Grade: 85/100

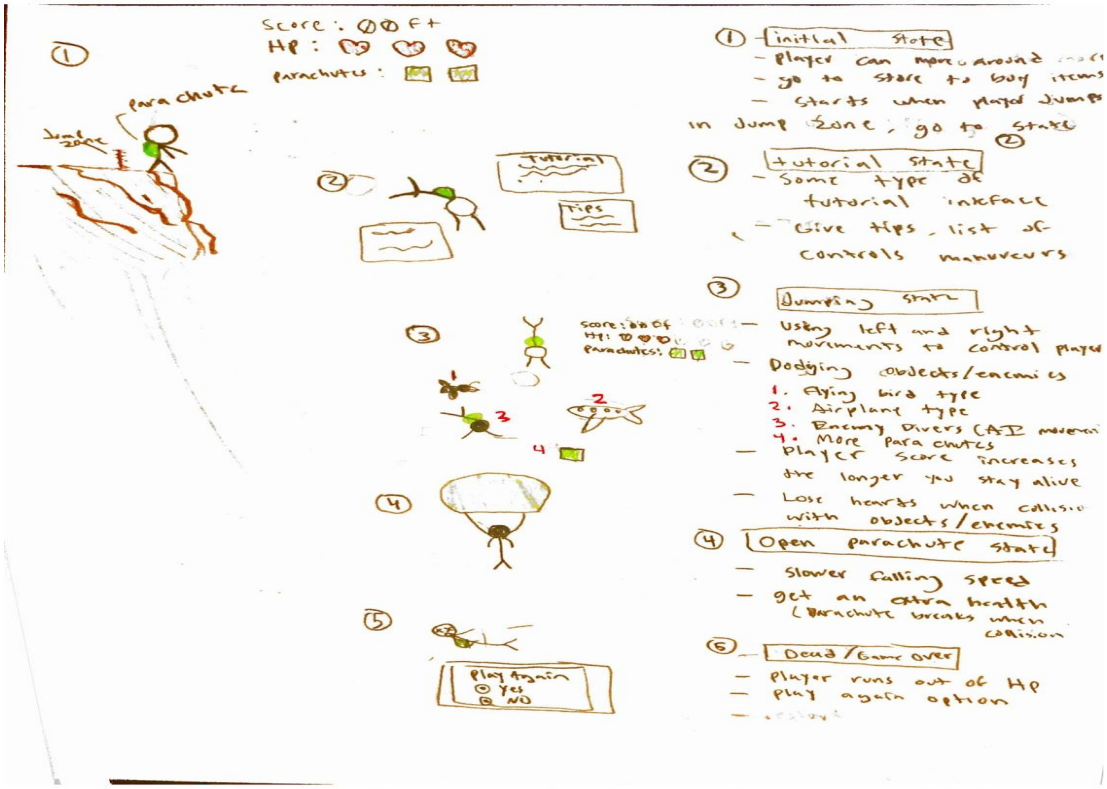
- The front (cover page) material is missing.
- The Gantt Chart needs to explicitly indicate which team member will carry out which tasks.
- Mastermind games need storyboarding similar to the daredevil game (no matter how simple it might be).
- The details of the Coggle chart need to be explained in the text. You can't just simply drop a chart or a figure in a document without explaining the context or the details.
- The discussion of "design choices" is poor and not really explaining the "why" of the choices.
- The mother game (incremental) design and relationship among different games, data flows, incentive (bonus) management, etc are discussed in general terms without blockdiagrams/storyboarding to outline hte specifics.

C. Solution Approach

Concepts, Algorithms, or Other Formal Solutions to Build the Product

In order to develop the new daredevil mini game, Stefon has to first flesh out a visual storyboard of how the game play would work. The storyboard was drawn out to illustrate the many states of the gameplay, for instance, there is an initial state of the game where the user is in control of either walking to the store to buy items or walking to the jump zone to start the actual daredevil like gameplay. The storyboard also included objects that would be used in each state like for one the enemies and dropped items would only be enabled in the diving state. The restart game menu would be enabled in the try again state after you die. At this time all game processes will sleep and if the user decides to try again you will start all over again to the initial state of the game where it was explained previously.

Daredevil Mini Game Draw Up Idea:

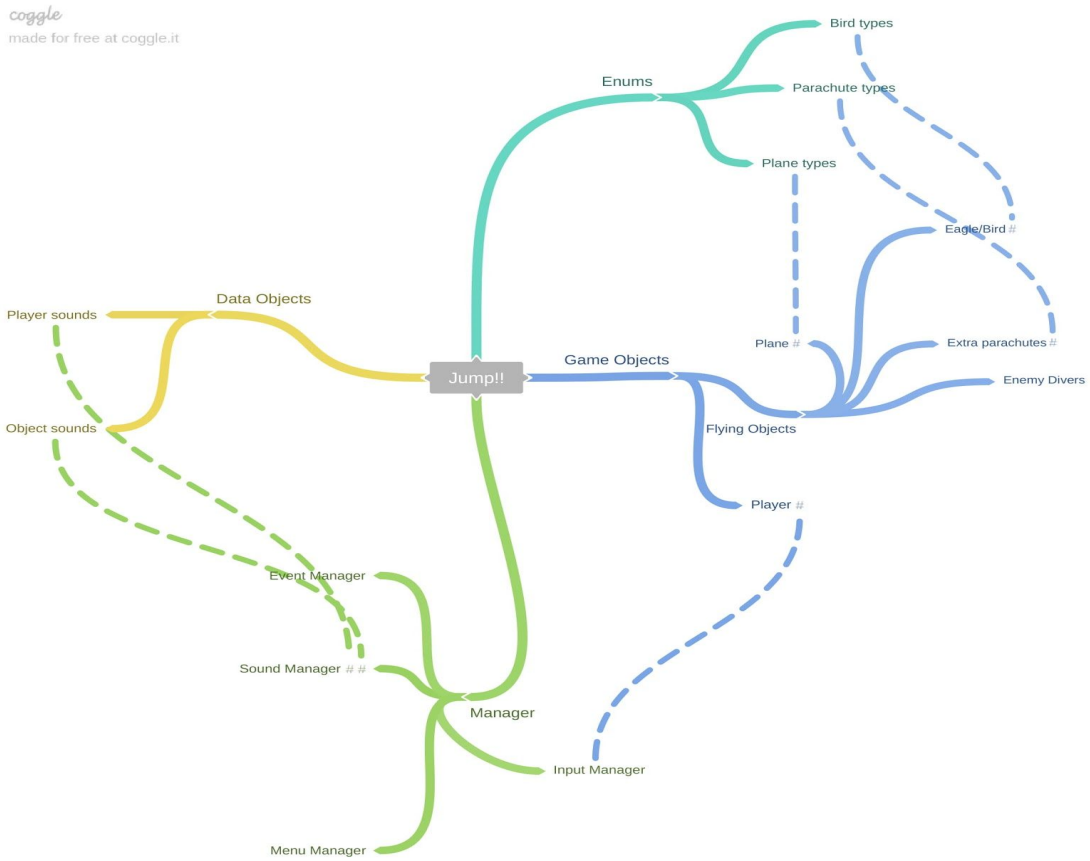


For the second new mini game, the main part of developing the collection of mastermind games is the underlying data structure that forms the game arenas. It is organized as a two dimensional matrix. The algorithms are quite simple in checking the correctness of each cell value. They only differ slightly between Kakuro and Sudoku. They are simple matrix algorithms. The other big part is connecting the front end to the game logic. Brendan needs to make sure that the graphics of the game are controlled by the game logic in a seamless fashion. Kakuro and Sokoban will have very similar menu interfaces.

Solution Selection Process - Discussion of Design Choices

After the storyboard is mapped out we could use the ideas and object from it to implement a diagram. Initially the idea was to create a class diagram but we decided to use Coggle to construct a diagram for the daredevil mini game. Coggle is a collaborative mind-mapping tool that helps you make sense of complex things. The coggle diagram helped showed the relationships between the game objects and states in a tree like structure. This will later serve to be useful during the coding process which has not been started yet for the daredevil game.

Daredevil Mini Game Coogle Chart:



The team is using the previous Sudoku mastermind game code as a framework to develop the new Kakuro game. This will be the most efficient way to develop the new mini game because the game arenas are similar. They are laid out as a two-dimensional grid in which each cell has a relationship to the other cells in its respective row, column, and neighboring cells. The Sudoku classes can be modified slightly to find these relationships.

Sokoban also has a similar game arena as the mentioned mini games. It is a two-dimensional grid in which a character pushes or pulls blocks to free themselves from a simple maze of sorts. The main focus of development will be on the game arena development and the way the player can interact with the blocks. There will be JSON files or another text file of some sort to set each level. This should be easy to customize new levels of varying difficulty. It also should be easy to test. Both mini games allow for a certain aspect of randomization of each level, which will cohere with our reward system that we have in place.

Detailed Implementation Framework

The team is developing the game using Unity. Unity allows us to model our games and develop robust code that we can apply to other mini games we might develop later on in the project or even teams after us may use. Our team plans to keep using Unity to develop and clean up the current “Mother-Game”, current mini games, and the new mini games. This software goes great with our current project and it work well with the database we are using.

MongoDB is another key software that we are using. It is a document oriented database program that makes use of JSON. We will use the database in order to allows us to save player data. The use of this database is critical to our project since we don’t want player losing all their progress after playing the game for an hour. It would negatively affect player experience if they had to restart from the beginning of the game each time they turned on the game. MongoDB is an off-the-shelf software that we will be using, but the team will need to learn and use the database schema the previous team has come up with.

The data components we are handing are the player usernames, emails, and passwords at the moment. The MongoDB database allows us to also store the players level, passive and active experience, coins, items, and overall general progress in each game and overall in the “Mother-game”. The data components will end up being something that has already been developed and stored correctly into the database. Our team's job is just to manage the data being stored and to check to see if there are any errors or flaws in this data. Another important connection our team will have to make to the previous years team will be checking to see if the data from the new mini games is being stored correctly once they are developed and deployed.

The hardware we have in use currently would be the EECS server which is currently hosting our game and its data. This is something we won’t have to assemble, it will be another off-the-shelf component in our project. Until we can get the game online using a web host we will continue to use the EECS server as our main housing for the game.

All of the software and hardware items are very closely tied together. If one items above breaks it may cause the entire game to go down. Our team needs to very careful in

documenting code when we change something. If the database goes down all user data should be correctly stored, saved, and recovered so progress isn't lost. If the EECS servers go down this may affect the players ability to log on and access the game. It is crucial for all the above parts to work efficiently for our project to be successful.

Throughout the semester we will be in touch with our mentor to get feedback weekly on the features we implement in the new minigames or even in the "Mother-Game". If Chris doesn't like a feature or wants to change something the team could split up the job and get it done. As a development team we have agreed that it is necessary to get feedback from Chris before and after implementing a new feature to the game. This way his vision for the project isn't lost.

Preliminary Software Testing Plan

To test the "Mother-Game" and existing three mini-games, the team has decided to individually play the game every week to see if the game difficulty, progress, and flow seem to work cooperatively. Our mentor Chris, will also help us by letting us know if the game needs to change something in order to better accommodate his research or even the students that are playing the game. If bugs are found during our gameplay we will notify Nigel. If the bug is small, the individual that has found the bug will be able to go in a fix it.

To test the mastermind and daredevil mini games, we will create JSON files to set custom stages. This will be quick and efficient as we can quickly parse multiple JSON files and test the different states of the game. We will be able to see which custom stages show errors, make tweaks, and redo the testing. Furthermore, we will use game testers, such as our mentor, the team members, and others to play the game on their own time. We will check back in with them to ask about their experiences and any other useful feedback they have for us.

Summary of the State of This Project

As of now the current game is coming together, there are still a couple loose ends that we need to fix. An issue with the "Mother-game" is that the player can level up during game play but the levels don't mean much as of now. Another problem with the incremental game is making it actually have a reward for leveling up. Implementing a leaderboard where players can rank up against each other would be a good way to inspire the players to play the game and level up (which will tie directly into classroom effort). This leaderboard will also need to be implemented on the server by giving each player a ranking variable and then writing a script in C# that can display this leaderboard from the game to the player. With the leaderboard an additional feature will need to be implemented. Not every player is competitive so we would need a way for them to not participate and not view the leaderboard. For these types of players a reward for leveling would need to be implemented, but we still need to discuss and figure out the best way to handle this, potentially new items could be added for them. We know that in the original game Anti-Idle (which our game is based on) that each time a player levels up they unlock a new minigame, so if we can get the new minigames implemented maybe can have the player level up a certain amount before they can unlock the new daredevil or mastermind minigame. The "Mother-game" had a couple issues in the tutorial section, but that has been

debugged and fixed for a mini-deployment. Our mentor needs to conduct some research on the classroom gamification, so we have been given our first deadline to get the current game in a working state with less bugs. Additional problems currently in the “Mother-game” include an achievement list that does not currently display anything. This will need to be tied into the rest of the minigames and the server. The last feature that needs to be fixed with the main incremental game is implementing boosters that can help out the players incremental progress as a consumable item. These boosters will be found in the minigames, and are currently not implemented in the game.

The RPG type game came in very rough shape when we started the project. There was a lot of features implemented, however some very critical bugs that prevented progression through the in-game tutorial needed to be fixed in order for the actual game to be playable. After a few weeks these bugs were fixed and the game is currently playable and ready to be deployed. Additional progress that needs to be made on this game is balancing the procedurally generated dungeons, and adding more quests to the main story. Refining the main gameplay loop of going into dungeons and selling/crafting items after finishing will be the next job to do on this game to make sure it is fun for the player to do over and over. If time permits additional quests will be added to the main storyline and more items will be added.

The mastermind game didn’t have many bugs in it, other than some small user interface errors. There is not an intuitive usability to the general user interface. For example, after the player presses the button to begin playing the game, it is unclear as to what the difference between the random stage and set stage modes are. In fact, there is no explicit description. There is also an option to save your game midplay. However, this is also not clear to the user. These simple usability faults can be solved with redesigning the interface with the user in mind. A tutorial will be added within the settings module to break down the functionalities of the menu and the gameplay. If possible, we may be able to include a quick thirty second video breaking all of this down.

As far as the new mini games go, the design will take from the Sudoku code base as the games are fairly similar in certain ways. The two new mini games are Sokoban and Kakuro. Two traditional puzzle games that are very popular to many audiences. Incremental progress on the production of these games will be measured with the delivering of prototypes during our weekly mentor meetings. We will need to proceed carefully, as we make sure to manage the mastermind game data with our MongoDB properly the whole process through.

Besides the small usability issues, the game is in good condition to be deployed in its current state. Players can clearly feel the difference in difficulty levels and the game is fully functional other than that.

The conqueror game is one of the minigames that is in really bad shape. Last years Senior Design team seemed to have rushed the game development process on this minigame causing it to have many game play flaws. As soon as the player entered the minigame it was hard to them to understand exactly how the game worked. The tutorial section of the game was nonexistent, the directions for the game were very short, and game play was terrible. As soon as the player clicked start s/he would immediately get shot by the enemy. There is no countdown menu before the game actually begins. The player was easily able to maneuver around the arena, but wasn’t able to shoot bullets at the enemy very well. The bullets the player

shoot are slow and not accurate according to where the player wanted them to go. The biggest issue the team ran into was when we beat the first level the second level is very hard to get past. As soon as you beat the first level you are immediately thrown into the second level where there are a lot of enemy projectiles flying around. It makes it hard for the player to dodge anything and the player dies in a matter of seconds. The last major issue is the gun acquisition. Currently, the user gets guns by destroying enemies and moving to the next level, but that is very poorly implemented. Selecting a new gun crashes the game for the player, so the player is stuck with only a level 1 gun. As a team we decided to get rid of the current conqueror game in order to develop an better one in its place.

The new mastermind game is still in its early development phase, Brendan has been looking through the various kinds of mastermind games that are available and easy to build. The difficulty has been being able to find a game that is enjoyable for the user to play. Currently, the only information on the new mastermind game is that it will be similar to game known as Kakuro. Brendan has been in the process of finding ways to take the already existing Sudoku code and implementing the new mastermind game with it.

The new daredevil mini game is still in its early development phase. Stefon has been able to draw up some high level gameplay diagram which will later need to be detailed for implementation. The game development is in its early stages. In a few short weeks we hope to start getting some coding underway.

Gantt Chart:

ID	Task Mode	Name	Leveling Delay	Duration	Start	Finish
1		**Gamification**	**0 edays**	**44 days**	**Tue 2/27/18**	**Fri 4/27/18**
2		**Bug Fixes**	**0 edays**	**44 days**	**Tue 2/27/18**	**Fri 4/27/18**
3		Incremental Bug Fixes	0 edays	44 days	Tue 2/27/18	Fri 4/27/18
4		Conqueror Bug Fixes	0 edays	44 days	Tue 2/27/18	Fri 4/27/18
5		Mastermind Bug Fixes	0 edays	44 days	Tue 2/27/18	Fri 4/27/18
6		**New Mastermind Mini Game**	**0 edays**	**44 days**	**Tue 2/27/18**	**Fri 4/27/18**
7		UML Diagrams	0 edays	4 days	Tue 2/27/18	Fri 3/2/18
8		Game Implmentation	0 edays	29 days	Mon 3/5/18	Thu 4/12/18
9		Testing	0 edays	11 days	Fri 4/13/18	Fri 4/27/18
10		**New Daredevil Mini Game**	**0 edays**	**7 days**	**Fri 3/2/18**	**Mon 3/12/18**
11		Game Implmentation	0 edays	6 days	Tue 2/27/18	Tue 3/6/18
12		Testing	0 edays	2 days	Mon 2/5/18	Tue 2/6/18
13		**New Conqueror Mini Gam**	**0 edays**	**44 days**	**Tue 2/27/18**	**Fri 4/27/18**
14		UML Diagrams	0 edays	4 days	Tue 2/27/18	Fri 3/2/18
15		Game Implmentation	0 edays	29 days	Mon 3/5/18	Thu 4/12/18
16		Testing	0 edays	11 days	Fri 4/13/18	Fri 4/27/18
17		Web Interface	0 edays	10 days	Mon 4/16/18	Fri 4/27/18