Project Report 2

Hannah Congleton & Ren Young

**Part 1-How to Get our Code to Run**

We are submitting two files because for some reason when our code is broken out into sections it will not finish running when the user clicks evaluate notebook. For a quick run version of the code, use the file that has the words “Quick Run” in the title. For this file you can just evaluate the notebook and access all the menus through the main screen. If you want to use the organized code each function must be run individually before the whole thing can be run. For either of these, after all necessary code has been run it will bring up a menu. Each of the buttons (except Solver) will bring the user to the respective section of the project. The Sudoku Games button brings up the list of sudoku games. Those buttons bring up their respective puzzle (except Mathdoku). The Solver button does not work yet. The Create Your Own button brings up the puzzle creation menu. The Instructions button brings up the menu with the buttons that correspond to each puzzle. The 9x9 and 12x12 buttons bring up the instructions for that puzzle.

**Part 2-Proposal of Approved Project**

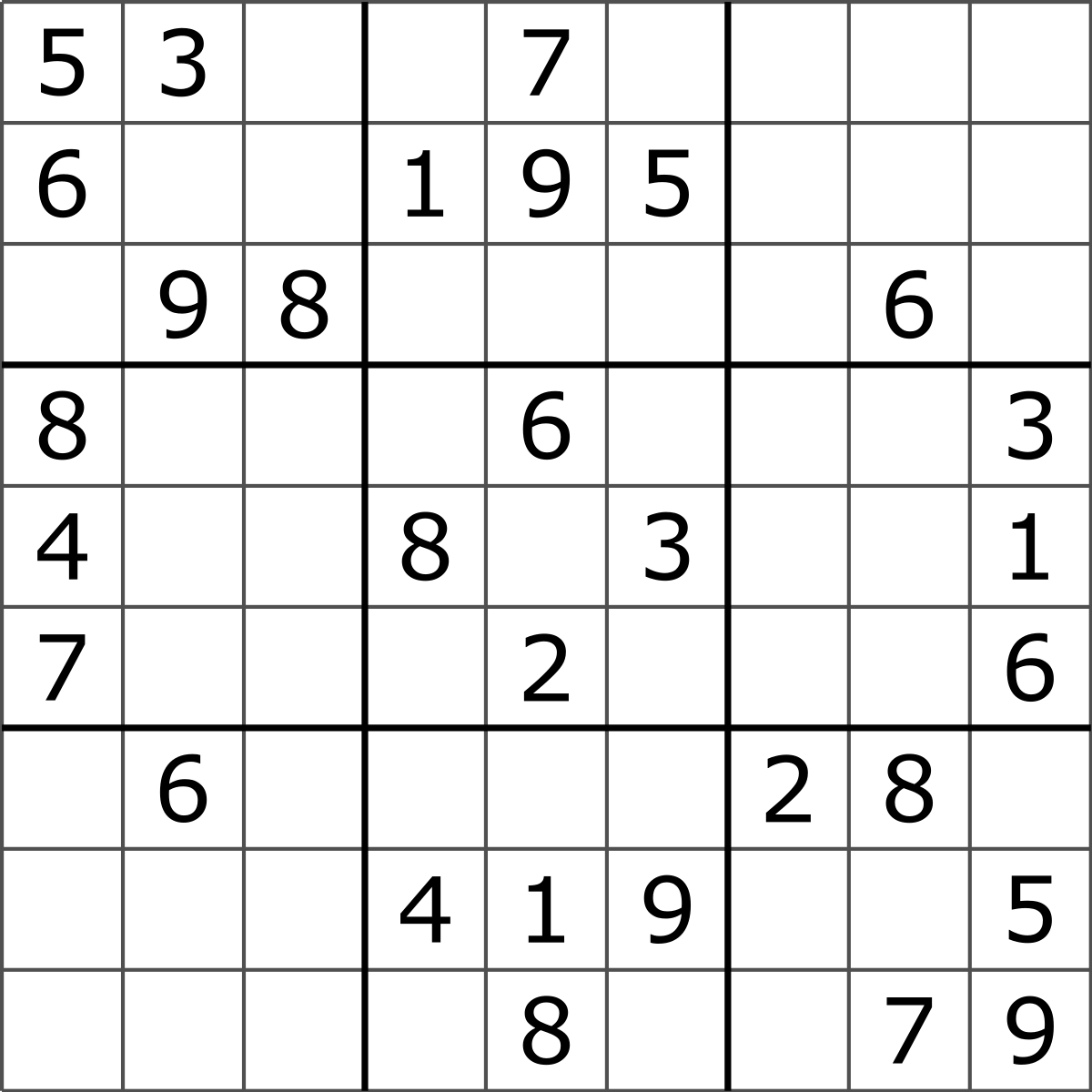
**Description**

Our project involves multiple parts, including interactive sudoku games (9x9 and variations), a sudoku solver, a sudoku generator, and a way for the user to create their own puzzles.

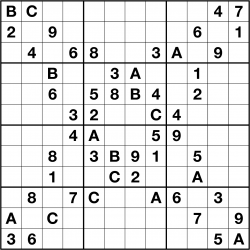
Interactive Sudoku Games

There will be multiple versions of the sudoku game, including the classic 9x9, 12x12, 16x16, diagonal, and mathdoku.

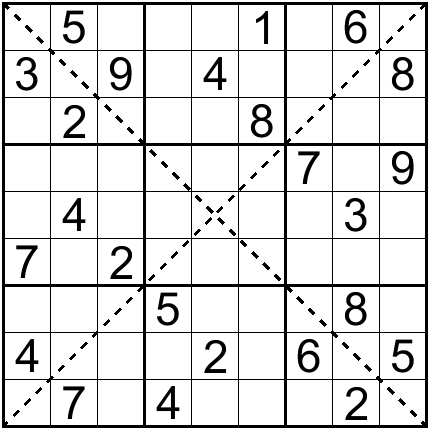
The 9x9 board is the classic game where every row, column, and square must contain the numbers 1-9.



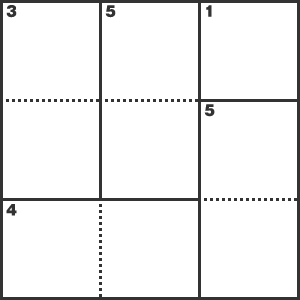
The 12x12 and 16x16 are just larger versions of the 9x9 board, with letters to replace the double digit numbers. The picture below is showing the 12x12, whereas the 16x16 would be bigger.



The diagonal version is the normal board, but the diagonal of the puzzle also has to have the numbers 1-9.



Mathdoku is a version of the game where there are smaller boxes within the board with small numbers in the corners. The numbers within the smaller boxes must add up to the small number. The picture below is just being used to show the small boxes, as the actual board will be a 9x9 sudoku board.



Sudoku Solver

The sudoku solver will include a spot where the user can input an already created 9x9 sudoku puzzle and the program will solve it.

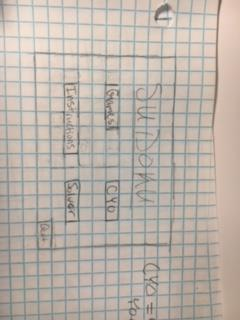
Sudoku Generator

The sudoku generator will generate random sudoku puzzles to be used in the classic 9x9 sudoku puzzle.

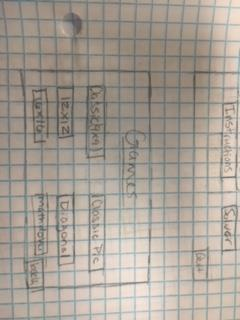
Create-Your-Own

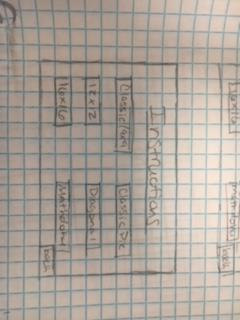
The create-your-own sudoku will be a board where the user can input their own puzzle, and it will check if it is a proper sudoku with only one way of solving it. If it is a proper sudoku puzzle, it will give the user the option to play it.

**What It Will Look Like**

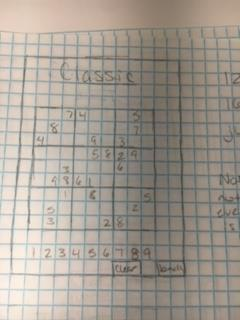


When you run the program the dialog box opens up to the main page where you can choose to play a sudoku game, create-your-own, solver, and instructions, there will also be a reference button and a quit button.

If you click on the games button then it takes you to all of the different versions of sudoku that you can play.



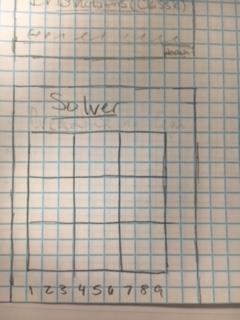
If you click on the instructions it take you to how to play all the different games and how to use the create-your-own and the sudoku solver. There will also be a back button to go to the main page again.



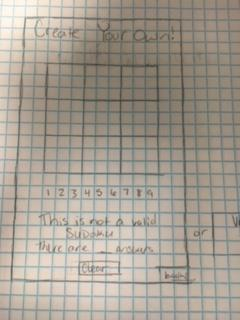
This is roughly what a game of classic 9x9 sudoku will look like where you click on the number you want at the bottom and then you can click on the board and it puts it there. There will also will be a clear button where it will clear all of the numbers that you put in. There will also be a back button where you can go to other parts of the program. Also, the puzzle will already be generated and the number that are chosen cannot be changed.

If you click on one of the instruction it will pop up and there will be a written explanation on what you wanted to learn. Once you are done, you can click the back button to go back.

Right now using code from grid for formatting of the diagonal



For the solver, you click the number on the button and then the square you want and it inputs it in. Once you have all of the numbers of the puzzle then there will be a solve button that you can click and it will solve it for you and show it. Once you are done, you can use a back button.



You also can create your own and input the numbers you want by clicking on the number and then on the spot you want. When you have all the numbers you want you can use the check button to check if is a valid sudoku puzzle or not. If it is you can choose to play it if you want. There is also a back button if you want to go to something else.

**Plan**

Week 10 (Mar 26-Mar 31):

1. Pop up menu with home screen
2. Interactive classic puzzle

Week 11 (Apr 1-Apr 7):

1. 12x12
2. 16x16
3. Beginning of Instructions
4. Begin Solver

Week 12 (Apr 8- Apr 14): First Progress Report

1. Mathdoku
2. Classic pic
3. Finish Solver
4. Sudoku checker
5. Finish Instructions
6. Start generator
7. Diagonal

Week 13 (Apr 15-Apr 21):

1. Finish generator

Week 14 (Apr 22-Apr 28):Second Progress Report

1. Create your own sudoku
2. Figure out what Final Details are

Week 14.5 (Apr 29-May 2):

1. Final Details
2. Project Report

May 4: Final Project Report

Update: We changed the classic pic to Hyperdoku

**Part 3-Schedule of Future Work**

**Plan**

Week 14 (Apr 22-Apr 28):Second Progress Report

1. Start Mathdoku
   1. Finish by April 30
2. Finish Solver
   1. By April 24
3. Sudoku checker
   1. Finish by April 28
4. Finish Instructions
   1. By April 26
5. Start Generator Implementation

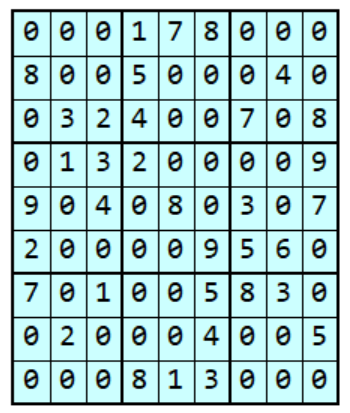
Week 15 (Apr 29-May 4):

1. Create your own sudoku
   1. Finish by May 4
2. Finish Generator
   1. Finish by May 5
3. Final Details
   1. Complete anything that has not been completed
   2. Check that everything is working correctly
4. Project Report
   1. Use the rest of the time for this
   2. Finish report and presentation
   3. Practice presentation

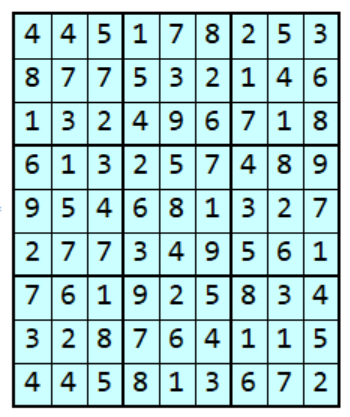
May 9: Final Project Report Due

**Part 4-Explanation of Strategies used**

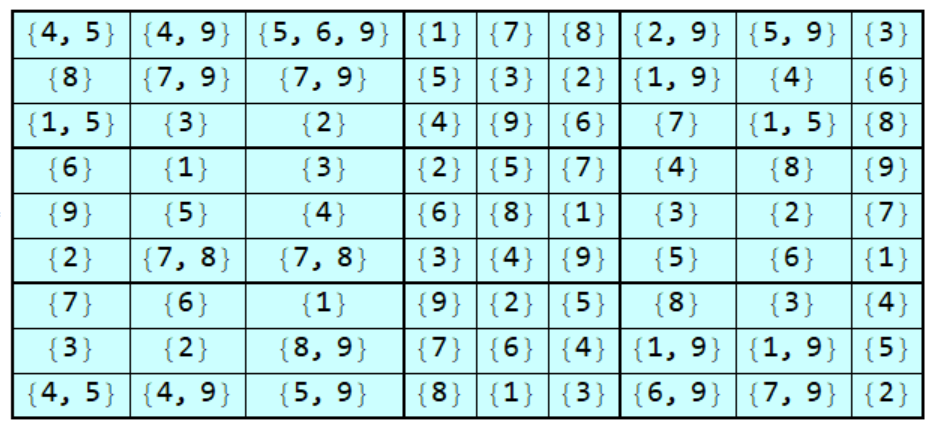
The first part with algorithms is the sudoku solver. It uses multiple functions, including sudokuSolver[board\_], removeDuplicates[row1], and guessAndCheckSolve[canList]. The sudokuSolver function will probably be split into multiple functions. The sudokuSolver starts by making a list of lists of all of the possible candidates for each individual square, such as the puzzle below.



Then, it calls the removeDuplicates function for each row. Next, it puts the columns into rows and calls the removeDuplicates function and puts the new rows back into the columns. After that, it puts the 9x9 boxes into rows and calls the removeDuplicates function. It then puts those back into the candidate list. It repeats these steps until it is unable to remove more candidates or until it is solved. The picture below is the solved version of the puzzle above, using the functions described.



The removeDuplicates function is used many times throughout the solver and provides the base for the solver. This function starts by making a list of all the numbers that are the only candidates for their square. After that, it removes those numbers from the rest of the candidate lists in that row. It also finds the numbers that only have one spot they can be in in the row. In the picture below this can be seen in the second row with the {1,9}, which are the possible candidates for that square, but that is the only square where a 1 could be. This is the same puzzle as above, but shown before the last steps were done.



The guessAndCheckSolve function will be the last part of this process. This function will use systematic guessing (trial and error) to find a solution if the simpler methods do not work. It will start by choosing a candidate from a list of candidates in one square and proceed with solving from there until it either solves it, can’t solve it, or needs more clues. If it can’t solve it, it eliminates that candidate and starts again with another candidate in the same square. If it needs more clues, it chooses a candidate from another square and continues, keeping track of the candidates chosen, and backtracking when needed. This method is part my own strategy and the strategy from a paper and corresponding weblink, which is <http://www.sudokuwiki.org/Sudoku_Creation_and_Grading.pdf> .

The second main part with algorithms will be the sudoku generator. This section will begin by filling nine random squares of the grid with 1-9. Then, it will use the solver to get rid of candidates, and put in some more numbers, repeating this until the board is full. It will then check that the puzzle is plausible. After that, it will remove a certain amount of numbers from the board and check that it is still solvable. This method is part my own strategy and the strategy from a paper, which is <http://www.sudokuwiki.org/Sudoku_Creation_and_Grading.pdf> .

**Part 5-Files or File Structures**

We do not use any files or file structures.

**Part 6-Major Data Structures**

We use a list of dynamic variables for each of the values in each sudoku type. The list is a nested list, with each value being a dynamic variable. The 9x9 list is shown below, but the other lists are of a similar structure.

dynam9List = {{Dynamic[x1], Dynamic[x2], Dynamic[x3], Dynamic[x4],   
 Dynamic[x5], Dynamic[x6], Dynamic[x7], Dynamic[x8],   
 Dynamic[x9]}, {Dynamic[x10], Dynamic[x11], Dynamic[x12],   
 Dynamic[x13], Dynamic[x14], Dynamic[x15], Dynamic[x16],   
 Dynamic[x17], Dynamic[x18]}, {Dynamic[x19], Dynamic[x20],   
 Dynamic[x21], Dynamic[x22], Dynamic[x23], Dynamic[x24],   
 Dynamic[x25], Dynamic[x26], Dynamic[x27]}, {Dynamic[x28],   
 Dynamic[x29], Dynamic[x30], Dynamic[x31], Dynamic[x32],   
 Dynamic[x33], Dynamic[x34], Dynamic[x35],   
 Dynamic[x36]}, {Dynamic[x37], Dynamic[x38], Dynamic[x39],   
 Dynamic[x40], Dynamic[x41], Dynamic[x42], Dynamic[x43],   
 Dynamic[x44], Dynamic[x45]}, {Dynamic[x46], Dynamic[x47],   
 Dynamic[x48], Dynamic[x49], Dynamic[x50], Dynamic[x51],   
 Dynamic[x52], Dynamic[x53], Dynamic[x54]}, {Dynamic[x55],   
 Dynamic[x56], Dynamic[x57], Dynamic[x58], Dynamic[x59],   
 Dynamic[x60], Dynamic[x61], Dynamic[x62],   
 Dynamic[x63]}, {Dynamic[x64], Dynamic[x65], Dynamic[x66],   
 Dynamic[x67], Dynamic[x68], Dynamic[x69], Dynamic[x70],   
 Dynamic[x71], Dynamic[x72]}, {Dynamic[x73], Dynamic[x74],   
 Dynamic[x75], Dynamic[x76], Dynamic[x77], Dynamic[x78],   
 Dynamic[x79], Dynamic[x80], Dynamic[x81]}};

Another major list is the list of possible candidates for every square. The line below is the code that generates the list. This is used in the sudoku solver.

candidateList = Table[k, {i, 1, 9}, {j, 1, 9}, {k, 1, 9}];

**Part 7-Overview**

The mainScreen[] function has the options to play one of the game varieties, Use the Solver, or do the Create-your-own. Can also use the back button to quit the program.

When you click the games button the sudokuGamesMainScreen[] function is ran which pops up the Main screen of the game varieties that shows all of the varieties. Possible varieties are the Classic 9x9, 12x12, 16x16, Diagonal, Hyperdoku, and Mathdoku. You also can go back to the main page with the back button. When you click the Classic (9x9) button it runs the playClassicSudoku[] function which pulls up a 9x9 board where you can click on the blanks and choose what you want to input. Once you are completed “YOU WIN” will pop up and will allow you to play another of the same version or go back to the games main screen and choose another version. There is also a back button so you can go back to the games main screen at any time. If you click on the 12x12 button it runs the play12Sudoku[] function which pulls up a 12x12 that you can play by clicking on the blank spots and type in the number or letter that you think goes there. Once you are completed “YOU WIN” will pop up and will allow you to play another of the same version or go back to the games main screen and choose another version. There is also a back button so you can go back to the games main screen at any time. If you click on the 16x16 button it runs the play16Sudoku[] function which pulls up a 16x16 that you can play by clicking on the blank spots and type in the number or letter that you think goes there. Once you are completed “YOU WIN” will pop up and will allow you to play another of the same version or go back to the games main screen and choose another version. There is also a back button so you can go back to the games main screen at any time. If you click on the Diagonal button it runs the playDSudoku[] function which pulls up a 9x9 with the diagonal boxes being darker than the rest of the board that you can play by clicking on the blank spots and type in the number or letter that you think goes there. Once you are completed “YOU WIN” will pop up and will allow you to play another of the same version or go back to the games main screen and choose another version. There is also a back button so you can go back to the games main screen at any time. If you click on the Hypersudoku button it runs the playHSudoku[] function which pulls up a 9x9 with 4 more boxes that are darker than the rest of the board that you can play by clicking on the blank spots and type in the number or letter that you think goes there. Once you are completed “YOU WIN” will pop up and will allow you to play another of the same version or go back to the games main screen and choose another version. There is also a back button so you can go back to the games main screen at any time. In the future If you click on the Mathdoku button it runs the playMSudoku[] function which pulls up a 9x9 with different shaped boxes with a small number in the upper left corner of the different boxes than the rest of the board that you can play by clicking on the blank spots and type in the number or letter that you think goes there. Once you are completed “YOU WIN” will pop up and will allow you to play another of the same version or go back to the games main screen and choose another version. There is also a back button so you can go back to the games main screen at any time.

When you click on the Instructions button the sudokuInstructionsMainMenu[] function is ran that shows a page of possible instructions that you would want. If you click on the 9x9 button you run the Instructions9[] function that shows the instructions of how to play the 9x9 game. Once you are done reading the instructions you can press the back button and go back to the instructions menu. If you click on the 12x12 button you run the Instructions12[] function that shows the instructions of how to play the 12x12 sudoku puzzle. Once you are done reading the instructions you can press the back button and go back to the instructions menu. In the future if you click on the 16x16 button you run the Instructions16[] function that shows the instructions of how to play the 16x16 game. Once you are done reading the instructions you can press the back button and go back to the instructions menu. In the future if you click on the Diagonal button you run the InstructionsDiag[] function that shows the instructions of how to play the 9x9 game. Once you are done reading the instructions you can press the back button and go back to the instructions menu. In the future if you click on the Hyperdoku button you run the InstructionsHyper[] function that shows the instructions of how to play the Hyperdoku game. Once you are done reading the instructions you can press the back button and go back to the instructions menu. In the future if you click on the Mathdoku button you run the InstructionsMath[] function that shows the instructions of how to play the Mathdoku game. Once you are done reading the instructions you can press the back button and go back to the instructions menu. In the future if you click on the Create Your Own button you run the InstructionCYO[] function that shows the instructions of how to use the Create Your Own. Once you are done reading the instructions you can press the back button and go back to the instructions menu. In the future if you click on the Solver button you run the InstructionsSolver[] function that shows the instructions of how to use the Solver. Once you are done reading the instructions you can press the back button and go back to the instructions menu.

When you click on the CYO button on the main screen you run the playCYO[] function that pulls a blank 9x9 board that you can input the numbers that you want and then in the future you can click the check button and it will tell you if that will work as a sudoku puzzle or not. If it does work you can play the sudoku puzzle you have now created. If it isn’t then you can continue changing till it is a viable sudoku puzzle. Whenever you want you can use the back button to go back to the main screen.

The checkSudoku[board\_] is a function that is used to check if a sudoku puzzle is solved correctly. It returns True or False depending on whether or not the puzzle is correctly solved. It is finished, but has not been implemented.The sudokuSolver[board\_] is the main sudoku solver function. It uses the removeDuplicates function and will eventually use the guessAndCheckSolve function to solve the puzzle. It is not yet completed because the guessAndCheckSolve function is not finished. The removeDuplicates[row1\_] function removes the duplicate numbers from the input row. It is complete and is used in the sudokuSolver function. The guessAndCheckSolve[canList\_] is not complete. When it is complete, it will use the systematic guessing method discussed in Part 4.

**Part 8-Description of What we Have Done/Learned/Found**

-**We have**:

* Created a pop-up menu with buttons for each of the project sections.
* Created a menu (accessed through main menu) for the sudoku puzzle variations with buttons for each of the sudoku variations
* Created an interactive 9x9 sudoku puzzle board
* Created an interactive 12x12 sudoku puzzle board
* Created an interactive 16x16 sudoku puzzle board
* Created an interactive Diagonal sudoku puzzle board
* Created an interactive Hyperdoku sudoku puzzle board
* Started working on the Instructions
* Almost finished the Solver
* Gotten rid of the quotation marks around the words in the Buttons, using ShowStringCharacters->False
* Created the sudoku checker

-**Problems that Need to be Solved**:

* Restrict length of input for the sudoku puzzles
* Change the numbers above 9 to letters in the 12x12 and 16x16 puzzles
* Make the original puzzle numbers bold
* Change the background to a picture instead of a plain color
* Finish the last bit of the solver
* Creating the mathdoku puzzle
* Creating the generator
* Finish the create your own puzzle and instructions menu
* Link the sudoku checker to the puzzles

-**Completed Increments**:

* Main pop-up Menu
* Interactive Classic(9x9) Puzzle
* Interactive 12x12 Puzzle
* Interactive 16x16 Puzzle
* Interactive Diagonal Puzzle
* Interactive Hyperdoku Puzzle
* 9x9 Instructions
* 12x12 Instructions
* Part of sudoku solver
* Sudoku Checker

Our current unresolved issues are:

* Restricting the length of input for the sudoku puzzles
  + As long as the user does not enter a number with multiple digits the input box works nicely
* Changing the numbers above 9 to letters in the 12x12 and 16x16 puzzles
  + Currently, the function does not allow for letters to be used instead of 10,11,12,13,14,15, and 16
* Changing the backgrounds of the menus to pictures instead of plain colors
  + The options we have tried have given us errors when trying to change the background picture
* Making the solver work-Almost Finished
  + Until this function is working we are unable to complete the generator and create your own puzzle.

**Part 9-Signed Statements**

Ren Young

Things I did:

* Base of mainScreen function[ ], including:
  + Outline functions to create the set up
  + Creating the base buttons, and linking the buttons to the right functions
* Base of sudokuGamesMainScreen[ ] function, including:
  + Outline functions to create the set up
  + Creating the base buttons , and linking the buttons for each sudoku variation to the right function
* checkSudoku function
* Base of sudokuInstructionsMainMenu[ ]
* Original function to create a board (Stuff we tried that didn’t work).
* Parts of Classic Sudoku
  + sudokuList
  + dynam9List
  + Base of playClassicSudoku[ ] function, including:
    - Creating the list of original numbers and InputFields to be used in the board
    - Outline functions to create the set up
    - Setting up the dividers in the Grid
    - Creating the base back button
* Sudoku Solver
  + All parts

Things I will do:

* Sudoku solver
  + finish
* Sudoku generator
* Implement sudoku checker

Online Signature:

Ren Young

On paper Signature: x

Hannah Congleton

Things I did:

* Main Screen
  + Formatted the rows: size, Color, spacing
  + Color Scheme and formatted the whole page
* Main Game Screen
  + Did the formatting and graphics of the page
  + Formatted the button
* Games
  + Graphics and formatting
  + Used the 9x9 to make the 12x12 and 16x16
  + Made the variables for the banks in the 12x12 and 16x16
  + Made the formatting for Diagonal and Hyperdoku
* Started the instructions

Mostly I edited and changed the base code and did the graphics.

Things I will Do:

* Graphics of the stuff we continue to do
* Even more formatting
* Create-your-own
* finish the instructions
* Mathdoku

Online Signature: Hannah Congleton

On Paper Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_