

Customizable Interactive Textbooks for Teaching Programming to Information Systems Students: Classroom Evidence and an AI-Era Roadmap

Ramadan Abdunabi
ramadan.abdunabi@colostate.edu
Department of Computer Information Systems
Colorado State University
Fort Collins, Colorado 80523, USA

Ilham AB Hbaci
hbaciilham@gmail.com
Mission Control Engineer, Fastly, Inc.
Fort Collins, Colorado 80528, USA

Ahmet Kurt
ahmet.kurt@etamu.edu
Department of Computer Science and Information Systems
East Texas A&M University, RELLIS Campus
Bryan, Texas 77807, USA

Sagar Bhatt
sagarbhatt.ons1087@gmail.com
GEICO Insurance
Parker, Colorado, 80134, USA

Abstract

A critical element of success for Information Systems (IS) students is the extent to which they practice programming, yet static textbooks offer little opportunity to write and run code while reading. Although adoption of interactive textbooks has grown, studies of their effectiveness in IS education remain limited. This study addresses that gap by comparing a traditional textbook with an interactive one across two consecutive offerings of a junior-level Java course taught by the same instructor with identical lectures, assignments, and assessments. One cohort of 33 students used a print textbook; the other, of 32, used zyBooks, an interactive textbook whose chapters the instructor customized to the course learning objectives. Findings indicate a substantial difference in engagement: 94% of the interactive cohort reported being engaged with the format compared with 24% of the print cohort, and 97% compared with 39% recommended its continued use. Students using the interactive textbook completed 95% of the practice activities embedded in their reading, while print-textbook students frequently reported never opening the book. Both cohorts improved significantly from pre-test to post-test, with no significant difference between them in learning gains or final course grades. Because the cohorts were consecutive and not randomized, these results are associational rather than causal. The study contributes classroom evidence supporting customizable interactive textbooks in IS programming education, together with a replicable implementation model for instructors. It also documents how the platform has since added features supported by artificial intelligence (AI), which motivate continued adoption and further research.

Keywords: Interactive Textbooks, Programming Education, Instructor Customization, Student Engagement, Information Systems Education, zyBooks.

Customizable Interactive Textbooks for Teaching Programming to Information Systems Students: Classroom Evidence and an AI-Era Roadmap

Ramadan Abdunabi, Ilham AB Hbaci, Ahmet Kurt, and Sagar Bhatt

1. INTRODUCTION

Introductory programming courses in Information Systems (IS) carry persistently high drop, fail, and withdraw rates, and the volume of deliberate practice a student accumulates is among the strongest predictors of who succeeds (Scherer et al., 2020; Zhang et al., 2013). Static print textbooks are poorly matched to that need: they describe code rather than execute it, give the instructor no visibility into whether reading was done, and return no feedback until an instructor grades the work.

Students report needing to see how code behaves rather than read prose about it (Rahman & Paudel, 2018), assigned-reading completion is low (Alhammad & Ku, 2019), and some instructors have stopped requiring a textbook at all (Gordon et al., 2021). Interactive textbooks embed executable code, animations, and auto-graded activities in the reading itself (Chaturvedi, 2023; Sosnovsky et al., 2025).

The practical problem is accountability. Allred and Murphy (2019) found students engaged only weakly with traditional textbook practice because the examples were neither graded nor trackable, and weak engagement raises drop, fail, and withdraw rates (Rahmani et al., 2024). An interactive textbook embeds practice in the reading, scores it automatically, and makes it visible to the instructor.

This paper reports what happened when one instructor replaced the required print textbook in a junior-level Java course with zyBooks, an interactive textbook customized to the course objectives (zyBooks, n.d.). Lectures, in-class exercises, assignments, and examinations were held constant across the two offerings, so the reading and practice medium was the only planned difference.

Because the platform itself changed over this period, we distinguish three kinds of feedback often conflated in the literature. Auto-grading reports whether a submission passes predefined test cases. Static hints are author-written explanations attached in advance to a particular wrong answer. Generative-AI feedback is composed by a large language model at request time. The Spring 2019 cohort had the first two only.

The study addresses four questions:

RQ1. Does replacing a traditional print textbook with a customizable interactive textbook change students' final course grades?

RQ2. Does it change students' self-reported engagement with assigned reading and practice, and their willingness to recommend the format?

RQ3. What do students report about the usefulness of, and their sense of connection to, each format?

RQ4. What patterns of practice and completion were observed among students using the interactive textbook?

Section 2 reviews four literatures the paper keeps separate. Section 3 describes the course, design, and measures. Section 4 documents the textbook as students experienced it, Section 5 the generative-AI features added later. Sections 6 through 8 report results, discuss them, and conclude.

The paper contributes an empirical classroom comparison with inferential tests and effect sizes; a pedagogical account of how interactive content, executable examples, automated feedback, and customization align to learning objectives; practical guidance for IS instructors; and a historically grounded view of an implementation evolving toward AI-supported learning.

2. LITERATURE REVIEW

The Joint ACM/AIS IS2020 task force reports that moving programming out of the IS core in IS2010 drew sustained criticism (Leidig & Salmela, 2020), and IS graduates often enter the workforce with lower programming competency than employers expect (Alwi & Karim, 2025; Leidig et al., 2019).

The task force therefore recommends restoring programming coverage sufficient for competency in a language such as Java (Leidig et al., 2019), which requires repeated application rather than exposure, and that is what interactive textbooks are built for (Pabst et al., 2023). Evidence on these platforms, however, has accumulated largely in Computer Science settings (Gordon et al., 2021); far less is known about IS students, who arrive with less programming background. That gap is what this study addresses.

2.1 Interactive Textbooks

Interactive learning is a hands-on technique used to introduce information and build knowledge. In programming it depends on the instructor demonstrating coding and the student writing code in and outside class. Immediate feedback lets students build knowledge incrementally (Krusche & Seitz, 2018) and gives instructors an accurate rather than inferred picture of performance (Goldhammer et al., 2017).

Interactive textbooks embed executable code, animations, and auto-graded questions in the reading. Students read, edit, and run programs in the browser without installing software (Ziatdinov & Valles, 2022). Compared with plain text, the format is associated with higher motivation and engagement and enables self-evaluation (Alhammad & Ku, 2019; Krusche & Seitz, 2018).

Edgcomb et al. (2015b) compared interactive and traditional textbooks across three universities and found significantly higher exam scores in the interactive condition, attributing part of the gain to covering the same topics with less expository prose (Edgcomb et al., 2015a). Spencer et al. (2020) report consistent associations with improved learning experience, while noting that most studies are quasi-experimental.

2.2 Timely Completion of Reading and Exercises

When practice is embedded, scored, and visible, students complete it earlier and more fully. Alshammari and Pivkina (2017) found from platform logs that students completing reading activities before the deadline outperformed those completing them late, with time management rather than activity volume carrying the association. Welter et al. (2025) report that assigning activities for points raised both completion and performance.

This mechanism is independent of how feedback is generated: it requires only that the activity be embedded, automatically scored, and reported to the instructor, all of which predate generative AI. Krusche and Seitz (2018) specify ten criteria for interactive assessment systems in their work on ArTEMiS; zyBooks satisfies these criteria, though it was not the system they evaluated.

Two instructor-controlled choices moderate the effect: whether activities carry points, and whether chapter sequence matches the course. Gordon et al. (2021) document zyBooks's shift toward customizable content, where instructors reorder, remove, and author sections. That capability is the one this study exercises.

2.3 Automated Feedback and Intelligent Tutoring Systems

Automated assessment predates AI by decades: auto-graders return pass or fail against test cases, while intelligent tutoring systems add a learner model to select the next problem or hint. Hattie and Timperley's (2007) synthesis establishes the principle these systems exploit, that feedback addressing task and process outperforms feedback reporting only a grade; theirs is not a study of any platform.

2.4 Generative-AI Feedback in Programming Education

Large language models add a further category: feedback composed at request time for the code submitted. Pankiewicz and Baker (2023) report increased persistence when GPT-generated feedback is

offered on programming assignments, and Alanazi et al. (2025) report positive effects in a meta-analysis confined to programming. Pooling 36 studies, Huang et al. (2026) estimate a moderate effect on achievement, larger in learner-centered settings, and Kaliisa et al. (2025) find AI feedback broadly comparable to human feedback. The evidence is mixed: Prather et al. (2024) document a widening gap in which generative AI helps some novices and harms others, and Fan et al. (2025) report learners offloading self-monitoring. Fecko et al. (2026) report a four-year comparison of a generative-AI textbook in engineering, the closest analogue to the study proposed in Section 7.6.

2.5 Theoretical Framing

The study is grounded in three theories, each explaining why an interactive textbook should change learning behavior (Figure 1). Social constructivism holds that understanding is built through interaction (Vygotsky, 1978); embedded activities and instructor visibility make that interaction routine. Information-processing theory explains how material is filtered, organized in working memory, and stored for recall (Driscoll & Burner, 2022), so a format requiring retrieval at the point of exposure should aid retention (Glaser & Richter, 2025). Self-efficacy theory holds that beliefs about capability shape persistence (Bandura, 1977); first-time programmers who run, modify, and retry code without penalty accumulate the mastery experiences those beliefs rest on.

Together they predict the pattern we test. Because they act on whether practice happens and how capable a student feels doing it, they predict a large difference in engagement; because a post-test and a semester grade aggregate many inputs, they predict a smaller one in performance (Chen, 2025; Sweller et al., 1998).

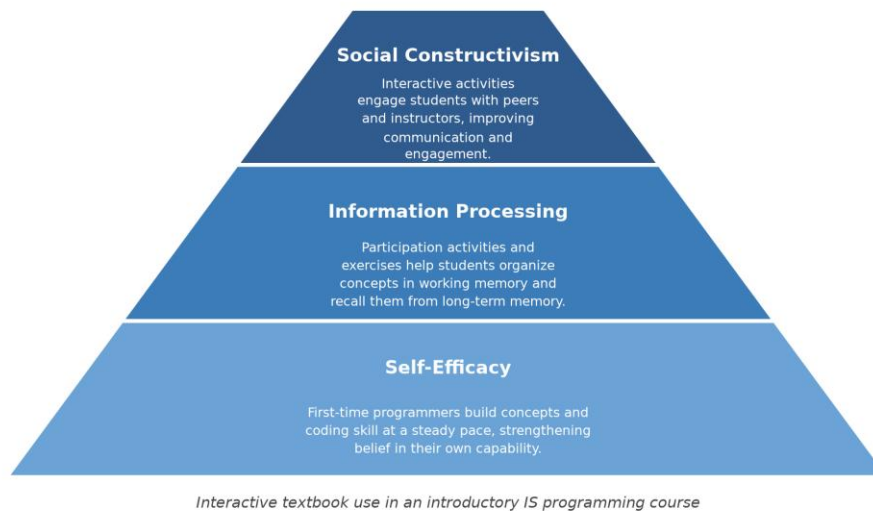


Figure 1. Theoretical framework supporting interactive textbook use.

3. COURSE CONTEXT AND STUDY DESIGN

3.1 Course, Participants, and Materials

The course is a junior-level, 16-week course meeting twice weekly for 75 minutes, required for Computer Information Systems majors. Sixty-five students participated: the Fall 2018 cohort ($n=33$) used the print textbook by Liang (2017), and the Spring 2019 cohort ($n=32$) used zyBooks, customized by the instructor before the semester.

Prior programming competency was surveyed at the start of both semesters; students rated themselves as having no experience, novice, intermediate, or expert ability (Allred & Murphy, 2019). Twenty-nine students in each cohort responded, and none reported expert ability. The distributions did not differ (Freeman-Halton exact $p=.410$, $V=.20$); Table 1 reports the counts. The cohorts did differ on gender,

24 male and 9 female against 31 male and 1 female (Fisher's exact $p=.013$), a non-equivalence addressed in Section 7.5.

	Textbook Group (n = 29)	zyBooks Group (n = 29)
No Experience Have not written code using Java	1	2
Novice Able to write a few lines of Java code	18	22
Intermediate Able to read and write basic lines of coding	10	5

Note: Four students in the textbook cohort and three in the interactive cohort did not complete the programming competency survey.

Table 1. Students' programming competency categories.

3.2 Design and Procedure

The study uses a convergent parallel mixed-method design. Course performance and survey-reported engagement are compared across two consecutive offerings of the same course, taught by the same instructor with the same lectures, exercises, assignments, and examinations, and open-ended responses are analyzed alongside them. The only planned difference was the reading and practice medium; students were not randomly assigned.

Both cohorts (N=65) completed reading and practice assignments across the same 8 topics over 16 weeks. Appendix C shows those topics in the zyBooks dashboard.

Students were required to read and practice before each new topic was introduced. The traditional cohort completed end-of-chapter exercises from the print text; the interactive cohort worked through zyBooks sections carrying less expository prose and more executable code, animations, and auto-graded participation and challenge activities.

Chapters were customized before the semester to align with the course objectives and to mirror the sequence of the Liang text, so both cohorts met the same topics in the same order. Section 4.3 documents that workflow.

After the instructor presented each topic using animated slides, each student completed an in-class exercise via Canvas for 3 points; absent students and non-submissions received zeros.

At semester end, students completed a Qualtrics survey including demographic data, two closed-ended engagement questions, and 11 open-ended textbook experience questions (see Appendix A).

Four additional 5-point Likert items measured the quality of zyBooks's essay questions (Appendix B), written following Reiner et al. (2002). Participation and completion data for the interactive cohort were retrieved from the zyBooks instructor dashboard.

3.3 Measures and Human-Subjects Protections

Three families of measure are reported. Performance is measured by a pre-test at the start of the semester and a post-test at the end, both delivered through the learning management system, and by the final course grade. Engagement is measured by two closed-ended self-report items and, for the interactive cohort only, by platform log data on completion. Perception is measured by 11 open-ended items analyzed thematically. Analysis Ns vary slightly because not every student completed every instrument.

Survey participation was voluntary and anonymous. Responses were collected in Qualtrics without identifiers, and grade data were de-identified before analysis. The protocol was reviewed by the Colorado State University Institutional Review Board and determined exempt (Protocol 045-19H).

3.4 Threats to Validity Carried Through the Analysis

Because the cohorts were consecutive and not randomized, three threats are carried through the interpretation: history, since events between the two semesters may differ; maturation of instruction, since the instructor taught the course a second time; and selection, since students chose their semester and the cohorts differ in gender composition. We therefore treat the comparison as evidence of association rather than causal effect.

4. THE INTERACTIVE TEXTBOOK AS IMPLEMENTED IN THE COURSE

4.1 Chapter Structure and Activity Types

The Spring 2019 cohort subscribed for approximately \$46 and accessed reading through a browser. Each topic opened with a statement of what students needed to understand (Figures 2a and 2b), followed by participation and challenge activities (Figures 3 and 4). Participation activities are short questions embedded in the exposition; challenge activities are longer, auto-graded problems requiring the student to write and run code in the embedded editor (Figure 5).

1.1 Programming (general) Present Note

Computer program basics

Computer programs are abundant in many people's lives today, carrying out applications on smartphones, tablets, and laptops, powering businesses like Amazon and Netflix, helping cars drive and planes fly, and much more.

A computer **program** consists of instructions executing one at a time. Basic instruction types are:

- **Input:** A program gets data, perhaps from a file, keyboard, touchscreen, network, etc.
- **Process:** A program performs computations on that data, such as adding two values like $x + y$.
- **Output:** A program puts that data somewhere, such as to a file, screen, network, etc.

Programs use **variables** to refer to data, like x , y , and z below. The name is due to a variable's value varying as a program assigns a variable like x with new values.

Figure 2a. Introducing a topic in zyBooks.

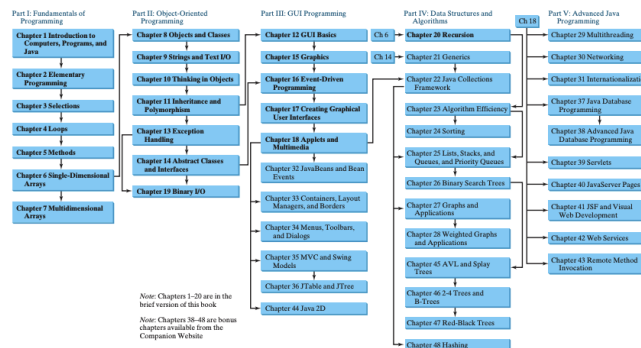


Figure 2b. Structure of Liang's Introduction to Java Programming and Data Structures.

PARTICIPATION ACTIVITY 1.1.1: A basic computer program.

☐ 2x speed

Computer program

```

x = Get next input
y = Get next input
z = x + y
Put z to output
            
```

Input (keyboard)

x: 2
y: 5

Output (screen)

7

Captions ^

1. A basic computer program's instructions get input, process, and put output. This program first assigns x with what is typed on the keyboard input, in this case 2.
2. The program's next instruction gets the next input, in this case 5.
3. The program then does some processing, in this case assigning z with $x + y$ (so $2 + 5$ yields z of 7).
4. Finally, the program puts z (7) to output, in this case to a screen.

Figure 3. Participation activity example in zyBooks.

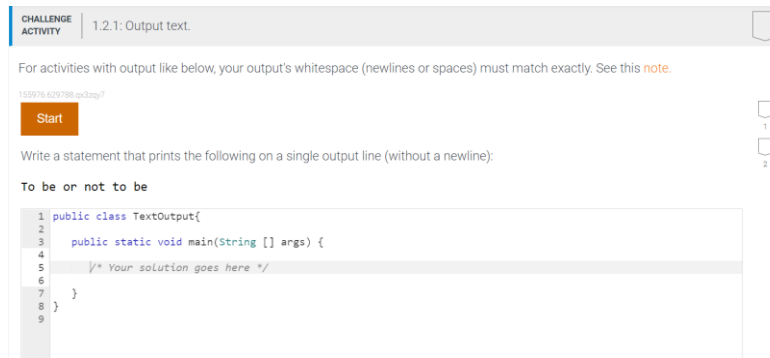


Figure 4. Challenge activity example in zyBooks.

Below is the zyBooks Development Environment (zyDE), a web-based programming practice environment. Click run to compile and execute the program, then observe the output. Change 20 to a different number like 35 and click run again to see the different output.

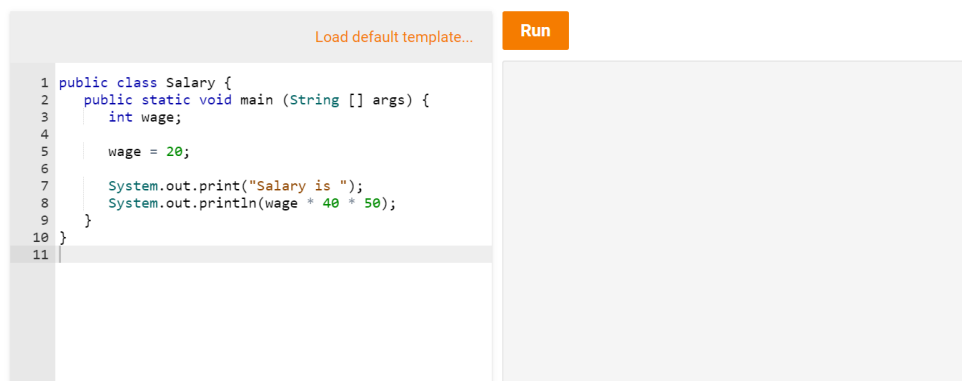


Figure 5. Editing and executing area.

Embedded question types included short answer, true/false, matching, and multiple choice. An incorrect answer triggered an immediate explanation of why it was wrong (Figure 6). These explanations are static: authored in advance and attached to a specific wrong answer. No generative-AI feedback existed in the platform at this time.

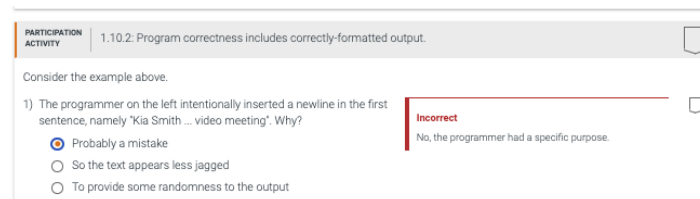


Figure 6. Feedback errors from zyBooks.

The capability with no counterpart in print is instructor visibility: the dashboard reports participation and completion for every student and activity (Figure 7), showing before class who has prepared. Table 2 maps each activity type to the objective it serves and how it is observed.

Activity type	Intended objective	What the student does	Observable measure
Worked example with animation	Build a schema for syntax and execution order	Steps through the animation, edits and reruns the example	Section marked complete
Participation activity	Force retrieval at the point of exposure	Answers a short embedded question, sees why an answer is wrong	Completion rate per activity
Challenge activity (zyLab)	Apply the concept to an unseen problem	Writes and runs a complete program against test cases, resubmits	Attempts and completion
Instructor-authored section	Align content and sequence to course objectives	Reads objectives, studies a UML diagram, answers design questions	Dashboard participation

Table 2. Course activities aligned to intended learning objectives.

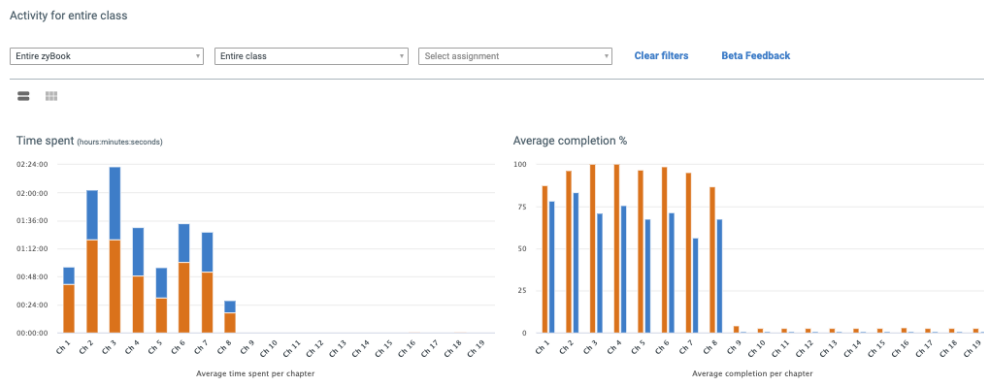


Figure 7. Class participation statistics from the zyBooks dashboard.

4.2 A Worked Example: One Activity End to End

A challenge activity proceeds as follows. The prompt states the task and fixes the output format: read the radius of a circular pizza, compute its area, and print it to two decimal places, the type of task students rated in Appendix B. The student writes a complete program in the embedded editor (Figure 5) and presses Run, and the platform runs it against instructor-defined test cases.

If a test fails, the student sees the failing case, the input, the expected output, and their program's actual output side by side. That is the whole of the feedback: a difference report, not a diagnosis. A student whose program prints 78.5 where 78.54 was expected must work out independently that the formatting specifier is at fault.

Two properties of this loop matter for the results in Section 6. The feedback is immediate but deterministic, leaving the diagnostic work with the student, and every submission is logged, so the instructor sees not only who finished but how many attempts it took.

Line-by-line animation suits simple code but does not by itself teach decomposition or design, and the customization described next is partly a response to that limit. The instructor-authored sections place a UML diagram and conceptual questions before the coding task (Figure 8h) and use longer zyLab assignments spanning multiple classes. Where the design burden is heaviest the course still relies on lecture and instructor contact.

4.3 Customizing Chapters to Course Objectives

Customization is what made the interactive textbook a replacement for the print text rather than a parallel resource. zyBooks lets an instructor reorder stock sections, remove them, and author new ones

inside a chapter (Gordon et al., 2021). Customizing a full chapter takes about two hours; later semesters require only incremental edits.

Step 1. Create a custom section. The instructor selects the chapter and clicks Create Section (Figure 8a), embedding new material inside the familiar structure (Cai et al., 2025).

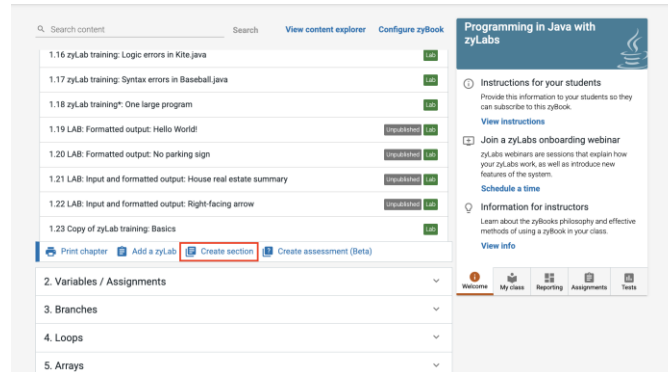


Figure 8a. Creating a custom zyBooks section.

Step 2. Assign a title. zyBooks numbers the section by position (Figure 8b). Explicit headings help learners build schemas before technical content (Ausubel, 1968; Elfeky et al., 2024).

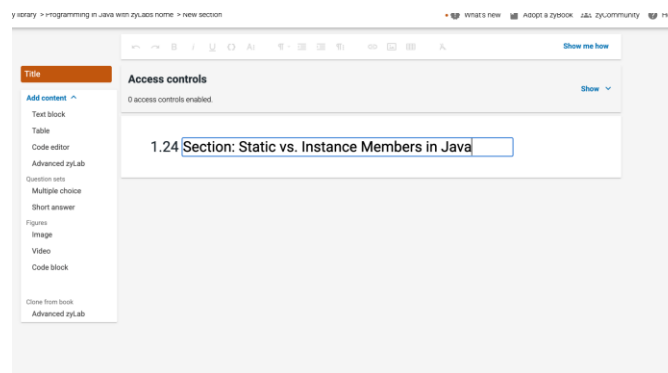


Figure 8b. Section title.

Step 3. Choose content types. The menu (Figure 8c) offers text blocks, tables, code editors, participation activities, images, and labs, pairing text and graphics as multimedia learning theory recommends (Mayer, 2024).

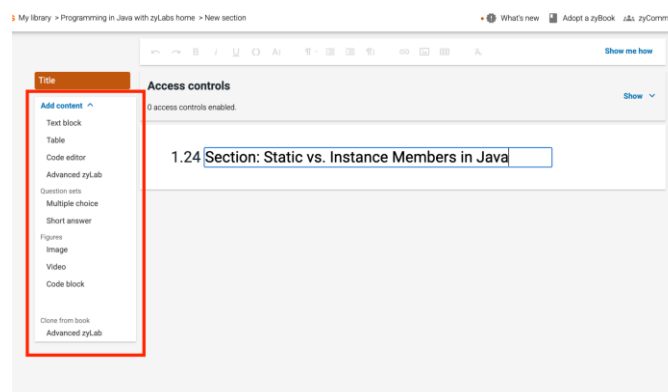


Figure 8c. Content menu.

Step 4. State learning objectives. A text block opens the section with explicit objectives (Figure 8d), supporting goal-directed attention (Guntur & Purnomo, 2024; Zimmerman, 2002).

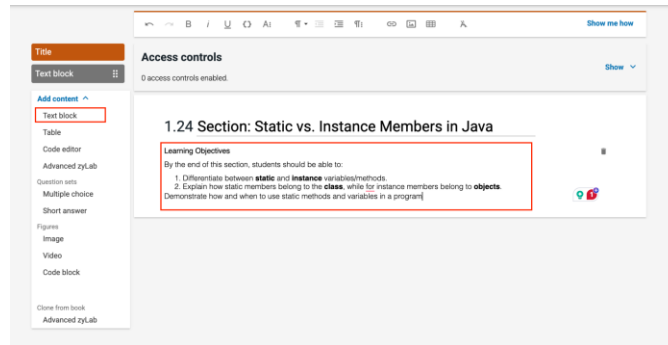


Figure 8d. Text block stating learning objectives.

Step 5. Add a concept overview table. A table (Figure 8e) organizes key terms and syntax rules for comparative processing (Ponce et al., 2025).

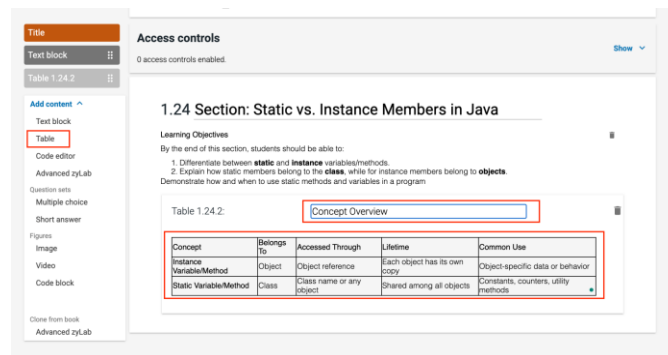


Figure 8e. Concept overview table.

Step 6. Embed a code illustration. The code editor (Figure 8f) carries an executable annotated example, applying the worked-example effect (Chen, 2025; Sweller et al., 1998).

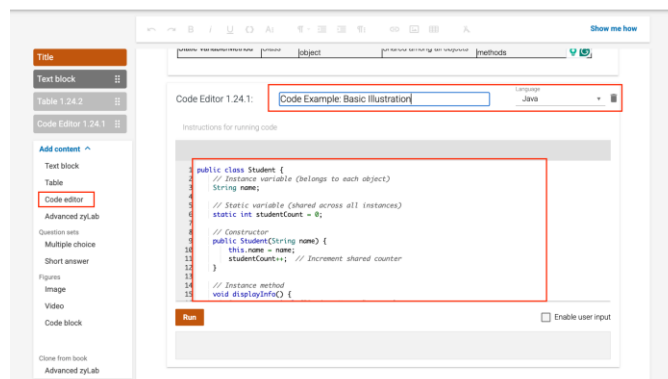


Figure 8f. Code editor with an executable example.

Step 7. Insert participation activities. Questions follow the example (Figure 8g) to force retrieval rather than re-reading (Glaser & Richter, 2025).

Figure 8g. Participation questions.

Step 8. Add a UML diagram. A class diagram with conceptual questions (Figure 8h) supports reasoning about class hierarchies before code is written (Cipriano et al., 2024).

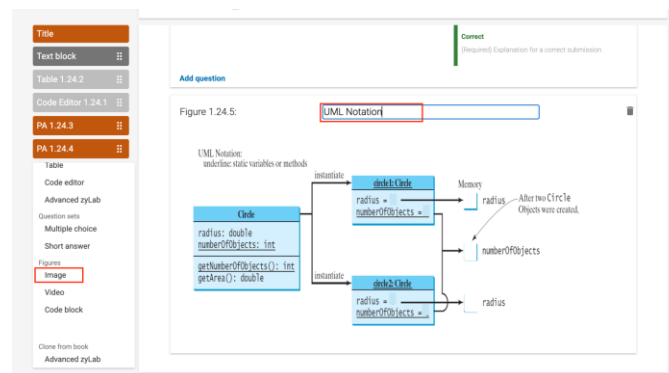


Figure 8h. UML class diagram.

Step 9. Add a challenge activity via zyLab. The advanced zyLab option (Figure 8i) carries an open-ended assignment with instructor-defined test cases, introducing desirable difficulty once scaffolding is in place (Lee & Lee, 2024; Sinha & Kapur, 2021).

Figure 8i. zyLab challenge activity.

5. PLATFORM EVOLUTION: AI-SUPPORTED FEEDBACK IN ZYBOOKS

The capabilities in this section were available to neither cohort and influenced no result in Section 6. zyBooks introduced AI Hints for zyLabs in 2025; the feature remains in beta, is off by default, and is controlled per lab (zyBooks, 2026). All screenshots here are from the instructor's live course, not the study period.

Step 1. Enable the feature. The instructor checks the zyLabs AI Hints box in lab settings (Figure 9a). Because hints are off by default and set per assignment, students can be required to attempt a problem unaided first.

2.28 LAB: Using math methods ⚙️ ^

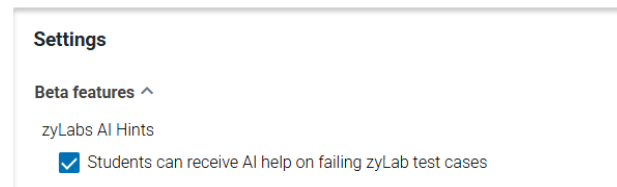


Figure 9a. Enabling AI Hints in zyLab settings. Current platform functionality, introduced after the 2019 study.

Step 2. Submission and response. A student whose code fails the test cases can request a hint rather than seeing only the difference report of Section 4.2 (Figure 9b). The hint responds to the student's own code with questions and directional nudges rather than the answer (Figure 9c), and students cannot compose their own prompts, a guardrail aimed at preserving productive struggle (zyBooks, 2026). Grading remains deterministic. The vendor also presents hints as building skill at working with AI, a claim that sits awkwardly with the no-prompt restriction and that we treat as untested.

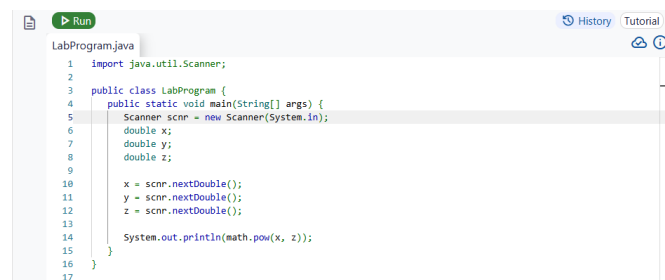


Figure 9b. A submission with AI Hints available. Current functionality; not part of the intervention evaluated here.

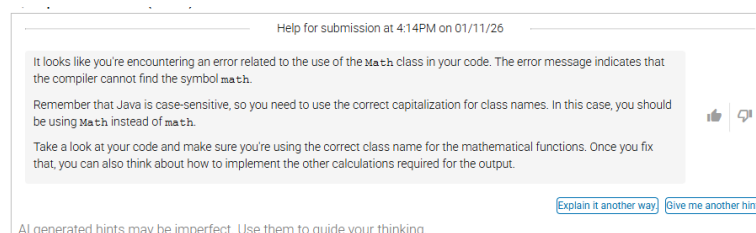


Figure 9c. An AI hint returned to a student. Captured from the instructor's live course in 2026 to illustrate the platform's subsequent evolution; not observed in the 2019 study.

A concrete example shows what changes. A novice writes a Java loop whose condition uses `<=` where `<` is required, so it runs one iteration too many. The deterministic report shows only that expected and actual output differ; a hint can instead point to the loop condition and ask what happens on the final iteration, without supplying the correction.

zyBooks has also added AI-literacy content to its programming titles: the Java zyBook used in this course now includes chapters in which students call AI APIs and assemble applications (zyBooks, 2025). Jiang et al. (2023) and Sosnovsky et al. (2025) survey this shift across intelligent textbooks. The vendor distinguishes business students, who need to use AI tools well, from computing students, who need to integrate them into applications, a line that IS programs straddle.

6. RESULTS

6.1 Quantitative Results

A mixed ANOVA with test (pre, post) within-subjects and cohort between-subjects was run on students with complete data, supplemented by paired and independent t-tests; final grades used an independent-samples t-test and dichotomous items used Pearson's chi-square (reported with Yates's continuity correction; phi is computed from the uncorrected statistic) with Fisher's exact confirmation. Levene's test supported homogeneity throughout (final grade $p=.854$), and pre- and post-test scores were normal in both cohorts (Shapiro-Wilk $p>.08$). Final grades departed from normality in the textbook cohort ($p=.006$), so that comparison is read with caution.

Scores improved substantially from pre-test to post-test, $F(1,61)=157.93$, $p<.001$, partial eta-squared $=.72$, and each cohort improved separately (textbook $t(31)=-8.27$, $dz=1.46$, $n=32$; interactive $t(30)=-9.89$, $dz=1.78$, $n=31$; both $p<.001$). The test-by-cohort interaction was negligible, $F(1,61)=0.13$, $p=.722$, partial eta-squared $=.002$, as was the between-cohort effect, $F(1,61)=1.64$, $p=.205$. The cohorts did not differ on the pre-test, $t(61)=-1.30$, $p=.198$, or the post-test, $t(61)=-0.89$, $p=.380$, $d=0.22$. Final course grades were slightly higher in the interactive cohort ($M=84.42$, $SD=10.00$, $n=31$) than the textbook cohort ($M=82.35$, $SD=9.66$, $n=32$), but non-significantly, $t(61)=0.84$, $p=.407$, $d=0.21$, 95% CI $[-0.28, 0.71]$. No performance measure separated the cohorts; the smallest detectable effect is $d=0.72$.

The engagement items produced large differences. Twenty-nine of 31 interactive-cohort respondents (94%) reported being engaged with the format against 8 of 33 (24%), chi-square(1, $N=64$) $=28.70$, $p<.001$, $\phi=.70$, $OR=45.3$ [8.8, 233.4]. Thirty of 31 (97%) would recommend continuing with the format against 13 of 33 (39%), chi-square(1, $N=64$) $=21.34$, $p<.001$, $\phi=.61$, $OR=46.2$ [5.6, 381.2]. Fisher's exact tests agreed ($p<.001$). Tables 3 and 4 report the details.

Survey Item	Response	Traditional Textbook (n = 33)	Interactive Textbook (n = 31)
Engaged in using this format of books	Yes	8 (24%)	29 (94%)
	No	25 (76%)	2 (6%)
Recommend the instructor to continue using this format in the future	Yes	13 (39%)	30 (97%)
	No	20 (61%)	1 (3%)

Table 3. Engagement evaluation.

Platform logs for the interactive cohort ($n=31$) recorded 1,513 completed activities: 1,249 participation (82.5%) and 264 challenge (17.5%). Mean completion was 95% for participation and 73% for challenge

activities (Figure 10). Students completed nearly all low-stakes embedded practice but stopped short on longer tasks. No comparable data exist for the textbook cohort.

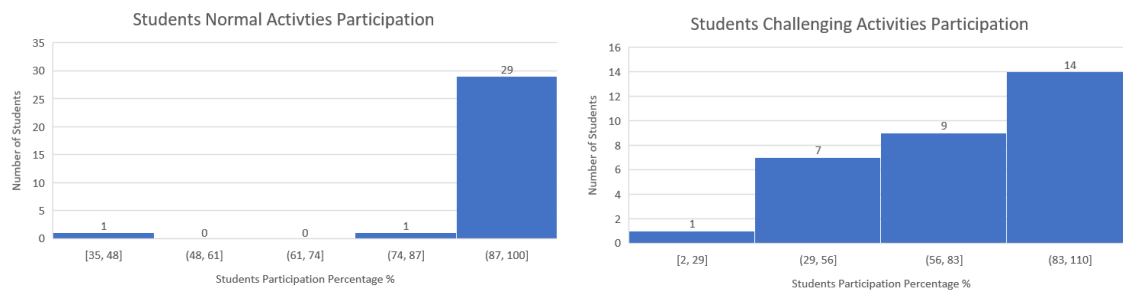


Figure 10. Student participation data.

Comparison	Traditional	Interactive	Test	p	Effect size
Pre-test to post-test (within)	n = 32	n = 31	$F(1,61) = 157.93$	<.001	partial eta-sq = .72
Test x cohort interaction	-	-	$F(1,61) = 0.13$	.722	partial eta-sq = .002
Cohort (between-subjects)	n = 32	n = 31	$F(1,61) = 1.64$	.205	partial eta-sq = .03
Post-test	n = 32	n = 31	$t(61) = -0.89$	.380	d = 0.22
Final course grade, M (SD)	82.35 (9.66)	84.42 (10.00)	$t(61) = 0.84$	.407	d = 0.21 [-0.28, 0.71]
Engaged with the format	8/33 (24%)	29/31 (94%)	chi-square(1) = 28.70	<.001	phi = .70; OR = 45.3
Would recommend the format	13/33 (39%)	30/31 (97%)	chi-square(1) = 21.34	<.001	phi = .61; OR = 46.2
Prior competency (3 categories)	n = 29	n = 29	Freeman-Halton exact	.410	V = .20
Gender (male/female)	24/9	31/1	Fisher's exact	.013	phi = .33

Table 4. Summary of inferential results.

Convergent validity was run for the four items measuring zyBooks's essay question quality. One item was dropped for a Pearson correlation below .30 (Robinson, 2018); for the remaining three, Cronbach's alpha was acceptable for so few items ($\alpha = .73$) and corrected item-total correlations exceeded .3 (Field, 2024). The items scored a mean of 3.84 ($SD = .77$) on a 5-point scale ($n = 31$).

6.2 Qualitative Results

Qualitative analysis proceeded by reading survey responses, pattern coding, and developing themes. Both groups agreed that the best way to learn programming is through doing and hands-on practice.

Both groups also agreed that the textbook, traditional or interactive, complements help from a teaching assistant rather than removing the need for it. One student noted, "I think a TA can help offer students a different approach when tackling new concepts. A textbook only has one way of providing students with information which may not resonate with everybody."

Three themes distinguished the two formats (Appendix D). On usefulness, interactive-textbook students found the material more applicable, valuing immediate feedback and progress tracking.

On practice facility, the interactive textbook offered easier access and more varied exercises. One student noted, "zyBooks was 10/10 when it came to completing the programming assignments because the practice code, we had to in zyBooks could be directly correlated to the code we needed to write for the programming assignments."

On connection, students using zyBooks reported higher motivation, while print-textbook students often described a disconnect from the material and several reported not opening it at all. One student shared, "Yes, it gave me more opportunities to program outside of class."

7. DISCUSSION

7.1 Interpreting the Pattern of Results

The results form a consistent pattern: a large difference in engagement and in willingness to continue with the format, and no detectable difference on any performance measure. Both cohorts learned substantially; neither format learned more. This is what Section 2.5 predicted: reading format acts on whether practice happens, while a post-test and a semester grade aggregate many inputs.

The cohorts were comparable on prior programming experience (Freeman-Halton exact $p=.410$) and did not differ on the pre-test, which removes the most obvious alternative explanation for the engagement difference. They were not comparable on gender, and we cannot rule out a contribution from that.

Self-reported engagement is corroborated behaviorally. Goldhammer et al. (2017) and Hoch et al. (2018) argue that engagement in digital environments can be assessed from interaction logs, and Fredricks and McColskey (2012) treat effort and participation as core indicators. The completion rates in Figure 10 are evidence of that kind, independent of survey self-report.

The qualitative themes align with the quantitative pattern. Both cohorts agreed that programming is learned by doing and regarded the textbook as complementary to instructor and teaching assistant support. Students using the print textbook frequently reported not using it at all, which is the practical core of the finding: a textbook's design is irrelevant if the book is not opened.

7.2 Why the Classroom Evidence Remains Relevant

Three reasons make the classroom data relevant now that the platform has changed. First, the comparison isolates the format: because generative AI did not exist in the platform, the difference reported here belongs to the interactive, customizable format, and as AI becomes default this comparison can no longer be run. Second, the mechanisms are platform-stable: embedded retrieval practice, worked examples, trackable completion, and dashboard visibility belong to the format, not the feedback engine. Third, it gives adopters a floor against which AI claims can be measured.

One boundary on all of this deserves stating plainly: the present study supports the interactive-textbook implementation; it does not experimentally establish the effectiveness of zyBooks's newer generative-AI features.

7.3 Supporting Learner Variability

Pedagogy must also accommodate differences among learners, with one qualification. The meshing hypothesis, that instruction matched to a self-reported learning style improves outcomes, has not held up under experimental test (Pashler et al., 2008), and we do not claim the platform works by matching styles. What the format offers is multiple simultaneous representations under learner control: prose, an animated execution trace, an editable runnable example, retrieval questions, and an open-ended coding task. Self-pacing is the operative variable.

7.4 Implications for IS Educators

For IS programs restoring programming coverage (Leidig & Salmela, 2020), the actionable findings are narrow. Assigning interactive activities for points converts optional practice into completed practice. Dashboards make preparation visible before class rather than after the first examination (Dai et al., 2025; Fiallos-Quinteros et al., 2025). Customization, at about two hours per chapter, let the interactive text replace the print text.

7.5 Limitations

Assignment was not random; students selected their semester. The cohorts differ significantly in gender composition (Fisher's exact $p=.013$), and unmeasured differences may remain. Because the offerings were consecutive, history effects and instructor maturation are confounded with the textbook change. Final grades departed from normality in the textbook cohort. Engagement is self-reported on two single-item measures, and log data exist for one cohort only. The sample is 65 students in one course at one institution, and the smallest detectable effect is $d=0.72$. The imbalance also limits what can be said about how these tools serve women in computing (Master et al., 2025). The study nonetheless supplies classroom evidence, implementation detail, and a baseline that can no longer be collected.

7.6 Future Work

The design that follows is a within-semester comparison with AI hints enabled for a randomly assigned half of the challenge activities and disabled for the other half, holding cohort, instructor, and semester constant, ideally across multiple IS courses. Planned measures include log variables tied to Section 2.5: submissions per activity, time to first correct submission, hint requests, and persistence after failure. Because generative AI helps some novices and harms others, the analysis should be pre-specified by prior competency (Prather et al., 2024).

8. CONCLUSION

Replacing a print textbook with a customizable interactive textbook produced a large difference in how students engaged with assigned reading and practice, and no detectable difference in measured performance. Ninety-four percent of interactive-cohort respondents reported being engaged with the format against 24% using print, and 97% against 39% would recommend its continued use. Both cohorts learned substantially over the semester.

The mechanism we can support is structural rather than technological. Practice embedded in the reading, scored automatically, and reported to the instructor gets done; practice deferred to ungraded end-of-chapter exercises frequently does not. Instructor customization made the substitution viable, letting the interactive text follow the course's sequence rather than requiring the course to follow the book's.

These findings predate generative AI. The zyBooks capabilities documented in Section 5 were introduced after the study period and were used by no student in this study. We present them as the reason the question is worth revisiting, not as an explanation of these results.

For IS educators, the practical conclusion is that the interactive, customizable format is worth adopting on the engagement evidence alone, and that claims about what AI adds should be evaluated against a baseline like this one rather than against a print textbook. Students in both cohorts described the textbook as complementing instructor and teaching assistant support rather than replacing it, and the same standard should apply to AI-supported feedback: it extends scalable practice, it does not substitute for human instruction.

9. REFERENCES

- Alanazi, M., Soh, B., Samra, H., & Li, A. (2025). The influence of artificial intelligence tools on learning outcomes in computer programming: A systematic review and meta-analysis. *Computers*, 14(5), Article 185. <https://doi.org/10.3390/computers14050185>
- Alhammad, R., & Ku, H. Y. (2019). Graduate students' perspectives on using e-books for academic learning. *Educational Media International*, 56(1), 75-91. <https://doi.org/10.1080/09523987.2019.1583460>
- Allred, J. B., & Murphy, C. A. (2019). Interactive electronic textbook use in higher education: Grades, engagement, and student perceptions. *International Journal of Innovation and Learning*, 25(3), 296-309. <https://doi.org/10.1504/IJIL.2019.098895>

- Alshammari, K., & Pivkina, I. V. (2017, October). Relationship between time management in courses with online interactive textbooks and students' performance. In *2017 IEEE Frontiers in Education Conference (FIE)* (pp. 1-5). IEEE. <https://doi.org/10.1109/FIE.2017.8190620>
- Alwi, S. K. K., & Karim, A. A. (2025). Bridging the employability gap: Aligning higher education curricula with industry-required competencies in the digital era. *Journal of Business Strategies*, 18(2). <https://greenwichjournals.com/index.php/businessstudies/article/view/768>
- Ausubel, D. P. (1968). *Educational psychology: A cognitive view*. Holt, Rinehart and Winston.
- Bandura, A. (1977). Self-efficacy: Toward a unifying theory of behavioral change. *Psychological Review*, 84(2), 191-215. <https://doi.org/10.1037/0033-295X.84.2.191>
- Cai, L., Msafiri, M. M., & Kangwa, D. (2025). Exploring the impact of integrating AI tools in higher education using the Zone of Proximal Development. *Education and Information Technologies*, 30, 7191-7264. <https://doi.org/10.1007/s10639-024-13112-0>
- Chaturvedi, R. (2023). Improved student learning experience in large programming classes using pseudo-flipped method. In *Computer Science & Information Technology (CS & IT)* (Vol. 13, pp. 101-110). AIRCC Publishing Corporation. <https://doi.org/10.5121/csit.2023.131209>
- Chen, C.-Y. (2025). Effects of worked examples with explanation types and learner motivation on cognitive load and programming problem-solving performance. *ACM Transactions on Computing Education*, 25. <https://doi.org/10.1145/3732791>
- Cipriano, B. P., Alves, P., & Denny, P. (2024). A picture is worth a thousand words: Exploring diagram and video-based OOP exercises to counter LLM over-reliance. In *Technology enhanced learning for inclusive and equitable quality education: 19th European Conference on Technology Enhanced Learning (EC-TEL 2024)* (pp. 75-89). Springer. https://doi.org/10.1007/978-3-031-72315-5_6
- Dai, W., Lin, J., Jin, F., Tsai, Y.-S., Srivastava, N., Le Bodic, P., Gašević, D., & Chen, G. (2025). Learning analytics for early identification of at-risk students and feedback intervention. *Journal of Learning Analytics*, 12(3), 102-125. <https://doi.org/10.18608/jla.2025.8735>
- Driscoll, M. P., & Burner, K. J. (2022). *Psychology of learning for instruction* (4th ed.). Pearson.
- Edgcomb, A., Vahid, F., & Lysecky, R. (2015a, October). Students learn more with less text that covers the same core topics. In *2015 IEEE Frontiers in Education Conference (FIE)* (pp. 1-5). IEEE. <https://doi.org/10.1109/FIE.2015.7344322>
- Edgcomb, A. D., Vahid, F., Lysecky, R., Knoesen, A., Amirtharajah, R., & Dorf, M. L. (2015b, June). Student performance improvement using interactive textbooks: A three-university cross-semester analysis. In *2015 ASEE Annual Conference & Exposition*. ASEE. <https://doi.org/10.18260/p.24760>
- Elfeky, A. I. M., Najmi, A. H., & Elbyaly, M. Y. H. (2024). The impact of advance organizers in virtual classrooms on the development of integrated science process skills. *PeerJ Computer Science*, 10, Article e1989. <https://doi.org/10.7717/peerj-cs.1989>
- Fan, Y., Tang, L., Le, H., Shen, K., Tan, S., Zhao, Y., Shen, Y., Li, X., & Gašević, D. (2025). Beware of metacognitive laziness: Effects of generative artificial intelligence on learning motivation, processes, and performance. *British Journal of Educational Technology*, 56(2), 489-530. <https://doi.org/10.1111/bjet.13544>
- Fecko, B., Dziak, J., Vince, T., & Molnár, J. (2026). Generative AI interactive textbook in electrotechnics: A four-year comparative study on student performance and inclusion. *Computers and Education: Artificial Intelligence*, 10, Article 100611. <https://doi.org/10.1016/j.caeai.2026.100611>

- Fiallos-Quinteros, J. C., Restrepo-Carmona, J. A., Guzman-Luna, J. A., & Jimenez-Builes, J. A. (2025). Using a learning analytics model to predict at-risk students in an introductory software programming course. In *Emerging technologies in learning* (pp. 379-394). Springer. https://doi.org/10.1007/978-3-031-80388-8_20
- Field, A. (2024). *Discovering statistics using IBM SPSS statistics* (6th ed.). Sage.
- Fredricks, J. A., & McColskey, W. (2012). The measurement of student engagement: A comparative analysis of various methods and student self-report instruments. In S. L. Christenson, A. L. Reschly, & C. Wylie (Eds.), *Handbook of research on student engagement* (pp. 763-782). Springer. https://doi.org/10.1007/978-1-4614-2018-7_37
- Glaser, J., & Richter, T. (2025). The testing effect in the lecture hall: Does it transfer to content studied but not practiced? *Teaching of Psychology*, 52(1), 17-26. <https://doi.org/10.1177/00986283231218943>
- Goldhammer, F., Naumann, J., Rölke, H., Stelter, A., & Tóth, K. (2017). Relating product data to process data from computer-based competency assessment. In D. Leutner, J. Fleischer, J. Grünkorn, & E. Klieme (Eds.), *Competence assessment in education* (pp. 407-425). Springer. https://doi.org/10.1007/978-3-319-50030-0_24
- Gordon, C., Lysecky, R., & Vahid, F. (2021, October). The shift from static college textbooks to customizable content: A case study at zyBooks. In *2021 IEEE Frontiers in Education Conference (FIE)* (pp. 1-7). IEEE. <https://doi.org/10.1109/FIE49875.2021.9637289>
- Guntur, M., & Purnomo, Y. W. (2024). A meta-analysis of self-regulated learning interventions studies on learning outcomes in online and blended environments. *Online Learning*, 28(3). <https://doi.org/10.24059/olj.v28i3.4025>
- Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research*, 77(1), 81-112. <https://doi.org/10.3102/003465430298487>
- Hoch, S., Reinhold, F., Werner, B., Richter-Gebert, J., & Reiss, K. (2018). Design and research potential of interactive textbooks: The case of fractions. *ZDM*, 50(5), 839-848. <https://doi.org/10.1007/s11858-018-0971-z>
- Huang, Y., Chen, S., Zhang, W., & Chen, M. (2026). Can generative AI feedback effectively enhance learning outcomes? A meta-analysis of 36 experimental and quasi-experimental studies. *Education Sciences*, 16(6), Article 816. <https://doi.org/10.3390/educsci16060816>
- Jiang, B., Gu, M., & Du, Y. (2023). Recent advances in intelligent textbooks for better learning. In H. Niemi, R. D. Pea, & Y. Lu (Eds.), *AI in learning: Designing the future* (pp. 247-261). Springer. https://doi.org/10.1007/978-3-031-09687-7_15
- Kaliisa, R., Misiejuk, K., López-Pernas, S., & Saqr, M. (2025). How does artificial intelligence compare to human feedback? A meta-analysis of performance, feedback perception, and learning dispositions. *Educational Psychology*, 46(1), 80-111. <https://doi.org/10.1080/01443410.2025.2553639>
- Krusche, S., & Seitz, A. (2018, February). ArTEMiS: An automatic assessment management system for interactive learning. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (pp. 284-289). ACM. <https://doi.org/10.1145/3159450.3159602>
- Lee, D., & Lee, Y. (2024). Productive failure-based programming course to develop computational thinking and creative problem-solving skills in a Korean elementary school. *Informatics in Education*, 23(2), 385-408. <https://doi.org/10.15388/infedu.2024.14>

- Leidig, P., & Salmela, H. (Eds.). (2020). *A competency model for undergraduate programs in information systems*. ACM. <https://doi.org/10.1145/3460863>
- Leidig, P. M., Ferguson, R. C., & Reynolds, J. H. (2019). Invited paper: IS2010: A retrospective review and recommendation. *Journal of Information Systems Education*, 30(4), 298-302. <https://jise.org/Volume30/n4/JISEv30n4p298.html>
- Liang, Y. D. (2017). *Introduction to Java programming and data structures, comprehensive version* (11th ed.). Pearson.
- Master, A., Alexander, T., Thompson, J., Fan, W., Meltzoff, A. N., & Cheryan, S. (2025). Causes and consequences of stereotypes: Interest stereotypes reduce adolescent girls' motivation to enroll in computer science classes. *Journal of Research on Technology in Education*, 57(1), 56-83. <https://doi.org/10.1080/15391523.2024.2402355>
- Mayer, R. E. (2024). The past, present, and future of the cognitive theory of multimedia learning. *Educational Psychology Review*, 36(1), Article 8. <https://doi.org/10.1007/s10648-023-09842-1>
- Pabst, D., Dosse, L. A., Wismer, S. E., & Barry, M. M. (2023, June). A comparative study on student performance using traditional and interactive textbooks. In *2023 ASEE Annual Conference & Exposition*. ASEE. <https://doi.org/10.18260/1-2--42369>
- Pankiewicz, M., & Baker, R. S. (2023). Large language models (GPT) for automating feedback on programming assignments. In *Proceedings of the 31st International Conference on Computers in Education (ICCE 2023)*. Asia-Pacific Society for Computers in Education. <https://doi.org/10.48550/arXiv.2307.00150>
- Pashler, H., McDaniel, M., Rohrer, D., & Bjork, R. (2008). Learning styles: Concepts and evidence. *Psychological Science in the Public Interest*, 9(3), 105-119. <https://doi.org/10.1111/j.1539-6053.2009.01038.x>
- Ponce, H. R., Mayer, R. E., & Méndez, E. E. (2025). Learner-generated and instructor-provided graphic organizers as aids to learning from text: A meta-analysis. *Educational Psychology Review*, 37, Article 107. <https://doi.org/10.1007/s10648-025-10084-6>
- Prather, J., Reeves, B. N., Leinonen, J., MacNeil, S., Randrianasolo, A. S., Becker, B. A., Kimmel, B., Wright, J., & Briggs, B. (2024). The widening gap: The benefits and harms of generative AI for novice programmers. In *Proceedings of the 2024 ACM Conference on International Computing Education Research* (pp. 469-486). ACM. <https://doi.org/10.1145/3632620.3671116>
- Rahman, M. M., & Paudel, R. (2018, October). Visual programming and interactive learning based dynamic instructional approaches to teach an introductory programming course. In *2018 IEEE Frontiers in Education Conference (FIE)* (pp. 1-6). IEEE. <https://doi.org/10.1109/FIE.2018.8658581>
- Rahmani, A. M., Groot, W., & Rahmani, H. (2024). Dropout in online higher education: A systematic literature review. *International Journal of Educational Technology in Higher Education*, 21(1), Article 19. <https://doi.org/10.1186/s41239-024-00450-9>
- Reiner, C. M., Bothell, T. W., Sudweeks, R. R., & Wood, B. (2002). *Preparing effective essay questions: A self-directed workbook for educators*. New Forums Press.
- Robinson, M. A. (2018). Using multi-item psychometric scales for research and practice in human resource management. *Human Resource Management*, 57(3), 739-750. <https://doi.org/10.1002/hrm.21852>

- Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2020). A meta-analysis of teaching and learning computer programming: Effective instructional approaches and conditions. *Computers in Human Behavior*, 109, Article 106349. <https://doi.org/10.1016/j.chb.2020.106349>
- Sinha, T., & Kapur, M. (2021). When problem solving followed by instruction works: Evidence for productive failure. *Review of Educational Research*, 91(5), 761-798. <https://doi.org/10.3102/00346543211019105>
- Sosnovsky, S., Brusilovsky, P., & Lan, A. (2025). Intelligent textbooks. *International Journal of Artificial Intelligence in Education*, 35(3), 967-986. <https://doi.org/10.1007/s40593-024-00451-9>
- Spencer, R., Comeau, E., Matchett, B., Biderman, M., Doria, N., Joy, P., & Numer, M. (2020). Interactive e-texts and students: A scoping review. *Canadian Journal of Education*, 43(1), 258-287. <https://www.jstor.org/stable/26954684>
- Sweller, J., van Merriënboer, J. J. G., & Paas, F. G. W. C. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, 10(3), 251-296. <https://doi.org/10.1023/A:1022193728205>
- Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes*. Harvard University Press.
- Welter, J., Adibi, Y., Barlow, R., Rodriguez, A., & Taylor, G. A. (2025). Student performance improvement in a circuit analysis course when interactive web-native textbook activities are assigned for points. In *2025 ASEE Annual Conference & Exposition*. ASEE. <https://doi.org/10.18260/1-2--57691>
- Zhang, X., Zhang, C., Stafford, T. F., & Zhang, P. (2013). Teaching introductory programming to IS students: The impact of teaching approaches on learning performance. *Journal of Information Systems Education*, 24(2), 147-155. <https://jise.org/Volume24/n2/JISEv24n2p147.html>
- Ziatdinov, R., & Valles, J. R., Jr. (2022). Synthesis of modeling, visualization, and programming in GeoGebra as an effective approach for teaching and learning STEM topics. *Mathematics*, 10(3), 398. <https://doi.org/10.3390/math10030398>
- Zimmerman, B. J. (2002). Becoming a self-regulated learner: An overview. *Theory Into Practice*, 41(2), 64-72. https://doi.org/10.1207/s15430421tip4102_2
- zyBooks. (n.d.). *zyBooks*. Retrieved June 26, 2026, from <https://www.zybooks.com>
- zyBooks. (2025, September 24). *Building AI literacy: Empowering students to explore and apply AI tools*. zyBooks Office Hours. <https://www.zybooks.com/building-ai-literacy-empowering-students-to-explore-and-apply-ai-tools/>
- zyBooks. (2026). *AI in zyBooks*. Retrieved August 30, 2026, from <https://www.zybooks.com/ai/>

APPENDIX A

Student Survey

Part I: Demographic Information

1. What is the class year?
2. What is your gender?
 - Male
 - Female
 - Other

Part II: Engagement in Programming Experience

1. I was engaged using (a traditional textbook/zyBooks) to learn to program. (Yes/No)
2. I would recommend the instructor continue using traditional (textbook/zyBooks) in the future. (Yes/No)

Part III: Textbook (Traditional or Interactive) Experience

1. Have you used the textbook to support your way of learning to program, explain how?
2. How often do you use the textbook for self-studying before each class?
3. Does the textbook give you more experiences and connections with the materials provided in class such as PowerPoint slides, and briefly justify your answer?
4. What are your thoughts on learning to program throughout the use of the textbook?
5. Explain to what degree you think the textbook helped you get prepared for the in-class exercises?
6. Explain to what degree you think the textbook helped you complete the programming assignments?
7. Do you think using the textbook eliminates the need for the TA or instructor help hours, explain how?
8. How does the textbook help you get prepared for exams?
9. What are the pros and cons of using the textbook in learning programming?
10. In your opinion, what is the most effective way to learn programming with less distress?
11. What do you recommend to the instructor to make the use of the textbook more effective in learning to program? Please, explain.

APPENDIX B
Interactive Textbook Essay Questions Evaluation

Evaluate 1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree

1. zyBooks's questions portray the exact ask. (e.g., Calculate the area of the circular pizza.)
2. zyBooks's questions have a single correct answer. (e.g., The Logic might differ but the output for the given input remains the same.)
3. zyBooks's questions have clear wordings to define the task. (e.g., Please initialize a variable of type integer.)
4. zyBooks's questions provide additional notes.

APPENDIX C zyBooks Main Dashboard

zyBooks My library

 What's new

 Adopt a zyBook

 zyC

 Search content

Search

[View content explorer](#)

[Configure zyBook](#)

Table of contents 

[About this material](#)

1. Introduction to Java



2. Variables / Assignments



3. Branches



4. Loops



5. User-Defined Methods



6. Arrays



7. Objects and Classes



8. Input / Output



APPENDIX D Qualitative Data Themes

Traditional Textbook	Interactive Textbook
Not Useful <i>"Never used the textbook"</i> <i>"Not at all"</i>	Useful <i>"I relied on zyBooks to teach me programming"</i> <i>"I use it all the time"</i>
Not a self-studying or practice tool <i>"I don't self-study the book before class"</i> <i>"I never use the textbook to help self-study before class"</i> <i>"The book did not help to prepare me for the in-class exercises."</i>	A self-studying and practice tool <i>"zyBooks was 10/10 when it came to completing the programming assignments because the practice code, we had to in zyBooks could be directly correlated to the code we needed to write for the programming assignments."</i> <i>"zyBooks is great to introduce the content and be prepared before lectures. It helped me understand the topics a lot more."</i>
Fewer experiences and connections <i>"The textbook doesn't really give more experience and/or connections than the power-point slides do. the examples are somewhat more confusing in the book which isn't necessary for beginner programmers and, in my opinion, makes concepts harder to grasp"</i> <i>"I don't believe the textbook gives a greater experience/connection with the materials provided in class. As stated before, I learn best through experience thus find greater benefit from tracing and practicing program assignments in class."</i>	More experiences and connections <i>"Yes, it gave me more opportunities to program outside of class"</i> <i>"Yes, I can relate what I am learning in class to the zyBooks questions."</i> <i>"I would say that the professor gave me the general knowledge and zyBooks gave me the practice"</i>
Doing, practicing, and having hands-on experience in programming learning <i>"Having less lecture time and more in-class interactive time. The most valuable classes were the ones that had hands-on programming when we could talk with each other and fumble through building a program together."</i> <i>"Practice examples in class and have students try them on their own before showing them the steps required to complete the program."</i>	Doing, practicing, and having hands-on experience in programming learning <i>"Daily practice is the best way to distress"</i> <i>"PRACTICE"</i>
Traditional Textbook is still a complementary resource in programming learning <i>"I think a TA can help offer students a different approach when tackling new concepts. A textbook only has one way of providing students with information which may not resonate with everybody."</i> <i>"I think that TA or instructor help hours are extremely beneficial to have as a resource. Having a person to speak with and explain your problems to that will help you through the thought process of a program is more useful than trying to develop a thorough understanding through the textbook alone."</i>	Interactive textbooks are still a complementary resource in programming learning <i>"There are still issues and problems you can run into that won't have a clear answer from zyBooks."</i> <i>"Sometimes it is best to be able to ask an actual person a question."</i>