Feasibility Study

Group 4 | November 7, 2020 | Homebase Defence

Problem Definition

The application we plan to make is a combination of Simon Says and Asteroids. The Asteroids component of the game will have a 5-by-5 grid, and the Simon Says component will have a 3-by-3 grid. When the game starts, one asteroid will spawn in one of the boxes at the top row of the grid, and a second asteroid in the second row. For Simon Says, it will show the player a pattern, and the player will then have to memorize it and press buttons to match that pattern. This component of the game has difficulty levels to it. The first level has a 3-button pattern, and it goes up to a 7-button pattern. After 7, it resets back to the 3-button pattern but shows each pattern a little faster, thus, the time to memorize the pattern is less. Each time the player gets the pattern correct, an asteroid is destroyed, and points are awarded to the player. If the user gets it wrong, the asteroid moves closer to the spaceship, and another asteroid spawns at the top row of the grid. The health bar has seven hits, and when the asteroid hits the ship, the health bar decreases by one point. After 7 hits the ship will be destroyed and the game is over. The score is calculated by multiplying the number of asteroids destroyed by 5 times the round number (to incorporate the time complexity of each round for Simon Says). The score will be printed at the bottom of the screen, and at the end of the game, it will notify the player if the high score has been beaten or not.

The end-users of the software who will play the game are students, video game players, and Mr. So. The recommended age of the user for playing this game is 5+ because the game is simple to understand, but may still count as violence/action which is not suitable for children under 5. For students who play, however, it is a good way to build and expand their memorization skills in a fun way.

Problem Analysis

One problem that we did have while we were working on the game was that the size of the GUI frame didn't fit the screen in Repl.it. The title and many other components were getting cut off. To solve this problem, we decided to work on the code in VS Code by cloning the GitHub repository, which then allowed everyone to work on the code together. The game will not work on small screens such as a phone, but it will work on big screens such as a laptop as it is not developed for responsive design yet. This program should be able to work on Windows, macOS, and other operating system software that can run Java. The time required for the user to learn to use the program is

approximately 5-10 minutes because the application is straightforward and will be clearly explained in the user manual. As of right now, there are no time constraints because there seems to be enough time to complete this project. The project manager has set up a schedule for different components for the project to be done, and that is the only time restraint. Although we are in the last stages of development, the schedule is tight due to other commitments and may need to be modified or pushed back if need be. Although the deadlines are close, they seem achievable considering the well-structured breakdown of the project.

End User Requirements/Recommendations

We got into contact with Ayush Vora; a fellow student at North Park Secondary School on the 30th of October 2020. We conducted this meeting via a Discord call wherein we shared our plan of action and ideas for the game. We received valuable feedback about the structure and functionality of the game. In this meeting, we came up with the idea of how to store the high score as an XML file and came to the conclusion that using object-oriented programming concepts could allow us to split our work more easily thereby allowing for easier collaboration between our various members without affecting another's work.

Software Project Plan

Statement of Work (SOW):

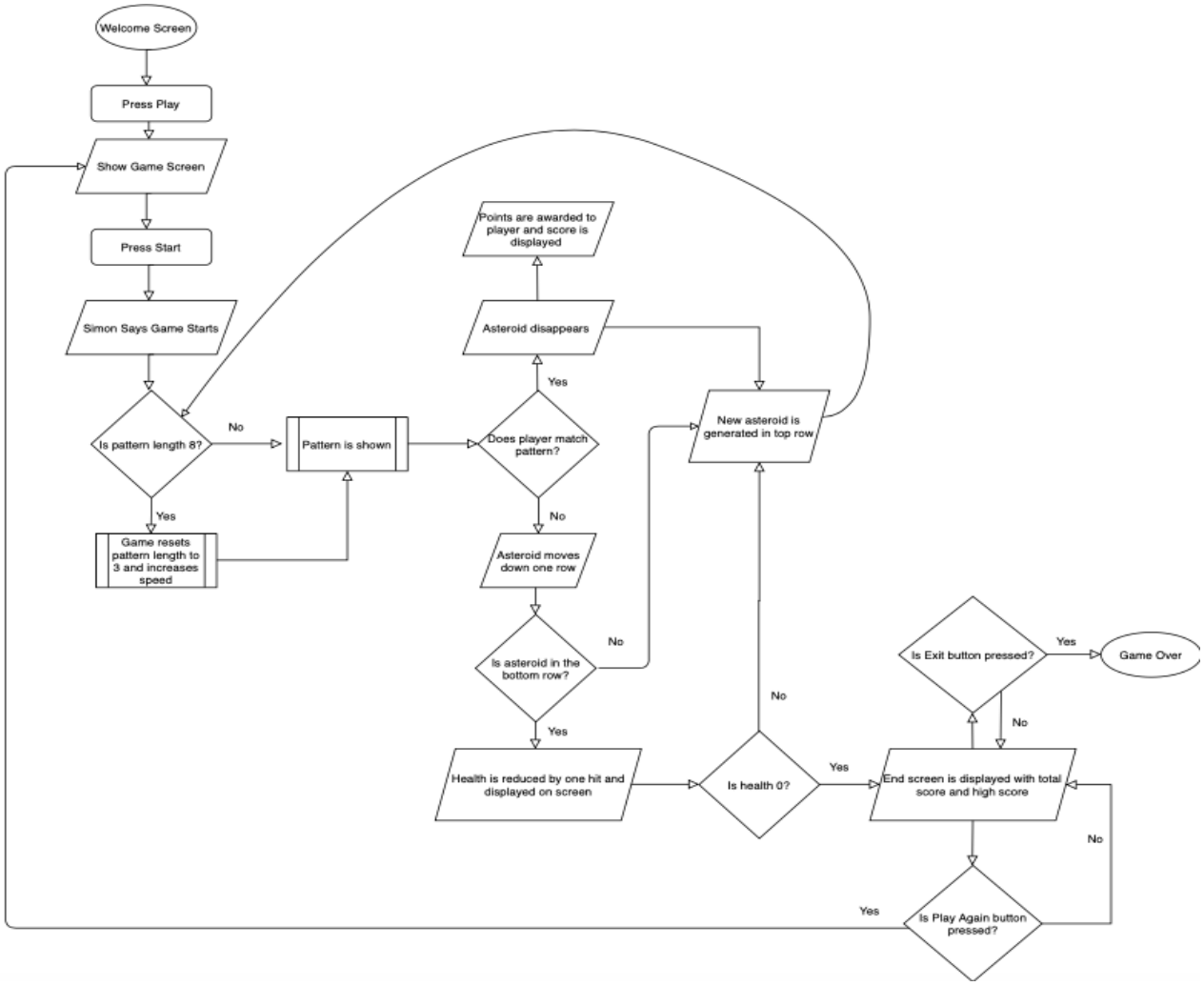
List of tasks

a. Scope Document: Report for an overview of the project. Created format, outline, sorted into different sections, assigned parts to each member, filled out, and formatted.
b. Feasibility Study: Report on the feasibility of the project. Created format, outline, sorted into different sections, assigned parts to each member, filled out, and formatted.
c. Introduction Screen: Opening screen to the game. Simply set up JFrame, welcome text, and play buttons.
d. Game (Asteroids component): Write code for asteroids component that generates new asteroids every round, shifts them, and uses if statements to check for destruction.
e. Game (Simon Says component): Creates a different pattern of blocks in a grid each round randomly. As the round progresses, more blocks are added to the pattern, and after the round is beat, the pattern is shown at a faster speed. Write code for the input part of the game where the user presses a set of buttons that is displayed in a pattern

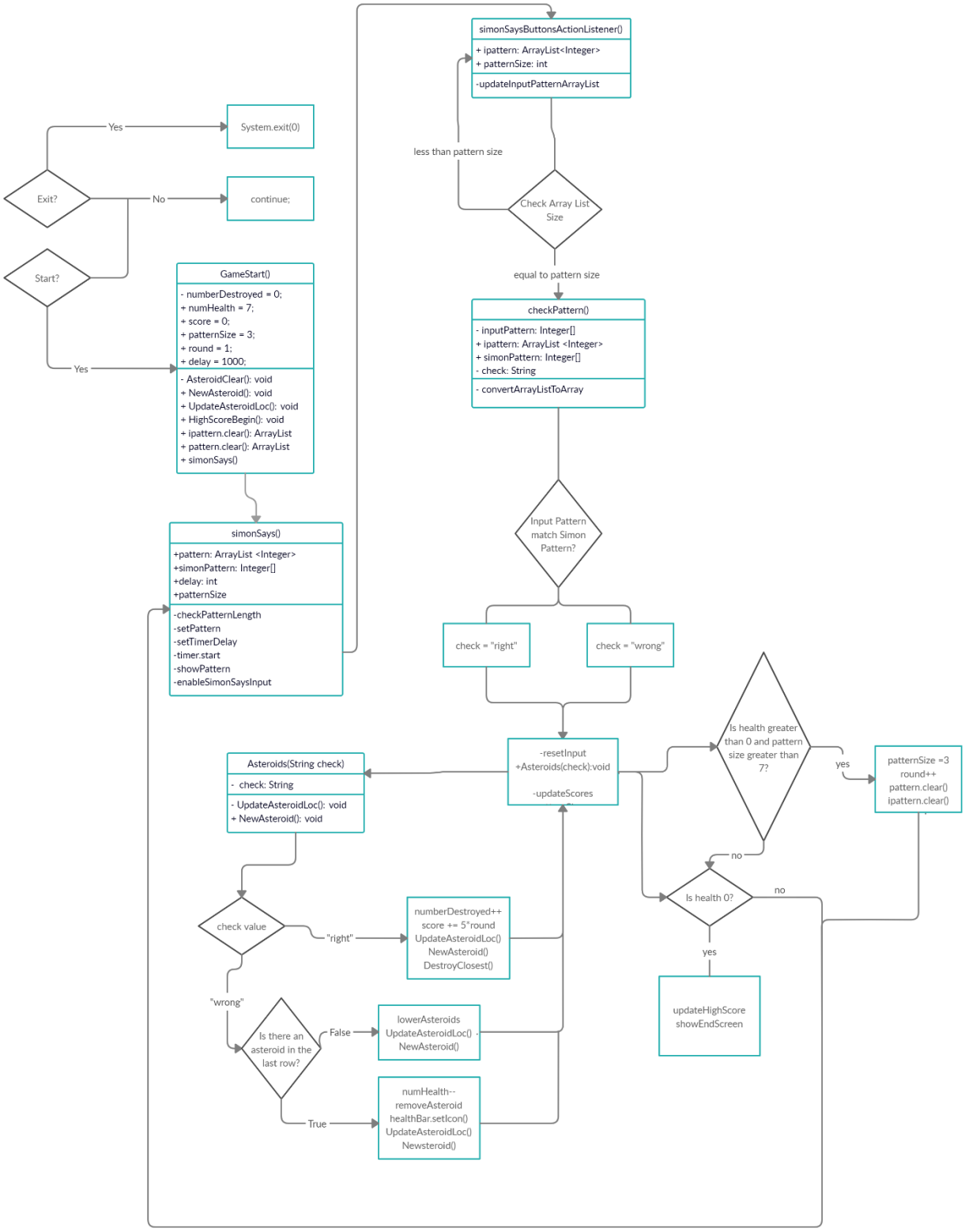
earlier. Uses if statements to check if the pattern is matched and destroys asteroids, else, asteroids shift closer to the Homebase/ship.

f. Game (Scores + Health bar): Increases score count if asteroids are destroyed and display an updated score each time input is given. Checks if an asteroid hit the mothership and if so, reduces the health bar.
g. Game Over Screen (High score + End): After player loses all health, display score, high score of all time, and end screen. Check for different cases; if a high score is beat, if a high score is tied, etc. Allow for replay, exit, and more.
h. Encapsulating the Game (Putting it All Together): Bring all components together using the same variables under the same project. Ensure the game goes from start to finish smoothly (no breaks).
i. User Guide: Create a document with different guides and documentation regarding the game. Create format, outline, sort into different sections, assign parts to each member, fill out, and format.
j. Testing/Verification: Everyone installs the project and tests it from scratch to simulate the end-user experience. Check for any bugs, exceptions, ensure all reports are written correctly, document/comment code, and place all files in one folder for submission.

Flow Chart displaying how the game progresses.



UML Chart displaying a basic outline of the code and its functions



Resource list

a. Stack Overflow - need a device connected to the internet (Available as long as the website is on)
b. YouTube - need a device connected to the internet (Available as long as the website is on)
c. Yahoo Answers - need a device connected to the internet (Available as long as the website is on)
d. Mr. So - Need to come to school, if not possible, have a device connected to the internet (during school hours from 8:30 am to 11:00 am, and 1:15 pm to 2:36 pm, from Monday to Friday, for quick response, else, email Mr. So)
e. Geeks for Geeks - need a device connected to the internet (Available as long as the website is on)
f. W3Schools - need a device connected to the internet (Available as long as the website is on)
g. Classmates - Need to come to school, if not possible, have a device connected to the internet, and need social media account (Ask questions to peer using social media)
h. Photoshop - Need the photoshop application

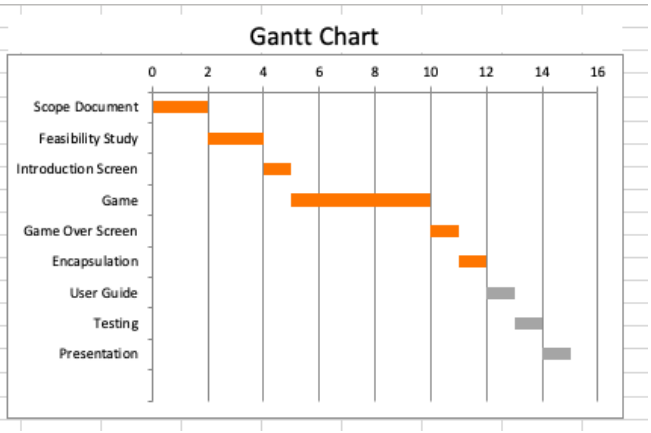
Work breakdown

a. Scope Document: 2 days, everyone works on it together (COMPLETE)
b. Feasibility Study: 2 days, everyone works on it together (COMPLETE)
c. Introduction Screen: 1 day, Harsh and Vinay set it up (COMPLETE)
d. Game (Asteroids component): 2 days, Sam sets it up (COMPLETE)
e. Game (Simon Says component): 2 days, Jaimil sets it up (COMPLETE)
f. Game (Scores + Health + High Score): 1-day, Harsh, Jainil, and Vinay set it up
g. Game Over Screen (End screen): 1-day, Jainil sets it up
h. Encapsulating the Game (Putting it All Together): 1 day, Jaimil and Sam set it up
i. User Guide: 1-day, Harsh takes lead, everyone helps
j. Testing/Verification: 1 day, everyone works together

Gantt Chart (As of November 7 - most recent):

Project Title:	Homebase Defense
Programmer's Name:	Sam Assareymuryil
Date:	07-Nov-20

SubTask	Start Time	Length of Task	Progress %
Scope Document	0	2	100
Feasibility Study	2	2	100
Introduction Screen	4	1	100
Game	5	5	100
Game Over Screen	10	1	100
Encapsulation	11	1	100
User Guide	12	1	0
Testing	13	1	0
Presentation	14	1	0



*Game components (Simon Says, Asteroids, health, and scores) are combined under the Game section.

Project schedule

*Note: the times listed are deadlines. Each task should be completed by the corresponding time. *

Thursday, November 5 - by 10:00 pm

- GitHub Repository & Repl.it Clones - Everyone (4:00 pm)
- Asteroids Code - Sam (6:00 pm)
- Simon Says Code - Jaimil (7:00 pm)

Friday, November 6 - by 10:00 pm

- High Score Code - Jainil (6:00 pm)
- Score Count Code - Harsh (8:00 pm)
- Health Bar Code - Vinay (10:00 pm)

Saturday, November 7 - by 4:00 pm

- Combine Components - Jaimil & Sam (4:00 pm)

Sunday, November 8 - by 10:00pm

- Installation Instructions (User Guide) - Jaimil (6:00 pm)
- Description of Purpose (User Guide) - Vinay (6:00 pm)
- Game Instructions (User Guide) - Harsh (6:00 pm)
- Special/Key Features (User Guide)- Jainil (6:00 pm)
- Screenshots (User Guide) - Sam (6:00 pm)

- Format the Final Document - Harsh (10:00 pm)

Monday, November 9- by 10:00 pm

- Game Testing + Debugging: Jainil and Sam (6:30 pm)
- Presentation + Preparation: Everyone (10:00 pm)

Risk plan

a. Not enough time:

Though we estimate to finish the code of the game by Friday, November 6th, problems can occur which could potentially delay the entire schedule. Problems such as: the internet going out, family issues, and computer crashes which can corrupt files. To solve this potential problem, our group decided to leave a day free on Saturday, November 7th. This was originally made as a rest day, but it can be used if necessary. This leaves our group an entire day to finish the rest of the code.

b. Code not uploading to GitHub

Another potential problem is the code not uploading to GitHub. This can occur when one is not able to pull any files due to issues such as the git process being crashed in the repository. Problems like these are tricky to solve as sometimes the error itself is confusing and hard to understand. Using online resources such as YouTube or peers in class can allow for this problem to be solved clearly.

c. Uncompleted work from group members

Though highly unlikely, another potential problem is peers not completing their required parts for the project on time. This can cause disorganization with the schedule as some individuals have to do more work to complete the unfinished part before the deadline. This can be solved easily as our group set deadlines leave much time left. Within this time, we can complete the unfinished work before the project is due.