.

Team V:

Venkat Adithya - 2019111025

Ruthwik Reddy - 2019101121

Kranthi Kumar - 2019101027

Puneeth Sai - 2019101064

Harsha Vardhan - 2019101122

Abhijith Ch - 2019111036

Varun Chowdary - 2019101114

Soma Datta - 2019111008

Lokesh Paidi - 2019101062

Srijith Padakanti - 2019114002

Cover art by: Rhuthik. P