ABSTRACT

Computational thinking (CT) is a thought process composed of computer science ideas and skills that can be applied to solve problems and better understand the world around us. With the increase in technology and computing, STEM disciplines are becoming interwoven with computing. In order to better prepare students for STEM careers, computational literacy needs to be developed in K–12 education. We advocate the introduction of computational literacy through the incorporation of CT in core science courses, such as biology. Additionally, at least some of this integration should be unplugged, or without computers, so that all schools can participate in developing computational literacy. These lessons integrate unplugged CT and science content to help students develop CT competencies and learn natural selection content simultaneously through a series of lessons in which unplugged CT is leveraged for natural selection learning within varying contexts. In these lessons, students engage in the creation of handwritten algorithmic explanations of natural selection. Students build CT skills while making sense of the process, resulting in converged learning about CT and science. This article presents a description of CT, the specifics of the classroom implementation and lessons, student work and outcomes, and conclusions drawn from this work.

Key Words: natural selection; computational thinking; secondary students.

Introduction

Computational thinking (CT) is a thought process composed of computer science ideas and skills that can be applied to solve problems and better understand the world around us (Wing, 2006). Although CT is the logic underlying computer science, it can be applied to other science and engineering disciplines and has uses in everyday life. People who can think computationally can deconstruct, abstract, and generalize information and think sequentially and algorithmically to solve problems and explain phenomena. With the increase in technology and computing, STEM (science, technology, engineering, and mathematics) disciplines are becoming interwoven with computing. In order to better prepare students for STEM careers, computational literacy needs to be developed in K–12 education. However, incorporating computing into K–12 education can be problematic: most teachers are not prepared to teach computer science, introducing an additional course increases students’ course loads in an already crowded curriculum, and computers are not readily available at all schools.

We advocate the introduction of computational literacy through the incorporation of CT in core science courses, such as biology. Additionally, at least some of this integration should be unplugged, or without computers, so that all schools can participate in developing computational literacy. Schools with access to technology can then expand on unplugged approaches to incorporate computing and programming. The vast majority of CT lessons focus on CT as programming to solve problems (Peel, 2019). Most science and CT integrated lessons utilize computers for modeling and simulating science phenomena or use computing to collect and manipulate data, both of which rely heavily on computers (Kahlehojulu, 2018). While these are valuable curricular examples, there is a missed opportunity to use CT for scientific explanation.

We describe a series of lessons designed and implemented in honors and general 10th-grade biology courses. These lessons integrate unplugged CT and challenging science content to help students develop CT competencies and learn natural selection content simultaneously. In these lessons, students engage in the creation of handwritten algorithmic explanations of natural selection. In other words, students write the sequence of steps that happen during the process of natural selection. Students build CT skills while making sense of the process, resulting in converged learning about CT and science. This approach shifts the focus of CT from engineering and problem solving to science and explanations based on evidence.

The lessons are informed by the evolution education research literature. For instance, student learning and application of natural selection content is context dependent. Students respond differently to plant, insect, and other animal items in terms of natural selection understanding (Nehm & Schonfeld, 2010; Heredia et al., 2016). Students who learn natural selection within one context struggle to explain natural selection in a new context (Peel et al., 2018).
Students also display well-documented misconceptions, such as needs-based, anthropomorphic, and individual-change misconceptions (reviewed in Gregory, 2009). Additionally, students have demonstrated context-specific misconceptions, such as a misunderstanding of bacteria in the development of antibiotic resistance (Peel et al., 2019b).

Here, we describe a series of lessons in which unplugged CT is leveraged for natural selection learning within varying contexts. The following sections present a description of CT, the specifics of classroom implementation and lessons, student work and outcomes, and conclusions drawn from this work.

**Computational Thinking Framework**

These lessons were designed using the UnPlugged Design of Alg oriThmic Explanations (CT UPDATE) framework for CT and science integration (Peel, 2019). This framework engages students in CT through the creation of scientific explanations. There are two CT facets used in this approach: CT Principles and CT Practices. The CT Principles are branching, iteration, methods, and variables, which are concrete parts of an algorithm (defined in Table 1). As students use the CT Principles, they are engaging in CT Practices, which are sequencing steps, abstracting information, generalizing, recognizing patterns, decomposing processes, and evaluating algorithms.

When branching is used, it involves **sequencing events and steps** based on a specific condition: if the condition is true, then something happens, or else another thing happens. To use iteration, students must **evaluate** a sequence of steps, **recognize patterns**, and identify how an algorithm can be simplified, or **abstracted**, with a loop. Incorporating methods requires students to **decompose** the science process, **generalize** the steps for use across contexts, and **abstract** important information. Using variables requires students to **evaluate** their algorithm and identify where variables can be used to make their algorithm more efficient, or to **generalize** their algorithm for use in multiple contexts.

| Table 1. Description of CT Principles. |
|----------------------------------------|
| **CT Principles** | **CT Practices** |
| **Branching** – checking a condition and choosing a path based on that condition; “if, then, else” statements | **Sequencing Steps** |
| **Iteration** – repeating a sequence of steps until a condition is met; loop | **Abstracting Information** |
| **Method** – an encapsulated sequences of steps, or mini-algorithms, with its own function | **Generalizing** |
| **Variable** – a value that can change within an algorithm, or with the use of that algorithm in different situations | **Recognizing Patterns** |
|                                         | **Decomposing Processes** |
|                                         | **Evaluating Algorithms** |

**Classroom Implementation with Natural Selection**

A series of five lessons were designed and implemented in 10 high school biology classes. This section describes the lessons and how to implement them. The lessons address the NGSS performance expectation HS-LS4-4: “Construct an explanation based on evidence for how natural selection leads to adaptation of populations” (NGSS Lead States, 2013). The performance expectation uses the science practice constructing explanations, the crosscutting concept cause and effect, and the disciplinary core idea adaptation (LS4.C). The overall time needed to implement these lessons is approximately six hours. The lesson sequence is as follows: introduction to CT, mountain sheep natural selection, bacterial natural selection, field mustard natural selection, and context-general natural selection. In each step, students use CT to create handwritten algorithmic explanations based on evidence. The lesson timeline is provided in Table 2, and descriptions of each lesson follow.

**Materials**

To implement these lessons as described, materials for each investigation are needed (for bacterial lab details, see Williams et al., 2018; for other sheep, mustard, and general lesson details, see the Supplemental Material available with the online version of this article). Students will need paper and writing utensils for algorithm creation, and white boards and markers for group algorithms. If the instructor utilizes Lightbot for the first lesson (optional), students will need computers, tablets, or phones to play the game, which can be downloaded from the app store, or accessed through a web browser at http://lightbot.com/hour-of-code.html.

| Table 2. Unit outline. |
|------------------------|
| **Lesson** | **Description** | **Duration (minutes)** |
| 1 | Introduce computational thinking. Create algorithms of a familiar process. | 90 |
| 2 | Explore natural selection in mountain sheep. Create mountain sheep algorithms. (Begin bacterial lab before this lesson.) | 45 |
| 3 | Explore bacterial evolution through lab. Create bacterial algorithms. | 135 |
| 4 | Explore natural selection in field mustard. Create field mustard algorithms. | 45 |
| 5 | Define natural selection terms. Revisit CT and variables as a way to generalize processes. Compare bacteria, sheep, and mustard algorithms. Create a general algorithm. | 45 |
Introduction to CT

The first lesson serves as an introduction to CT and CT Principles of branching, iteration, methods, and variables. In prior implementations, the game Lightbot has been used to engage students in CT (for further details, see Peel & Friedrichsen, 2018; for instructional materials, see Supplemental Material). Lightbot is a drag-and-drop visual programming game in which students create steps for a robot to follow. The game has three levels: Basics, Procedures, and Loops. The game allows students to use iteration through loops, and methods through procedures. Students play different levels of the game, reflect on each level, and then the CT Principles are defined (Table 1). Using Lightbot is optional and not necessary. However, we found Lightbot to be a quick and fun way for students to use CT Principles before we introduce terminology. To eliminate the need for a classroom set of computers, the teacher, or volunteer students, can play the game through a single class computer projected for the class to see. Alternatively, to make the lesson completely unplugged, various levels of the game can be printed and distributed to the class, in which case students would write the steps on paper or whiteboards and act out the steps. To view a video of the implementation of this intro lesson, visit https://youtu.be/S9wvZkCSAU8.

After an introduction to CT Principles, the instructor defines each CT Principle using examples from the game and students’ everyday experiences. Next, students practice using CT Principles in groups by creating algorithms explaining their process for getting ready in the morning, a familiar process. Students typically struggle with this at first, so it is helpful to show and discuss examples of different algorithms for getting ready in the morning (Figure 1). These examples depict two different approaches to algorithm creation with flowchart and numbered styles. These also show different usage of CT Principles.

Mountain Sheep Natural Selection

The goal of this lesson is to support students in creating algorithmic explanations of natural selection based on scientific evidence. As such, students are presented with a series of evidence statements about a mountain sheep population (Chinn & Duncan, 2014). In this population, horn sizes are gradually getting smaller over time as hunters kill the sheep with the largest horns. Then students are given four models with algorithmic explanations (see Supplemental Material). Based on the evidence presented, students are tasked with choosing the algorithm and model that best explain the situation. This allows students to use scientific evidence and see various algorithms of the natural selection process to help familiarize them with algorithmic explanations and CT Principles. Next students use these experiences to create their own algorithmic explanations of the changes in the mountain sheep population. In order to foster creativity in algorithm creation, students do not copy the algorithm they chose in the prior step, but are encouraged to create algorithms that reflect their own understanding of the process. Figure 2 is an example of one student’s mountain sheep algorithmic explanation. This student had correct conceptions of natural selection and used branching, iteration, and multiple variables.

Bacterial Natural Selection

Next, students investigate the development of variation in a bacterial population over time. During the lab, students plate an aliquot of an initial bacterial culture on an agar + Luria broth plate with a blank paper disk (control) and a disk treated with antibiotic. The original culture is allowed to grow in a tube for two days. Note that the lab should be started prior to the mountain sheep investigation to give bacteria time to incubate. On day 3, a second aliquot from the culture tube is plated on a second agar + Luria broth plate with a blank disk and an antibiotic disk, which is also incubated for two days. On day 5, students observe both plates and determine that the first aliquot population was susceptible to the antibiotic because of the zone of inhibition around the antibiotic disk and that the second aliquot is resistant because the bacteria grow close to the antibiotic disk. Due to accumulated mutations, the population of bacteria that grows in the culture tube for an additional two days has more resistant individuals. For a more detailed description of the lab procedure and lab materials, see Williams et al. (2018).

Figure 1. Example algorithmic explanations of getting ready in the morning.
After the lab investigation, students are presented with two possible models that could explain their lab results. Students use the evidence from their lab investigation to determine which model best explains the development of antibiotic-resistant bacteria. The models are used to help scaffold student understanding of random mutations in a population. Students then create algorithmic explanations of antibacterial resistance based on evidence from the lab experience. Figure 3 depicts a student example of a bacterial algorithmic explanation. This student used branching in step 3, iteration and a method in step 1, and a variable for the type of bacteria to correctly explain natural selection in this context.

Field Mustard Natural Selection

In this lesson, students continue to use evidence to explain natural selection in a population of field mustard. As students move through the unit, the teacher provides less scaffolding. In this lesson, students are given pieces of evidence to evaluate without a model or algorithm to assess and choose; they are told that a population of field mustard shifts from late flowering to early flowering after a five-year drought (adapted from https://www.nescent.org/eog/eognews.php?id=27.html). After evaluating the evidence, students again create algorithmic explanations of antibacterial resistance based on evidence from the lab experience. Figure 3 depicts a student example of a bacterial algorithmic explanation. This student used branching in step 3, iteration and a method in step 1, and a variable for the type of bacteria to correctly explain natural selection in this context.

Context-General Natural Selection

Since natural selection learning is known to be contextualized, and students struggle to apply natural selection to new contexts, the goal of this lesson is to synthesize the process of natural selection and scaffold the application of natural selection knowledge to any context. At this point, the teacher discusses and defines key natural selection ideas, including population vs. individual, mutation, selection pressure, favorable trait, differential survival, population shift, and the process of natural selection. Led by the teacher, students work through how to create a generalized algorithm by first using a more familiar concept, a food chain. Since creating a general algorithm will be new to students, we practiced with an easy example, food chains, to get students ready for the more complicated case of natural selection. Two specific food chains are presented, and one generalized food chain is created through the use of this student may have the misconception that plants change in order to survive a selective pressure. This is a key opportunity for peers and the teacher to provide feedback to the student to help with understanding the process.

In Figure 5, a student explains natural selection correctly, using variables, iteration, and branching to explain the process. When providing feedback on this explanation, the teacher could ask the student to identify a process in the algorithm other than natural selection. The student has already used the word “process” in step 4 when referring to reproduction, which is an example of a method. This student implicitly called a method by using “reproduction,” but feedback has the potential to help her recognize the method and incorporate it explicitly in her algorithmic explanation. This student also displays a potential misunderstanding of plant reproduction in step 3 by saying plants pass on mutations asexually, when, in reality, mutations are passed on both asexually and sexually. This may prompt further learning activities on plant reproduction if many students discuss only sexual or asexual reproduction in plants.
of variables. A generalized algorithm is useful because it can be used to explain many instances of that process. Thus, an algorithm with producer, primary consumer, secondary consumer, and so on can be much more applicable and useful than an algorithm with grass, caterpillar, bird, and so on. In this case, producer, primary consumer, and secondary consumer are each a variable, and the value of that variable is set according to the context. So, in the specific case, producer = grass and primary consumer = caterpillar. To view a video of the implementation of this lesson and associated instructional materials, visit https://youtu.be/14omwM-3Y1Y (and see Supplemental Material).

To apply this to natural selection, students compare and contrast their bacteria, sheep, and mustard algorithms with the following guiding questions: What is different between these algorithms? What is the same? Where can we use variables to generalize? In this discussion, think-pair-share can be used, followed by a class discussion of these questions. Similarities between the algorithms are the steps in the process, such as the introduction of some sort of pressure. Differences between the algorithms include specifics, such as organism and selective pressure. Another key difference between the three is the time scale. Bacterial populations develop variation in a matter of days because they rapidly reproduce, as seen in the lab. Plants reproduce slower than bacteria, but generally faster than animals, meaning that plant populations develop variation and population shifts faster than animals. Students should discuss how the specifics of each algorithm can be generalized with a variable. For example, selection pressure is a variable used to generalize drought, antibiotics, and hunting.

Based on this discussion, students create a context-general algorithmic explanation of natural selection. Afterward, students are placed in groups where they each share their algorithm. This fosters the development of natural selection and CT understandings because students can see how others are thinking about natural selection and how they depict it with algorithmic explanations. Then students work as a group to create a consensus generalized algorithm on white boards. The groups engage in a gallery walk in which students give feedback to other groups and receive feedback on their algorithm. This is an opportunity for the teacher to identify areas of misunderstanding and misconceptions. The teacher should give targeted feedback on both natural selection and CT Principles. For example, some students may not have included iteration in their algorithms, or may have said that individual organisms change after the selection pressure. This feedback is key for sense making about natural selection and CT. Students return to their groups and revise their algorithms on the basis of the feedback. This final revision and algorithm creation can be done individually or in groups, at the discretion of the teacher. Figure 6 is an image of a student's final generalized algorithmic explanation of natural selection. This student was able to explain natural selection accurately with the use of variables, iteration, a method, and branching.

○ Overview of Student Learning & Perspectives

Based on our research findings, students learned natural selection (Peel et al., 2019a) and developed CT competencies (Peel, 2019) through this approach. Students’ use of natural selection factors (mutation, initial variation, selection pressure, favorable trait, differential survival, reproduction, and population shift) significantly increased from pre- to post-unit algorithms. Their correct sequencing of the natural selection process also significantly increased after the unit, with most students correctly sequencing natural selection steps. The number of misconceptions (needs-based, anthropomorphic, and individual-change) significantly decreased over the course of the unit. Similarly, students’ use of CT Principles (branching, iteration, methods, and variables) significantly increased from pre- to post-unit algorithmic explanations. Additionally, the data suggest that the generalized lesson was key for both natural selection and CT sense making. Students indicated that creating algorithmic explanations across the unit helped them better understand natural selection, and the repetition gave them the practice they needed to better apply CT Principles.

○ Conclusion

We have described a successful approach to integrating CT and science content that does not require computers or programming knowledge. In this series of lessons, students use CT to create unplugged algorithmic explanations of natural selection based on evidence in several contexts: mountain sheep, bacteria, field mustard, and general. The creation of unplugged algorithmic explanations allows students to make sense of natural selection and CT simultaneously. Analysis of student work indicates that this combination of CT and science content helps students learn natural selection and develop CT competencies. In terms of natural selection content learning, this approach yielded higher learning gains than a prior approach in which modeling was used to facilitate student understanding of natural selection in an antibacterial-resistance context (Peel et al., 2019b). The addition of different organismal contexts and the use of CT to scaffold across these contexts supported students’ natural selection learning. Given this integration’s success, there is potential for further CT and science integrations in which CT is leveraged for understanding other biological processes, such as mitosis and meiosis, ecological interactions, body system functions, photosynthesis, and many more.

| Iteration | Method | Branching | Variables |
|-----------|--------|-----------|-----------|
| 1. Population exists in its environment | Population | Differential survival |
| 2. Random mutations cause variation in the population | Selective Pressure | If individual has adaptation, it survives more often |
| 3. Reproduces repeating step 2, until selective pressure is applied | Adaptation | Else individual survives less often |

Figure 6. Transcribed student example of a final generalized natural selection algorithmic explanation.

References

Chinn, C. & Duncan, R.G. (2014). Promoting Reasoning and Conceptual Change in Science (PRACCIS) Curriculum: Evolution. Retrieved from https://sites.google.com/a/gse.rutgers.edu/praccis-promoting-reasoning-and-conceptual-change-in-science/home.

Gregory, T.R. (2009). Understanding natural selection: essential concepts and common misconceptions. Evolution: Education and Outreach, 2, 156–175.
Heredia, S.C., Furtak, E.M. & Morrison, D. (2016). Exploring the influence of plant and animal item contexts on student response patterns to natural selection multiple choice items. *Evolution: Education and Outreach, 9*, article 10.

Kalelioğlu, F. (2018). Characteristics of studies conducted on computational thinking: a content analysis. In *Computational Thinking in the STEM Disciplines* (pp. 11–29). Cham, Switzerland: Springer.

Nehm, R.H. & Schönfeld, I.S. (2010). The future of natural selection knowledge measurement: a reply to Anderson et al. *Journal of Research in Science Teaching, 47*, 358–362.

NGSS Lead States (2013). *Next Generation Science Standards: For States, by States*. Washington, DC: National Academies Press.

Peel, A. (2019). Developing computational thinking competencies and natural selection understanding through unplugged algorithmic explanations. PhD dissertation, University of Missouri, Columbia.

Peel, A. & Friedrichsen, P. (2018). Algorithms, abstractions, and iterations: teaching computational thinking using protein synthesis translation. *American Biology Teacher, 80*, 21–28.

Peel, A., Sadler, T.D. & Friedrichsen, P. (2019a). Learning natural selection through computational thinking: unplugged design of algorithmic explanations. *Journal of Research in Science Teaching, 56*, 983–1007.

Peel, A., Zangori, L., Friedrichsen, P., Hayes, E. & Sadler, T. (2019b). Students’ model-based explanations about natural selection and antibiotic resistance through socio-scientific issues-based learning. *International Journal of Science Education, 41*, 510–532.

Royal Society (2012). Shut down or restart? The way forward for computing in UK schools. https://royalsociety.org/~media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf.

Williams, M.A., Friedrichsen, P.J., Sadler, T.D. & Brown, P.J. (2018). Modeling the emergence of antibiotic resistance in bacterial populations. *American Biology Teacher, 80*, 214–220.

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

AMANDA PEEL is a Postdoctoral Scholar in the Learning Sciences Department at Northwestern University. She can be contacted at amanda.peel@northwestern.edu. TROY D. SADLER is the Thomas James Distinguished Professor of Experiential Learning in the School of Education at the University of North Carolina at Chapel Hill. He can be contacted at tsadler@unc.edu. PATRICIA FRIEDRICHSEN is an Associate Professor in the Learning, Teaching, and Curriculum Department at the University of Missouri. She can be contacted at friedrichsenp@missouri.edu.

Nominations for 2021 NABT Awards are due March 15th
https://nabt.org/Awards-NABT-Award-Nomination-Form

Your dedication to your students... has NOT gone unnoticed.