Traditional to transformative: Empowering education with computational thinking

Adri van Nieuwkerk
Opti-Num Solutions, South Africa, adri.vannieuwkerk@optinum.co.za

Prebantha Moodley
University of the Witwatersrand, South Africa, prebantha.moodley@wits.ac.za

Abstract

In 2020, the COVID-19 pandemic upended global routines, compelling a re-evaluation of established systems and daily life. Educational institutions faced the challenge of transitioning traditional teaching and learning practices to online platforms. This paper presents the continued collaboration, catalysed by COVID-19 lockdown, of industry partners, Opti-Num Solutions, MathWorks and the University of the Witwatersrand (School of Chemical and Metallurgical Engineering), on the second-year computing for process engineering course. The course lends itself favourably to online learning and assessments and is the ideal case study to explore the delivery of computational thinking-based coursework. We present findings from 5 years of this collaboration with purposeful interventions to enhance student learning and feature computational thinking relevant to the current climate. This includes the implementation of self-paced online courses and integration of an automated assessment tool. Results from a student survey show that approximately 95% of students had little to no experience in programming due to the absence of related courses in high school, limited opportunities, lack of resources, and no prior need or requirement to learn programming. However, after completing the self-paced online course, most students felt competent and empowered in their programming skills showing that these and future interventions impact computational learning.

Keywords: programming, COVID-19 interventions, digital tools, industry-academia collaboration

1 Introduction

1.1 Computational Thinking

It is now essential to include computational thinking skills in STEM courses, due to the need of these skills in industry, as expressed by the World Economic Forum, "Future of Jobs Report" in 2023 (World Economic Forum, 2023). The authors recognise that, at the time of delivering interventions for the computing for process engineering course (CHMT2011) and when writing this paper, definitions for computational thinking are fluid and continue to evolve. For the purpose of this paper, the authors have identified two definitions that encapsulate the approach for including certain course interventions.

First, computational thinking is "the conceptual foundation required to solve problems effectively and efficiently (i.e., algorithmically, with or without the assistance of computers) with solutions that are reusable in different contexts." (Valerie J. Shute, 2017). In addition to this, computational thinking vocabulary should include the second definition "CT encompasses the thought processes of abstraction, decomposition, algorithmic design, evaluation, and generalization." (Selby, 2013). In the case of CHMT2011, computational thinking is the fundamental stepping stone for students to achieve key learning outcomes.

1.2 Computing for process engineering: Course Overview

The computing for process engineering course (CHMT2011) is an introductory programming course taught at second year within the school of chemical and metallurgical engineering (CHMT) at the University of the Witwatersrand (Wits). The course aims to equip students with proficiency in computer programming to solve engineering problems, with the intention to develop a sense of importance for computers and programming in addressing realistic process engineering challenges (School of Chemical and Metallurgical Engineering, 2024). The course incorporates a computational thinking approach to teaching programming principles, i.e. enabling students to understand programming as a language for problem solving.

Key learning outcomes of the course: understand core programming principles, execute programs, understand data types, control flow, logic, functions, visualize data and create scripts. These were achieved through various learning tools, such as lectures, tutorials, assignments, tests and exams. MATLAB was the programming language of choice for this course. This decision was made based on (a) the availability of MATLAB via the Campus-Wide License (The MathWorks, Inc, 2025) (b) the ease of use of the integrated development environment and (c) the digital teaching resources and training made available to Wits.

MathWorks is a leading developer of mathematical computing software which engineers and scientists use and rely on to accelerate the pace of discovery and innovation. The campus license is a teaching and research license offering from MathWorks, providing open access to software and training for all faculty staff, researchers and students.

The course was previously taught in duplicate lessons to cover large groups of students (150 students average class size) where student submissions (often handwritten) were marked by the lecturer or tutors. This course was the first point of reference for computational thinking within the CHMT2011 curriculum; illustrating to students how to problem-solve using a programming language. It is important to note that computational thinking is still not considered a learning outcome or objective for this course, creating an incorrect perception of the value of the course.

2 Addressing Urgency

Wits, like many academic institutions around the world, was forced to re-evaluate the delivery of instruction while maintaining learning outcomes during the COVID-19 pandemic. In the case of CHMT2011, the course lent itself well to being taught online however an online transition proved a great challenge due to the immediacy of lockdown. There was little to no time for lecturers to mobilise online resources and foster an online mindset, thus ensuring motivation to learn for this course and meet learning outcomes became a driving a factor.

Opti-Num, the sole authorised distributors of MATLAB in southern Africa, have strong ties in academia due to their support for universities, enhancing instructional learning and innovative research. Additionally, they understood that a new approach was needed in this time. Collaborating with them presented an opportunity:

1. Intervention tools: Leveraging MathWorks resources and training to transition “classroom teaching” to self-paced learning in an online environment. This meant that learning about coding using the MATLAB interface was something students could learn independently.
2. Evaluation methods: Integration of an automated grading tool, MATLAB Grader, within the Wits learning management system (LMS), meant that students coding assignments and submissions were assessed timeously and with immediate feedback (unlike the tedious marking of handwritten coding assignments which result in delayed feedback).

3 Intervention Tools

Several tools, from the university and the campus license, were leveraged to address the transition of course lessons and assessments to online teaching and learning. The intervention tools are presented in this section and the implementation discussed in Section 4.

It is important to note that prior to 2020 none of the MATLAB related interventions presented here were used to deliver the course; tools that were used included MATLAB, MATLAB documentation and online videos and course notes developed and created by the previous lecturer. These notes are currently used.

3.1 Course Notes and Programming Documentation

CHMT2011 includes well-developed course notes with additional reference material, such as MathWorks Documentation (The MathWorks, Inc., 2025). Documentation pages are sources of help for learning about and working with MATLAB software.

3.2 Self-Paced Online Training Courses

The integration of MathWorks online courses (The MathWorks, Inc., 2025), with the Wits LMS, meant that an array of online courses were accessible to students. These courses covered programming topics such as,

core programming skills, data visualisation, artificial intelligence, modelling and simulation. Two courses from this suite were used:

MATLAB Onramp: A 2-hour introductory course focused on the basics of programming, commonly used functions and workflows, i.e., navigating the interface, saving, creating files in different formats.

Solving non-linear equations with MATLAB: A 3-hour course focused on the basics of solving non-linear equations within MATLAB. This includes root-finding methods to solve nonlinear equations thus preparing students for problems they will encounter in the engineering curriculum.

3.3 Automated Assessments

Automated marking of coding assessments ensures that students practice coding skills via a platform that provides instant feedback on their performance of the task. MathWorks offers an automated assessment tool, MATLAB Grader (The MathWorks, Inc., 2025) which was integrated into the Wits LMS, where students access assessments, submit their code, receive instant feedback on the correctness of their submissions, and are graded automatically. The coding assignments were provided by a MathWorks repository of existing ready-to-go problems and tests, known as Assessment Items (The MathWorks. Inc., 2025).

3.4 Online Videos and playlists

In previous years of instruction, Dr Ming included online videos, which he created, in the course notes. Following this, Dr Moodley created additional video playlists as supplementary material to classroom activities and tutorials. Students accessed videos via the course notes. Topic-specific videos were shared on a weekly basis.

4 Implementation of Interventions

4.1 Evaluation Methods

Table 1 (Section 8,Appendix) maps out a high-level view of a 5-year period of implementation; summarising the intervention tools used per year as well as the learning outcomes. The choice of intervention depended on (a) method of teaching e.g. lockdowns in COVID meant that all teaching was done online, (b) the size of the class e.g. average class size of 150 students meant that each year required slightly different approaches. (c) changes within the faculty and CHMT.

4.2 Evidence of Impact

A survey was sent out in 2021, to students who completed the CHMT2011 course in 2020 and 2021. The survey aimed to (1) collect anecdotal feedback of student's learning experience, (2) to determine the efficacy of interventions and (3) explored students' knowledge of graduate employability skills required for industry.

Since the survey in 2021, new cohorts of students have participated in the updated version of the course. Apart from University-Wide course evaluations, new student cohorts have not taken part in formal surveys for this particular course. The survey, distributed as a Microsoft Form, contained 14 questions, with an average completion time of 9 minutes. 22 students completed the survey and provided feedback.

The class of 2020 was the first to complete the MATLAB Onramp, where 82% of the 2nd year cohort had little to no experience in programming due to the absence of programming courses in first year or at high school. When asked about the use of the MATLAB Onramp course, some students reported it as "Challenging" or "Difficult", and 50% of students reported an "Exciting experience":

"The Onramp course was so helpful, and it wasn't pure video. The interactive part helped me understand what was being conveyed."

In 2021, when MATLAB Grader was integrated into the Wits LMS, students submitted coding assignments and received instant feedback on the correctness of their submissions. When asked about the experience of receiving instant feedback on coding assignments, students reported that they enjoyed the platform:

"The instant grading was really good as it did not give me the anxiety of waiting for results".

When asked about their opinion on employability skills needed upon graduation, and the usefulness of learning programming skills for solving engineering problems; all surveyed students answered that programming skills were vital for seeking jobs. A portion of students also commented that, upon further research there were many more programming languages that they should learn and that they feel confident in doing so now that they have *"covered the basics"*:

"I went for a job interview and was asked about my computer proficiency and the skills that I learnt, and this course was definitely a talking point and interested the interviewer. Thank you for the course as it truly equipped me with valuable skills and I've always procrastinating learning to code, but this course helped me start that journey!"

Overall, the sentiment from students was that the ability to program was important for problem solving, specifically when it comes to thinking about and solving engineering problems:

"It was quite interesting to use programming to solve engineering problems. And just developing the skill of programming, which I think is valuable in many different domains."

Upon completion of the CHMT course, students believed they were more competent in programming and felt empowered to expand their programming skills:

"Provided a good foundational understanding of MATLAB programming and an exposure to several scenarios in which MATLAB programming could be used to solve an interesting problem. It was beneficial for the CHMT2011 work and understanding the fundamental of programming as a skill in general."

5 Discussion

Navigating the transition from classroom to online teaching was a significant challenge during the COVID-19 lockdown. Student and staff access to resources was severely restricted and collaborative effort with industry partners slowed. Specifically for course CHMT2011, internet access and use of a personal computer was not possible for many students. While the university provided such resources for many students, peer-learning and engagement with lecturers remained an essential aspect of the university experience. From the perspective of the student, transitioning from the physical classroom to digital instruction was difficult and challenged their entire "way of learning".

To complicate matters further, students were faced with learning programming. In the case of MATLAB, the logic used in problem solving is similar to that of spoken English and the syntax is relatively easy to learn. However, English is typically a Wits student's second or third spoken language which meant they were faced with learning a new language as well as the ability to convert logic and problem solving into readable code. Having uncovered this, lecturers were encouraged to emphasize problem solving over the word "coding" so that students focus on the engineering problem and become syntax agnostic.

The intervention tools and their implementation presented in this paper focus on the resources available from MathWorks. It is possible to take a similar approach with any programming language within any engineering degree. In South Africa, other branches of engineering traditionally emphasise programming throughout a student's four-year degree. Historically, for chemical engineers this has not been the case. This must change for students to be better equipped for a working world that is valuing computational thinking and associated skills (World Economic Forum, 2023).

At this stage the student is aware of basic engineering principles, equations and formulae, and is able to solve specific problems without the use of code. By taking similar problems with similar engineering concepts, the student is able to use programming constructs to arrive at a solution that was previously iterated by hand. The challenge however is to efficiently combine and stagger concepts in engineering and programming, such that students can synthesis the information and engineering specific lecturers are not lecturing programming principles. This is still under consideration.

Given the success of the automated assessments and self-paced course interventions, we believe it is necessary for students in this discipline to be exposed to programming outside of this second-year course. By including 'programming for engineering problem solving' in other courses, students can solve course specific problems and build better computational thinking skills. Discussions with lecturers are ongoing to present a more coherent curriculum that creates continuity for computational thinking. This would involve re-introducing fundamental engineering problems in CHMT2011 but asking the student to solve it using a programming language.

6 Conclusion

This work presents industry-academia collaborations over a 5-year period. Whilst the COVID pandemic catalysed this shift, engineering faculties can benefit from reassessing delivery of courses and aligning learning content to better suit the current industry climate. We have shown that by leveraging existing and new digital tools, it is possible to meet and expand on learning outcomes in university courses that are tailored toward developing computational thinking, such as the second-year computers course in this practice paper. Including a computational thinking approach meant that the focus moved from 'learning to code' to 'problem solving using a programming language', thus reframing student's perspective of the course.

The results of the student survey show that these interventions have positively impacted students learning, and beyond this has encouraged them to leading their learning despite many challenges. Self-paced online courses, integrated LMS-assessments and instant feedback enhance student learning and support the computational thinking paradigm. Additionally, by approaching each student cohort differently, we are able to improve each year from lessons learnt the previous year. These improvements include adjusting the way assessments are conducted and the way new and existing tools are used as shown in this practice paper, all while still meeting course outcomes and university quality protocols regarding engineering courses.

To further support the work here, the authors are now involving other educators in the engineering curriculum to include a programming element in their course where students can solve specific engineering problems and enhance their computational thinking skills. This will create a steady thread for computational thinking that previously was broken across the years. An initiative of this scale however requires careful planning within the discipline curriculum and is currently still under consideration.

7 References

Etter, D. (1993). *Engineering Problem Solving with MATLAB.* New Jersey: Prentice-Hall.

School of Chemical and Metallurgical Engineering, C. (2024). CHMT2011 Computing for Process Engineering Course Brief.

Selby, C. a. (2013). *Computational thinking: the developing definition .* Southampton: University of Southampton.

The MathWorks, Inc. (2025, March 27). *Campus-Wide License*. Retrieved from MathWorks: https://www.mathworks.com/academia/campus/resources/campus-wide-license-products.html#resources

The MathWorks, Inc. (2025, March 27). *Documentation*. Retrieved from MathWorks: https://www.mathworks.com/help/index.html?s_tid=CRUX_lftnav

The MathWorks, Inc. (2025, March 27). *MATLAB Grader*. Retrieved from MathWorks: https://www.mathworks.com/products/matlab-grader.html

The MathWorks, Inc. (2025, March 27). *Online Courses*. Retrieved from MATLAB Academy: https://matlabacademy.mathworks.com/?page=1&sort=featured

The MathWorks. Inc. (2025, March 27). *Assessment Content*. Retrieved from MathWorks: https://www.mathworks.com/products/matlab-grader/assessment-content.html

Valerie J. Shute, C. S.-C. (2017). Demystifying computational thinking. *Educational Research Review*, 142-158. doi:https://doi.org/10.1016/j.edurev.2017.09.003

World Economic Forum. (2023). *Future of Jobs Report 2023.* Geneva: World Economic Forum. Retrieved from https://www.weforum.org/reports/the-future-ofjobs-report-2023/

8 Appendix

Note: Table 1 in Landscape on next page (pg 8).

Table 1: Timeline of the intervention tools

	Year 1: 2020	**Year 2: 2021**	**Year 3: 2022** *(Note 5)*	**Year 4: 2023** *(Note 5)*	**Year 5: 2024**
Course Notes	Yes	Yes	Yes	Yes	Yes
SELF-PACED ONLINE COURSES					
MATLAB Onramp *Learning Outcomes 1-3*	Used as an assignment (toward course mark)	Used as an assignment (toward course mark)	Used as an assignment (toward course mark)	Used as an assignment (toward course mark)	Used as a pre-requisite (does not count toward course mark) (*Note 1)*
Solving non-linear equations with MATLAB *(Note 6)* *Learning Outcomes 1,2*	Assignment	Assignment	Assignment	Assignment	Assignment
AUTOMATED ASSESSMENTS					
MATLAB Grader *Learning Outcomes 1-4*	No	Assignment	Assignment	Assignment	Tutorials, tests and exams
MATLAB Grader Assessments *Learning Outcomes 1-4*	No	No	Exams	Exams	Tutorials, tests and exams.
Online videos and playlists *(Note 2)* *Learning Outcomes 1-4*	Yes	Yes	Yes	Yes	Yes
COURSE MARK DISTRIBUTION (*Note 3)*					
MATLAB Tutorials *(Note 4)* *Learning Outcomes 1-4*	0	0	0	0	15 (10)
Assignments (%)	25 (2)	25 (2)	25	25	15 (1)
Tests (%)	25 (2)	25 (2)	25	25	30 (1)
Exam (%)	50 (1)	50 (1)	50 (1)	50 (1)	40 (1)

Continue to next page

Key learning outcomes:
(as per the course brief)

1. Understand the core principles underlying any programming language and understand how to execute a computer program.
2. Understand the basics of primitive data types, program control flow, program logic, and functions.
3. Plotting data and visualization using MATLAB.
4. Be able to apply these concepts and create MATLAB scripts that solve engineering problems.

Notes:

1. In 2023, the faculty introduced "Introduction to Programming with MATLAB", a 6-week course for the first-year engineering students. This meant that students would have covered key learning outcomes before they entered CHMT2011. MATLAB onramp was encouraged to refresh skills, no marks were awarded.
2. These consisted of online videos and playlists, one of which was created by Dr Ming.
3. Total course marks add up to 100%. The number in parenthesis is the number of assessments within that category. The percentage shown is the total for that category.
4. MATLAB tutorials refer to a collection of engineering problems from MATLAB Grader. Students were given a defined number of attempts to solve the problem and submit within a given timeline. As the difficulty of the problems increased these parameters were adjusted. Consideration was given to special class circumstances.
5. In 2022 and 2023, the course was delivered by a part-time sessional lecturer. The data supplied is to the best of our knowledge.
6. The assignment consists of three parts, based on an engineering problem the student would have encountered in their first year. This includes mass balances with a reactor, separators and mixer components. Part 1 introduces the student to solving non-linear equations, part 2 asks the student to solve the problem using a flow chart and the 5-step engineering methodology (Etter, 1993). Part 3 is submitting code that solves for the mass balance and stream data across the process flow.