Software Design
Spring ‘18
Group Cetus
Development Plan

Abstract: This document is meant to review the portions of the programming assignments along with breakdowns and reviews of what the team did. The plan is set up in phases and also a brief overview of the phase itself. Along with the ***Phases of Development*** there is an overview of the teams dynamic called the ***Program Review*** that follows.

AAR - After Action Review/Report

Phases of Development

Phase 0: Construct layout of buttons and establish development pipeline. 

Overview: Phase 0 was the basic phase of getting the main game window setup and get the group to develop a plan for the future of the program, so to avoid possible issues with integration of new techniques later on down the road.

AAR: Team built basic structure well and planned for future developments.

Phase 1: Integrate game tiles into game window 

Overview: Phase 1’s main goal is to establish picture tiles into the game. Along with that, the team will give the tiles abilities to swap position with tiles inside the center grid.

AAR: Team constructed communication protocol for the game. Tiles communicate to each other via the game window. Once a tile is pressed, if another is pressed, the data in each are swapped (if legal). Along with that, some beautification was done to the game layout.

Phase 2: Implement communication protocol. 

Overview: The task for this phase to to successfully get the game logic working along with data pass between tiles.

AAR: The tiles are able to switch information. As of now we are passing colors while we wait for the game tiles.

Phase 3: Create and Implement .mze streaming protocol 

Overview: Purpose of this iteration is to create the streaming and delivery method for the .mze files given to us from the instructor. Not only from reading in the int vals and float vals, but integrating these values into the maze level design.

AAR: One of the better iterations for our team. Work was shared well between all the team members. Along with division of work, the team came together and accomplished the task in a timely manner.

Phase 4: Add rotation to tiles and randomizing start 

Overview: Phase 4 involved getting the rotations to work with the tiles along with creating randomization of startup and error notification.

AAR: Our team did extremely well at this iteration. From our planning, we were able to pretty much accomplish everything required of us in a timely manner. We were able to complete over 50% of the project requirements in just a couple hours working together.

Phase 5: Allow player to load/Save game 

Overview: The fifth phase will involve attempting to setup the load and save system within the game, to give the player a more full and fulfilling experience while playing our game.

AAR: During this iteration, our team successful created a save/load routine for the maze game. The team all came in on their weekends to complete the task, and for the most part, everyone stayed on task and completed what needed to be done in a timely manner.

Phase 6: Players Win! 

Overview: The final phase of this program will be having the player win!

AAR: This phase was easy for the team. Why? Because as a team we collaborated and constructed the code so well that it led to this iteration to fit perfectly into our existing code.

Program Review

With the program finishing up and the semester coming to an end, we would like to recap all of the phases. The document to follow will be split up similar to the phase diagram above, but with each phase being expanded upon.The focus of this review is not the assigned tasks for the iterations/phases, but how the team tackled the specifications and division of work along with the camaraderie developed throughout the semester.

Phase 0:

Phase Zero was the intro phase that helped to develop and discover our team and how the roles were going to be given out throughout the semester. With the project task of basically building the foundation for the code and the structure of the game window, our team dynamic started to shine through the overcast pathway which laid ahead of us for the remainder for the semester. The tasks sort of feel into place; Robert was our “go-to” coder while Gokul, Kevin and Austin worked together to be our main code group. My [Nic’s] main task was to tackle documentation along with assisting the team when needed. In summary, Phase Zero was less about coding and more about coming together as a team and finding our common grounds.

Phase 1:

After our team found our groove, phase one was where we really needed to start communicating about planning for a future. With the focus of the phase one iteration to build the tile windows, the communication protocol throughout the game needed to be setup in such a

way that it would set us up for the future. Along with building the windows, as a team, this is where more tasks were divvied out in order for us to effectively meet the timeframe. As a team, this is where the time taken to get to know one another paid off. We quickly and concisely split up tasks and met the deadline in the right time, with plenty of time to spare; Planning and teamwork won again.

Phase 2:

Phase Two was one of the few iterations where the team didn't have to do much mostly because we did the hardest parts in the phase prior. There honestly isn't much to recap phase two, so instead of wasting your time, I will move on to phase three.

Phase 3:

Phase Three was the team development phase. With a file streaming protocol needing to be put in place with assignment constraints, the team had it's members shift to take on this difficult task. Usually, Robert wanted his hand at everything, but this iteration, everyone wanted to test out their Java abilities to see if they were capable enough to tackle the burdensome task; And the team came out on top! With the main tasks being taken by Gokul and Kevin, the two set out to create a read-in capability that fit the given requirements and also made the code slightly future proof and robust for the iterations to come. The two worked exceptionally together and brought a finished product well deserving for us to get a solid 95/100 for the programming assignment.

Phase 4:

As the school year went on, the team got more and more busy, not being able to devote as much time as they could by themselves to the project, but that didn't stop good ol' Cetus. Might be cliche, but the team once again, defied the odds. Instead of assigning tasks and meeting back up the following week, the team set aside some time to work together at the meetings to complete a majority of the specifications within a decent amount of time (1-2 hours). With the focus of this iteration on randomization and making the game more dynamic (not in the programming sense but the actual definition). The hardest thing our team found for this iteration was giving the tiles the abilities to rotate and reprint the tiles to match. We processed many ideas on how to tackle the issue, but in the end, the team came up with an algorithm that worked well and worked every time.

Phase 5:

The school semester was drawing to an end and with that, so were the myriad of projects the team members faced in their respective classes, but you guessed it, team Cetus came out on top! The focus of phase five was to create the save/load protocol for the game, a task that was not easy in it's own right. The team needed to split up more and more of the code than ever before, even giving Nic more code to tackle. During this phase, Gokul took the saving,

Kevin took the loading, Austin took the design and implementation of the save window, and Nic (along with the helpful hand of Robert) was tasked with transitioning and updating the reset feature in the game window. Phase 5 was hands down the most involved phase the team had to date, but the team went head first into the fight. From coming in on the weekends to strong communication throughout the timeframe, the team finished up a save/load protocol that we felt was worthy of the program.

Phase 6:

"*It's the final countdown".* Surprisingly, this was the easiest phase for the team! Due to the code construction previously designed in phase 3, the team merely had to compare the positions of the tiles with the location they were in the load array (the array the tiles were loaded in when the file was initially setup). With this, the team's main objective was to complete the code and provide documentation that supported the program outside of code comments.

RECAP:

Throughout this program and class, the team developed a very robust and complex game (joke). Although the game itself is simple, the code constructed by the team was not. From designing the foundation for the code to building the interfaces accordingly so the future iterations of the game came and went without a ton of code reconstruction, team Cetus developed and, more importantly, delivered a finished product worthy of the title, "*Pretty decent*". The maze program was an excellent learning experience not only in code, but also the development strategies needed for efficient software design.