Google Summer of Code' 2024 Proposal

Project: Developing 8 Math games for Sugar

Basic Info

Name	Spandan Barve
Github	marsian83
Email	spandan567@gmail.com
Portfolio	https://marsian.dev
College	IIIT Gwalior, Madhya Pradesh, India
Languages	English, Hindi, Marathi
Timezone	Indian Standard Time (IST / UTC +05:30)

Why Sugar Labs

I found Sugar Labs and thought it was something that could make an impact. They're all about making learning easier with technology. So, I started exploring its projects. By late January 2023, I got interested as a developer, especially because it aligned perfectly with my love for Python and game development, having experience with pygame.

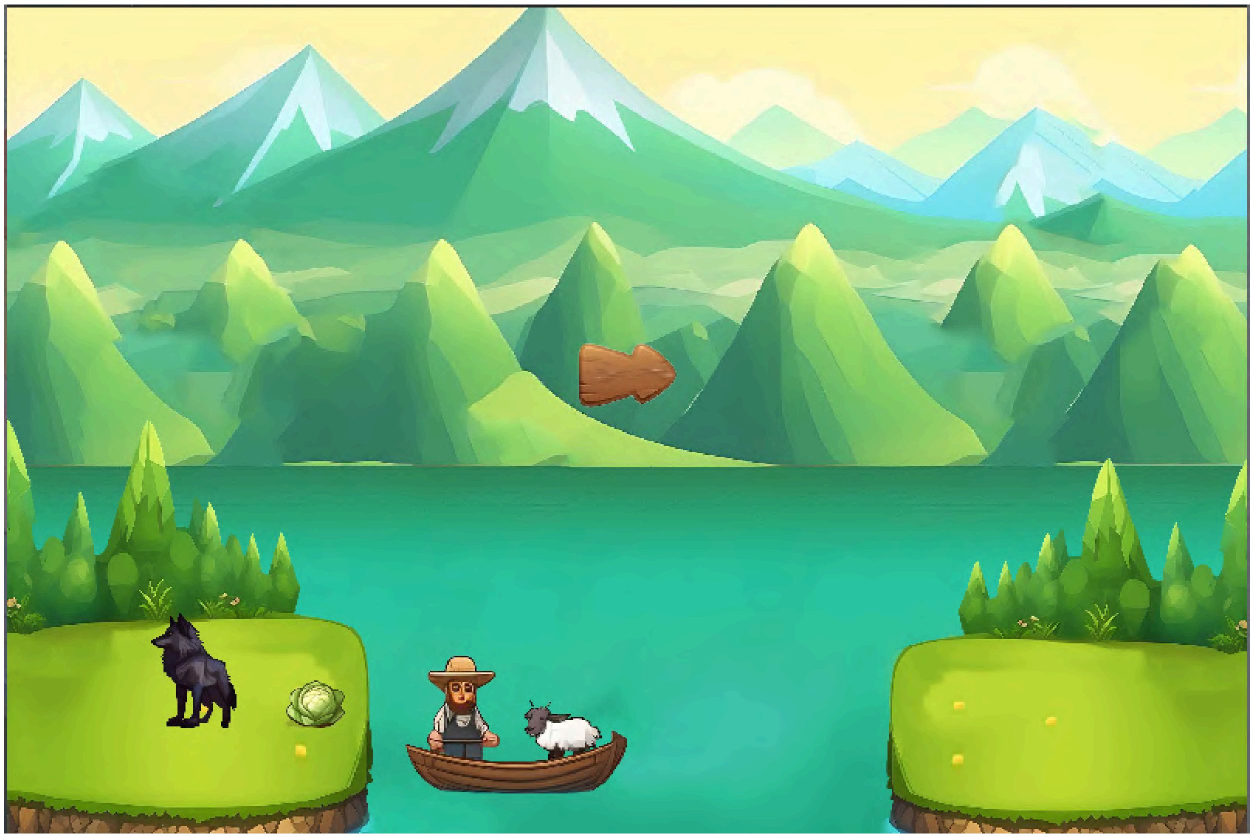
What really hooked me was Sugar's mission to provide open-source educational tools and fun activities for kids. I appreciated their efforts, especially the One Laptop per Child initiative.

Although I didn't get selected for GSoC last time, I'm excited to apply again for 2024. I'm looking forward to potentially contributing to Sugar Labs and making a positive impact on education through technology.

Contributions to Sugar

River Crossing Activity

Github Repo: http://github.com/marsian83/river-crossing-activity



I have made a River Crossing Activity for the sugar desktop environment inspired by Goat, Cabbage and Wolf. The game involves a farmer trying to cross a river with his belongings.

- ➢ I have implemented this activity in a modular manner such that more levels / characters and themes may be added with ease.
- ➢ The assets used in the game are self designed.

- ➢ Implemented an update system logic which allows the separation of \`views\` and \`components\`
    - ○ Views describe the currently opened page / screen
    - ○ Components describe the various parts within a view

- ➢ The implementation for this is done by implementing generic classes like :
    - \`Drawable\`makes it easier to create something which will be drawn on display surface
    - \`Clickable\` encapsulates all the logic required to make an on_click event for an object
    - \`ContainerBox\` which resembles an automatic flowing container which justifies and positions items automatically and symmetrically.

I aim to use these classes and create more such generic classes for future activities as well which will be developed by me. This framework can be used for my future activities including but not limited to the activities specified in this proposal.

Highlighting my past Pull Requests to various sugar activities

S.No.	Activity Name	Pull Request	Status
1.	jamath-activity	#35 Refactor: new class Juego_button & Bugfix: sound repeating when hovering on buttons	Merged
2.	fractionbounce	#17 made the ball draggable with mouse : added grabbed property to Ball and _mouse_motion_cb to Bounce	Merged
3.	dotsAndBoxes	#19 Bug Fix : Updating colors makes the game state disappear	Merged
4.	numberrush-activity	#9 feature: show correct answer after game over	Merged
5.	Bridge	#33 Made the toolbar show the currently selected tool	Merged
6.	ball-and-brick-activity	#18 replaced TestGame entries with gameLoop() at new game and continue event	Merged
7.	block-party-activity	#32 bugfix: score overflow at 4+ digits	Merged
8.	Block-party-activity	#30 Upgraded deprecated Gdk.color.parse to newer Gdk.color_parse	Merged
9.	browse-activity	#121 updated 'btn' class to have cursor: pointer so that it feels interactable	Merged
10.	numberrush-activity	#10 Added latest screenshot and added missing screenshots	Merged

Project Goal

I plan to at least develop 8 Math games for Sugar within the GSoC'24 timeline. The different visuals representations and assets in the below proposal are self designed made in Adobe photoshop. The development of these activities will include basic functionalities of the game, which are not mentioned particularly like Sound.

How will it Impact Sugar Labs

Upon completion of this project, Sugar will now have 8 more fully functional and developed math games activities. It will help children learn different difficult math concepts like, Pascal triangle, graph theory, latin squares and many other complex mathematical notions in a gamified and visually child friendly manner.

Tools and Technologies

Python, Pygame, Sugar

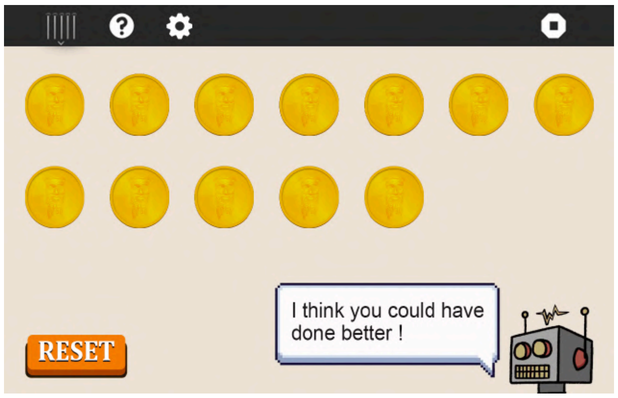
Project Type

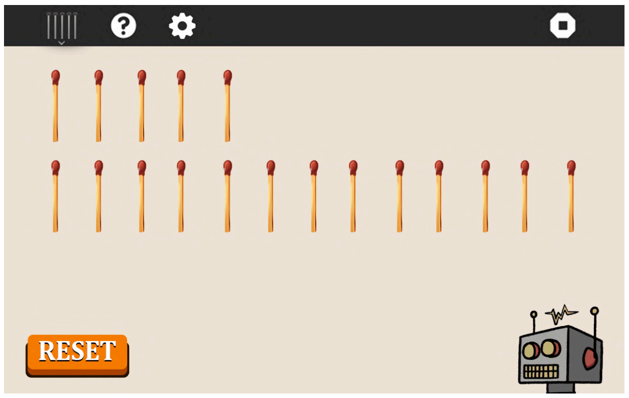
350 hours

Following are the description of the activities I aim to develop:

1. NIM Game

The below images show a concept art designed by me of how I envision the Nim activity to look like.



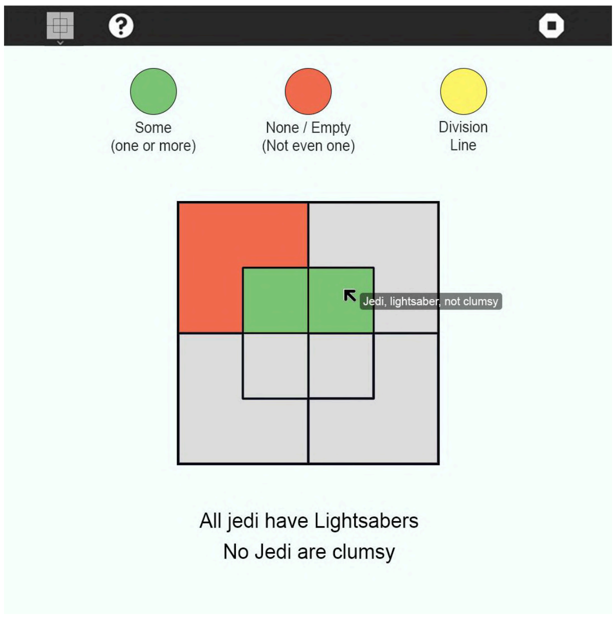


The activity will consist of an item which will be arranged in rows and each participant will take turns removing a number of these items. The participants will be the user and the computer (Robot). The last person to remove an item will win the game.

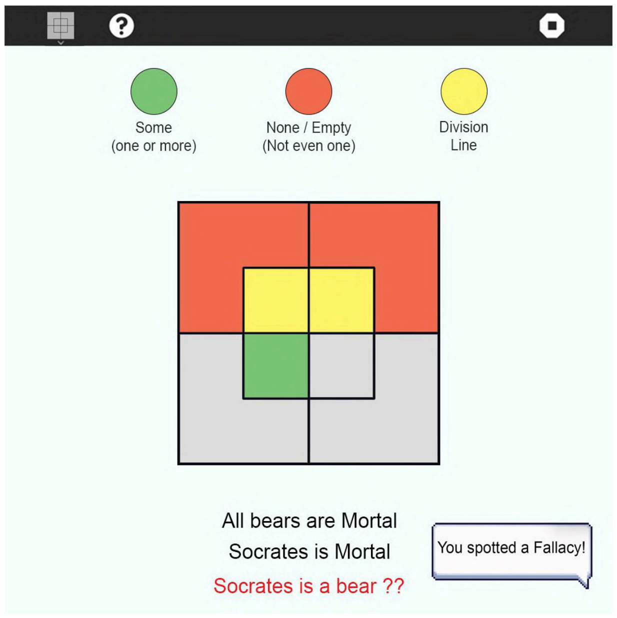
- Settings will be included in the game which will let you customize the configurations and visuals of the game
- Visual settings will include the enemy character(for eg: the robot in the images) and the item involved in the game (for eg: the coins or matchsticks as shown)
- Configurations include no. of rows and no. of items per row. You can also set a custom limit on how many items at most a player can remove per turn.
- The AI for this game will be implemented using the "Winning Arithmetic Progression" aka "Zero Nim sum" concept used by many Nim-like games.
- The enemy will, every now and then, show a message to the player as in the cut-the-knot demo inspired by Alexander Bogomolny.
- Since the aforementioned game demo makes it impossible for the player to win, It will also be true if a perfect strategy bot is implemented, thus adding difficulties will be important. Normal / Difficult / Impossible modes will be added to the game. The difficulty modes will be implemented by having a chance for the computer to intentionally make a mistake.

2. Lewis Carroll's Game of Logic

I plan on implementing Lewis Carroll’s Game of Logic inspired by a demo found here by Gustavo Cruz on YouTube.



All jedi have Lightsabers
No Jedi are clumsy



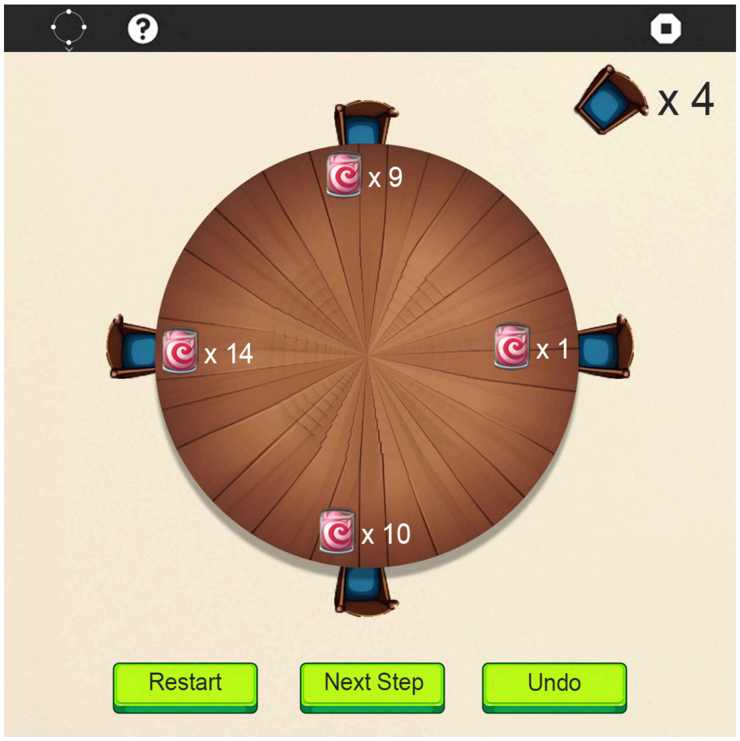
All bears are Mortal
Socrates is Mortal
Socrates is a bear ??

You spotted a Fallacy!

- The game will contain a given set of logical / propositional syllogisms like for example: “Some cakes are wholesome”.
- After selecting one of these, the game board will reflect the involved attributes (like cakes, wholesome).
- After selecting the Syllogism, the player will mark the given trilateral board to try and best represent the given propositions.
- Sometimes, this may lead to Fallacies as shown in the example screenshot. So children playing the game will also learn about how fallacies may be derived from simple logic.
- Hovering on the grid will also reflect the logical association it holds as shown in the image.
- Clicking on a grid box will cycle its state between the 4 possible states:
  - Some : some x are m
  - None : no x are m
  - Division Line

    - Unmarked
- We will also reflect if the grid as marked by the player is correct or not, alongside reporting any fallacies which may arise.
- The syllogisms and their respective answers & fallacies will need to be hardcoded and will be stored in a JSON file to make it easier for future contributors to add more propositions as they want.

3. The Candy Game



The game of candy will have the following game in perspective of the player:

1. Selecting the Number of Children: Players can choose the number of children to distribute the candies among.
2. Initial Candy Distribution: Following this, a random distribution among the children will be done ensuring each child receives an even number of candies.

3. Visual Setup: Once candies are distributed, The player is presented with a visual representation of the children arranged in a circle, each with their allocated candies.
4. Initiating Candy Exchange: The Player can initiate the candy exchange by clicking the "Blow Whistle" / “Next Step” button. The candies will be distributed as per the game’s logic (*each student simultaneously gives half of his or her own candy to the neighbor on the right. Any student who ends up with an odd number of pieces of candy gets one more piece from the teacher.*)
5. Error Handling: If any child ends up with an odd number of candies after the exchange, the whistle button will be disabled. The Player will then be asked to give one candy to the affected child to ensure an even distribution.
6. Repetition: The process continues until all children possess the same number of candies.

One suitable data structure for this game can be a circular linked list. Each node in the linked list represents a child, with attributes such as the number of candies they possess and a reference to the next child in the circle.
The game will be made such that it dynamically adjusts its interface and visualization based on the number of children selected by the player. This ensures that the game remains playable and visually appealing regardless of the chosen number of children.

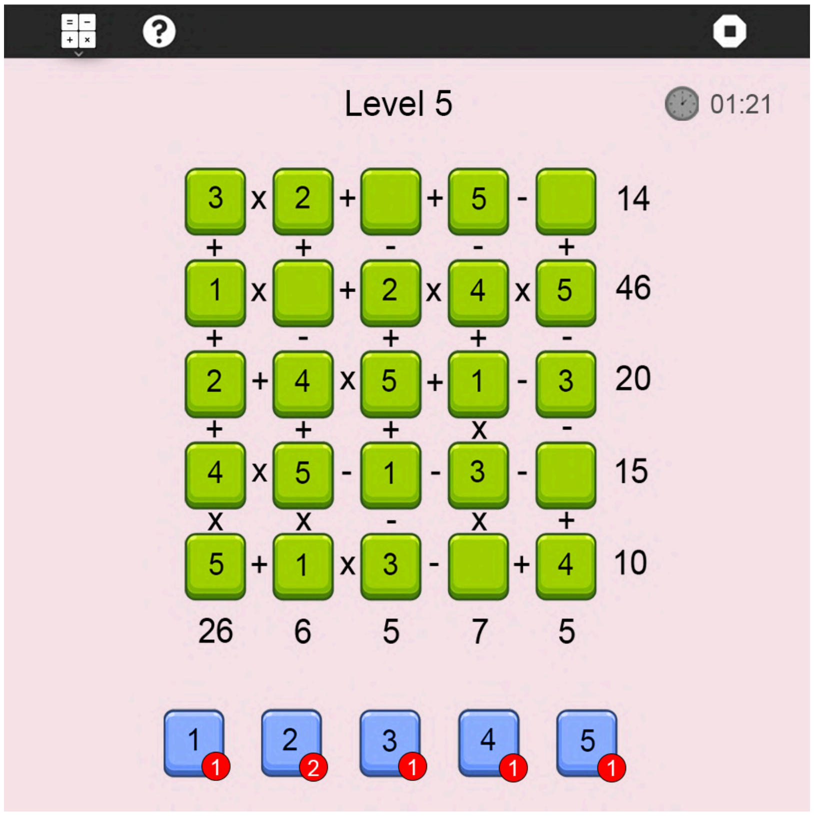
4. Number Guessing Game

I'm planning to create the game with two options:
Option 1: Player Guesses
In this mode, the player selects a difficulty level. Each level determines how many guesses the player can make. For instance, "Hard" allows only 5 guesses, while "Easy" gives 20. The player then guesses a number between 1 and 100. After each guess, the game provides feedback whether the guessed number is higher or lower than the actual number. The player continues guessing until they find the correct number.

<u>Option 2: Computer Guesses</u>

Here, the player simply responds "lower" or "higher" to sets of numbers presented by the computer. "Yes" indicates that the player's number is within the set, while "No" means it's not. Through this process, the computer narrows down the possibilities until it accurately guesses the player's number.

5. Latin Squares



I plan to develop the activity with the following considerations:

Game Mechanics:

1. Players are presented with a matrix containing missing values and operators between each cell.

2. The result of each row and column is displayed at the right and bottom of the matrix respectively.
3. Players must fill in the missing values to satisfy the equations.
4. Each number in a row and column must appear only once.

Interactive Gameplay:

1. Players can hold and drop cells within the matrix to rearrange them and test different combinations.
2. The game progresses through levels, starting with a 2x2 matrix.
3. Each level introduces either additional rows and columns or more missing values, increasing the complexity.

Scoring System:

1. A score is calculated based on how quickly the player solves the matrix.
2. The faster the solution, the higher the score.
3. A high score is saved to track player progress and achievement.

We will let the player choose the square size and then we will generate the board algorithmically. We may use an algorithm as such :
Let n be the size of the Latin square

1. Let matrix be an empty array
2. Generate a row (array containing letters from 1 to n) and then shuffle it randomly using a function the likes of python's `random.shuffle`
3. Rotate row by 0 to n and append the rotated row in matrix
4. Shuffle the matrix itself again (`random.shuffle`)
5. Assign random operators between the cells and generate answers
6. Remove a set number of cells to obtain a final square with missing pieces which will be filled by the user.

6. Pascal Triangle Visualisation

This activity provides a visualization of the Pascal triangle with various options to modify and interact with it. Players can modify these three values to experiment with the triangle's configuration.

1. The "modulo" function determines the values of each cell in the triangle. In simple terms, if modulo 3 is selected, the triangle will contain digits 0, 1, and 2.
2. The "rows" option corresponds to the number of rows in the triangle.
3. The "player" option determines the filling of the first and last cells of each row. For example, if "player" is selected as n, the first and last cells of n rows will be filled according to the player's choice.

Each cell of the triangle is calculated using the formula
$p = q1 + q2 \mod N$
N is the chosen modulo, and q1 and q2 are the cells directly above cell p.

Additionally, players can switch between displaying numbers and colors in the triangle for added visual variety and exploration.
This activity will offer an engaging way to explore the properties and patterns of Pascal's triangle while allowing players to customize their experience and experiment with different settings.
Reference: https://demonstrations.wolfram.com/PascalLikeTrianglesModK

7. Three Utilities Puzzle

The activity will explain to the child the concept of non planar graphs. The graph used in the 3 utilities puzzle is Kurtowski's graph $K_{3,3}$, which is a bipartite completely connected graph with sets of 3, 3 nodes.
The game requires that 3 nodes of houses get connected to 3 nodes of suppliers (eg: Water, Electricity, Petrol) without having any connection pipe overlap
The aim is to show the child that the graph is non planar and can never be drawn in a manner such that no edges intersect (aka embedding of graph)

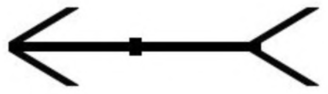
The player will be able to move the edges as they want in order to try to make the non intersecting supply line (Graph Embedding)
The player will also be able to move the nodes (Houses and Suppliers) to attempt drawing the embedding.
After trying enough, the player can look at the solution and understand that it is impossible and also learn why.

> Realizing that implementing this game will require developing the entire graph simulation logic. I reckon we could take this a step ahead by also creating an activity for understanding Graph Theory as a whole which will have logic for all sorts of graph representation concepts. This idea may be implemented as per allowance of time and discretion of the mentors.

8. Illusion Box

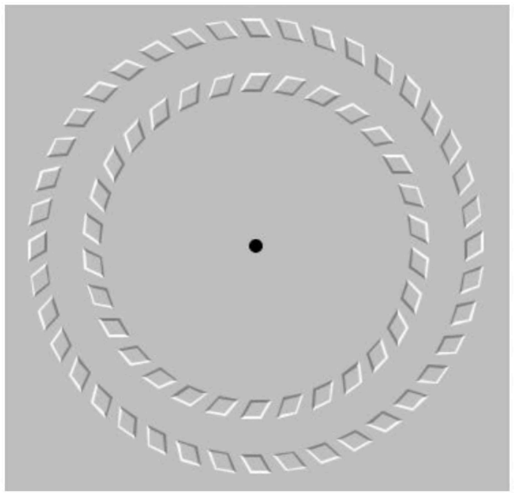
There are many illusion puzzles on the Cut The Knot website. These illusions can be incorporated into a single activity. When the user hovers over the illusions, they will be able to see the true meaning behind the illusion.
The illusions I plan to include in the activity are:

1. Judd Illusion
The player will be given the below image, asking if the point is at the center of the line or not. The halves appear to have different lengths depending on the direction in which the arrow heads point. Upon hovering the illusion disappears when the arrow heads are removed.

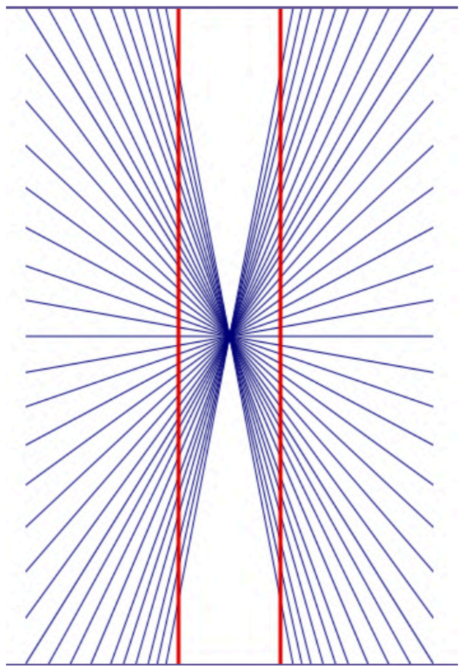


<u>2. Revolving Circles Illusion</u>

When you move your head back and forth keeping the focus on the center dot, the circles formed by the rhombi seem to rotate.

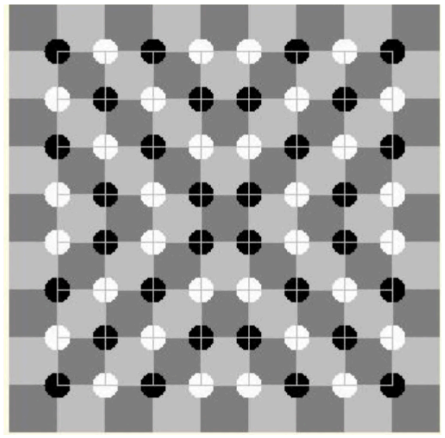


<u>3. Hering's Illusion</u> Looking at the red lines and seeing if they seem parallel or slightly bowed outwards. Upon hovering the cursor over the image to make the blue lines disappear, the player will discover that indeed the red lines were parallel



4. Bulging lines illusion

The user will be presented with the following image, with two options to switch the color from. In GLWB the lines appear to be bulging away from the center of the drawing. However, for GLBW they appear to bend towards the center.
(The color designation is G for gray, L for light gray, B for black, and W for white.)
Upon hovering, the circles will be removed and the illusion will be revealed to have straight lines.



If time permits I will also add more illusions to this activity.

Project Timeline

Community Bonding Period 1st May - 26th May	➢ Design various assets for the first 4 games. ➢ Implement game design and algorithms for different games. ➢ Create player flowcharts for all the games. ➢ Maintain regular updates with mentors and seek possible modifications and feedback on any game.
Week 1 27th May - 2nd June	➢ **Start working on the NIM Game Activity**. Develop the basic structure and functionality of it ➢ Integrate customizable settings to permit players to adjust game parameters and further Implement difficulty settings to adapt the AI's strategy. ➢ testing the NIM activity to identify and resolve any bugs or errors.
Week 2 3rd June - 9th June	➢ Develop game mechanics of **Lewis Carroll's Game of Logic** with existing assets, integrating player interaction. ➢ Describe JSON file for storing syllogisms ➢ Implement interactive features like tooltips and result messages (Fallacies & Correctness)
Week 3 10th June - 16th June	➢ Bug test the previous developed activity, Receive feedback from mentors, apply any changes if proposed. ➢ **Initiate development of the Candy Game activity** by designing the menu screen and implementing the pre-game window. ➢ Write and implement the application's logic,

	ensuring functionality, and conduct thorough testing to validate its operation.
Week 4 17th June - 23rd June	➢ Design in game visual setup and other functionalities, integration with assets. ➢ Create the end game screen and Test the activity thoroughly, identifying and resolving any bugs or errors ➢ **Start with the 4th activity, Number-guessing-game** and make the initial setup and menu for the activity.
Week 5 24th June - 30th July	➢ Make the option1 Gameplay for the activity and implement different difficulty levels in it. ➢ Start with the option2 Gameplay and test for the bugs in the Option1 Gameplay.
Week 6 1st July - 7th July	➢ Complete the option2 gameplay, and test for any possible bugs in it. ➢ **Document the work** done till now in these 4 activities, Continuously receive feedback from mentors and include them in the documentation for future considerations.
Week 7 (5 Days) Mid Term evaluation 8th July - 12th July	➢ Submit the Mid-Term Evaluation, By now 4 activities will be completed. ➢ **Design assets** for the next 4 games.
Week 8 (9 Days) 13th July - 21st July	➢ Start working on the **Latin Squares activity**. ➢ Make the menu screen and basic gameplay. ➢ Make a game loop and also introduce different

	levels in the game. ➢ Add score and High-Score features to the activity. ➢ Test for any bugs in the activity.
Week 9 22nd July - 28th July	➢ Start with the **Pascal-Triangle activity**. ➢ Make the visual representation for the activity without much interaction with hard coded values. ➢ Make block representation for the activity after the numerical representation is made. ➢ Later add an option to modify rows, modulo, and player for the pascal triangle.
Week 10 29th July - 4th August	➢ Test the Pascal-triangle activity and resolve any bugs if found. ➢ Start with the **Three-Utilities-Puzzle Activity**. ➢ Develop Graph representation logic. ➢ Adding a button to learn the solution to the puzzle is impossible. ➢ Add an explanation for why this happens and attach further resources.
Week 11 5th August - 11th August	➢ Debug the Three-Utilities-Puzzle activity and resolve any bug if found. ➢ Start with the **Illusion box activity**. Make the menu screen of the activity. Add different options to select any particular illusion. ➢ Make the First 2 illusions and test them.
Week 12 12th August - 18th August	➢ Start with the next 2 illusions and finally complete the activity. If time permits add more illusions to this activity.

	➢ **Document the final work**, seek feedback on the final documentation from the mentors.
Final Evaluation 19th August - 26th August	➢ By this time, Successfully **at least 8 activities** will be developed for Sugar. ➢ Submit the final evaluation for Google Summer of Code' 2024, and all the documented work.

Note: Testing the activity will involve checking it on different screen sizes to ensure compatibility across devices. and examining edge cases and ensuring error-free performance. Defining the user flow, and scope of these games in the community bonding period will ensure smooth making of these activities within the coding period.
Regular meetings and feedback will be taken from mentors to ensure smooth working and making of these activities.

Other Commitments

Sugar labs is the only organization I'm applying to and thus have no commitment under GSoC towards any other organization.
My summer vacations align with the GSoC timeline thus I have no special commitments towards my college, no exams are scheduled during this period either.
I would be awake 9:00 IST to 02:00 IST (as per 24hr format) and would be completely reachable within these times

Post GSoC'24 Plan

I aim to firstly keep these activities updated, the ones I worked on during my GSoC period and regularly maintain them. I also plan on continuing contributing to Sugar Labs after my GSoC period, in other activities as well. I also wish to become a mentor for the next GSoC session.