”Mmm... Donuts!” Using Real-World Scenarios in a First-Year Programming Course

Dr. John K. Estell, Ohio Northern University

An active member of ASEE for over 25 years, Dr. John K. Estell was elected in 2016 as a Fellow of ASEE in recognition of the breadth, richness, and quality of his contributions to the betterment of engineering education. Estell currently serves on the ASEE Board of Directors as the Vice President of Professional Interest Councils and as the Chair of Professional Interest Council III. He has held multiple ASEE leadership positions within the First-Year Programs (FPD) and Computers in Education (CoED) divisions, and with the Ad Hoc Committee on Interdivisional Cooperation, Interdivisional Town Hall Planning Committee, ASEE Active, and the Committee on Diversity, Equity, and Inclusion. Estell has received multiple ASEE Annual Conference Best Paper awards from the Computers in Education, First-Year Programs, and Design in Engineering Education Divisions. He has also been recognized by ASEE as the recipient of the 2005 Merl K. Miller Award and by the Kern Entrepreneurial Engineering Network (KEEN) with the 2018 ASEE Best Card Award. Estell received the First-Year Programs Division’s Distinguished Service Award in 2019.

Estell currently serves as an ABET Commissioner and as a member on ABET’s Accreditation Council Training Committee. He was previously a Member-At-Large on the Computing Accreditation Commission Executive Committee and a Program Evaluator for both computer engineering and computer science. Estell is well-known for his significant contributions on streamlining student outcomes assessment processes and has been an invited presenter at the ABET Symposium on multiple occasions. He was named an ABET Fellow in 2021. Estell is also a founding member and current Vice President of The Pledge of the Computing Professional, an organization dedicated to the promotion of ethics in the computing professions.

Estell is Professor of Computer Engineering and Computer Science at Ohio Northern University, where he currently teaches first-year programming and user interface design courses, and serves on the college’s Capstone Design Committee. Much of his research involves design education pedagogy, including formative assessment of client-student interactions, modeling sources of engineering design constraints, and applying the entrepreneurial mindset to first-year programming projects through student engagement in educational software development. Estell earned his BS in Computer Science and Engineering degree from The University of Toledo and both his MS and PhD degrees in computer science from the University of Illinois at Urbana-Champaign.

Dr. Stephany Coffman-Wolph, Ohio Northern University

Dr. Stephany Coffman-Wolph is an Assistant Professor at Ohio Northern University in the Department of Electrical, Computer Engineering, and Computer Science (ECCS). Research interests include: Artificial Intelligence, Fuzzy Logic, Game Theory, Teaching Computer Science to First-Year, K-12 Outreach, and Increasing Diversity in STEM.

Ian Meyer Kropp

"Mmm… Donuts!" Motivating CS1 Students through a Real-World Programming Scenario

Abstract

This complete, evidence-based practice paper describes experiential work involving the application of the entrepreneurial mindset in the first computer programming course (CS1). Teaching CS1 has always been challenging, with first-year students trying to learn problem-solving design methodologies expressed in a "foreign" language. When materials are presented abstractly - *i.e.*, emphasizing computing for its own sake - the result is often a disconnect: our students cannot envision how the concepts being taught can be meaningfully applied to real-world situations. Consequently, it is important for CS1 instructors to provide appropriate context allowing students to make connections between their coursework and providing value to others through their coding. The approach presented here involves creating programming assignments featuring real-world scenarios – something that could be encountered in an everyday activity. To illustrate this approach, this paper will frame a typical CS1 problem – calculating the price of a business transaction and subsequently accepting payment and providing change to the customer – through the familiar scenario of buying donuts at a local donut shop. Students are provided with such artifacts as the donut shop's menu, government publications for calculating sales tax, and donut shop photos. Students are primed for success through preliminary laboratory assignments separately focusing on the professional responsibilities for calculating sales tax, making change, and formatting monetary output while emphasizing the importance of breaking problems down into their components. This approach has successfully been used as our first "major" CS1 programming assignment, as demonstrated by both quantitative and qualitative post-activity survey data. Interested readers are encouraged to download all materials associated with this assignment via the provided Engineering Unleashed resource link.

1. Introduction

How to teach various aspects of the introductory programming course - commonly referred to as "CS1" - has been the subject of many papers for over 50 years [1]. While certain aspects have evolved over time, such as advances in programming languages and software development tools, other aspects have remained the same, most notably the difficulty experienced by many in teaching their students problem solving and design skills [2]. Course assignments play a significant role in the CS1 student's experience, to the extent that it's viewed as being integral to the success of the course, in that an assignment must be accessible, engaging, challenging only with respect to the material it is intended to teach, and be worthy of the time and effort invested [3]. Additionally, a 2001 study on factors contributing to the success of students in introductory computer science courses concluded that the best predictor of success was the students' comfort level and recommended that instructors should provide a welcoming environment [4]. A 2017 literature review on students' misconceptions in introductory programming provides an additional perspective: that an ability to apply effective instructional approaches and tools in addressing students' difficulties is vital to successfully teaching CS1 [5].

So how can instructors create assignments that are effective in teaching pertinent concepts that are also accessible, engaging, and welcoming? While a full list is beyond the scope of this paper, we offer one format that has been successfully used in our CS1 course: real-world problems that are interesting and relatable. By grounding assignments in reality, instructors can place the assignment's emphasis on creating value for a group of people by writing a program that solves one or more problems being experienced. This emphasis on the social utility of programming through using real problems and data helps drive student interest while steering away from an overemphasis of the technical aspects of programming - *i.e.*, emphasizing programming for its own sake [6]. However, it must be kept in mind that these are first-year students, not practicing computing professionals, so appropriate assignments and accommodations are needed. For appropriateness, the posed problem should be one that is within the spectrum of a typical teenager's life experiences and thus relatable; for accommodating, the assignment should be delivered in a way that makes problem solving using technical skills acquired in CS1 the primary focus. Collectively, these various aspects help to motivate students through the perceived relevance of the assignment to career goals and societal needs [7].

2. Course Setting

Programming 1, the CS1 course at Ohio Northern University (ONU), is a four-credit hour C++ programming course that focuses on basic programming concepts (variables, conditionals, loops, pointers etc.), early data structures, debugging, and documenting software. The course meets for three 50-minute lecture sessions and a 165-minute computer lab session. Programming 1 is a required course for students in a wide range of majors including: computer science, computer engineering, electrical engineering, data analytics, mathematics, and physics. The student population also includes those pursuing a CS minor from disciplines such as mechanical engineering, manufacturing technology, statistics, and finance. As expected with a course housed within a college of engineering, the majority of the students are first-year engineering students. Furthermore, as computer science is considered by ABET as an engineering topic [8], emphasis is placed on instructing students regarding appropriate engineering design methodologies, such as problem clarification, application of relevant constraints and criteria, scoping, testing, validating, and iterative improvements. As ONU is a member of the Kern Entrepreneurial Engineering Network (KEEN), elements of entrepreneurial mindset (EM) represented in terms of curiosity, connections, and creating value [9] are also embedded within the course content, as part of a larger goal of instilling EM across all curricula within our College of Engineering [10].

3. Literature Review: Word Problems

Word problems are verbal descriptions of problem situations wherein one or more questions are raised, the answer to which can be obtained by the application of mathematical operations to numerical data available in the problem statement [11]. Generally, they are not liked by students - often, they're hated. An article by Nancy Knop contained within [12] states that part of the reason why is that the brain pathways for processing language symbols and for processing numeric symbols do not completely overlap, thus requiring translations between these symbolic languages, thereby making the math more difficult. Additional research supports the contention that various linguistic verbal components not related to arithmetic contribute to this difficulty [13]. Another

reason is that the student is no longer encountering problems akin to "being fed with a baby spoon" - that is, tasks that are "easy" as they have a well-rehearsed solution pathway that can be applied to produce an answer. Similarly, the employment of superficial solution strategies has been found to be a primary reason for students providing solutions that are inconsistent with the situations described in the word problems [14]. Part of this can be attributed to what has been described as the "suspension of sense-making" phenomenon where students ignore relevant aspects of reality when answering such questions [15], such as stating that 2½ trips are needed to transport 50 people on a 20-seat bus. Finally, students' early exposures to word problems tend to cause dislike as either they had an inability to solve them or did not find such problems personally relevant [16].

So why use word problems if they are so reviled? First, the story aspect of word problems can elicit different, more effective solution strategies, especially when situational models are involved [17]. Second, multiple researchers have raised as an issue the lack of realism in word problems, potentially having a negative impact on the students' learning, attitudes, and beliefs; however, if properly contextualized, word problems can serve as simulations of real-world situations [18]. To accomplish this, those crafting word problems need to be aware of the impact of authenticity on sense making when describing the out-of-school "task context" that represents some task situation in the real life of the target audience [14]. Establishing appropriate context is important; for example, discussing the application of the Luhn Algorithm for detecting single-digit errors with credit card data entry will not resonate with students too young to be in possession of a credit card. Additional motivation can result when students encounter problems posed in the context of out-of-school interests, as such word problems contain sufficient context personalization to connect to real-world experiences [19]. Taking into consideration real-world experiences when crafting word problems constitutes exercises in modeling, where students are expected to develop an abstract formal structure grounded in aspects of reality. This resultant word problem provides students with a written description of some situation that must be understood using reasonable assumptions prior to its being modeled [15]. To paraphrase [20], computing, like pure mathematics, is semantically empty if treated as a formal system; only when it is applied or related to the real world does its usefulness become revealed. Word problems allow instructors to provide the appropriate social rationality allowing students to make such connections.

So how does an instructor create a word problem that's a compelling real-world scenario? The following attributes have been gathered from [21][22]:

- *Relatable*: the student audience needs to be able to connect with the problem.
- *Solvable*: the problem needs to have at least one solution that is within the scope of the students' knowledge base.
- *Open-ended*: getting away from a simplistic, fill-in-the-blank solution allows students to engage their critical thinking and problem-solving skills.
- *Narrative*: embedding the programming skills to be displayed within a human-oriented context gives the problem more immediacy and relevance.
- *Visual*: adding a visual element makes the problem much more engaging.
- *Challenging*: forcing students to think critically at every step instead of relying on rote memorization encourages engagement.

Of particular note is that of narrative - that is, the art of telling a story. While a traditional mathematical word problem is far removed from the construct of a mathematical story [23], concepts of storytelling can be used to help drive engagement and motivation. While there are many suggested formulas to help structure a story, all that's really needed are two things: character and conflict [24]. The simplest method for addressing character is to cast the reader as protagonist; otherwise, additional steps are needed to create a character that students can relate to. Conflict is developed through establishing what constitutes the problem via, in this case, an accomplishment story [25] where the protagonist is trying to achieve a solvable goal. The resolution is performed via the writing of the program, which is open-ended in that there are multiple approaches to coding a solution, and which is also challenging in that one must consider each step of the program coding process from data input onwards. Storytelling allows for the narrative illustration of technical subjects [26], helping students to mentally visualize abstract concepts.

4. Word Problem Example: Jim's Donut Shop

To illustrate the effectiveness of a CS1-oriented word problem that uses a real-world scenario, let us start with a typical example found in a CS1 textbook [27]:

> *Write a program that simulates a vending machine. A customer selects an item for purchase and inserts a bill into the vending machine. The vending machine dispenses the purchased item and gives change. We will assume that all item prices are multiples of 25 cents, and the machine gives all change in dollar coins and quarters. Your task is to compute how many coins of each type to return.*

In this word problem there is a character, but the reference to the "customer" is provided without any further description. This fails to engage the reader. There's also no sense of conflict in this description, as the reader is simply provided with the operational parameters of a vending machine. While the problem is solvable, as vending machines do exist in the real world, and students do purchase items from vending machines, it is not fully relatable as the inner workings of such machines are, to students, a mystery. So, while it is a word problem, it doesn't really tell a story.

While students cannot relate to being a vending machine, they can relate to working in the food service industry - most college students, for example, either have had a job at a restaurant or knows a classmate who has worked at one. And many have participated in handling financial transactions with a customer, where one calculates the cost of the purchased items, accepts payment, and makes change. So why not repackage this word problem within an appropriate real-world context that makes for a more relatable problem? A scenario involving… donuts.

Mmm, donuts [28].

The donut problem is used as the first major programming assignment - referred to as "Major Problems" or MPs for short - in the CS1 course. A donut shop is used because of the relative

simplicity of the pricing model, compared to the menus found at other restaurants. The word problem for MP1, as provided to the students, begins with the following premise:

> *Assume that you are an employee at Jim's Donut Shop in Vandalia. Your supervisor, upon hearing that you're taking a programming class, asks you to write a program that calculates the cost of a customer's purchase. Fortunately for you, while there are over 40 varieties of donuts available, there are only two pricing categories for donuts: regular and fancy. Those varieties considered as "regular" donuts are priced at 75 cents individually, or you can get a dozen for $7.99. "Fancy" donuts, on the other hand, are priced at 85 cents each, or at $8.49/dozen. Also available at Jim's are their humungous apple fritters, priced at $1.50 each (sorry, no volume discount).*

With this paragraph, the word problem is presented as a story with both character and conflict. The reader is the protagonist, tasked with imagining that they are a donut shop employee. The photo of the donut shop's interior where products are stored and customer interactions occur, presented in Figure 1, helps students with visualizing the scenario. The prices provided are the actual prices on the donut shop's menu board.

Figure 1. Interior of Jim's Donut Shop.

In the next paragraph, the assignment describes the operation of the program. Although this is somewhat contrary to the desire for open-endedness, this is the students' first major programming assignment in CS1, so some handholding is warranted. It should be noted that, at this stage in the course curriculum, students have covered the concepts of sequence and selection, but not iteration, hence the instruction for having the program executed once per customer:

Your program is to execute once per customer, and is to ask for the number of regular donuts, fancy donuts, and apple fritters purchased as shown in the example runs on the next page. For purposes of calculating cost, you will need to determine the number of donuts purchased first in terms of dozens; any donuts left over are then priced individually. The cost must include the sales tax, which for food purchases within Vandalia is 7.5%. After displaying the cost, you must obtain the amount of payment received; you may assume that the payment is either equal to or greater than the cost. From this, you are to calculate the change to be provided to the customer and then display that information as indicated by the provided example runs.

Students are supported in their efforts on this first major programming assignment by the inclusion of example runs from a successfully implemented donut program in the assignment handout (Figure 2), allowing them to verify the correctness of their code via replication:

SAMPLE RUNS (user input shown in ***red bold italics***)

Run #1:
```
Number of regular donuts ordered: 1
Number of fancy donuts ordered: 1
Number of apple fritters ordered: 1
Customer owes $3.33
Customer pays $5.00
Changed owed is $1.67 - 1 dollar, 2 quarters, 1 dime, 1 nickel, 2 pennies.
```

Run #2:
```
Number of regular donuts ordered: 12
Number of fancy donuts ordered: 0
Number of apple fritters ordered: 0
Customer owes $8.59
Customer pays $10.00
Changed owed is $1.41 - 1 dollar, 1 quarter, 1 dime, 1 nickel, 1 penny.
```

Run #3:
```
Number of regular donuts ordered: 28
Number of fancy donuts ordered: 30
Number of apple fritters ordered: 2
Customer owes $47.36
Customer pays $50.00
Changed owed is $2.64 - 2 dollars, 2 quarters, 1 dime, 4 pennies.
```

Figure 2: Sample example runs of donut program.

Additionally, students are presented with the scoring rubric used to evaluate both the readability of their code and their adherence to documentation standards. Laboratory assignments are also used to break the MP down into smaller chunks; for example, one assignment is to implement the sales tax algorithm. Students are provided with the document published online by the Ohio Department of Taxation [29] that is used by professional software developers to implement and verify their sales tax calculations. The inclusion of sales tax calculations in this manner strengthens the realistic aspects of this assignment.

5. Assessment Methodology

In order to test the research premise that a real-world scenario would make for an effective word problem-based programming assignment, students were asked to voluntarily fill out a post-activity survey. The survey and accompanying research proposal was reviewed and given exempt status by the Ohio Northern University Institutional Research Board. A full copy of the survey questions is provided in Appendix A. A total of four demographic, 20 quantitative, and four qualitative questions were posed. For the quantitative questions, a 7-point Likert scale with a rank ordering from "strongly agree" to "strongly disagree" was employed. The survey included questions about the students' experiences and opinions with respect to "story problems" (the term used as it is more familiar to our students than "word problems"), and their views on the degree to which the various tasks within the Jim's Donut Shop assignment related to real-world situations.

In Fall 2022 there were three sections of the course offered with a total enrollment of 78 students; from these students, 66 survey responses (85% response rate) were received. At the time of administering the survey, there were 18 first-year computer engineering and 21 first-year computer science students enrolled; from these two categories, 13 responses (72%) were received from computer engineering majors and 20 responses (95%) were received from computer science majors. As this research focuses on motivating first-year students that intend to pursue a major that contains a significant code writing element, the data analysis presented herein focuses on these 33 respondents out of a study population size of 39 (85%).

6. Assessment Results and Discussion

When asked about story problems, a significant majority of students indicated that they had at least some prior experience in completing story problem assignments (with 97% reporting some level of agreement), as shown in Figure 3.

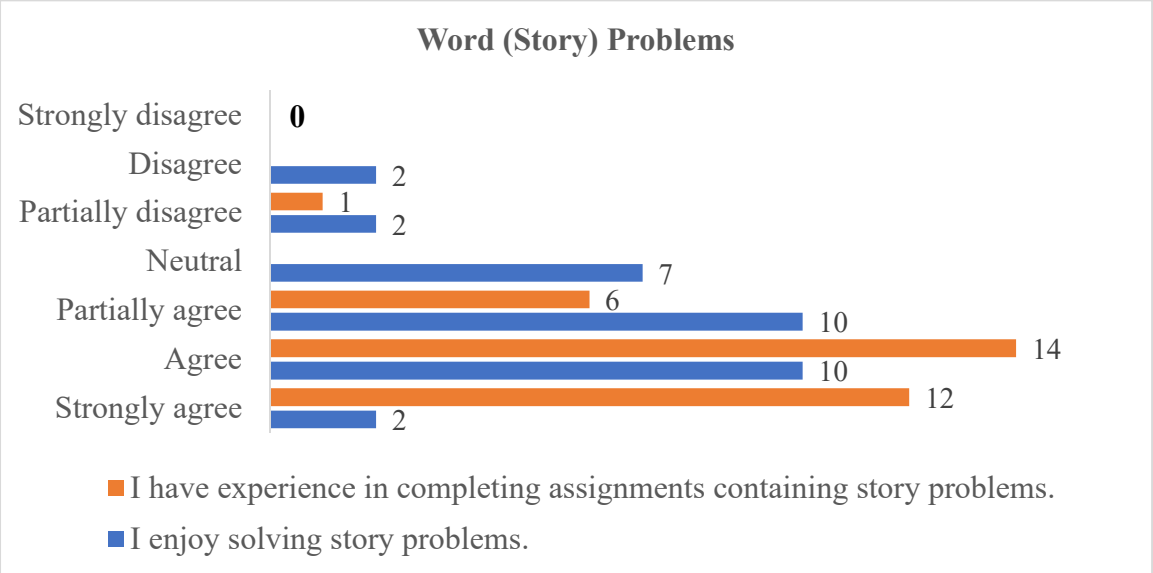

Figure 3. Word (AKA Story) problems.

It is noted, however, that not all students enjoyed solving such problems, with only 67% reporting in the affirmative. A majority still had a positive opinion, more than what might be expected from, say, a survey of all recent high school graduates, probably due to their affinity toward mathematics. Additionally, Figure 4 shows that most students (90%) have come into the CS1 course with experience in completing assignments containing real-world situations.

Figure 4. Experience with real-world assignments.

Figure 5 shows that calculating cost, sales tax, and change within MP1 were all considered as real-world problems.

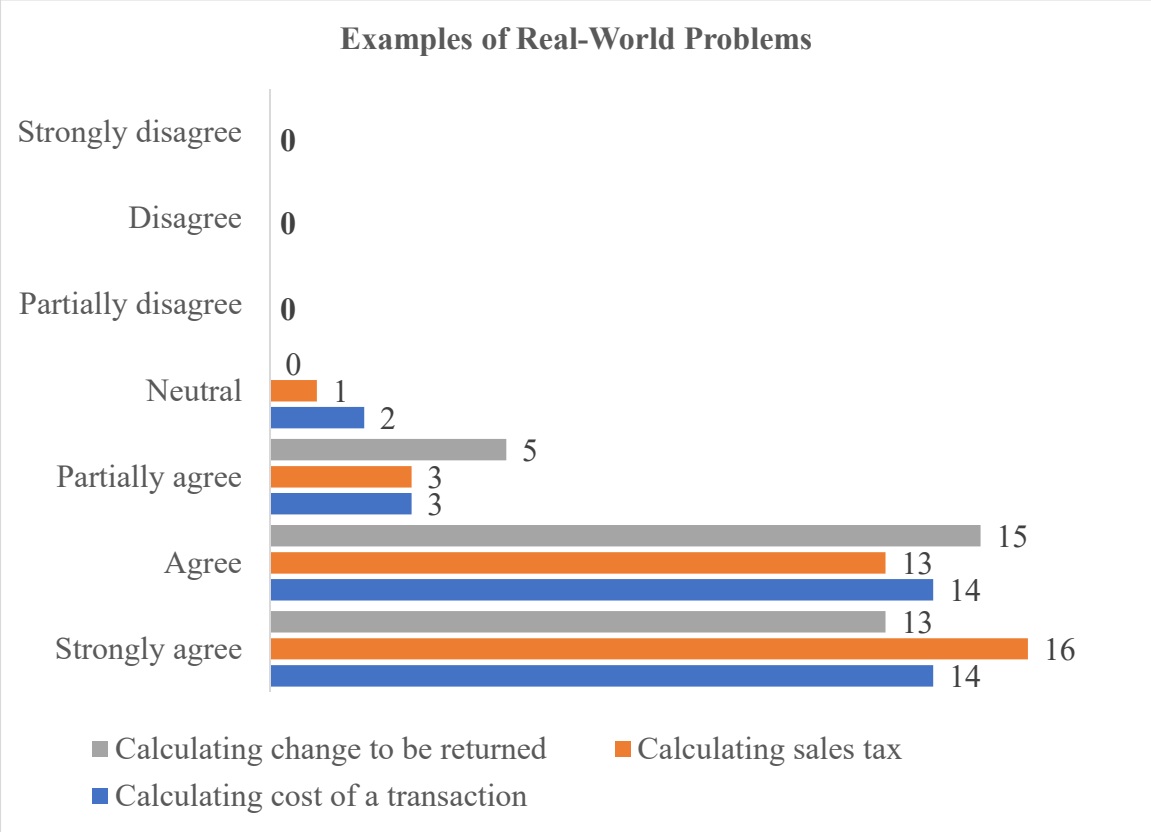

Figure 5. Opinions on examples of real-world problems.

Because this assignment used a government-issued document that specifically addressed how to legally calculate and round sales tax amounts, the survey measured student attitudes towards this component of the MP (Figure 6).

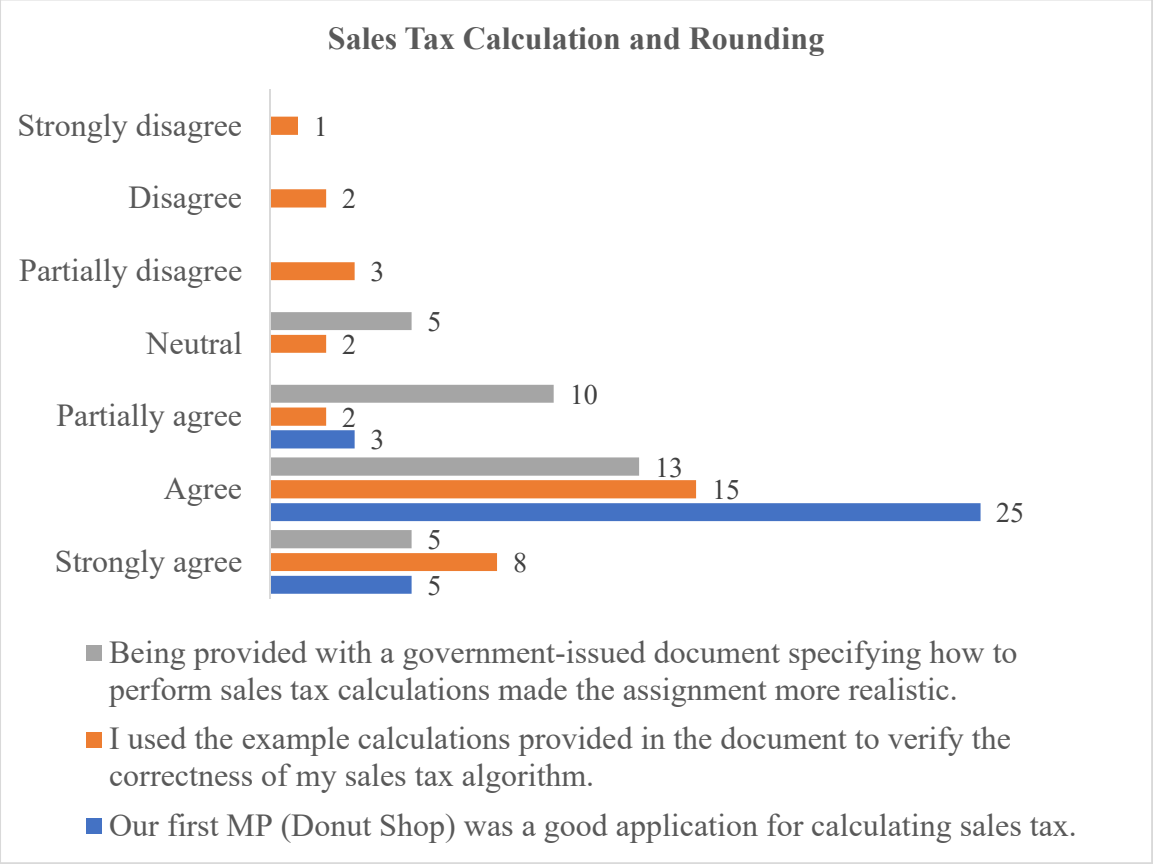

Figure 6. Sales tax questions.

It is somewhat surprising that 15 out of the 33 students in our sample were either neutral or just partially agreeing that an official document distributed by the Ohio Department of Taxation was realistic, when in fact the specified procedures listed in such government documents serve as a "gold standard" that all corporate entities operating within the state must adhere to. To obtain additional information from students regarding this aspect of the programming assignment, two qualitative questions were included with the survey. For the question, "in what way(s) do you believe that learning to correctly compute sales tax is important?" 29 (out of 33) responses were received. Most students responded in similar ways, in that such computations are "used every single day" as "it is a necessary part of any business's functions, so its accuracy is widely applicable to real world situations" because "we deal with sales tax all the time when we purchase pretty much any item." One student took a contrarian view in that they "don't think it's something we need to program, and most people know how to take a percentage and multiply it by their total," yet another student opined that such calculations are "a seemingly easy yet important function, which can translate over to programming in a way that allows for education on realistic programming." Finally, one student connected this concept to the marketplace by stating that "it's very important for the retail and service industries, who need to operate POS terminals that can

compute sales tax properly." With respect to the question, "in what way(s) do you believe learning how to correctly round numbers when dealing with monetary amounts is important?" the students needed to demonstrate some personal insight as to what some of the headaches might be, as the negative implications were not discussed in class. While some responses amongst the 30 received superficially hit the mark, such as "it occurs in the real world," "money only goes two decimal places," and "because not every number ends at the hundredths place and we don't have half cents," others were cognizant of the potential economic implications. Examples of such comments include the potential impact on customers, in that "if you round over, you could have angry customers for charging them too much" while several noted the potential impact on a business, one stating that "because even small rounding errors can add up over time causing… a store to lose a decent amount of money each year."

Finally, the big question: did students feel that they were asked to solve a realistic problem?

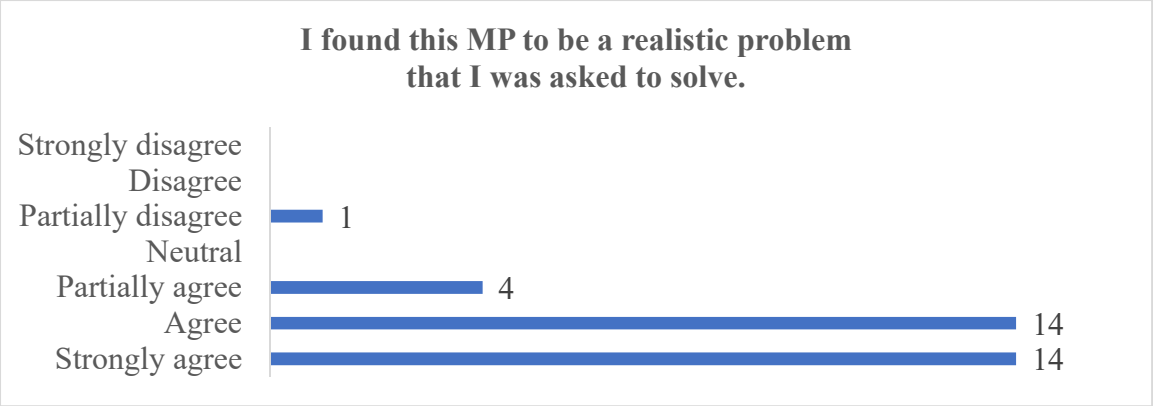

Figure 7. Was this assignment a realistic problem?

Here, there is overwhelming agreement (97%) that it was realistic to some degree, with 42% expressing strong agreement with this statement. Many of the qualitative responses to the question "in what way(s) do you believe that MP has helped you to develop professional skills" alluded to the realistic aspects of this assignment. One student appreciated that the problem "is relevant enough for a beginner… to understand its relevance, but its not too complicated" for a beginning programmer to come up with the code for the solution. It was, as another student mentioned, "a realistic challenge to provide a better understanding of programming." As much of this assignment was to be completed outside of both lab and lecture, a positive side effect were comments along the lines of "this MP allowed me to work outside of an educator's supervision and experience a small taste of what an outside programming job could be like… it felt more real than a lab question to calculate area or draw a diamond." Finally, one comment summarized the value of this assignment as follows:

> This MP has helped me to better understand concepts of real-world situations dealing with money, but it has also helped me expand on my programming knowledge. I think this was a great experience to teach students about how to apply coding to a real-world situation while also nailing down key concepts such as use of if-statements, proper variable naming, being consistent with the way you code and key elements like braces, etc. without being too overwhelming as a first large assignment."

7. Student Suggestions and Instructor Reflections

The final qualitative question was, "in what way(s) could this MP assignment have been improved?" Out of the 29 responses received, 12 had statements to the effect that "I don't think there's anything to be improved" while several others had positive comments like "it was a good start to our MP's." From the comments relating specifically to the assignment, a couple pointed out the negative ramifications that resulted from the provided selection of example transactions with respect to implementing proper sentence punctuation, including both Oxford commas and an ending period, in that either no change was needed or all change included pennies, such as:

```
Changed owed is $2.64 - 2 dollars, 2 quarters, 1 dime, 4 pennies.
```

This lack of "better variance in change results" caught some students off guard when they used demonstration test data that included change ending with something other than pennies, such as:

```
Changed owed is $2.60 - 2 dollars, 2 quarters, 1 dime.
```

as their punctuation algorithm incorrectly assumed that only the words "penny" and "pennies" would be followed by a period, and thus they hardcoded accordingly. As one student noted, "I came into the lab to test my MP and didn't expect to have to have my period transfer to the dime if needed." One student surprisingly complained about the rounding, in that "it was very finnicky, and some students were even having trouble walking through and finding the issue with a tutor/TA/instructor assisting them" and concluded with a request for "more instruction on how to deal with different situations in rounding… prior to the start of this MP would definitely help future students." However, another student stated "we learned the things we needed to complete the assignment, which helped later on while working on the MP." This student did go on to state a desire for "a more entertaining topic, which would be hard because we are beginner programmers." While there was carping on such topics as having points deducted for poor commenting or for not formatting the output exactly as called for in the specifications, most comments in this area were more along the lines of "I found it perfectly challenging and enjoyable for my skill-level, and the grading felt very fair and clear to interpret."

From the instructors' standpoint, the fact that all examples (save for the one where no change was needed) ended in pennies always being the last coinage unit was an unfortunate oversight that caused some students to naively assume that that's true for all cases. For those of us who have experienced many cash-based transactions over our lifetime we know that it is not true – but by growing up in an increasingly cashless society, most of our students do not have many experiences in giving or receiving change associated with a purchase. Sadly, some students caught by the test where the change ended in a dime fixed the period punctuation for just the dime in order to pass that specific test, but did not correct their remaining code for cases where the change ends with a different coinage unit, such as a nickel or a quarter. Consequently, they were surprised to see point deductions being taken for having incorrect code, even though their code passed the in-lab coding test. This issue can be addressed by providing a more exhaustive regimen of sample and test runs that go through all such change-making combinations. Additionally, providing a learning exercise ahead of code testing to allow students to discern that change can end in something other than pennies would make for a good "aha" moment that could then lead students in the correct direction.

8. Conclusions

One definition of entrepreneurial mindset, found in [30], centers on the set of cognitive behaviors that orient an engineer towards opportunity recognition and value creation in any context, not just that of an entrepreneurial venture. Specifically, it is about the engineer's (and in this context, also the computer scientist's) orientation toward problem solving. Given that this work takes place with the first major assignment in the first discipline-specific course that our computer engineering and computer science majors take, our focus with respect to KEEN's "3C's" approach to entrepreneurial mindset [9] – curiosity, connections, and creating value – is both embedded and introductory. The use of posing problems through storytelling helps to engage our students' curiosity by relating such problems to people for whom they will be designing a solution for. One way of making connections is by integrating information from various sources to gain insight. How many students outside of a business college have ever considered the nuts and bolts of how sales taxes are calculated? Or subsequently demonstrate an awareness of risks involved if the calculation is not performed correctly? Finally, creating value often stems from learning from, and persisting through, failure. Programming computers involves failure in abundance, especially for beginning programmers. Being taught early on about testing one's programs provides students with the on ramp to learning as they try out new concepts by incorporating them into algorithms and programs.

Overall, the primary learning objectives of the assignment were met: students correctly calculated the cost of a purchase, followed a specified algorithm for computing the sales tax, and determined the correct change to be returned. Additionally, almost all of the students agreed that this assignment was a real-world problem and displayed elements of EM in their qualitative responses. Finally, our survey data indicated that students appreciate assignments focused on real-world problem solving, even when given in the form of a scenario-driven story problem.

9. Resources

Experience reports should readily supply all interested readers with materials developed "in-house" to aid in adoption efforts by others. Accordingly, a "Card" - *i.e.*, an information repository - has been created for this paper on the Engineering Unleashed website operated by KEEN [31]. This card provides instructional materials for the Jim's Donut Shop Assignment with all of the materials mentioned in this paper, including the rubric, examples of student work, the survey questions, etc. These materials can be freely downloaded, reviewed, adopted, and if desired modified, by anyone for use in their courses under the Creative Commons CC BY-NC license [32].

References

[1] B. A. Becker and K. Quille, "50 Years of CS1 at SIGCSE: A review of the evolution of introductory programming education research," in *Proc.50th ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE '19)*, pp. 338–344, doi: 10.1145/3287324.3287432.

[2] N. B. Dale, “Most difficult topics in CS1: results of an online survey of educators,” *ACM SIGCSE Bull.,* vol. 38, no. 2, pp. 49–53, Jun. 2006, doi: 10.1145/1138403.1138432.
[3] T. J. Feldman and J. D. Zelenski, “The quest for excellence in designing CS1/CS2 assignments,” in *Proc. 27th ACM SIGCSE Tech. Symp. on Comput. Sci. Educ. (SIGCSE '96), pp.* 319–323, doi: 10.1145/236452.236564.
[4] B. Cantwell Wilson and S. Shrock, “Contributing to success in an introductory computer science course: a study of twelve factors,” in *Proc. 32nd ACM SIGCSE Tech. Symp. Comput. Sci. Educ. (SIGCSE '01)*, pp. 184–188, doi: 10.1145/364447.364581.
[5] Y. Qian and J. Lehman, “Students’ Misconceptions and Other Difficulties in Introductory Programming: A Literature Review,” *ACM Trans. Comput. Educ.*, vol. 18, no. 1, Article 1, 24 pages, Mar. 2018, doi: 10.1145/3077618.
[6] R. E. Anderson, M. D. Ernst, R. Ordóñez, P. Pham, and S. A. Wolfman, “Introductory programming meets the real world: using real problems and data in CS1,” in *Proc.45th ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE '14),* pp. 465–466, doi: 10.1145/2538862.2538994.
[7] L. J. Barker, C. McDowell, and K. Kalahar, “Exploring factors that influence computer science introductory course students to persist in the major,” in *Proc.40th ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE '09)*, pp. 153–157, doi: 10.1145/1508865.1508923.
[8] ABET, “FAQs for EAC C3 & C5 Criteria Changes.” ABET. https://www.abet.org/wpcontent/uploads/2019/04/FAQs-for-EAC-C3-C5-4-8-2019.pdf (accessed January 3, 2023).
[9] KEEN, “The Framework for Entrepreneurially Minded Learning.” Engineering Unleashed. https://engineeringunleashed.com/framework (accessed January 3, 2023).
[10] J. B. Hylton, D. Mikesell, J.-D. Yoder, and H. LeBlanc, “Working to instill the entrepreneurial mindset across the curriculum,” *Entrepreneurship Educ. Pedagogy*, vol. 3, no. 1, pp 86-106, Jan. 2020, doi: 10.1177/2515127419870266.
[11] L. Verschaffel, F. Depaepe, and W. Van Dooren, “Word Problems in Mathematics Education,” in *Encyclopedia of Mathematics Education,* S. Lerman, Ed., Dordrecht, Germany: Springer, 2014, doi: 10.1007/978-94-007-4978-8_163.
[12] D. Berg, “25th Anniversary Special: The Problem with Word Problems.” Making Math Real. https://www.makingmathreal.org/25th-anniversary-special-the-problem-with-word-problems-a-research-basis-part-i/ (accessed January 3, 2023).
[13] G. Daroczy, M. Wolska, W. D. Meurers, and H.-C. Nuerk, “Word problems: A review of linguistic and numerical factors contributing to their difficulty,” *Frontiers in Psychology*, vol. 6, Article 348, 13 pages, Apr. 2015, doi: 10.2289/fpsyg.2015.00348.
[14] T. Palm, “Impact of authenticity on sense making in word problem solving,” *Educ. Stud. Math*, vol. 67, pp. 37–58, Jan. 2008, doi: 10.1007/s10649-007-9083-3.
[15] B. Greer, “Modelling reality in mathematics classrooms: The case of word problems,” *Learning and Instruction*, vol. 7, no. 4, pp. 293-307, Dec. 1997, doi: 10.1016/S0959-4752(97)00006-6.
[16] D. M. Rembert, N. A. Mack, and J. E. Gilbert, “Exploring the Needs and Interests of Fifth Graders for Personalized Math Word Problem Generation, “in *Proc.18th ACM Int. Conf. Interaction Design and Children (IDC '19),* pp. 592–597, doi: 10.1145/3311927.3325309.
[17] K. R. Koedinger and M. J. Nathan, “The Real Story Behind Story Problems: Effects of Representations on Quantitative Reasoning,” *J. Learning Sciences,* vol. 13, no. 2, pp. 129-164, doi: 10.1207/s15327809jls1302_1.
[18] T. Palm, “Word problems as simulations of real-world situations: A proposed framework,” *For the Learning of Mathematics*, vol. 26, no. 1, pp. 42-47, Mar. 2006.
[19] C. Walkington and M. Bernacki, “Students authoring personalized ‘algebra stories’: Problem-posing in the context of out-of-school interests,” *J. Mathematical Behavior*, vol. 40, part B, pp. 171-191, Dec. 2015, doi: 10.1016/j.jmathb.2015.08.001.
[20] K. Reusser and R. Stebler, “Every word problem has a solution—The social rationality of mathematical modeling in schools,” *Learning and Instruction*, vol. 7, no. 4, pp. 309-327, Dec. 1997, doi: 10.1016/S0959-4752(97)00014-5.
[21] B. Hall, “3 Tips for Creating Math Word Problems That Boost Critical Thinking.” Edutopia. https://www.edutopia.org/article/3-tips-creating-math-word-problems-boost-critical-thinking (accessed January 3, 2023).
[22] G. Walker, “Word problems boring your students to death? Here’s how to fix them.” GameWise. https://gogamewise.com/how-to-make-math-word-problems-engaging/ (accessed January 3, 2023).

[23] N. V. Trakulphadetkrai, J.-A. Aerila, and S. Yrjänäinen, “Bringing mathematics alive through stories,” in *Story in Children's Lives: Contributions of the Narrative Mode to Early Childhood Development, Literacy, and Learning,* Cham, Switzerland: Springer, 2019, pp. 199-225, doi: 10.1007/978-3-030-19266-2_11.
[24] D. McDermon. “How to Tell a Story.” New York Times. https://www.nytimes.com/guides/smarterliving/how-to-tell-a-good-story (accessed January 3, 2023).
[25] P. Hodges, “How to Write a Story 101: Conflict.” The Write Practice. https://thewritepractice.com/conflict-101/ (accessed January 3, 2023).
[26] C. H. Papadimitriou, “MythematiCS: in praise of storytelling in the teaching of computer science and math,” *ACM SIGCSE Bull*., vol. 35, no. 4, pp. 7–9, Dec. 2003, doi: 10.1145/960492.960494.
[27] C. S. Horstmann, *Brief C++: Late Objects, Enhanced eText.* Hoboken, NJ, USA: Wiley, 2020.
[28] J. Swartzwelder (Writer) and J. Reardon (Director), “Homer at the Bat,” (Season 3, Episode 17), in A. Jean and M. Reiss (Executive Producers), *The Simpsons*, Gracie Films: Twentieth Century Fox Film Productions, Feb. 20, 1992.
[29] Ohio Department of Taxation, “ST 2005-05 – Sales and Use Tax Calculation and Rounding Change Effective 01/01/2006 - December, 2005,” tax.ohio.gov. https://tax.ohio.gov/business/ohio-business-taxes/sales-and-use/information-releases/st200505 (accessed January 3, 2023).
[30] J. M. Bekki, M. Huerta, J. S. London, D. Melton, M. Vigeant, and J. M. Williams, “Opinion: Why EM? The Potential Benefits of Instilling an Entrepreneurial Mindset,” *Advances Eng. Educ.*, vol. 7, no. 1, pp. 1-11, Fall 2018.
[31] “Jim's Donut Shop - A Real-World Scenario for Introductory Programming,” Engineering Unleashed. https://engineeringunleashed.com/card/3540
[32] Creative Commons. “Attribution-NonCommercial 4.0 International (CC BY-NC 4.0). License.” Creative Commons. https://creativecommons.org/licenses/by-nc/4.0/ (accessed January 3, 2023).

Appendix A. Survey Questions

Demographics:

1. Please select your class year
2. Please enter your major(s) and CS minor, if applicable (see list)
3. Please inform us regarding your relationship with the following course subject: Algebra
4. Please inform us regarding your relationship with the following course subject: Physics

Quantitative – 7-point Likert scale:

5. I have experience in completing assignments containing story problems.
6. I enjoy solving story problems.
7. I have experience in completing assignments containing real-world situations.
8. I prefer assignments featuring real-world problems to solve.
9. I do not like assignments that have more than one correct solution.
10. Rounding values to a certain number of decimal places is an example of a real-world problem.
11. Calculating a checksum does not have real-world applications.
12. Calculating the cost of a transaction is an example of a real-world problem.
13. Calculating sales tax is an example of a real-world problem.
14. Calculating amount of change to be returned is an example of a real-world problem.
15. Being provided with a government-issued document specifying how to perform sales tax calculations made the programming assignment more realistic.
16. I used the example calculations provided in the "ST 2005-05 – Sales and Use Tax Calculation and Rounding Change" document to verify the correctness of my sales tax algorithm.
17. Our first MP (Donut Shop) was a good application for calculating sales tax.
18. Our first MP was a good application for learning to round correctly when dealing with money.
19. Having a laboratory exercise over calculating how to make change helped to prepare me for completing this MP.
20. Having a laboratory exercise over calculating sales tax helped to prepare me for completing this MP.
21. Using meaningful variable names was important for this first MP.
22. Because of these assignments, I have a better understanding of why numerical rounding is an important subject in programming.
23. Because of these assignments, I have a better understanding of why numerical storage is an important subject in programming.
24. I found this MP to be a realistic problem that I was asked to solve.

Qualitative:

25. In what way(s) do you believe that learning to correctly compute sales tax is important?
26. In what way(s) do you believe learning how to correctly round numbers when dealing with monetary amounts is important?
27. In what way(s) do you believe this MP has helped you to develop professional skills?
28. In what way(s) could this MP assignment have been improved?