Nathan Boersma, Sage Hackbarth, Michael Harmon, Peter Hapke, Michael Olson, Austin Stoulil

4/19/2018

POS Games

We decided as a group that we were going to stay together for the second group project because we all decided that building games would be enjoyable as well as easily broken up into separate pieces for each member to work on. We also added Austin to our group and we deciding that building games would help with eliminating the possibility that some members may not have anything to work on. As a group, we decided to build mini games. These games include: Space Invaders, GO!, Memory/Simon Says, Three Card Monte, Pong, and a scratch ticket game. This allows for variety and allows members who wanted to use graphics to be able to incorporate that in to their games, while allowing other members to stick to text-based games if they desired. We also decided on mini games due to them being fun and something that we could/would actually play in real life.

One of the goals of the group was to be able to incorporate graphics in to at least one game but more games if possible. Another goal that was set was to be able to have each person create a game. This was a goal that at first, we were somewhat flexible on. We had discussed creating groups and having each group create a game or two but we were lucky enough to be able to have every member contribute a game to this project. A goal that we were unable to completely achieve was to emulate cout and cin. This created some problems for a few games that were originally coded in Dev C++ and converted over. Peter tried to make it as seamless as possible but there was unfortunately no good way to make it work without changing the code in the conversion process.

At the bottom of the report, we have decided to add in more in depth answers from each member of the group we will also use a brief overview of each response. Peter Hapke

contributed approximately 25 hours and was able to accomplish: building the Main Window, create the game GO!, incorporate external exe interface for text-based games, diagnose and resolve numerous bugs, and gathering and compiling/incorporating code to a working state. Austin Stoulil added between 20-25 hours and worked on the Pong game. A major accomplishment of his was making the game 2 player which allows it to be more enjoyable. This quite a large undertaking as it required allowing 2 players to play as well having many moving pieces that involved boundaries, collisions, and score tracking. Nathan Boersma contributed 18 hours to the group project. He decided to stick with a text based game because it was something that he understood and felt more comfortable doing. An accomplishment of his was being able to create 2 versions of the Memory game. The game allows a user to play a color version and a number version. Another accomplishment was being able to incorporate time limits for users to give answers as well adding different levels for the players.

Michael Harmon worked about 10 hours on Find The Ace and Guess My Number games. A major accomplishment for him was that he was not familiar with windows forms using C++ and was able to learn a lot about how it works. The Find The Ace game displays 3 cards and the player is to pick a card until the Ace is revealed. The Guess My Number game allows the player to enter a number within the range and lets the user know if they are close to that number. He struggled with using the form program style. One of my main struggles was figuring out how to randomize the cards which I was ultimately unable to accomplish. He was able to randomize the number guessing game. He also struggled with using GitHub.

Our group faced many individual challenges but we didn't seem to face many large group challenges that some other groups may have faced. We believe that this is a result of having already worked together for the most part. We understood how we worked and thought which allowed us to be able to stay ahead of the tighter timeline for this group project. Some of

the individual problems we faced were problems with Visual Studio's. These VS problems included: Winforms, downloading Visual C++ packages, using files from GitHub, configuration problems, and many others. Luckily, Peter was able to help with a majority of these problems. Another challenge that our group faced was trying to figure out how to work with Winforms. This took and added a lot of time for some of the group members.

One problem that was a challenge to both individuals and to the group as a whole was with Github. This was a result of GitHub not knowing how to handle certain files that Visual Studios adds. This was a challenge that we were never able to fully fix so we had to come up with a workaround. The workaround was to send Peter the files and have him add them in to the "master" game due to him being the one that oversaw the GitHub project files. In many instances, this took hours for some to workaround or resolve.

A major accomplishment of the group was to be able to submit a finished project for each person. Some games may have some glitches or may not be working 100% like the actual game but for each person/game we had working pieces. This is a major accomplishment due to the limited time that we had for the second group project. A few ways we could build on our project is to: add scoring for every game, add more games, add graphics and winforms to every game, eliminate flickering in the Pong game, add menus for each game, and make the main menu as well as individual game menus more appealing. These are just a few ways that we could/can add to our project in the future.

There were a lot of individual accomplishments that include things such as: creating a multiplayer Pong game that Austin was able to accomplish. Another individual accomplishment was Peter being able to assist and create the framework so that the rest of us were able to be able to incorporate our games. See below for Peter's breakdown of accomplishments, challenges, and goals. Nathan's accomplishment was being able to use colors as well as being

able to create 2 different versions of the Memory game. Michael O was able to get a working prototype for the game Space Invaders which was an accomplishment considering how complex the game is to build. Michael H. was able to build 2 working games and he learned and incorporated many new things that he had not previously known. Sage was able to create a nice looking scratch off game which completed our team's goal of having contribution from everyone.

Not every game is functioning exactly how the individual wanted or desired but the fact that we were able to use code from every member is an accomplishment when considering the complexity of some of the programs.

Peter Hapke: Time - 25 hours

Accomplishments - Main Window, Go!, External exe interface, Various debugging, Compilation

For my part of the project I worked on creating a framework that would allow the group to create smaller programs and then plug them into the framework. To accomplish this I set out to create a WinForm that would redirect to another winform, maintaining positioning, and then would resume when the new window was closed. This separation allowed the group to create a series of programs and compile them together later. I encountered some difficulty with the positioning element of this framework because the WinForm auto generated code sections are extremely sensitive to alteration. I had tried to pass the position by constructor, however that broke the window and would erase code when run, as the automatic code system tried to fix the

problem by restarting. I ended up being able to change the the position by setter method once it was constructed.

In order to open up the possibilities for games created by the group I tried to find a method to include simple console programs with cin/cout. Originally I looked into spoofing the cin and cout functions so that those commands inside a working WinForm would redirect accordingly and thus leave us with a WinForm shell but a text based console logic, unchanged from what it would normally be. This proved to be successful in spoofing cout; however, cin would require a blocking call that I could not come up with a good/easy fix to correct. All the fixes possible would require tying the spoofed cin/cout more directly to the individual WinForm and altering the standard code too far away from a console application. I ended up scrapping the idea and instead elected to create a helper function that would simply open a new console window by opening a separate compiled exe file. This proved successful and was implemented into the final program. THere was some difficulty in creating a helper class because of the managed class and system scope, particularly that std::string is not the same as System::String. Opening an exe does work but has a incongruity with the design of the program, neither its position is set to be the same as the main window nor does the main window hide upon creation and disappear when closed. I tried to create a callback/event hook that would accomplish this but was unable to do so because they are separate processes when started in this way.

The board for the game of go was fairly easy and straightforward to program as a two player game, creating an AI was not attempted because it is a longstanding difficulty of programming only recently being able to beat a person. The go game itself is simple a number of tiles that need to be toggled on and off for which I used a checkbox object, skinned as a button object to give the appearance of a toggle button. Each time the button is pressed the button is disabled and colored according to the current players turn. I had some difficulty in this

simple setup because I did not realize at the start that a managed class cannot have any members that are not pointers. The difficult part of this program was to create a function that would look at the board itself and according to the rules of go would eliminate any connected series of pieces that no longer had viable moves. The program needed to cycle through all of the squares which took quite a bit of resources and needed some optimization so that it would not get bogged down. Because it was also possible that the order it scanned objects in could result in two objects colliding it also needed a way to merge two objects before it would consider whether they should be deleted. I did not program into the game two generally agreed upon rules as a result of the optimization, resolving a ko fight nor preventing suicide moves. This would not affect the scores of players in the former case and in the later would only be a problem for players that wanted to lose anyway.

Austin Stoulil: Time - 20 - 25 hours

Accomplishments:

For my part of the program, I decided to work on creating a Pong game using the windows forms graphics. I created this by using shape objects for rectangles. The game is a two player game that lets you play pong. The game displays the controls at startup and includes a pause feature. Player 1 moves up and down using the w and s keys, and Player 2 moves up and down using the numpad 8 and numpad 5 keys. You can pause the game by pressing P on the keyboard. I think that it was a good first attempt at using the windows forms graphics objects in order to make a game, but there are still some things that I would like to improve on.

One thing that I would like to improve would be the flickering issue when drawing the graphics. In order to make the graphics animated, I needed to update the position of the pong ball and paddles, and then redraw them in a new location and clear out the old graphics. Due to

this it creates a strange flickering effect while it runs. I would also like to improve on the hit detection of the pong ball with the paddles, as it can be a little bit glitchy at times. One of the other main things that I would have liked to improve would be the response time when holding a key to move. I tried looking up ways to remove the initial delay between when windows checks for if a button is being held down, but it proved to be too big of an issue to fix before the presentations.

Overall though I think it was a good first experience with using the graphics objects in Visual Studio. I haven't gotten much experience with programming graphics in the past, so I felt that I wanted to try something new and experiment with what I could do with graphics. There are ways to fix the flicker issue and input delay issues, but they proved to be a bit more involved than what I had time for. Overall though I think that it was a good program, and I would like to improve it in these ways to make it better, and get more programming experience.

Nathan Boersma: 18 hours

Accomplishments:

For my part of the project, I decided that I wanted to build a memory game. This is something I decided because I enjoy playing games like Simon Says and Bop-It. My original goal was to build one game that required memory. The first game I built was the number memory game which was probably more difficult than the color memory game. Once I was finished with the number memory game, I had many of the pieces I would need to build the color version. An accomplishment was being able to create both games. Another accomplishment was incorporating a timer that gives the user 3 seconds to respond before the player loses. I was also able to create levels that made it more difficult the further the user got.

The biggest challenge that I faced was trying to work with the Winforms. This become so frustrating trying to figure out how to create random numbers, included a timer, and transferring my working code over to be able to work with Winforms that I decided to keep my games as text-based games. Another challenge I faced was working with GitHub. On two occasions, I had to fix the properties and configuration of my Visual Studios project so that I could get the code to compile. This took quite a bit of time the first time I faced this challenge because some of the things I came across on the internet weren't helpful or made the problem worse.

In the future, I would like to improve on my project by converting it over to use Winforms and buttons. I tried briefly to do that during the project timeline but ran out of time and decided that having a text-based game was better than not having a working game at all. My ideal color memory game would include 4 buttons that the user would click to identify the colors that flashed in the proper order. This would be similar to the game Simon. I would also like to add a card matching game that used images and when the user clicked the image it would flip it and then the player would select another card and if it matched, they would receive points and the 2 cards would disappear until there were no cards left.

Michael Harmon: 10 Hours

Accomplishments:

I really learned a lot about Windows Forms and how they interact with C++. I was able to use images and incorporate them into my Find The Ace Game. I really enjoyed learning new things about Visual Studio as well and figuring out how things work and solving problems I was having. I struggled with locating the section of code that allowed global variables. I wanted my randomization to be a global variable but was unable to do so. I attempted to use a function and return the random number to my game but was getting errors and could not figure out what was

going on with it. My games were whole however I am missing randomization of my cards for Find The Ace and a global randomization for my Guess My Number game. I spent about 10 hours on the two games and a majority of my time was spent doing trial and error and figuring out the best way to handle things. I was happy about being able to create two games. I was frustrated with how the Windows Forms works at times.This was a good experience for me and allowed me to deep dive into the windows forms and learn a lot of new things.

Sage Hackbarth 16 hours

Accomplishments:

I got most of the game done, but hit an immovable wall. The look of it was pretty good and I thought it would have worked perfectly fine like it is. Being able to remake the entire program three times without it royally messing up felt a little good too. I figured out, with help, that arrays do not work in Visual Studios, where to put all of your work in a C++ Winform, and how to create a random number generator that randomises many aspects.

What I did not think was so great was that Visual Studios takes all your basic knowledge of some programming languages and rips it away from you by having so much overhead stuff. I could not use arrays and somethings just did not work in general. I hit a wall with an error that says “The data necessary to complete this operation is not available yet”. I looked it up and found that it is a common error with no fix. Even recreating my program a couple times did not help, nor commenting out the new code cinse the rise of that error.

What I would have done to expand the project is to make another game and find out what the problem is with this one so i can finish it. If the project worked, I would create animations for each of the boxes so it looked like they were being scratched off.

Michael Olson: 17 hours

I got very few of the game I wanted complete time was probably one of my largest barriers, having other classes and other assignments to do mixed other smaller things cut down on the amount of time I thought I had to complete this assignment. My next biggest problem was figuring out that you can't make a picturebox array which literally flips my entire structure on its head. Enemies movement, player bullet movement, enemy bullet movement, collision, and player lives were all going to be using arrays causing me to have to completely rethink how I was going to do this project after already spending 10 hours trying to get it setup and make it work.

I did end up allowing player movement, shooting and correct enemy movement which is a very good start on a project like this, but the game currently lacks collision (you can’t kill anything), a scoring system, alien bullets, the ability to have multiple rounds going up the screen, and a game over screen. Proof of concept is there, but without major restructuring of the way the bullets and aliens are, I do not think the current structure will be able to complete this project the way that I would want.