Journal of Mathematical Behavior 65 (2022) 100932

Contents lists available at ScienceDirect

Journal of Mathematical Behavior

journal homepage: www.elsevier.com/locate/jmathb

A task to connect counting processes to lists of outcomes in combinatorics

Adaline De Chenne [a,*], Elise Lockwood [b]

[a] Oregon State University, Department of Mathematics, 320 Kidder Hall, Corvallis, OR, 97331, United States
[b] Oregon State University, Department of Mathematics, 064 Kidder Hall, Corvallis, OR, 97331, United States

ARTICLE INFO

Keywords:
Combinatorics
Student thinking
Discrete mathematics
Computational setting

ABSTRACT

Research has shown that solving counting problems correctly can be difficult for students at all levels, and mathematics educators have sought to identify strategies and interventions to help students reason conceptually about combinatorial tasks. A set-oriented perspective (Lockwood, 2014) is a way of thinking about counting problems that emphasizes the importance of reasoning about the set of outcomes being counted. From a set-oriented perspective, one possible type of intervention is to have students focus on the sets of outcomes rather than formulas and expressions, and specifically to reason about the structure of the set of outcomes. Yet, reasoning about sets of outcomes is not sufficient for students to make connections between outcomes and counting processes. In this paper, we investigate tasks where students wrote computer code to enumerate the set of outcomes in a specific order by implementing listing processes, and they were then asked to determine a specific numbered outcome in their list by using the structure of their enumeration scheme. We clarify particular aspects of a set-oriented perspective that were productive for students, and we demonstrate that tasks that asked students to name a specific outcome in their list elicited meaningful connections between counting processes and sets of outcomes. Further, such tasks reinforce desirable mathematical practices such as leveraging structure and connecting representations.

1. Introduction

Suppose you were given the following task: "If you had a lexicographic list of all arrangements of the letters in the word ROCKET, what would be the 80th outcome in your list?" This type of combinatorial task, which we refer to as an "*m*th element task," can be solved in multiple ways that leverage reasoning that we value in combinatorics education. Students' engagement with this type of *m*th element combinatorial task is the focus of this paper.

Counting problems are mathematical tasks in combinatorics that require critical thinking and ingenuity (Kapur, 1970; Tucker, 2002). These problems have meaningful applications in a variety of mathematical and scientific fields, including probability, statistics, chemistry, and computer science. Further, combinatorics, as a central topic in discrete mathematics, is an essential tool for solving relevant problems in our current times (e.g., Lockwood et al., 2021). Prior literature has revealed that counting can be a difficult domain for students to master (e.g., Annin & Lai, 2010; Batanero et al., 1997; Hadar & Hadass, 1981), often because problems do not have prescribed, foolproof methods to solve them, and because they can involve implementing processes that may seem logically

* Corresponding author.
E-mail addresses: dechenna@oregonstate.edu (A. De Chenne), Elise.Lockwood@oregonstate.edu (E. Lockwood).

https://doi.org/10.1016/j.jmathb.2021.100932
Received 1 February 2021; Received in revised form 3 December 2021; Accepted 16 December 2021
Available online 29 December 2021

sound but contain subtle errors (e.g., Annin & Lai, 2010; Lockwood, 2014).

Some key work in combinatorics education literature has focused on students reasoning carefully about the set of outcomes they are counting (e.g., Lockwood, 2013, 2014; Lockwood & Gibson, 2016; Wasserman & Galarza, 2018). In particular, Lockwood (2014) suggested that directly attending to outcomes may allow students to understand problem types and formulas better, and prior research has sought to find creative interventions to help students connect their counting processes to the outcomes they generate. Researchers have also explored the role of computational environments and activities in supporting students' engagement with sets of outcomes (e.g., Medova & Ceretkova, 2021; Lockwood, In Press; Lockwood & De Chenne, 2020, 2021). In such work, researchers leverage the ability of the computer to produce large sets of outcomes efficiently. Ultimately, this body of work has demonstrated effective ways to orient students toward reasoning about sets of outcomes, but there is more to explore about how to encourage students to reason about outcomes.

Given prior work that highlights the value of having students focus on outcomes, we are motivated to examine ways to connect counting processes to sets of outcomes that deepen students' understandings of the problem. In this paper, we focus on how students solved problems in an intervention designed to focus their attention on structure within the set of outcomes, where they reasoned about how outcomes of a given process may be listed in an organized way. We refer to such tasks as "*m*th element" tasks (which we elaborate in the methods section), where students are asked to identify a particular element in a list of outcomes they have described or created. In the data we present, students reason about lists they created in a computer program, but we conceive of *m*th element tasks as being applicable beyond a computational environment. We use the idea of 'listing processes' (defined in Section 3.2.1), which provide one way to illustrate relationships between counting processes and sets of outcomes; specifically, in our data, students reason about lists of outcomes they created by writing computer code in Python, where the term list is used in reference to a set of outcomes in a specific order. Although the use of the computational environment is an intrinsic part of our data, the student work we present is work done by hand; we argue that the results should not be interpreted strictly as being influenced by the computational environment. We believe that this highlights how the use of a computational setting can influence by-hand work without the focus being solely on the computational setting itself.

With this paper, we seek to make three contributions to the combinatorics education literature. First, we address how a more robust understanding of counting principles can be reinforced through strategic combinatorial tasks whose goal extends beyond only determining a cardinality. Our contribution to this body of literature comes from asking questions that relate to the organization of a list of outcomes, whereby that organization is used to compute a specific outcome. Second, we extend current literature that investigates how a computational setting can be used to augment combinatorial reasoning conventionally done by hand (e.g., Medova & Ceretkova, 2021; Lockwood et al., 2021; Lockwood & De Chenne, 2020, 2021). Our contribution to this literature comes from the objects students reason about, which are lists generated by a computer program the students wrote. These lists are not tractable to create by hand, and although students can describe the lists without a computer program and reason about the organization of the list, we believe it is reasonable to assume that having written the computer program and having access to the complete list (often consisting of thousands of outcomes) can affect how the students reason about it. Third, we aim to elaborate and extend Lockwood's (2013) model of students' combinatorial thinking through our discussion of listing processes and lists of outcomes. In this way, because our data focus on student responses to a particular task, we also contribute to the body of work that has proposed innovative instructional interventions that help students reason conceptually about counting problems.

In the following sections, we first briefly situate the *m*th element task within the literature and perspectives on combinatorial thinking. We then describe the tasks and provide a mathematical discussion. We offer data examples that motivate the desire to ask questions that require reasoning about structure in a set of outcomes. Then, we demonstrate two distinct (but related) solution methods students used to solve the tasks. In presenting the examples of student work, we attempt to highlight the important combinatorial ideas and broader mathematical practices the students used and engaged with to solve the tasks. Throughout this paper, we attempt to answer the following research questions:

1) What are some aspects of a set-oriented perspective that students demonstrated while solving *m*th element tasks?
2) In what ways do those aspects of a set-oriented perspective reinforce the relationship between counting processes and lists of outcomes?

2. Background literature

2.1. Student difficulties in combinatorics

There is evidence that students at all levels face difficulties when solving elementary combinatorics problems correctly (e.g., Batanero et al., 1997; Kavousian, 2008; Lockwood & Gibson, 2016), and counting problems can be particularly difficult to teach (e.g., Annin & Lai, 2010) and learn (e.g., Hadar & Hadass, 1981). Some documented errors include distinguishing between orderings when ordering does not matter, or conversely, failing to distinguish between orderings when order does matter (e.g., Batanero et al., 1997). Students can also find it difficult to verify their solutions (e.g., Eizenberg & Zaslavsky, 2004) and distinguish between problem types (e.g., Lockwood, 2014), which can lead them to apply inappropriate formulas that yield incorrect answers.

A number of researchers have sought ways to improve students' combinatorial thinking and activity. This has included designing tasks that will evoke particular combinatorial concepts or approaches (e.g., Eizenberg & Zaslavsky, 2004; Lockwood et al., 2015; Tillema, 2013; Tillema & Gatza, 2016), investigating the role of representations and listing (e.g., Braithwaite & Goldstone, 2013; Wasserman & Galarza, 2019), developing instructional interventions to teach combinatorial content or practices (e.g.,

CadwalladerOlsker et al., 2012; Lockwood et al., 2015; Reed & Lockwood, 2021), and describing ways of thinking (e.g., Halani, 2012; Lockwood, 2014) that might help students solve counting problems more successfully. This has also included outlining potential learning trajectories in combinatorics across which students might progress (e.g., Maher et al., 2011; Tillema, 2020). These interventions and explorations into students' combinatorial thinking have been valuable contributions to the field, and broadly we add to such findings and resources that fundamentally help to improve the teaching and learning of combinatorics. In the following paragraphs we elaborate our contribution to this literature base more explicitly.

2.2. Research that focuses on sets of outcomes

As noted in the introduction, one line of research has particularly focused on having students think about the sets of outcomes they are trying to count. Specifically, Lockwood (2013) suggests that a reason for difficulties with counting may be that students do not connect the formulas/expressions used for each problem type to the counting process or the set of outcomes being counted. That is, they do not understand conceptually why a counting process leads to a formula/expression, and/or why a counting process counts a set of outcomes or produces a particular list of outcomes. Lockwood (2014) suggested that teachers and researchers try to help students develop and adopt a set-oriented perspective, which is a way of thinking in which students view counting as fundamentally involving enumerating a set of outcomes. Additional related work has investigated how students connect counting processes and solutions among questions where the sets of outcomes have different natural encodings (e.g., Wasserman & Galarza, 2019), and showing that by-hand listing behaviors (of both complete and partial sets of outcomes) could reinforce the logic behind counting processes and assist students in reasoning about appropriate formulas (Lockwood & Gibson, 2016).

Such work is in line with and builds on other researchers' reports of counting processes and outcomes, describing ways in which students might connect outcomes with a process for generating them. For example, English (1991, 1993) described an *odometer strategy* among young children in the context of creating outfits, which involves holding an item constant and systematically cycling through other items. Halani (2012) later elaborated this and described an odometer way of thinking, including standard odometer and wacky odometer, where systematic cycling is extended to more than items (such as cycling through positions). In addition, Tillema (2013, 2018) has investigated various ways in which students might symbolize or represent outcomes they are trying to count, and he has emphasized the importance of such representations in developing students' combinatorial reasoning. Further, in a psychological study, Braithwaite and Goldstone (2013) explored the role of representations and formalisms among psychology students in a combinatorial context. Ultimately, Braithwaite and Goldstone summarize their findings by saying the following:

In our view, this summary highlights how our work fits into the existing literature – we are exploring a specific way to give "active encouragement" to help students make meaningful connections between combinatorial representations (particularly counting processes and outcomes, and sometimes formal expressions and formulas).

Our overall goal and contribution in this paper is to offer a novel way to reinforce the relationship between counting processes and sets of outcomes through *m*th element tasks. Our contribution is not just the tasks themselves, but rather we emphasize two features of these tasks that make them particularly well-suited to eliciting productive ways of thinking for students. First, they require students to reason about a certain level of specificity with regard to processes and outcomes, and, second, they offer connections to computing and computational perspectives. We thus present an intervention that is aimed to help students successfully solve counting problems specifically by drawing students' attention to sets of outcomes that were, in our case, introduced in a computational setting.

2.3. Computing in combinatorics education

Along with research that discusses how listing outcomes in combinatorics can reinforce counting principles, there is a growing body of work that investigates how computational settings can be used to foster combinatorial reasoning. In particular, Lockwood and De Chenne (2020, 2021) have reported on students who write computer programs in Python that list entire sets of outcomes for combinatorics problems, and Medova and Ceretkova (2021) have reported on undergraduate students' coding abilities and ability to solve combinatorial problems. Medova and Ceretkova (2021) found that the ability to write code was an important factor for combinatorial reasoning. Such studies demonstrate possible connections between reasoning computationally and combinatorically that merit further consideration, particularly those that seek to identify particular aspects of computer programming that can impact combinatorial reasoning. This work that has focused on computing in combinatorics education is part of a growing body of work that explores computing and computer programming in STEM education (e.g., Caballero et al., 2019; Sengupta et al., 2013; Wagh et al., 2017; Weintrop et al., 2016) and in mathematics education in particular (e.g., Benton et al., 2018; Feurzeig et al., 2011; Kotsopoulos et al., 2017; Papert, 1980; Sinclair & Patterson, 2018).

3. Theoretical perspectives, listing processes, and motivating examples

The theoretical perspective we use is Lockwood (2013) model for student thinking in combinatorics, which we use to frame broadly how students reason about the tasks we present in this paper. We introduce our notion of "listing processes" to be instantiations of solutions that enumerate (and list) sets of outcomes. Listing processes are a refinement of Lockwood (2013) model that align with her set-oriented perspective (2014), where a solution to a counting problem is the cardinality of a set of outcomes. In Section 3.3, we offer two short data examples of students using a set-oriented perspective to emphasize that connecting counting processes and lists of outcomes is not trivial, and requires attention and intervention. Our inclusion of these data examples is to motivate the tasks we present in this paper, which we believe target connections that are valuable in combinatorial reasoning yet are not fully realized by the

students in the examples.

3.1. Lockwood's model of students' combinatorial thinking

In framing our work in this paper, we draw upon Lockwood (2013) model of students' combinatorial thinking. This model has been used and refined by Lockwood and colleagues over the years (e.g., Lockwood et al., 2015; Lockwood et al., 2021) and has been used by a number of researchers to situate students' combinatorial thinking (e.g., Medova & Ceretkova, 2021; Mota & Ferreira, 2021; Wasserman & Galarza, 2019). In this model (Fig. 1), Lockwood describes three different components of students' combinatorial thinking: formulas/expressions, counting processes, and sets of outcomes.

Formulas/expressions are combinations of numbers, operations, or variables that describe a general relationship among variables or yield some numerical value. Counting processes are the enumeration procedures, real or imagined, in which one engages as they solve a counting problem. Sets of outcomes are the set of elements being counted in a given counting problem – the cardinality of the set of outcomes is commonly the answer to a given counting problem. Lockwood (2013) emphasizes that there is a relationship between the counting process one employs and the sets of outcomes that process generates or organizes, in the sense that certain counting processes will generate and organize outcomes in a particular way. Lockwood has repeatedly called for ways in which to reinforce connections between counting processes and sets of outcomes (e.g., Lockwood, 2013, 2014; Lockwood & Gibson, 2016; Lockwood et al., 2015; Lockwood & Purdy, 2019; Lockwood et al., 2021), and she has argued that student engagement in this relationship can deepen their mathematical understanding of counting problems and can be beneficial for students' combinatorial thinking and activity (Lockwood & Gibson, 2016; Lockwood et al., 2015).

Lockwood and colleagues have explored potential ways to help reinforce this relationship between counting processes and sets of outcomes. Specifically, Lockwood and Gibson (2016) examined students' by-hand listing activity and demonstrated a positive correlation between listing activity and correctly solving counting problems among novice counters; in this case, by-hand listing served as a mechanism by which to connect counting processes and sets of outcomes. In another example, Lockwood and De Chenne (2020) used computer programming (specifically, coding in Python) to help to reinforce relationships between counting processes and sets of outcomes for students, showing how students' understanding of outcomes could be leveraged by their implementation of processes written in the form of computational commands.

In considering the focus on sets of outcomes, Lockwood (2014) promoted what she called a set-oriented perspective, which is a way of thinking where reasoning about sets of outcomes is central to the counting process. This is a fairly broad construct that emphasizes the importance of having students conceive of the outcomes they are trying to count, rather than focusing primarily on formulas or procedures, which are sometimes not well understood or justified (e.g., Annin & Lai, 2010; Batanero et al., 1997; Lockwood, 2014).

In considering our students' work in this current study, we look for evidence of this particular perspective, which is demonstrated by utterances and inscriptions that suggest they are reasoning about counting as enumerating sets. As we will discuss further, the students reasoned about sets of outcomes created using odometer strategies, which resulted in lists of outcomes. Hence, in our case a set-oriented perspective practically means reasoning about sets that have certain regularities when represented as a list created through an odometer strategy. Even more, we suggest that the students simply having a set-oriented perspective (as defined by Lockwood, 2014) is not sufficient for solving the tasks we pose, and that they need to understand how specific processes relate to the lists of outcomes generated by those processes. We will thus argue that there are additional aspects of the set-oriented way of thinking that arise in our data as students connect specific counting processes both to overall sets of outcomes, and to specific organizations of outcomes (which we call lists of outcomes). We articulate those aspects as additional features of a set-oriented perspective that are potentially necessary for students to understand the relationship between counting processes and lists of outcomes. We now specify terms we are considering in this paper, characterizing how they represent particular (slightly different) instances of what Lockwood

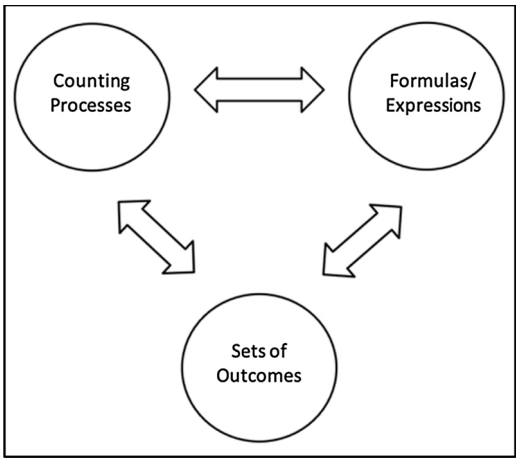

Fig. 1. Lockwood's (2013) model of students' combinatorial thinking.

has previously described.

3.2. Lists of outcomes, listing processes, and a mathematical example

Lockwood (2013) has typically talked about *sets* of outcomes, where the term set is intentionally meant to entail the mathematical properties of a set – namely, that repeated elements are not allowed, and that arranging or organizing the elements in a different way does not change the content of the set itself. For example, the set of outcomes from arranging the letters A, B, and C without repeating letters is {ABC, ACB, BAC, BCA, CAB, CBA}. This set consists of those six outcomes, regardless of how they are listed or ordered. Importantly, a set has a unique cardinality that is not affected by how elements within the set are ordered or arranged. When considering combinatorics (and solving counting problems), it makes sense to focus on *sets* of outcomes because we typically want to know the cardinality of the set of outcomes, as that determines the answer to a counting problem. This is at the heart of a set-oriented way of thinking (Lockwood, 2014).

However, counting processes generate outcomes that are organized and listed in a particular way, and different counting processes may generate differently ordered or organized lists of outcomes. For instance, the following arrangements of A, B, C constitute the same set as above, but it is organized differently and is thus a different list: {ABC, CBA, BCA, BAC, ACB, CAB}. We suggest that while it can still make sense to discuss the outcomes as a set, focusing on *sets* may distract from the fact that a particular counting processes can be thought of as an enumeration of the outcomes in a specific *order*. We will instead argue below for considering *lists* of outcomes, where a list differs from a set by placing an order on the elements.

3.2.1. Listing processes

In particular, we define the term *listing process* to be a specific type of counting process that generates and enumerates the outcomes, resulting in a list of outcomes. We include in this definition counting processes that describe how to generate and enumerate the outcomes, provided that a unique list of outcomes satisfies the process. That is, a listing process must contain a sufficient amount of detail so as to be able to produce a list of outcomes, where there is only one possible order that the outcomes could be in. We require this level of detail intentionally, because answering questions regarding the organization of the list (including the *m*th element questions we discuss in this paper) requires that the outcomes be placed in an order that is unique. Yet, we want to include entire physical lists of outcomes (including those printed on a computer screen), as well as partial lists of outcomes (provided they are sufficiently detailed), and descriptions of lists of outcomes (such as a computer program that would produce a list of outcomes if ran).

Generally, as a particular kind of counting process, a listing process can be used for many types of sets of outcomes. For the tasks discussed in this paper, we focus exclusively on lists where the individual outcomes are encoded as arrangements of objects (including words and characters). This includes Cartesian product problems, and problems of arrangement without repetition. Common listing processes for these problems use an odometer strategy (English, 1991, 1993 ; Halani, 2012), where one object cycles through all possible values while the other objects are held constant. All data reported on in this paper use such listing processes.

3.2.2. An example of a listing process

As an example of how a specific listing process and the broader counting process characterized by an odometer strategy can reinforce each other, we consider generating arrangements of the letters in the word PHONE. This problem is a running example that we will use when discussing student work. We intentionally use PHONE, and not, say, ABCDE, as the letters are not organized alphabetically and students must critically think about how the order of the letters affects the list of outcomes they produce. A typical solution to such a problem might entail reasoning about the positions of the letters in each arrangement, which is reflected in the broad counting process described as follows. We start with five positions that represents the 5-character word that our outcome will be, and we note the that first position can be occupied by one of five letters. For each letter we choose for the first position, the second position can be occupied by one of four letters, and there are thus $5*4 = 20$ ways to occupy the first two positions. We can repeat this process for the remaining three positions to result in $5*4*3*2*1 = 120$ ways to arrange the letters. Again, this broad description represents a counting process that involves reasoning about the positions of the characters, and permits an application of the odometer strategy (English, 1991, 1993; Halani, 2012).

If we were to list outcomes according to this counting process (using the odometer strategy), the listing process would generate a list of outcomes with a specific structure. In particular, the resulting list of outcomes can be grouped into five parts that are distinguishable by the letter in the first position; the first group of 24 outcomes begin with P, the second begins with H, the third begins with O, the fourth begins with N, and the fifth begins with E. Then, each of these parts has four parts, which are similarly distinguished by the letter in the second position. Likewise, each of the four parts has three parts, and so on. In this way, the set of outcomes that resulted from our listing process has a structure that reflects the expression 5*4*3*2*1. While 5! is a more typical way to write this, we will use repeated multiplication to draw attention to the relationship between the expression and the structure of the list. Fig. 2 shows a visualization of how the arrangements of PHONE beginning with the letter N are structured as four groups of three groups of two groups of one element (these represent the 73rd through 96th outcomes, but we could imagine four other similarly structured outcomes that start with the letters P, H, O, and E). This more specific generation of the complete list of outcomes reflects the result of our listing process.

Thus, another way to frame prior research on the relationships between counting processes and sets of outcomes (e.g., Lockwood, 2013; Lockwood et al., 2015) is that a certain counting process will generate a list of outcomes. If the process is systematic enough, one can actually predict where a certain outcome will occur in that list, or what outcome is located in a certain position in the list.

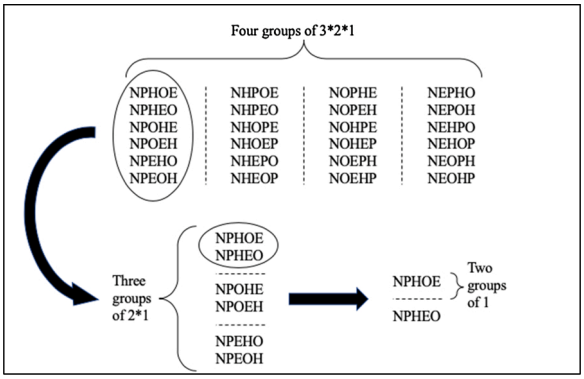

Fig. 2. Arrangements of the letters in PHONE that begin with N.

3.3. Motivating data examples

Given Lockwood's (2013, 2014)) focus on reinforcing sets of outcomes, one might wonder whether or not this reinforcement is something that students need help to reason about. Here, we briefly offer additional motivation for why we want to find ways to help students connect counting processes and lists of outcomes (and why we might want to implement tasks like the *m*th element tasks to help us do that). We highlight two brief examples that demonstrate that students do not always connect counting processes and lists of outcomes as precisely or explicitly as we may want, even if they demonstrate that they are using a set-oriented perspective. Here students connected counting processes and lists of outcomes (via listing processes) in ways that were incorrect or incomplete. We highlight these not to paint the students negatively or to criticize their work, but to emphasize that connecting counting processes and lists of outcomes is not trivial and requires attention and intervention. These data are taken from phases of the broader project in which our study is based, in which students used a computational setting (computer programming in Python, specifically) to solve counting problems by writing computer programs that listed the set of outcomes. By presenting these examples, we seek to make the case that there are connections between listing processes and counting processes that are not immediately clear to students who list outcomes and use a counting process separately. In each case, the students[1] used a computational environment to implement a listing procedure after finding a mathematical expression for the number of outcomes on paper.

The Heads/Tails Problem shows an instance where students wrote a listing process but incorrectly predicted the order of outcomes of their listing process. The Password Problem demonstrates a student giving a correct but incomplete description of the connection between a counting process and the list of outcomes. Again, broadly these examples motivate our goal of developing and implementing tasks (such as the *m*th element task) that will help to reinforce these ideas for students.

3.3.1. The Heads/Tails problem

In this problem, we asked undergraduate students Charlotte and Diana (pseudonyms) the Heads/Tails problem, which asked them to write a computer program to list the outcomes of flipping a coin seven times. The computer program they wrote listed the outcomes using a standard odometer strategy (Halani, 2012), which results in a lexicographic list. Charlotte and Diana stated that there were $2^7 = 128$ outcomes. The following excerpt shows a brief discussion of the structure they predicted the list would have before they ran the code.

> Interviewer 1: Yeah. And then how do you think they're gonna be structured?
> Charlotte: It'll be like – They all seem – I don't know.
> Diana: I think it'll probably do like seven heads first, and then six heads and one tails, and five heads and two tails, maybe?
> Charlotte: And go all the way down?
> Diana: And go all the way down.
> Charlotte: And then go to seven tails, and all the way down?

Notably, the students here did seem to demonstrate a set-oriented way of thinking, as they reasoned about and articulated specific outcomes that would be generated. Yet, they were not correct in their description of the structure of list. By articulating the number of heads and tails followed by "go all the way down," we take that Charlotte and Diana were describing that the list would be organized by the number of tails in each outcome. This organization differs from the odometer strategy significantly, which may indicate that the pair were describing an organization they thought was appropriate for the set of outcomes, but not one that was informed by the computer program they wrote. Further, we would categorize Charlotte and Diana's description as incomplete, because they did not

[1] These students are Charlotte, Diana, and Charlie (pseudonyms). Charlotte and Diana were female undergraduate students who were recruited from a vector calculus class. Charlie was a male undergraduate student recruited from an Introduction to Computer Science class. These three students had not taken a discrete mathematics class in college, and selection interviews (in the case of Charlotte and Diana) and selection surveys (in the case of Charlie) indicated that they did not try to recall formulas when solving counting problems.

articulate how the outcomes with a fixed number of heads would be organized.

After running their code and examining the printed list of outcomes, we saw more evidence that their initial listing process was not fully conceived. After they ran the program, Charlotte observed that "the last column goes heads, tails, heads, tails, heads, tails," which would not occur in the organization they previously described, and Diana stated "I'm not exactly sure how it's sorting it." They also indicated that an outcome with six heads was listed after an outcome with five heads, which demonstrated that they did not see their predicted organization in the list of outcomes. Even though the students had found the correct number of outcomes, it is not clear to what extent they understood how their listing process organized the list of outcomes, even after seeing the entire list of outcomes. Again, this is not meant to criticize the students, but to point out that reasoning about organization of a list of outcomes is non-trivial, even if the students write a computer program to list their outcomes in a specific manner.

3.3.2. The passwords problem

In this problem, we asked an undergraduate computer science student Charlie to find the number of three-letter passwords that can be made from the letters A, B, C, D, E, and F that do not repeat letters. There are $6 \cdot 5 \cdot 4 = 120$ such passwords, which Charlie argued by appealing to the number of available letter options at each stage of a three-stage process. Charlie's listing process also used the odometer strategy, which reflects six groups of five groups of four. He was able to talk through his computer program line by line, indicating what each line accomplished, which we take to be evidence that he understood the listing process. The following exchange shows how Charlie connected his counting process to organization of the list of outcomes.

> Interviewer 2: Do you see – is there any kind of the way in which that six times five times four is reflected in that list? And if the answer's no that's fine but I just am curious.
> Charlie: Um. […] I'm trying to – what I'm initially thinking is looking at how many times each letter of the first one comes up so A, so [points and counts by hand] 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20. So, the first letter being A comes up 20 times. So that kind of confirms what I was thinking earlier where the first letter was just A then it's going to print 20 times. So, if the very first letter is A or say one specific value, we know it's going to print 20 times and that first letter which starts off as A can then – then has six possible different outcomes – A, B, C, D, E, F.
> Interviewer 1: Mm-hmm.
> Charlie: So, that's kind of thinking of it as like six different groups of 20 all printing at once. So that's kind of where I get the 120 from.

In this excerpt, we interpret that Charlie saw the list as being partitioned into six groups of twenty, but he did not see each group of twenty as being partitioned into five groups of four. Charlie later reinforced this position, as seen in the following excerpt.

> Interviewer 1: So, you said you saw this entire list as kind of in six chunks – you know by – you said like As for example. Do you see it as something similar within just the As or just thinking about it as one big clump of 20?
> Charlie: Yeah, I'm just kind of thinking of it as like one big clump of 20 possible outputs when the value is A.

Even after questions intended to lead Charlie to notice that each group of twenty could be further organized as five groups of four, he repeatedly stated that he saw the organization as six groups of twenty. We take this episode to be a demonstration where a student observed the broadest organizational patterns, but not the narrower organizational patterns. Because we see value in connecting mathematical expressions to organizations of lists of outcomes correctly and thoroughly, we are interested in ways to engage students in reasoning about both broad and narrow organizational patterns. These data examples make clear that a set-oriented perspective (Lockwood, 2014) is not sufficient by itself *to convey these ideas of connecting processes and outcomes*, and that other interventions are needed. Because a computer programming environment provides a means for students to reason about lists of outcomes (e.g., Lockwood & De Chenne, 2020), we are also motivated to use the computer to generate artifacts for students to use in these interventions.

4. Methods

4.1. Participants

The data for our results come from two groups of students. The first group consisted of three undergraduate students (pseudonyms Dennis, Veronica, and Jason) who were interviewed for seven 90-minute sessions. All three of the students were CS majors enrolled in and recruited from an introductory computer science (CS) course at a large public university. One student was taking an introductory discrete mathematics course concurrently with the interviews, but selection interviews revealed that they had not taken courses involving counting prior to the interviews and they were not familiar with counting problems and formulas.

The second group of students consisted of two undergraduate students (pseudonyms Zach and Levi) majoring in mathematics. Data were collected during their upper-level mathematics course designed for pre-service teachers (taught by the second author); Zach was a math major not enrolled in the secondary program, while Levi was a math major enrolled in the secondary program. They were senior-level mathematics majors with strong mathematics backgrounds; both students were taking a combinatorics class for upper-level undergraduate students and graduate students at the time of the interviews, and, as math majors, they had taken both a discrete mathematics course that involved basic counting. Data collection occurred during class, and one member of the research team (the first author) filmed Zach and Levi and asked them questions as they worked through tasks; we also recorded the screen of the computer on which they were working.

We note that we are only sharing in-depth data from the two groups listed above, in part because they offered particularly

compelling examples, and because it was practically feasible to share results from a smaller set of students for this paper. However, we used this task with a slightly larger pool of students (an additional set of 3 students and an additional 4 pairs of students), so we gave the task to 16 total students, whereas we are reporting on just 5 students. We point this out to acknowledge that we are not attempting to overly generalize our findings, and our goal is not to claim we have a set of tasks that will certainly be effective for all students. Rather, we want to show what kinds of reasoning, conceptions, and practices can be elicited by this kind of task. That is, we intentionally focus in more detail on fewer students, because it allows us to examine, analyze, and report their reasoning and activity in more depth. We acknowledge that our small *n* is a limitation of our study.

4.2. Data collection

The data were collected during teaching experiments (in the sense of Steffe and Thompson (2000)) with the first group of students, and during the classroom activity with the second group of students. In both cases, we collected video and audio-recording of the students, and we recorded the screens of the computers on which they worked. These three sources of data allowed us to have a clear view of the students' work, as it allowed us to triangulate what students were writing on the computer, physically articulating, and saying audibly. The data for the first group (Dennis, Veronica, and Jason) were taken in the context of two groups of three students working simultaneously on the same tasks. The data for the second group (Zach and Levi) were taken in the context of a classroom, where members of the classroom were paired and they worked on the same tasks as the first group, with the addition of a homework assignment. Our data collection is closely involved with the particular tasks that we designed and implemented, which we describe in the next section.

In both cases, students worked in groups to solve combinatorial problems in a computational setting. These students worked in Jupyter notebooks (which allow them to write Python code and text simultaneously) in CoCalc, an online computing environment. Using Jupyter notebooks allowed the students to write Python code and easily run the code. Further, Jupyter notebooks allow for multiple cells of code, where each cell can be run apart from each other on the same page. Hence, students solved different problems on the same page, allowing them to scroll up or down in order to review their previous work. Jupyter notebooks also allows students to write using Markdown syntax, so they could give written responses to questions inside the notebook. Each group worked on one computer, they were expected to take turns writing computer code, and they had paper and writing utensils. Most of the combinatorics problems asked them to create and implement listing processes that would list sets of outcomes they had counted by hand. Hence, in the tasks described in the next section, the students had already written computer code to create a list of outcomes they subsequently used to solve the tasks.

4.3. Tasks and mathematical discussion

4.3.1. Tasks

The tasks we focus on in this paper involve asking students to find a certain outcome in a list of outcomes that has been generated by a listing process. In particular, we ask questions of the form, "What is the *m*th outcome in your list of outcomes?" where *m* is an integer typically chosen to be large enough to deter the student from simply counting (by hand) the first *m* outcomes in their list. Ultimately, we argue that simple variations of this kind of task can be used with a variety of undergraduate students to elicit combinatorial thinking and reasoning.

We start by asking for specific numbers, such as the 15th, 80th, or the 165th outcome. Then, a natural extension is to investigate how one might find the general *m*th outcomes in a list, where *m* is a variable and not a particular number. This kind of question only makes sense if there is some ordinal list of the outcomes, and we are thus intentional about the language of "your list of outcomes," because that clarifies that the *m*th element must be found in relation to a particular list. Because there is no objective correct answer to the question without reference to a particular list, it can only be asked after a student generates (or is given) a particular list by using a particular process. In other words, different processes that produce the same *set* of outcomes may produce different *lists* of outcomes, and so the *m*th outcome in those two lists may be different. Nonetheless, as we will see, even for particular lists that students generate, it makes sense for them to think about what elements might occur in what position, and we argue that this can be an effective way for them to think about their counting processes and their lists of outcomes.

We initially designed these tasks in the context of students first articulating a listing process, and then implementing the listing process in a computer program to produce a printed list of outcomes. Thus, when we asked for a particular outcome, it was always in the context of a list the student had already generated via the computer. Lists that order the outcomes in a different way yield different outcomes, and while sometimes students listed the outcomes in a novel way, typically they listed outcomes using the odometer strategy, generating lists that were organized in lexicographic order. By lexicographic, we mean a generalization of alphabetical ordering where the alphabet is an ordered list of distinct symbols rather than a language's alphabet specifically. Importantly, the lexicographic ordering depends on that initial ordering of the original alphabet, and in problems in our study, students listed lexicographically based on how elements in a set were written in problem statements. That is, when students were arranging letters in the word PHONE, they treated the letters P, H, O, N, and E as appearing in that order (and so that was the alphabet), and not organized in a traditional alphabetical way, as E, H, N, O, and P. This is related to our point above that the tasks are only appropriate to ask in relation to a particular listing process.

4.3.2. Illustrative mathematical examples and mathematical discussion

We briefly discuss an example of the kind of questions we posed to students, and we will return to it when we describe student work.

For the given list of arrangements of the word PHONE described in Section 3.2.2, we pose the question, "What is the 80th outcome in this list?" We describe how to find the 80th outcome, noting as above that the list we are using is in lexicographic order. PHONE was intentionally chosen so as not to be alphabetical (such as ABCDE is), as this requires an additional level of coordination when reasoning about the order of the letters chosen at each step of the listing process.

To find the 80th outcome, we recognize that the list is partitioned into five parts (distinguished by the first letter), each of which has $4*3*2*1 = 24$ outcomes. Outcomes 1–24 start with P, outcomes 25–48 start with H, outcomes 49–72 start with O, and outcomes 73–96 start with N. Hence, the 80th outcome must start with N. Alternatively, as $3*4! < 80 < 4*4!$, the first letter must be the fourth of the alphabet. The outcomes starting with N are partitioned into four parts as noted in Fig. 2 (the possible second letters being P, H, O, and E), each with $3*2*1 = 6$ outcomes. So, as $3*4! + 1*3! < 80 < 3*4! + 2*3!$, the 80th outcome must be in the second group, and the second letter must be H. The outcomes starting with NH are partitioned into three parts (the possible third letters being P, O, and E), each with $2*1 = 2$ outcomes. To keep track, we point out that the three parts that start with NHP, NHO, and NHE count outcomes 79–80, 81–82, and 83–84, respectively. Hence, the 80th outcome is in the first of these parts, and the third letter must be P. Finally, the outcomes starting with NHP are partitioned into two parts (the fourth letters being O and E), and as the 80th outcome is the second of these then the fourth letter must be E, which implies the fifth letter must be O. Hence, the 80th outcome is NHPEO.

We make a couple of comments about the kind of reasoning involved in solving this task. First, we note that to answer the question sufficiently of what the 80th outcome in the arrangements of PHONE is, we must be aware of the structure of the list of outcomes. The above solution was broken into five steps, one for finding the letter in each position. For each position, we identified the letters that could occupy it and determined how many arrangements of the remaining letters began with each. This reinforced the connection between the counting process and the list of outcomes (via the implementation of the listing process we described). The types of counting problems we used when asking *m*th element questions were ones that could be listed using an odometer strategy, which reflects a very regular structure in the list of outcomes that we describe as multiplicative. Solving *m*th element questions for lists which are not multiplicatively regular may be more difficult. However, listing processes for problems not solved by the multiplication principle typically have some regularity (albeit not multiplicative regularity), and tasks that make use of or rely on this regularity may be posed to foster connections between counting processes and lists of outcomes.

This type of question poses challenges for students with a variety of experience with counting and computing, which is a point we will demonstrate in our data. As we aim to show, the *m*th element question necessitates attention to the set of outcomes that are being counted and their connections to the counting process, and it does not overemphasize formulas that might be applied without understanding (e.g., Braithwaite & Goldstone, 2013; Lockwood, 2014).

4.3.3. Description of computational environment

The data we present in this study are instances of students reasoning about lists they had created by writing a computer program in Python. We are intentionally not wanting to create a dichotomy between computational and by-hand work, and we follow other researchers (e.g., Farris et al., 2020; Sengupta et al., 2018; Vieira et al., 2019) in taking a broader view of a computational setting as encompassing many kinds of reasoning and activity (including unplugged or by-hand work). The computational environment undoubtedly influenced the student work, as they were often asked to connect aspects of their computer programming to the work they did by hand. All of the work we present is work that the students did by hand, but it occurred within the broader experience of them engaging with the computer and reasoning about lists they had created by writing (or describing) a computer program. We thus acknowledge the broader computational environment in which our students engaged; but, as we will note in the avenues for future research, we also suspect that students could engage productively with *m*th element tasks in situations that do not also involve computational aspects. In this section, we provide an example of what students would have programmed in Python during the interviews.

We continue our example of the PHONE problem and relate it to a computational setting. We can take the combinatorics problem "How many ways are there to arrange the letters in the word PHONE?" and translate it to the computational problem "Write a program to list all the arrangements of letters in the word PHONE." While the combinatorics problem focuses on finding a numerical answer, the computational problem focuses on the structure of the set of outcomes and challenges the student not only to find a way of representing outcomes, but also to articulate a process that generates the outcomes. By articulating the listing process, the student may be able to create meaningful distinctions between problem types and deepen their understanding of how and why formulas correspond to the sets of outcomes they count.

For the tasks we gave the students, which we discuss further below, we would first have them think about what the output of code

```
phone = 'PHONE'

for i in phone:
    for j in phone:
        if j!=i
            for k in phone:
                if k!=i and k!=j:
                    for l in phone:
                        if l!=i and l!=j and l!=k:
                            for m in phone:
                                if m!=i and m!=j and m!=k and m!=l:
                                    print(i+j+k+l+m)
```

Fig. 3. Python code that generates a list of arrangements of the letters in the word PHONE.

(such as the code in Fig. 3 below) would yield. Sometimes the students would generate the code themselves, sometimes they would examine existing code. Then, once they had predicted the output, and then ran the code to see the output, we would ask them questions about the *m*th outcome in the list that the code had generated.

We argue that Fig. 3 reflects the mathematical expression 5*4*3*2*1 we have previously described. In particular, for each of the 5 letters in PHONE (represented in the code by *i*) the code cycles through the 4 letters not equal to *i* (represented in the code by *j*), and for each *i* and *j* the code cycles through the 3 letters not equal to *i* or *j* (represented in the code by *k*). This process repeats for *l* and *m*, and then prints the outcome (*i*,*j*,*k*,*l*,*m*), revealing a 5*4*3*2*1 structure in the list. In this way, we point out that this particular code creates a lexicographic list, essentially creating a list of outcomes that is organized according to 5 big sections that start with each of the five letters, as we have described above.

As shown in our motivating examples, generating a set of outcomes according to a listing process may not be enough to help students identify a corresponding structure in the list of outcomes or how the listing process relates to an imagined counting process. In such cases, students may use the resulting outcomes to verify that their code produces their desired list, but they do not focus on identifying or leveraging any structure within that list. While we do not mean to suggest that students must be able to identify structure within the list of outcomes in order to be successful counters, it may be the case that understanding the structure can lead to a richer understanding of the counting process and the formulas (this is in line with findings from Lockwood (2013, 2014)).

4.4. Data analysis

The audiotapes from the teaching experiments were transcribed. For the results in this paper, we reviewed the transcripts for episodes where the students worked on *m*th element questions. We isolated these episodes and engaged in open coding and memoing, allowing us to analyze the students' reasoning when solving these problems, and to see how their reasoning changed over time. Our analysis focused on recurring solution techniques (described in our Results section), and how these solution techniques used underlying structure in the list of outcomes and referenced counting processes they had previously used to find numerical solutions to counting problems. In doing so, we constructed a narrative of how each group solved *m*th element problems, how their solutions demonstrated their understanding of counting problems, and how their solution techniques became more refined as the episodes progressed. We selected data to present in this paper so as to demonstrate the progress each group of students made in refining their solution processes. As the data will show, these episodes demonstrated how the students used their understanding of counting problems to reason about structure in lists of outcomes and to create a solution technique that they used on multiple problem types.

5. Results

We have reported more details on aspects of this project elsewhere (Lockwood & De Chenne, 2020, 2021), and here we share findings from two groups of students. Two solution methods emerged in these data, and we refer to them as the *narrowing process* and *factorial representation* methods. While the Factorial Representation method is an extension of the narrowing process, we will discuss the two separately. These solution methods draw from listing processes the students created when writing computer code in Python to solve counting problems, where the students justified the methods by using the regularities of the lists as they relate to a counting process. Both groups of students used the narrowing process in similar ways, but only Levi and Zach used the factorial representation method. Because the narrowing process was used in similar ways between the two groups of students, for brevity we will not report Levi's and Zach's use of it. In Section 5.1 we will describe Dennis', Veronica's, and Jason's development of the narrowing process, and in Section 5.2 we will describe Levi's and Zach's development of the factorial representation method. By discussing the developments of the methods, we attempt to demonstrate the insight that the students had into the connections between their counting processes, sets of outcomes, Python code, and the lists of outcomes the Python code produced. We demonstrate how these tasks drew out aspects of a set-oriented way of thinking that necessitated not just thinking about outcomes, but connecting those outcomes to listing processes. As noted, while we focus just on two groups of students here, this work is representative of what we found with another set of three students and in the entire class in which Zach and Levi were enrolled (four additional pairs of students).

5.1. Narrowing process

In this section, we discuss how students Dennis, Veronica, and Jason solved questions of the form "What is the *m*th element in your list of outcomes?" by employing a process to iteratively refine (or narrow) which element was the *m*th. We call this process the "narrowing process" because Veronica described it as a process of continuously narrowing down the outcomes. Enacting this process successfully required understanding that the list of outcomes could be partitioned in nested groups of equal size, each of which had a regular structure. The process also necessitated counting the number of outcomes in each part (or sub-part), thereby connecting the counting process to the list of outcomes explicitly at every step. We did not instruct the students to use this solution method, and here we document how the students created and used the narrowing process in increasingly sophisticated ways. We will show that the students developed meaningful connections between their solutions to these problems, the counting and listing processes they employed when solving counting problems, and the lists of outcomes in the problems.

Students used the narrowing process on a list of outcomes they had produced as a result of writing computer code in Python. These lists were for problems that could be solved by using the multiplication principle, and the lists were produced using the standard odometer strategy. The numerical solutions were products, such as $6 \times 5 \times 4$, where each term in the product corresponds to the number of elements cycled through in each stage of the odometer strategy. For example, a list produced by the odometer strategy that

has size $6 \times 5 \times 4$ can be partitioned into six groups of five groups of four elements. Each of these groups correspond to cycling through the possible elements as elements in other groups are held constant. In such a list, the narrowing process is the process of determining the choice at each stage by determining the lower and upper bounds for the outcomes that used that choice. For example, a list of size $6 \times 5 \times 4$ can be partitioned into six groups of twenty, where each group is distinguished by the choice made in the first stage. Hence, choice 1 accounts for outcomes 1 through 20, choice 2 accounts for outcomes 21 through 40, and the other outcomes are partitioned similarly. Accordingly, if we were to identify the 28th outcome in the list, we could narrow down the outcome because we know that choice 2 was taken at the first stage. By narrowing the choices at subsequent stages, we can determine the *m*th outcome.

5.1.1. Narrowing process on Cartesian product problems

In this section, we describe how students Dennis, Veronica, and Jason developed the narrowing process on Cartesian product problems. Cartesian product problems are those where the set of outcomes can be described as a Cartesian product. Lists of these outcomes have especially regular features because the counting process used can be broken into a number of stages where the options (rather than just the number of options) at each stage are the same.

Dennis, Veronica, and Jason first used the narrowing process to find the 15th outfit in the list that was produced from the following code in Fig. 4, which produces all ordered triples of the given sets of shirts, pants, and belts. This question was posed in the context of their work on the Outfits problem, which asks about outfits created from four types of shirts, three types of pants, and two types of belts. The code in Fig. 4 was given to the students, and before we asked for the 15th outfit, we had already asked the students to find a mathematical expression that gives the number of ordered triples, to create a tree diagram that gives the number of ordered triples, and to relate the structure of the code to the tree diagram and the mathematical expression. They had completed these tasks successfully.

This code produces a list of 24 outfits, one for each combination of shirt, pants, and belt, and the students had already run the code to produce the list. We asked them to find the 15th outcome without counting by hand, but rather to reason based on the structure of the list. Due to the small number of outcomes, they could easily check the answer by counting down the list of outcomes until they reach the 15th – this was intentional, and as the first problem of this type it was designed so that the students could easily check their work. To find the 15th outfit, Dennis used the following reasoning:

> *Dennis*: The way that I would look through this is, so you have t– so you have a tee, so that–that means, so we have four different shirts, three different pants, two different belts for each shirt, four shirts each shirt has three different possible paths, each pant–pair of pants has two different possible paths. So, two times three is six, so that's six possible paths per shirt. So then that tee that's six. Polo, that brings us up to 12.

We infer that Dennis began his process by reasoning about the number of options for each shirt by connecting the list of outcomes ("four shirts, each shirt has three different possible paths, each pant—pair of pants has two different possible paths") to a mathematical expression ("two times three is six, so that's six possible paths per shirt"). He then used the fact that 15 is in the third group of six to argue that *Sweater, Khaki, Brown* is the 15th outcome in the list.

Later, we asked these students to solve the Heads/Tails problem, which asked them to describe code that would list all the outcomes of flipping a coin 8 times (there are 256 outcomes of flipping a coin 8 times.). This question immediately followed a similar question involving 4 coin flips ("Write a program to list (and determine the total number of) outcomes of flipping a coin 4 times"), whose solution the students generalized to the case of flipping a coin 8 times. They solved the Heads/Tails problem by describing a program with 8 nested loops, each cycling through the alphabet {H,T}, and we then asked them to find the 165th outcome in the list that their code would generate. The students reasoned that the first 128 outcomes started with heads, and so the first column must be T, and, within the tails, the first 64 outcomes are heads, so the second column must be H. In particular, when determining that the third column must be a T, Veronica made the following observation, and she explicitly used the language of narrowing:

> Veronica: So, I'm like continuing to narrow it down. So then within these 64 tails, heads, 32 of them are going to be tails, heads, heads and that would be 128 and 32 which is 160 so that's too low. So, we want the 32 that's going to be tails, heads, tails and then that's going to bring us to, um, so that's going to be the second half.

By "continuing to narrow it down," Veronica described a process of finding the correct option of heads or tails in each position by narrowing the location in the list down to the part of the partition where the 165th outcome occurs. Of the 256 outcomes, the first 128 began with heads, and the second 128 began with tails. Because 165 is greater than 128, the 165th outcome must have begun with a tail, and the students narrowed their search to only the latter 128 outcomes. Similarly, the students reasoned that of the latter 128 outcomes, the first 64 (outcomes 129 through 192) had heads in the second position, and the second 64 (outcomes 193 through 256) had tails in the second position. Hence, the second position must have had a head, and the students narrowed their search to outcomes 129 through 192, whose first two positions are tails and heads.

5.1.2. Narrowing process on arrangement without repetition problems

The two problems discussed above (the Outfits problem and the Heads/Tails problem) focused on finding outcomes of a Cartesian product problem, where the listed outcomes can be partitioned in such a way that the structure of each part is identical. In addition to Cartesian products, we asked the students to solve problems that involved arrangements without repetition (such as the PHONE problem). While the outcomes of Cartesian products can be listed and partitioned so that each part has an identical structure, the list of outcomes for problems involving arrangement without repetition can be partitioned so that each part has the same size, but the structure of each part is no longer exactly identical. For example, when finding the third coin flip of the 165th outcome in the Heads/Tails problem, the structure of the partition of outcomes starting with T H _ _ _ _ _ _ does not depend on whether the first two coin flips were heads or tails. There will always be the same number of ways to arrange the following positions, and the ways in which the

```
Shirts = ['tee', 'polo', 'sweater', 'tanktop']
Pants = ['jeans', 'khaki', 'shorts']
Belts = ['brown', 'black']

outfits = 0
for i in Shirts:
    for j in Pants:
        for k in Belts:
            print(i,j,k)
            outfits = outfits+1
print(outfits)
```

Fig. 4. Dennis, Veronica, and Jason's code to solve the Outfits problem.

remaining six positions are arranged will be identical no matter what the first two elements are (TT, TH, HT or HH). For arrangements without repetition, this is no longer the case. When arranging the letters of PHONE, if PH are the first two elements, there are different options for the remaining three letters than if EO are the first two elements, even though there are still six options in either case. Hence, when using the narrowing process to find specific outcomes from the list of outcomes, one must be cognizant of the structure of each part of the list of outcomes.[2]

We asked the students to create code that generated and counted all arrangements of the word PHONE (the problem statement is in Sections 2 and 3). They wrote the code presented in Fig. 2 above, which represents an ordered list of outcomes that is lexicographical according to the ordering of the alphabet of letters P, H, O, N, E in that order. After writing the code and finding the number of arrangements, we asked them to find the 80th outcome in their generated list. To answer the problem, they continued to use their narrowing process, now applied to a problem where the letter in a given position affects the possible letters in subsequent positions. The following describes how they found the first letter.

> Jason: So then six times four is 24. So you know that the first 24 are gonna start with P.
> Veronica: Okay.
> Jason: We know it doesn't– so we know it doesn't start with P because 80 is greater than 24.
> Dennis: So then 48—
> Jason: We know that the next one is going to be, and that's going to get 48 options.
> Dennis: And then 72– and then
> Jason: and then 72 and so then
> Veronica: So we know it starts with an N.

When using the narrowing process, they described that the outcomes were generated in groups of 24, as they count in multiples of 24 to find the first letter. To find the second letter, the students attended to the letters that could occupy subsequent positions after N is used, which differs from the process used in prior problems.

> Jason: So the N– the… 72 is the last O. So the 73 is going to be N. So 73 through 78–three four, five, six, seven, eight, right–is going to be NP. And then the next is going to be an NP is NH. The first NH. The second NH. There's 70… is 80. So it's the second number of NH series. NH. Where's PHONE written out? N. H. P. O. E. And then you have NH.

In these excerpts, the students articulated that the remaining arrangements were grouped in sets of six, as they counted the six numbers starting at 73, linking the counting process to a listing process. However, after finding that the second letter was H, the students returned to how PHONE was written in the original alphabet (P, H, O, N, E, in that order) in order to determine the remaining letters.

> Veronica: Then have four options. P only gets us to 78, so we know we have to go with H. So that's the second one. Then we eliminate H as an option and we just have POE and that gets us to 80. So we know that that's the one that we have. And then the, so the 79th option is going to be the first option. Um, so that would be the NHPOE.
> Dennis: And so this would be NHPEO?
> Veronica: Yeah. Yes. Yeah. Yeah. So that's exactly what we got.

As the students applied the narrowing process, they used the fact that nested partitions occur by eliminating letters that have been used. Further, they used the multiplicative nature of the set of outcomes in each step of their narrowing process. We contend that this narrowing activity can reinforce the relationship between the counting process and lists of outcomes by focusing on how a given listing process generated the entire list of outcomes, because it makes explicit the idea that at each stage in the counting process the options will be organized in some particular way. This can serve to justify why the counting process works.

The students applied and refined this process in subsequent problems, demonstrating their understanding of how the counting process relates to the structure of the list of outcomes. One such question was the ROCKET problem, which asked them to identify the 80th arrangement of the word ROCKET, where the students followed the same listing procedure as they had with PHONE (again, using

[2] This is an important realization when justifying the multiplicative nature of the mathematical expression. See Lockwood et al. (2017) and Lockwood and Purdy (2019) for more in depth discussions of multiplication in counting and independence as related to Cartesian products.

an odometer strategy). The students' work in Fig. 5 shows how they applied their narrowing process (specifically, they created each nested part by removing a possible letter), and it shows how finding the 80th arrangement of the word ROCKET connects the counting process directly to the structure of the list of outcomes. In Fig. 5, the first column (from the left) shows the letters R, O, C, K, E, and T. Above the column they wrote 6*5!, which we infer was meant to articulate that the list of outcomes was partitioned into 6 groups of 5! = 120, as evidenced that next to R was 120, next to O was 240, and the students labeled each subsequent letter with multiples of 120. Further, the R was circled and did not appear in the next column, which we interpret was meant to show that R was the first letter in the 80th arrangement. In the second column they listed the letters O, C, K, E, and T, and above the column they wrote 6*5*4!. This seems to demonstrate that the students were thinking of the list of outcomes as further partitioned into groups of 4! = 24, as evidenced by the numbers labeled next to the letters. For subsequent columns, the students repeated the process of writing the possible letters, writing the size of each part, and labeling each letter with how many outcomes were in each part.

We argue that their narrowing process is visually represented in the figure, as the students explicitly characterized the size and structure of the (sub-)parts at each step, while also demonstrating that a choice in letter for each position determined the possible choices of letters in subsequent positions. This figure also demonstrates that the students symbolized their narrowing process in part through the use of mathematical expressions, as characterized the top line of the figure where the students wrote a product for every column. This might indicate that expressions such as $6 \cdot 5!$ and $6 \cdot 5 \cdot 4!$ represented the list of outcomes themselves, rather than the cardinality of the list of outcomes. This minor distinction appears to be an additional aspect of Lockwood's (2014) set-oriented perspective, in that the mathematical expressions were used and manipulated to reason about the list of outcomes, instead of using the list of outcomes to reason about the mathematical expressions.

In this section, we have demonstrated that students solved problems of the form "what is the *m*th element in the list of outcomes" by attending to the relationship between the structure of the list of outcomes and the counting process. In the episodes we presented, the students used their mathematical expressions to articulate and manipulate the underlying structure of the lists of outcomes. We suggest that by doing so, the students reinforced and enriched their understanding of the counting principles used to solve each problem, such as the multiplication principle when used on a problem involving arrangements without repetition. While articulating their solution process and finding an outcome, the students justified their counting process and explained how each step of their counting process corresponded to the structure of the set of outcomes. Further, the students were afforded opportunities to engage in manipulation of structure (as observed in Fig. 5), which is a valuable mathematical practice. Their work on these problems highlights aspects of their way of thinking that extended beyond thinking just about sets of outcomes, but that demonstrated that they connected the outcomes to particular listing processes they had developed. That is, it was not just that they could reason about or articulate what outcomes of the problems entailed, but they could specify how their listing process generated and organized those outcomes, and how their listing process was reflected in their mathematical expression.

5.2. Factorial representations

We now demonstrate how students Levi and Zach used the narrowing process to develop a solution process that uses factorial representations to find the *m*th element of problems involving arrangement without repetition. The end-result of the factorial representation method is a sum, where each summand corresponds to one step of the narrowing process. This solution method can thus be seen as an extension and encoding of the narrowing process. We did not anticipate this solution method being used by the students, yet we discuss in Section 6.3 how learning trajectories from the narrowing process to using positional representations of integers to solve *m*th element problems might be studied in the future. In creating this solution method, these students reinforced counting processes, engaged in mathematical exploration, and encoded a systematic process into a mathematical expression.

Using factorial representations to solve these problems is similar to using fixed base representations (such as base 10 or base 2) to solve problems involving Cartesian products of the same set. For example, if we were to flip a coin eight times, we could easily find a correspondence between sequential coin flips and eight-digit numbers in binary that include leading zeros (such as 00001001 or 10010110), where a 1 denotes a head being flipped and a 0 denotes a tail being flipped. For problems involving arrangement without repetition, instead of a fixed-base representation we can use a mixed-radix representation (Knuth, 1997) called factorial representations. Every non-negative integer can be written uniquely as some $\sum_{k=1}^{N} a_k k!$, where $0 \leq a_k \leq k$, $a_N \neq 0$, and the representation of the integer as $(a_N, a_{N-1}, \ldots, a_2, a_1)_!$ is called the factorial representation. Just as a 0 or 1 might correspond to a head or tails, each a_k corresponds to a choice at the kth stage. Mixed-radix representations are not commonly taught in undergraduate mathematics curricula, and based on Levi's and Zach's responses in these data we do not think that either of them had previously worked with

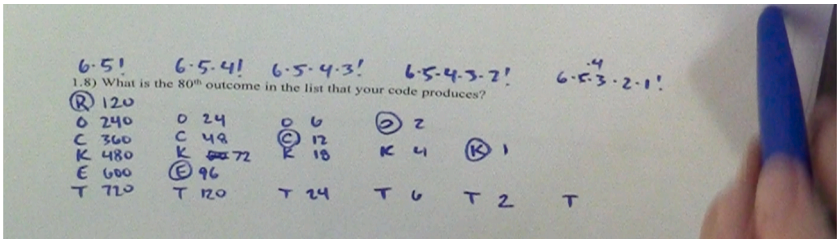

Fig. 5. The students' solution to finding the 80th outcome in their list of outcomes of the ROCKET problem.

mixed-radix representations.

To demonstrate how factorial representations can be used to find the *m*th element of a list of arrangements without representation, consider the PHONE problem which asks for the number of ways to arrange the letters in PHONE (there are $5! = 120$ such arrangements). Similar to the narrowing process, we can iteratively select the a_k s so as to progressively eliminate all outcomes that come before our desired outcome. We will use the expression $a_4 4! + a_3 3! + a_2 2! + a_1 1!$, with each $0 \leq a_k \leq i$. The greatest number we can represent in this way is $4 \cdot 4! + 3 \cdot 3! + 2 \cdot 2! + 1 \cdot 1! = 119$, and the least number we can represent in this way is $0 \cdot 4! + 0 \cdot 3! + 0 \cdot 2! + 0 \cdot 1! = 0$. Hence, there are 120 numbers with these representations, which we will show corresponds to the 120 arrangements of the word PHONE in a natural way. Because we typically start counting lists from the number 1 but the smallest number we can represent is 0, we will show that we can find the 80th outcome by using the representation for 79 (which is the 80th number that can be represented). If the arrangements of PHONE were written in lexicographic order, there would be five groups of 4! outcomes (distinguished by starting with each of the letters P, H, O, N, and E). These five groups correspond to the leading digits of 0, 1, 2, 3, and 4. As we have $3 \cdot 4! \leq 79 < 4 \cdot 4!$, then $c_1 = 3$, which corresponds to the first letter being N. In the narrowing process, we would now reduce our word to PHOE, and repeat the process. As we have $3 \cdot 4! + 1 \cdot 3! \leq 79 < 3 \cdot 4! + 2 \cdot 3!$, then $c_2 = 1$, which corresponds to a second letter of H. In the narrowing process, we now reduce our word to POE. We repeat this process, ultimately finding that $3 \cdot 4! + 1 \cdot 3! + 0 \cdot 2! + 1 \cdot 1! = 79$, which corresponds to selecting N from PHONE, H from PHOE, P from POE, and E from OE, which leaves O as the last letter. Hence, the 80th element in the list of arrangements of the word PHONE (or, equivalently, the 79th indexed element when beginning with 0) is NHPEO. This process of using the factorial representation to encode the relevant information to solve the problem can be repeated for any number between 0 and 119. For other problems involving arrangement without repetition, the process can be extended to any number of objects being arranged.

5.2.1. *Factorial representations for arrangement without repetition*

In this section, we demonstrate how Levi and Zach created and used factorial representations to find specific elements in a list of outcomes. Levi and Zach created this solution method while finding the 80th outcome of the PHONE problem, where they first reasoned that, "We know it comes in chunks of 24" and "So we know we're in the third chunk," and then using the output of their code to count the remaining eight outcomes. This suggests that they weren't just thinking about outcomes, but they understood that their particular listing process had organized those outcomes in a certain way that had a regular structure that they could leverage. That is, once they were within the third chunk (which started at 73) they simply counted up in their list of outcomes to find the 80th outcome of NHPEO. While it was good that they reasoned about large chunks, we did not want them simply to rely on physically counting within the chunk that started with N – we wanted them to develop a more general argument about what the 80th outcome might be. This prompted the first author to ask the question, "How would you guys reason about that if you didn't have the list of outcomes in front of you?" The students then solved the problem without explicitly counting through the output of their code. Similar to the students discussed previously, Zach and Levi started with a narrowing process. However, they began to reason about ways to encode their solution method into a mathematical expression, as demonstrated by the following excerpt.

> Zach: You could say, oh it starts with N because we know it's at least 72 and it's less than 96, so it's gotta be starting with N. And then, so, gosh, we can think about it as a linear combination of the factorials, right?
> Levi: Yeah. Yeah.
> Zach: So it's like four factorial, c_1 four factorial, plus c_2 three factorial, plus c_3 two factorial, plus c_4 one factorial, and last one's fixed based off of these four [wrote the expression in Figure **6**]. So we know that c_1 is three, so we pick the third digit, c_2 is one, so we pick the first digit available.

When using "linear combination," they also began to write out the formula $c_1 4! + c_2 3! + c_3 2! + c_4 1!$, where the c_i s represent the numbers found using the narrowing method (Fig. 6). By doing so, we interpret that the students were connecting the list of outcomes to the listing process used in the odometer strategy. We also note that Zach's suggestion to write a linear combination of factorials was not explicitly prompted by the interviewer, but rather he seemed to formulate a connection between their work and his own prior knowledge of and experience with linear combinations. After writing down this preliminary formula, Zach and Levi proceeded to find a

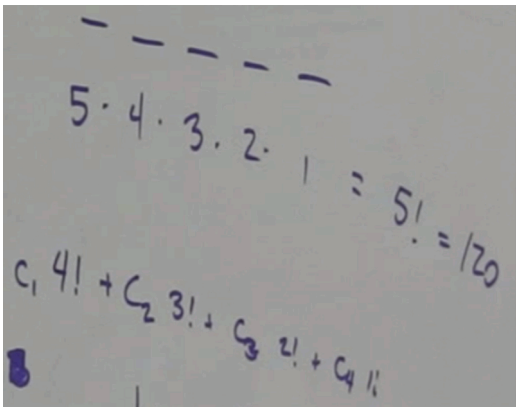

Fig. 6. Zach and Levi's initial characterization of the factorial representation solution method.

way to use the listing process in such a way that they could write 80 as a sum of multiples of factorials so that the multipliers of the factorials encoded the necessary information to find the 80th outcome in the list.

After reasoning that the narrowing process could be encoded into sums of multiples of factorials, the students started to reason about how to express 80 as some sum $c_4 4! + c_3 3! + c_2 2! + c_1 1! + c_0 0!$. The students did not appear to be aware that representing 80 would encode the information for the 81st outcome, which we take as evidence that the students were creating the solution process in the moment, and not relying on prior knowledge of the solution method. Further, by letting $c_0 \neq 0$, the students were able to represent 80 as both $3 \cdot 4! + 1 \cdot 3! + 1 \cdot 2! + 0 \cdot 1! + 0 \cdot 0!$ and $3 \cdot 4! + 1 \cdot 3! + 0 \cdot 2! + 1 \cdot 1! + 1 \cdot 0!$. We infer that the students would not have let $c_0 = 1$ if they were already comfortable using factorial representations, so this suggests that they were still coming to understand the idea of factorial representations and how they might encode the information to find the 80th outcome. However, they eventually used their knowledge of the list of outcomes to reason that $c_0 = 0$, as seen in the following discussion.

By "I have no other choice," we interpret that Zach was referring to only having one option for the last letter after the first four letters of the arrangement have been fixed. Hence, this is an example of reasoning about mathematical expressions using the list of outcomes, where their choice of coefficients c_i encoded information about the list. In addition, they were reasoning in a sophisticated way about nuanced details when expressing numbers in a factorial representation. By asking to find the 80th outcome of a problem where repetition does not occur, the students encountered a novel situation where they could construct a positional representation system to encode the necessary information, but where the representation system was not a base-number system (e.g., binary or decimal). By encountering an unfamiliar representation system, the students were required to think about details that are often overlooked with base-number representation systems.

The students continued to discuss the use of 0! in the mathematical expression until they told the interviewer, "We stumped." The interviewer then intervened by saying that the 0! was unnecessary, and she asked them for the least and greatest numbers they could represent using their representations. This comment from the interviewer was intended to direct the students to see that writing 79 rather than 80 in a factorial representation encodes the information they were looking for. After finding that the least number was 0 and the greatest number was 119, the students became confident that rather than trying to represent 80 in a factorial representation, they should represent 79. We interpret that the students became aware that the 120 numbers represented in the factorial representation were 0 through 119 rather than 1 through 120, and so the 80th outcome in a list with a starting index of 0 is the outcome corresponding to 79. After finding the representation for 79, they confirmed that the process encoded in the representation for 79 corresponded to the original arrangement they had found (with the help of the generated list of outcomes).

Zach and Levi continued to use factorial representations of integers to allow them to compute the mth outcome of later problems involving arrangement without repetition. In doing so, the students encoded the narrowing process into a mathematical expression that could be more easily computed, where the mth element could be read off from the expression. Neither student was previously aware that integers could be represented using factorial representations, so their development and use of factorial representations demonstrates that the students were thinking critically about their solution techniques.

As another example of Zach and Levi's work, on a homework assignment we asked the students to find the 2452nd outcome from the list of arrangements of the letters A, B, C, D, E, F, and G, if the arrangements were listed in lexicographic order. They applied their solution method of calculating the factorial representation of 2,451 in order to find the corresponding arrangement of the letters. During class, Zach and Levi presented their solution to this problem to other students in the class. Fig. 7 shows the students' dialog together with a representation of what they wrote on the board as they spoke; their work demonstrates how the students were able to codify the narrowing process into a mathematical expression from which they could read off the answer without further reference to the list of outcomes. It is important to point out that because the factorial representation method was created by using a connection between the list of outcomes and the numerical solution, we do not believe the factorial representation method lessens the students'

Spoken Dialogue:		Written on White Board:
Zach:	So for, two--what is it?	
Levi:	Twenty four fifty one.	
Zach:	Twenty four fifty one, how many six factorials can we fit into that?	
Levi:	Three.	2,451 = 3·6!
Zach:	And how much is left over?	
Levi:	Umm… some amount [laughs] … 291 is left over.	
Zach:	291.	
Levi:	Which two 120s can fit in there.	
Zach:	Right, so plus 2 times 5 factorial.	2,451 = 3·6! + 2·5!
Levi:	Minus 240, so now you have 51 left over.	
Zach:	You can fit two--	
Levi:	Two 24s. Minus that 48.	2,451 = 3·6! + 2·5! + 2·4!
	And you can fit zero 3 factorials in there.	2,451 = 3·6! + 2·5! + 2·4! + 0·3!
Zach:	Plus 1 times 2 factorial?	
Levi:	Plus 1 times 2 factorial plus 1 times 1 factorial.	2,451 = 3·6! + 2·5! + 2·4! + 0·3! + 1·2! + 1·1!

Fig. 7. Zach and Levi's use of their factorial representation method.

use of the relationship between the two. That is, we do not believe that using the factorial representation method is in conflict with our belief that focusing on the relationship between a list of outcomes and the counting process used to count the outcomes enriches students' understanding of counting problems. In fact, being able to develop this factorial representation method suggests that they had developed a rich understanding of how their listing process generated and organized outcomes.

After writing $2,541 = 3{\bullet}6! + 2{\bullet}5! + 2{\bullet}4! + 0{\bullet}3! + 1{\bullet}2! + 1{\bullet}1!$, the students read off the answer as DCEAFGB, where the D corresponds to the 3 in $3{\bullet}6!$ (counting from 0), the C corresponds to the 2 in $2{\bullet}5!$, and the remaining letters are found in similar ways. We interpret, then, that the students were able to solve additional problems by using this same technique they developed using the factorial representation approach.

Although we present the factorial representation method as separate from the narrowing process, we can also interpret it as an extension of the narrowing process. This extension entailed codifying the choices made in the narrowing process as coefficients in a mathematical expression, thereby creating a solution that could be computed without the need to reason directly about the outcomes. The mathematical expressions used in this solution method do not represent the cardinality of a set of outcomes (as discussed in Lockwood (2013) model), but they do encode information about a list of outcomes. And while the students first reasoned about lists of outcomes to create and articulate the factorial representation method, after they were comfortable with the solution method, they used the mathematical expressions to reason about the list of outcomes. Again, this points to an important aspect of Lockwood's set-oriented perspective, which is the bidirectionality of reasoning between sets of outcomes and mathematical expressions.

In this section, we have demonstrated how students leveraged relationships between counting processes and lists of outcomes and connected them to the factorial representation system, initially using the same reasoning (drawing on a narrowing process) as Dennis, Veronica, and Jason. They then developed a more general solution technique, which involved working with factorial representations (which they had not seen before). To work with these representations, the students explicitly focused on regularity within the lists of outcomes, and they leveraged such regularity to articulate a general relationship that they could use on subsequent problems. They also engaged in some formal symbolization as they expressed their factorial representation. This relatively simple and straightforward task thus fostered rich mathematical reasoning and desirable mathematical practices for students with diverse counting experiences.

6. Discussion, limitations, and avenues for future research

6.1. Discussion

Problems of the form "what is the *m*th element in your list of outcomes?" give opportunities for students to make direct links between their counting processes, numerical expressions, and lists of outcomes, as their solutions depend on the ordering of the outcomes. With these problems, students must reason about the organization of the list of outcomes and use a set-oriented perspective because the problems ask about the nature of the outcomes instead of just the size of the set of outcomes. Attending to the structure in the list of outcomes may help students better understand a counting process or numerical expression; or, reflexively, a counting process or numerical expression can inform the structure of the list of outcomes. For example, precisely describing the structure of each partition of outcomes in the PHONE problem can provide intuition as to *why* using the multiplication principle is appropriate (specifically, why the number of possible letters in each subsequent position decreases by one, and why the number of possible letters in each position is a term in a product). Alternatively, the multiplicative counting process used in the PHONE problem may provide intuition as to why the set of outcomes can be listed in a particular way. By focusing on the structure in the list of outcomes (e.g., by applying the "narrowing process"), the students were attending to the same structure that informs the counting process. We observed this behavior in the students in our data, whose solutions relied on partitioning (and partitioning again) the set of outcomes into parts with similar structure. This process helped students clarify the use of the multiplication principle in the examples we have provided, and it also reinforced the implications of having no repeated characters in problems involving arrangement without repetition (e.g., the PHONE problem). In this way, *m*th element tasks served to reinforce the relationship between counting processes and lists of outcomes.

Prior research had specifically called for additional ways to help students make connections among multiple combinatorial representations (Braithwaite & Goldstone, 2013) and among these components of Lockwood (2013) specifically. In particular, Lockwood et al. (2015) conclude by saying, "Our study suggests that we need to continue to investigate more deeply the ways in which students connect their counting processes with their sets of outcomes, and we feel that the multiplication principle is a key aspect of this connection" (p. 60). In our work on the *m*th element task, we have more deeply investigated ways in which students might make such connections, and we identified the importance of having students understand listing processes in generating outcomes (including understanding operations like the multiplication principle). Our work thus sheds light on the importance of not only having students engage with sets of outcomes, but also in understanding what processes generate and organize those outcomes. We have found simple-to-pose tasks that effectively helped our students connect these components of the model, and we did so within a novel computational setting.

We also note that formulas and expressions played a different role in *m*th element problems than other combinatorics problems, where finding a numerical expression is typically the goal of the problem. In *m*th element problems, there is a greater emphasis on using a mathematical expression to reason about a list of outcomes. This was exemplified in the ROCKET problem, where Dennis, Jason, and Veronica used representations such as $6 \cdot 5!$ and $6 \cdot 5 \cdot 4!$ to articulate how their list was structured. Further, Levi's and Zach's use of the factorial representation method demonstrated how they encoded the structure of the list of outcomes in a mathematical expression, and then used the factorial representations to reason about individual outcomes. This direction of reasoning (using formulas/expressions to reason about lists/sets of outcomes) was less clear with the motivating data examples with Charlotte, Diana, and

Charlie. We value this direction of reasoning in a set-oriented perspective because it can allow for a greater understanding of the list/sets of outcomes, and allow for connections among different counting problems. A possible takeaway from these data is the importance of bidirectionality of reasoning between formulas/expressions and the other components of Lockwood's model.

Students' work presented here also serves to clarify some productive aspects of a set-oriented perspective (Lockwood, 2014), as well as to illustrate that students may need help to use a set-oriented perspective in a way that positively reinforces their mathematical understanding. In Section 3.3 (Motivating Data Examples), we observed that students can reason correctly about sets of outcomes and counting processes individually, but it was not trivial for them to connect the two correctly and thoroughly. Because one of the intended purposes of a set-oriented perspective is to reinforce the relationship between sets of outcomes and counting processes so that students have a better conceptual understanding of why a solution is correct, it is imperative to investigate when and why there is a gap in understanding the relationship. By providing students with tasks that sought to connect sets or lists of outcomes to counting processes, we observed that students were able to reflexively reason between the two. One of the productive aspects of a set-oriented perspective in these tasks, then, was the change from an articulation of the global structure of the list (e.g. a broad statement that the list was partitioned by letters in each position) to an analytical and precise articulation of each step. In doing so, the students seemed to view the objects being counted as useful to reason about in their own right, as opposed to only being used as a means to arrive at the cardinality of the set.

The students' reasoning demonstrated in these data might also signify the importance of asking questions not strictly related to cardinality. Here, the *m*th element questions asked about the structure in a list of outcomes, and while the structure was informed by a mathematical expression the goal of the task was not to count the cardinality of the list. Other questions that do not directly target cardinality might also illustrate productive ways of using a set-oriented perspective. For example, Lockwood and De Chenne (2021) discuss encoding outcomes in a counting problem. By asking students to reason about how outcomes of a counting problem might be encoded (or inscribed), the students might develop an understanding about the outcomes that can inform how to find the cardinality of the set without the goal being to find the cardinality. Doing so may also deflect the view that solving counting problems is a matter of identifying and applying a correct formula or algorithm, and encourage the students to find their own connections between counting problems by connecting to the sets of outcomes.

Student work presented here also demonstrated valuable ways of reasoning mathematically, some of which were not anticipated by the authors. In developing the narrowing process, the students made use of structure by connecting how lists of outcomes were organized to specific terms in a mathematical expression. In developing the factorial representation solution, the students codified the narrowing process so that terms in a sum represented choices made at different steps in the narrowing process. Although both authors were aware of factorial representations, neither had thought to apply them to *m*th element problems. The factorial representation solution shows reasoning about structure, as well as representing the solution process as enumerated choices at each step, where the numbered choices correspond to a mathematical expression to find a specific element in a list.

Previous studies have examined how computer programming can be used in combinatorics education (e.g., Lockwood & De Chenne, 2020, 2021), and these studies examined how students engage with the computer as a way to reinforce combinatorial understanding. We should not de-emphasize the role of the computer in this study. The students had written code in Python to create lists of outcomes, and they had used these lists to reason about counting problems. However, these data are examples of students reasoning about artifacts created by a computer program they wrote, rather than reasoning directly about computer code. In this study, the role of the computer was to create objects and representations that the students studied and analyzed, thereby allowing for additional mathematical reasoning. For example, the lists of outcomes presented in this study often consisted of hundreds or thousands of outcomes, and students can have difficulty when reasoning about lists with such large cardinalities. Subtle errors are common in such problems (e.g., Eizenberg & Zaslavsky, 2004), and this study may contribute one way to aid in students detecting these errors. Although De Chenne and Lockwood (2020) have articulated some ways that a computational environment can be used to detect these errors, *m*th element problems might assist students in detecting errors that relate to issues of organization of the list of outcomes.

We also note again that counters with various levels of coding and counting experience were able to engage with these tasks, and solving the tasks may make a solution process seem more concrete, in the sense that the students could conceive of specific, rather than more general, outcomes of a process. While the general outcome might be helpful to some students, using and reasoning about specific outcomes may help students who find it difficult to differentiate between problem types. For example, in the PHONE problem, working with specific outcomes (such as NHPEO) allows more opportunities for students to make sense of why letters are not repeated than does working with general outcome. Some students with more experience solving counting problems, on the other hand, may be successful in reasoning about general outcomes, and they may not need to refer to specific outcomes to correctly solve combinatorics problems. Despite their ability to correctly apply formulas and/or expressions when solving counting problems, their intuition regarding the set of outcomes may be less developed. Thus, these questions may have the capacity to deepen the students' understanding of the problems.

6.2. *Limitations of the study*

While we are enthusiastic about the ways in which the *m*th element tasks fostered connections for students, we also acknowledge limitations of our work presented here. One limitation of the study is that we share results from only a handful of students (a group of 3 and a pair). We do not intend to extrapolate too widely from this small set of students, nor do we claim that our results will necessarily hold for students more broadly. Our goal instead is to claim that we have demonstrated evidence that the *m*th element tasks we asked seemed to effectively support students in connecting specific counting processes to lists of outcomes. We can think of our results as offering an existence proof of the kinds of reasoning these tasks elicited, which were useful and productive for students in multiple

ways. As we have noted, the fact that we have students with a variety of mathematical and computational experience does mitigate this limitation some, but we still want to be very clear about what we are trying to claim.

In addition, the student work presented here comes from the context of using a computer (and specifically using CoCalc to read and write Python code using a Jupyter notebook) to explore problems in combinatorics. The students had each taken at least an introductory computer science course, and so they were familiar with the basic commands used in the code. When finding the *m*th element from a set of outcomes, it was always after the students had written a program to enumerate the outcomes, or they had written code to enumerate the outcomes of a similar problem. This may be a drawback of our study, as many courses teaching elementary combinatorics do not use computing. However, we argue that these types of problems (ones that ask about a set or list of outcomes without the goal being to find the cardinality or list) might benefit students who do not also engage in computing. That is, as we discussed in Section 4.3.3, students could still reason about such problems without using a computer or the representation of code. They could imagine or partially write a list of outcomes and make an argument about what the *m*th outcome might be, or such lists could be provided to them.

6.3. Avenues for future research

There are several opportunities for ways to extend and build upon the findings presented in this paper. First, researchers could investigate whether our findings do hold among larger and broader populations of students. This might entail conducting teaching experiments with students with different background and preparations, or implementing the *m*th element tasks in whole-class settings and seeing what kind of reasoning is elicited for students. Such work would let us test the effectiveness of such tasks and to allow for stronger claims about generalizability.

The factorial representation solution method can be viewed as an extension or encoding of the narrowing process. And while Levi and Zach came up with the solution method themselves, it is reasonable to believe that scaffolding from the narrowing process to the factorial representation method can be used in future research. For example, asking students to reason about bijections between fixed base number systems and lists of outcomes for Cartesian product problems might be used to introduce the concept of using number systems to find *m*th elements. In problems such as the PHONE problem, one might ask students to determine an expression using 4! for the first outcome starting with N, or determine an expression using 4! and 3! for the first outcome starting with NE. Connections like these that explicitly leverage a progression of ideas from the narrowing process to the factorial representation could be a potential starting point for investigating and developing learning trajectories in combinatorics. Such work would align with existing efforts (such as those described in Maher et al., 2011 and Tillema, 2020) to describe longitudinal trajectories through which students could develop and progress.

In addition, while we suspect that students could reason about and leverage non-computational versions of these tasks (that is, when a computer and code is not readily available as a tool for students), this could be investigated empirically. Researchers could pose these kinds of in strictly by-hand setting and see if the tasks can still effectively help students connect their listing processes with the set of outcomes the process generates.

Finally, ultimately, we have tried to elaborate aspects of a set-oriented way of thinking that facilitate connections between listing processes and outcomes. While Lockwood (2014) provided a broad construct of a perspective toward counting, there is more room to investigate more specific features and aspects of that way of thinking that might be beneficial for students. Future research could explore this way of thinking in more detail.

Author statement

Adaline De Chenne: Data collection and analysis, Conceptualization, Writing **Elise Lockwood:** Data collection and analysis, Conceptualization, Writing.

Data availability

Raw data and other supplementary material are available upon request.osf.io/7zpcj

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1650943. The idea for these tasks was inspired by a talk by Sue Doree's at the PNW MAA Sectional Meeting in 2019.

References

Annin, S. A., & Lai, K. S. (2010). Common errors in counting problems. *Mathematics Teacher, 103*(6), 402–409.

Batanero, C., Navarro-Pelayo, V., & Godino, J. (1997). Effect of the implicit combinatorial model on combinatorial reasoning in secondary school pupils. *Educational Studies in Mathematics, 32*, 181–199.

Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of Child-Computer Interaction, 16*, 68–76.

Braithwaite, D. W., & Goldstone, R. L. (2013). Integrating formal and grounded representations in combinatorics learning. *Journal of Educational Psychology, 105*(3), 666.

Caballero, M. D., Chonacky, N., Engelhardt, L., Hilborn, R. C., Lopez del Puerto, M., & Roos, K. R. (2019). Picup: A community of teachers integrating computation into undergraduate physics courses. *The Physics Teacher, 57*(6), 397–399.
CadwalladerOlsker, T., Engelke, N., Annin, S., & Henning, A. (2012). Does a statement of whether order matters in counting problems affect students' strategies?. In *Electronic Proceedings of the 15th Annual Meeting of the Research on Undergraduate Mathematics Education*.
De Chenne, A., & Lockwood, E. (2020). Student verification practices for combinatorics problems in a computational environment. *Electronic Proceedings for the Twenty-Third Special Interest Group of the MAA on Research on Undergraduate Mathematics Education*, 96–103.
Eizenberg, M. M., & Zaslavsky, O. (2004). Students' verification strategies for combinatorial problems. *Mathematical Thinking and Learning, 6*(1), 15–36.
English, L. D. (1991). Young children's combinatorics strategies. *Educational Studies in Mathematics, 22*, 451-47.
English, L. D. (1993). Children's strategies for solving two- and three-dimensional combinatorial problems. *The Journal of Mathematical Behavior, 24*(3), 255–273.
Farris, A. V., Dikes, A. C., & Sengupta, P. (2020). Grounding computational abstractions in scientific experience. *Proceedings of the 13th International Conference of the Learning Sciences (ICLS 2020)*, 1333–1340.
Feurzeig, W., Papert, S., & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487–501.
Hadar, N., & Hadass, R. (1981). The road to solving a combinatorial problem is strewn with pitfalls. *Educational Studies in Mathematics, 12*(4), 435–443.
Halani, A. (2012). Students' ways of thinking about enumerative combinatorics solution sets: The odometer category. In *Electronic Proceedings for the Fifteenth Special Interest Group of the MAA on Research on Undergraduate Mathematics Education* (pp. 59–68).
Kapur, J. N. (1970). Combinatorial analysis and school mathematics. *Educational Studies in Mathematics, 3*(1), 111–127.
Kavousian, S. (2008). *Enquiries into undergraduate students' understanding of combinatorial structures*. Unpublished doctoral dissertation. Simon Fraser University.
D. Knuth . The Art of Computer Programming 3rd 2 Addison-Wesley 1997; 65-66 ISBN 0-201-89684-2.
Kotsopoulos, D., Floyd, L., Khan, S., Kizito Namukasa, I., Somanath, S., Weber, J., et al. (2017). A pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education, 3*, 154–171. https://doi.org/10.1007/s40751-017-0031-2
Lockwood, E. (2013). A model of students' combinatorial thinking. *The Journal of Mathematical Behavior, 32*, 251–265. https://doi.org/10.1016/j.jmathb.2013.02.008
Lockwood, E. (2014). A set-oriented perspective on solving counting problems. *For the Learning of Mathematics, 34*(2), 31–37.
Lockwood, E. (2021). Leveraging prediction and reflection in a computational setting to enrich undergraduate students' combinatorial thinking. *Cognition and Instruction*. In press.
Lockwood, E., & De Chenne, A. (2020). Using conditional statements in Python to reason about sets of outcomes in combinatorial problems. *International Journal of Research in Undergraduate Mathematics Education, 6*, 303–346. https://doi.org/10.1007/s40753-019-00108-2
Lockwood, E., & Gibson, B. (2016). Combinatorial tasks and outcome listing: Examining productive listing among undergraduate students. *Educational Studies in Mathematics, 91*(2), 247–270. https://doi.org/10.1007/s10649-015-9664-5
Lockwood, E., & Purdy, B. (2019). Two undergraduate students' reinvention of the multiplication principle. *Journal for Research in Mathematics Education, 50*(3), 225–267.
Lockwood, E., Reed, Z., & Caughman, J. S. (2017). An analysis of statements of the multiplication principle in combinatorics, discrete, and finite mathematics textbooks. *International Journal of Research in Undergraduate Mathematics Education, 3*(3), 381–416. https://doi.org/10.1007/s40753-016-0045-y
Lockwood, E., Reed, Z., & Erickson, S. (2021). Undergraduate students' combinatorial proof of binomial identities. *Journal for Research in Mathematics Education, 52*(5), 539–580. https://doi.org/10.5951/jresematheduc-2021-01.
Lockwood, E., Swinyard, C. A., & Caughman, J. S. (2015). Patterns, sets of outcomes, and combinatorial justification: Two students' reinvention of counting formulas. *International Journal of Research in Undergraduate Mathematics Education, 1*(1), 27–62. https://doi.org/10.1007/s40753-015-0001-2
Maher, C. A., Powell, A. B., & Uptegrove, E. B. (Eds.). (2011). *Combinatorics and reasoning: Representing, justifying, and building isomorphisms*. New York: Springer.
Medova, J., & Ceretkova, S. (2021). Relation between algorithmic and combinatorial thinking of undergraduate students of applied informatics. *Paper Presented at the 14th International Congress on Mathematics Education*. remote presentation.
Mota, B., & Ferreira, R. A. T. (2021). Guiding students' reinvention of combinatorial operations. *Paper Presented at the 14th International Congress on Mathematics Education*. remote presentation.
Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*, 351–380. https://doi.org/10.1007/s10639-012-9240-x
Reed, Z., & Lockwood, E. (2021). Leveraging a categorization activity to facilitate productive generalizing activity and combinatorial reasoning. Online first *Cognition and Instruction* https://doi.org/10.1080/07370008.2021.1887192.
Sengupta, P., Dickes, A., & Farris, V. (2018). Toward a phenomenology of computational thinking in K-12 STEM. In M. Khine (Ed.), *Computational thinking in the STEM disciplines*. Cham: Springer. https://doi.org/10.1007/978-3-319-93566-9_4.
Sinclair, N., & Patterson, M. (2018). The dynamic geometrisation of computer programming. *Mathematical Thinking and Learning, 20*(1), 54–74. https://doi.org/10.1080/1096065.2018.1403541
Steffe, L. P., & Thompson, P. W. (2000). Teaching experiment methodology: Underlying principles and essential elements. In R. Lesh, & A. E. Kelly (Eds.), *Research design in mathematics and science education* (pp. 267–307). Mahwah, NJ: Lawrence Erlbaum Associates.
Tillema, E. S. (2013). A power meaning of multiplication: Three eighth graders' solutions of Cartesian product problems. *The Journal of Mathematical Behavior, 32*, 331–352.
Tillema, E. S. (2018). An investigation of 6th graders' solutions of Cartesian product problems and representation of these problems using arrays. *The Journal of Mathematical Behavior, 52*, 1–20.
Tillema, E. S., & Gatza, A. (2016). A quantitative and combinatorial approach to non-linear meanings of multiplication. *For the Learning of Mathematics, 36*(2), 26–33.
Tucker, A. (2002). *Applied combinatorics* (4th ed.). New York: John Wiley & Sons.
Vieira, C., Magana, A. J., Roy, & Falk, M. L. (2019). Student explanations in the context of computational science and engineering education. *Cognition and Instruction, 37*(2), 201–231. https://doi.org/10.1080/07370008.2018.1539739
Wagh, A., Levy, S., Horn, M., Guo, Y., Brady, C., & Wilensky, U. (2017). Anchor code: Modularity as evidence for conceptual learning and computational practices of students using a code-first environment. *CSCL Proceedings*, 656–659.
Wasserman, N., & Galarza, P. (2019). Conceptualizing and justifying sets of outcomes with combination problems. *Investigations in Mathematics Learning, 11*(2), 83–102.
Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., et al. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*, 127–147. https://doi.org/10.1007/s10956-015-0581-5