Sujay Jakka

Joanie Morris

COMP-3240

21 November 2022

Discrete Structures Final Project Paper

I chose to do my project on counting more specifically combinations. Furthermore, the problem I am solving pertains to my startup social media platform called Bump!. Ironically, in the era of connectivity, the timeless problem of losing touch with friends, like friends from childhood, high school, and college, remains ever present. These may be friends you had valuable memories with and enjoy being around; however, it is awkward for some people to reach out to those friends they have not spoken to in a while. There needs to be an opportunity to casually reconnect with others while feeling comfortable and not coming off as needy or random. Furthermore, there needs to be an easier way for individuals to meet people that have common interests and common friends nearby because there's no way to know when or even if you will run into these people in person. These are the problems that Bump! is here to solve. Bump! is a platform where unexpected encounters become meaningful connections. Our aim is to target the age group of 14-24 which is roughly 45 million people in America. First, you create an account with your first name, last name, profile pic, and phone number. Then you can choose up to 8 interests. This app functions like a normal social media platform where you can send messages, videos, and photos but fundamentally it is different. Every week, you are paired with two people. These people can be existing friends – friends you have already added on the app – or they could

be mutual friends – a friend of a friend on the app who you haven't added. We take into account your location, the number of mutual friends in common, and the number of interests in common when pairing you with a mutual friend. If you are paired with them, they are paired with you. These people you are paired up with every week are called Bumps(like I bumped into you this week). If you interact with anyone on the app whether that is chatting or sending pictures you work towards tokens. If you interact with your Bumps you earn tokens faster. With these tokens, throughout the month you can use a token to specifically add people by interest. You just search up one or multiple interests, and a list of people will appear. Towards the top of the list is people who are nearby that have the most mutual friends as you. You can then choose to use a token and add this person. Furthermore, tokens reset every month.

Now you may be wondering how exactly I incorporated Bump! into this project. My project is centered around counting the different ways for an individual to be assigned their Bumps for the week. We are assuming the Bumps are based solely on the user's interests and not existing friends or mutual friends like the actual platform. The program asks the user to input the number of users that each like a certain interest. These interests are Sports, Music, Reading, Movies & Tv, and Cooking. It also asks the user to input either “y” for yes or “n” for no for every interest. For every interest the user enters “y” for the total amount of people to choose from will increase by the number of people that have only that as their interest. For example, if I input that the amount of users who only like sports is 100 and I also enter “y” indicating I like this interest then the total amount of people to choose from will go from 0 to 100. If I then input 200 users only like music and then enter “y” the number of total users to choose from will

increase by 200 becoming 300. Then if I say 50 users like reading and I enter “n” then the total number of users to choose from will not increase at all, it will stay at 300. The total number of users to choose from is my “n” for the combination formula. Lastly, I ask the user how many Bumps they want for the week and they can choose any number between 2 to 6 inclusive. On the app, users will only have two Bumps every week but for the sake of making the project a little more complex, I am allowing the user to input a number between 2-6 inclusive. This number is our “k” in the combination formula. Now the program has all the information necessary to find the total amount of combinations for the user's Bumps for the week. It will implement the combination formula which is (n!) / ((n-k)! * (k!)) and output the result.

My topic, combinations, is the focal point of the program. All the user inputs I ask are so I can know my “n” and “k” values for the combination formula. Furthermore, my topic plays a huge role in solving the real-world problem I chose because without the combination formula or counting it would be next to impossible for me to implement a way to count the number of different combinations for the user's Bumps for the week. I do not know where to even start to implement a solution without using the combination formula or any other counting formulas. Furthermore, the topic was not too difficult to implement, but I did run into some limitations while trying to demonstrate my topic in my program. Normally if I wanted to code for example C(5, 3) I would find 5!, 3!, and 2!. Then I would do 5! / (3! * 2!) to find the answer. However if I used this train of thinking for something like C(100, 3) I would run into an overflow error. There is not enough bits for even a 64 bit integer for 100! or 97!. Which is why I instead implemented an algorithm that found n! / (n-k)! which from the above example would be (100 * 99 * 98).

However, because my “k” value will only be between 2-6 inclusive I went ahead and found k! normally. Overall, this project has helped me gain a deeper understanding of how to implement factorial in programming because I never had to implement it before. It has also helped me simulate to a certain degree the number of different combinations for a user’s Bumps for the week which was fun.