MATH TOOLS PROJECT

Florian Brandsma
Oleksandr Gudenko
Oulu University of Applied Sciences

CONTENTS

1 INTRODUCTION

This report details our work done on the Mathematic Tools Project, that we are currently working on to implement everything we have learned over the course of two periods. The project is made per request of Oulu University of Applied Sciences, as part of our second period.

1.1 The assignment

Our task was to make a web based mathematics tool, which offers a range of different math functionalities for the user. We were meant to make the following tools:

1. Number system conversions
    - A tool for numbering system conversions, containing at least the four numbering systems: binary, octal, decimal and hexadecimal.

2. Number system outputs
    - Display a table showing decimals 0-50 in binary, octal and hexadecimal numbering systems.

3. Combinations
    - A simple tool for combinatorics, which calculates either combinations or permutations, with the option to choose a sampling.

4. Truth tables
    - A basic set of truth tables for basic operations.

5. Random values
    - A tool to test random number distributions between a given range of numbers.

6. Free choice tool
    - A tool working with a mathematical discipline of our choice.

1.2 Requirements

The project was to be made within three weeks, with the deadline being December 18th, 2017. Additional requirements were to make use of GitHub, to conveniently share and backup our work.

2 THE WORK ENVIRONMENT

Most of our work was done within the school campus. The tools that we used were as following:

1. Microsoft Office
    - Office was used to create a planning and keep track of our work using a timetable.
2. Balsamiq
    - This tool was used to create mockups of our website.
3. Git/GitHub
    - We used the Git environment to collaborate on our work.
4. Visual Studio Code
    - This IDE is used to edit our JavaScript, HTML and CSS code for the website.
5. Browsers
    - Google Chrome and Mozilla Firefox were used to compile our code.

3 DEFINITIONS

Our task was to make a web based mathematics tool, which offers a range of different math functionalities for the user.

3.1 Website

The website will be clear to understand and easy to use, packaged in a simple and modern layout. The individual tools will have their own page, as not to overwhelm the user.
The navigation of the website remains the same in all cases, so that only the tools themselves are swapped out.

3.2 Mathematic tools

Below is a description on how the tools will function:

1. Number system conversions
    - The user can convert the four numbering systems by using a two-way input. They can freely select different numbering systems to convert to and from

2. Number system outputs
    - Display a table showing decimals in 0-50 and their value in the other numbering systems. The table is scrollable.

3. Combinatorics
    - We implement combinatorics by way of a license plate generator.

4. Truth tables
    - The user can input a sum to calculate by selecting variables and operators from a series of dropdown menus. The result will be displayed in a truth table.

5. Random values
    - Allow the user to freely input a range of numbers to generate a random number from. The generated number is displayed on the screen, as well as added to a table, in descending order. Finally, duplicate results are added on top of each other and display the amount of times of this happening.

6. Temperature converter
    - Much like the number system converter, the user can use this tool to convert any temperature from and to Celsius, Kelvin and Fahrenheit. The results will also be displayed in a slider, mimicking the effect of a thermometer.

4 IMPLEMENTATION

4.1 Planning

Before diving straight into our workload, we made a planning detailing all the functions we were going to make for our project. This was accompanied with a rough time estimation, making sure we would be able to complete everything in time.

4.2 Designing

To agree on the design, we were going to implement into our project, we decided to each make several mockups of the same webpages in Balsamiq. After putting our own interpretations on paper, we showed our results to several teachers to get feedback and guidance on how to proceed. The result has been a combination of both our visions, picking the best features.

The overall feel of the website should be self-explanatory. We believe the user shouldn't need to read instructions before having to use a tool, but jump right into what they came to do.

We decided on using a simple navigation system, that remains the same throughout the pages. To save space on each page, the header is presented by highlighting the selected option.

The space saved is precious, as we don't want the user to have to scroll down to use the features. The entirety of the tools should be presented immediately.

The tools themselves must be easy to use as well, requiring a minimal amount of input from the user, while still granting as much freedom as possible.

4.3 Implementing

We divided the workload so that JavaScript functionality was mainly done by Florian and the layout was done by Oleksandr, using HTML and CSS.

By the time of writing this report, not all features have been fully completed yet. The progress on these features will be described later.

Below we each describe what we did for each function.

- General features

  The first challenge was deciding where to begin. For this we developed the front-page prototype and a common.css file for the general styling. Everything else was built on this foundation. This became the first layer of our website, on which we continued building. The navigation menu was developed here and the common.css includes all the main customizations and modifications which is used across the entire

website, to promote consistency across the board. This includes button design, shading, font style, page size, spacing blocks, color schemes etc.
Common.css was updated throughout the development process, whenever new features were required.
For each page, separate centered input classes were created. Additionally, divs were used to properly space the element's x and y dimensions.

- Math tools
  1. Number system conversions
     a. JavaScript

        The number system converter was first on the list, so it felt only natural to start with that. The core feature of the tool is that it can convert a number from one number system to another. This could be done by simply allowing the user to only input a decimal and convert that to the desired number system, but that felt too easy.

        I started working on the conversions myself, rather than using the built-in tools in JavaScript to do so. We had over two weeks left, so there was plenty of time to experiment and mess around.

        First, I had to determine how to make the calculations, when you want to convert from binary to hexadecimal, for example. The easiest way to do this was to first convert the input value to decimal, by using the base of the currently selected numbering system. From here I could freely convert the result of that output to determine the user's desired result.

        Our converter works so that the user can use a two-way input. This means that the code needs to check where the input has come from so that it places the result in the right "output" field.

        Calculations are also done automatically when switching between number systems in the selection fields, so the correct result is always displayed.

b. HTML/CSS

The challenge was to make the layout symmetric, usable and clear for the user. This was possible to achieve with the help of a table, creating cells and placing them in the right position.

The second problem that I encountered were the dropdown lists. The customization options were limited, and it was not enough.

The solution came in the form of open lists, which highlights selected options.

2. Number system outputs

a. JavaScript

Followed up by the number system converter was the number system output, where we were to display a table, showing the decimal value 0-50 in other numbering systems. The conversion function I had made for the previous tool could be used for this one as well, with minor alterations.

One option was to “hardcode” the entire table. But I certainly didn’t feel like doing that, especially when I could automate the whole process.

Using the previous code, I simply cut out the part where it converts to decimal. Instead I used a for loop and pushed the index through the other conversion function.

For each time this was done, I added a new row to the table with as many cells as given number systems. It’s possible to remove number systems, or make up entirely new ones. The table has no problem displaying this.

The assignment specified the need for a “clear” button. We didn’t understand why, as the desired output was of a fixed size. I reached out to the teacher, with the result being that this button was only necessary if the size of the table otherwise interferes with the rest of the layout. In our case, it does not, as it’s contained within a slider-field.

b. HTML/CSS

   Two tables were created for different purposes. This decision helped with alignment and implementation as this combined table has the desired look with all the functionality it needed.

   Table alignment was an issue I had difficulties with. Thanks to the teacher's recommendations of adding an id tag to table cells I was able to align everything as I wanted.

3. Combinatorics (License plate generator)
   a. JavaScript

      No JavaScript functions have been made for this tool yet.

   b. HTML/CSS

      The table solution was implemented to position each element. Buttons are highlighted by shadows and a hover effect. I encountered the same issue with dropdown menus as in the number system converter, but this time I knew how to handle it.

      After trying, I found out that open lists suited the page's layout and kept them.

      Now the blue element; this time the challenge was unexpected. The blue element is present on each page of the website, but there was no good place to put it here. After trying different suggestions, we agreed on making the blue element a part of the page's title.

4. Truth table
   a. JavaScript

      The truth table is a challenge which I'm currently tackling. The idea was to let the user freely choose their own calculations, and a generated table would update accordingly. These choices are listed in several dropdown menus; as many as there are variables (such as p, q, r), with operators in between. To not make it too complicated, we resorted to only using logical "and" and "or" operators.

The first challenge was to list the standard truth table. I had searched far and wide on the web to find a possible solution to this problem, but to no avail. After a while I noticed a pattern.
The first variable sequence always switched from false to true, after half of the table's total length, which of itself was determined by calculating the number of constants to the power of the variables. In case of p, q, r and true, false: $2^3 = 8$.
Thus, in the first sequence, the Boolean value changed after ($2^3/2=$) 4.
In the next row, the sequence switched twice as often. This process was automated by adjusting the calculation, to decrease the outcome. The next calculation would be ($2^2/2=$)2, with the last always being (2/2=)1.
This scales to any size, even if the list of variables includes the entire alphabet. For our purposes, we'll stick to p, q and r.

Now, we must put the table to work. With the calculated constant values in place, this task is not such a difficult one. However, we don't typically choose the easy way out, not when we still have plenty of time before the deadline. So, like most of our tools, the user decides the calculation.

As mentioned before, the clearest way to do this is by using dropdown menus to select the desired variables and operators. Just like the table, these are made to fit the size of the given number of variables.
In the case of p, q and r, there will be three variable dropdowns, split by two operator dropdowns.
Whenever you change your selection, the table will update accordingly.

b. HTML/CSS
   Work in progress

5. Random values
    a. JavaScript

        Probably the simplest feature, the random value tool did not take too long to implement. While we had decided to list our random outcomes in a table, the how of it wasn't yet decided on.

        The user always chooses the random range themselves. My idea was to create a table the size of the given range, displaying all possible numbers immediately. Whenever you generate a number, the value displaying the times generated would increment.

        My partner vouched against this, stressing that it might be difficult for the user to find their generated number in the list. Instead, we should list the numbers in the order which they came in.

        We reached a compromise; we'll add newly generated numbers to the table as they come, but sort them from lowest to highest.

        This made the implementation somewhat trickier, but worthwhile none the less.
    b. HTML/CSS

        Users will be able to select a range and generate numbers as many times as they want. Shadows and hover effects highlight buttons, which makes them clear to distinguish from other elements. Buttons also give feedback when pressed, by performing an animation of going down, as if it's being pressed physically. It was difficult to achieve the desired symmetry, but in the end, it turned out alright. In my opinion, this is the best designed page layout of the project.
6. Free choice (Temperature converter)
    a. JavaScript

        No JavaScript functions have been made for this tool yet.
    b. HTML/CSS

        The challenge was to make a functional layout for a converter while keeping the overall design consistent. After alignments and visuals were done, the next challenge was consistency. *Lightskyblue* color is a part of our overall design and is present on every page, but there was no obvious place to add it.

A great solution came in the form of highlighting the chosen conversion option by changing its background color when it’s selected. This way, *lightskyblue* will be present and the consistency will remain intact.

5 TESTING

Extensive tests have been performed throughout the development of the project. Before moving on to the next step, we made sure that everything works as intended. Even after moving on, we go back to see if nothing broke.

At times we found parts that could be improved on, to improve the user experience, we went back and implemented these features.

6 POSSIBILITIES OF FURTHER DEVELOPMENT

With a few days to go before the deadline, some work remains. The implementation of JavaScript functions in the temperature converter, license plate generator and the finalization of the truth table. As for the project itself; it’s simply a school project and will not see further development after the deadline has been reached.
That being said, the website has been coded to be very flexible when it comes to free choice of input sizes.

7 CONCLUSION

Our co-operation on the project has been a pleasant one. We divided the workload evenly, according to our skillsets.

We have created a website which offers a set of tools to solve several mathematical problems. This was done with the user in mind, ensuring the highest quality of user experience. To accomplish this, we strived to create a self-explanatory layout, while still giving the user as much freedom as possible.

The website runs like a well-oiled machine internally, with flexible code relying on the input of the user, and not other way around.