STUDENTS AS CO-CREATORS

A Curriculum Design Collaboration

CODEMATIC: Bridging the Gap between Level 4 Maths and Programming

Student Partners

Alexey Andrienko, Joshua Higgs, Hannah Zlatkovic, Noel Varga, Oleksandr Bondarenko

Academic Partners

Alexander Bolotov, Natalia Yerashenia, Ester Bonmati Coll, Barbara Villarini

School of Computer Science and Engineering

Academic Year 2022-2023

A STUDENTS AS CO-CREATORS PROJECT

UNIVERSITY OF WESTMINSTER

Introduction

Our project team comprised academics responsible for teaching extensive Mathematics and Software Development modules at the School of Computer Science and Engineering. Naturally, these disciplines are closely intertwined, as various aspects of software development rely on mathematical concepts. For instance, complex programming instructions resemble our natural language's construction of compound statements using connectives such as AND, OR, IF...THEN – known as Booleans in Mathematics and Programming. The precise semantics and applications of Boolean expressions in the Sciences are crucial, as improper usage can lead to unintended outcomes or results contrary to expectations. This is just one example of the profound connection between these two subjects.

Despite numerous efforts to integrate Mathematics and Software Development in the computing curriculum, there is currently no coherent strategy or shared understanding among academics regarding the best approaches to deliver the material of these subjects to students as a synergy of the disciplines. Recently, the pedagogical research community in Mathematics and Computer Science has increasingly emphasised this synergy, mainly focusing on "computational thinking" and "mathematical thinking" concepts. As a result, these concepts serve as the context for the team's research project.

One vital aspect of computational thinking is data management, which encompasses two main principles: data abstraction and representation.

Certain problem-solving scenarios necessitate specific reasoning when dealing with data structures. For instance, when provided with an initial data collection (data storage), creating a new collection by modifying the initial data using rules, methods, etc., becomes essential. For example, given a collection of numbers, one might need to double the data (numbers) to create a new collection.

The *research questions* of our project have been:

1. What elements define the concept of "computational thinking"?
2. What types of problem-solving tasks apply to both level 4 Mathematics and Software development courses?
3. How do we establish a conceptual link between level 4 Mathematics and Software development modules, incorporating relevant topics, themes, algorithms, and reasoning techniques?
4. How can we enhance outputs 1 and 2 by providing additional examples that are pertinent to mathematics and software development modules at levels 5 and 6, thereby expanding the scope and impact of our proposal?

The *aims* of our research have been:

1. Create an open-source library of problem-solving tasks and their solutions, suitable for both level 4 Mathematics and software development education, with the aim of maximising its impact on learners.
2. Establish a comprehensive conceptual link between level 4 Mathematics and Software development modules, integrating relevant topics, themes, algorithms, and reasoning techniques.
3. Devise an evaluation methodology to assess the educational benefits of the synergy of Mathematics and Computing in students' learning experiences within the disciplines of Mathematics and Computer Science.
4. Utilising the methodology from item 3, investigate the current impact of integrating Mathematics and Computing in student learning for both Mathematics and computer science disciplines.

5. Propose strategies to expand the existing level 4 library by incorporating examples and tasks pertinent to levels 5 and 6, thus enhancing the resource's usefulness across various academic stages.

This research project holds great significance for both the team and our undergraduate students as it offers a holistic perspective on studying the interplay between Mathematics and Software development, surpassing their individual paths. For students involved, the project has been a valuable opportunity to engage in practical and meaningful research, where they can analyse intricate and challenging tasks while also creating and presenting their solutions.

From the academics' standpoint, this research is of fundamental importance as it allows them to enrich their teaching practices through a fresh approach that focuses on illuminating the inherent relationship between Mathematics and Software development. Additionally, the project provides a deeper understanding of the specific aspects of Mathematical and computational thinking, contributing to better cater to the current students' educational needs. The team of academics currently leads various Mathematical and Software development modules across all levels, involving around 800 students. Consequently, by integrating most of the project's outcomes into teaching, it will directly impact a substantial student population.

Moreover, given the potential integration of Mathematics into other disciplines like Sciences and Business, this project can extend its impact significantly, reaching a broader audience.

Background and Rationale

Mathematics and computing are interconnected fields that play crucial roles in modern problem-solving and decision-making. Computational thinking (CT) involves using algorithms and logical reasoning to solve problems via technology and computing tools whereas Mathematical thinking (MT) involves abstract reasoning and logical deduction by way of mathematical concepts and methods. This literature review will examine the integration of computational and mathematical thinking in higher education and critically review the existing strategies and approaches for developing a common understanding between the two disciplines. The aim is to provide a recommendation for future research and practice to enhance the integration of Mathematics and Computing.

Although CT and MT share fundamental similarities such as abstraction, algorithmic thinking and logical reasoning; there are distinct differences that make them both valuable problem-solving tools in their own right. CT is inspired by computer science principles and helps to promote creativity, logical reasoning and algorithmic skill which is why it is a valuable skill in computing. The process involves breaking down complex problems into parts that are smaller and more manageable, known as decomposition, devising algorithms, and utilising abstraction and automation to find efficient solutions (Wing, 2006). In contrast, MT is a cognitive process that hinges on mathematical reasoning and critical analysis to identify patterns, formulate opinions, prove theorems, and draw conclusions from deductive logic (Devlin, 2003).

The integration of CT into mathematics education has helped to transform problem-solving for students, providing them with a structured and systematic approach to tackling complex mathematical problems (Grover & Pea, 2013). By adopting algorithms and computational techniques, learners can enhance their understanding of abstract concepts and develop creative solutions to their problems. Computational tools, such as programming languages and mathematical software, have become essential tools in the technological revolution for modelling real-world scenarios and conducting mathematical simulations. Making CT and MT valuable assets to students when preparing to navigate a technology-driven world (Grover & Pea, 2013; Wing, 2006) by not only enhancing students' problem-solving abilities but furthering their creativity and innovation when exploring various ways to apply mathematical concepts in practical and interdisciplinary contexts.

Historically, the development of Mathematics and Computing has been closely entangled. In the mid-twentieth century, mathematicians pioneered the development of the first computers to automate the cumbersome calculations involved in their research (Knuth, 1998). As computational power grew, so did the opportunities for innovative mathematical applications, fueling a symbiotic relationship between the two fields. Regarding education, the introduction of the integration utilising programming languages specifically designed for educational purposes such as BASIC and LOGO (Papert, 1980). These initiatives aimed to enhance mathematical understanding through hands-on problem-solving and algorithmic thinking, yet the limitations of the technology and pedagogical approaches of the time-constrained their success. The increase in accessibility to computers from the 2000s renewed those efforts. Cultivating the understanding that programming can also be used to develop computational thinking skills, not just be used as a tool for solving mathematical problems (Wing, 2006). In the last decade, the growing demand for STEM-related skills in the global workforce has supported the escalation of integrating Mathematics and Computing in education (Honey et al., 2014). As the technological revolution continues to evolve, it is becoming increasingly clear that the intersection of Mathematics and Computing will continue to shape the evolution of education.

The integration of CT and MT in higher education is vital in equipping students with essential problem-solving and analytical skills whereby current approaches to teaching both skill sets aim to strengthen both mathematical and computational competency. However, the potential for future approaches lies in enhancing their integration and addressing emerging challenges.

Mathematics and Computing within the current education curriculum from K-12 and up to higher education have various different connections. These connections though are primarily based on topics such as matrices, vectors and recursion, etc. What it lacks currently is teaching how these two topics are intertwined with each other and what skills you can take from one and use in the other. The critical connection between the two is how you think while problem-solving. This is called Mathematical Thinking (MT) and Computational Thinking (CT). These two ways of thinking are connected heavily through the idea of Abstraction, Decomposition and Logical Reasoning, they provide a framework for identifying issues, analysing and subsequently solving them. The difference comes with how they are taught within education. MT is taught through traditional courses that emphasise abstract reasoning, proof-writing, and problem-solving (Devlin, 2003; Jones, 2018). CT is typically introduced in programming courses, where students learn algorithmic thinking and problem decomposition using programming languages and software tools (Wing, 2006; Grover & Pea, 2013). A study done by the university of Oslo highlights the links between mathematics and computing from the perspective of a student. The end results showed that “students see similarities across domains and find these relevant enough to merit a shift in activity or use resources from different domains simultaneously. Our students used mathematics as a resource to write and understand code, and to interpret the output. Conversely, they used both code and output as the basis for doing mathematical work that helped them understand and modify their computer programs“ (Sand et al., 2022).

We also came across several attempts of using specific methods and techniques to teach programming and mathematics “together”: in (Vaninsky, 2007) we can see some examples of this teaching based on the development and analysis of a simple game; (Orland, 2020) presents an account of the topics of the object-oriented programming framework that are closely linked with maths.

However, what these papers lack is a systematic approach utilising the analysis of problem-solving tasks across the range of programming and mathematics topics utilising mathematical and programming solutions for the same tasks. Our project aims to tackle exactly this problem - to create a library of such problems, provide template solutions and make these available to the students.

The future of mathematical and computational thinking in higher education will likely see greater integration and innovation to address emerging challenges. One potential approach is the development of integrated curricula that weave mathematical and computational thinking throughout the students' academic journey (Wing, 2006).

Technological advancements also open doors for more interactive and immersive learning experiences. Virtual reality, simulations, and gamification can be utilised to create engaging environments where students can explore mathematical and computational concepts hands-on (Grover & Pea, 2013). These interactive tools can help students to garner a better understanding of abstract concepts and subsequently encourage a more creative approach to problem-solving.
Moreover, incorporating emerging fields like data science and artificial intelligence into the curriculum will be crucial (Devlin, 2003). Since these fields heavily rely on mathematical and computational thinking and are driving forces behind numerous industries, integration of their concepts into mathematics and computing courses appear to be the next logical step. By exposing students to how CT and MT can be used in different contexts, educators become closer to achieving the recommended long term goal of helping students acknowledge and utilise the transferability of these vital skills (Barr et al., 2011).

In conclusion, integrating CT and MT in higher education equips students with crucial problem-solving skills for the evolving technological landscape. Real-world applications and interdisciplinary projects bridge theory and practice, enhancing the relevance of education. Future approaches should focus on seamless integration through problem-based learning and immersive experiences like virtual reality. Incorporating data science and artificial intelligence concepts will prepare students to address data-driven challenges in various industries. Collaboration between mathematics and computing departments, along with faculty development, is vital for successful implementation. The symbiotic relationship between Mathematics and Computing will continue to shape education, producing adept problem-solvers and critical thinkers for the digital era.

Project Surveys and Workshop

Survey Creation

Our team has created surveys for university students and lecturers to learn more about Programming and Mathematics related knowledge and issues.
The purpose of the survey for students was to find out what topics were not explained well or were hard to understand in Mathematics and Programming and, if it was understandable, what external or internal sources the students used to study. Also, the team wanted to know if the students understood or saw any links between the two subjects and if they did what it was.
The survey for lecturers mainly focused on what requirements the students need to meet to understand and complete the subsequent year's modules. Lecturers could tell what previous knowledge they want a student to have and what topics most students struggle with in both subjects.

The links to the surveys are available below.

Students Survey link:
https://docs.google.com/forms/d/e/1FAIpQLSeKsXnbMP5dLW9ZciajkiwjFHndH-62okjkr_qUl8C4ZVkBbw/viewform?usp=share_link

Lecturers Survey link:

https://docs.google.com/forms/d/e/1FAIpQLSf0qzPSjn6bhvgljw8a36r4oFb_VyHyzaTPjIdnjb0ve4YoDQ/viewform?usp=share_link

Survey Publishing

The surveys were available to fill in on the 6th of April, 2023. On this day, a workshop was held where both parties could come in and fill them in person. This gave an opportunity to talk further about the questions both with students and lecturers. Since the team needed more answers for processing, these surveys were also sent out to all students and selected lecturers.

Workshop

Our team invested significant efforts in organising the workshop which we successfully conducted on April 6th, 2023 (11:00-14:00). This was a very engaging workshop dedicated to showcasing the CODEMATIC project to both students and lecturers. The workshop served as an interactive platform to introduce the key aspects of our project, its objectives, and potential impacts on the learning process.

Figure 1. The CODEMATIC Team at the Workshop

The event was well-attended, with approximately 15 academic staff and 20 students participating, reinforcing the pertinence and intrigue of our project within the academic community.

During the workshop, attendees were shown a specially created video illustrating the project's aim, looping continuously throughout the event to allow a consistent visual representation of our work and to engage with the audience. This media tool was an effective method to communicate our project's essence, fostering a deeper understanding among the participants.

The *workshop video* is accessible via the following link:
https://universityofwestminster-my.sharepoint.com/:v:/g/personal/bonmate_westminster_ac_uk/EW0deYizC3tGnGqN4ZAmvD4BVBtSXh2UiDgTEEv0YIdArQ?e=I4rBrB

A significant portion of the workshop was dedicated to inviting participants to complete our surveys. These surveys, designed to glean insights from both students and lecturers, played a pivotal role in informing our project's direction and strategies. The feedback we received helped shape our project and validated some of our initial hypotheses.

Additionally, we presented our Ontology during the workshop, an intricate network of connections bridging Maths for Computing and Programming modules. This interactive segment aimed to stimulate thoughtful discussion about potential improvements and changes to the existing links between the modules. Attendees' perspectives and suggestions were invaluable and have influenced our project's ongoing development.

We concluded the workshop with a Q&A session. This open dialogue offered an opportunity for students and staff to seek clarity, express their thoughts, and provide real-time feedback on the CODEMATIC project. The active participation we witnessed during this session underscored our project's relevance and the keen interest it sparked among the academic community.

In summary, the workshop was not just a platform to present our work; it was a catalyst for deeper engagement with the CODEMATIC project, encouraging a broader understanding and appreciation for our dedicated efforts.

The room plan for the COPDEMATIC Workshop is presented in Fig. 2.

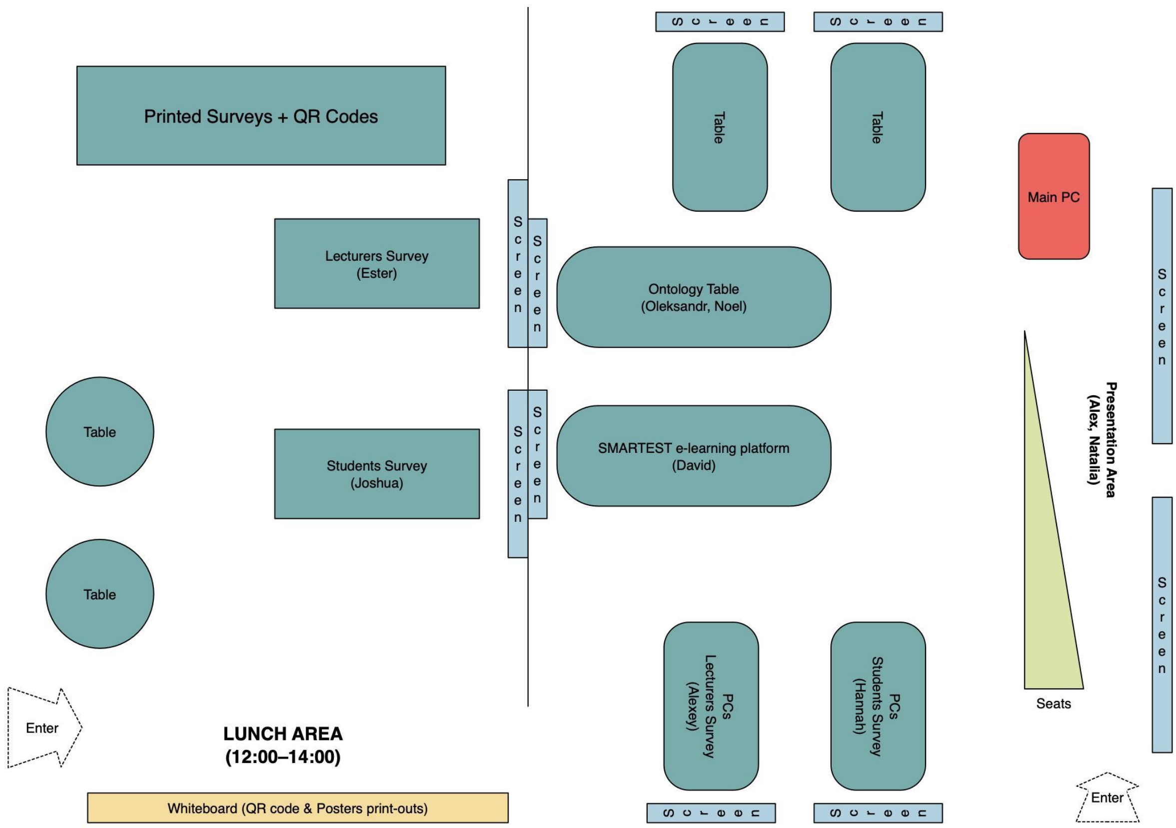

Figure 2. CODEMATIC Workshop Room Plan

Survey procession

The procession of the surveys was done by one of our team members, Noel Varga. This procession document was created as an Excel file where all the answers given by the students and lecturers were divided. The number of the different answers was counted, and the percentage of the total number of answers was calculated to see which response was the most common. The created documents of the processed survey answers are available at these two links:

Students Survey Processed Data Excel file:
https://docs.google.com/spreadsheets/d/1fp4thAzTgSUG7qgptvMzvPhNJGCtE-xq/edit?usp=share_link&ouid=106039130874729422138&rtpof=true&sd=true

Lecturers Survey Processed Data Excel file:
https://docs.google.com/spreadsheets/d/1arYxg7SpsjLyrY_C_BbnBJ9N-ejwB-yX/edit?usp=share_link&ouid=106039130874729422138&rtpof=true&sd=true

CODEMATIC Ontology Development

What is Ontology?

An Ontology is a set of concepts and categories in a subject area or domain that shows their properties and relations between them. In our team's case Software Development and Mathematical topics. We use Ontology to share a common understanding of the structure of information among people or software agents, enable the reuse of domain knowledge, make domain assumptions explicit, and analyse domain knowledge. Creating the ontology is similar to representing a data set and its structure for other programs. We use ontology to create the structure for all the topics in Software Development and Mathematics and the links between said subjects' topics. The created ontology is also the structure for our website's content. To build the ontology our team used Protégé.

Protégé environment

Protégé is a program used to create and maintain ontologies. This environment makes creating ontologies easier to understand, and it doesn't require learning to code an ontology. Our team has created notes and we based the ontology's structure on it.
The structure of the ontology is based on two classes, Topics of Maths Modules and Topics of Software Development Modules and these modules have subclasses.
If we take Maths Modules' Topics subclasses first, these are the topics of the Level 3 Foundation Math Module and Level 4 Mathematics for Computing Module. Level 4 Mathematics for Computing Module has five topics as its subclasses at the time of writing of this report. These subclasses are Discrete Mathematics, Elementary Logic, Linear Algebra, Probability Theory, and Relations and Functions Topic. In some instances, the topics also have subclasses which describe the topics further in depth. For example, the Discrete Mathematics Topic has its subtopics - Graph Theory and Trees Topic, Intervals set Builder Topic, Sets and Operations with Sets, and Tableaux Logics.
The previously mentioned Software Development Modules have Level 4 Software Development II Module and Level 5 Object Oriented Programming Module at the time of writing this report. Level 4 Software Development II Module has multiple subclasses: Array and Debugging, Classes and Objects, Data Types and Control Structures, Files and Testing, Methods and Functions, Programming methodologies, Queue and Stacks, and Sort and Search.
We structure the ontology in this way to be able to create links between the two subjects more efficiently. For example, we have a link between Mathematics in Computing's Graph Theory and Trees and Software Development II Sort and Search. The link between the two topics is created using annotations. Here a sibling property is created called a relationship. We use this annotation to show the connection via a clickable link between the two topics. This annotation also has a sub-property

called “Explanation of relationships". This, as the name suggests, explains the link between the topics. The explanation, in this case, is: “Linear search can be represented as a branch of a tree. For example, in a depth-first search, we traverse a tree top-down (from the root to the leaves) exploring branches of a tree (that are linear sequences of nodes) and can implement various tasks, for example, comparing the weight assigned to the root node with the weights of the successor nodes along this branch.” (see screenshots)

CODEMATIC Ontology Development

Fig.3 screenshot shows the protégé environment. On the left, the classes and subclasses are visible, while on the right, the annotations. The annotations house different types of information, like the label of a said subclass, comments about it, additional sources, and the link to a different topic.

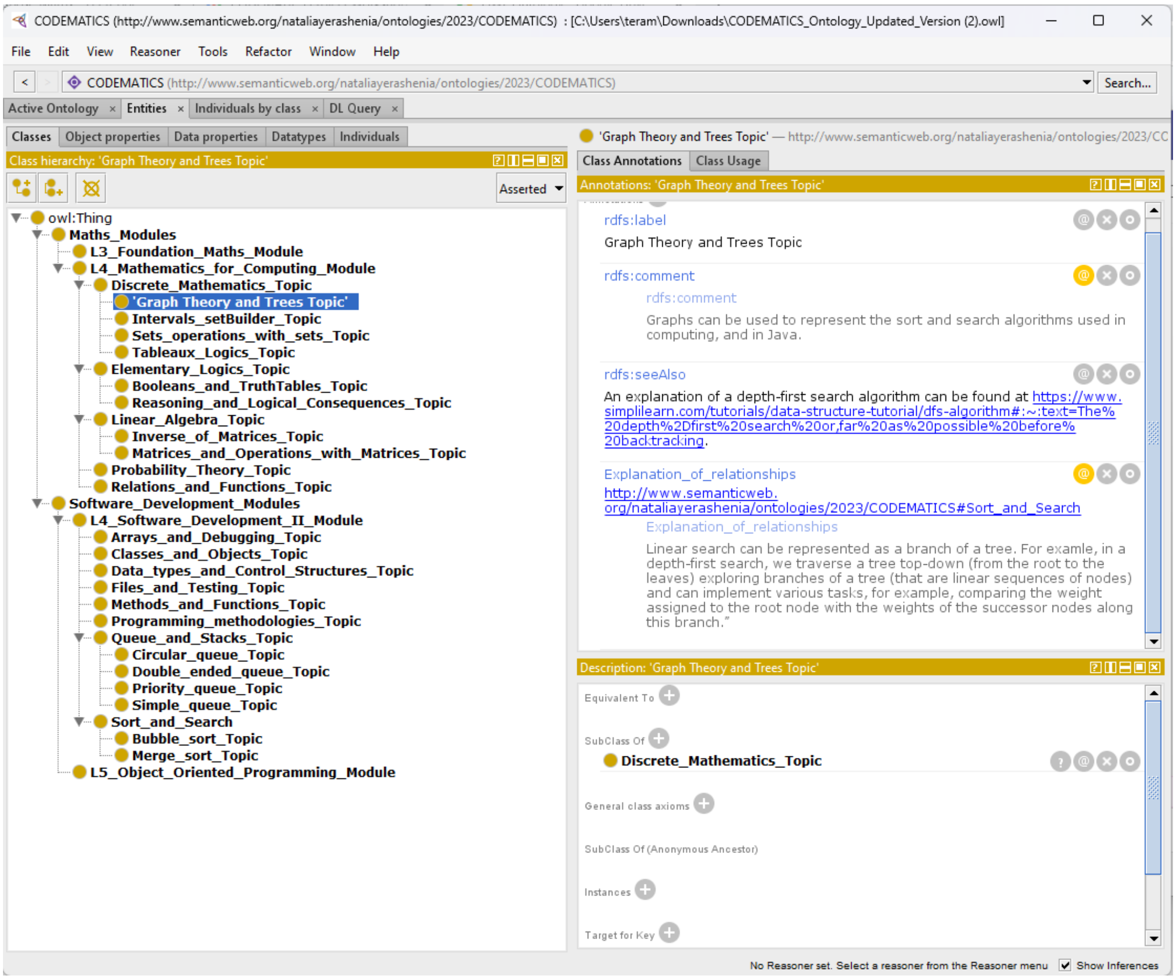

Figure 3. The structure of the CODEMATIC OWL Ontology file in Protégé. Graph Theory and Trees Topic

Fig. 4 screenshot shows the connected topic of the previous screenshot.

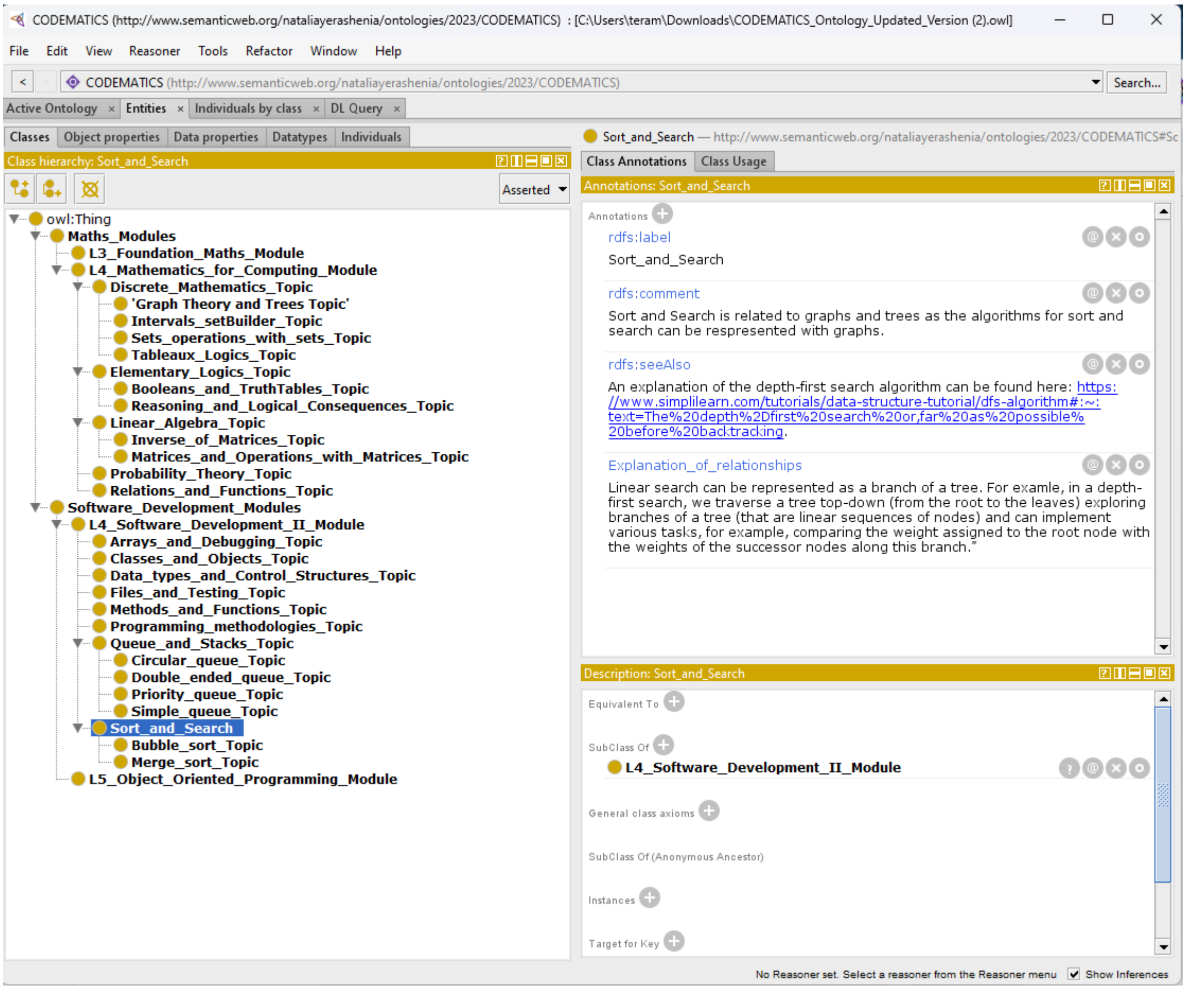

Figure 4. The structure of the CODEMATIC OWL Ontology file in Protégé.
Sort and Search Topic

The link to the *CODEMATIC Ontology OWL file*:
https://drive.google.com/file/d/1pnjZF9GRPgx7bs0Alev-VRaDTHhBKfiN/view?usp=share_link

Website Development

The website development process was planned and fulfilled by student partners Joshua Higgs and Hannah Zlatkovic.

Home Page

The website's homepage combines functionality and style effectively.
In the navigation bar, the website title and subtitle are clearly displayed (Figure 5), and future direct links to different sections like Math, Programming, and Resources have been prepared.
A key part of the website was making sure to use the correct semantic tags to employ clear structure and accessibility for all users.

```html
<nav>
    <h1><a href="/index.html">CODEMATICS</a></h1>
    <p>Bridging the gap between Mathematics and Programming</p>

</nav>
```

Figure 5. Navigation/Header for home page

The main section showcases progress indicators for Maths and Coding, encouraging user engagement. This is supported by two action buttons, "Continue Maths" and "Continue Coding." The progress bars utilise an inner div to visually represent progress *(Fig.6.)*, which can be dynamically controlled through JavaScript. While the progress bars currently do not automatically update/adjust this opens the possibility for potentially adding that in the future as the groundwork is already completed

```html
<div class="mathsProgress progressStyles">

    <div class="progress-bar-container">
        <div class="progress-bar" id="my-progress-bar-maths"></div>
    </div>
    <script src="javascript/indexscripts.js"></script>

    <div><a href=""><button class="progressButtons">Continue Maths</button></a></div>
</div>
<div class="codeProgress progressStyles">

    <div class="progress-bar-container">
        <div class="progress-bar" id="my-progress-bar-code"></div>
    </div>
    <script src="javascript/indexscripts.js"></script>

    <div><a href=""><button class="progressButtons">Continue Coding</button></a></div>
</div>
```

Figure 6. Progress bar for maths and coding

The middle section offers a structured table presenting Mathematics and Programming questions on key topics, each linked to a dedicated HTML page, facilitating an interactive learning experience.
The final section promotes third-party resources like Codecademy, Udemy, FreeCodeCamp, and Codingbat, complete with relevant images and titles.
The CSS styling across sections ensures clarity and efficient use of space. The main and final sections utilise a flex container layout for neat arrangement, while the middle section employs a monochromatic scheme, padding, and centred text for easy readability. While the current design is functional and user-friendly, there are hints at the potential for additional interactive features.

Content Pages

The content area for these pages follows a left & right alignment. The left section has multiple sub-sections: the task description, a visual example, an informational box, and a pseudocode presentation. The task description and informational boxes are well highlighted using a different background colour. The pseudocode is structured in a table format for better understanding.
The right section is more interactive. It provides an area for users to enter their code and a button to run it. The output is displayed in another area within this section. Both the code input area and the output area are stylised to make them distinct.
The webpage is styled for an improved user experience. The universal box-sizing border box ensures a consistent layout. The body is set to display flex to align the content horizontally. The left and right sections have equal widths, and they are both padded to create space around the content. The backgrounds of various sections are highlighted using different colours for clarity.

The interactive parts of the page like the text area for user code and the run code button, are styled for a more engaging look and feel. The output area has its own styling to make it clear and distinct. The images are set to take the entire width and height of their containers.

In conclusion, the design and coding of the website exhibit thoughtful planning and careful implementation. The HTML structure and CSS styling have been optimised for clarity, efficient use of space, and ease of navigation. Using flex containers in various sections ensures a responsive design that adapts to different screen sizes.
The content of the question pages is designed with the user in mind, providing clear task descriptions and structured pseudocode while offering interactive code input and output areas. The distinctive styling of different sections and components aids in user comprehension and engagement.

Screenshots in Fig. 7-10 show the user view of the website.

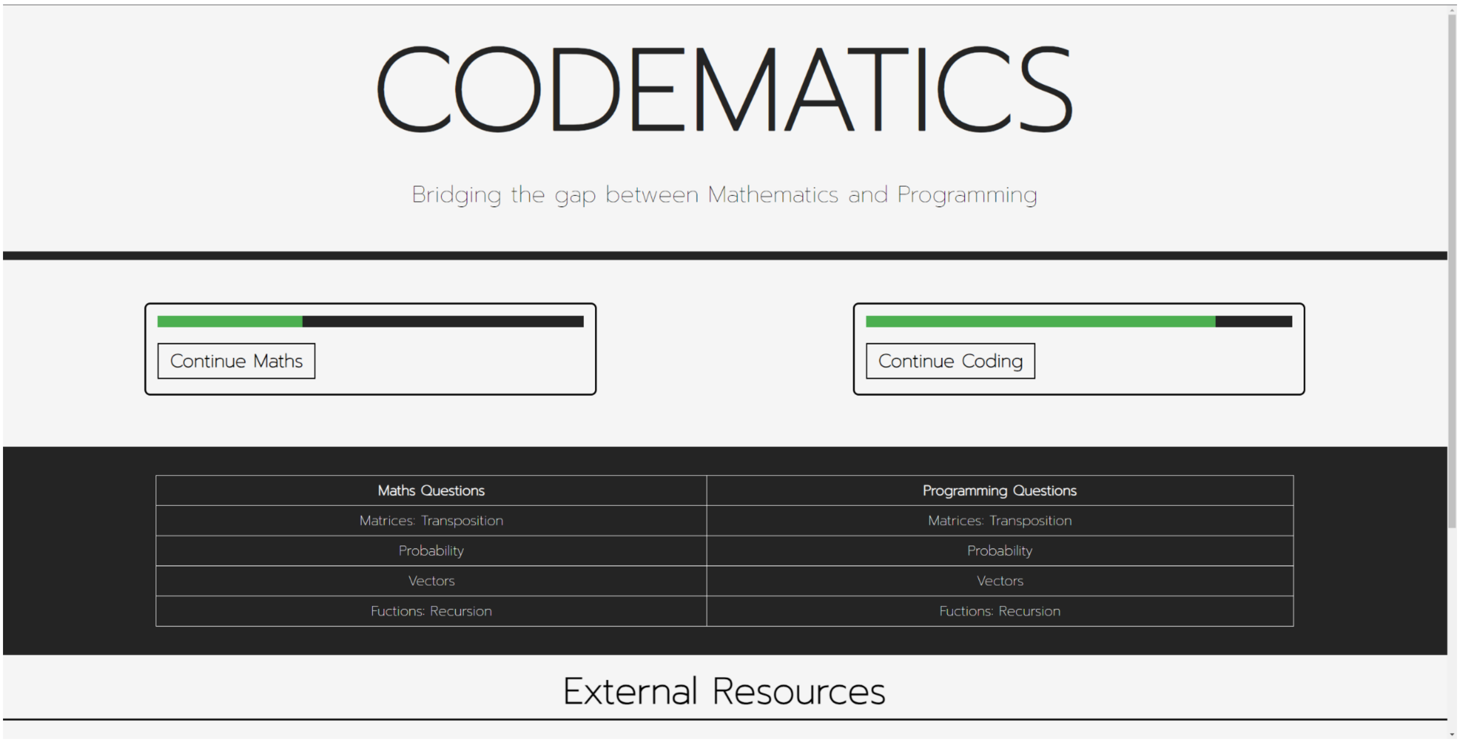

Figure 7. Website Main Page

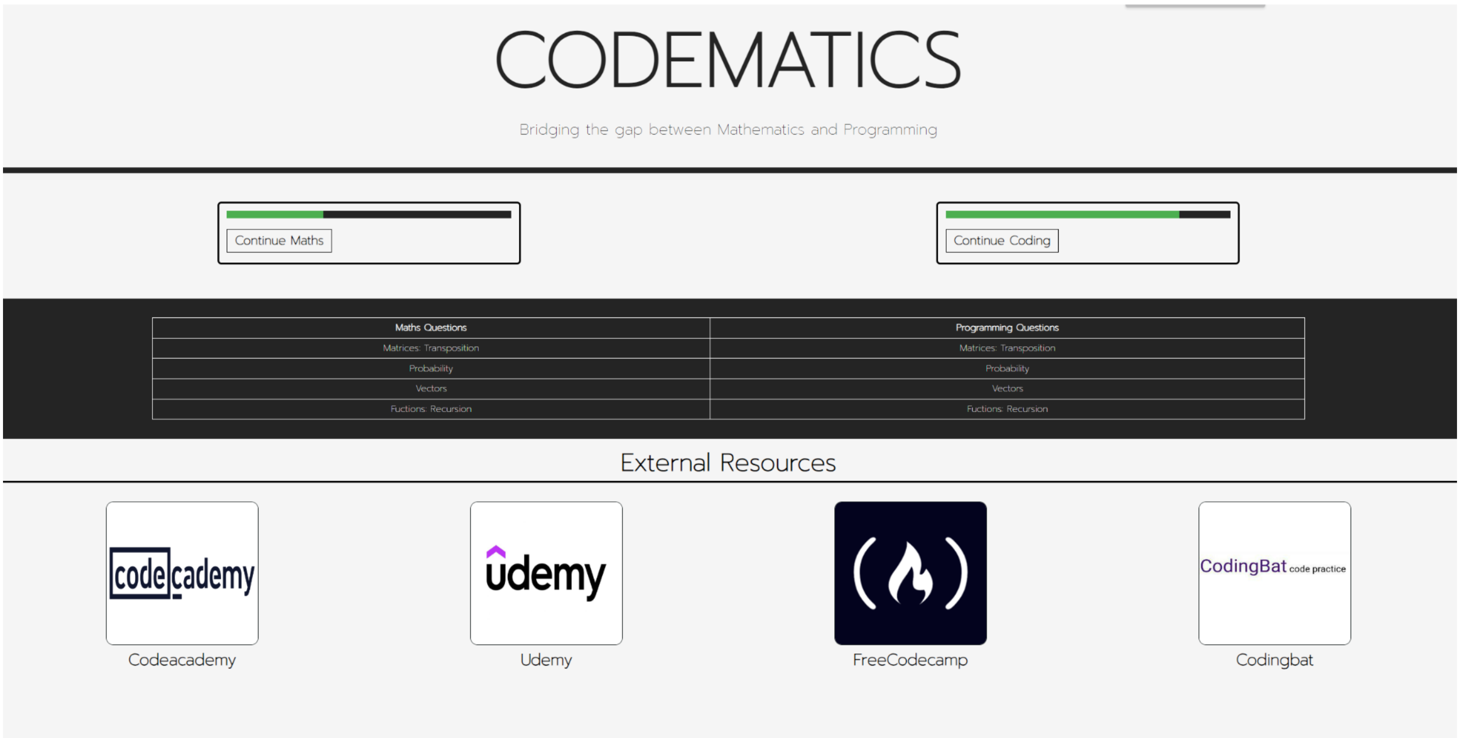

Figure 8. Website Main Page. External Resources

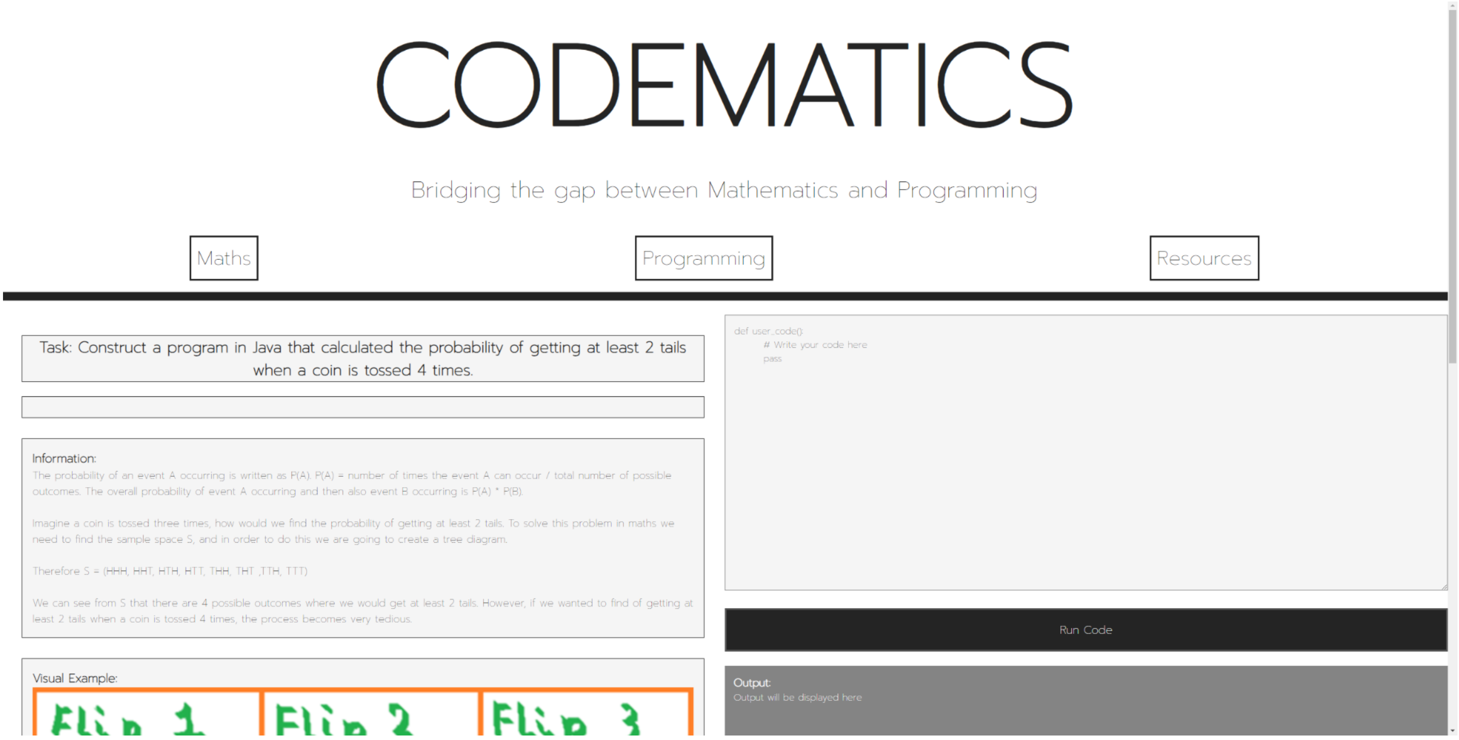

Figure 9. Task Example Page

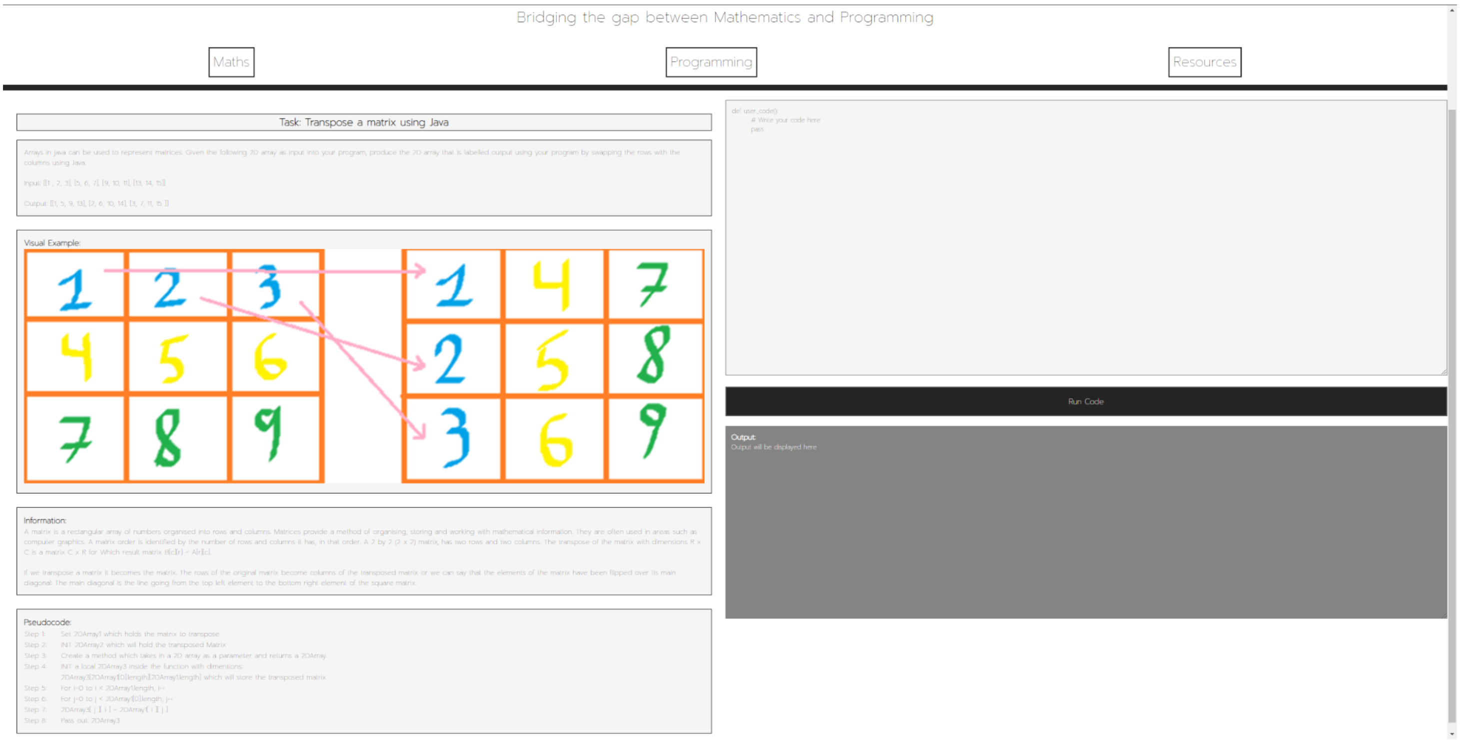

Figure 10. Task Example Page

Website Content Creation

The content creation part was accomplished by the student partners, Alexey Andrienko, Noel Varga and Olexandr Bondarenko.

The team created an Ontology for the interrelated maths and computing topics that are studied at level 4 to have a structure and to connect the topics to help fill the content of the website. This section describes an example of the content creation process.

Trying to create some content for our website which links mathematics and computing modules together, we explored the existing strategies for achieving this.

The analysis of the students' survey results has shown that matrices, probability, graphs, calculus, and recursion were all topics that students thought would help them if there was more content on them. Also, during level 4 students spent a lot of time learning matrices, and, taking into account a natural link of matrices in mathematics and arrays in programming, we decided to focus on that topic together with the topic of probability to produce maths/computing linked problem-solving tasks. Another reason for our decision to focus on matrices and probability is that we found them to be interesting and it was easy to imagine how we could connect maths and computing in some of the tasks that students worked on as part of their courses. We used the same approach for all potential topics.

Use case:

While we looked through the level 4 Mathematics for Computing module and Software development modules aiming to come up with problem-solving tasks that we could use for our website, we found appealing tasks which involved the transposition of a matrix. We refer to the notes available on Blackboard on the University of Westminster website (Bolotov, 2022). At first, the transposition of a matrix can seem confusing, and it is not easy to picture it in your mind. We thought about how we could transpose a matrix using a Java program, and we worked on it. We realised that any element with index `matrix[i][j]` would have to become `matrix[j][i]`, i.e., swapping the element's indices as can be seen from the code snippet in Fig 11.

```java
public static int[][] transpose(int[][] matrix) {
    int[][] answer = new int[matrix[0].length][matrix.length];
    for(int i=0; i < matrix.length; i++){
        for (int j = 0; j < matrix[0].length; j++){
            answer[j][i] = matrix[i][j];
            //As we iterate over the rows and columns of the original matrix
            //we assing each element of the original matrix into a new matrix
            //such that the index of each element in the new matrix has the
            //row index equal to the column index of the corresponding
            //element in the original matrix, and for the column index of
            //that element to be equal to the row index of the original
            //element in the original matrix. Basically swapping the rows
            //for columns and the columns for rows.
        }
    }
    return answer;
}
```

Figure 11. A JAVA method that can transpose any matrix (two-dimensional array)

Our solution, in fact, is a good example of the synergy of two disciplines:
Being unsure about how we could store the new matrix (or two-dimensional array) in the Java program, we appealed to the “outside” world and to determine what dimensions the new matrix should have. Luckily, we found such an example on LeetCode (LeetCode, 2023). The way to create the new two-dimensional array to store the transposed matrix would be to create a two-dimensional array with the same number of rows as the number of columns in the old array (matrix) and with the same number of columns as the number of rows in the old array, as can be seen in Fig. 11.
Once we understood how to transpose a matrix in a Java program, we started working on designing this as a task for students to solve based on what they have learned in Maths and programming modules. In particular, our solution involved constructing a diagram of what happens to a matrix when it is transposed and explaining, in mathematical terms, what a matrix transposition does; this is illustrated by an example of a 3x3 matrix as shown in Fig. 12.

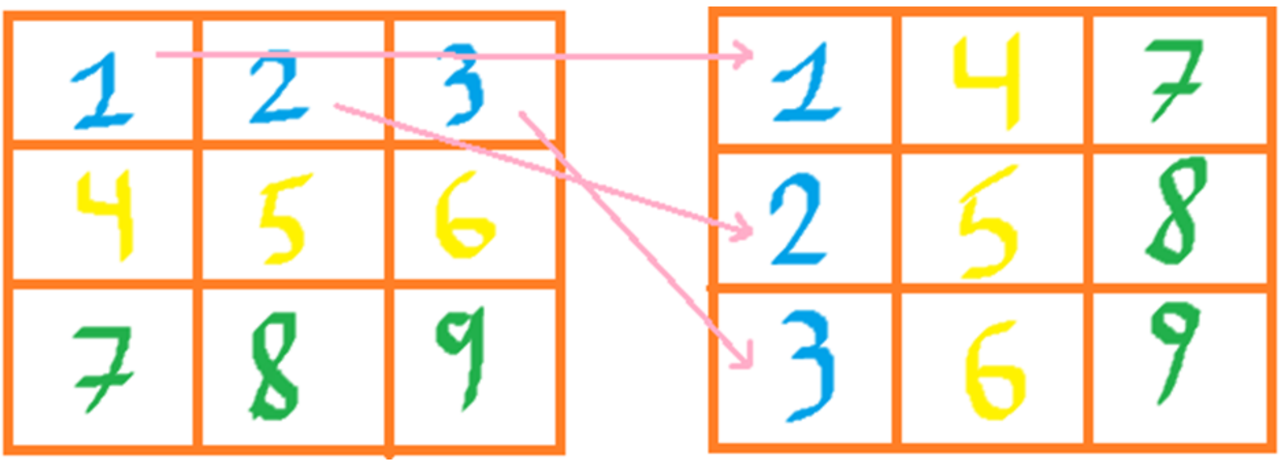

Figure 12. The diagram shows what happens to a matrix when it is transposed.

Once we created a Java solution, which can transpose any two-dimensional array (matrix), we added some comments to the actual code to show what the program is doing, as can be seen in Fig. 11. We also supply a pseudocode to help the end-user understand the programming implementation of the maths concept of matrix transposition. We then followed the same strategy for developing the rest of the examples.

Team Communication and Tools

Throughout the CODEMATIC project, the team adopted various communication channels and tools to ensure seamless collaboration, transparency, and efficiency in their collective efforts.

Email Communications: In the initial stage, the team organised their communications primarily through email. It allowed for formalised exchanges and served as a platform for sharing essential documents and information.

In-Person Meetings: At the beginning of the project, the team conducted two in-person meetings. These face-to-face interactions facilitated detailed discussion, planning, and strategising for the project's execution. We also met in person during our workshop.

Microsoft Teams: Post study week 12, the main communication hub shifted to Microsoft Teams. The platform enabled weekly online calls (of one-hour duration), creating a virtual workspace where team members could discuss progress, clarify doubts, and brainstorm ideas. The Teams chat further fostered an environment where quick updates and informal communications could occur.

Google Drive: To collaborate on project files and documents, a shared Google Drive folder was used. It provided a centralised location where all team members could access and modify files, ensuring that everyone was working with the latest versions and updates.

Google Drive Link (can be accessed by UoW accounts): https://drive.google.com/drive/folders/1sJMkqwUo9oFrKpAZ-zQBqPS6ycqAhibv?usp=share_link

Task Management with Trello: The project's tasks were meticulously managed using Trello. This tool allowed the team to categorise, assign, and track the progress of various tasks. It created transparency in workload and responsibilities, enhancing efficiency in project execution.

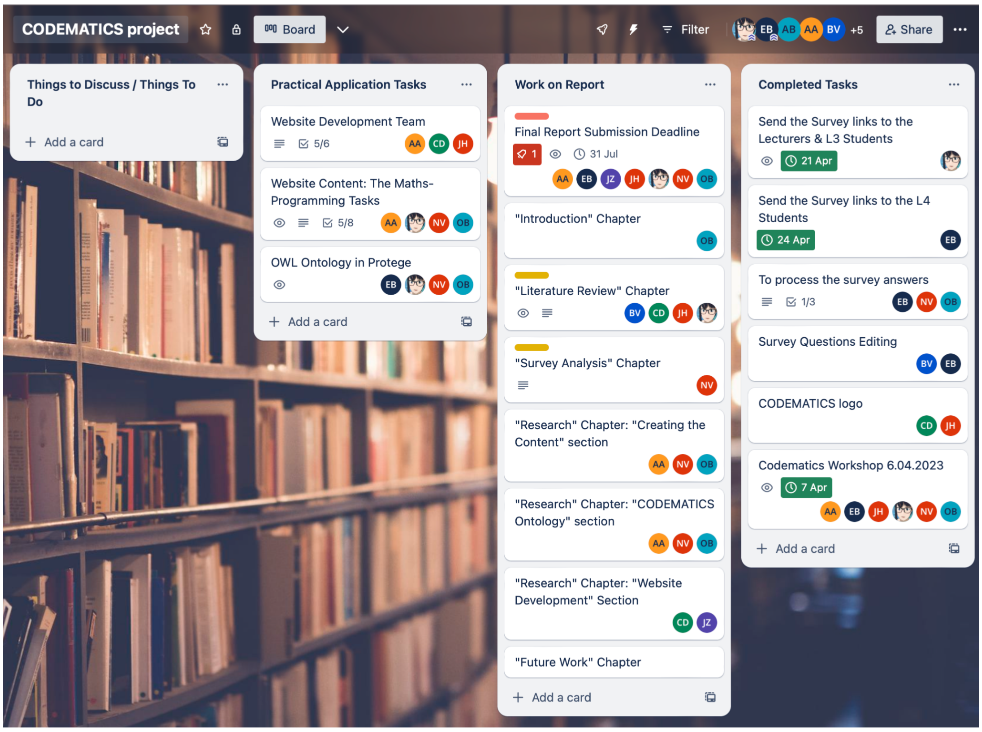

Figure 13. CODEMATIC Trello board

Website Data Management on GitHub: For the technical side of the project, specifically the management of website data, GitHub was employed. It enabled version control and allowed for collaborative coding, which is crucial in a project with multiple contributors.

Our GitHub Link: https://github.com/Yerashenia/CODEMATICS

Surveys and Workshop: Additional tools like Google Forms were used for creating and distributing surveys, while a workshop was organised to gather insights directly from students and lecturers. Moreover, the workshop allowed live communication with the attendees (students and academics), we were able to discuss our work in progress, seek live feedback, and even familiarise ourselves with other educational software developments done by our colleagues at the School of CS & Engineering.

In summary, the blend of traditional and modern methods ensured the team could adapt to various needs and preferences, thus fostering a cohesive and efficient working environment. Adopting these diverse tools reflects the project's commitment to leveraging technology to enhance educational synergy between Mathematics and Software Development.

What We Learned

Noel Varga – I learned many new things by participating in the CODEMATIC project. This includes working in a team, managing tasks, meeting deadlines, and making sure to keep an efficient workflow. I appreciate that I got to be part of this team and I really enjoyed that I can do something productive and helpful.

Alexey Andrienko – As a result of being part of the CODEMATIC project, I now understand more about how learning Mathematics and Computing can complement each other. I also improved my programming skills by creating problem-solving tasks for potential users of our website.

Dr Alexander Bolotov, academic partner – I believe this has been one of the most successful co-creators projects I was part of. It was indeed a co-creation of the project outputs by both student and academic partners. It has been an absolute pleasure to work with this fantastic team! What I personally learnt: the topic is “hot” - teaching both maths and software development for years, I always witnessed that students, in general, do not draw parallels between these subjects. To draw such parallels has been my aim for years and, finally, this is becoming true! I am sure that for academics, this project has given a nice starting point for “bridging” the gap between these two subjects.

Josh Higgs - By participating in the CODEMATIC Project I was able to expand upon and learn new skills which include working within a team, tackling problems independently and also how to research more obscure topics which lack the depth you might normally find when researching. I believe many of these skills will help me in the coming years during my studies. I also was able to further develop my programming skills by being able to put more focus on the web scripting language HTML and the styling language CSS while also being able to dabble in some JS just to keep my memory refreshed and ready for year 2.

Hannah Zlatkovic – Being able to partake in this project has broadened my understanding of the relationship between maths and programming and helped to garner a deeper appreciation for the theorem I have been taught during my studies. Having the ability to work alongside those who have a passion for our topic of research has fostered an energy which I hope to carry with me into my next year of education.

Oleksandr Bondarenko - Participation in the CODEMATIC Project allowed me to develop my soft skills such as teamwork, deadlines management and the ability to present my own ideas. My maths and coding skills were also improved while I was doing research on this project. And now I can clearly see that it is impossible to study Programming efficiently without learning Mathematics as well.

References

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning and Leading with Technology.* Available at: https://www.learningandleading-digital.com/learning_leading/20110304?pm=2&pg=24#pg24 (Accessed: 30 July 2023).

Bolotov, A. (2022) Mathematics for Computing. Lecture Notes. University of Westminster Blackboard (2022) MODULE: 4COSC002W. Available at: https://learning.westminster.ac.uk/ (Course 4COSC002W Mathematics for Computing) (Accessed: 29 July 2023)

Devlin, K.J., 2000. The math gene: How mathematical thinking evolved and why numbers are like gossip (Vol. 329). New York: Basic Books.

Grover, S. and Pea, R., 2013. Computational thinking in K–12: A review of the state of the field. Educational researcher, 42(1), pp.38-43.

Honey, M., 2014. Committee on Integrated STEM Education Margaret Honey, Greg Pearson, and Heidi Schweingruber, Editors.

Knuth, D.E., 1998. The Art of Computer Programming Volumes 1-3 Boxed Set.

LeetCode (2023) 867. Transpose Matrix. Available at: https://leetcode.com/problems/transpose-matrix/ (Accessed: 29 July 2023).

Noy, N.F. and McGuinness, D.L., 2001. Ontology development 101: A guide to creating your first ontology. (Available at https://protege.stanford.edu/publications/ontology_development/ontology101.pdf, Accessed: 29 July 2023)

Orland, P. (2020) How improving your math skills can help in programming: TechTarget, CIO. Available at: https://www.techtarget.com/searchcio/post/How-improving-your-math-skills-can-help-in-programming (Accessed: 30 July 2023).

Papert, S.A., 2020. Mindstorms: Children, computers, and powerful ideas. Basic books.

Stanford Center for Biomedical Informatics Research. A free, open-source ontology editor and framework for Building Intelligent Systems, protégé. Available at: https://protege.stanford.edu/ (Accessed: 29 July 2023).

Sand, O.P., Lockwood, E., Caballero, M.D. and Mørken, K., 2022. Three cases that demonstrate how students connect the domains of mathematics and computing. The Journal of Mathematical Behavior, 67, p.100955. Available at: https://doi.org/10.1016/j.jmathb.2022.100955.

Vaninsky, A.Y., 2007. Teaching Mathematics and Computer Programming Together. In Innovations in E-learning, Instruction Technology, Assessment, and Engineering Education (pp. 51-54). Springer Netherlands.

Wing, J.M., 2006. Computational thinking. Communications of the ACM, 49(3), pp.33-35.