






ARTICLE

Student responses to creative coding in biomedical science education

Phillip Gough¹  | Oliver Bown²  | Craig R. Campbell³  |
Philip Poronnik³  | Pauline M. Ross⁴ 

¹Affective Interactions Lab, School of Architecture, Design and Planning, The University of Sydney, Camperdown, New South Wales, Australia

²Faculty of Art and Design, University of New South Wales, Kensington, New South Wales, Australia

³FMH Media Lab, Education Innovation, School of Medical Sciences, Faculty of Medicine and Health, The University of Sydney, Camperdown, New South Wales, Australia

⁴School of Life and Environmental Sciences, The Faculty of Science, The University of Sydney, Camperdown, New South Wales, Australia

Correspondence

Pauline M. Ross, School of Life and Environmental Sciences, The Faculty of Science, The University of Sydney, Camperdown, New South Wales, 2006, Australia.

Email: pauline.ross@sydney.edu.au

Abstract

Biomedical science students need to learn to code. Graduates face a future where they will be better prepared for research higher degrees and the workforce if they can code. Embedding coding in a biomedical curriculum comes with challenges. First, biomedical science students often experience anxiety learning quantitative and computational thinking skills and second biomedical faculty often lack expertise required to teach coding. In this study, we describe a creative coding approach to building coding skills in students using the packages of Processing and Arduino. Biomedical science students were taught by an interdisciplinary faculty team from Medicine and Health, Science and Architecture, Design and Planning. We describe quantitative and qualitative responses of students to this approach. Cluster analysis revealed a diversity of student responses, with a large majority of students who supported creative coding in the curriculum, a smaller but vocal cluster, who did not support creative coding because either the exercises were not sufficiently challenging or were too challenging and believed coding should not be in a Biomedical Science curriculum. We describe how two creative coding platforms, Processing and Arduino, embedded and used to visualize human physiological data, and provide responses to students, including those minority of students, who are opposed to coding in the curriculum. This study found a variety of students responses in a final year capstone course of an undergraduate Biomedical Science degree where future pathways for students are either in research higher degrees or to the workforce with a future which will be increasingly data driven.

KEYWORDS

Arduino, biomedical science education, creative code, data visualization, processing

1 | INTRODUCTION

Over the last decade, major reports have identified quantitative analysis, mathematical reasoning, and computational

Philip Poronnik and Pauline M Ross are equal authorship to this study.

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial](https://creativecommons.org/licenses/by-nc/4.0/) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2022 The Authors. *Biochemistry and Molecular Biology Education* published by Wiley Periodicals LLC on behalf of International Union of Biochemistry and Molecular Biology.

tools as core competencies for biomedical education.^{1–5} Critical in this is coding. Coding is the process of transforming, summarizing and representing collected data into meaningful representations using statistical software. Without coding, data manipulation and modeling, visualization and communication is more challenging.^{6–9}

Biomedical science graduates who successfully learn to code in this era of big data will have an advantage. To effectively learn to code, however, requires experimentation and iteration.^{7,15,10–15} As a consequence, in more recent times undergraduate curriculum reviews have been focussed on embedding quantitative analysis, mathematical reasoning and computational tools in multiple locations.^{1,2,5,16} Coding skills provide biomedical science students the tools needed to model and understand the emergent properties of complex biological systems.² Coding skills are also more likely to lead biomedical science students to graduate research and employment. Jobs requiring coding skills are also increasing faster than the overall job market and this is not just for those graduates who are expert coders.⁸

Many biomedical science students, however, also experience anxiety and have poor self-efficacy in quantitative skills. As a consequence, many do not even attempt quantitative problems.^{3,17–22} Studies have found an increase in student anxiety and quantitative problem solving, which can be, but is not always dependent on the level of mathematics attempted at high school and competency level.^{3,5,17} Student perceptions that coding and quantitative skills in general are uninteresting or irrelevant limit the development the undergraduate biomedical science curriculum.^{3,4} Further, biomedical faculty can also feel ill-equipped and unmotivated to embed coding into the curriculum.⁶ Educators are apprehensive about introducing coding to biomedical classes because they lack experience argue that the curriculum is already full and if coding is introduced then something else must be taken out.^{12,23,24}

Coding can also be challenging for biomedical faculty whose skill sets may not necessarily align with the diversity of student quantitative and data skills. Like students, faculty can experience anxiety with the range of student responses, which range being overwhelming too not sufficiently challenging. This diversity occurs because biomedical science students do not always elect to take courses that teach coding.⁷ Apart from anxiety, other reasons for the lack of attraction for students include the perception that coding lacks interest and creativity. Coding, however, is inherently creative^{25,26} and creativity is an essential part of learning science.²⁷ Creative coding is a potential medium of creative expression, which may allow students to overcome their mathematical and computational fears and increase motivation.^{28–30} Creative

coding while not distinct from other approaches to coding allows more emphasis on exploratory search and discovery. Creative coding also allows greater access to visualization and representation of data and has the advantage of being more meaningful to non-expert audiences.³¹ As a result, creative coding has the potential to be an accessible entry point to teach coding skills in biomedical science education, rather than just in art design and computer science.³²

If biomedical students are to effectively explore research data and contribute to scientific discourse, then they must learn how to code. Given the diminishing gap between life and computational science, coding skills are becoming essential for graduates.⁴ Those students who code can be the next generation of successful biomedical scientists and graduates will have digital fluency in a time of technological tsunami.²⁸ Addressing this ongoing issue is arguably becoming critical. Already a significant period of time has elapsed since the National Research Council¹ and the Vision and Change in Undergraduate Biology Education² campaigned for an increase of quantitative skills in undergraduate biosciences curriculum stated that “connections between the biological sciences and the physical sciences, mathematics and computer science are rapidly becoming deeper and more extensive”.¹

AAAS stated two of six core competencies are the “ability to use quantitative reasoning and the ability to use modelling and simulation”.^{2, p.14} Statements from the UK by Koenig³ emphasized that the key issues were “ensuring that all bioscience graduates have the confidence and understanding to participate in increasingly quantitative and interdisciplinary research”. In Australia, “mathematics and quantitative skills and interdisciplinary perspectives of students” are critical to embed in Australian curricula.^{33, p.12} Questions remain, however, whether students sufficiently master skills in data analysis to enter the workforce with confidence. The aim of this study was to evaluate the embedding of creative coding in a biomedical capstone course by an interdisciplinary faculty team, to determine whether it develops students’ skills through a quantitative and qualitative measurement of student responses to guide future curriculum designs.

The pedagogical approach used here makes the concrete experience of coding central to the learning process, with students engaged as participants in the coding activities in a task-oriented context.³⁴ Experiential learning pedagogies have been used to teach coding skills,^{35,36} to students who are learning coding outside of computer science and engineering fields.³⁷ Experiential learning provides both breadth and depth of learning experience and encourages higher-order thinking skills³⁸ about the applications of technology in the biomedical space,

particularly for these students who were in their final year of undergraduate studies. This approach allowed students to develop better relationships through collaborative problem solving³⁸ and social constructivist theory.³⁹ The learning activities included four elements, similar to Morris' revision of Kolb's model of experience-based learning³⁴ (1) *A concrete experience*, providing a breadth of experience within the context of the unit, (2) *reflection* on action through a reflexive written activity, (3) *abstract conceptualization* by creating and testing a working hypothesis; and (4) *active experimentation* by executing their own code. Social constructivist pedagogies by Vygotsky³⁹ were also used in the design to ensure that social interactions and collaborations build cognitive understanding. In addition to creating an experiential learning activity, to introduce students to coding, faculty demonstrated by trial and error the writing code in real time in front of the class.⁴⁰ This strategy introduced students to the syntax of writing code, and the thinking process for writing code⁴¹ and is shown to be both effective and preferable to students studying introductory coding⁴² compared to other approaches. Further to the experiential learning cycle and collaborative process, live coding encourages students to write code through iteration, which has the advantage of allowing learning by mistakes and building student confidence as well as introducing necessary knowledge about structures and syntax of coding.⁴²

2 | METHODS

2.1 | Learning outcomes

Creative coding activities were embedded over a nine-week two-year period into a final year capstone course in a Biomedical Science program in a large metropolitan research-intensive university. Already in the first and second year of the three-year Medical Science program students had experiences with data analysis and coding in R. At this university "Information and Digital literacy" and "Inventiveness" form two of nine graduate qualities that students are expected to achieve. The learning outcomes also included the achievement of the graduate quality for information and digital literacy, that is, "Information and digital literacy is the ability to locate, interpret, evaluate, manage, adapt, integrate, create and convey information using appropriate resources, tools and strategies". There is also a graduate quality called "Inventiveness", which is defined as "generating novel ideas and solutions". Additionally, specific learning outcomes for this activity in this course were that students should be able to:

- i. Synthesize research findings from your practical classes and relevant literature to report the data in the appropriate format for scientific communication
- ii. Communicate data and scientific findings to diverse audiences

2.2 | Learning design and implementation

In the first year of this study, we used a widely used creative coding platform based on the Java programming language called Processing. Processing aims to make computational expression available to anyone. For example, Processing is used by computer scientists to make art and by artists to write code.^{26,43} For biomedical students it provides an entry point to coding through quick, iterative development of a creative representation of data. In Processing, code files are referred to as "sketches", which allows an iterative approach to writing code and solving computational problems, which we wished to encourage with our students.⁴⁴ Code sketches are readily accessible, and parameters can easily be altered by students to create unique and interesting results based on data. Processing has an active online community, which has created many libraries, tools, examples, and instructional resources to help beginners. An outstanding example of the use of Processing to learn science is *The Nature of Code*,⁴⁵ an online book and learning resource that uses creative coding to demonstrate the principles of physics. Students were given a brief outline of their role as a "data detective" exploring a data set of normal human physiology of an open dataset from the Centre for Disease Control (US). Students were introduced to the basics of Processing and principles of data visualization. Following this, students created code sketches with the imported data set of human physiology to explore and interrogate through data visualization. Students were provided with Sketch templates so they did not have to write code from scratch – rather they were able to apply their understanding of coding basics to easily alter many variables and parameters to create unique outcomes. The learning environment mirrored a design studio, where students worked in small collaborative teams to address an open-ended question.^{46–48} Students were asked to create data visualization, which would be understood by a "normal" person. From this, student groups produced a visualization and a short narrative for assessment. Through these data visualization rules were in the hands of the students, allowing them to experiment with representations of data, apply general rules as they saw fit, but also adapt them to a more abstract, creative representation. The goal of incorporating Arduino was to allow students to

capture their own data and visualize it, rather than simply using an existing dataset.

In the following year, students collected their own human physiological data using the Arduino UNO platform and Processing. Arduino is an open-source electronic prototyping platform and coding language that enables users to create bespoke, interactive electronics.⁴⁹ Platforms that combine physical computing devices, such as Arduino, BBC Micro: bit or Lego Mindstorms, are known to develop students understanding of the abstract programming concepts and processes.⁹ Prior to the collection of data, students created a hypothesis on homeostasis and used two of the available sensors: EMG (Electromyogram), temperature sensor, pulsimeter or a 3-axis accelerometer to measure human physiological variables. Following this they used Processing to visualize the data.

The activities occurred in practical laboratories of the final year of a Medical Science program and were jointly designed by faculty working in interdisciplinary teams from the Schools of Medical Science and Architecture, Design and Planning. Our approach was to introduce basic coding skills through live-coding demonstrations. This method teaches the syntax (rules) and output of coding by the teacher writing actual code from scratch, making the instructors thinking visible to the students.⁵⁰ Students were able to follow along with activities, as sample data and templates were also provided. Coding workshops introduced three concepts: (1) loading data, (2) mapping or scaling data and (3) using the mapped data to manipulate a visual attribute. All of these tasks are closely related to the goal of visualizing data. Students were shown how to load the data into Processing from .csv text files, conduct basic calculations (such as using a loop to find the mean of a data set), map the data from the data scale to an appropriate scale for drawing, and then translate the mapped data to a visual attribute, such as the color, length or position of a graphical object. By following along with the instructor, students were able to create animated graphs and abstract illustrations that visually represented the qualitative data they were given. Students were also introduced to the foundational principles of reading values from analogue sensors using Arduino, and how this can be recorded by a computer as a .csv file. They loaded, mapped and used these data for a visualization.

2.3 | Student demographic data and scheduling of classes

This study occurred from 2018 to 2019. In 2018, there were 203 students, with 129: 74 females to males, which is a gender breakdown of 64:36%. In 2019 there were 132 students in the course with 85: 47 females to males, which is an

identical gender break down of 64: 36% female to males, respectively. Being prior to COVID-19 there were more domestic students compared to international students. In 2018 and 2019, there was an identical ratio of 85:15% domestic and international students, respectively. All were on campus students, attending face-to-face classes and not learning on-line. This creative coding exercise with Arduinos was included in the curriculum to replace an alternative to an already existing data coding exercise. There were three compulsory classes, which were scheduled in each year in the form of workshops over a three-week period. These scheduled face-to-face classes were in weeks 6, 7, and 9 of a 13 week semester. Extra sessions were also provided for those students who wanted additional assistance and feedback prior to the assessment submission. Evaluation of the success of the creative coding workshops was assessed by the video group assessment, which was worth 10% of the total course marks and involved students communicating what they had learnt by making a video.

2.4 | Student surveys

To evaluate student responses to these activities, quantitative and qualitative questionnaires were used to survey students. The quantitative survey included 10 multiple choice questions (5-point Likert scale) and the qualitative survey included three open response questions where students identified the best aspects, areas of improvement, and gave other comments (Table S1). In the first year, there were 112 responses from 170 students who attempted the 10-question survey with a response rate of 66%. In the second year, there were 70 responses from 132 students who attempted the survey in the class with a response rate of 53%. The questions in the surveys were similar in each year, although in the second year, question 7 was updated to include both Processing and Arduino, that is, "Following these classes I can see many opportunities and possibilities to use Processing/Arduino to display sensor data in health applications" (Table S1). There were 160 open-ended responses, 71 comments on the best aspects of the course, 70 comments on improvements, and 19 other comments (Table S1). This study was approved by Human Ethics Committee (2017/751).

2.5 | Data analysis

Quantitative survey data were analyzed using K-means clustering. This is an unsupervised, discrete machine-learning algorithm that groups data into a selected number of discrete clusters, each representing a heterogeneous group.⁵¹ We selected the number of clusters based

on a silhouette analysis using the Python v3.8.5 and scikit-learn v0.23.2 in Jupyter Lab. The silhouette analysis identifies whether clusters are accurate by showing how well each object (in our case, each student) has been classified: how similar it is to the cluster it is assigned to, when compared to other clusters. This allows us to determine the number of clusters that are in the data. The clustering algorithm ran nine times, first to create two groups, then three, and four, etc., up to 10. This showed how accurate the clusters are, as students were divided into different numbers of clusters. We chose to run from two to 10 clusters, because a general rule-of-thumb is that the number of clusters (k) for a given number of objects (n) should be approximately equal to the square root of half n : $k = \text{sqrt}(n/2)$. As the clustering does not allow for incomplete or missing data, we were only able to run the algorithm with 171 student responses (Year 1, $n = 107$; Year 2, $n = 64$), giving an expected eight clusters. The final number we chose, based on the silhouette analysis, was that there are five clusters present in the data. Further increasing the number of clusters only increased the inaccuracy of the clustering. To understand the relationship between questions in the survey, and to investigate whether the year (and therefore the approach we used) was a significant driver of the student responses, we calculated correlations between survey questions, year and clusters. These were calculated using the SciPy v1.5.2, the statistics package that is part of Python 3.8. We used a Spearman correlation test, as our data were categorical

rather than normally distributed, according to a Shapiro-Wilk Test (statistic = 0.878, $p < 0.0001$: the very low p -value indicates the data is not normally distributed).

Qualitative comments were thematically tagged comments and analyzed using an affinity diagramming method.⁴⁸ This process involves the researchers grouping thematically related comments and producing emergent themes from these groups. Responses from students who responded to both the quantitative survey and open-ended answers were tallied by cluster and emergent themes categorized. For this activity Q11 (three best aspects of this activity) had 129 comments in total and Q12 (three improvements for this activity) had 92 comments in total. The responses to each question were analyzed separately identify thematic areas independently. The number of responses to each emergent theme represents the proportion of students whose responses was grouped into each theme.

3 | RESULTS

The mean scores for both years revealed very little difference between the cohorts (Figure 1). Students generally were not confident with coding (Q1, mean = 1.7, SD = 1.1) and that Processing was challenging to learn (Q3, mean = 4.4, SD = 0.8). Students reported that coding should be introduced earlier in their degree program (Q8, mean = 1.7, SD = 1.0). When asked what year of

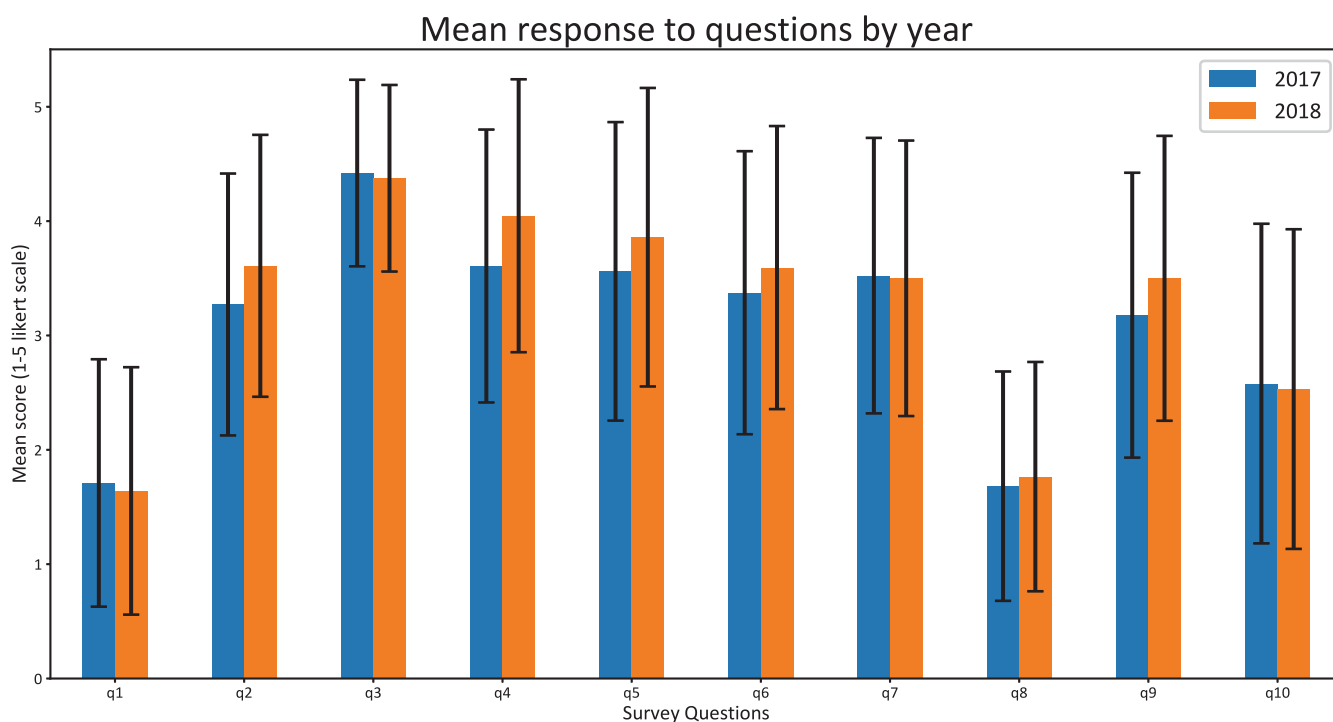


FIGURE 1 Mean response to 10 survey questions in classes in 2017 and 2018

study this should be introduced in the curriculum they stated never. Students showed little interest in taking up creative coding as a hobby (Q10, mean = 2.6, SD = 1.4) and responses to other questions were between 3 and 4 (Q2, Q4–Q9 Figure 1). Between the 2 years, there were no significant differences.

Clustering revealed groupings of responses (Figures 2 and 3). Cluster 1 and 5 had the greatest contrast of responses, while clusters 2, 3 and 4 were relatively the same. Cluster 1 comprised 15% of the students ($n = 25$) and included students who had confidence in coding before the creative coding activities

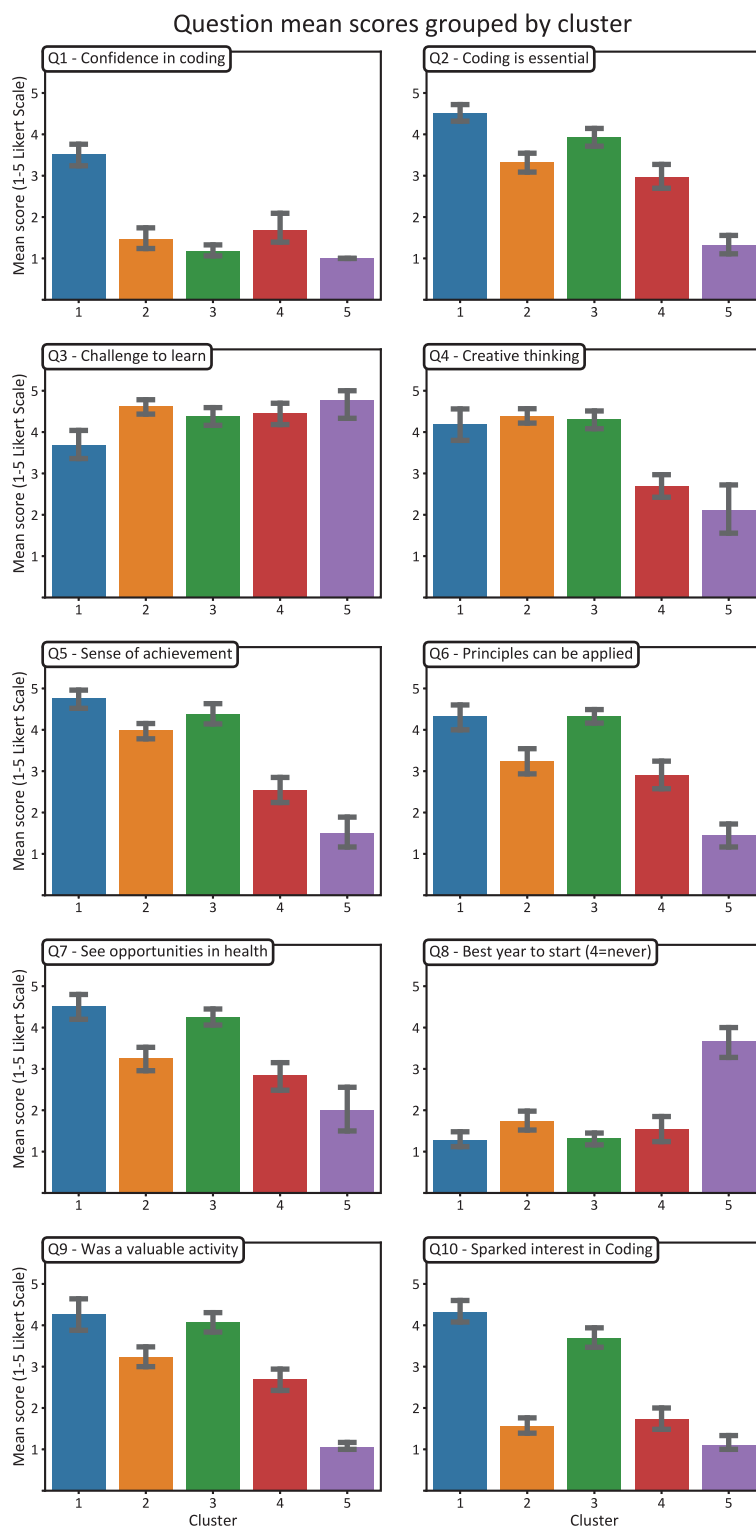


FIGURE 2 The mean response to each question grouped by cluster identified from the K-means algorithm

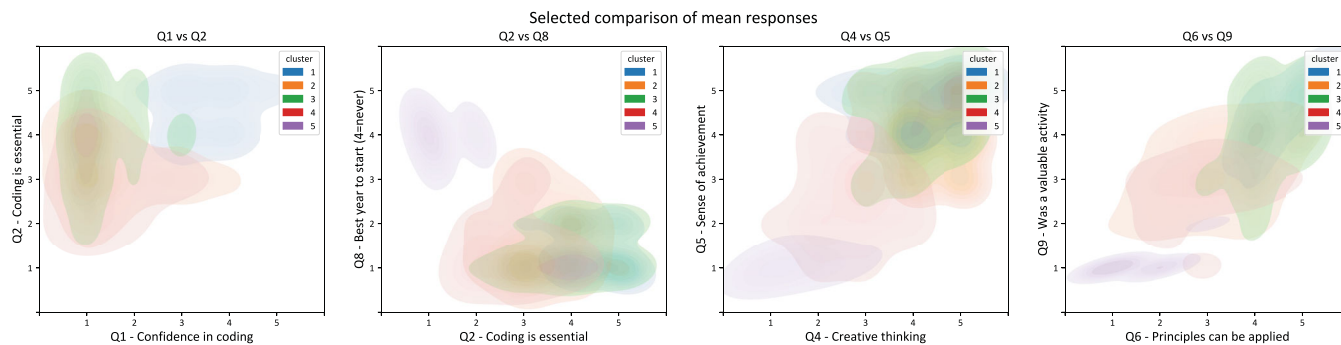


FIGURE 3 The mean answer of each cluster to selected questions, clockwise from top left: Q1 and Q2, Q4 and Q5, Q2 and Q8, Q6 and Q9

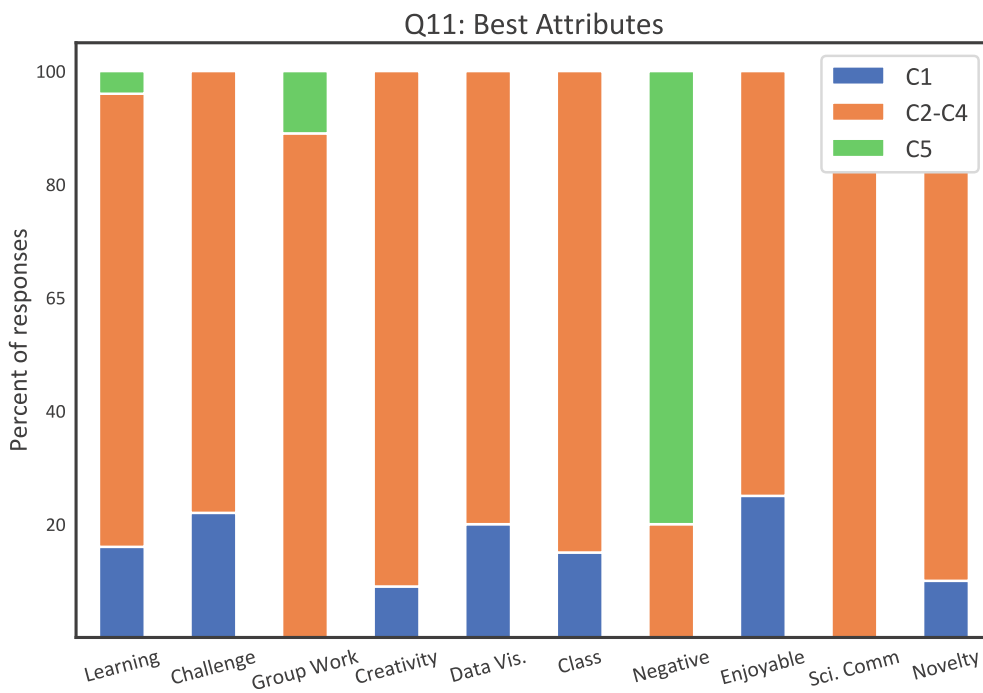


FIGURE 4 The percent of responses for open-ended question of the best attributes (Q11) of the activity grouped by cluster

(Q1). These students appreciated the activity and were competent with coding (Figure 3). Their qualitative feedback indicated they enjoyed the novelty of the activity and the opportunity to think different (Figure 4) and wanted to learn more about functions in processing:

Its like learning a new language, its fun and interactive, enjoyable. (Q11, best aspect)

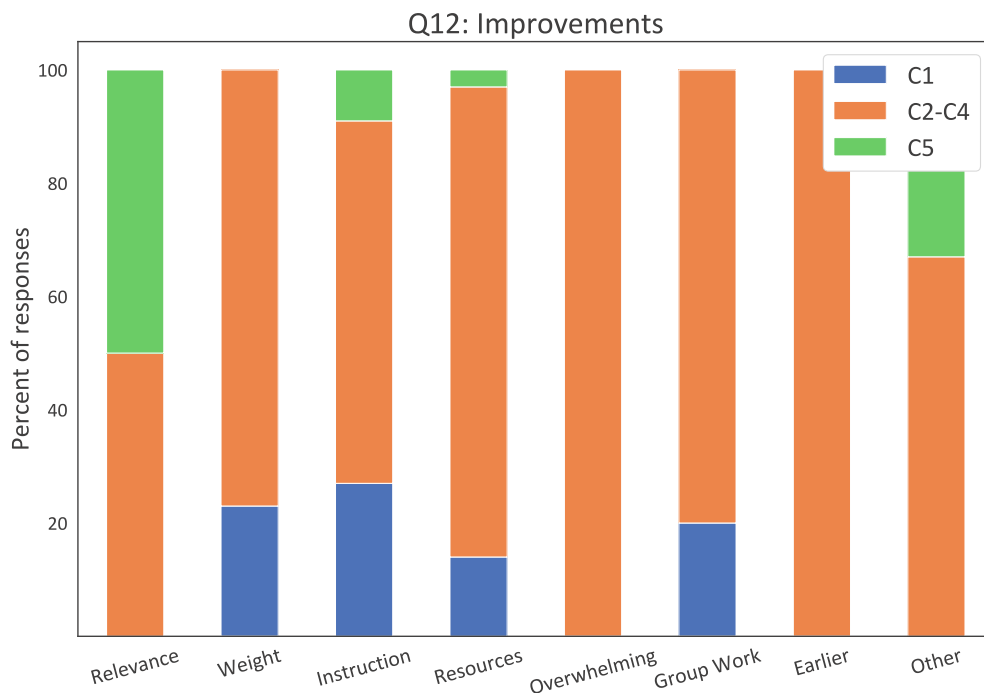
Changing gear to think differently. (Q11, best aspect)

Maybe focus less on the graphic design/creativity and more on different functions in processing. Give more examples of places it has been used. (Q12, area of improvement)

The areas of improvement identified by Cluster 1 were around the need for more detailed instruction and that the time required to complete the task, in comparison to the assessment weight was too low (Figure 5).

Cluster 2 with 27% of students ($n = 46$), cluster 3 with 29% of students ($n = 49$) and cluster 4 with 19% of students ($n = 33$) were similar to each other. Clusters 2, 3 and 4 found the coding challenging, and generally agreed that principles of data visualization could be applied in creative code (Q6), and there were opportunities for creative code (Q7) (Figures 2 and 3). Students in cluster 4 were similar to cluster 5, with the exception that they believed that coding should be introduced early in the degree (Q8, mean = 1.5, SD = 0.9) (Figures 2 and 3). Open-ended responses to Q11 on the best aspects of this activity indicated that students both found it challenging

FIGURE 5 The percent of responses for open-ended question needs improvement (Q12) grouped by cluster



and creative and the capacity to communicate science (Figure 4):

Thinking creatively given the data structure and what will be showcased. Very high learning curve which is difficult but fun. (Cluster 2)

Able to implement it in science for obvious purposes, enjoyed data viz aspect of processing in science, challenging. (Cluster 3)

Finding different ways to visualise data, trying to figure out what's wrong with the code, trying to figure out what was actually right with the code. (Cluster 3)

Exploring unique and intriguing way of communicating science to science and lay people, interesting (I loved the pressing that allowed sound with your mouse!), learning a new albeit difficult skill that I never before believed I would be capable of learning. (Cluster 3)

Learning a new way to represent quantitative data in a non-numerical way, learning how to code. (Cluster 4)

Students commented that to improve the activity required more support, because it was overwhelming and

should have been experienced earlier in the curriculum (Figure 5). Comments included:

Learning it earlier, pretty tricky, maybe have more interactive tutorials before jumping into assessment. (Cluster 2)

Provide processing at earlier stage and allow increasing difficulty, insufficient sessions, and all-encompassing textbook or notes to refer would be great for self-study e.g. if I wanted to animate my data it would be great to know of a recommended source to retrieve code from. (Cluster 3)

Step-by-step programming instruction, an intensive tutorial, applying to variety of data. (Cluster 4)

I was very worried about the coding assignments at the start of the semester. But now, I've actually come to really enjoy them. Although challenging at times, these assignments have been one of the best parts of this unit. (Cluster 3, Q13)

Students in Cluster 5 which represented 11% of students ($n = 18$ students) stood out because these students did not enjoy the activities and felt that creative coding should not be included in the curriculum (Figures 2–4; Q8, mean 3.7, SD = 0.8). This cluster also indicated that

coding should not be taught and was not relevant skill for graduates (Figure 5; Q2, mean = 1.3, S.D. = 0.5). They also indicated that they did not have the background for coding and that this required more instructions and resources to learn (Figure 5).

Teach as first year or as an elective subject -- not in human physiology, don't make something we did once worth so much of the grade. (Q12)

Whilst it was an ok assignment, we should not have learned it in human physiology, it should have been a course in first year. (Q13)

You need to add more computer science tutors for these classes - it's really hard to rely on 1/2 tutors when there are 30-40 students in the class. The pace should be slower - many students don't have the background for coding. (Q12)

There was no correlation between student responses and the year of implementation. Student responses were also not dependent on whether coding was only Processing or when Arduino and Processing were combined (Table 1). There was no correlation between the student confidence in coding before the activity, and their feeling that the

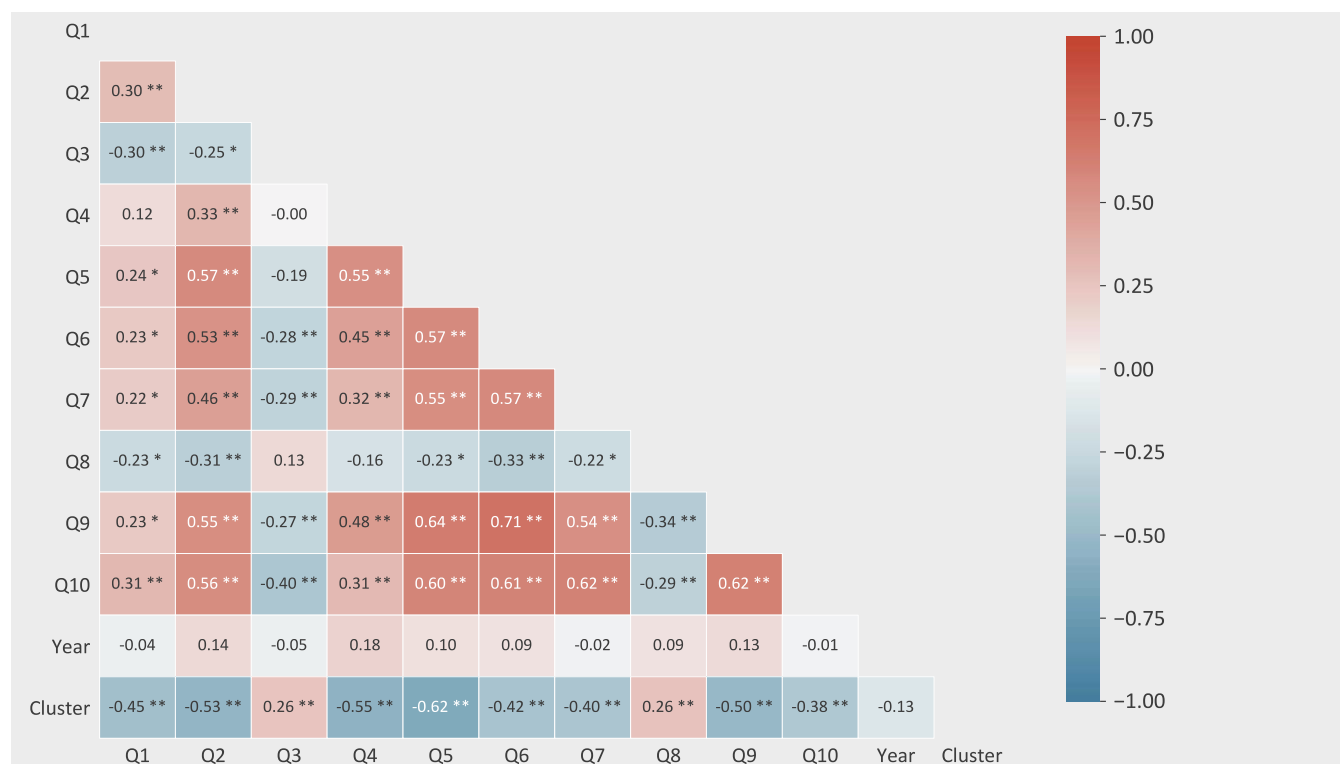
activity challenged their creative thinking (Spearman correlation between Q1 and Q4: $r = 0.11$, $p = 0.12$). That is, the students in cluster 1, who were generally confident in coding felt that their creative thinking was challenged (cluster 1, Q1 mean = 3.5, Q2 mean = 4.2). Students in clusters 2 and 3 generally reported a low confidence in coding, but also felt that their creative thinking was challenged (cluster 2: Q1 mean = 1.5, Q4 mean = 4.4; cluster 3: Q1 mean = 1.2, Q4 mean = 4.3). These three clusters represent 120 students, or 70% of the entire cohort. The varied levels of confidence in coding do not indicate that the students will not feel that their creative thinking will not be challenged. There was a strong correlation between the how students view data visualization and that the activity considered to be valuable (Spearman correlation between Q6 and Q9: $r = 0.71$, $p < 0.0001$). There was also a moderate correlation between creative thinking and a sense of satisfaction after completing the tasks (Spearman correlation between Q4 and Q5: $r = 0.54$, $p < 0.0001$). In contrast students in cluster 5 thought coding should never be included in the curriculum.

4 | DISCUSSION

Biomedical science curricula need to provide students with 21st-century employable skills. Overall, this study found that there were significant challenges to include

TABLE 1 Correlations between responses. * $p < 0.01$, ** $p < 0.001$. Negative correlation is shown in blue, and positive correlation in red

Q1													
Q2	0.30 **												
Q3	-0.30 **	-0.25 *											
Q4	0.12	0.33 **	-0.00										
Q5	0.24 *	0.57 **	-0.19	0.55 **									
Q6	0.23 *	0.53 **	-0.28 **	0.45 **	0.57 **								
Q7	0.22 *	0.46 **	-0.29 **	0.32 **	0.55 **	0.57 **							
Q8	-0.23 *	-0.31 **	0.13	-0.16	-0.23 *	-0.33 **	-0.22 *						
Q9	0.23 *	0.55 **	-0.27 **	0.48 **	0.64 **	0.71 **	0.54 **	-0.34 **					
Q10	0.31 **	0.56 **	-0.40 **	0.31 **	0.60 **	0.61 **	0.62 **	-0.29 **	0.62 **				
Year	-0.04	0.14	-0.05	0.18	0.10	0.09	-0.02	0.09	0.13	-0.01			
Cluster	-0.45 **	-0.53 **	0.26 **	-0.55 **	-0.62 **	-0.42 **	-0.40 **	0.26 **	-0.50 **	-0.38 **	-0.13		
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Year	Cluster	



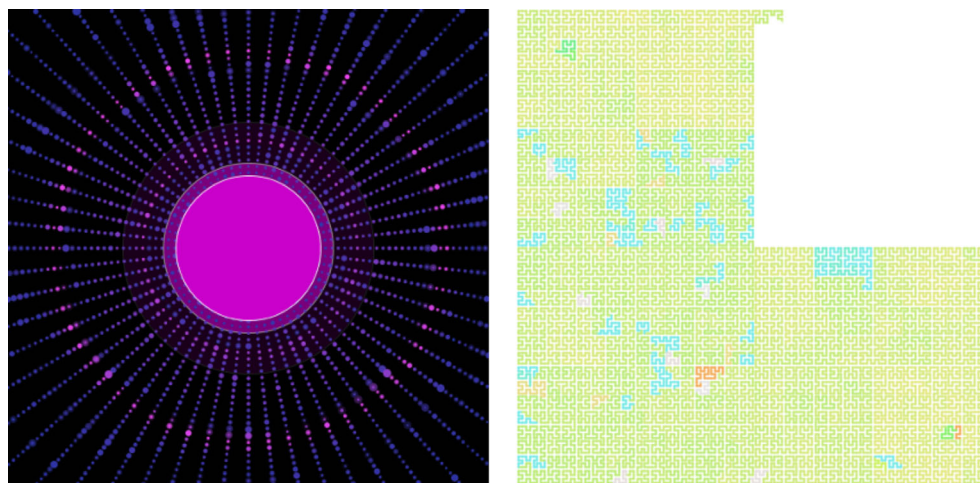
creative coding in the biomedical science curriculum. In this study, the majority of students while not confident and found creative coding challenging, were also able to see the benefits of coding and received a sense of satisfaction and achievement. There was a smaller but vocal minority who were unable to see the benefits of coding and did not want it in the curriculum, even at third- and final-year courses. The interdisciplinary team of faculty was essential for the activity. Faculty from the School of Medical Sciences provided expertise in the measurement of human physiology while faculty from the School of Architecture, Design and Planning provided expertise in creative coding and technical support. Developing coding skills in students can be a significant, and daunting challenge for biomedical faculty who often do not have expertise and experience in coding but who can provide the disciplinary expertise.⁵² This interdisciplinary model of curriculum implementation implemented here which combined skills of faculty across schools provided expertise in human biology and creative coding that overcame any deficit of skills that any one faculty team member may have if working alone. Researchers and educators have become increasingly aware that problem solving, and curriculum design requires the integration of expertise and skills across disciplines.^{1,16,48,53}

Coding activities described here will not transform biomedical science students into software developers. The strategy described here is to provide opportunities for students to understand and apply fundamental principles of coding. They were expected to manipulate code templates to load, scale and transform it to colors and shapes to create data-driven visualization rather than become experts. Creativity was important part of the process. Science is an inherently creative field, and creative code activities like those described here have the potential to develop skills in students that many may believe are beyond their capabilities.²⁷ An important feature of this activity is that students

learned that they could confidently navigate code and overcome any anxiety around creative coding being too difficult. The concrete tasks provided allowed students to gain the confidence in the fundamental concepts of code. Working in teams, also provided students with greater confidence to use coding and problem-solving. Regardless of whether students go on to be champion coders, their experiences will form the basis for quantitative and statistical competence, computational thinking and form a new relationship with data. Students will be better equipped to collaborate with and communicate their needs to expert informatics professionals.

The curriculum design used the Processing platform for creative coding. Processing was chosen over other creative code tools, such as P5js (web-based creative code using JavaScript) or D3JS (web-based data visualization using JavaScript), general code languages such as Python or mathematical coding such as R or MATLAB for two reasons. First, D3js, and to some extent P5js (the JavaScript implementation of Processing) require some additional knowledge about the way websites work (such as HTML and CSS) and may have created an additional barrier. Processing also allows students to load data and use fewer steps than P5js or D3JS. Further Processing presents a novel approach for the whole class and gives the students the ability to create something that is different from the kind of visualization that is difficult (or impossible) to create using tools designed for statistics or data visualization (Figure 6). This form of representation is better suited to the capabilities and flexibility of Processing and is beyond the capabilities of familiar software packages, such as Microsoft Excel. This gives students a clear and understandable goal for engagement. If students were drawing scatter plots with code, then they may rightly question the value of the exercise. Clear goals help students make decisions about where to expend their effort.⁵⁴ In the first year of design and delivery of

FIGURE 6 An example of the templates given to students. Left: An abstract representation of data, which is animated to show the movement of a pulse (pink dots) over time. Right: An L-system representing a timeline of data



this task, students used an already pre-existing dataset to select and visualize data. In the second year, Arduinos were used to measure physiological variables, which students selected and collected from themselves in groups. This had the result of increased engagement and gave students greater appreciation of how biomedical devices could be simply made to capture data.

The student's prior experience seemed to make some difference to what they enjoyed about this exercise. Clusters C1 and C3 are similar but can be set apart by C1 having much greater confidence in coding. They also had a view of coding being essential, experiencing less challenge to learn the Processing platform, and a stronger feeling that their interest in creative coding was sparked. The experienced students also can be set apart through their open-ended responses. The experienced C1 students commented that they enjoyed using Processing because it allowed a lot of creative thinking, to use their skills in a creative way. C1 students also stated is "easier to debug than other software", and that the visual output "allows 3D" – which we did not teach, but a student literate in coding would easily be able to find online resources to help achieve this. In contrast, the students in C3 who were less experienced with coding had more positive comments related "being able to present data in novel ways, promoting creativity - something often neglected in a Bachelor of Science" and that had their first experiences with coding. The key area of improvement that differed between the groups was that students in C3 requested further in-class instruction on fundamentals of coding, which is expected as they were less experienced. It is clear that coding should be introduced earlier into the curriculum. The challenges faced by Biomedical sciences curriculum, is that often the first and second year of the curriculum is more generic and covers fundamental concepts, but often neglects more specialist and more contextual learning designs. We need to have more open to conversations with a wider range of faculty about the importance of embedding critical skills like coding earlier on in the curriculum. Unfortunately, attempts to have conversations which create changes in the curriculum are often stymied by faculties and disciplines which are defensive and instead want to keep the traditional approach and "service teach" to large and diverse sets of students. There is an argument for a curriculum, which better serves a smaller more contextualized set of students, which also builds the cohort experience earlier on.

Coding together with data visualization (Figure 6) allows ideas to be generated and communicated.⁵⁵ Creative coding has great potential to develop coding skills in biomedical science students because it first flattens the learning gradient for absolute beginners, and then

simplifies the cycle of development and testing. Both of these factors help to maintain the all-important control and creative flow required to undertake exploratory engagement with relevant contextualized data. The exploration of human physiology data through creative coding without worrying about the underlying "nuts and bolts" of computational analysis can then be framed in terms of "what are we trying to ultimately achieve?" as opposed to the anxiety of "what are we going to learn next? or "what if I cannot do the next task and fail?" This was a solution for the majority of students in the class. The small minority of students who were against coding were, however, very vocal about their dissatisfaction. Perhaps one solution to this is to embed explicitly the expectation of coding in the first year first semester units.

This study has shown how creative coding activities can be embedded into the curriculum and delivers opportunities for students to build confidence in coding. It has also shown that experiential and social constructivist theories are useful approaches,^{34,39} on which to base learning strategies so that students develop skills in coding. Given coding has universal value,⁵⁶ learning to code has broad educational benefits, including problem-solving, collaboration, self-management and critical thinking, communicating and thinking about ideas.^{55,57}

This study also provides a learning design for faculty to introduce graduate qualities and course level learning outcomes on digital literacy skills and inventiveness or creativity in the curriculum. There is a need to introduce digital literacy skills in the curriculum, and a danger that students can progress through their Biomedical science program without being exposed to any coding at all. With coding skills biomedical science students are more likely to be job ready and employable. A major reason why this activity was overall well received by students, is because the teaching faculty were organized and between them had a diversity of skills from physiology to coding and data visualization. This learning design intentionally brought together faculty with these skills from across diverse disciplines and fields, in this case medical science and architecture and design. This learning design with of interdisciplinary faculty teams with a diversity of already pre-existing skills was key to success in the implementation. As a result of this success, we have since gone on to build Medical Science interdisciplinary capstone courses, where diverse teams of faculty are assembled from Arts, Social Science, Science and Medical Science schools and faculties. If we can overcome barriers between faculties and build student confidence and engagement, then we are more likely to have better prepared biomedical science graduates with basic coding literacy to face the 21st-Century workforce.

ACKNOWLEDGMENT

Open access publishing facilitated by The University of Sydney, as part of the Wiley - The University of Sydney agreement via the Council of Australian University Librarians.

ORCID

Phillip Gough  <https://orcid.org/0000-0003-3687-6365>

Oliver Bown  <https://orcid.org/0000-0001-9520-0192>

Craig R. Campbell  <https://orcid.org/0000-0002-0843-7579>

Philipp Poronnik  <https://orcid.org/0000-0002-6246-528X>

Pauline M. Ross  <https://orcid.org/0000-0002-8714-5194>

REFERENCES

- National Research Council. BIO2010: transforming undergraduate education for future research biologists. Washington, DC: National Academies Press; 2003.
- AAAS (2011) Vision and change in undergraduate biology education: a call to action. Final Report of a National Conference, American Association for the Advancement of Science. Available from: <https://visionandchange.org/>
- Koenig J. A survey of the mathematics landscape within bioscience undergraduate and postgraduate UK higher education. The Higher Education Academy: UK Centre for Bioscience; 2011.
- Feser J, Vasaly H, Herrera J. On the edge of mathematics and biology integration: improving quantitative skills in undergraduate biology. *CBE – Life Sci. Educ.* 2013;12:124–8. <https://doi.org/10.1187/cbe.13-03-0057>
- Matthews KE, Adams P, Goos M. Quantitative skills as a graduate learning outcome: exploring students' evaluative expertise. *Assess Eval High Educ.* 2017;42(4):564–79. <https://doi.org/10.1080/02602938.2016.1161725>
- Pevzner P, Shamir R. Computing has changed biology-biology education must catch up. *Science.* 2009;325(5940):541–2. <https://doi.org/10.1126/science.1173876>
- Qin H. Teaching computational thinking through bioinformatics to biology students ACM SIGCSE. *Bulletin.* 2009;41(1):188–91. <https://doi.org/10.1145/1539024.1508932>
- Tuomi P, Multisilta J, Saarikoski P, Suominen J. Coding skills as a success factor for a society. *Educ Info Technol.* 2018;23(1):419–34.
- Cederqvist AM. An exploratory study of technological knowledge when pupils are designing a programmed technological solution using BBC micro:bit. *Int J Technol Des Educ.* 2020; 0123456789:355–81. <https://doi.org/10.1007/s10798-020-09618-6>
- Dahlstrom MF. Using narratives and storytelling to communicate science with nonexpert audiences. *Proc Nat Acad Sci.* 2014;111-(Supplement_4):13614–20. <https://doi.org/10.1073/pnas.1320645111>
- Downs JS. Prescriptive scientific narratives for communicating usable science. *Proc Nat Acad Sci.* 2014;111(Supplement_4): 13627–33. <https://doi.org/10.1073/pnas.1317502111>
- Swaid SI. Bringing computational thinking to STEM education. *Procedia Manu.* 2015;3:3657–62. <https://doi.org/10.1016/j.promfg.2015.07.761>
- McClain CR. Practices and promises of Facebook for science outreach: becoming a “nerd of trust”. *PLoS Biol.* 2017;15(6):1–9. <https://doi.org/10.1371/journal.pbio.2002020>
- Killikelly A. Using the tools of informal science education to connect science and the public. *J Micro Biol Educ.* 2018;19(1): 1–5. <https://doi.org/10.1128/jmbe.v19i1.1427>
- Nayak S, Iwasa J H. (2019) Preparing scientists for a visual future: visualization is a powerful tool for research and communication but requires training and support, *EMBO Reports* (October). <https://doi.org/10.15252/embr.201949347>, 20.
- Bone EK, Ross PM. Rational curriculum processes: revising learning outcomes is essential yet insufficient for a twenty-first century science curriculum. *Stud High Educ.* 2019;46(2):394–405. <https://doi.org/10.1080/03075079.2019.1637845>
- Ashcraft MH. Math anxiety: personal, educational, and cognitive consequences. *Curr Dir Psych Sci.* 2002;11(5):181–5. <https://doi.org/10.1111/1467-8721.00196>
- Lyons IM, Beilock SL. When maths hurts: math anxiety predicts pain network activation in anticipation of doing math. *PLoS One.* 2012;31:e48076. <https://doi.org/10.1371/journal.pone.0048076>
- Quinnell R, Thompson R, LeBard RJ. It's not maths; it's science: exploring thinking dispositions, learning thresholds and mindfulness in science learning. *Int J Math Educ SciTech.* 2013;44(6): 808–16. <https://doi.org/10.1080/0020739X.2013.800598>
- Matthews KE, Hodgson Y, Varsavsky C. Factors influencing students' perceptions of their quantitative skills. *Int J Math Educ SciTech.* 2013;44(6):782–95. <https://doi.org/10.1080/0020739X.2013.814814>
- Flanagan KM, Einarson J. Gender, math confidence, and grit: relationships with quantitative skills and performance in an undergraduate biology course. *CBE – Life Sci. Educ.* 2021;16:1–11. <https://doi.org/10.1187/cbe.16-08-0253>
- Williams KR, Wasson SR, Barrett A, Greenall RF, Jones SR, Bailey EG. Teaching hardy-Weinberg equilibrium using population-level Punnett squares: facilitating calculation for students with math anxiety. *CBE – Life Sci. Educ.* 2021;20:1–16. <https://doi.org/10.1187/cbe.20-09-0219>
- Czerkawski BC, Lyman EW. Exploring issues about computational think. *High Educ Tech Trends.* 2015;59(2):57–65. <https://doi.org/10.1007/s11528-015-0840-3>
- Hsu T-C, Chang SC, Hung Y-T. How to learn and how to teach computational thinking: suggestions based on a review of the literature. *Comput Educ.* 2018;126(11):296–310. <https://doi.org/10.1016/j.compedu.2018.07.004>
- Bohm D, Peat FD. *Science order and creativity.* Milton Park: Routledge Press; 2010.
- Maeda J. *Creative code.* London: Thames and Hudson; 2004.
- McWilliam E, Poronnik P, Taylor PG. Re-designing science pedagogy: reversing the flight from science. *J SciEduc Tech.* 2008;17(3):226–35. <https://doi.org/10.1007/s10956-008-9092-8>
- Smith BK. Design and computational flexibility. *Digit Creat.* 2006;17(2):65–72. <https://doi.org/10.1080/14626260600878589>
- Romeike, R. (2007) Three drivers for creativity in computer science education. *Proc. of the IFIP-Conference on Informatics, Mathematics and ICT.*
- McCormack J D'IM. *Computers and creativity: the road ahead.* Computers and creativity. Berlin/Heidelberg: Springer; 2012. p. 421–4. https://doi.org/10.1007/978-3-642-31727-9_16
- Gough P, De Bérigny C, Bednarz T. Affective and effective visualisation: communicating science to non-expert users. 2014

- IEEE Pacific Visualization Symposium. Yokohama: IEEE; 2014. p. 335–9. <https://doi.org/10.1109/PacificVis.2014.39>
32. Knochel AD, Patton RM. If art education then critical digital making: computational thinking and creative code. *Stud Art Educ.* 2015;57(1):21–38. <https://doi.org/10.1080/00393541.2015.11666280>
 33. Ross PM, Taylor C, Jones SM, Johnson L. (2014) Vision and Innovation in Biology Education (VIBENet): final report. Available from: <https://ltr.edu.au/vufind/Record/365385>.
 34. Morris H. Experiential learning – a systematic review and revision of Kolb's model. *Interact Learn Environ.* 2020;28(8):1064–77. <https://doi.org/10.1080/10494820.2019.1570279>
 35. Ibrahim MF, Huddin AB, Hashim FH, Abdullah M, Rahni AAA, Mustaza SM, et al. Strengthening programming skills among engineering students through experiential learning based robotics project. *Int J Eval Res Educ.* 2020;9(4):939–46. <https://doi.org/10.11591/ijere.v9i4.20653>
 36. Krusche S, Reichart B, Tolstoi P, Bruegge B. Experiences from an experiential learning course on games development. *Proceedings of the 47th ACM Technical Symposium on Computing Science Education.* 2016;582–7. <https://doi.org/10.1145/2839509.2844599>
 37. Wunderlich L, Higgins A, Lichtenstein Y. Machine learning for business students: an experiential learning approach. *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education.* 2021;1:512–8. <https://doi.org/10.1145/3430665.3456326>
 38. Coker JS, Heiser E, Taylor L, Book C. Impacts of experiential learning depth and breadth on student outcomes. *J Exp Educ.* 2017;40(1):5–23. <https://doi.org/10.1177/1053825916678265>
 39. Vygotsky L. *Mind in society.* London: Harvard University Press; 1978.
 40. Paxton J. Live programming as a lecture technique. *J Comput Sci Coll.* 2002;18(2):51–6.
 41. Raj AGS, Patel JM, Halverson R, Halverson ER. Role of live-coding in learning introductory programming. *Proceedings of the 18th Koli Calling International Conference on Computing Education Research.* 2018;13:1–8. <https://doi.org/10.1145/3279720.3279725>
 42. Rubin MJ. The effectiveness of live-coding to teach introductory programming. *Proceeding of the 44th ACM Technical Symposium on Computer Science Education.* 2013;651–6. <https://doi.org/10.1145/2445196.2445388>
 43. Maeda, J. (2009) When a Good Idea Works. Available from: <https://www.technologyreview.com/s/414828/when-a-good-idea-works/>
 44. Processing Foundation. (2021) Processing.org. Available from: <https://processing.org/>
 45. Shiffman D, Fry S, Marsh Z. *The nature of code.* California, USA: D. Shiffman; 2012.
 46. Brandt CB, Cennamo K, Douglas S, Vernon M, McGrath M, Reimer Y. A theoretical framework for the studio as a learning environment. *Int J Technol Des Educ.* 2013;23(2):329–48. <https://doi.org/10.1007/s10798-011-9181-5>
 47. Gómez Puente SM, Van Eijck M, Jochems W. A sampled literature review of design-based learning approaches: a search for key characteristics. *Int J Technol Des Educ.* 2013;23(3):717–32. <https://doi.org/10.1007/s10798-012-9212-x>
 48. Tomitsch M, Wrigley C, Borthwick M, Ahmadpour N, Frawley J, Kocaballi B, et al. *Design. Think. Make. Break. Repeat. A handbook of methods.* Second ed. Amsterdam, The Netherlands: BIS Publishers; 2020.
 49. Arduino. (2018) Arduino: <https://www.arduino.cc/>
 50. Raj ADS, Ketsuriyonk K, Patel JM, Halverson R. Does native language play a role in learning a programming language? *Proceedings of the 49th ACM technical symposium on computer science education;* 2018. p. 417–22. <https://doi.org/10.1145/3159450.3159531>
 51. Merceron A, Yacef K. Educational data mining - a case study. *Proceedings of the 12th international conference on artificial intelligence in education.* AIED, Amsterdam, The Netherlands: IOS Press; 2005. p. 467–74.
 52. Mor Y, Mellar H, Warburton S, Winters N, editors. *Practical design patterns for teaching and learning with technology.* Rotterdam: Sense Publishers; 2014. <https://doi.org/10.1007/978-94-6209-530-4>
 53. National Research Council (NRC). *A new biology for the 21st century: ensuring the United States leads the coming biology revolution.* Washington, DC: National Academies Press; 2009.
 54. National Academies Press. *How people learn II.* Washington, D.C.: National Academies of Sciences Engineering and Medicine; 2018. <https://doi.org/10.17226/24783>
 55. Bers MU. *Coding as a playground: programming and computational thinking in the early childhood classroom.* New York: Routledge; 2018.
 56. Resnick M, Siegel D. Invited commentary a different approach to coding. *Int J People-Oriented Program.* 2015;4(1):1–4. <https://doi.org/10.4018/IJPOP.2015010101>
 57. Popat S, Starkey L. Learning to code or coding to learn? *System Rev., Comput Educ.* 2019;128(1):365–76. <https://doi.org/10.1016/j.compedu.2018.10.005>

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

How to cite this article: Gough P, Bown O, Campbell CR, Poronnik P, Ross PM. Student responses to creative coding in biomedical science education. *Biochem Mol Biol Educ.* 2023;51(1): 44–56. <https://doi.org/10.1002/bmb.21692>