California University of Pennsylvania

Department of Mathematics, Computer Science,
Information Systems, and Engineering Technology

CSC 492: Senior Project II, Spring 2021

Virtualization of a Turing Machine:
A Turing Machine Simulator

User Manual

Brett George

Ji Sue Lee

James Merenda

Instructor Comments/Evaluation

Table of Contents

Introduction

Project Overview

The goal of the Virtualization of a Turing Machine website is to enhance the teaching methodology of teachers who wish to explain the fundamentals of designing Turing machines. The project may also be used as a general tool for learning and refining current ideas for Turing machine concepts. The main deliverable consists of a website package that must be hosted on a server. The files within include multiple HTML, CSS, and JavaScript files to form a cohesive website for ease of access and use. The website includes three primary interfaces for development of Turing machines. With this deliverable, it will demonstrate the team's understanding of software development and techniques while also demonstrating the team's knowledge of remote development techniques.

Problem

Understanding the basics when it comes to designing a Turing machine may be difficult to convey in standard teaching methodologies. Without a visual or interactive representation of a Turing machine, students or enthusiasts alike may find issues in understanding how transitions may interact with each other.

Solution

The idea of the Virtualization of a Turing Machine website was conceived by a professor teaching Turing machine concepts who wanted an interactive way to teach their students concepts of a Turing machine in a more effective manner. The project will translate transitions compiled and produce animations based on the input provided. A state diagram is generated as well based on the transitions provided to better visualize how each transition is connected.

Intended Audience

The intended audience for this document includes teachers, professors, or students who want to learn or demonstrate the fundamentals of a Turing machine. In addition to these parties, Dr. Weifeng Chen of California University of Pennsylvania, as well as any faculty member from the Department of Mathematics, Computer Science and Information Systems that are involved in the evaluation process of Senior Projects.

Secondary Audience

The secondary audience consists of others who may stumble across this website. These users may be general computer science enthusiasts or others interested in Turing machines. They may use the website for their own personal use or educational purposes.

Motivation

The motivation behind this project stems from a professor's interest in allowing students to further understand how a Turing machine works. Through this tool, the professor may add an example program for the class to view how it works. From there, the professor can make edits to transitions on the fly during another run and demonstrate what happens to the results due to changes made.

Another motivation for this project was accessibility. If a student or anyone interested is inclined to visit the website, they may try writing programs from their local machine or on the internet. On the website, there are multiple ways for the user to view helpful information to guide them on how to write programs for the Turing machine via pop-up windows, the console, or referencing the user manual.

System block diagram

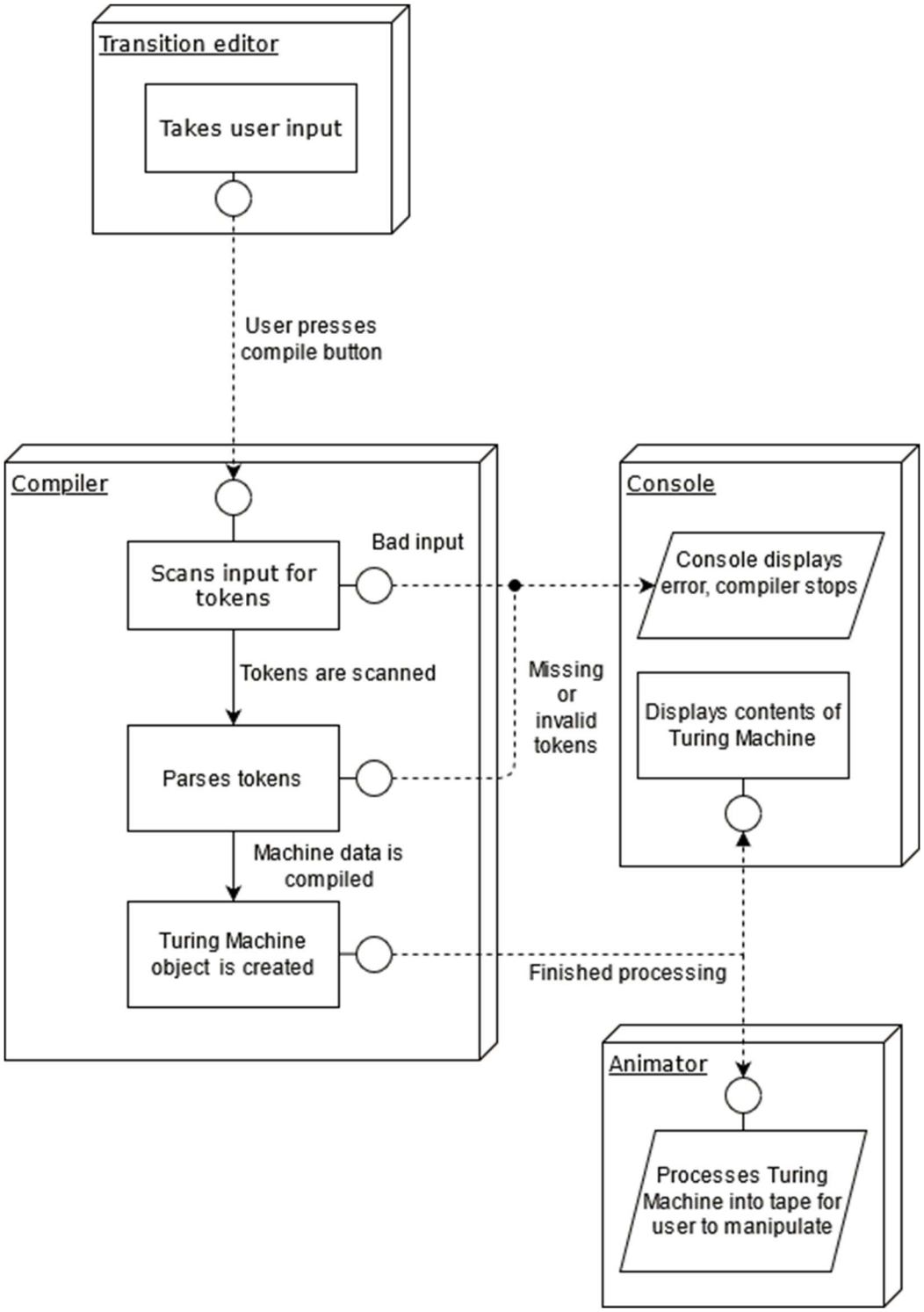

Figure 1: System Block Diagram

Figure 1 describes interactions between the user and the Virtualization of a Turing Machine project. The user will interact with the web interface via typing and mouse clicks. The user may create their own machine transitions by typing them into the *Transition Editor* or selecting an example machine from clicking the *Select Pre-Made Machine* button. The user may also upload and save machines they create by clicking *Upload Machine* and *Save Machine* respectively.

If the user were to click the *Compile* button, the text from the *Transition Editor* is sent to the *Compiler* for scanning, parsing, and object creation. If the scanning and parsing sections of the *Compiler* finds any errors, an appropriate error message is written to the console with a line number, indicating where the error is. If the compilation process was successful, the machine information is written to the console and a machine object is created for future use in the *Transition Animator*.

Based on the machine object generated by the *Compiler*, the *Transition Animator* section will perform reads and writes while moving the tape left and right to perform actions written by the user based on tape input. The animation may be played automatically by clicking the *Play* button and paused by clicking the *Pause* button. When paused, the animation may be reset by clicking the *Reset* button. This will restore the animation to the base state of the machine, ready to replay for the user. If the user decides to edit transitions midway through the animation, they may change transitions in the *Transition Editor* then click the *Edit Machine* button to confirm the changes. If the edit is successfully compiled, then the animation will perform the transitions as specified by the user. If the user cannot see the full output from the tape, they may use the *Move Right* and *Move Left* buttons to move the tape in the specified direction.

Project implementation details

Difference from Design Document

In the Design Document, the team prepared objectives to follow throughout the development of the final product. Throughout the document discusses the team's implementations of the Virtualization of a Turing Machine of the user interface, the compilation process, the animation output, and supplemental error messages from compilation.

There was also the idea for implementing state diagram generation from compiled machines. This idea was explored, but later dismissed in favor of developing necessary features over continuing exploring the added feature. The overall premise seemed easy to implement on the surface but led to more complexity in how to flesh out the details as development progressed.

Also in the user interface, the ability to toggle playback was separated into two buttons, *Play* and *Pause* respectively. The feature to play and pause the tape animation was planned to be merged into one button. This was never fleshed out but works as two separate entities.

The option to remove user input from tape was merged into the compilation process instead of a separate entity. The feature was developed initially, but after discussions with the client, they preferred to have the input compiled with the transitions rather than a separate field. The ability to toggle the visibility of this area was also phased out due to this change.

Most of the error messages from the console were also consolidated into one module rather than three separate categories. Some of the team members felt having one file to control

all warnings and error messages in one contained file would be easier to maintain.

VIRTUALIZATION OF A TURING MACHINE

Project Start: Mon, 25-Jan-2021
Display Week: 10

TASK	ASSIGNED TO	PROGRESS	START	END
Phase 0: Preparation				
Task 0.1: Create Github Repository	James Merenda	100%	25-Jan-21	25-Jan-21
Task 0.2: Create Basic Guide to Use Git/Github	James Merenda	100%	25-Jan-21	26-Jan-21
Task 0.3: Create and Sync Github Page for Project	James Merenda	100%	26-Jan-21	26-Jan-21
Task 0.4: Create Gantt Chart	James Merenda	100%	27-Jan-21	31-Jan-21
Phase 1: Coding		100%		
Task 1.1: Design UI		100%		
Task 1.11: Write HTML Structure for Website	James Merenda	100%	1-Feb-21	12-Feb-21
Task 1.111: Editor Field	Brett George and Jisue Lee	100%	1-Feb-21	12-Feb-21
Task 1.112: Console Field	Brett George and Jisue Lee	100%	1-Feb-21	12-Feb-21
Task 1.113: Tape Field	Brett George and James Merenda	100%	1-Feb-21	12-Feb-21
Task 1.12: Write CSS Styles For HTML Structure	Brett George and James Merenda	100%	1-Feb-21	12-Feb-21
Task 1.2: Create Compiler		100%		
Task 1.21: Write Syntax Parser	Brett George	100%	13-Feb-21	7-Mar-21
Task 1.22: Write Transition Animation Instruction Routine	James Merenda	100%	13-Feb-21	7-Mar-21
Task 1.23 Create Common Error/Warning Compilation Messages	Jisue Lee	100%	13-Feb-21	7-Mar-21
Task 1.3: Create Transition Animations		100%		
Task 1.31: Write Step Left/Right Animation	James Merenda	100%	15-Feb-21	7-Mar-21
Task 1.4: Write User Interaction Area		100%		
Task 1.41: Transition Speed Controller	James Merenda	100%	15-Feb-21	7-Mar-21
Task 1.42: Play Transitions Control	Jisue Lee	100%	15-Feb-21	7-Mar-21
Task 1.43: Step Transition Control	Jisue Lee	100%	15-Feb-21	7-Mar-21
Task 1.44: Reset Transition Control	James Merenda	100%	15-Feb-21	7-Mar-21
Phase 2: Debugging		100%		
Task 2.1: Debug UI	James Merenda	100%	25-Jan-21	24-Apr-21
Task 2.2: Debug Compiler	Brett George	100%	25-Jan-21	24-Apr-21
Task 2.3: Debug Transition Animations	James Merenda	100%	25-Jan-21	24-Apr-21
Task 2.4: Debug Compilation Messages	Jisue Lee	100%	25-Jan-21	24-Apr-21
Task 2.5: Create Sample Programs		100%		
Task 2.51: Program 1	Brett George	100%	25-Mar-21	1-Apr-21
Task 2.52: Program 2	Jisue Lee	100%	25-Mar-21	1-Apr-21
Task 2.53: Program 3	James Merenda	100%	25-Mar-21	1-Apr-21
Task 2.54: More as necessary	James Merenda	100%	-	-
Task 2.6: Debugging Sample Programs	All	100%	20-Mar-21	1-Apr-21
Phase 3: Create User Manual		100%		
Task 3.1: Write Overview and Application Section	James Merenda	100%	4-Apr-21	24-Apr-21
Task 3.2: Write Community or Social Implications Section	James Merenda	100%	4-Apr-21	24-Apr-21
Task 3.3: Create System Block Diagram Section	Jisue Lee	100%	4-Apr-21	24-Apr-21
Task 3.4: Write Use of Software Engineering Principles Section	Jisue Lee	100%	4-Apr-21	24-Apr-21
Task 3.5: Write User's Manual Section	Brett George	100%	4-Apr-21	24-Apr-21
Task 3.6: Write References, Table of Index/Glossary, Appendices	All	100%	4-Apr-21	24-Apr-21

Figure 2: Gantt Chart

A portion of the Gantt chart in Figure 2 represents the rough outline of job designations distributed throughout team members. Each member gave an estimated amount of time per individual assignment. Throughout the development process, some jobs were expanded while others were deemed unnecessary and removed from the chart. Overall, the team managed to stay

on schedule. There were times where certain features needed more time to develop and the chart was updated accordingly.

A further detailed spreadsheet may be found in the following web address: https://docs.google.com/spreadsheets/d/1EQWwdWq2BfARVs8DkjIhUrHznQLNYlUAlYkkYqjR00c/edit?usp=sharing

Challenges During Implementation

For all three members, it was our first time developing a web application, so the team had to take time to get a handle on its intricacies. JavaScript was easier to understand as a traditional programming language, but it does feature new ways to use functions team members never seen before, like function expressions. HTML and CSS were slightly more difficult as they are scripting languages and not programming languages.

Due to time constraints, some features had to be cut or trimmed back. Most notably, the state diagram feature was cut completely since it took too much time to implement in a basic state, when the time could be better used to improve already implemented features.

Another large challenge the team faced was current world events. Due to the pandemic, the team struggled with a lack of motivation at the start, which led to less progress than expected.

Despite these setbacks, the team accomplished their original goal in mind. Using the fundamentals of software development and the aid of documents from previous phases, the Virtualization of a Turing Machine project was completed.

Use of Software Engineering Principles

The team followed guidance provided by and discussed in Senior Project I (CSC 490) from the Fall semester of 2020 and Senior Project II (CSC 492) from the Spring semester of 2021 to create the Virtualization of a Turing Machine project. Each phase of the project was broken down into requirements, specification, design, and implementation phases. Further breakdowns of the necessity of each phase is elaborated in the following text.

The requirements phase was accomplished early on in CSC 490 which the team created a document to reference an overview of the project. This document entails the process of developing and documenting each phase of the entire project. For the most part, aspects of the project were maintained and unchanged throughout the development cycle from the original goal.

The specification phase was completed midway through the semester of CSC 490. The document was essential to elaborate on each process and its function in the project as a whole. The resulting document produced in this phase provides information regarding cohesion between modules, processes, and functionality in the project. Charts and figures were included throughout the document to further illustrate each task.

The design phase finished off at the end of the CSC 490 semester. In this phase, a document was created, detailing a concise instruction set of how the project would be developed in the following semester. This roadmap helped the Virtualization of a Turing Machine team to maintain the project's purpose and design philosophy.

Finally, the implementation phase was employed in the CSC 495. This phase included the bulk of development time to create the project. The project workload was split between team members and worked on accordingly throughout the semester until completion. The workload for each team member also included rigorous testing throughout the semester for quality

assurance as development went on. With the combined skills of the project team and the project documents from the previous semester, the team was able to achieve their goal and produce a final product.

User Manual

Syntax to Define a Machine

For all inputs, alphabetic characters can be entered as uppercase or lowercase, but will be printed as lowercase. Any information in parentheses below defines user input. Unless noted, a valid character consists of any alphanumeric characters, symbol characters (ex: !, @, %) and whitespace characters (space or tab).

Input

input = “(input name)”;

Defines the string the machine will perform operations on. The input name can consist of any combination of characters but requires at least one character.

Blank

blank = “(blank character)”;

Defines the character to be used as the blank character. It can be any singular character, but strings consisting of more than one character will be rejected.

Start

start = (start state);

Defines the name of the start state. Can be any combination of characters and length excluding whitespace characters, if there is at least one character. It does not need to be surrounded by quotation marks like the input or blank.

Accept

accept = (accept state);

Defines the name of the accept state. Uses the same requirements as the start state.

States

-(state name) (character) [(new character), (direction), (new state name)];

Defines a state's name, the character it accepts, the new character, the direction to move and the new state name. Requirements:

State name: A string of at least one non whitespace character. This requirement is also used for the new state name.

Character: Any singular character. Also used for the new character.

Direction: Either a singular l for left or a singular r for right.

To define a singular state with multiple possible accepted characters, a user can use the following syntax:

-(state name) (character 1) [(new character), (direction), (new state name)]

(character 2) [(new character), (direction), (new state name)]

...

(character n) [(new character), (direction), (new state name)];

Accept states can be simply defined by:

-(accept state name);

Comments

Any line that starts with a # will be read as a comment and will not be processed.

Defining a Turing Machine

At minimum, a valid Turing Machine must have the input string, blank character, start state name and accept state name defined, along with the start state and accept state's transitions. Additional state transitions can be defined as needed but are not required like the start and accept state's transitions.

Site Usage

Layout Buttons

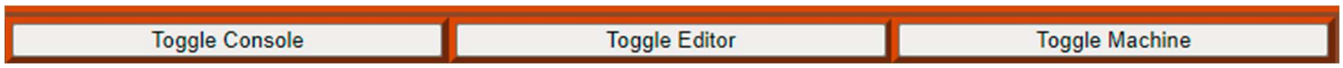

Figure 3: Layout Buttons

Toggle Console - Shows or hides the console at the bottom of the screen

Toggle Editor - Shows or hides the editor at the left side of the screen

Toggle Machine - Shows or hides the machine at the right side of the screen

Transition Editor

Figure 4: Transition Editor

Compile Machine –

Compile's machines based on given input.

Save Machine –

Saves the input as a text file. The filename can be defined by the user and will be saved to the user's destination of choice on their computer.

Select Pre-Made Machine –

Displays a dropdown menu where a premade machine can be selected. See *Example Programs* for further details.

Upload Machine –

Allows a user to upload custom machines from their files to the Transition Editor.

Machine Animator

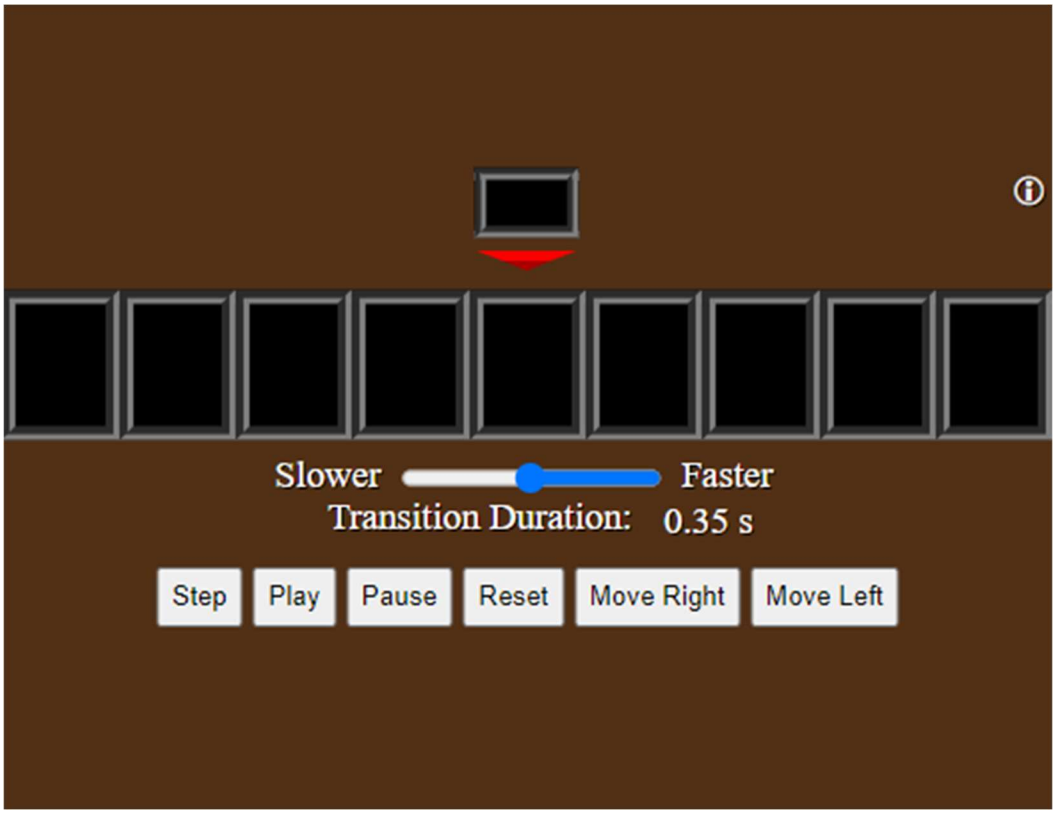

Figure 5: Transition Animator

Step –

Processes one character in the input string.

Play –

Steps through the entire string until the entire string is processed or it reaches the end of the tape.

Pause –

Halt's animation playback.

Reset –

Resets the tape to the start of the input string.

Move Right –

Move right on the tape without processing the character.

Move Left –

Move left on the tape without processing the character.

Note: In Firefox, the cells extend out of the boundaries of the machine. This is purely visual and does not affect operation.

Example programs

Language Acceptors

Palindrome

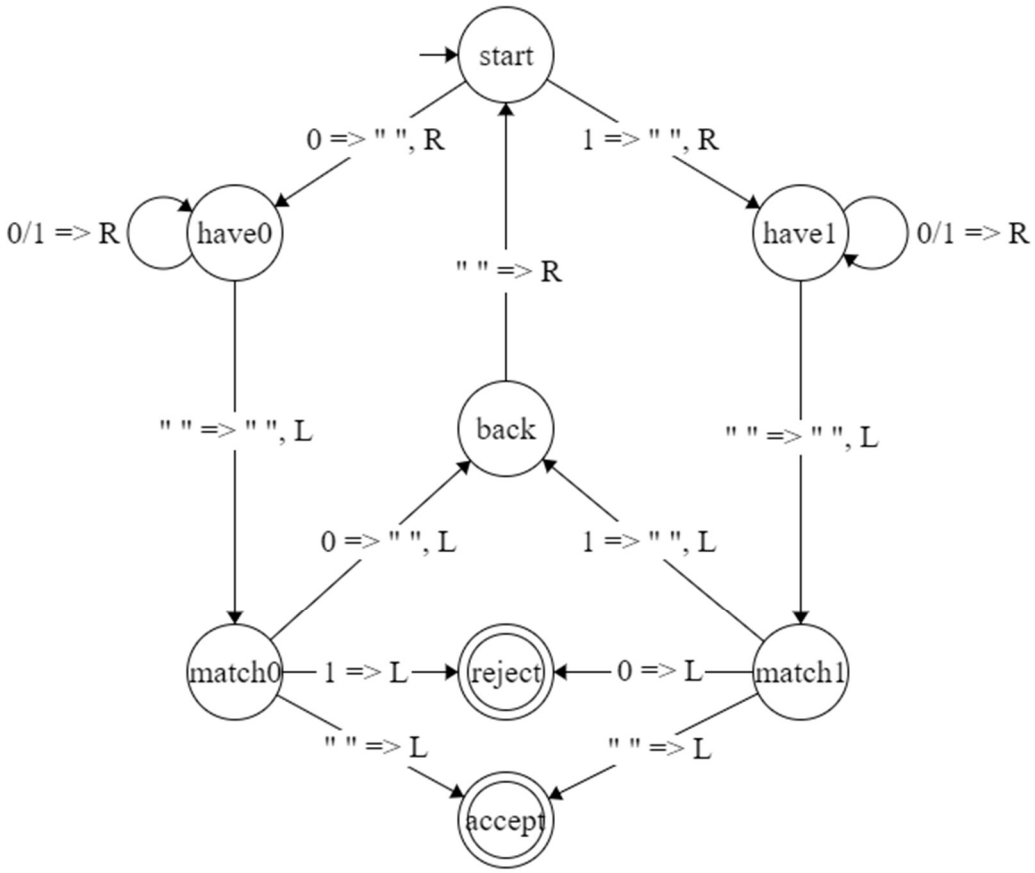

Figure 6: Palindrome State Diagram

Example Inputs: " ", 0, 1, 101, 010, 1001.

Expected Output: None.

Machine Description: Accepts palindromes of binary numbers.

Binary Pong

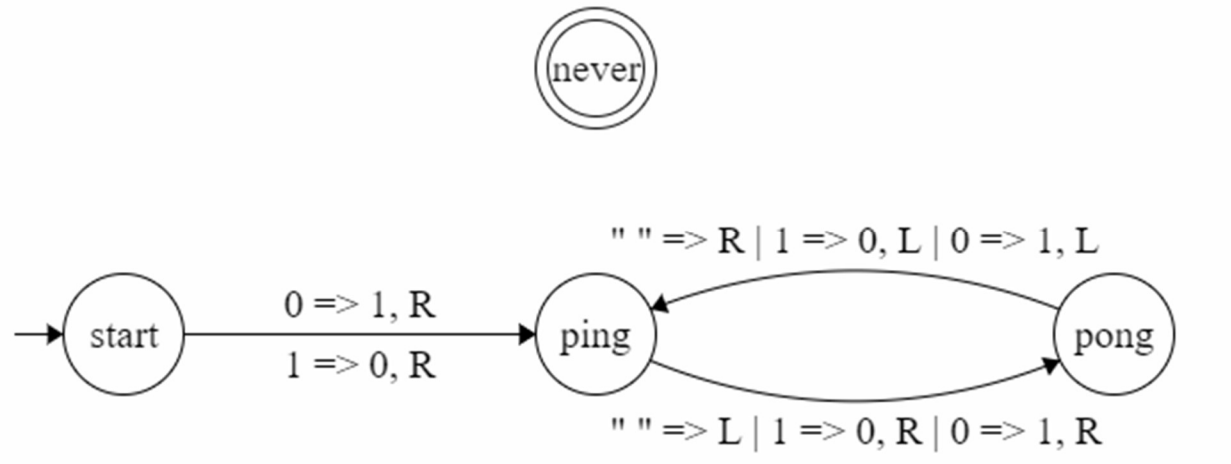

Figure 7: Binary Pong State Diagram

Example Inputs: None.

Expected Output: The read head moves left to right from each side.

Machine Description: Simulates a game of table tennis between two combatants.

The Classic

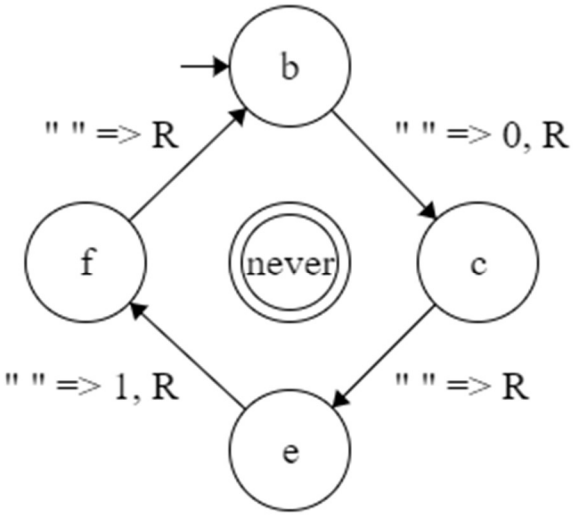

Figure 8: The Classic State Diagram

Example Inputs: None.

Expected Output: An endless string of "0 1 0 1…".

Machine Description: This is the first machine proposed by Alan Turing in his 1936 paper: "On Computable Numbers, with an Application to the Entscheidungs problem".

*wcw, where w is an element of {a,b}**

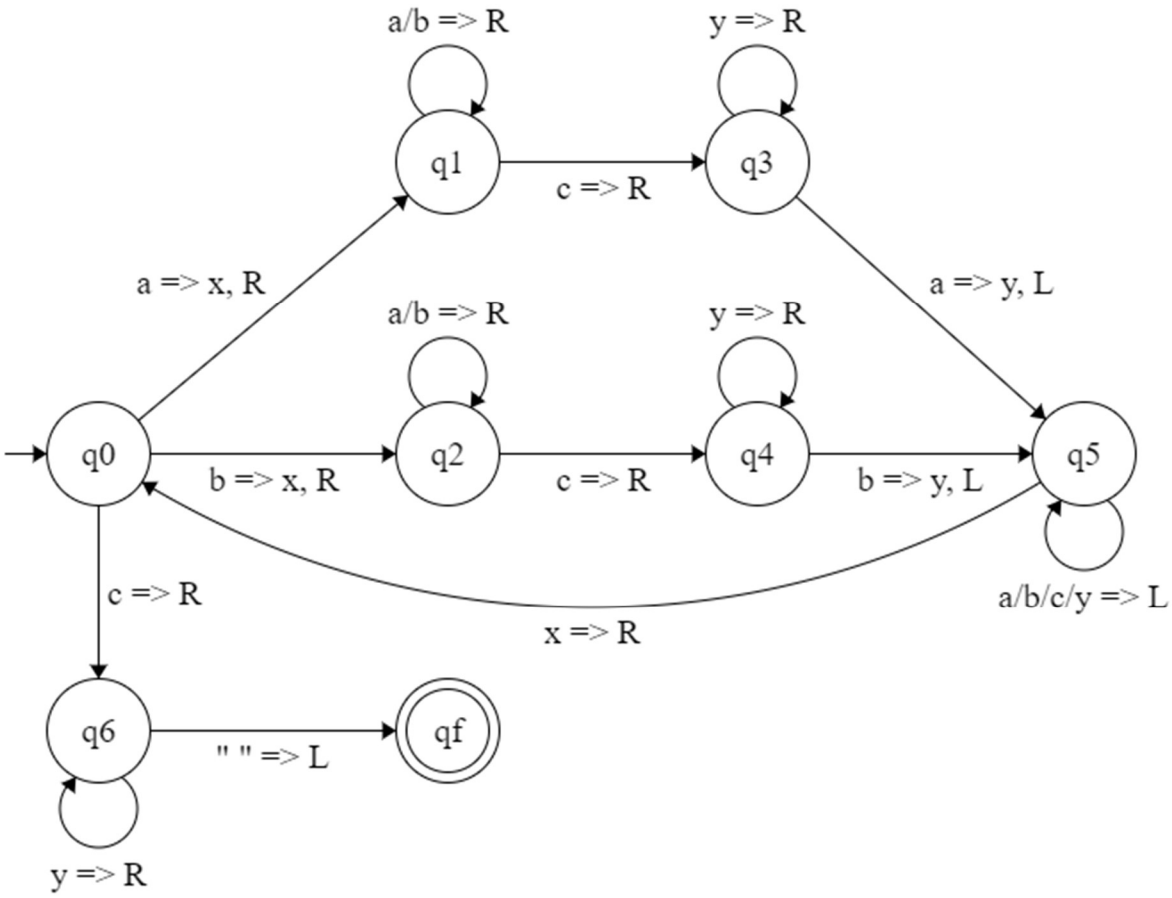

Figure 9: wcw, where w is an element of {a,b}* State Diagram

Example Inputs: c, aca, bcb, abcab, bacba, or (a|b)*c(a|b)*.

Expected Output: $y^n c y^n$ where n >= 0.

Machine Description: Accepts (a|b)*c(a|b)* strings.

Language Transducers

Binary Increment

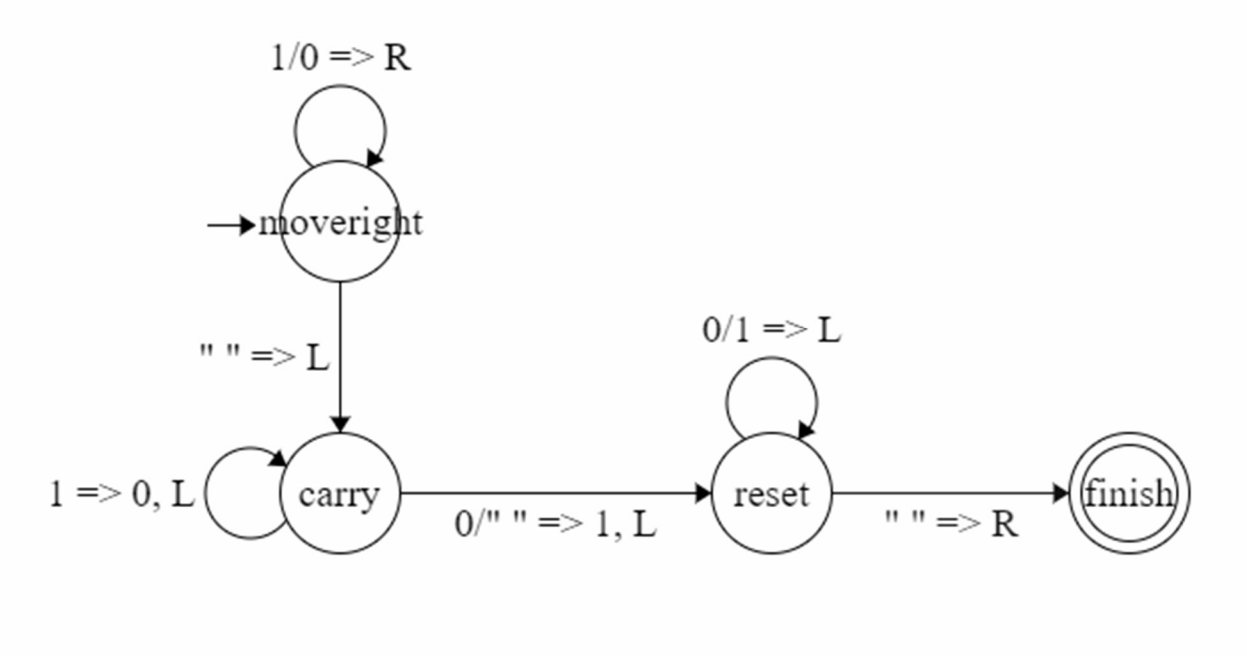

Figure 10: Binary Increment State Diagram

Example Inputs: 0, 1, 01, 10, or $(0|1)^+$.

Expected Output: The binary equivalent of the input plus 1.

Machine Description: Accepts a binary number and returns the binary value incremented by 1.

Binary Addition

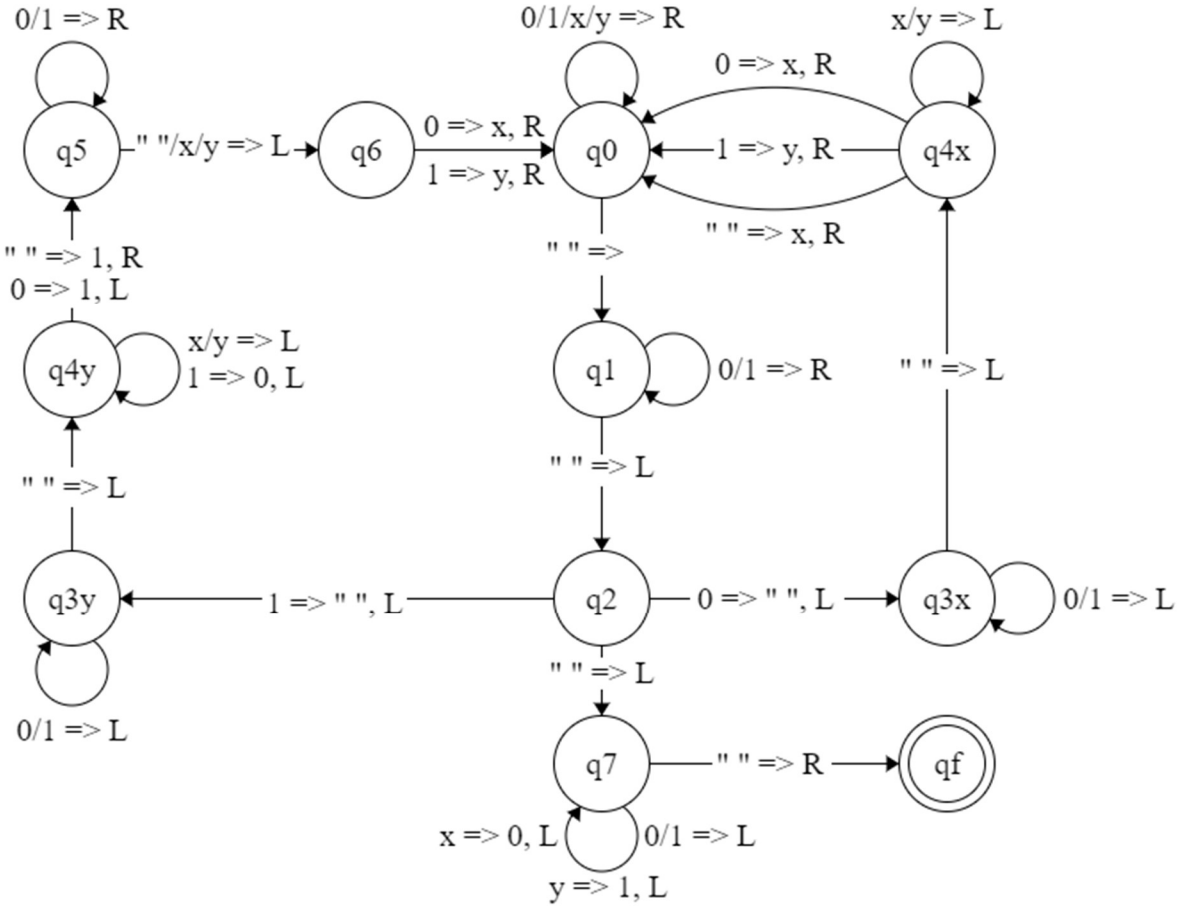

Figure 11: Binary Addition State Diagram

Example Inputs: "101 11", "11 1", or "(0|1)* (0|1)*".

Expected Output: The sum of two input binary numbers.

Machine Description: Accepts two binary numbers separated by a space and sums them together.

Binary Complement

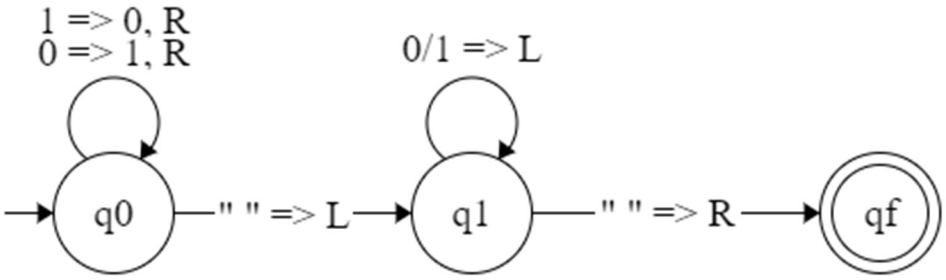

Figure 12: Binary Complement State Diagram

Example Inputs: 1010, 111, or $(1|0)^{+}$.

Expected Output: The complement of the binary number.

Machine Description: Converts a binary number into its complement.

x-y, where x is greater than y

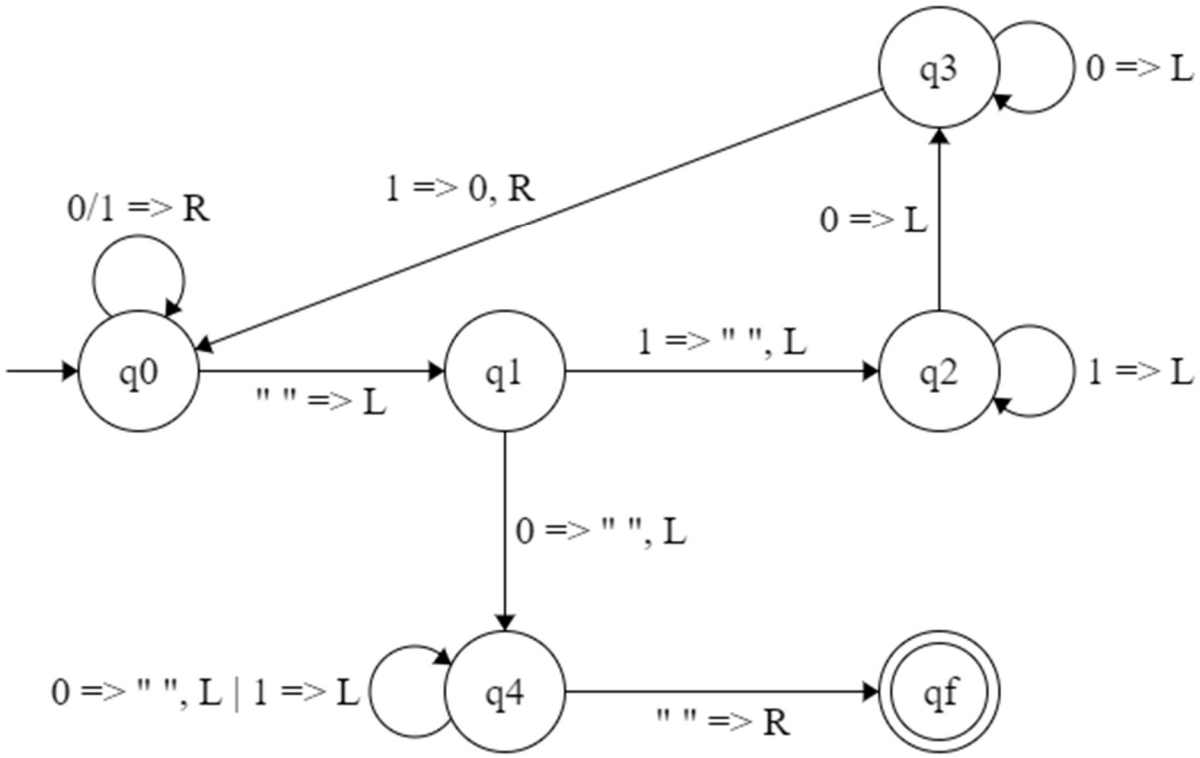

Figure 13: x-y, where x is greater than y State Diagram

Example Inputs: 1101, 111101, or 1^x01^y where x>y.

Expected Output: The difference of x - y in 1s.

Machine Description: This takes two unary numbers separated by a zero and outputs the difference of them.

Installation Instructions

In the provided zip archive, the following items will be provided:

- A *www* folder containing the website assets for the project.

The following instructions are based on whether the user wants to install this project's content to a web server or run the project locally on their machine.

Installing on a web server/website:

To install this project on a website, simply take the entire *www* folder or the contents of that directory in the zip archive and move it anywhere into a web server or website's directories. After moving the project's assets to the desired directory, the project may be accessed by launching the *index.html* file in a modern web browser.

- Note that the contents of the *www* folder need to be within the same directory or else errors may occur.

Installing on a local machine:

To install this project on a local machine, like the previous instruction - take the entire *www* folder or the contents of that directory in the and move it to a local machine directory. After moving the project's assets to the desired directory, the project needs a web server environment to function properly. The following instructions explain a simple method in acquiring the environment to facilitate the project. There are other ways to garner the same result in other methods, but this will work in most cases.

Setting up a local web server environment:

Windows/Mac/Linux:

1. Install Python 3.x
2. Open Command Prompt
3. Navigate to the directory where the *index.html* file is in the project assets
4. Enter the following command:
    a. *python -m http.server*
5. Open a modern web browser of choice
6. Enter the following address in the address bar and hit enter:
    a. *localhost:8000*

After completing the following steps, the project will be ready to use. For more information on how to complete this process, visit the following website for further guidance:

https://developer.mozilla.org/en-US/docs/Learn/Common_questions/set_up_a_local_testing_server

Creating Additional Example Programs:

To append additional example programs in the Select Pre-Made Machine dropdown menu, the following files are edited:

- *www/index.html*
- *www/scripts/manipulatePrograms.js*

In the *index.html* file, navigate to the section called *selection_list.* In this unordered list, is the selection of pre-made machine examples enclosed in <li> tags and categorized with <lh> tags respectively. By clicking one of these <li> tags, it will load the machine relevant to the name in the *Transition Editor*.

To add another option to this list, choose an appropriate name for this machine. Using this name, enclose this name in <li> tags like the rest of the list. This new statement must be inside of the <ul> tags or it will not render properly. From here save the file and open the next file, *manipulatePrograms.js*.

In this file contains a multidimensional array. Each element in this array is a pair of strings. The first element is the name of the program and the second element is the transition code for the *Transition Editor*. **The first name element of the array must contain the same name as the one from the <li> tags.** The second transition code element is enclosed in ` symbols to save formatting from the file to the *Transition Editor* field on the website. Save the file and click the *Select Pre-Made Machine* button

Consider the following example:

```
96                  <ul id='selection_list' style='display: none;'>
97                      <lh>Language Acceptor</lh>
98                      <li id='selection_option'>Palindrome</li>
99                      <li id='selection_option'>Binary Pong</li>
100                     <li id='selection_option'>The Classic</li>
101                     <li id='selection_option'>wcw, where w is an element of {a,b}*</li>
102                     <lh>Language Transducer</lh>
103                     <li id='selection_option'>Binary Increment</li>
104                     <li id='selection_option'>Binary Addition</li>
105                     <li id='selection_option'>Binary Complement</li>
106                     <li id='selection_option'>x-y, where x is greater than y</li>
107                 </ul>
```

Figure 14: index.html Example Programs Location

Above in Figure 14 is a screenshot of the Select Pre-Made Machine dropdown HTML markup in *index.html*. Let's say a user wants to add a program named "My Machine" to the list. The user adds the appropriate name to the HTML markup as shown below in Figure 15:

```
96                  <ul id='selection_list' style='display: none;'>
97                      <lh>Language Acceptor</lh>
98                      <li id='selection_option'>Palindrome</li>
99                      <li id='selection_option'>Binary Pong</li>
100                     <li id='selection_option'>The Classic</li>
101                     <li id='selection_option'>wcw, where w is an element of {a,b}*</li>
102                     <lh>Language Transducer</lh>
103                     <li id='selection_option'>Binary Increment</li>
104                     <li id='selection_option'>Binary Addition</li>
105                     <li id='selection_option'>Binary Complement</li>
106                     <li id='selection_option'>x-y, where x is greater than y</li>
107                     <li id='selection_option'>My Machine</li>
108                 </ul>
```

Figure 15: User Input into Example Programs

The user saves this file and moves on to open *manipulatePrograms.js*. From here, the user will navigate to the array which holds all the example programs as shown below in Figure 16.

```
2   var programs = [
3   [
4   'Binary Increment',

5
6   `# Binary Increment - takes a binary number and increments
7   #                     it by 1.
8   # Input - a single binary number without spaces.
9   # Output - the sum of the binary number plus 1.
10
11  input = "1011";
12  blank = " ";
13  start = moveright;
14  accept = finish;
15
16  # move to the end of the binary number.
17  -moveright      1   [1, r, moveright]
18              0   [0, r, moveright]
19              " " [" ", l, carry];
20
21  # increment rightmost digit and update following digits.
22  -carry      1   [0, l, carry]
23          0   [1, l, reset]
24          " " [1, l, reset];
25
26  # set read head to first item of result.
27  -reset      " " [" ", r, finish]
28          0   [0, l, reset]
29          1   [1, l, reset];
30
31  -finish;`
32  ],
```

Main Array Container

Example Program Name

Example Program Container

Example Program Transition Code

Figure 16: manipulatePrograms.js Layout

Using the format above from Figure 16, the user may add their custom example to the website.

References

Finite State Machines and Context Free Grammars:

Chen, Weifang. “Theory of Languages” California University of Pennsylvania, Eberly, Fall 2020. Lecture.

HTML, CSS, and JavaScript references:

W3Schools Online Web Tutorials. (n.d.). W3Schools Online Web Tutorials. Retrieved 2021, from https://www.w3schools.com/

Inspiration for Project:

Turing machine visualization. (n.d.). Turing Machine Visualization. Retrieved 2021, from https://turingmachine.io/

Turing machine simulator. (n.d.). Anthony Morphett. Retrieved 2021, from http://morphett.info/turing/

Online Turing Machine Simulator. (n.d.). Online Turing Machine Simulator. Retrieved 2021, from https://turingmachinesimulator.com/

Table of Index / Glossary

Turing Machine -

a mathematical model of computation that defines an abstract machine that manipulates symbols on a strip of tape according to a table of rules.

HTML -

Hypertext Markup Language, a standardized system for tagging text files to achieve font, color, graphic, and hyperlink effects on World Wide Web pages.

CSS -

Cascading Style Sheet, used to format the layout of Web pages.

JavaScript -

an object-oriented computer programming language commonly used to create interactive effects within web browsers.

Compiler -

a program that converts instructions into a machine-code or lower-level form so that they can be read and executed by a computer.

Transition Editor -

the interface the user writes Turing machine transitions in.

Console -

the area where messages from the compiler are displayed.

Appendix: Team Details and Individual Contributions

Brett George -

Brett George was the main force behind creating the compiler for the Virtualization of a Turing machine project. He was also responsible for developing the additional functionality to edit machine transitions "on-the-fly".

Jisue Lee -

Jisue Lee was responsible for creating error messages based on every accountable error that the user may encounter. She also contributed to optimizing JavaScript code throughout the project to save on redundancy and increase clarity.

James Merenda -

James Merenda contributed to the project by creating most of the visual elements on the website including the website structure, stylizing CSS, and implementing the magnetic tape with animations as well. He was also responsible for creating several quality of life features throughout the website like the information buttons that give quick tips on each area of the project.

The Team -

The team overall contributed to this document as well as documentations throughout the project. They also collaborated on debugging features as they were rolled out to the project.

Appendix: Code listing

machine.css

```
input[type="file"] {
    display: none;
}

.options {
    position: absolute;
    top:1%;
    bottom: 95.5%;
    left:1%;
    width: 98%;
    display: flex;
    flex-direction: row;
    gap: .71%;
}

.option_buttons {
    width: 90%;
    text-align: center;
    text-decoration: none;
    display: inline-block;
    outline-color: #b3562e;
    outline-width: 4px;
    outline-style: outset;
}

.option_buttons:hover {
    cursor: pointer;
}

.option_buttons:onpressdown {
    outline-style: inset;
}

.ide {
    display: flex;
    flex-direction: column;
}

.console {
    position: absolute;
    top: 74.5%;
    bottom: 1%;
```

```
    left: 1%;
    right: 1%;
    outline-color: #b3562e;
    outline-width: 4px;
    outline-style: inset;
    background-color: #523015;
    padding-top: 3px;
    padding-left: .5%;
    padding-right: .5%;
    display: flex;
    flex-direction: column;
}

.console__display {
    width: 100%;
    height: 99%;
    padding-top: .1%;
    padding-bottom: .1%;
    resize: none;
}

.editor {
    position: absolute;
    left: 1%;
    top: 6%;
    right: 50.35%;
    bottom: 27%;
    outline-color: #b3562e;
    outline-width: 4px;
    outline-style: inset;
    background-color: #523015;
    display: flex;
    flex-direction: column;
}

.editor__buttons {
    display: flex;
    font-size: 11px;
    padding-top: .6%;
    padding-left: 1%;
    padding-right: 1%;
}

.upload__button {
    width: auto;
    height: 19.7px;
```

```
    padding: 1%;
    padding-top: .8%;
    margin-top: .6px;
    margin-left: .7px;
    background-color: #efefef;
    font-family: sans-serif;
    color: black;
    text-shadow: none;
    outline: .25px solid #767676;
}

#load:hover, #save:hover, #select:hover {
    cursor: pointer;
    background-color: lightgrey;
}
.upload__button:hover {
    cursor: pointer;
    background-color: lightgrey;
    outline: 2px outset #666666;
    outline-offset: -1px;
    box-shadow: 0 0 0 0;
}

.editor__text {
    padding-top: 1%;
    padding-left: 1%;
    padding-bottom: .7%;
    border: 1px thin black;
    height: 96%;
}

.editor__text__line-numbers {
    font-weight: bolder;
    text-align: right;
    overflow:hidden;
    word-wrap: normal;
    width: 7%;
    height: 100%;
    background-color:lightgray;
    border: 1px solid black;
    resize: none;
}

.editor__text__code-area {
    width: 91%;
    height: 100%;
```

```
    resize: none;
    border: 1px solid black;
    word-wrap:none;
    background-color:whitesmoke;
}

.machine {
    position: absolute;
    top: 6%;
    left: 50.35%;
    right: 1%;
    bottom: 27%;
    background-color: #523015;
    outline-color: #b3562e;
    outline-width: 4px;
    outline-style: inset;
}

.state_diagram_view {
    position: flex;
    width: 100%;
    height: 40%;
    background-color: #523015;
}

#state_diagram {
    border: 1px solid black;
    background-color: lightgray;
    /*width: 100%;
    height: 100%;
    border-bottom: 3px dotted #b3562e;*/
}

.tape_view {
    position: flex;
    height: 50%;
}

.machine_buttons {
    padding-top: .1%;
    gap: .3%;
    height: 30px;
}
#selection_list
{
    position: absolute;
```

```
    transition-duration: 0.35s;
    max-height: 50%;
    width: 300px;
    overflow-y: auto;
    left: 190px;
    top: 10px;
}

#selection_list lh
{
    display: block;
    width: 100%;
    padding-top: 5px;
    padding-bottom: 5px;
    border-style: solid;
    border-color: white;
    border-width: 1px;
    background-color: #b3562e;
    text-align: center;
    user-select: none;
}

#selection_list li
{
    float: left;
    list-style: none;
    text-decoration: none;
    display: block;
    width: 100%;
    padding-top: 5px;
    padding-bottom: 5px;
    border-style: solid;
    border-color: white;
    border-width: 1px;
    background-color: black;
    text-align: center;
    user-select: none;
}

#selection_list li:hover
{
    cursor: pointer;
    background-color: grey;
}
```

style.css

```
fieldset{display: table-column;}
* {
   box-sizing: border-box;
   -moz-box-sizing: border-box;
}

body
{
        font-family: DejaVu Sans Mono;
        font-size: 14pt;
        text-shadow: 1px #000000;
        color: #f0eeec;
        background-color: #454545;
}

ul
{
        padding: 0;
}

a:link, a:visited
{
 color: white;
 text-decoration: none;
}

a:hover
{
        text-decoration: underline;
}

.wrapper
{
        position: absolute;
        width: 100%;
        height: 100%;
        top: 0;
        left: 50%;
        transform: translate(-50%,0%);
        -ms-transform: translate(-50%,0%);
}

.main-wrapper
{
```

```
        position: absolute;
        width: 75%;
        min-width: 1080px;
        max-width: 1920px;
        height: 100%;
        max-height: 1080px;
        min-height: 734px;
        top: 0;
        left: 50%;
        transform: translate(-50%,0%);
        -ms-transform: translate(-50%,0%);
}

.main
{
        position: absolute;
        width: 100%;
        height: 80%;
        min-height: 600px;
        max-width: 1920px;
        max-height: 1080px;
        background-color: #784d2c;
        outline-color: #b3562e;
        outline-width: 4px;
        outline-style: outset;
}

.navigation ul
{
        list-style-type: none;
        display: inline;
        position: absolute;
        right: 0px;
        top: 0px;
}

.navigation ul li
{
        display: inline-block;
        padding: 5px;
        padding-left: 15px; padding-right: 15px;
        height: 30px;
        margin-top: 5px; margin-left: 10px;
        text-decoration: none;
        color: white;
        background-color: #784d2c;
```

```
        outline-color: #b3562e;
        outline-width: 4px;
        outline-style: outset;
}

.navigation ul li:hover
{
        text-decoration: underline;
}

.documentation
{
        position: absolute;
        width: 1080px;
        margin-top: 120px;
        left: 56%;
        -ms-transform: translate(-50%, -50%);
        transform: translate(-50%, -50%);
}

.documentation a:hover
{
        text-decoration: underline;
}

.documentation a
{
        float: left;
        width: 305px;
        height: 100px;
        margin-bottom: 20px;
        padding-top: 40px;
        padding-left: 1%;
        padding-right: 1%;
        text-align: center;
        color: white;
        background-color: #523015;
        outline-color: #b3562e;
        outline-width: 4px;
        outline-style: inset;
}

#reports-container
{
        float: left;
        width: 33.3%;
```

```
        min-width: 350px;
        max-width: 500px;
        height: 241.8px;
        display: block;
        padding-top: 20px;
        margin-bottom: 30px;
        /*background-color: #b3562e;*/
        text-align: center;
}

#reports-container ul
{
        margin-top: -4px;
        text-decoration: none;
        list-style-type: none;
}

#gantt-chart
{
        position: relative;
        width: 100%;
        height: 100%;
}

#gantt-chart iframe
{
        width: 100%;
        height: 94%;
        min-height: 565px;
        max-width: 1920px;
        max-height: 1035px;
}

#gantt-chart a
{
        width: 100%;
        text-align: center;
}

.bottom-bar
{
        position: absolute;
        width: 100%;
        height: 40px;
        bottom: 5px;
        background-color: #784d2c;
```

```
        outline-color: #b3562e;
        outline-width: 4px;
        outline-style: outset;
}

.read-head
{
        position: absolute;
        top: 24%;
        left: 50%;
        -ms-transform: translate(-50%, -50%);
        transform: translate(-50%, -50%);
        border-left: 25px ridge transparent;
        border-right: 25px ridge transparent;
        border-top: 10px ridge red;
}

.read-head-indicator
{
        margin: 0;
        position: absolute;
        top: 10%;
        left: 50%;
        -ms-transform: translate(-50%, -50%);
        transform: translate(-50%, -50%);
        width: auto;
        max-width: 100%;
        height: 35px;
        overflow: hidden;
        padding-left: 20px;
        padding-right: 20px;
        background-color: black;
        border-style: groove;
        border-width: 6px;
        border-color: grey;
}

.t-machine
{
        width: 99.93%;
        height: 100%;
        /*background-color: white;*/
        margin: 0;
        margin-top: 15%;
        position: relative;
        padding-left: 2px;
```

```
}

.tape
{
	width: 100%;
	height: 75px;
	background-color: black;
	position: absolute;
	top: 50%;
	left: 50.02%;
	-ms-transform: translate(-50%, -50%);
	transform: translate(-50%, -50%);
	white-space: nowrap;
	overflow: hidden;
	display: table;
}

.tape-cell
{
	position: absolute;
	width: 11.11%;
	height: 75px;
	border-style: groove;
	border-width: 7px;
	border-color: grey;
}

.cell-contents
{
	position: absolute;
	top: 50%;
	left: 50%;
	transform: translate(-50%, -50%);
	-ms-transform: translate(-50%, -50%);
}

.speedSelectorBounds
{
	position: absolute;
	width: 100%;
	color: black;
	top: 77%;
	left: 50%;
	transform: translate(-50%, -50%);
	-ms-transform: translate(-50%, -50%);
}
```

```
#speedSelector
{
	direction: rtl;
	position: absolute;
	width: 25%;
	top: 50%;
	left: 50%;
	transform: translate(-50%, -50%);
	-ms-transform: translate(-50%, -50%);
}

#slowLabel, #fastLabel
{
	position: absolute;
	top: 50%;
	left: 50%;
	transform: translate(-50%, -50%);
	-ms-transform: translate(-50%, -50%);
	color: white;
}

#slowLabel
{
	left: 31%;
}

#fastLabel
{
	left: 69%;
}

#currentSpeedLabel
{
	width: 200px;
	margin-top: 45px;
	margin-left: 31%;
	color: white;
}

#currentSpeed
{
	width: 100px;
	margin-top: -20px;
	margin-left: 63%;
	color: white;
```

```
}

.about-description
{
	width: 94%;
	height: auto;
	padding: 10px;
	background-color: #523015;
	outline-color: #b3562e;
	outline-width: 4px;
	outline-style: inset;
}

.profiles-wrapper
{
	position: absolute;
	width: 94%;
	height: auto;
	top: 55%;
	left: 50%;
	-ms-transform: translate(-50%, -50%);
	transform: translate(-50%, -50%);
}

.profile
{
	float: left;
	width: 32%;
	height: auto;
	background-color: #523015;
	outline-color: #b3562e;
	outline-width: 4px;
	outline-style: inset;
	text-align: center;
}

.info_button {
   position: absolute;
   top: -0.4%;
   right: 1.2%;
   font-size: 15pt;
   width: auto;
   height: auto;
   user-select: none;
}
```

```
.alternate_button {
	color: black;
}

.info_button:hover {
   cursor: pointer;
   color: #b3562e;
}

.help-box
{
	position: absolute;
	width: 350px;
	height: 375px;
	padding-left: 10px;
	padding-right: 10px;
	padding-top: 3px;
	padding-bottom: 5px;
	text-align: justify;
	top: 5%;
   right: 5%;
   z-index: 1;
	background-color: #784d2c;
	outline-color: #b3562e;
	outline-width: 6px;
	outline-style: ridge;
	color: white;
	font: initial;
	overflow-y: auto;
}

.help-box a
{
	color: red;
}

#help-machine-message
{
	top: 11%;
}

#help-machine
{
	right: 0.6%;
}
```

```
#help-console-message
{
	right: 3%;
	top: -195%;
}

#help-console
{
	right: .6%;
}

.help-link-behavior
{
	cursor: pointer;
	user-select: none;
	width: auto;
	height: auto;
}
```

animation.js

```
var cell = document.querySelectorAll('.tape .tape-cell');

var basecellpos = [];   //cell positions when website is loaded
var cellpos = [];       //live positions of each cell

var speedSelector = document.getElementById("speedSelector");   //slider from webpage
var desiredSpeedRaw = parseInt(speedSelector.value)/100; //raw value from slider in
miliseconds
var desiredSpeed = desiredSpeedRaw + 's';
var displaySpeed = desiredSpeed;

for(let i=0;i<cell.length;i++)
{
        basecellpos[i] = parseFloat(cell[i].style.left);
        cellpos[i] = parseFloat(cell[i].style.left);
}


function moveRight()
{
        let tapeBeginning = findBeginning();
        let tapeEnd = findEnd();

        if(tapeEnd != 100)
        {
                for(let i=tapeEnd;i>tapeBeginning;--i)
                {
                        let pos = parseFloat(cellpos[i]) - 11.11 + '%';
                        cell[i].style.left = pos;
                        cellpos[i] = pos;
                }
        } else {
                //if initial cell in the tape is reached, alert the user
                alert('Maximum side reached.');
        }
}

function moveLeft()
{
        let tapeBeginning = findBeginning();
        let tapeEnd = findEnd();
        //console.log(tapeEnd);

        if(tapeEnd != 0)
```

```
        {
                for(let i=tapeBeginning;i<tapeEnd;++i)
                {
                        let pos = parseFloat(cellpos[i]) + 11.11 + '%';
                        cell[i].style.left = pos;
                        cellpos[i] = pos;
                }
        } else {
                //if final cell in the tape is reached, alert the user
                alert('Maximum side reached.');
        }
}

//reset cell positions to default
function resetPos()
{
        for(let i=0;i<cell.length;++i)
        {
                let pos = 0;
                pos = parseFloat(basecellpos[i]) + '%';
                cellpos[i] = basecellpos[i];
                cell[i].style.left = pos;
        }
}

//finds cell one position before viewable cells on the right side
function findBeginning()
{
        let startCell = 0;

        //get cell leftmost on the tape
        do
        {
                startCell++;
        }
        while(cell[startCell].style.left == '-11.11%');

        //set startCell to one position behind previous result
        startCell -= 1;

        return startCell;
}

//finds cell one position before viewable cells on the left side
function findEnd()
{
```

```
	let endCell = 100;
	//get cell rightmost on the tape
	while(cell[endCell].style.left == '99.99%')
	{
		endCell--;
	}

	//set startCell to one position behind previous result
	endCell += 1;

	return endCell;
}

function getSpeed()
{
	desiredSpeedRaw = parseInt(speedSelector.value)/100;
	return desiredSpeedRaw;
}

//get user-selected speed from slider and post value in seconds below slider
speedSelector.oninput = function() {
	var speedSelectorValue = getSpeed();
	document.getElementById('currentSpeed').innerHTML = speedSelectorValue + '
s';
	for(let i=0;i<cell.length;i++)
	{
		cell[i].style.transitionDuration = speedSelectorValue + 's';
		desiredSpeed = speedSelectorValue + 's';
	}
}
```

app.js

```
import layout from "./layout.js";
import compiler from "./compiler.js";
import consoleDisplay from "./consoleDisplay.js";
import machine from "./machine.js";

let turingMachine = new machine();

//window.machine = undefined;
let machineLayout = new layout();
let markupCompiler = new compiler();
let machineConsole = new consoleDisplay();

//let turingMachine = new machine(machineConsole);

var appError;//temporary fix

markupCompiler.loadCode = function () {
   this.userCode = document.getElementById('code-area').value;
   appError = this.scanTokens();
   if(!appError){ //while still tokens to scan
      appError = this.parseTokens(turingMachine);
   }
         else{
                  console.log("compiler failed");
   }

   if(appError){ //should probably have a generic error here since it should have already
thrown errors
      console.log(appError);
   }
   else
   {
      machineConsole.displayMachine(turingMachine);
   }
};
```

compiler.js

```
import machine from "./machine.js";
import "./mapper.js";
import errorHandler from "./errorHandler.js";

export default class compiler {

    //various regex combos
    SINGLE_CHAR_TOKENS = /#|-|=|;|,|\[|\]/;
    LOOK_AHEAD_TOKENS = /'|"/;
    WHITESPACE_TOKENS = /\s/;

    inputString_RegEx = /.+/;
    blankString_RegEx = /^.$/; //forces a single char
    startOrAcceptState_RegEx = /\S+/;

    stateName_RegEx = /\S+/;
    potentialRead_RegEx = /^.$/;
    writeAction_RegEx = /^.$/;
    directionAction_RegEx = /^(r|l)$/;
    nextStateAction_RegEx = /\S+/;

  constructor() {
    this.numTokens = 0;
    this.tokens = new Array();

    this.tempInputString = "";
    this.tempBlankString = "";
    this.tempStartStateString = "";
    this.tempAcceptStateString= "";

    this.potentialReads_Set = new Array();
    this.actions_Set = new Array();
    this.states_Set = new Array();

    this.errorHandler = new errorHandler();

    this.userCode = undefined;
    this.loadCode = undefined;
    this.editRunning = false;

    document.getElementById('load').addEventListener('click', () => this.loadCode());
    document.getElementById('edit_current_machine').addEventListener('click', () =>
this.setEditFlagAndCompile());
  }
```

```
setEditFlagAndCompile(){
   this.editRunning = true;
   document.getElementById('load').click();
   this.editRunning = false;
}
scanTokens() //looking for tokens
{
   let code = this.userCode.toLowerCase();
   let numTokens = 0;
   let index = 0;
   let stopScan = false;
   let tempToken = "";
   let lookAheadMatch = "";
   let activeToken = false;
            let lineNumber = 1;

   //this.tokens = {undefined};
            this.tokens = [];

   while(index < code.length && !stopScan)
   {
      if(this.SINGLE_CHAR_TOKENS.test(code[index])) //if it's a singular char
      {
         if(activeToken) //started a token already
         {
            this.tokens[numTokens] = [tempToken, lineNumber];
            numTokens++;
            tempToken = "";
            activeToken = false;
         }

         if(code[index] == '#') //if it's a comment
         {
            do{
               index++;
            }while(index < code.length && code[index] != '\n');

                                    lineNumber++; //since index increments again after

         }
         else //single char token
         {
            tempToken += code[index];
            this.tokens[numTokens] = [tempToken, lineNumber];
            tempToken = "";
```

```
            numTokens++;
        }
    }

    else if(this.LOOK_AHEAD_TOKENS.test(code[index])) //if it's a string and we
need to look for its end
    {
        if(activeToken)
        {
            this.tokens[numTokens] = [tempToken, lineNumber];
            numTokens++;
            tempToken = "";
            activeToken = false;
        }

        lookAheadMatch = code[index];

        do{
            if(this.WHITESPACE_TOKENS.test(code[index]) && code[index] != " ")
            {
                stopScan = true;
                this.errorHandler.printBadEOL(lineNumber);
            }
            tempToken += code[index];
            index++;
        }while(index < code.length && code[index] != lookAheadMatch);

        if(index >= code.length && !stopScan)
        {
            stopScan = true;
            this.errorHandler.printBadEOF(lineNumber);
        }
        else
        {
            tempToken += code[index];
            this.tokens[numTokens] = [tempToken, lineNumber];
            numTokens++;
            tempToken = "";
            activeToken = false;
        }
    }

    else if(this.WHITESPACE_TOKENS.test(code[index]))
    {
        if(activeToken)
        {
```

```
            this.tokens[numTokens] = [tempToken, lineNumber];
            numTokens++;
            tempToken = "";
            activeToken = false;
          }

                                    if(code[index] == "\n") { //if new line
                                        lineNumber++;
                                    }

        }

        else{
          tempToken += code[index];
          activeToken = true;
        }

        if((index == code.length-1 && activeToken))
        {
          this.tokens[numTokens] = [tempToken, lineNumber];
          numTokens++;
        }
        index++;
      }
      this.numTokens = numTokens;
                this.cleanTokens();

      return stopScan;
    }

        cleanTokens() { //strips "/' from tokens so we don't have to keep checking them
every time
                let index = 0;

                while(index < this.numTokens) {
                        if(this.tokens[index][0].includes("\"") ||
this.tokens[index][0].includes("\'")) {
                                this.tokens[index][0] =
this.tokens[index][0].substr(1,this.tokens[index][0].length-2)
                        }
                        index++;
                }
                return;
        }

    parseTokens(turingMachine)
```

```
{
   //let turingMachine = undefined;
   let index = 0;
   let stopParse = false;
   let addState = false;

   let tempNumPotentialReads = 0;
   let tempNumActions = 0;
   let numStates = 0;

   let tempState = new Array();

   let tempErrorCode = 0;

   while(index < this.numTokens && !stopParse)
   {
      switch(this.tokens[index][0])
      {
         case "input":
            stopParse = this.match_Input(index);
            if(!stopParse)
            {
               index+=3;
            }
            break;

         case "blank":
            stopParse = this.match_Blank(index);
            if(!stopParse)
            {
               //console.log("blank");
               index+=3;
            }
            break;

         case "start":
                                          //since these use the same func, just group together
         case "accept":
            stopParse = this.match_StartOrAcceptState(index);
            if(!stopParse)
            {
               index+=3;
            }
            break;

         case "-": //grabs a state, its potential reads and potential nexts
```

```
                stopParse = this.match_StateName(index);
                if(!stopParse)
                {
                   index+=2;
                   if(this.tokens[index][0] != ";") //if we haven't reached the end of a line
                   {
                      tempErrorCode = tempNumPotentialReads =
this.match_PotentialReads(index);
                      while(tempNumPotentialReads > 0 && !stopParse)
                      {
                         index += (tempNumPotentialReads*2)-1;
                         tempNumPotentialReads = 0;

                         stopParse = this.match_ActionSet(index);
                         if(!stopParse)
                         {
                            //index+=(tempNumActions*2)+1;
                                                                  index += 7;

                            if(this.tokens[index][0] == ";")
                            {
                               addState = true;
                            }
                            else{
                               tempNumPotentialReads = this.match_PotentialReads(index);
                            }
                         }

                         else{
                            this.errorCode = tempErrorCode;
                            stopParse = true;
                            //console.log("expected action set");
                         }

                      }
                      if(tempNumPotentialReads < 0)
                      { //update this
                         this.errorCode = tempErrorCode;
                         stopParse = true;
                      }
                   }
                   else
                   {
                      addState = true;
                   }
```

```
            if(addState)
            {
               tempState[0] = this.tempStateName;
               tempState[1] = this.potentialReads_Set;
               tempState[2] = this.actions_Set;
               this.states_Set[numStates] = tempState;
               numStates++;
               tempState = {undefined};
               this.potentialReads_Set = new Array();
               this.actions_Set = new Array();
               addState = false;
            }
         }

         else
         {
            this.errorCode = tempErrorCode;
            //console.log("expected alphanumeric statename");
            stopParse = true;
         }

         break;
                                    //this is the end of the - case

                           default:

   this.errorHandler.printBadStartLine(this.tokens[index]);
                                    stopParse = true;
                                    console.log("improper start");
                                    break;
   }
   index++;
}

            if(!stopParse) { //don't need to check these if it already failed
                  if(this.tempInputString == "")
                  {
                        stopParse = true;
                        this.errorHandler.printBadInput();
                  }
                  else if(this.tempBlankString == "")
                  {
                        stopParse = true;
                        this.errorHandler.printBadBlank();
                  }
```

```
                        else if(this.tempStartStateString == "")
                        {
                                stopParse = true;
                                this.errorHandler.printBadStart();
                        }
                        else if(this.tempAcceptStateString == "")
                        {
                                stopParse = true;
                                this.errorHandler.printBadAccept();
                        }
                        else if(!this.checkExistingStartState())
                        {
                                //console.log("no start good");
                                stopParse = true;
                                this.errorHandler.printBadStartDef();
                        }
                        else if(!this.checkExistingAcceptState())
                        {
                                stopParse = true;
                                 this.errorHandler.printBadAcceptDef();
                        }
                }

        if(!stopParse) //sanity check this
        {
                        if(this.states_Set[0] != null)
                        {
                if(this.editRunning && this.checkEdited(turingMachine))
                {
                   turingMachine.editMachine(this.states_Set);
                   console.log(turingMachine);
                }
                else{
                   updateTape(this.tempInputString, this.tempBlankString);
                   turingMachine.createMachine(this.tempInputString,
this.tempBlankString,this.tempStartStateString, this.tempAcceptStateString,
this.states_Set);
                   console.log(turingMachine);
                }

this.tempInputString=this.tempBlankString=this.tempStartStateString=this.tempAcceptSt
ateString = "";
                this.states_Set = new Array();
                        }
                        else{stopParse = true;}//no states defined
                   }
```

```
                //1 is true, 0 is false

        return stopParse;
    }

    checkExistingStartState()
    {
        let i = 0;
        let exists = false;
        while( i < this.states_Set.length && !exists)
        {
            //console.log(`${this.states_Set[i][0]}:${this.tempStartStateString}`);
            if(this.states_Set[i][0] == this.tempStartStateString)
                exists = true;
            i++;
        }
        return exists;
    }

    checkExistingAcceptState()
    {
        let i = 0;
        let exists = false;
        while( i < this.states_Set.length && !exists)
        {
            if(this.states_Set[i][0] == this.tempAcceptStateString)
                exists = true;
            i++;
        }

        return exists;
    }

    match_Input(index) //look for a valid input string
    {
        let stopParse = false;
        if(this.tokens[index+1][0] == "=" &&
this.inputString_RegEx.test(this.tokens[index+2][0])
                        && this.tokens[index+3][0] == ";")
        { //if set of tokens is =, a valid input string and ;
            this.tempInputString = this.tokens[index+2][0];
        }
                else {
                        this.errorHandler.printBadInputDef(this.tokens[index+1][1]);
                        stopParse = true;
```

```
                }
                //don't need to check if tempInputString is empty cause it won't accept an
empty string

      return stopParse;
   }

   match_Blank(index) //look for a valid blank char
   {
      let stopParse = false;
      if(this.tokens[index+1][0] == "=" &&
this.blankString_RegEx.test(this.tokens[index+2][0])
                        && this.tokens[index+3][0] == ";")
      { //if set of tokens is =, a single char and ;
         this.tempBlankString = this.tokens[index+2][0];
         stopParse = false;
      }
                else {
                        this.errorHandler.printBadBlankDef(this.tokens[index+1][1]);
                        stopParse = true;
                }
                //again, can't have empty strings

      return stopParse;
   }

   match_StartOrAcceptState(index) //look for valid start and accept states
   {
      let stopParse = false;
      if(this.tokens[index+1][0] == "=" &&
this.startOrAcceptState_RegEx.test(this.tokens[index+2][0])
                        && this.tokens[index+3][0] == ";")
      { //if set of tokens is =, a valid string and ;
         if(this.tokens[index][0] == "start")
         {
            this.tempStartStateString = this.tokens[index+2][0];
         }
         else
         {
            this.tempAcceptStateString = this.tokens[index+2][0];
         }
      }
                else {
                        this.errorHandler.printBadStateDef(this.tokens[index+1][1]);
                        stopParse = true;
                }
```

```
                //needs to have an start/accept name

        return stopParse;
    }

    match_StateName(index) //looks for a valid state name
    {
        let stopParse = false;
        if(this.stateName_RegEx.test(this.tokens[index+1][0]))
        {
            this.tempStateName = this.tokens[index+1][0];
        }
                else {
                        stopParse = true;
                        this.errorHandler.printBadStateName(this.tokens[index+1]);
                }
        return stopParse;
    }

    match_PotentialReads(index) //looks for all the potential reads
    {
        let returnVal = 0;
        let stopMatch = false;

        let tempPotentialReadsString = "";

        while(returnVal > -1 && !stopMatch)
        {
            if(this.potentialRead_RegEx.test(this.tokens[index][0])
                                || (this.tokens[index][0] != null &&
this.tokens[index][0].length == 1))
            { //if a string, not null and has a length of 1
                returnVal++;
                                tempPotentialReadsString += this.tokens[index][0];

                if(this.tokens[index+1][0] != ",")
                {
                    this.potentialReads_Set.push(tempPotentialReadsString);
                    stopMatch = true;
                }
                else
                {
                    index+=2;
                }
            }
            else
```

```
        {
                                this.errorHandler.printBadStateRead(this.tokens[index -
1]);
                                //-1 to get the state name and not the potential read
            returnVal = -209;//no potential reads found
                                //fix this later
        }
      }
      return returnVal;
   }

   match_ActionSet(index)
   {
                let stopParse = false;
      let returnVal = 0;
      let tempActionsString = "";

      if(this.tokens[index][0] == "[")
      {
         if((this.writeAction_RegEx.test(this.tokens[index+1][0]))&&
this.tokens[index+2][0] == ",")
         { //if valid replacement state character and comma
            if(this.directionAction_RegEx.test(this.tokens[index+3][0]) &&
this.tokens[index+4][0] == ",")
            { //if r or l and comma
               if(this.nextStateAction_RegEx.test(this.tokens[index+5][0]) &&
this.tokens[index+6][0] == "]")
               {
                                                tempActionsString +=
this.tokens[index+1][0] + "\n" + this.tokens[index+3][0] +
                                                "\n" + this.tokens[index+5][0];
                  this.actions_Set.push(tempActionsString);
                  //returnVal = 3;
               }
               else{
                  this.errorHandler.printBadNextStateAction(this.tokens[index+5],
this.tokens[index+6][0]);
                                                stopParse = true;
                                          }
            }
            else{
               this.errorHandler.printBadDirection(this.tokens[index+3]);
                                       stopParse = true;
            }
         }
         else
```

```
            {
                this.errorHandler.printBadWriteAction(this.tokens[index+1],
this.tokens[index+2][0]);
                                    stopParse = true;
            }
        }
        else{
                                this.errorHandler.printNoBracket(this.tokens[index][1]);
                                stopParse = true;
        }

        return stopParse;
    }

    checkEdited(tm)
    {
        let retVal = false;

        if(tm.input == this.tempInputString && tm.blank == this.tempBlankString &&
tm.start == this.tempStartStateString && tm.accept == this.tempAcceptStateString)
        {
            let i = 0;
            let currentStateStillExists = false;
            while(i < this.states_Set.length && !currentStateStillExists)
            {
                console.log(this.states_Set[i][0]);
                if(tm.currentState[0] == this.states_Set[i][0] || tm.stateStuckIn ==
this.states_Set[i][0])
                {
                    currentStateStillExists = true;
                    retVal = true;

                    console.log("good to go");
                }
                i++;
            }
            if(!currentStateStillExists)
            console.log("current state did not exist, restarting machine");
        }
        else{
            console.log("you edited essential machine parts, restarting machine");
        }

        return retVal;
    }
}//end of class
```

consoleDisplay.js

```
export default class consoleDisplay {

   constructor() {
      this.console = document.getElementById("consoleText");
      document.getElementById('clearConsole').addEventListener('click', () =>
this.clearConsole());
   }

   displayErrorTemp(errorNum){
      this.console.value = `error: ${errorNum}`;
   }

   clearConsole()
   {
      this.console.value = "";
   }

   testFunction()
   {
      this.console.innerHTML = "it works";
   }

         setValue(value) {
                  this.console.value = value;
         }

   displayMachine(machine)
   {
      this.console.value =`Successful Machine Build!\n\n`;
      this.console.value +=`input:\t\t\t\"${machine.input}\"\n`;
      this.console.value +=`blank character:\t\"${machine.blank}\"\n`;
      this.console.value +=`starting state:\t\t${machine.start}\n`;
      this.console.value +=`accepting state:\t${machine.accept}\n\n`;

      this.console.value =`Successful Machine Build!
                  i: ${machine.input}
                  b: "${machine.blank}"
                  s: ${machine.start}
                  a: ${machine.accept}\n`;

      for(let i = 0; i < machine.numStates;i++)
      {
         let numOfValues = 0;
         this.console.value += `${machine.states[i][0]}:\n`
```

```
            while(numOfValues < machine.states[i][1].length)
            {
               this.console.value += `\tread:\"${machine.states[i][1][numOfValues]}\"`;

               let WDN = machine.states[i][2][numOfValues];

               this.console.value += `    write:\"`;
               WDN = WDN.replace("\n",'\"    direction:\"');
               WDN = WDN.replace("\n",'\"    next state:\"');

               this.console.value += `${WDN}\"\n`;

               numOfValues++;
            }
         }

      }

      machineIsStuckInAStateThatDoesNotExist(stateStuckIn, lastState)
      {
         this.console.value = `The machine is stuck in -${stateStuckIn} (Transistioned to
from ${lastState})
         Possible causes:
                  1. -${stateStuckIn} does not have any transition functions defined for the
symbol under read head`;
      }

      stateHasNoAvailableTransitions(stateStuckIn, currentRead)
      {
         this.console.value = `The machine is stuck in -${stateStuckIn}
         Possible causes:
                  1. -${stateStuckIn} does not have any transition functions defined for
${currentRead}`;
      }
}
```

errorHandler.js

```
import consoleDisplay from "./consoleDisplay.js";

export default class errorHandler { //prints errors, wowee

	constructor() {
		this.display = new consoleDisplay();
		//double check this is correct
	}

	printBadEOF(lineNumber) {
		//-100
		this.display.clearConsole();
		this.display.setValue("Expected end of quotes but reached end of file on
line", lineNumber);
	}

	printBadInput() {
		//-200, double check if needed
		this.display.clearConsole();
		this.display.setValue("No input defined");
	}

	printBadBlank() {
		//-201
		this.display.clearConsole();
		this.display.setValue("No blank character defined");
	}

	printBadStart() {
		//-202
		this.display.clearConsole();
		this.display.setValue("No start state defined");
	}

	printBadAccept() {
		//-203
		this.display.clearConsole();
		this.display.setValue("No accept state defined");
	}

	printBadStates() {
		//-204
		this.display.clearConsole();
		this.display.setValue("No states were defined for the machine");
```

```
        }

        printBadInputDef(lineNum) {
                //-205
                this.display.clearConsole();
                this.display.setValue(`Input was not properly defined on line
${lineNum}\n`);
        }

        printBadBlankDef(lineNum) {
                //-206
                this.display.clearConsole();
                this.display.setValue(`Blank was not properly defined on line
${lineNum}\n`);
        }

        printBadStateDef(lineNum) {
                //-207
                this.display.clearConsole();
                this.display.setValue(`State was not properly defined on line
${lineNum}\n`);
        }

        printBadStateName(tokArr) {
                //-208
                this.display.clearConsole();
                this.display.setValue(`State name cannot contain whitespace characters
                Inputted state name: ${tokArr[0]}
                Line number: ${tokArr[1]}\n`);
        }

        printBadEOL(lineNumber) {
                //-220
                this.display.clearConsole();
                this.display.setValue(`Expected end of quotes before new line on line
${lineNumber}\n`);
        }

        printBadStateRead(tokArr) {
                //-209
                this.display.clearConsole();
                this.display.setValue(`Expected list of potential reads for state
${tokArr[0]} on line ${tokArr[1]}\n`);
        }

        printNoBracket(lineNum) { //210
```

```
            this.display.clearConsole();
            this.display.setValue(`Missing opening bracket for action set on line
${lineNum}\n`);
        }

        printBadStateSyntax(stateContext, symbolContext) { //don't need \n with template
literals
            this.display.clearConsole();
            this.display.setValue(`Potentially incorrect syntax for action set.
            State: ${stateContext}
            Symbols: ${symbolContext}
            Expected: [char, (l|r), state name]\n`);
        }

        printBadWriteAction(tokArr, comma)          { //211
            this.display.clearConsole();
            this.display.setValue(`Expected a valid write action and a comma on line
${tokArr[1]}
            Write action: ${tokArr[0]}
            Comma: ${comma}\n`)
        }

        printBadNextStateAction(tokArr, bracket) { //212
            this.display.clearConsole();
            this.display.setValue(`Expected a valid state name and a closing bracket
on line ${tokArr[1]}
            State name: ${tokArr[0]}
            Bracket: ${bracket}\n`);
        }

        printBadDirection(tokArr) { //213
            this.display.clearConsole();
            this.display.setValue(`Expected a singular r or l on line ${tokArr[1]}
            Current token: ${tokArr[0]}\n`);
        }

        printBadStartLine(tokArr) { //214
            this.display.clearConsole();
            this.display.setValue(`Current token on line ${tokArr[1]} is not a valid
line start
            Current token: ${tokArr[0]}
            Expected: input, blank, start, accept or -\n`);
        }
}
```

help.js

```
var helpWindowCodeEditor = document.querySelector('#help-code-editor-message');
var helpWindowMachine = document.querySelector('#help-machine-message');
var helpWindowStateDiagram = document.querySelector('#help-state-diagram-
message');
var helpWindowConsole = document.querySelector('#help-console-message');

//information button display in transition editor
document.getElementById('help-code-editor').addEventListener('click', (event) => {
	if(helpWindowCodeEditor.style.display === 'none')
	{
		helpWindowCodeEditor.style.display = 'block';
		document.getElementById('help-code-editor').innerHTML = '&#x2715;';
	} else {
		helpWindowCodeEditor.style.display = 'none';
		document.getElementById('help-code-editor').innerHTML = '&#x1F6C8;';
	}
});

//information button display in animator
document.getElementById('help-machine').addEventListener('click', (event) => {
	if(helpWindowMachine.style.display === 'none')
	{
		helpWindowMachine.style.display = 'block';
		document.getElementById('help-machine').innerHTML = '&#x2715;';
	} else {
		helpWindowMachine.style.display = 'none';
		document.getElementById('help-machine').innerHTML = '&#x1F6C8;';
	}
});

//information button display in console
document.getElementById('help-console').addEventListener('click', (event) => {
	if(helpWindowConsole.style.display === 'none')
	{
		helpWindowConsole.style.display = 'block';
		document.getElementById('help-console').innerHTML = '&#x2715;';
	} else {
		helpWindowConsole.style.display = 'none';
		document.getElementById('help-console').innerHTML = '&#x1F6C8;';
	}
});
```

layout.js

```
export default class layout{
   constructor()
   {

      this.codeArea = document.getElementById('code-area');
      this.lineNumbers = document.getElementById('line-numbers');
      var lineCount = 1;
      this.consoleShown = 1;
      this.editorShown = 1;
      this.machineShown = 1;
      this.layout = 1;

      //initializes functions so they can be defined in main/index.js

      //adds scroll event listener to code-area. allows the sync with line-numbers scroll
      document.getElementById('code-area').addEventListener('scroll', (event) => {

      //function to run on scroll
         this.syncEditorScroll();
      });

      /*
      adds input listener to code-area

      if the number of lines changes in code-area, then the number of lines shown in line-
numbers will be synced
      */
      document.getElementById('code-area').addEventListener('input', (event) => {

         var linesActual = this.countLines();
         //function to run on input
            if(linesActual != lineCount)
            {
               lineCount = linesActual;
               this.lineNumbers.value = "";
               for(let i = 1;i <= lineCount;i++)
               {
                  this.lineNumbers.value += i +"\n";
               }
            }
         });

      //adds click event listener to the reset button
      document.getElementById('select').addEventListener('click', (event) => {
```

```
        const isButton = event.target.nodeName === 'BUTTON';
        if (!isButton) {
            return;
        }
        //function to run on click
        this.onSelectClick();
        });

                //adds click event listener to the reset button
    document.getElementById('toggle_console').addEventListener('click', (event) => {
        const isButton = event.target.nodeName === 'BUTTON';
        if (!isButton) {
            return;
        }
        //function to run on click
            this.toggleConsole();
        });

    document.getElementById('toggle_editor').addEventListener('click', (event) => {
        const isButton = event.target.nodeName === 'BUTTON';
        if (!isButton) {
            return;
        }
        //function to run on click
            this.toggleEditor();
        });

    document.getElementById('toggle_machine').addEventListener('click', (event) => {
        const isButton = event.target.nodeName === 'BUTTON';
        if (!isButton) {
            return;
        }
        //function to run on click
            this.toggleMachine();
        });

}

onLoadClick(){
    console.log(document.getElementById("code-area").value);
}

onSelectClick(){
    console.log("Select");
}
```

```
//syncs line-numbers with code-area scroll
syncEditorScroll(){
   this.lineNumbers.scrollTop = this.codeArea.scrollTop;
}

//returns number of lines of text are in code-area
countLines()
{
   let lines = 1;
   for(let i = 0; i <= this.codeArea.value.length;i++)
   {
      if(this.codeArea.value[i] == '\n')
         lines++;
   }
   return lines;
}

toggleConsole()
{
   if(this.consoleShown){
      document.getElementById("console").style.display = "none";
      document.getElementById("editor").style.bottom = "1%";
      document.getElementById("machine").style.bottom = "1%";
   }

   else{
      document.getElementById("console").style.display = "";
      document.getElementById("editor").style.bottom = "27%";
      document.getElementById("machine").style.bottom = "27%";
   }

   this.consoleShown = !this.consoleShown;
}

toggleEditor()
{
   if(this.machineShown == 1)
   {
      if(this.editorShown){
         document.getElementById("editor").style.display = "none";
         document.getElementById("machine").style.left = "1%";
      }

      else{
         document.getElementById("editor").style.display = "";
         document.getElementById("machine").style.left = "50.35%";
```

```
            }

            this.editorShown = !this.editorShown;
        }
    }

    toggleMachine()
    {
        if(this.editorShown == 1)
        {
            if(this.machineShown){
                document.getElementById("machine").style.display = "none";
                document.getElementById("editor").style.right = "1%";
            }

            else{
                document.getElementById("machine").style.display = "";
                document.getElementById("editor").style.right = "50.35%";
            }

            this.machineShown = !this.machineShown;
        }
    }
}
```

machine.js

```
import "./mapper.js";
import "./animation.js";
import consoleDisplay from "./consoleDisplay.js";
export default class machine {

    constructor(machineConsole) {
        this.states = new Array();
    this.input = "";
    this.blank = '';
    this.start = "";
    this.accept = "";
    this.numStates = 0;
        this.result = "";
        this.currentState = "";
        this.playback;
        this.lastStateTransitionedTo = "";
        this.stateStuckIn = "";
        this.HelpImStuckInAStateThatDoesNotExist = false;

        this.machineConsole = machineConsole;

        document.getElementById('step').addEventListener('click', () =>
this.findTransition());
        document.getElementById('play').addEventListener('click', () =>
this.playMachine());
        document.getElementById('pause').addEventListener('click', () =>
this.stopTheBus());
        document.getElementById('reset').addEventListener('click', function(){
            document.getElementById('load').click();
            //if reset button is clicked, simulate a click on the load button
        });
    }

    createMachine(input, blank, start, accept, states_Set)
    {
        this.states = new Array();
    this.input = input.slice(0);
    this.blank = blank;
    this.start = start;
    this.accept = accept;
    this.numStates = states_Set.length;
        this.result = input;
        this.lastStateTransitionedTo = "";
        this.HelpImStuckInAStateThatDoesNotExist = false;
```

```
for(let i = 0; i < this.numStates;i++)
{
   this.states[i] = new Array(states_Set[i][0], states_Set[i][1], states_Set[i][2]);
}
            this.currentState = this.findState(this.start);
   }

   editMachine(states_Set)
   {
            this.states = new Array();
            this.numStates = states_Set.length;
            this.lastStateTransitionedTo = "";
            this.HelpImStuckInAStateThatDoesNotExist = false;

            for(let i = 0; i < this.numStates;i++)
{
   this.states[i] = new Array(states_Set[i][0], states_Set[i][1], states_Set[i][2]);
}
            this.currentState = this.findState(this.currentState[0]);
   }

   getStates() { //these three mostly for canvas
            return this.states;
   }

   getStart() {
            return this.start;
   }

   getAccept() {
            return this.accept;
   }

   playMachine()
   {
            this.playback = setInterval(function(){
                     document.getElementById('step').click();
            }, getSpeed()*1000);
   }

   stopTheBus()  //I mean, stop the machine...
   {
            clearInterval(this.playback);
   }

   findTransition() {
```

```
		//descriptors for looking through state array
		const STATENAME = 0; //ex [0][0] "start"
		const POTENTIALREADS = 1; //ex [0][1] "0"
		const TRANSITIONSTEPS = 2; //ex [0][2] "1;r;q0"

		let currentRead = getCellUnderHead();
		let currentState = this.currentState;

		let error = 0;
		if(this.HelpImStuckInAStateThatDoesNotExist)
		{

	this.machineConsole.machineIsStuckInAStateThatDoesNotExist(this.stateStuckIn
, this.lastStateTransitionedTo);
		}
		else if(currentState[STATENAME] != this.accept && error >= 0)
		{
			for(let i = 0; i < currentState[POTENTIALREADS].length; i++)
			{

	if(currentState[POTENTIALREADS][i].includes(currentRead)){//proceed with
the parallel transition steps

	this.performTransition(currentState[TRANSITIONSTEPS][i], this.result);
						i=currentState[POTENTIALREADS].length +1;
					}
					else if(i == currentState[POTENTIALREADS].length -
1){//if the state does not recognise the character
						//this.errorHandler.printInvalidCharacter();
						console.log("cannot transition:" + currentRead);

	this.machineConsole.stateHasNoAvailableTransitions(this.stateStuckIn,
currentRead);
					}
			}
		}

		if(currentState[STATENAME] == this.accept)//is the machine done?
		{
			clearInterval(this.playback);
		}

	}

	performTransition(transitionSteps)
	{
```

```
        let validNextState = -1;
        let writeChar = "";
        let direction = "";
        let nextState = "";

                writeChar = transitionSteps[0];
                direction = transitionSteps[2];
                nextState = transitionSteps.substr(4);

        updateCell(writeChar);

        if(direction == "r")//need to visually move the read head to the right
        {
                moveRight();
        }
        if(direction == "l")//need to visually move the read head to the left
        {
                moveLeft();
        }
        validNextState = this.findState(nextState);
        if(validNextState != -1)
        {
                this.currentState = validNextState;
                this.HelpImStuckInAStateThatDoesNotExist = false;
        }
        else{
                this.HelpImStuckInAStateThatDoesNotExist = true;
                this.lastStateTransitionedTo = this.currentState[0];
                this.stateStuckIn = nextState;
                this.currentState[0] = nextState;
        }
}

findState(stateName)
{
        document.querySelector('.read-head-indicator').innerHTML = stateName;
        let i = 0;
        while(i < this.states.length && stateName != this.states[i][0])
        {
                i++;
        }
        if(i < this.states.length)
                return this.states[i];
        else
                return -1; //ERRORS AHHHHH
}
```

```
}
```

manipulatePrograms.js

```
//collection of example programs from user select
var programs = [
[
'Binary Increment',

`# Binary Increment - takes a binary number and increments
#                 it by 1.
# Input - a single binary number without spaces.
# Output - the sum of the binary number plus 1.

input = "1011";
blank = " ";
start = moveright;
accept = finish;

# move to the end of the binary number.
-moveright      1   [1, r, moveright]
                0   [0, r, moveright]
                " " [" ", l, carry];

# increment rightmost digit and update following digits.
-carry          1   [0, l, carry]
                0   [1, l, reset]
                " " [1, l, reset];

# set read head to first item of result.
-reset          " "     [" ", r, finish]
                0       [0, l, reset]
                1       [1, l, reset];

-finish;`
],
[
'Palindrome',

`# Palindrome - accepts binary palindromes.
# Input - a binary number that mirrors in the middle,
#       including an empty value.
# Output - none.

input = "0110";
blank = " ";
start = start;
accept = accept;
```

```
-start 0   [" ", r, have0]
       1   [" ", r, have1]
       " " [" ", r, accept];

-have0 0   [0, r, have0]
       1   [1, r, have0]
       " " [" ", l, match0];

-have1 0   [0, r, have1]
       1   [1, r, have1]
       " " [" ", l, match1];

-match0 0   [" ", l, back]
          1   [1, l, reject]
          " " [" ", l, accept];

-match1 1   [" ", l, back]
          0   [0, l, reject]
          " " [" ", l, accept];

-back 0   [0, l, back]
      1   [1, l, back]
      " " [" ", r, start];

-reject;
-accept;`
],
[
'Binary Addition',

`# Binary Addition - takes two binary numbers and sums
#                  them together.
# Input - two binary numbers separated by a space.
# Output - the sum of the two binary numbers entered.

input = "110110 101011";
blank = " ";
start = q0;
accept = qf;

# move to the right until a blank space is encountered.
-q0 " " [" ", r, q1]
   0      [0, r, q0]
   1      [1, r, q0]
   x      [x, r, q0]
```

```
    y      [y, r, q0];

# move right until another space is encountered.
-q1 " " [" ", l, q2]
    0      [0, r, q1]
    1      [1, r, q1];

# when a blank space is encountered, move left and remove
# the first digit seen or a blank space is encountered.
-q2 0    [" ", l, q3x]     # branching path for 0
    1      [" ", l, q3y]     # branching path for 1
    " "    [" ", l, q7];     # cleanup

# move left until a blank space is encountered, ignoring
# 0 and 1.
# branching path for 0.
-q3x " " [" ", l, q4x]
     0      [0, l, q3x]
     1      [1, l, q3x];

# move left until a blank space is encountered, ignoring
# 0 and 1.
# branching path for 1.
-q3y " " [" ", l, q4y]
     0      [0, l, q3y]
     1      [1, l, q3y];

# accepts the rightmost digit from the first binary number.
# if the number is a 0, it is replaced with an x.
# if the number is a 1, it is replaced with a y.
# if a blank space is encountered, add an x symbol.
# this occurs all while ignoring x and y symbols.
-q4x 0   [x, r, q0]
     1      [y, r, q0]
     " " [x, r, q0]
     x      [x, l, q4x]
     y      [y, l, q4x];

# accepts the rightmost digit from the first binary number.
# if the number is a 0, it is replaced with a 1.
# if the number is a 1, a carry occurs and maintains the same
# state until a 0 or blank space is encountered and write 1.
# this occurs all while ignoring x and y symbols.
-q4y 0   [1, l, q5]
     1      [0, l, q4y]
     " " [1, r, q5]
```

```
    x     [x, l, q4y]
    y     [y, l, q4y];

# move to the right until a symbol is encountered, then move
# left.
-q5 " "  [" ", l, q6]
   0     [0, r, q5]
   1     [1, r, q5]
   x     [x, l, q6]
   y     [y, l, q6];

# replace 0 with x and 1 with y, then return to repeat the cycle.
-q6 0    [x, r, q0]
   1     [y, r, q0];

# clean up symbols and return to the beginning of the binary sum
# calculated.
-q7 x    [0, l, q7]
   y     [1, l, q7]
   " " [" ", r, qf]
   0     [0, l, q7]
   1     [1, l, q7];

-qf;`
],
[
'Binary Pong',

`# Binary Pong - the read head bounces back and forth.
# Input - a binary digit and another binary digit separated by a space.
# Output - a really intense game of table tennis.

input = "0 1";
blank = " ";
start = start;
accept = never;

# first serve on the left side.
-start 0   [1, r, ping]
       1   [0, r, ping];

# receives the pitch and passes back.
-ping " " [" ", r, ping]
      1   [0, l, pong]
      0   [1, l, pong];
```

```
# also receives and hits back.
-pong " " [" ", l, pong]
     1     [0, r, ping]
     0     [1, r, ping];

# the game will never end
-never;`
],
[
'The Classic',

`# The Classic - The first example machine given by Alan Turing
#              in his 1936 paper:
#              "On Computable Numbers, with an
#              Application to the Entscheidungs problem".
# Input - none
# Output - an endless sequence of 0 1 0 1...

input = " ";
blank = " ";
start = b;
accept = never;

-b " " [0, r, c];
-c " " [" ", r, e];
-e " " [1, r, f];
-f " " [" ", r, b];

-never;`
],
[
'Binary Complement',

`# Binary Complement - accepts a binary number and outputs the complement of it.
# Input - a binary number.
# Output - the binary complement of the input.

input = "1011";
blank = " ";
start = q0;
accept = qf;

-q0 1   [0, r, q0]
   0   [1, r, q0]
   " " [" ", l, q1];
-q1 1    [1, l, q1]
```

```
          0         [0, l, q1]
          " " [" ", r, qf];

-qf;`
],
[
'x-y, where x is greater than y',
`# x-y, where x is greater than y - accepts two strings of 1s separated by a
                          0 and outputs the difference of x and y.
# Input -  two strings of 1s separated by a 0.
# Output - the difference between x and y.

input = "1111011";
blank = " ";
start = q0;
accept = qf;

-q0 1   [1, r, q0]
   0   [0, r, q0]
   " " [" ", l, q1];

-q1 1 [" ", l, q2]
   0 [" ", l, q4];

-q2 1 [1, l, q2]
   0 [0, l, q3];

-q3 1 [0, r, q0]
   0 [0, l, q3];

-q4 0   [" ", l, q4]
   1   [1, l, q4]
   " " [" ", r, qf];

-qf;`
],
[
'wcw, where w is an element of {a,b}*',
`# wcw, where w is an element of {a,b}* - accepts two strings of a combination of
                                  a's and b's in the same order separated
                                  by a c.
# Input -  two strings of 1s separated by a 0.
# Output - if each side matches, an equal amount of y's on each side of a c.

input = "aabcaab";
blank = " ";
```

```
start = q0;
accept = qf;

-q0 a [x, r, q1]
        b [x, r, q2]
        c [c, r, q6];

-q1 a [a, r, q1]
        b [b, r, q1]
        c [c, r, q3];

-q2 a [a, r, q2]
        b [b, r, q2]
        c [c, r, q4];

-q3 a [y, l, q5]
        y [y, r, q3];

-q4 b [y, l, q5]
        y [y, r, q4];

-q5 c [c, l, q5]
        b [b, l, q5]
        a [a, l, q5]
        x [x, r, q0]
        y [y, l, q5];

-q6 y [y, r, q6]
        " " [" ", l, qf];

-qf;

`
]
];

var cWindow = document.getElementById('consoleText');

//display list of example programs
document.getElementById('select').addEventListener('click', (event) => {
        let list = document.getElementById('selection_list');
        if(list.style.display === 'none')
        {
                list.style.display = 'block';
        } else {
                list.style.display = 'none';
```

```
        }
});

//check to see if user loses focus in selection window
document.getElementById('selection_list').addEventListener('mouseleave', function(e) {
        var container = document.getElementById('selection_list');
        container.style.display = 'none';
});

//retrieve clicked program option and load related program
document.querySelectorAll('#selection_option').forEach(item => {
        item.addEventListener('click', event =>{
                console.log(item.innerHTML);
                let list = document.getElementById('selection_list');
                let selectedProgram = item.innerHTML;
                loadSelectedProgram(selectedProgram);
                list.style.display = 'none';
        })
});

//find program associated with passed name
function loadSelectedProgram(selectedProgram)
{
        document.getElementById('code-area').value = '';

        let pName = selectedProgram;
        let rProgram;

        for(let i=0;i<programs.length;i++)
        {
                if(programs[i][0] === pName)
                {
                        rProgram = programs[i][1];
                }
        }

        document.getElementById('code-area').value += rProgram;
        cWindow.value = `${pName} successfully loaded.`;
}

//upload a file via file button
document.getElementById('upload').addEventListener('change', function() {
        document.getElementById('code-area').value = '';

        let fr = new FileReader();
        fr.onload = function(){
```

```
                document.getElementById('code-area').value += fr.result;
        }

        fr.readAsText(this.files[0]);
});

//save machine from website via text document.
document.getElementById('save').addEventListener('click', function() {
        let fileName = prompt('Enter the name you wish to save the machine as: ',
'myMachine.txt');
        let abort = false;

        if (fileName == null || fileName == '') {
                cWindow.value = 'File name was left empty, aborting file save.';
                abort = true;
        } else {
                cWindow.value = `File name: ${fileName} was chosen.\n`;
        }

        if(!abort)
        {
                let machineText = document.getElementById('code-area').value;
                let blob = new Blob([machineText], {type:'text/plain'});
                let link = document.createElement('a');

                link.download = fileName;
                link.href = window.URL.createObjectURL(blob);

                document.body.appendChild(link);
                link.click();

                cWindow.value += `File ${fileName} successfully created.`;

                setTimeout(() => {
                        document.body.removeChild(link);
                        window.URL.revokeObjectURL(link.href);
                }, 100);
        }
});
```

mapper.js

```
//define all existing cells
var cell = document.querySelectorAll('.tape .tape-cell');

//get value from cell in main tape under read head
function getCellUnderHead()
{
	return cell[findBeginning() + 5].querySelector('.cell-contents').innerHTML;
}

//apply transition result to cell under read head
function updateCell(result)
{
	let cellUnderHead = findBeginning() + 5;
	cell[cellUnderHead].querySelector('.cell-contents').innerHTML = result;
}

//reset tapes to default states
function updateTape(updatedTape, blankChar)
{
	//reset cells to default positions
	resetPos();
	//clear contents from cells add
	for(let i=0;i<cell.length;++i)
	{
		cell[i].querySelector('.cell-contents').innerHTML = blankChar;
	}
	let tapeBeginning = findBeginning();
	for(let i=tapeBeginning+5;i<cell.length;++i)
	{
		if(updatedTape[i-(tapeBeginning+5)] != undefined)
		{
			cell[i].querySelector('.cell-contents').innerHTML = updatedTape[i-
(tapeBeginning+5)];
		}
	}
}
```

about.html

```
<!--about page-->
<!DOCTYPE html>
<html>
<head>
	<title>Virtualization of a Turing Machine - About</title>
	<link rel="stylesheet" href="css/style.css"/>
</head>
<body>
	<div class="wrapper">
		<div class="main-wrapper">
			<h2>Virtualization of a Turing Machine</h2>
			<span class="navigation">
				<ul>
					<a href="index.html"><li>Home</li></a>
					<a href="about.html"><li>About</li></a>
					<a href="weekly_updates.html"><li>Weekly
Updates</li></a>
					<a
href="documentation.html"><li>Documentation</li></a>
					<a href="schedule.html"><li>Schedule</li></a>
				</ul>
			</span>
			<div class="main">
			<center>
				<i><h2>What is this place?</h2></i>
				<div class="about-description">
					This website's main function is a visualization tool
for teaching the concepts of a Turing machine and aid professors in explaining the
fundementals of how a Turing machine works. It also serves as our semester-long
assignment for CSC-492 at California University of Pennsylvania for the following
individuals mentioned below.
					<br>
					<br>
					For more information on this website and how to
use it, please refer to the documentation tab for a user manual.
					<br>
					<br>
					Workings Involved:
					HTML, CSS, JavaScript.
				</div>
			</center>
			<div class="profiles-wrapper">
				<center>
					<i><h2>Who's involved?</h2></i>
```

```
                                </center>
                                <div class="profile">
                                        <h3>Brett George</h3>
                                </div>
                                <div class="profile" style="margin-left: 2%; margin-right:
2%">
                                        <h3>Jisue Lee</h3>
                                </div>
                                <div class="profile">
                                        <h3>James Merenda</h3>
                                </div>
                        </div>


                        </div>
                        <div class="bottom-bar">
                                <div style="float: left; margin-left:20px;margin-
top:7px;">Authored in Cooperation by: Brett George, Jisue Lee, James Merenda</div>
                                <div style="float: right; margin-right:20px;margin-
top:7px;">Last Update: Apr-2021</div>
                        </div>
                </div>
        </div>
</body>

</html>
```

documentation.html

```
<!--documentation page-->
<!DOCTYPE html>
<html>
<head>
	<title>Virtualization of a Turing Machine - Documentation</title>
	<link rel="stylesheet" href="css/style.css"/>
</head>
<body>
	<div class="wrapper">
		<div class="main-wrapper">
			<h2>Virtualization of a Turing Machine</h2>
			<span class="navigation">
				<ul>
					<a href="index.html"><li>Home</li></a>
					<a href="about.html"><li>About</li></a>
					<a href="weekly_updates.html"><li>Weekly
Updates</li></a>
					<a
href="documentation.html"><li>Documentation</li></a>
					<a href="schedule.html"><li>Schedule</li></a>
				</ul>
			</span>
			<div class="main">
				<center><h2>Project Documents</h2></center>
				<div class="documentation">
					<a href="docs/virtualization of a turing machine -
requirements document.pdf">View Requirements Document</a>
					<a href="docs/visualization of a turing machine -
specification document.pdf" style="margin-left: 2%; margin-right: 2%">View
Specifications Document</a>
					<a href="docs/visualization of a turing machine -
design document.pdf">View Design Document</a>
					<a href="docs/visualization of a turing machine -
user manual.pdf">View User Manual</a>
				</div>
			</div>
			<div class="bottom-bar">
				<div style="float: left; margin-left:20px;margin-
top:7px;">Authored in Cooperation by: Brett George, Jisue Lee, James Merenda</div>
				<div style="float: right; margin-right:20px;margin-
top:7px;">Last Update: Apr-2021</div>
			</div>
		</div>
	</div>
```

```
</body>

</html>
```

index.html

```
<!--index page-->
<!DOCTYPE html>
<html>
<head>
	<title>Virtualization of a Turing Machine - Home</title>
	<link rel="stylesheet" href="css/style.css"/>
	<link rel="stylesheet" href="css/machine.css"/>
</head>
<body>
	<div class="wrapper">
		<div class="main-wrapper">
			<h2 style='user-select: none;'>Virtualization of a Turing
Machine</h2>
			<span class="navigation">
				<ul>
					<a href="index.html"><li>Home</li></a>
					<a href="about.html"><li>About</li></a>
					<a href="weekly_updates.html"><li>Weekly
Updates</li></a>
					<a
href="documentation.html"><li>Documentation</li></a>
					<a href="schedule.html"><li>Schedule</li></a>
				</ul>
			</span>
			<div class="main">
				<div class='options'>
					<!--<button class='option_buttons'
id='toggle_layout'>Console Position</button>-->
					<button class='option_buttons'
id='toggle_console'>Toggle Console</button>
					<button class='option_buttons'
id='toggle_editor'>Toggle Editor</button>
					<button class='option_buttons'
id='toggle_machine'>Toggle Machine</button>
				</div>
				<div  class = 'editor' id='editor'>
					<div class ='editor__buttons'>
						<button class='editor__buttons'
id='load'>Compile Machine</button>
						<button class='editor__buttons'
id='edit_current_machine'>Edit Machine</button>
						<button class='editor__buttons' type='button'
id='save'>Save Machine</button>
```

```
                                                <button class='editor__buttons'
id='select'>Select Pre-Made Machine</button>
                                                <label class='upload__button'
for="upload">Upload Machine</label>
                                                <input type="file" id='upload'
accept='.txt'></input>
                                                <div class='info_button' id='help-code-
editor'>
                                                        &#x1F6C8;
                                                </div>
                                                <div class='help-box' id='help-code-editor-
message' style='display: none;'>
                                                        <h3><i>What is this?</i></h3>
                                                        <hr>
                                                        This is the <i>Transition Editor</i>.
<br><br>This field is responsible for converting transitions into information the machine
can process.<br><br>To make a machine, there is certain syntax that must be present for
the Turing Machine to function:
                                                        <ul>
                                                                <li><i><a class='help-link-
behavior' href='#help-legal-symbols'>Define legal symbols.</i></a></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-blank-characters'>Define blank characters.</i></a></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-states'>Define a start and accept state.</a></i></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-transitions'>Define transitions.</a></i></li>
                                                        </ul>
                                                        There are also a selection of example
programs to choose from by selecting the <i>Select Pre-Made Machine</i> button. For
explanations on how these machine's work. Click the relevant name of the machine
below.<br>
                                                        <ul>
                                                                <li><i><a class='help-link-
behavior' href='#help-binary-palindrome'>Binary Palindrome</a></i></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-binary-pong'>Binary Pong</a></i></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-the-classic'>The Classic</a></i></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-binary-increment'>Binary Increment</a></i></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-binary-addition'>Binary Addition</a></i></li>
                                                                <li><i><a class='help-link-
behavior' href='#help-binary-complement'>Binary Complement</a></i></li>
                                                        </ul>
```

```
                                                    <hr>
                                                    <div id='help-legal-
symbols'><i><b>Legal Symbols</b></i></div><br>
                                                        Legal symbols are defined by
the <i>**</i> phrase. It is an extra security during compilation so foreign symbols are
identified and alerted to the user when an illegal input symbol or transition symbol is
detected.<br><br>The syntax is written as shown below:<br><br>
                                                        <i>** = "symbols";</i><br>
                                                    <hr>
                                                    <div id='help-blank-
characters'><i><b>Blank Characters</b></i></div><br>
                                                        Blank characters are defined
with the <i>blank</i> phrase. It is a value used to fill each cell in the Turing Machine
tape with the user-defined blank character.<br><br>The syntax is written as shown
below:<br><br><i>blank = "blankSymbol";</i><br>
                                                    <hr>
                                                    <div id='help-states'><i><b>Start
and Accept States</b></i></div><br>
                                                        The start and accept states are
transitions that begin and end the Turing machine respectively. These states may be
named anything the user defines.<br><br>The syntax is written as shown
below:<br><br><i>start = startName;<br>accept = acceptName;</i><br>
                                                    <hr>
                                                    <div id='help-
transitions'><i><b>Transitions</b></i></div><br>
                                                        Transitions are moves the
Turing machine makes based on the current state and input read from the tape. The user
may define their own transitions in the code editor.<br><br>The syntax is written as
shown below:<br><br><i>-name readSymbol [symbolToWrite, direction,
nextState];</i><br><br>
                                                        Multiple transition definitions
may be assigned to the same state as well like the following:<br><br>-name readSymbol
<br>[symbolToWriteA, direction, nextState]<br>[symbolToWriteB, direction,
nextState];
                                                    <hr>
                                                    <div id='help-binary-
palindrome'><i><b>Binary Palindrome</b></i></div><br>
                                                        The Binary Palindrome
machine
                                                    <hr>
                                                    <div id='help-binary-
pong'><i><b>Binary Pong</b></i></div><br>
                                                        The Binary Pong machine
                                                    <hr>
                                                    <div id='help-the-
classic'><i><b>The Classic</b></i></div><br>
```

```
                                                    The Classic machine is an
example Turing machine proposed by Alan Turing himself as a demonstration of how
these sorts of machines may operate.
                                                    <hr>
                                                    <div id='help-binary-
increment'><i><b>Binary Increment</b></i></div><br>
                                                    The Binary Increment
machine accepts a string of binary and increments its value by 1.
                                                    <hr>
                                                    <div id='help-binary-
addition'><i><b>Binary Addition</b></i></div><br>
                                                    The Binary Addition machine
takes two binary numbers separated by a '+' symbol and sums them together.
                                                    <hr>
                                                    <div id='help-binary-
complement'><i><b>Binary Complement</b></i></div><br>
                                                    The Binary Complement
machine accepts any binary number and produces its complement.
                                                    <hr>
                                                    <br>
                                                    For more information regarding the
Transition Editor and other functionality, refer to the documentation tab for a user's
manual.
                                                </div>
                                            </div>
                                            <ul id='selection_list' style='display: none;'>
                                                <lh>Language Acceptor</lh>
                                                <li id='selection_option'>Palindrome</li>
                                                <li id='selection_option'>Binary Pong</li>
                                                <li id='selection_option'>The Classic</li>
                                                <li id='selection_option'>wcw, where w is
an element of {a,b}*</li>
                                                <lh>Language Transducer</lh>
                                                <li id='selection_option'>Binary
Increment</li>
                                                <li id='selection_option'>Binary
Addition</li>
                                                <li id='selection_option'>Binary
Complement</li>
                                                <li id='selection_option'>x-y, where x is
greater than y</li>
                                            </ul>
                                            <div class ='editor__text'>
                                                <textarea class='editor__text__line-numbers'
id='line-numbers' placeholder='1' readonly="true"></textarea>
```

```
                                        <textarea class='editor__text__code-area'
id='code-area'spellcheck="false"></textarea>
                                    </div>
                                </div>
                                <div class = 'console' id= 'console'>
                                    <div class='info_button' id='help-console'>
                                        &#x1F6C8;
                                    </div>
                                    <div class='help-box' id='help-console-message'
style='display: none;'>
                                        <h3><i>What is this?</i></h3>
                                        <hr>
                                        This is the <i>Console</i>.<br><br>
                                        Information related to compilation and the
Turing Machine animation will be displayed here.
                                    </div>
                                    <div class ='console__buttons'>
                                        <button id='clearConsole'>Clear
Console</button>
                                    </div>
                                    <div class ='console__display'>
                                        <textarea class='console__display'
id='consoleText'readonly="true"></textarea>
                                    </div>
                                </div>
                                <div class='machine' id='machine'>
                                    <div class='tape_view'>
                                        <div class="t-machine">
                                            <div class='info_button' id='help-
machine'>
                                                &#x1F6C8;
                                            </div>
                                            <div class='help-box' id='help-
machine-message' style='display: none;'>
                                                <h3><i>What is
this?</i></h3>
                                                <hr>
                                                This is the <i>Turing
Machine Simulation</i>.<br><br>
                                                Once an acceptable machine
is compiled, transitions are performed automatically by pressing <i>Play</i> or stepped
sequentially by pressing <i>Step</i>. While playing automatically, pressing
<i>Pause</i> will pause the machine during a transition.
                                            </div>
                                            <div class="read-head-
indicator"></div>
```

```
                                                    <div class="read-head"></div>
                                                    <div class="tape">
                                                        <div id="0" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="1" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="2" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="3" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="4" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="5" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="6" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="7" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="8" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="9" class="tape-cell"
style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="10" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="11" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="12" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="13" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
```

```
<div id="14" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="15" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="16" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="17" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="18" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="19" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="20" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="21" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="22" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="23" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="24" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="25" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="26" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="27" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="28" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
```

```
<div id="29" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="30" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="31" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="32" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="33" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="34" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="35" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="36" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="37" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="38" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="39" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="40" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="41" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="42" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
<div id="43" class="tape-cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-contents"></div></div>
```

```
                                                    <div id="44" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="45" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="46" class="tape-
cell" style="left: -11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="47" class="tape-
cell" style="left: 0.00%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="48" class="tape-
cell" style="left: 11.11%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="49" class="tape-
cell" style="left: 22.22%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="50" class="tape-
cell" style="left: 33.33%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                        <div id="51"
class="tape-cell" style="left: 44.44%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="52" class="tape-
cell" style="left: 55.55%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="53" class="tape-
cell" style="left: 66.66%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="54" class="tape-
cell" style="left: 77.77%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="55" class="tape-
cell" style="left: 88.88%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="56" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="57" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="58" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
```

```
<div id="59" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="60" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="61" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="62" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="63" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="64" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="65" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="66" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="67" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="68" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="69" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="70" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="71" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="72" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
<div id="73" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
```

```
                                             <div id="74" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="75" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="76" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="77" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="78" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="79" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="80" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="81" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="82" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="83" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="84" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="85" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="86" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="87" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                             <div id="88" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
```

```
                                                    <div id="89" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="90" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="91" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="92" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="93" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="94" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="95" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="96" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="97" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="98" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="99" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                    <div id="100" class="tape-
cell" style="left: 99.99%; transition-duration: 0.35s;"><div class="cell-
contents"></div></div>
                                                </div>
                                                <div class="speedSelectorBounds">
                                                    <div
id="slowLabel">Slower</div>
                                                        <input
id="speedSelector" onload="setDefaultSpeed()" type="range" min="0" max="70"
value="35"/>
                                                    <div
id="fastLabel">Faster</div>
```

```
                                                    <div
id="currentSpeedLabel">Transition Duration:
                                                    </div>
                                                    <div id="currentSpeed">0.35
s</div>
                                                </div>
                                            </div>
                                            <center>
                                                <div class='machine_buttons'>
                                                    <button
class='machine_buttons' id='step'>Step</button>
                                                    <button
class='machine_buttons' id='play'>Play</button>
                                                    <button
class='machine_buttons' id='pause'>Pause</button>
                                                    <button
class='machine_buttons' id='reset'>Reset</button>
                                                    <button
class='machine_buttons' onclick='moveRight()'>Move Right</button>
                                                    <button
class='machine_buttons' onclick='moveLeft()'>Move Left</button>
                                                </div>
                                            </center>
                                        </div>
                                    </div>
                                </div>
                                <div class="bottom-bar">
                                    <div style="float: left; margin-left:20px;margin-
top:7px;">Authored in Cooperation by: Brett George, Jisue Lee, James Merenda</div>
                                    <div style="float: right; margin-right:20px;margin-
top:7px;">Last Update: Apr-2021</div>
                                </div>
                            </div>
                        </div>
                        <script src="scripts/app.js" type="module"></script>
                        <script type="text/javascript" src="scripts/animation.js"></script>
                        <script type="text/javascript" src="scripts/mapper.js"></script>
                        <script type="text/javascript" src="scripts/manipulatePrograms.js"></script>
                        <script type="text/javascript" src="scripts/help.js"></script>
</body>

</html>
```

schedule.html

```
<!--schedule page-->
<!DOCTYPE html>
<html>
<head>
	<title>Virtualization of a Turing Machine - Home</title>
	<link rel="stylesheet" href="css/style.css"/>
</head>
<body>
	<div class="wrapper">
		<div class="main-wrapper">
			<h2>Virtualization of a Turing Machine</h2>
			<span class="navigation">
				<ul>
					<a href="index.html"><li>Home</li></a>
					<a href="about.html"><li>About</li></a>
					<a href="weekly_updates.html"><li>Weekly
Updates</li></a>
					<a
href="documentation.html"><li>Documentation</li></a>
					<a href="schedule.html"><li>Schedule</li></a>
				</ul>
			</span>
			<div class="main">
				<div id="gantt-chart">
					<iframe
src="https://docs.google.com/spreadsheets/d/e/2PACX-
1vQtfhrn04jDiBUxEN7b4hJ0z2_0z4HT9AimyeGHGNHfMEx3EH0mFEVtsHQ43fglH
DGVo1OF3_TntU03/pubhtml?gid=1959293812&amp;single=true&amp;widget=true&a
mp;headers=false"></iframe>
					<center><a
href="https://docs.google.com/spreadsheets/d/1EQWwdWq2BfARVs8DkjIhUrHznQLN
YlUAlYkkYqjR00c/edit?usp=sharing">Download Here</a></center>
				</div>
			</div>
			<div class="bottom-bar">
				<div style="float: left; margin-left:20px;margin-
top:7px;">Authored in Cooperation by: Brett George, Jisue Lee, James Merenda</div>
				<div style="float: right; margin-right:20px;margin-
top:7px;">Last Update: Apr-2021</div>
			</div>
		</div>
	</div>
</body>
```

```
</html>
```

weekly_updates.html

```
<!--weekly_updates page-->
<!DOCTYPE html>
<html>
<head>
	<title>Virtualization of a Turing Machine - Weekly Updates</title>
	<link rel="stylesheet" href="css/style.css"/>
</head>
<body>
	<div class="wrapper">
		<div class="main-wrapper">
			<h2>Virtualization of a Turing Machine</h2>
			<span class="navigation">
				<ul>
					<a href="index.html"><li>Home</li></a>
					<a href="about.html"><li>About</li></a>
					<a href="weekly_updates.html"><li>Weekly
Updates</li></a>
					<a
href="documentation.html"><li>Documentation</li></a>
					<a href="schedule.html"><li>Schedule</li></a>
				</ul>
			</span>
			<div class="main" style="overflow-y: scroll">
				<!--<center><h1>To Be Updated</h1></center>-->
				<div id="reports-container">
					<ul>
						<li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vTJR6ye0gELsrAcGinC1rG3ijVQQM4yQYRxhr1iy7PqELSvZHvdSPrwvYB5P4KDig
IewRqMf9S6iZj6/embed?start=false&loop=true&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
						<li><a
href="https://docs.google.com/presentation/d/1mx5FWwB1IT-M6CT-ldm7CJerR20V90-
xvRiSXPW5bzE/edit?usp=sharing">Download This Slideshow</a></li>
						<li><a href="docs/weekly report 1-31-
2021.pdf">View Weekly Report 1-31-2021</a></li>
					</ul>
				</div>
				<div id="reports-container">
					<ul>
						<li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vSqECRRk6UeCUQiwjJhnd3FM9iMWUcEEnEHd_SHx2rcyljzYyquL_U93v3vTmUg-
```

```
P72s85DltJw92av/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1DeUZsob1jJ02oZBsHpbvXR5wB3I50Lp
UKkoQ7_lWGlg/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 2-7-
2021.pdf">View Weekly Report 2-7-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vRI9PB8OKqBGTV27Uj_z03ACDy0-bb51Gj9snQEqwSUE_B3rcI7-tU-
Vc_wqXOl2d7uTB-pF0DSBSxQ/embed?start=false&loop=false&delayms=3000"
frameborder="0" width="350" height="215" allowfullscreen="true"
mozallowfullscreen="true" webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1vNKLcgjj7Y9uE1MtIKojg3BHUQroyN-
alJrWKUrk_h0/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 2-14-
2021.pdf">View Weekly Report 2-14-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vQuKnVrxv1VeiMRaJsMddifCZfnKuNAstjB_XmUkAZALg5RRstBtSuWemOotqGyJ
jO_sMPZ7fe57KwU/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1Ffplq8S087-c-
puitlLx7O3WVo4avNDMnHR81nVKojg/edit?usp=sharing">Download This
Slideshow</a></li>
                                        <li><a href="docs/weekly report 2-21-
2021.pdf">View Weekly Report 2-21-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vQvT57bQZyjpbdWbNhLjrRRXJMg4g8qJOnfp7x33cTwWMJaxI_jpWGeEfsUXF-
```

```
YyGDtrtf0qJ0N_3TN/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1nQARbIOPhi2heNaXlUX96v6Ef_RDVB
d6X83BYlAawjY/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 2-28-
2021.pdf">View Weekly Report 2-28-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vTT5d3gjH6aEjiSfBmkd88EEkGxxcKj4dOlhJUdQLwmJv0lErdVDrtMrbAhybDX76jo
-KtO8NoTaWht/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1hhSFpMAxZHY2J3SbCbeQXAk5vnf9Fy
RGtnyxEDqSKyY/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 3-7-
2021.pdf">View Weekly Report 3-7-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vTE8YemUB4TqZq3n58GVPCu86eExo_6D_sdCQadl6p3zaAsiPDRhnjbrqkgj48JPcT0
Y6zl3WmzkKSX/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1_yo1gABDf1hZBUdGtqClotgQSbNdHrP
n59k7yFlldjE/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 3-14-
2021.pdf">View Weekly Report 3-14-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vRHn24JXZo1d5b3zXWnaQXWAK10Taee0vH3KAO-
7dmmQc9uRMbbgvmNqJnaMahoViH26vJQKtalli_z/embed?start=false&loop=false&de
```

```
layms=3000" frameborder="0" width="350" height="215" allowfullscreen="true"
mozallowfullscreen="true" webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1YS1RS9Uyzvr63a2ptaAnVQ1lLhJ5RT2q
E6pczOx2P9c/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 3-21-
2021.pdf">View Weekly Report 3-21-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-1vQgxPGxgUbl-
Pmns7VqUeWctCy5lpLhh7i_ce7KhLUWIhD_MhTX-
0OP_RwQ814j0sFysN91wB1jYa1_/embed?start=false&loop=false&delayms=3000"
frameborder="0" width="350" height="215" allowfullscreen="true"
mozallowfullscreen="true" webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1co6VSxGb0WzkC00HciSW-
AylR9A3nRBzkY_KCTTe5cI/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 3-28-
2021.pdf">View Weekly Report 3-28-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vRiwBXcL6E6wCHmLeqfCdq5bwXCq9-
C8HmViDKqMOGRyudcQdHiC7wsbAWlkOmWTcuo-
8HOnXzl_7tp/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                        <li><a
href="https://docs.google.com/presentation/d/1YZWwWjrFNl1AgYhEKtXpF3zxx1sfvM
JD2tPYE0V2MBA/edit?usp=sharing">Download This Slideshow</a></li>
                                        <li><a href="docs/weekly report 4-4-
2021.pdf">View Weekly Report 4-4-2021</a></li>
                                    </ul>
                                </div>
                                <div id="reports-container">
                                    <ul>
                                        <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vTkuZUR78q03Zs0opMUvtJrbs6ZxEUXfe1AXGB9PP_FPYqPEa4xKpUxA4MeJOQ
wkS7HvI0PnBVo2p0Y/embed?start=false&loop=false&delayms=3000"
```

```
frameborder="0" width="350" height="215" allowfullscreen="true"
mozallowfullscreen="true" webkitallowfullscreen="true"></iframe></center></li>
                                                <li><a
href="https://docs.google.com/presentation/d/1BqvpExddvGhmMzGPO2JkJMQc9dtRIF
D6O6ggN_Uz2FI/edit?usp=sharing">Download This Slideshow</a></li>
                                                <li><a href="docs/weekly report 4-11-
2021.pdf">View Weekly Report 4-11-2021</a></li>
                                            </ul>
                                        </div>
                                        <div id="reports-container">
                                            <ul>
                                                <li><center><iframe
src="https://docs.google.com/presentation/d/e/2PACX-
1vTmCNMRjvIFasyZ3gU8oMXs6JNje4iojrYqis5lMuNX9fr6O4OsdRTC4I5wMhkIVR
BfZaLTzDp-bb4X/embed?start=false&loop=false&delayms=3000" frameborder="0"
width="350" height="215" allowfullscreen="true" mozallowfullscreen="true"
webkitallowfullscreen="true"></iframe></center></li>
                                                <li><a
href="https://docs.google.com/presentation/d/12in4dYEenNoZYrfbn0YbnDu-
v45jC3MN0nYmZTeL3yM/edit?usp=sharing">Download This Slideshow</a></li>
                                                <li><a href="docs/weekly report 4-18-
2021.pdf">View Weekly Report 4-18-2021</a></li>
                                            </ul>
                                        </div>
                                    </div>
                                    <div class="bottom-bar">
                                        <div style="float: left; margin-left:20px;margin-
top:7px;">Authored in Cooperation by: Brett George, Jisue Lee, James Merenda</div>
                                        <div style="float: right; margin-right:20px;margin-
top:7px;">Last Update: Apr-2021</div>
                                    </div>
                                </div>
                            </div>
                        </body>

                        </html>
```

Appendix: Workflow Authentication

I, Brett George, hereby attest that I have done the work documented herein.

Signature: _____

Date: _____

I, Ji Sue Lee, hereby attest that I have done the work documented herein.

Signature: _____

Date: _____

I, James Merenda, hereby attest that I have done the work documented herein.

Signature: _____

Date: _____