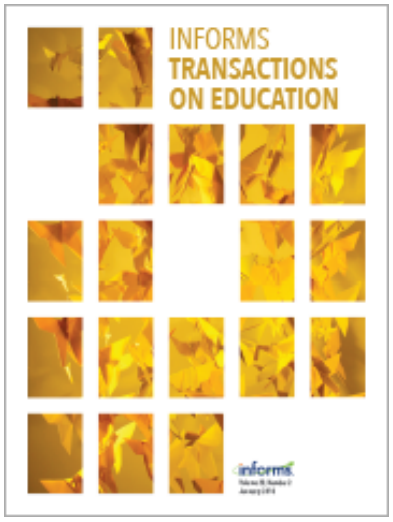
This article was downloaded by: [216.73.217.128] On: 06 November 2026, At: 19:26
Publisher: Institute for Operations Research and the Management Sciences (INFORMS)
INFORMS is located in Maryland, USA



INFORMS Transactions on Education

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Case Study—AdWords to AdWards: AI as a Coding Partner in Teaching Online Optimization Algorithms

Dessislava Pachamanova, Vera Tilson, Keely Dwyer-Matzky

To cite this article:
Dessislava Pachamanova, Vera Tilson, Keely Dwyer-Matzky (2026) Case Study—AdWords to AdWards: AI as a Coding Partner in Teaching Online Optimization Algorithms. INFORMS Transactions on Education

Published online in Articles in Advance 05 Nov 2026

. https://doi.org/10.1287/ited.2025.0147ca

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. You are free to download this work and share with others, but cannot change in any way or use commercially without permission, and you must attribute this work as "*INFORMS Transactions on Education*. Copyright © 2026 The Author(s). https://doi.org/10.1287/ited.2025.0147ca, used under a Creative Commons Attribution License: https://creativecommons.org/licenses/by-nc-nd/4.0/."

Copyright © 2026 The Author(s)

Please scroll down for article—it is on subsequent pages

With 12,500 members from nearly 90 countries, INFORMS is the largest international association of operations research (O.R.) and analytics professionals and students. INFORMS provides unique networking and learning opportunities for individual professionals, and organizations of all types and sizes, to better understand and use O.R. and analytics tools and methods to transform strategic visions and achieve better outcomes.
For more information on INFORMS, its publications, membership, or meetings visit http://www.informs.org

https://pubsonline.informs.org/journal/ited

INFORMS TRANSACTIONS ON EDUCATION
Articles in Advance, pp. 1–10
ISSN 1532-0545 (online)

AdWords to AdWards: AI as a Coding Partner in Teaching Online Optimization Algorithms

Dessislava Pachamanova,[a,*] **Vera Tilson,**[b] **Keely Dwyer-Matzky**[c]

[a] Babson College, Wellesley, Massachusetts 02457; [b] Simon Business School, University of Rochester, Rochester, New York 14627; [c] Department of Medicine and Pediatrics, University of Rochester Medical Center and Golisano Children's Hospital, Rochester, New York 14642

*Corresponding author

✉ dpachamanova@babson.edu (D. Pachamanova), vera.tilson@simon.rochester.edu (V. Tilson), keely_dwyermatzky@urmc.rochester.edu (K. Dwyer-Matzky)

https://orcid.org/0000-0002-1373-1553 (D. Pachamanova), https://orcid.org/0000-0001-8917-6122 (V. Tilson)

Received: May 24, 2025
Revised: March 20, 2026
Accepted: July 26, 2026
Published Online in Articles in Advance: November 5, 2026

https://doi.org/10.1287/ited.2025.0147ca

Copyright: © 2026 The Author(s)

Open Access Statement: This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. You are free to download this work and share with others, but cannot change in any way or use commercially without permission, and you must attribute this work as "*INFORMS Transactions on Education*. Copyright © 2026 The Author(s). https://doi.org/10.1287/ited.2025.0147ca, used under a Creative Commons Attribution License: https://creativecommons.org/licenses/by-nc-nd/4.0/."

Abstract. This case illustrates teaching online optimization algorithms for sequential decision making using generative artificial intelligence (GenAI) as an assistant, making complex concepts in algorithm design, implementation, and evaluation accessible even to nontechnical student audiences. Students are asked to translate an unfamiliar healthcare setting into the Google AdWords problem. The well-known structure of the Google AdWords problem both facilitates students' conceptual learning and makes it easier for GenAI tools to assist with algorithm implementation. However, the complexity of the healthcare environment makes the mapping difficult, requires discussions with "experts" with domain knowledge and various stakeholders in the process, and raises ethical questions. The case is based on an actual process improvement project aimed at maximizing the use of a hospital's capacity. The main character, Dr. Kelly, heads an observation unit and had tried a predictive model for patient assignments to her unit versus the hospital wards, which, however, can sometimes lead to inefficient bed capacity management. She is looking to improve patient allocation decisions made in real time. The case empowers students to (1) identify situations in which sequential decision-making models and online algorithms are appropriate and preferable to predictive models, (2) discover the concept of inventory balancing, (3) map its application in an unfamiliar setting (healthcare) to a well-studied setting (advertising) through inquiry and stakeholder analysis, and (4) embrace GenAI as a partner in algorithm implementation and testing. Realistic data, code, and models are provided. The rich context of the case makes it appropriate for use in courses in operations management, business analytics, marketing, machine learning, and operations research, both at the advanced undergraduate level and at the masters/MBA level.

Supplemental Material: The Teaching Note is available at https://www.informs.org/Publications/Subscribe/Access-Restricted-Materials.

Keywords: online algorithms • inventory balancing • prescriptive analytics • healthcare • artificial intelligence • generative AI • AI-assisted coding • ethics • stakeholder engagement

1. Introduction: Case Learning Goals

Online (also referred to as "real-time") optimization algorithms are algorithms that respond to new information as it arrives. A classic example is the Google AdWords problem (Mehta et al. 2007). The goal in the classical Google AdWords problem is for an online advertising platform to allocate a sequence of advertising spaces associated with continuously arriving search queries to budget-constrained advertisers in a way that maximizes the platform's expected revenue. The BALANCE algorithm, developed by Kalyanasundaram and Pruhs (2000), provides an elegant solution to the general online matching problem that the Google AdWords problem represents.

Online optimization algorithms have been developed for different contexts in operations management, supply chain management, and, more generally, analytics-driven decision making, such as inventory balancing for real-time optimization of personalized assortments (Golrezaei et al. 2014), order fulfillment optimization (Acimovic and Graves 2015, Acimovic and Farias 2019), shadow routing for flexible multiserver pools (Stolyar and Tezcan 2011), appointment scheduling (Liu et al. 2019), and hospital bed allocation (Dwyer-Matzky et al. 2021). These applications represent substantial and important areas of business practice, which, however, are challenging to teach to a business audience. A hands-on approach to learning quantitative concepts is

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

particularly successful with business students (Xu and Babaian 2021); however, teaching advanced sequential decision-making algorithms typically requires implementation through coding. A coding background and strong quantitative skills are not generally taken for granted with audiences with non-STEM backgrounds (Weathers and Aragón 2019). At the same time, tremendous advances in technology and generative artificial intelligence (GenAI) in recent years provide tools to make STEM concepts accessible to these different audiences and represent a natural next step in business education.

The default modeling environment for business school cases is typically Microsoft Excel—a familiar and widely used spreadsheet tool (Seal et al. 2020, Diamant 2024). Programming has generally been considered time-consuming and challenging to teach to the business school audience (Murray 2014). However, in our recent experience, business school students are interested in learning more about coding despite feeling some trepidation about it—and learning more makes them feel empowered. The case we present in this article combines Microsoft Excel and Python, an open-source programming language. Students can test their approach in Excel through guided exercises, but then use the insights to generate Python code so that their algorithmic approach can be generalized and implemented on a larger scale.

The case guides students to generate the code using GenAI tools, thus providing an introduction to AI-assisted programming (sometimes referred to as "vibe programming" or "vibe coding"). AI-assisted programming describes the process of a human providing instructions to an AI system in natural language, which the AI system interprets to generate, refine, and debug code to implement the instructions. Instead of manually writing and inspecting each line of code, the human guides the AI through iterative prompts and feedback cycles. Although there is an ongoing debate on approaches, effectiveness, and consequences of using AI-assisted programming, it has been making steady inroads into software development in practice (Meske 2025, Ray 2025) as well as into computer science education (Kusper and Szabó 2025, Geng et al. 2026).

The case is presented from the point of view of Dr. Kelly, the medical director of an observation unit (OU) at a midsize hospital in the United States. Dr. Kelly had been using a predictive model to determine which observation status patients are more likely to "flip," or switch to inpatient status, and have to be transferred from the OU to the hospital wards. If observation status patients are predicted to flip, they would be sent to the hospital wards directly instead of being assigned to the OU first. However, Dr. Kelly observed that this policy tended to result in an imbalance of occupancy of the beds in her OU and the hospital wards. The case asks students to implement an online budget-balancing algorithm designed for the classical Google AdWords problem using Microsoft Excel and Python (with the help of GenAI), evaluate performance, and then map the inputs of this online optimization algorithm to the OU context to address Dr. Kelly's problem. To make the student experience realistic and comprehensive, the case further provides an opportunity for instructors to review predictive analytics concepts if they wish, as the predictive model is provided in the case. This case is based on a real project, described in Dwyer-Matzky et al. (2021), and instructors may find it helpful to read the paper to understand the rich context in which the decisions in the case need to be made in practice.

The case has several primary learning objectives and emphasizes critical thinking skills:

1. *Problem framing*. To recognize when a situation merits designing algorithms for online optimization.
2. *Algorithm design*. To understand the main idea behind online optimization algorithms and important concepts for their performance evaluation, such as the "clairvoyant" algorithm and competitive ratio.
3. *Algorithm inputs*. To determine inputs for the algorithm that make sense in the context addressed by the algorithm.
4. *Assessment relative to status quo*. To understand how to demonstrate the value of an algorithm to important stakeholders.

The case also imparts valuable and timely experience with GenAI tools, including experience with:

1. *Human-AI interface*. Discovering metrics of importance to stakeholders and specifics of the possible operationalization of an algorithm is paramount, even if one can rely on AI help in coding the actual algorithm.
2. *Prompt engineering*. Designing good prompts to guide GenAI toward a desired solution can save time and reduce implementation error.
3. *Debugging*. When coding an algorithm with GenAI's help, it is important to have created simple examples beforehand, in which the answers are known, so that the output of the code can be tested.

We emphasize that developing students' coding skills with GenAI tools is not the primary objective of the case. The case already has an ambitious scope; it exposes students to the process of learning about a problem in an unfamiliar context, identifying whether a particular algorithmic approach is appropriate, researching how the algorithmic approach can be implemented in that context, and learning how the success of the algorithmic approach can be tested and evaluated. However, GenAI tools are an important component, as they support students on their learning journey by making the core ideas—online optimization algorithms and algorithmic thinking more broadly—accessible and less intimidating even to students without a STEM background.

The rest of this case article summarizes existing literature on pedagogical approaches to online algorithms

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

and GenAI use in the classroom (Section 2), outlines the case in more detail (Section 3), and shares our classroom experience teaching it (Sections 4 and 5). The case article ends with concluding remarks and an appendix containing a list of the resources provided with the case.

2. Literature Review

Although familiar in computer science, to the best of our knowledge, online algorithms such as BALANCE have not been incorporated in published cases in the fields of operations research and decision sciences. However, closely related areas—revenue management, demand management, markdown management, inventory management—often deploy algorithms that share common characteristics (Phillips 2003, Dhumal et al. 2008, Shumsky 2009, Liu et al. 2013, Drake and Atkins 2021). Teaching cases on optimization algorithm concepts targeted at non-STEM students typically introduce algorithms through classroom games, high-level discussions, and Excel spreadsheet implementation (Vliegen and Zonderland 2017, Kong 2019, Iranpoor 2021, Harris and Forbes 2023, Beliën et al. 2025, DiMichele and Forbes 2025, Van den Broeke and Mlinar 2025). Visualization tools have also been suggested to facilitate the process of teaching algorithms (Kress and Dornseifer 2015).

Our search revealed a significant gap in the literature when it comes to articles that specifically address using GenAI to teach algorithms to business audiences. This case contributes to the literature by addressing this gap.

Although GenAI tools such as ChatGPT have been around since late 2022, many instructors are wary of their use (Ross 2023). Multiple disciplines, for example, computer science, have had to rethink their approaches to accomplishing course goals given the ease with which GenAI tools complete assignments and exams of foundational concepts (Ellis et al. 2024). However, there is also widespread recognition that GenAI tools significantly enhance the students' experience. When using GenAI tools in computer science classes, students can be provided with support in code generation (Abolnejadian et al. 2024), timely and personalized feedback (Nguyen and Allan 2024), enhanced error messaging and debugging tips (MacNeil et al. 2024), and code explanations (MacNeil et al. 2022, Leinonen et al. 2023, Taylor et al. 2024). Students also register higher levels of engagement with the material (Liu and M'hiri 2024).

General classroom practices to take full advantage of GenAI tools have been suggested but are still nascent (Ting and Hong 2024). In particular, there is a lack of understanding of how to guide GenAI tools effectively to generate better code (Chen and Liang 2024) and how to assess learning outcomes when students utilize GenAI tools (Freire et al. 2023, Agbo et al. 2025).

It is known that GenAI tools foster learning autonomy while reducing student cognitive load (Prather et al. 2023), but studies advocate for balanced integration of GenAI in the classroom, where GenAI "complements rather than replaces" (Agbo et al. 2025). In line with such studies, this case takes the view of GenAI tools as instructor assistants to help business students understand online optimization algorithm concepts while becoming aware of the power and limitations of GenAI tools.

3. Flow of the Case

The case described in this case article allows for introducing a wide range of important concepts in operations research, operations management, management science, marketing, computer science, and modeling more generally. Topics include using predictive models, transitioning from a predictive to a prescriptive setting and eventually dynamic decision making, testing a model, as well as ethical implications of employing a model in a particular setting (healthcare). We explain the flow of the case in more detail in this section. The appendix of this case article lists the extensive additional resources we provide for instructors and students, including a detailed teaching note, Excel model files, and scaffolded Python notebooks. Each subsection in this section maps to a corresponding subsection in the teaching note (starting with "TN.") that provides more detail and specific answers.

3.1. Setting Up the Case Background

Section TN.2.1 of the teaching note contains a set of introductory questions and answers to help an instructor start the class discussion. If the background of the students allows, the instructor could also discuss the decision tree predictive model provided in Appendix A of the case. (However, we note that this is not necessary. The instructor can skip the interpretation of the predictive model. In our experience, to proceed with the case, it suffices to explain to students that the predictive model would simply recommend whether a patient should be assigned to the OU or the wards.) If the instructor has time to spend on introducing predictive models or if students in the class have previously covered such models, the teaching note lists a number of interesting questions that could be asked, including questions about model interpretation, the reason for developing this particular type of model, and the ethics of using a predictive model in the healthcare context.

3.2. Testing the Recommendations of a Predictive Model on Real Data

The concepts of testing the recommendations of an algorithm and the appropriate process for doing so are often unfamiliar to business students, even if they are obvious to most instructors with an analytics background. We provide an Excel file **ModelTesting_Student.xlsx** that contains a generated scenario (which can

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

be thought of as historical data) for patient arrivals in a day. The file can be used in class to demonstrate the bed inventory in the OU and the hospital wards if various decision-making rules are used. This can also provide context and an easy explanation of the concept of an "all-knowing" algorithm for students, who might have read the case in advance but feel intimidated by a term such as "clairvoyant." The file **ModelTesting_Instructor.xlsx** contains the answers. See Section TN.2.2 in the teaching note for further discussion.

3.3. From Predictive to Prescriptive Analytics

The central theoretical concept in the case is online optimization algorithms. The case explains the main concept and provides examples in Appendix B. Good references for the original Google AdWords problem are Mehta et al. (2007) and Mehta (2013).

In our experience, business students are often intimidated by mathematical notations such as the description of the "clairvoyant" problem and the definition of the function $\Psi(x)$ for calculating the reward penalty given the remaining budget of each advertiser. We have included some explanation in the case, but the key to explaining this concept is utilizing the Excel file **OnlineAlgorithmsIllustration_Student.xlsx** we provide in class.

The file **OnlineAlgorithmsIllustration_Student.xlsx** begins to set up the implementation of the online algorithms in Python by defining the inputs to the Google AdWords problem in separate worksheets (**BiddersBudgets**, **QueryTypesBidsClickThroughRates**, and **QueriesSequence**). Students will be asked to map these inputs to the healthcare setting in the next step of the case. At the current stage, outlining the inputs to the familiar Google AdWords online algorithm and then illustrating the algorithm in a separate spreadsheet helps students delineate the algorithm part from the inputs, helping the instructor make the point later that many algorithms can be very similar despite different contexts if the inputs are correctly mapped.

The worksheets **AlgLinear** and **AlgExp** contain an implementation of the simple example of online optimization algorithm from Appendix B in the case. The worksheet **AlgLinear** allows the instructor to explain the intuition for various aspects of an online algorithm, including how the rewards are calculated and the role of the function $\Psi(x)$, how the decisions are made, and how the budgets are updated. Business students are usually familiar with Excel, and illustrating the example in Excel first before the upcoming implementation with Python ensures that students can visualize and understand the algorithm itself. Worksheet **AlgExp** is set up so that the instructor can assign it to students in breakout groups. It enables students to understand how the algorithm is modified by using an exponential function $\Psi(x)$. (For many business students, the concept of an exponential is unfamiliar (or not revisited since high school), so even the action of typing it up in a spreadsheet helps them understand what the role of the function $\Psi(x)$ is in the algorithm's recommendations.)

The file **OnlineAlgorithmsIllustration_Instructor.xlsx,** provided with the instructor materials, contains the answers to the student group exercise. In addition to the inputs of the algorithm in the first three worksheets (**BiddersBudgets**, **QueryTypesBidsClickThroughRates**, and **QueriesSequence**) and the actual algorithm implementations in worksheets **AlgLinear** and **AlgExp**, the instructor could discuss with the class what outputs could be of interest to the decision maker. For more information, see Section TN.2.3 in the teaching note.

3.4. Developing Code with the Help of AI Tools

Having seen the flow of the online algorithm in the Excel file **OnlineAlgorithmsIllustration_Student.xlsx**, the students understand the algorithm better and feel more comfortable +with the idea of trying to code it. We distribute the input file **AdWordsOnlineAlgorithmInput.xlsx** and provide students with instructions on how to approach writing effective prompts for AI tools. Section TN.2.4 in the teaching note describes the prompts we provided. Students can use any AI coding assistant capable of generating, explaining, and debugging Python code. Examples at the time of writing included ChatGPT, Claude, Gemini, and GitHub Copilot.

A helpful review of these tools is available at https://clickup.com/blog/microsoft-copilot-alternatives/.

If the instructor has some margin in terms of class time, a fun assignment for students is to ask a GenAI tool to summarize research principles for writing good prompts for programming with GenAI tools before getting started with the assignment. These tips can sometimes be a bit generic and unintelligible, so we created a more specific version for the context of the case (see also Hamilton 2025):

1. *Think about the user interface*. What would you like the format for the inputs and the outputs to the program to look like? The more generalizable it is (e.g., passing in an input file instead of typing in entries), the easier it is to extend to other versions of the algorithm and to other contexts.

2. *Think about the core algorithm*. What do you want the code to do? Describe each step of the algorithm unambiguously.

3. *Think about how to check whether the code is doing the correct thing*. A good approach is to develop several simple scenarios where the answers are known so that the outcome of the algorithm can be tested in those scenarios.

4. *Test and iterate*. Have a process by which you generate additional prompts to correct issues with the algorithm you have implemented with the help of

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

GenAI tools. Do this often to catch errors early, ideally at each logical step in the implementation.

Section TN.2.4 in the teaching note provides a step-by-step approach to implementing an online optimization algorithm with GenAI that results in code that takes in an input file, runs the algorithm, and produces the outputs of interest. We use the **AdwordsOnlineAlgorithmInput.xlsx** file (which contains the first three worksheets of the **OnlineAlgorithmsIllustration_Student.xlsx** file that the students had already seen) as input. The Python notebook **IterativeOnlineAlgorithmCoding_Student.ipynb** contains step-by-step instructions of the exploration one would do with GenAI, including backup code students can run in case they cannot get their code to work. The full code (with answers and generated output) is provided with this case's supplemental materials in the Python notebook **IterativeOnlineAlgorithmCoding_Instructor.ipynb**.

Some instructors may be interested in closing the gap between giving students a full prompt and having them write it themselves from scratch. Our suggestion, based on review of the computer science literature and on our own experience as educators, is to provide intermediate versions of the Python notebook to students at the end of each stage of the case. This ensures that students have the opportunity to experiment with prompt writing, but then all students can proceed to the next stage of the case. To facilitate this "scaffolding" strategy for instructors, we provide four Python notebooks (all starting with **IterativeOnlineAlgorithmCoding …**) that contain various degrees of detail; please see a detailed description of the content of the notebooks in the appendix to this case article. Instructors can use these notebooks to structure a prompt writing exercise if they wish, as the level of information across the four notebooks graduates from only instructions (**IterativeOnlineAlgorithmCoding_Student_Shell.ipynb**) to sample code (**IterativeOnlineAlgorithmCoding_Student.ipynb**) to answer code for the AdWords algorithm implementation but not the OU implementation (**IterativeOnlineAlgorithmCoding_Student_AdWordsAnswers.ipynb**) and, finally, to complete answers for the instructor, including the OU mapping exercise implementation (**IterativeOnlineAlgorithmCoding_Instructor.ipynb**). We have found it especially useful to start with **IterativeOnlineAlgorithmCoding_Student_ Shell.ipynb**, which allows students to try to generate prompts themselves, and then distribute **IterativeOnlineAlgorithmCoding_Student_AdWordsAnswers.ipynb** at the end of the exercise to ensure that students who were not successful in generating correct code for the AdWords algorithm can be included in the remaining case activities and complete the portion of the case that deals with mapping the AdWords algorithm to the upcoming OU part of the case assignment (described in Section 3.5 of this case article and Section TN.2.5 of the teaching note.)

3.5. Mapping a Known Algorithmic Setting to a New Context

A very important part of the case is translating a new context into a familiar context. This helps students understand the generalizability and cross-domain transferability of what they are learning. After developing code that takes as input the file **AdWordsOnlineAlgorithmInput.xlsx**, students are asked to analyze the context of the OU problem and decide whether there is a mapping between the Google AdWords problem inputs and the OU context. Section TN.2.5 in the teaching note lists possible questions, exercises, and answers.

Having mapped the inputs to the Google Adwords problem to the OU patient placement problem, students can fill out an Excel input file (**OUAlgorithmInput_Student.xlsx**) and modify their AdWords code to obtain recommended patient placement for a simulated sequence of patient arrivals. Possible answers are provided in the file **OUAlgorithmInput_Instructor.xlsx**. There are several tricky parts to the implementation, described in detail in Section TN.2.5 in the teaching note. The last code section in the file, **IterativeOnlineAlgorithmCoding_Student_AdWordsAnswers.ipynb**, provides instructions for students on how to troubleshoot these implementation issues, and the file **IterativeOnlineAlgorithmCoding_Instructor.ipynb** contains the answers for instructors.

3.6. Wrap-Up

This case empowers students to (1) identify situations in which sequential decision-making models and online algorithms are appropriate and preferable to predictive models, (2) discover the concept of inventory balancing, (3) map its application in an unfamiliar setting (healthcare) to a well-studied setting (advertising) through inquiry and stakeholder analysis, and (4) embrace GenAI as an assistant in algorithm implementation and testing. A final wrap-up allows an instructor to point out to students the multifaceted nature of what they have just learned and accomplished.

4. Student Background, Reactions, and Learning Outcomes

This case has been used twice in a master's-level elective course on sequential decision making and reinforcement learning taken by a mix of part-time MBA, full-time MBA, MS in business analytics, and MS in finance students. Students typically come to the course with basic knowledge of MS Excel, exploratory data analysis, regression modeling, and simulation (covered in the core courses in our institutions' master's programs), although some students may have also taken more advanced electives such as machine learning and optimization. We use the case in the third class of the semester, after students have reviewed exploratory data

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

analysis and regression models, and have been introduced to writing simple scripts for GenAI that would produce Python code for basic exploratory data analysis and predictive models. The case combines well with the topics covered in Pachamanova et al. (2022a, b), and the latter can be used to review descriptive and predictive analytics before the case described in this article is introduced to students. However, this case is completely self-contained and can be used independently as well. Other potential courses where this case could be a good fit would be at the end of core MBA business analytics courses (to help students understand the transition from static to dynamic models), computer science courses (to teach core algorithmic concepts), or marketing courses (to illustrate algorithms for online advertising). We cover the case in a three-hour session, but the case can be expanded to 4.5 hours or split into two sessions; see the teaching plan in Section TN.3 of the teaching note and Section 5 in this case article. Students have access to Excel and use their own laptops during class.

Student reactions to the case have been very positive. We present below summaries of student opinion survey results for one section of the master's-level sequential decision making elective. Institutional review board (IRB) approval was obtained for this project.

Students were given brief online surveys that were anonymous and voluntary in accordance with IRB guidelines. The survey questions are shown in Table 1.

Although students were told that participation in the survey was optional, the response rate was high. There were 29 responses out of 32 attendees (a 91% response rate). Figure 1 summarizes the students' answers to question 1 from Table 1. The students ranked the first learning competency, "mapping a business situation to a problem definition so that algorithms developed for a different application can be used to address the current situation" as their top takeaway from the case, followed by "learning about the concept of an online algorithm."

Overall, students were very satisfied with their learning and experience with the case. 100% respondents answered "1" or "2" to questions 2 and 3 in Table 1, respectively. A sample of comments from the anonymous student evaluations is provided below:

> "A great learning experience."

> "Overall the case was well structured and the group exercises were helpful in discussing difficult topics in a group setting. Reinforced my understanding of what is important in problem solving."

> "Well, compared with the previous […] class I took last semester, this approach is much better! I can imagine the usage of the case in the real life business. Using healthcare context also enlightened me with new knowledge about the landscape of the industry."

> "Difficult material but great and well organized."

> "Very interesting and engaging case."

> "Loved it, thank you."

> "I loved it. I can't believe I learned this much in one afternoon. I appreciated understanding moving from descriptive, to predictive to prescriptive analytics keeping into consideration dynamic decision making. The application of online optimization is all around us, and it's great to see how this magic comes to life."

Some students felt the mathematics was a bit complex, especially in calculating the rewards in the online algorithm, but overall, they felt that the concepts got clarified through the in-class exercises and group discussions.

5. Lessons Learned

This section summarizes observations from teaching this case that might be helpful to other instructors. Sections TN.4 and TN.5 in the teaching note contain additional detail, as well as information about student assessment.

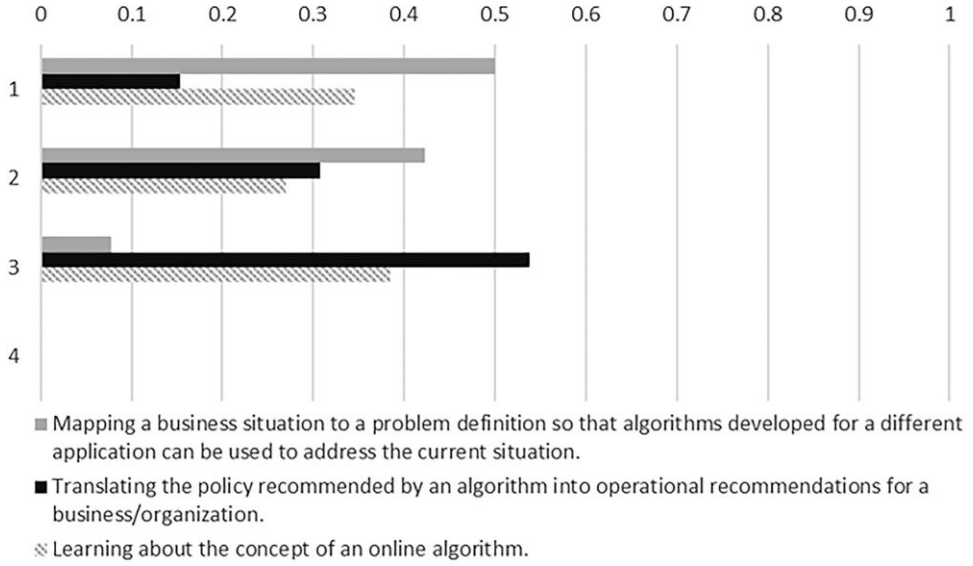
First, business students might find some notations in the case intimidating (e.g., the summation sign $\sum$ and the Greek letter Ψ that represents the penalty function).

Table 1. Wording of Questions from the Student Survey

Q1. *Please rank, in order of importance to you, the following learning competencies from the Online Optimization for Hospital Observation Unit Patient Placement exercise, with 1 being most important:*
- *Mapping a business situation to a problem definition so that algorithms developed for a different application can be used to address the current situation.*
- *Translating the policy recommended by an algorithm into operational recommendations for a business/organization.*
- *Learning about the concept of an online algorithm.*
- *Other (please specify)*

Q2. *Please rank your learning in the observation unit case:*
1. Learned a surprising amount 2. Learned a good deal 3. Neutral 4. Did not learn much 5. Did not learn at all
Q3. *Please rank your experience with the case:*
1. Very Satisfied 2. Satisfied 3. Neutral 4. Dissatisfied 5. Strongly Dissatisfied
Q4. *Please use this space for any additional comments you would like to make about the Online Optimization for Hospital Observation Unit Patient Placement case exercise.*

Figure 1. Distribution of Student Answers for Survey Question 1



Notes. The vertical axis shows the ranking given to each option. Tabulated as percentage of responders who gave a particular ranking.

It may therefore be helpful for instructors to prepare the students by explaining a few aspects of the case before assigning it. We note that the prework assignment (see Section TN.3 in the teaching note) does not rely on students being able to read the mathematical notation. Thus, students can be encouraged to skim through the case without worrying about not understanding the mathematical notation and then read the appendix of the case, which clarifies the concepts introduced in the case through simple examples. We also tell our students that it is helpful for them to have some experience reading academic writing.

Second, although the case is modular and instructors can select to teach only parts that are most relevant for their audience, we strongly recommend not skipping the Excel implementation of the online optimization algorithm in file **OnlineAlgorithmsIllustration.xlsx** and going over it in class (see Section 3.3 in this case article and Section TN.2.3 in the teaching note). It is very important for the success of the case. The spreadsheet model clarifies the mechanics of the online optimization algorithm for students before tackling the more complex implementation using coding with GenAI. The student comments we collected indicate that the algorithm implementation was very helpful for clarifying notations they did not understand from the case. Ultimately, our students were satisfied to have struggled a bit with reading the case and then navigated the challenge because they enjoyed the approach to teaching the subject as a whole.

Third, although GenAI tools have been constantly improving, prompting GenAI to generate code may result in errors. We have flagged two instances of such errors in the Python notebook included with the case (**IterativeOnlineAlgorithmCoding_Instructor.ipynb**) and explained them in Section TN.2.4 in the teaching note. However, given the probabilistic nature of GenAI, it is not possible to anticipate all coding errors that could occur. It is therefore very important to have test cases. In class, we emphasize the need to understand and double check whether the simple examples in the appendix of the case work for the code generated with AI. GenAI can be prompted to explain how the code can be debugged or make changes to the code directly, and its capabilities are getting better over time. We also recommend that instructors take advantage of the scaffolded versions of the file **IterativeOnlineAlgorithmCoding_Student.ipynb** we provide to allow students who cannot get their GenAI-assisted code to work to catch up with the rest of the class at important stages of the case discussion.

Fourth, when teaching the case, we assumed that students did not have experience with prompts, and we provided full prompts as a way for them to learn. We also provided backup code if they could not ultimately obtain correct code. This worked very well with our student audience, and no student felt left behind in class. We note that our course taught online optimization concepts, and our students' ability to use GenAI tools was not the focus of the course. However, the case could potentially be used in computer science classes to teach scaffolding prompts to optimize code writing. There is a growing literature in computer science and engineering education on iterative prompt generation; see, for example, Denny et al. (2023, 2024) and Pădurean et al. (2025).

Effective prompt generation is still an evolving field, and no consensus appears to have been reached among

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

educators on the most effective way to do it (Agbo et al. 2025). Therefore, we are wary of providing specific instructions that may become outdated by the time this case is published. For example, many contemporary GenAI systems can generate, execute, and iteratively refine code within integrated development or analysis environments. Instructors can easily find resources to help students write better prompts, including by prompting GenAI to suggest ways to improve. That said, our students were quickly able to transition from being given a prompt to writing good prompts themselves.

Finally, we acknowledge that this case might be intimidating to business school instructors who teach primarily using Excel. We note that the case can be taught entirely using Excel — an instructor could cover the material in Sections 3.1–3.3 in this case article (Sections TN.2.1–TN.2.3 in the teaching note) and then move directly to Section 3.5 in this case article (Section TN.2.5 in the teaching note). In other words, students can be asked to do the mapping of the OU context to the AdWords context without implementing the algorithm in Python.

However, in our experience, students enjoy interacting with GenAI tools to implement the online optimization algorithm. It may be helpful for instructors to know that although some experience with Python (or another programming language) is recommended, instructors do not need to be proficient programmers to teach the coding part of the case successfully. Helpful references on using Python in the OR classroom include Alderson (2022) and Isken (2025). GenAI tools can also advise instructors on platforms within which Python can be effectively used in class. We encourage instructors who are not as familiar with Python simply to follow the outline of the Python notebook we provide with the case (**IterativeOnlineAlgorithmCoding_Instructor.ipynb**) and to enter the prompts in the notebook to recreate a similar experience for their students.

6. Conclusions

Taking advantage of GenAI as a translator of human instructions to computer instructions is one of the most impactful and useful applications of GenAI in education. The case described in this article presents an opportunity to illustrate complex sequential decision-making algorithms through GenAI's support as an implementation assistant. The main pedagogical value of the case is in mapping a healthcare context to a well-studied problem (Google AdWords) so students can leverage existing research, understand algorithmic transferability across domains, and use AI to translate their understanding into executable code. However, the process of creating the appropriate inputs for the algorithm very much requires human input, simulating how a real industry project might work. Student feedback on the case has been overwhelmingly positive, and we hope that other instructors will find the context, approach, and outcomes from the case useful for their courses' learning goals.

Acknowledgments

This case was a finalist in the 2025 Decision Sciences Institute Best Teaching Case Competition. The authors like to thank the dedicated staff at the hospital with which we collaborated, as well as our students, who helped us test the case and provided useful feedback. The authors are also very grateful to the ITE editor, associate editor, and three anonymous referees for their helpful comments and suggestions.

Appendix. List of Resources Provided with the Case

A.1. Data Files

OUSummaryData_Student.xlsx: Contains the data from Table 1 of the case.

A.2. Teaching Note

Contains discussion of the topics introduced in this case article, including a teaching plan.

A.3. Model Files

ModelTesting_Student.xlsx: A spreadsheet template with a generated scenario (can be thought of as historical data) for patient arrivals in a day that can be used to observe the bed inventory in the OU and the hospital wards if different decision-making rules are used.

ModelTesting_Instructor.xlsx: Answers to ModelTesting_Student.xlsx.

OnlineAlgorithmsIllustration_Student.xlsx: A spreadsheet template with the setup for the simple example from Appendix B of the case that can be distributed to students so they can practice implementing the online algorithm with an exponential (instead of linear) function in worksheet **AlgExp**. (see Section TN.2.3.)

OnlineAlgorithmsIllustration_Instructor.xlsx: Complete file, with answers to the student exercise in worksheet AlgExp from file **OnlineAlgorithmsIllustration_Student.xlsx**.

AdWordsOnlineAlgorithmInput.xlsx: An Excel file that can be used as input to the Google AdWords implementation with Python. (See Section TN.2.4.)

OUOnlineAlgorithmInput_Student.xlsx: An incomplete Excel file that students are asked to fill out as part of a group exercise in class and can be used as input to the online algorithm implementation with Python. (See Sections TN.2.4–TN.2.5.)

OUOnlineAlgorithmInput_Instructor.xlsx: Completed Excel file that students are asked to complete as part of a group exercise in class and can be used as input to the online algorithm implementation with Python. (See Sections TN.2.4–TN.2.5.)

IterativeOnlineAlgorithmCoding_Student_Shell.ipynb: Python notebook with step-by-step instructions for students to implement the algorithm; can be distributed to students for work during the class. No code provided, only instructions.

IterativeOnlineAlgorithmCoding_Student_AdWordsAnswers.ipynb: Python notebook that is the same as **IterativeOnlineAlgorithmCoding_Student_Shell.ipynb**; however, the final code for the AdWords algorithm is provided. Can be

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.

distributed by instructors if students are not successful in generating correct code for the AdWords algorithm so that all students can complete the algorithm for the OU part of the assignment.

IterativeOnlineAlgorithmCoding_Student.ipynb: Python notebook that is the same as **IterativeOnlineAlgorithmCoding_Student_AdWordsAnswers.ipynb** but also contains code generated with step-by-step instructions for students to implement the algorithm; can be distributed to students for work during the class. The questions also contain answer code for students to use if they cannot get the correct code on their own. Appropriate if the student audience is not familiar with writing prompts at all. Does not contain answers to the OU portion of the case.

IterativeOnlineAlgorithmCoding_Instructor.ipynb: Complete Python notebook with step-by-step instructions for students and corresponding answers for instructors, including the full implementation of the OU exercise.

References

Abolnejadian M, Alipour S, Taeb K (2024) Leveraging ChatGPT for adaptive learning through personalized prompt-based instruction: A CS1 education case study. Mueller FF, Kyburz P, Williamson JR, Sas C, eds. *CHI EA'24: Extended Abstracts of the CHI Conf. Human Factors Comput. Systems* (Association for Computing Machinery, New York), 1–8.

Acimovic J, Farias VF (2019) *The Fulfillment-Optimization Problem*, INFORMS TutORials in Operations Research (INFORMS, Catonsville, MD), 218–237.

Acimovic J, Graves SC (2015) Making better fulfillment decisions on the fly in an online retail environment. *Manufacturing Service Oper. Management* 17(1):34–51.

Agbo FJ, Olivia C, Oguibe G, Sanusi IT, Sani G (2025) Computing education using generative artificial intelligence tools: A systematic literature review. *Comput. Ed. Open* 9:100266.

Alderson DL (2022) Interactive computing for accelerated learning in computation and data science. *INFORMS Trans. Ed.* 22(2): 130–145.

Beliën J, De Boeck L, Goossens D, Quttineh NH, Vermeir J (2025) Game—The Game of Thrones quest for optimality: A role playing game for teaching the art of linear programming. *INFORMS Trans. Ed.* 25(3):250–256.

Chen X, Liang J (2024) Pair programming with ChatGPT. *SIGCSE 2024: Proc. 55th ACM Tech. Sympos. Comput. Sci. Ed.*, vol. 2 (Association for Computing Machinery, New York), 1600–1601.

Denny P, Kumar V, Giacaman N (2023) Conversing with copilot: Exploring prompt engineering for solving CS1 problems using natural language. *SIGCSE 2023: Proc. 54th ACM Tech. Sympos. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 1136–1142.

Denny P, Leinonen J, Prather J, Luxton-Reilly A, Amarouche T, Becker BA, Reeves BN (2024) Prompt problems: A new programming exercise for the generative AI era. *SIGCSE 2024: Proc. 55th ACM Tech. Sympos. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 296–302.

Dhumal P, Sundararaghavan PS, Nandkeolyar U (2008) "Cola-Game": An innovative approach to teaching inventory management in a supply chain. *Decision Sci. J. Innovative Ed.* 6(2):265–285.

Diamant A (2024) Introducing prescriptive and predictive analytics to MBA students with Microsoft Excel. *INFORMS Trans. Ed.* 24(2):152–174.

DiMichele N, Forbes M (2025) Puzzle—queen puzzles: An educational approach to integer programming techniques. *INFORMS Trans. Ed.* 26(3):247–253.

Drake MJ, Atkins RD (2021) An inquiry-based, partial information exercise for teaching inventory management. *Decision Sci. J. Innovative Ed.* 19(4):241–249.

Dwyer-Matzky K, Pachamanova D, Tilson V (2021) Accounting for capacity: A real-time optimization approach to managing observation unit utilization. *Naval Res. Logist.* 68(5):534–555.

Ellis ME, Casey KM, Hill G (2024) ChatGPT and Python programming homework. *Decision Sci. J. Innovative Ed.* 22(2):74–87.

Freire AP, Cardoso PCF, Salgado ADL (2023) May we consult ChatGPT in our human-computer interaction written exam? An experience report after a professor answered yes. Paiva R, de Araújo AMC, Falcão TP, Aguiar YPC, eds. *IHC'23: Proc. XXII Brazilian Sympos. Human Factors Comput. Systems* (Association for Computing Machinery, New York), 1–11.

Geng F, Shah A, Li H, Mulla N, Swanson S, Raj GS, Zingaro D, Porter L (2026) Exploring student-AI interactions in vibe coding. *Proc. 28th Australasian Comp. Educ. Conf. (*ACM, New York), 45–54.

Golrezaei N, Nazerzadeh H, Rusmevichientong P (2014) Real-time optimization of personalized assortments. *Management Sci.* 60(6):1532–1551.

Hamilton RE (2025) 7 tips for effective system prompting: A developer's guide to building better AI applications. Accessed April 8, 2025, https://circleci.com/blog/7-tips-for-effective-system-prompting/#c-consent-modal.

Harris M, Forbes M (2023) The snake eggs puzzle: Preparing students for Benders decomposition. *INFORMS Trans. Ed.* 23(3):210–217.

Iranpoor M (2021) Knights exchange puzzle—Teaching the efficiency of modeling. *INFORMS Trans. Ed.* 21(2):108–114.

Isken MW (2025) Spreadsheet modeling and wrangling with Python. *INFORMS Trans. Ed.* 25(2):152–168.

Kalyanasundaram B, Pruhs KR (2000) An optimal deterministic algorithm for online B-matching. *Theoret. Comput. Sci.* 233(1–2):319–325.

Kong N (2019) Active game-based learning of dynamics modeling and simulation in biomedical systems engineering. *INFORMS Trans. Ed.* 20(1):16–25.

Kress D, Dornseifer J (2015) LAVES: An extensible visualization tool to facilitate the process of learning and teaching algorithms. *INFORMS Trans. Ed.* 15(3):201–214.

Kusper G, Szabó C (2025) Vibe coding in education. *2025 Internat. Conf. Emerging eLearn. Tech. Appl. (ICETA)* (IEEE, Piscataway, NJ), 506–511.

Leinonen J, Denny P, MacNeil S, Sarsa S, Bernstein S, Kim J, Tran A, Hellas A (2023) Comparing code explanations created by students and large language models. *ITiCSE 2023: Proc. 2023 Conf. Innovation Tech. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 124–130.

Liu M, M'hiri F (2024) Beyond traditional teaching: Large language models as simulated teaching assistants in computer science. *SIGCSE 2024: Proc. 55th ACM Tech. Sympos. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 743–749.

Liu N, Van De Ven PM, Zhang B (2019) Managing appointment booking under customer choices. *Management Sci.* 65(9):4280–4298.

Liu Q, Zhang X, Liu Y, Lin L (2013) Spreadsheet inventory simulation and optimization models and their application in a national pharmacy chain. *INFORMS Trans. Ed.* 14(1):13–25.

MacNeil S, Tran A, Mogil D, Bernstein S, Ross E, Huang Z (2022) Generating diverse code explanations using the GPT-3 large language model. Vahrenhold J, Fisler K, Hauswirth M, Franklin D, eds. *ICER'22: Proc. 2022 ACM Conf. Internat. Comput. Ed. Res.*, vol. 2 (Association for Computing Machinery, New York), 37–39.

MacNeil S, Denny P, Tran A, Leinonen J, Bernstein S, Hellas A, Sarsa S, Kim J (2024) Decoding logic errors: A comparative study on bug detection by students and large language models. Herbert N, Seton C, eds. *ACE'24: Proc. 26th Australasian Comput. Ed. Conf.* (Association for Computing Machinery, New York), 11–18.

Mehta A (2013) Online matching and ad allocation. *Foundations Trends Theoret. Comput. Sci.* 8(4):265–368.

Mehta A, Saberi A, Vazirani U, Vazirani V (2007) Adwords and generalized online matching. *J. ACM* 54(5):22-es.

Meske C, Hermanns T, Von der Weiden E, Loser KU, Berger T (2025) Vibe coding as a reconfiguration of intent mediation in software development: Definition, implications, and research agenda. *IEEE Access* 13:213242–213259.

Murray S (2014) Why MBA students are learning to code. *GMAC* (July 19), https://www.businessbecause.com/news/mba-jobs/2675/why-mba-students-learn-to-code.

Nguyen H, Allan V (2024) Using GPT-4 to provide tiered, formative code feedback. *SIGCSE 2024: Proc. 55th ACM Tech. Sympos. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 958–964.

Pachamanova D, Tilson V, Dwyer-Matzky K (2022a) Case article—Machine learning, ethics, and change management: A data-driven approach to improving hospital observation unit operations. *INFORMS Trans. Ed.* 22(3):178–187.

Pachamanova D, Tilson V, Dwyer-Matzky K (2022b) Case—A data-driven approach to improving hospital observation unit operations. *INFORMS Trans. Ed.* 22(3):188–194.

Pădurean VA, Denny P, Gotovos A, Singla A (2025) Prompt programming: A platform for dialogue-based computational problem solving with generative AI models. *ITiCSE 2025: Proc. 30th ACM Conf. Innovation Tech. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 458–464.

Phillips R (2003) Teaching pricing and revenue optimization. *INFORMS Trans. Ed.* 4(1):1–10.

Prather J, Denny P, Leinonen J, Becker BA, Albluwi I, Craig M, Keuning H, et al. (2023) The robots are here: Navigating the generative AI revolution in computing education. *ITiCSE-WGR'23: Proc. 2023 Working Group Rep. Innovation Tech. Comput. Sci. Ed.* (Association for Computing Machinery, New York), 108–159.

Ray PP (2025) A review on vibe coding: Fundamentals, state-of-the-art, challenges and future directions. Preprint, submitted May 9, https://www.techrxiv.org/doi/full/10.36227/techrxiv.174681482.27435614/v1.

Ross EM (2023) Embracing artificial intelligence in the classroom. *Harvard Graduate School Ed.* (July 20), https://www.gse.harvard.edu/ideas/usable-knowledge/23/07/embracing-artificial-intelligence-classroom.

Seal KC, Leon LA, Przasnyski ZH, Lontok G (2020) Delivering business analytics competencies and skills: A supply side assessment. *INFORMS J. Appl. Anal.* 50(4):239–254.

Shumsky RA (2009) Case article—Introductory integrative cases on airline revenue management. *INFORMS Trans. Ed.* 9(3):135–139.

Stolyar AL, Tezcan T (2011) Shadow-routing based control of flexible multiserver pools in overload. *Oper. Res.* 59(6):1427–1444.

Taylor A, Vassar A, Renzella J, Pearce H (2024) dcc–help: Transforming the role of the compiler by generating context-aware error explanations with large language models. *SIGCSE 2024: Proc. 55th ACM Tech. Sympos. Comput. Sci. Ed.*, vol. 1 (Association for Computing Machinery, New York), 1314–1320.

Ting G, Hong Y (2024) An inquiry into AI-supported classroom transformation. *2024 6th Internat. Conf. Comput. Sci. Tech. Ed. (CSTE)* (IEEE, Piscataway, NJ), 114–118.

Van den Broeke M, Mlinar T (2025) Game—The competitive car production game: Teaching theory of constraints and linear programming. *INFORMS Trans. Ed.* 26(3):233–246.

Vliegen IMH, Zonderland ME (2017) Game—The BedGame—A classroom game based on real healthcare challenges. *INFORMS Trans. Ed.* 17(3):128–133.

Weathers D, Aragón O (2019) Integrating analytics into marketing curricula: Challenges and effective practices for developing six critical competencies. *Marketing Ed. Rev.* 29(4):266–282.

Xu JJ, Babaian T (2021) Artificial intelligence in business curriculum: The pedagogy and learning outcomes. *Internat. J. Management Ed.* 19(3):100550.

Downloaded from informs.org by [216.73.217.128] on 06 November 2026, at 19:26 . For personal use only, all rights reserved.