Toyokawa et al. *Research and Practice in Technology Enhanced Learning* (2025) 20:39
https://doi.org/10.58459/rptel.2025.20039

Research and Practice in Technology Enhanced Learning

RESEARCH **Free and Open Access**

Supporting program comprehension with data-enhanced active reading

Yuko Toyokawa [1,3] *, Prajish Prasad [2], Izumi Horikoshi [3], Rwitajit Majumdar [4] and Hiroaki Ogata [3]

*Correspondence:
toyokawa.yuko.2j@kyoto-u.ac.jp.jp
Graduate School of Social Informatics,
Kyoto University,
Japan
Academic Center for Computing and Media Studies,
Kyoto University,
Japan
Full list of author information is available at the end of the article

Abstract

This study proposes a novel pedagogical approach to applying active reading (AR) strategies and the learning analytics dashboard to program comprehension (PC) in programming courses. The objective was to visualize the code-reading behaviors of novice programming learners using learning logs and promote code comprehension. The strategy was applied to students in computer science classes at a liberal arts college in India. The results show that the utilization of the dashboard positively influenced students' learning behaviors outside the classroom. It was recognized as an effective means of supporting PC, highlighting the need to elaborate on how to adopt dashboards in code reading tasks. The study confirms that reflecting on learning using the dashboard can promote learners' metacognitive skills, regardless of subject or language. It contributes to AR research by demonstrating new practical benefits of AR strategies for PC.

Keywords: Program comprehension, Active reading, Learning analytics

Introduction

Program comprehension (PC) is a skill and competency requiring many aspects of knowledge, such as critical thinking, communication, and collaboration. It has been defined as how individuals construct a mental model of a program (Izu et al., 2019). Software developers routinely engage in reading code and documentation (Begel & Simon, 2008). Just as the prerequisite for writing is being able to read written text, writing and running programs require being able to read and accurately understand written codes.

Busjahn et al. (2015) have suggested that programming source code and natural language text differ lexically and syntactically. Therefore, PC requires a specific approach that differs from natural language text reading. However, some past studies have approached PC from pedagogical aspects, such as classifying and organizing educational (Lobato &

Walters, 2017) and PC tasks (Izu et al., 2019). These studies include a series of PC tasks commonly used for reading activities in language learning, such as annotating, modifying, and explaining (Freeman et al., 2014; Izu et al., 2019). Begel and Simon (2008) focused on socialization—communicating and collaborating with others and coordinating while reflecting together—that professional novice programmers need and suggested incorporating these experiences into their curriculum to enable them to cope adequately. Against this backdrop, it is expected that active reading (AR) strategies, which have the same or similar pedagogical approaches adopted in PC classes and which value reflecting on learning and collaboration, could be applied to PC learning.

This study addresses whether AR strategies can be applied to PC. AR strategies have been applied to natural language texts in past research, but no examples exist of it being used for PC, which differs from natural language reading. Additionally, many PC studies are primarily based on interviews and surveys (Izu et al., 2019; Nelson et al., 2017; Sentance & Waite, 2017), and no past study has investigated the effectiveness of AR strategies in PC learning contexts using technology, by analyzing and visualizing the logs obtained from learners' learning processes. This study proposes an AR strategy in a learning analytics (LA) environment, called Data Enhanced Active Reading (DEAR), that contributes to LA and computer science research by investigating the effectiveness of AR strategies and LA dashboards in PC.

The remainder of this paper is organized as follows. The second section reviews previous studies on PC, AR, and LA, and the third section explains the DEAR framework and how it can be instantiated for PC to introduce, as a learning context, the Learning and Evidence Analytics Framework (LEAF). The fourth section introduces the research context, participants, and the PC class activity procedure and describes the data collection and analysis methods used for PC, as well as the behaviors analyzed. The fifth section presents the results of the investigation, the sixth section discusses the applicability of DEAR to PC as this study's contribution, along with its limitations, and the last section presents the conclusions and future research ideas.

Literature review

Program comprehension (PC)

PC is defined as a 'cognitive psychological process' (Wagner & Wyrich, 2022) in which individuals construct a mental model of a program (Izu et al., 2019). In learning, PC is conceptualized as a learning task that the learner engages in to encounter an artifact, which represents a program in some way (Izu et al., 2019). The ultimate purpose of learning programming is to compose and run programs. However, to be able to do so, it is necessary first to read and understand the written code. PC requires various abilities, including literacy, critical thinking, problem-solving, decision-making, information, and information and communication technology (ICT) literacy. Busjahn et al. (2015) argued that acquiring proper code-reading skills is essential as it contributes to problem-solving skills; thus, acquiring proper reading behaviors is an important goal of the programming science curriculum.

Research on PC learning has been approached from various perspectives, and programming pedagogical approaches such as those suggested by Predict-Run-Investigate-Modify-Make (Sentance & Waite, 2017) propose first addressing code understanding. The primary goal, especially for novice programming learners, is to be able to read and understand codes, for which the class curriculum should be designed to explicitly teach the structure of the language and solutions to problems. Some examples relating to pedagogical aspects are studies that include the classification of educational tasks (Lobato & Walters, 2017) and reports on programming activity tasks (Izu et al., 2019). Code comprehension tasks in programming classes include annotations, modifications, and explanations (Freeman et al., 2014; Furqani et al., 2018; Izu et al., 2019; Murphy et al., 2012), which are commonly used for reading activities in language learning. Izu et al. (2019) conducted a literature review and interviewed programming teachers, using a block model to summarize and organize a matrix of PC tasks in terms of learning dimensions and levels to describe a possible learning trajectory for a complex task. They confirmed that code reading is complex; hence, classes need to be conducted by combining a variety of tasks. The PC learning activities in their report include reading, tracing (Nelson et al., 2017), annotating (McCartney et al., 2004), highlighting (Kramer et al., 2019), and explaining codes in one’s own words (Murphy et al., 2012), which are commonly used in reading comprehension in natural language text. They also include pedagogical approaches, such as process-oriented guided inquiry learning (Kussmaul, 2012), which advocates a process of reading, analyzing, and reflecting on what has been learned, and the ‘Predict, Observe, Explain’ strategy (Furqani et al., 2018), which involves a process of prediction, observation, and explanation. All these pedagogical approaches to PC can be said to be comparable to AR strategies used in natural language texts. If this is the case, can the AR strategies used in natural language texts be applied to PC learning?

Active reading (AR) strategy and supporting it with learning analytics (LA)

AR is a reading-learning strategy often used in reading classes. It encourages learners to activate their existing knowledge and experience and relate them to new information to make sense of the written text (Ogle, 1986; Spivey, 1987). This pedagogical strategy is based on constructivist theories, with the concept of a student-centered, active learning approach (Cheek, 1992; Fosnot & Perry, 1996; Spivey, 1987; Yager & Lutz, 1994). It provides practice that encourages learners to engage deeply in reading, using techniques such as annotating and sharing with others (Pulver, 2020). For example, Survey, Question, Read, Recite, and Review (SQ3R) and Survey, Question, Read, Record, Recite, and Review (SQ4R) are popular strategies, as they are systematically structured and can be easily incorporated into classes (Anjuni & Cahyadi, 2019; Aziz, 2020; Basar & Gürbüz, 2017), thus commonly used in language learning classrooms for reading learning.

SQ4R is an extended version of SQ3R, with the Record phase added as the fourth phase. These strategies involve activities, such as getting the gist before reading (Survey), asking questions about what they are about to read (Question), deeply reading into the text as if searching for answers to questions that come to their mind (Read), sometimes leaving annotations, and highlighting important points (Record). It also elicits learners to recite the material in their own words (Recite), encouraging them to review and reflect on their understanding (Review), fostering further development and application after reading. In the modern era of technology, reading classes often incorporate e-books and e-learning tools to create active learning-based instruction into curricula. An advantage of learning using ICTs is that learning logs that could not be collected using paper-based learning can now be accumulated through actual learning.

The LA research field contributes to learning and teaching by investigating and analyzing learning logs generated in online learning environments and providing feedback for decision-making and support based on learning progress and status (Jivet et al., 2018; Siemens & Baker, 2012). LA dashboards are used as visualization tools and interventions to help teachers and learners make informed decisions about the learning process (Jivet et al., 2018). Dashboards generally assist teachers in various ways, such as monitoring the learning status of multiple students in real-time (Park & Jo, 2015, 2019), providing feedback (Ali et al., 2012), identifying at-risk students (Essa & Ayad, 2012), and encouraging teachers to take appropriate actions (Park & Jo, 2015). Information visualized in dashboards can affect students' psychology and behavior while reflecting and improving self-awareness (Verbert et al., 2013). They can also aid student collaboration and cooperation, thus increasing student engagement and improving retention and performance outcomes (Park & Jo, 2015).

To support AR learning from LA perspectives, an LA-enhanced system called LEAF (Ogata et al., 2018) has been used (Toyokawa et al., 2024). On its dashboards, reading logs obtained from AR learning were analyzed and visualized to provide learners and teachers with feedback in the English learning context. The study confirmed that using the e-book reader in the LEAF system improved AR performance, and using the AR dashboard to reflect on the reading process positively impacted learners' attitudes towards learning. Studies suggest that learning within an LA-enhanced context has a positive impact on learners' cognition and metacognition (Duval, 2011; Schwendimann et al., 2016; Verbert et al., 2013), and supports teachers in their educational practice (Ahn et al., 2019; Bao et al., 2021).

The study aimed to determine whether the AR strategy for reading comprehension in an LA-enhanced language learning context is applicable and effective for PC across learning contexts. Common PC tasks encompass adding highlights, annotations, explanations, and revisions, as well as reading and explaining code either to oneself or a partner (Izu et al., 2019). AR strategies also constitute annotating, explaining, and collaborating with others (Anjuni & Cahyadi, 2019; Aziz, 2020; Basar & Gürbüz, 2017; Pulver, 2020). The AR strategy and PC tasks take the same approach and are used as comprehension activities in the same way. Therefore, we believe that AR can cover PC tasks. Further, LA dashboards can be expected to enhance PC activities. Table 1 compares existing AR and PC tasks and shows the position of this research from an LA perspective.

Past studies have revealed the difference between reading natural language text and reading codes (Busjahn et al., 2015), although few have attempted to apply AR strategies used for natural language text reading to code reading. This study applied the AR strategy, using LEAF, to investigate its applicability to PC as class activities and its effectiveness for students and teachers.

Table 1 Comparison of AR, PC, and DEAR tasks

Activities	AR approaches (Anjuni & Cahyadi, 2019; Aziz, 2020; Basar & Gürbüz, 2017)	PC tasks (Izu et al., 2019)	AR with an LA dashboard
Prediction	✓	✓ (Furqani et al., 2018)	✓
Question	✓	✓ (Kussmaul, 2012)	✓
Record (annotation)	✓	✓ (Freeman et al., 2014; McCartney et al., 2004)	✓
Record (highlighting/marker)	✓	✓ (Kramer et al., 2019)	✓
Reading speed	✓	x	✓
Recite (explain)	✓	✓ (Freeman et al., 2014; Furqani et al., 2018; Murphy et al., 2012)	✓
Reflect & evaluate	✓	✓ (Kussmaul, 2012)	✓
Collaboration	✓	✓ (Begel & Simon, 2008)	✓
Utilization of logs	x	x	✓

Learning context

Data-enhanced active reading framework

DEAR—an AR framework within the LA-enhanced context—was designed based on the SQ4R strategy and adapted to AR learning activities using LEAF. Its reading strategy comprises three phases: pre-reading, while-reading, and post-reading. In the pre-reading phase, students activate prior knowledge from the information they find in the text and work to predict a rough outline of what they are about to read. They also create questions, if required, about what they want to know about the content. These predictions and questions are recorded in a memo. In the while-reading phase, students use a timer to track their reading speed, use markers to highlight unknown words or important ideas and leave annotations in memos, if required, while reading carefully. These activities can be performed individually and in collaboration with others. In the post-reading phase, the students consolidate and deepen the content by translating what they read into their own words, writing summaries, and answering comprehension questions.

Learning activities, such as annotations left in memos and highlights using markers, are analyzed and visualized on the dashboard. Reflecting on learning using the LA dashboard individually or with others encourages learners to review their own and others' outcomes, confirm what they have understood and what they need to review, decide what to do next, and improve their reading performance and motivation. The LA dashboard is also intended to encourage teachers to monitor the attempts of individual students and the class as a whole, grasp what the students have understood and what they have not, and intervene and give feedback, if necessary. In addition, by checking on their students' progress, the teachers can review the contents of the class, plan and prepare for subsequent activities and classes, and reflect on their own teaching methods and approaches. The DEAR activity cycle for teachers and students is presented in Figure 1.

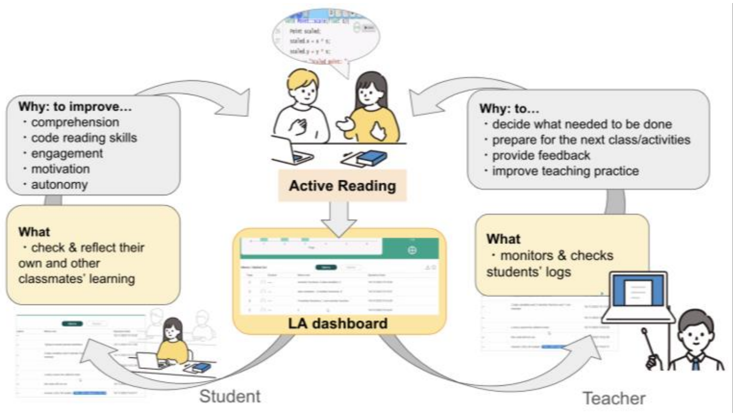

Why: to improve...
・comprehension
・code reading skills
・engagement
・motivation
・autonomy
What
・check & reflect their own and other classmates' learning
Active Reading
LA dashboard
Why: to...
・decide what needed to be done
・prepare for the next class/activities
・provide feedback
・improve teaching practice
What
・monitors & checks students' logs
Student
Teacher

Fig. 1 DEAR for teachers and students

Data-enhanced active reading for program comprehension

In this study, we constructed an interaction landscape framework to apply DEAR to PC activities. Its procedural aspects are based on the three phases of DEAR: pre-reading, while-reading, and post-reading, including PC tasks such as predicting, identifying, tracing, explaining, reflecting, and evaluating. Before attempting to read the code, the students create an image or predict the overall program structure and record what they have understood from the insights in a memo. Their logs, as outcomes, are shared in class and confirmed by the teacher using the dashboard. In while-reading activities, students first read to identify the rough implementation content and target code while being conscious of their reading speed, using a timer. Then, they answer the questions in the text, using markers to trace errors or target codes, and explain their answers to the questions, using a memo to understand the target code. Finally, they reflect and evaluate what they have understood as output, using the dashboard individually or collaboratively with others. If necessary, they may adjust and refine the code after reflecting and receiving feedback. They repeat these series of steps until they understand the target code. Figure 2 depicts the DEAR interaction landscape framework for PC activities.

LEAF: BookRoll and the dashboard as the learning analytics learning environment

To view the learning materials provided by the teacher, this study used BookRoll, a teaching and learning material-distributing system, in LEAF (Ogata et al., 2018). The LEAF framework has four components: 1) a Learning Management System like Moodle; 2) BookRoll; 3) a Learning Record Store (LRS) that accumulates and stores learning logs such as BookRoll operations and engagements; and 4) LA dashboards that analyze and

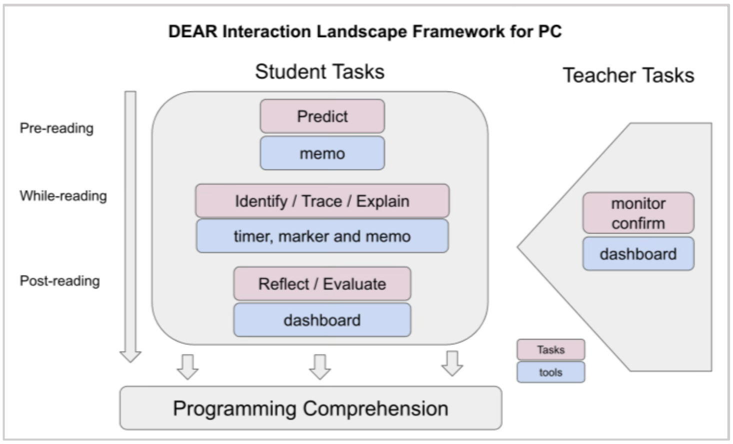
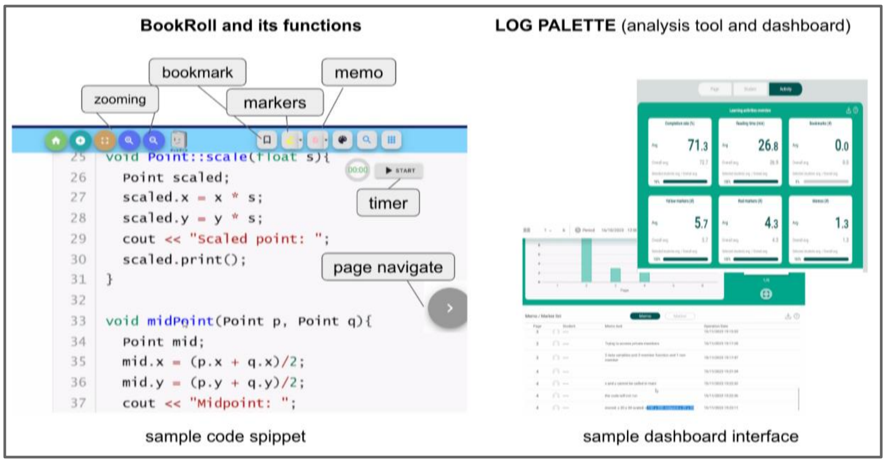


Fig. 2 DEAR instantiated for PC


BookRoll and its functions
LOG PALETTE (analysis tool and dashboard)
bookmark
memo
zooming
markers
timer
page navigate
71.3
26.8
0.0
5.7
4.3
1.3
sample code spippet
sample dashboard interface

```
void Point::scale(float s){
  Point scaled;
  scaled.x = x * s;
  scaled.y = y * s;
  cout << "Scaled point: ";
  scaled.print();
}

void midPoint(Point p, Point q){
  Point mid;
  mid.x = (p.x + q.x)/2;
  mid.y = (p.y + q.y)/2;
  cout << "Midpoint: ";
```

Fig. 3 Examples of interfaces of the BookRoll reading system and LOG PALETTE

visualize the accumulated logs. BookRoll includes features that facilitate reading, such as memos, markers, timers, and bookmarks, and students can annotate directly on the screen by handwriting or using a keyboard. These memos, markers, and timers facilitate reading comprehension of natural language texts. Memos (text or handwritten) can be used to leave annotations, markers for highlighting, and timers to measure and record reading speed. Dashboards can also be used to review, reflect, and decide on subsequent actions. In this study, students in the control group used BookRoll (including timers) only to view code learning materials, whereas those in the experimental group used BookRoll to read codes and perform DEAR activities using markers, memos, and timers. Learners' actions, such as their using markers, timers, and memos, are stored in the LRS and visualized on LogPalette, which is a collection of LA dashboard modules that analyze and visualize learning performance and artifacts created in BookRoll. Examples are 1) a module that visualizes and analyzes operations, such as viewing time and the use of markers and memos for each page of learning materials, 2) a module that shows learning activities for each student, and 3) a module that displays the total number and per-student average of learning activities. In the experimental group, the teacher and students mainly used the operation module to check the annotations students left in memos and reflect on learning. Figure 3 depicts examples of BookRoll's user interfaces and analysis tools.

Experimental design method

To investigate the potential of DEAR for PC, an experiment was conducted using BookRoll and the LA dashboard in the LEAF system. Given the need for background knowledge to understand how the proposed DEAR code reading approach could be a scaffolding to empower students' code reading performance, a quasi-experimental design was employed to investigate the following research questions (RQs):

RQ1: How effective is the DEAR strategy in PC learning contexts?

RQ2: How do the students and teachers perceive the DEAR strategy in PC learning contexts?

Research context and participants

The datasets were obtained from second-year college students (average age: 20 years). Most of these students had enrolled in a programming course for beginners at a liberal arts college in India after completing an introductory programming class. This experiment targeted two class sections: one as a control group and the other as an experimental group, comprising 27 and 20 students, respectively, who had consented to participate. Both groups' programming experience and motivation varied with their academic backgrounds. While the control group comprised students majoring in data science (n=11; 40%), computer science (CS) (n=5; 18%), applied math (n=2; 7%), psychology (n=1; 3%), and other fields (n=8; 9%); the experimental group included students majoring in CS (n=13, 61%), finance (n=2, 9%), psychology (n=1, 4%), business (n=1, 4%), and other fields (n=3, 19%). The same instructor taught both classes and conducted PC activities using LEAF. The flow of the PC classes is discussed next.

Program comprehension class activity procedure

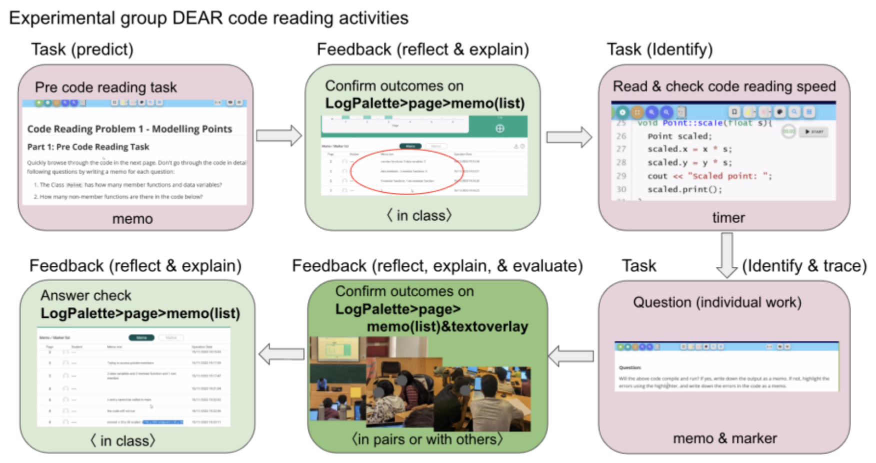
The classes were conducted face-to-face, twice a week, for 12 weeks from September to December. This experiment was conducted during the sixth and seventh weeks (two weeks), with four classes each. Before the experiment began, students in each class were given some trials to get used to operating BookRoll functions.

The objectives of the experimental tasks were to quickly and effectively identify the program's target codes and overall goal while working on DEAR PC tasks and apply their newly gained information to code composition. The lessons in both classes were planned in consultation with the teacher. During the experiment, the teacher and researchers continually reviewed and revised class activities in each lesson and exchanged feedback, questions, comments, and evaluations. The classes were conducted using basic PC tasks, such as tracing, annotating, explaining the codes in their own words, and reflecting on what they had learned during code reading, as indicated in past literature (Izu et al., 2019; Kussmaul, 2012). The experimental group performed these annotation and explanation tasks using memos and markers in BookRoll and confirmed and reflected on them using the dashboard. Figure 2 depicts the conformity of the interaction tasks.

Regarding the flow in each class, in the pre-reading phase, the teacher first reviewed the previous learning and explained the target activity. Students in the experimental group were additionally asked to browse through the code quickly, answer a question provided in the material to obtain an overview of the code before actually reading it, and write it in a memo. The purpose of making students perform this task was so that by applying their previous knowledge, they would connect it with new information. The teacher responded by checking the students' visualized annotations on the dashboard and providing feedback.

During the while-reading phase, the teacher got the students to read the code individually. A timer was used to measure the time taken to read. After students completed their individual reading, they answered questions provided in the material (e.g., could the code be compiled and run?). While working on the task, students in the experimental group were asked to use a marker to highlight any lines that were found to have errors or write the output of the code as a memo. Thereafter, the teacher instructed them to check the answers on their dashboards. Students in the experimental group explained, adjusted, and confirmed their answers to questions using their dashboards to visualize their answers. Based on their dashboard visualization, the teacher highlighted the points that needed explanation, shared them with the entire class, and explained the answers by eliciting answers from students. Figure 4 depicts a workflow example of the DEAR PC activities implemented in the experimental group.

The control group students performed their class activities without using a dashboard. After conducting the cycle as a PC activity, the teacher moved on to code composition and the next activity. These students followed the same workflow for class activities, except for prediction in the pre-reading phase, highlighting with markers and leaving annotations in a memo in the while-reading phase, and reflecting using the dashboard in the post-reading phase. Figure 5 describes the class activity procedure used in this study, and the PC tasks for each phase of AR and their purposes are summarized in Appendix Table A1.


Experimental group DEAR code reading activities
Task (predict)
Pre code reading task
memo
Feedback (reflect & explain)
Confirm outcomes on LogPalette>page>memo(list)
〈 in class〉
Task (Identify)
Read & check code reading speed
timer
Task (Identify & trace)
Question (individual work)
memo & marker
Feedback (reflect, explain, & evaluate)
Confirm outcomes on LogPalette>page> memo(list)&textoverlay
〈in pairs or with others〉
Feedback (reflect & explain)
Answer check LogPalette>page>memo(list)
〈 in class〉

Code Reading Problem 1 - Modelling Points

Part 1: Pre Code Reading Task

Quickly browse through the code in the next page. Don't go through the code in detai following questions by writing a memo for each question:

1. The Class Point has how many member functions and data variables?
2. How many non-member functions are there in the code below?

```
void Point::scale(float s){
  Point scaled;
  scaled.x = x * s;
  scaled.y = y * s;
  cout << "Scaled point: ";
  scaled.print();
```

Question:

Will the above code compile and run? If yes, write down the output as a memo. If not, highlight the errors using the highlighter, and write down the errors in the code as a memo.

Fig. 4 A workflow example of DEAR PC activities for the experimental group

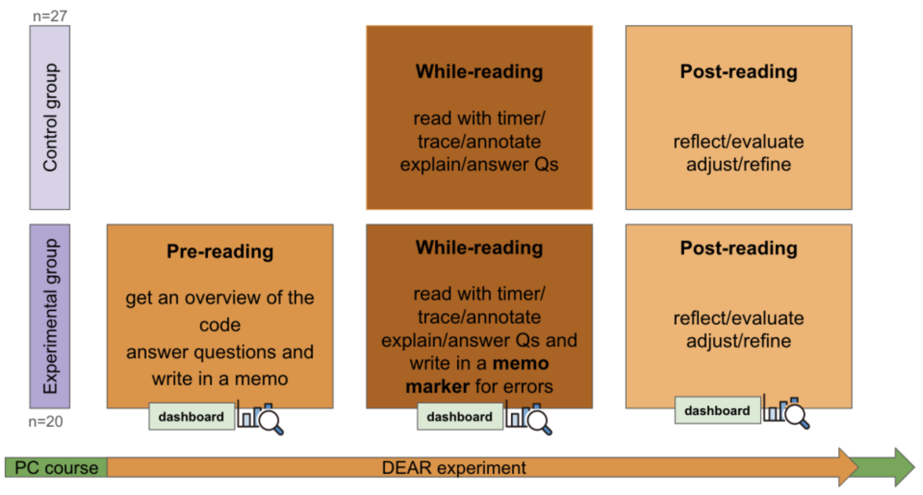

n=27
Control group
While-reading
read with timer/ trace/annotate explain/answer Qs
Post-reading
reflect/evaluate adjust/refine
Experimental group
n=20
Pre-reading
get an overview of the code
answer questions and write in a memo
dashboard
While-reading
read with timer/ trace/annotate explain/answer Qs and write in a memo marker for errors
dashboard
Post-reading
reflect/evaluate adjust/refine
dashboard
PC course
DEAR experiment

Fig. 5 PC class activity procedures of the experimental and the control groups

Data collection and analysis

Interaction logs

The log data of both the experimental and control groups, from the trials before the experiment, during its eight sessions, and out-of-class autonomous attempts during its seventh and eighth weeks using BookRoll, were aggregated in the LRS in the experience application programming interface (xAPI) format and analyzed. As indicators of students' learning engagement, this study extracted the number and duration of specific learning behaviors from LRS as follows: 1) in-class and out-of-class reading time to determine students' reading engagement time; 2) operation counts, including page navigation, the number of markers for highlighting to identify specific target codes and errors and the number of memos for answering pre-reading questions to predict the program's purpose, and answer its code-reading questions; 3) time logs of timer usage to check reading speed; and 4) the number of times the LA dashboard was checked in and outside the class to reflect on the learning outcomes. Of the 13606 logs accumulated in the LRS during the classes, learning logs obtained from the participants who consented came to 6,742 (n=47). Furthermore, data on reading speed were extracted from the TIMER_START, TIMER_PAUSE, and TIMER_STOP time logs and calculated as the number of words in the material being read divided by the time taken (in seconds) x 60. The experimental group's student logs are summarized in Appendix Table A2. To answer RQ 1, the applicability and effects of DEAR to PC were evaluated by examining logs obtained from using the LEAF framework (see Table 2) and analyzed to determine whether the difference was significant between the means of the control and experimental groups.

Table 2 Summary of indicators and statistical description for this study

Indicators	N	Missing	Mean	SD	Max	Min
In-class reading time	E 20	0	226.880	91.435	332.11	15.58
(minute)	C 26	1	200.900	82.064	346.41	16.78
Operation counts	E 20	0	137.400	70.696	279	22
(times)	C 26	1	47.923	20.016	82	6
Reading speed	E 20	0	133.970	68.517	362.2	10
(rate)	C 23	4	102.133	35.098	168.5	53
Dashboard access	E 17	3	2.471	1.125	6	1
(times)	C 0					
Out-of-class reading time	E 17	3	145.056	121.480	431.05	0.88
(minute)	C 14	13	60.326	66.545	214.06	0.75
Out-of-class dashboard access	E 10	10	1.80	1.135	4	1
(times)	C 0					

Survey instruments

A post-survey related to students' perceptions of PC activities was conducted for the experimental group to enable them to reflect on how they performed code reading activities during the experimental session. Students responded to nine post-survey items. Survey questions regarding code comprehension activities were developed based on the study skill inventory proposed by Congos (2011), the programming learning scale proposed by Sáez-López et al. (2016), and recommendations for dashboard development suggested by Jivet et al. (2018).

The initial three survey questions inquired about the dashboard's utility in deepening awareness and reflection on code, specifically if the dashboard served as a means of deepening the awareness and reflection on code by allowing students to review their own memos and those of others, aiding in better comprehension, facilitating communication with others, and sharing knowledge or understanding. The next three questions were about the utility of the overall AR strategy: if the AR strategy fostered a deeper interest in PC, actively had students engaged in class activities, and enhanced their understanding of specific aspects of programming through AR exercises. The last three questions were about the impact of each phase of DEAR, inquiring about the usefulness of using a timer to measure reading speed to improve PC, the impact of leaving annotations and highlighting, and the significance of pre-reading activities in providing an overview for a better understanding of the PC. These nine questions were combined to ascertain students' perceptions of the DEAR activities. A 5-point Likert-type scale was adopted to measure the students' perceptions, ranging from 5 = 'strongly agree' to 1 = 'strongly disagree'. Twenty-two participants in the experimental group completed the web form questionnaire after the experiment. Besides the students' survey results, the comments from the teacher and observation notes were also considered while answering RQ 2—the students' and teacher's perspectives on DEAR for PC. The purposes and contents of the post-survey are summarized in Table 3.

Table 3 Student survey questions

Constructs	Question Items
Usefulness of the Dashboard (Jivet et al., 2018) (for individual reflection)	Q1 Using the dashboard to see my own and others' notes enhanced my awareness and reflection of the code.
Usefulness of the Dashboard (Jivet et al., 2018) (for communication purposes)	Q2 I communicated with others using the dashboard.
Usefulness of the Dashboard (Jivet et al., 2018) (for collaboration)	Q3 Sharing information about code reading with others using the dashboard was helpful for code comprehension.
Overall PC class activities (Sáez-López et al., 2016)	Q4 After going through BookRoll, I became more interested in programming code reading.
Overall PC class activities (Sáez-López et al., 2016)	Q5 I actively participated in the class activities.
Overall PC class activities (Sáez-López et al., 2016)	Q6 I understood the specific aspects of programming through the reading activities.
Usefulness of DEAR (Congos, 2011) (while-reading: reading speed)	Q7 Using a timer to feel the speed of reading code encouraged me to read the code more efficiently.
Usefulness of DEAR (Congos, 2011) (while-reading: record)	Q8 Taking notes in a memo and highlighting with markers helped me understand how to program specific aspects of the problem.
Usefulness of DEAR (Congos, 2011) (pre-reading: prediction)	Q9 The activity of understanding the general outline before reading helped me understand the specific aspects of the code.

Results

Effects of the interaction process of program comprehension

We first checked the assumption of normality. We conducted a normality test of the dataset to determine the appropriateness of the test using JASP. The results show that the data are not normally distributed for operation counts ($p<0.001$), out-of-class reading time ($p=0.001$), and reading speed ($p<0.001$), while data are normally distributed for in-class reading time ($p=0.065$), which was presumed due to the small sample size. No difference existed in the amount of time for in-class reading time since both classes had the same amount of time (90 minutes) for each class. Furthermore, operation counts were significantly different since students in the experimental group used more BookRoll functions (such as memos and markers) in class. Thus, this study focused on out-of-class reading time and reading speed, and nonparametric tests were conducted to compare those variables between the experimental and the control groups to answer RQ1 (see Table 4). The tests revealed a significant difference between the two groups in terms of out-of-class reading time, $W=172$, $E=17$, $C=14$, $p=0.036$, with medium effect size, $r=0.445$, and reading speed, $W=311$, $E=20$, $C=23$, $p=0.049$, with medium effective size, $r=0.352$. The results

Table 4 Nonparametric test results between the experimental and the control group

Parameter	Grp	N	Mean	SD	W	P	r	Cohen's d
Out-of-class reading time (m)	E	17	145.05	121.48	172	0.036*	0.445	0.865
	C	14	60.32	66.54				
Reading speed	E	20	133.97	68.51	311	0.049*	0.352	0.585
	C	23	102.13	35.09				

Note: *p<0.05

suggest that the experimental group students spent significantly more time reading (or studying) out-of-class and showed faster code reading speed than the control group.

Practitioner's perspectives on proposed program comprehension activities

For RQ2, students' perceptions of DEAR in PC were verified by looking at the anonymous responses given to the survey questions by 22 students. Questions 1, 2, and 3 were about the dashboard. In response to survey question 1—personal reflections on using the dashboard—by selecting 'strongly agree' and 'agree', 68% said that dashboards allowed them to gain more information about code reading by reflecting on their own and others' artifacts, as well as more awareness and reflection on specific codes. In response to questions 2 and 3—sharing information about code reading and communicating with others using the dashboard—approximately 40–50% of the students gave positive answers, approximately 40% answered neutral, and approximately 10–20% disagreed. In response to question 4—PC activities using BookRoll and the dashboard—50% said they had become interested in PC after going through the activities, but 27% disagreed. In response to question 5—actively participated in class—32% chose 'strongly agree' and 54% 'agree' (i.e., 86% agreed), while the answers 'neutral' and 'disagree' had low percentages of 9% and 5%, respectively. Survey questions 6 through 9 were about the effects of the DEAR strategy. In response to question 6—whether students understood the specific programming content through PC activities—27% chose 'strongly agree' and 45% 'agree' (i.e., 72% agreed), 18% were neutral, and 9% disagreed. In reply to question 7—effectiveness of checking code reading speed while reading the code—32% selected 'strongly agree', 36% 'agree' (i.e., 68% agreed), 18% were neutral, and 14% disagreed. Regarding question 8—highlighting with markers and leaving annotations in memos—9% strongly agreed, 45% agreed, 32% were neutral, and 14% disagreed; thus, almost half of them agreed, while the other half disagreed or did not care. Regarding question 9—whether understanding the overview before reading helped them understand specific aspects of the code—over 80% agreed (strongly agree: 32%; agree: 50%), while none disagreed. The results of the survey responses are illustrated in Figure 6.

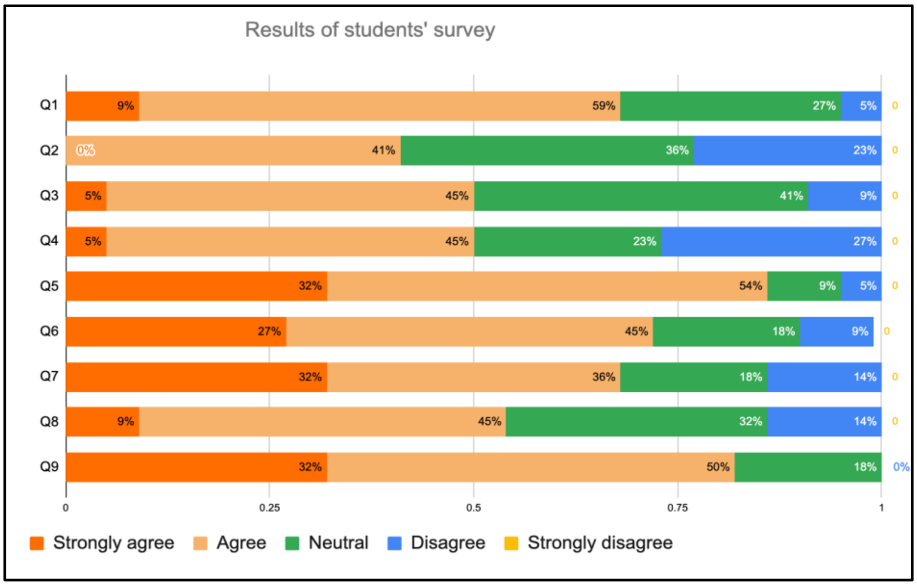


Fig. 6 Results for each question of the student survey (n=22)

Additionally, the survey outcomes were supported by the teacher's comments and observation memos (examples follow):

Comments from the teacher: I had a good time using BookRoll. It was useful in making students read the code. Without this experiment, I would not have been so systematic in giving them code-reading activities. Most of them (students) used the dashboard. It was useful to understand the range of responses students gave to the problems. I got a sense of the different opinions that students had.

From observation and message exchange with the teacher: The control group was generally more interactive and voluble, asking questions and interacting. Some students even gave their code-reading strategy in class. In the experimental group, some students asked questions in class, but generally, the class was less engaged, unlike students in the control group, who were more interactive.

Discussion

In response to the research questions based on the logs data and survey responses, this section first discusses the overall effects of the DEAR strategy on PC, followed by the impact of each phase of DEAR activities and the limitations of this study and future research scope.

Effects of data-enhanced active reading to program comprehension

The results of the nonparametric tests to answer RQ1 showed that the students in the experimental group spent more time working on PC out of the class (E: M=145.05, C: M=60.32). Based on the teacher's comments and observations, one can conclude that the students in the experimental group were more passive than those in the control group. However, several students tried out DEAR using memos and markers autonomously. For example, it was confirmed that students were drawing new markers (ADD_MARKER), erasing markers drawn during class (DELETE_MARKER), and leaving new annotations (ADD_MEMO). Some students also voluntarily checked the dashboard. Thus, the study confirmed that DEAR engagement in class positively impacted PC learning outside class. The results also support the suitableness of the programming code reading pedagogical approaches to PC in previous studies, such as annotating (Freeman et al., 2014; McCartney et al., 2004), analyzing and reflecting (Kussmaul, 2012), processing prediction, observation and explanation (Furqani et al., 2018), modifying and explaining (Freeman et al., 2014; Furqani et al., 2018; Izu et al., 2019; Murphy et al., 2012), and communicating and collaborating with others (Begel and Simon, 2008). Moreover, as the teacher indicated that the dashboard helped students read the code, enabled him to understand the range of students' responses to the questions, and different opinions of students, the results support past LA studies that suggest learning with a dashboard has a positive impact on learners' cognition and metacognition (Duval, 2011; Schwendimann et al., 2016; Verbert et al., 2013), and teachers in their educational practice (Ahn et al., 2019; Bao et al., 2021). With all these tasks in place, DEAR was proven adaptable to novice learners' PC activity.

The survey results for RQ2 showed that a high percentage of the participants (over 70%) said they understood the specific aspects of the target code through the PC activities and showed high motivation to participate in class. Survey questions 1, 2, and 3 pertained to dashboards. The results indicated that the participants found sharing and reflecting comparative data on the dashboard useful in understanding codes. Thus, the dashboard can be a tool for novice programming learners to reflect on their learning, particularly for those who did not have the answers themselves, needed review, and wanted clues from others. The results support the effectiveness of self-reflection with the dashboard (Duval, 2011; Schwendimann et al., 2016; Verbert et al., 2013).

Conversely, about 10–20% of the respondents did not agree with the idea of using the dashboard for sharing or communicating with others. The neutral numbers were also high, leading to the conclusion that dashboards may be more useful as tools for personal reflection rather than sharing or communicating. Using the dashboard to communicate with others is an important element of DEAR. Begel and Simon (2008) point out challenges that new graduates working in the software development industry face, such as communicating with others, collaborating, and making adjustments based on reflections, and to adequately address such challenges, recommend that these should be included in the educational curricula. The correlation between using dashboards for personal reflection and interacting with others is noteworthy. Additionally, the collaborative aspects of dashboards that facilitate information sharing and their impact on code comprehension should be further investigated. To verify the impacts of the records, a visualization method using a dashboard that can efficiently allow teachers and learners to confirm the code retrospectively should be considered.

Impact of each phase of DEAR activities

Pre-reading tasks

Approximately 82% answered that pre-reading activities helped them to understand the general outline and specific aspects of the code. The importance of pre-reading activities has received attention in research on natural language texts, such as the advanced organizer (Ausubel, 1978; Elfeky et al., 2020) and cognitive reading strategies (Gustanti & Ayu, 2021). The result indicates the potential of pre-code reading activities to help students identify specific code elements and grasp an overview of programming.

While-reading tasks: Awareness of PC speed

Regarding checking code-reading speed, the teacher commented that as he gave his students plenty of time to work on PC code reading, he did not consider code-reading speed as very important. The reading speed rate is often measured in natural language text reading; hence, it was considered that this perspective could also be applied to PC. As a result, 68% of participants answered that cognizance of their code reading speed enabled them to read codes more efficiently. Word-level automaticity—the ability to quickly and easily decode or recognize words both in and out of context—is thought to be important in reading natural language texts (Hudson et al., 2005). Therefore, the reading speed of natural language texts, such as in learning English, is often measured to improve readers' reading comprehension rate and reading comprehension. Wagner and Wyrich (2022) examined the influence of intelligence and certain personality traits on PC and identified fluid intelligence, visual perception, and cognitive speed as the influencing factors. The result of the nonparametric test in this study revealed that the reading speed of the experimental students was faster than that of those in the control group. The experimental group having a higher reading speed indicates that they read the code faster. It is necessary to anticipate that actual programming practice would require quickly reading, understanding, completing, and running codes. Programming codes may be recognized at the token, code, or line levels rather than at the word level, as in natural language text. For future studies, this study recommends using reading speed measurement to improve the efficiency of programming code reading, considering calculation methods suitable for code reading, and examining the correlation between code understanding and code reading time, which were not considered in this study.

While-reading tasks: Record (highlighting and leaving annotations)

In this study, students were asked to use markers to identify and trace the code and explain what they anticipated or understood in a memo. While approximately half agreed to record tasks, the other half disagreed or said they did not care about highlighting with markers and making annotations in memos. Hence, one can infer that highlighting and annotating, commonly conducted in natural language text reading, may not be useful in code reading. The logs revealed that two students who studied with BookRoll outside the class used ADD_MEMO and ADD_MARKER, and one used DELETE_MARKER on the markers drawn in class. The reason for keeping records was to later review and reflect on what was understood or not understood. A student did not check the dashboard in class despite using memos and markers. In addition, nine students did not check the dashboard after learning outside the class, whereas two students who did not study with BookRoll outside the class had logs of accessing the dashboard. DEAR promotes activities such as checking remaining records (logs) on the dashboard and, thereafter, deciding on the next course of action. Future studies will further investigate the purpose of highlighting and leaving annotations and how the logs can be visualized and used for PC.

Post-reading tasks: Dashboard for PC

The DEAR tasks include checking, reflecting, and evaluating PC using a dashboard. Teachers select learning tasks and provide a series of learning activities tailored to learners' cognition and knowledge levels to help them reach their goals (Lobato & Walter, 2017). In this study, the students' artifacts were visualized on an LA dashboard and used by the teacher as a kind of decision-making tool (e.g., allowing the teacher to check students' understanding levels in class and provide instant feedback on the spot). Teachers first check and make assumptions about what students initially understand and what they can learn next, and accordingly construct and modify the lesson's goals. Thus, the dashboard served as a scaffolding to smoothly execute the learning trajectory.

This study revealed that the DEAR strategy, which further enhances existing AR strategies used for natural language learning with an LA dashboard, can be applied to code reading for novice programming learners in a PC learning context. However, challenges with the DEAR strategy were also recognized. The following section discusses the limitations of this study and points out future work scope, including the need to improve the dashboard.

Limitations and future work

Reliability of the post-survey responses

First, we must acknowledge the limitation regarding the post-survey on the perception of DEAR among the students in the experimental group. Whether the survey was reliable depends on the length of the questionnaire, the quality of the questions, and their fit to the group being measured (Brown, 1997). In this study, first, the number of participants was small (n = 20). Moreover, even though questions were considered with reference to the existing survey items related to AR and PC, the inference was based on a small number of survey items of nine questions. These might have affected the reliability and generalizability of the results. The small number of questions was because we had to limit them to avoid disrupting the class, as we wanted to have the students answer the questions within a limited time. Therefore, we selected questions to fit this study based on questionnaires used in the literature on AR activities, based on the Study Skill Inventory proposed by Congos (2011), programming learning scale proposed by Sáez-López et al. (2016), and recommendations for dashboard development from Jivet et al. (2018). We also tried to triangulate based on the observations and understanding of the teacher to address this limitation. The teacher's comments and class observations are considered sufficient supplements to support the reliability of the post-survey.

Equivalency of study groups and participants

The second limitation pertained to the participants' programming experience. They were beginner course students in programming, yet almost all had completed an introduction course, and some were majors in CS and other programming-related fields. Thus, some of them perhaps already had some code comprehension knowledge, which might have affected their PC learning behaviors and perceptions. The AR strategies used in natural language text reading provide learners with a more systematic reading approach and are hence suitable for novice readers with less experience in reading, who can learn and acquire a reading strategy and apply it to understand the content. In other words, AR may not be necessary for advanced learners to have their own reading strategies. More precise results could emerge in future studies by applying DEAR for PC in K-12 or introductory courses.

The grouping of the experiment also affects research results. As the experiment was conducted in a real learning context, the experimental and control groups were randomly selected without controlling for equivalency. As the semester progressed, the teacher recognized that students in the control group participated more actively in class and understood better than those in the experimental group. Additionally, students in the control group were more interactive, and their engagement was stronger, even though more than 60% of students in the experimental group said in the survey that they had actively participated in PC class activities. The results indicate a possibility of the contextual factors of the two groups influencing them; hence, establishing the equivalency of study groups should be considered in future research.

Devising dashboards for code reading

Future studies should highlight the need for dashboards to review and reflect, specifically for code reading and visualizing the code-reading process on dashboards. This study showed that the dashboard supported the teacher and students by confirming students' understanding and progress and as a scaffolding for decision-making in class. In this regard, DEAR, which uses AR strategy and LA technology, is effective for code reading. However, the dashboard was created to reflect on AR learning in natural language text and was not initially meant for code comprehension. In other words, there is a lack of ingenuity in visualizing the code comprehension process in an easy-to-understand manner, and the elements which are geared towards user needs for code comprehension. The literature suggests the benefits of designing dashboards for user-specific factors and needs (Ahn et al., 2019). Considering this and the distinct pedagogical concepts of improving and supporting learning (Jivet et al., 2018), designing dashboards that show the correct and necessary choices for programming learners is essential.

Conclusion

This is the first study to apply an AR strategy to PC, introduced as a possible option for PC activities from the LA perspective. The effectiveness of this approach was demonstrated by the positive impact of the utilization of the LA dashboard on the learning behavior of students in the experimental group outside class. The proposed DEAR strategy was proved to be applicable to PC tasks for novice programming learners. Despite the limitations of the reliability of the post-survey, the equivalence of the learning groups and participants, and the usability of the LA dashboard, it is hoped that future studies will expand DEAR for PC to test its suitability for individual learners based on their needs, motivation, programming experience, and other aspects, and consider visualization methods of PC logs to meet their needs. This study contributes across multiple LA, AR, and programming science research fields. DEAR, an integration of AR strategies and LA, demonstrates one of the potential pedagogical and curriculum design implications that can be applied to PC.

Appendix

Table A1 PC tasks and their purpose

Active reading phases	Class activities	Control group	Experimental group	Purpose of activities
Pre	Some explanations from the teacher	✓	✓	To introduce the learning target
	Ask questions related to learning		✓ Ask some questions related to codes to students (teacher shows the dashboard to the students)	Pre-AR activities: activate prior knowledge related to the learning and relate it to the new information to understand the concept better
While	Read individually	✓ Timer	✓ Timer	To check students' code reading speed
	Record individually, Highlight unknown and/or important points, Leave annotations (answers to questions)		✓ Marker & memo	1. To make students aware of important parts and parts that they did not understand 2. To let students reflect on their learning later 3. To activate different cognitive processes
Post	Reflection, Pair or group work	✓ Exchange notes, talk to neighbors (mediated by whatever they have)	✓ In a memo (mediated by dashboard, reflect on the dashboard in pairs or groups)	To help them understand what was understood and what was not and decide on what needed to be done next

Table A2 Summary of logs for each indicator from the students in the experimental group

Student	In-class reading engagement (m)	Operation counts	Marker counts	Memo counts	Dashboard access	Reading speed (average)	Out-of-class reading engagement (m)	Out-of-class dashboard access
E1	216.68	99	0	14	2	178.766	321.86	3
E2	276.73	103	0	8	6	173.266	10.83	0
E3	230.53	100	0	1	3	58.9	136.03	1
E4	306.88	196	112	11	2	137.95	0.88	0
E5	293.66	180	102	10	3	95.025	219.11	0
E6	267.21	247	65	19	2	130	94.03	1
E7	325.78	279	47	18	2	152.45	203.26	2
E8	155.25	44	0	3	2	99.55	0	3
E9	163.08	87	0	1	3	362.2	0	1
E10	15.58	22	0	0	0	10	1.18	0
E11	143	110	22	5	0	91.7	68.65	0
E12	228.43	154	16	11	2	164.8	431.05	4
E13	256.38	118	0	11	2	143.45	59.11	1
E14	287.58	113	5	4	3	110.5	27.06	0
E15	319.5	244	78	27	3	108.15	184.5	0
E16	332.11	124	9	15	1	142.65	233.91	0
E17	82.81	118	14	7	1	186.05	58.71	1
E18	234.1	234	14	13	3	130.75	213.13	0
E19	85.5	67	3	5	0	85.15	0	0
E20	316.8	109	0	3	2	118.1	202.65	1

Abbreviations
AR: Active reading; CS: Computer science; DEAR: Data-enhanced active reading; ICT: Information, and information and communication technology; LA: Learning analytics; LEAF: Learning and evidence analytics framework; LRS: Learning record store; PC: Program comprehension; RQ: Research question; SQ3R: Survey, Question, Read, Recite, and Review; SQ4R: Survey, Question, Read, Record, Recite, and Review.

Acknowledgements
The authors would like to thank the students in the programming classes for their contributions to the study.

Authors' contributions
YT conceptualized the study, performed the data analysis, and drafted the original manuscript.

PP provided lesson design ideas, implemented the programming classes, provided insight, and checked analysis results.

RM conceived the idea of the study, provided insight into the framework of the overall argument, and checked analysis results.

IH contributed to conducting the analysis and provided insight.

HO initiated the framework of the overall argument and supervised the conduct of this study.

RM and HO acquired funding for the research.

The authors read and approved the final manuscript.

Authors' information
YT: Graduate School of Informatics, Kyoto University, Yoshidahonmachi, Sakyo Ward, Kyoto, 606-8501, Japan

PP: School of Computing and Data Sciences, FLAME University, Pune, Maharashtra, India

YT, IH, & HO: Academic Center for Computing and Media Studies, Kyoto University, Yoshidahonmachi, Sakyo Ward, Kyoto, 606-8501, Japan

RM: Research and Educational Institute for Semiconductors and Informatics, Kumamoto University, 2-39-1 Kurokami, Chuo-Ku, Kumamoto City, Kumamoto, 860-0862, Japan

Funding
This work is partially funded by NEDO JPNP20006 and JPNP18013, JSPS KAKENHI (A) JP23H00505 and (B) JP22H03902 and, National Institute for Educational Policy Research: Educational Data Analysis and Research Promotion Project FY2023-2025.

Availability of data and materials
The datasets used and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Graduate School of Informatics, Kyoto University, Japan

[2] School of Computing and Data Sciences, FLAME University, India

[3] Academic Center for Computing and Media Studies, Kyoto University, Japan

[4] Research and Educational Institute for Semiconductors and Informatics, Kumamoto University, Japan

Received: 22 January 2024 Accepted: 19 September 2024
Published online: 3 February 2025

References

Ahn, J., Campos, F., Hays, M., & DiGiacomo, D. (2019). Designing in context: Reaching beyond usability in learning analytics dashboard design. *Journal of Learning Analytics*, *6*(2), 70–85. https://doi.org/10.18608/jla.2019.62.5

Ali, L., Hatala, M., Gašević, D., & Jovanović, J. (2012). A qualitative evaluation of evolution of a learning analytics tool. *Computers & Education*, *58*(1), 470–489. https://doi.org/10.1016/j.compedu.2011.08.030

Anjuni, G. R., & Cahyadi, R. (2019). Improving students' reading comprehension through Sq3r (Survey, Question, Read, Recite and Review) technique. *Project (Professional Journal of English Education)*, *2*(1), 1–6. https://doi.org/10.22460/project.v2i1.p1-6

Ausubel, D. P. (1978). In defense of advance organizers: A reply to the critics. *Review of Educational Research*, *48*(2), 251–257. https://doi.org/10.3102/00346543048002251

Aziz, I. N. (2020). Implementation of SQ3R method in improving the students' basic reading skill. *EDUCATIO: Journal of Education*, *5*(1), 97–106. https://doi.org/10.29138/educatio.v4i1.179

Bao, H., Li, Y., Su, Y., Xing, S., Chen, N. S., & Rosé, C. P. (2021). The effects of a learning analytics dashboard on teachers' diagnosis and intervention in computer-supported collaborative learning. *Technology, Pedagogy and Education*, *30*(2), 287–303. https://doi.org/10.1080/1475939X.2021.1902383

Basar, M., & Gürbüz, M. (2017). Effect of the SQ4R technique on the reading comprehension of elementary school 4th grade elementary school students. *International Journal of Instruction*, *10*(2), 131–144. https://doi.org/10.12973/iji.2017.1029a

Begel, A., & Simon, B. (2008). Novice software developers, all over again. In M. E. Caspersen, R. Lister & M. Clancy (Eds.), *Proceedings of the Fourth International Workshop on Computing Education Research* (pp. 3–14). ACM. https://doi.org/10.1145/1404520.1404522

Brown, J. D. (1997). Reliability of surveys. *Shiken: JALT Testing & Evaluation SIG Newsletter*, *1*(2), 18–21. http://www.jalt.org/test/PDF/Brown2.pdf

Busjahn, T., Bednarik, R., Begel, A., Crosby, M., Paterson, J. H., Schulte, C., Sharif, B., & Tamm, S. (2015). Eye movements in code reading: Relaxing the linear order. In *2015 IEEE 23rd International Conference on Program Comprehension* (pp. 255–265). IEEE. https://doi.org/10.1109/ICPC.2015.36

Cheek, D. W. (1992). *Thinking constructively about science, technology, and society education: General introduction and from the creation to the flood.* Suny Press.

Congos, D. H. (2011). *Starting out in community college*. McGraw-Hill.

Duval, E. (2011). Attention please! Learning analytics for visualization and recommendation. In P. Long, G. Siemens, G. Conole & D. Gašević (Eds.), *Proceedings of the 1st International Conference on Learning Analytics and Knowledge* (pp. 9–17). ACM. https://doi.org/10.1145/2090116.2090118

Elfeky, A. I. M., Masadeh, T. S. Y., & Elbyaly, M. Y. H. (2020). Advance organizers in flipped classroom via e-learning management system and the promotion of integrated science process skills. *Thinking Skills and Creativity*, *35*, 100622. https://doi.org/10.1016/j.tsc.2019.100622

Essa, A., & Ayad, H. (2012, April). Student success system: Risk analytics and data visualization using ensembles of predictive models. In S. Dawson, C. Haythornthwaite, S. Buckingham Shum, D. Gasevic & R. Ferguson (Eds.), *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (pp. 158–161). ACM. https://doi.org/10.1145/2330601.2330641

Fosnot, C. T., & Perry, R. S. (1996). Constructivism: A psychological theory of learning. *Constructivism: Theory, Perspectives, and Practice*, *2*(1), 8–33.

Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., & Wenderoth, M. P. (2014). Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*, *111*(23), 8410–8415. https://doi.org/10.1073/pnas.1319030111

Furqani, D., Feranie, S., & Winarno, N. (2018). The effect of predict-observe-explain (POE) strategy on students' conceptual mastery and critical thinking in learning vibration and wave. *Journal of Science Learning*, *2*(1), 1–8.

Gustanti, Y., & Ayu, M. (2021). The correlation between cognitive reading strategies and students' English proficiency test score. *Journal of English Language Teaching and Learning*, *2*(2), 95–100. https://doi.org/10.33365/jeltl.v2i2.1452

Hudson, R. F., Lane, H. B., & Pullen, P. C. (2005). Reading fluency assessment and instruction: What, why, and how? *The Reading Teacher*, *58*(8), 702–714. https://doi.org/10.1598/RT.58.8.1

Izu, C., Schulte, C., Aggarwal, A., Cutts, Q., Duran, R., Gutica, M., Heinemann, B., Kraemer, E., Lonati, V., Mirolo, C., & Weeda, R. (2019). Fostering program comprehension in novice programmers-learning activities and learning trajectories. In G. Rößling, M. Giannakos, B. Scharlau, R. McDermott, A. Pears & M. Sabin (Eds.), *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education* (pp. 27–52). ACM. https://doi.org/10.1145/3344429.337250

Jivet, I., Scheffel, M., Specht, M., & Drachsler, H. (2018). License to evaluate: Preparing learning analytics dashboards for educational practice. In A. Pardo, K. Bartimote-Aufflick, G. Lynch, S. Buckingham Shum, R. Ferguson, A. Merceron & X. Ochoa (Eds.), *Proceedings of the 8th International Conference on Learning Analytics and Knowledge* (pp. 31–40). ACM. https://doi.org/10.1145/3170358.3170421

Kramer, M., Barkmin, M., & Brinda, T. (2019, July). Identifying predictors for code highlighting skills: A regressional analysis of knowledge, syntax abilities and highlighting skills. In B. Scharlau, R. McDermott, A. Pears & M. Sabin (Eds.), *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 367–373). ACM. https://doi.org/10.1145/3304221.3319745

Kussmaul, C. (2012). Process-oriented guided inquiry learning (POGIL) for computer science. In L. S. King, D. R. Musicant, T. Camp & P. Tymann (Eds.), *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (pp. 373–378). ACM. https://doi.org/10.1145/2157136.2157246

Lobato, J & Walter, C. D. (2017). A taxonomy of approaches to learning trajectories and progressions. In J. Cai (Ed.), *Compendium for research in mathematics education* (pp. 74–101). National Council of Teachers of Mathematics.

McCartney, R., Moström, R. J., Sanders, E. K. & Seppälä, O. (2004). Questions, annotations, and institutions: Observations from a study of novice programmers. In *Proceedings of the 4th Koli Calling International Conference on Computing Education Research (KoliCalling'04)* (pp. 11–19). ACM.

Murphy, L., McCauley, R., & Fitzgerald, S. (2012). 'Explain in Plain English' questions: Implications for teaching. In L. S. King, D. R. Musicant, T. Camp & P. Tymann (Eds.), *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education (SIGCSE'12)* (pp. 385–390). ACM. https://doi.org/10.1145/2157136.2157249

Nelson, G. L., Xie, B., & Ko, A. J. (2017, August). Comprehension first: Evaluating a novel pedagogy and tutoring system for program tracing in CS1. In J. Tenenberg, D. Chinn, J. Sheard, L. Malmi, J. Tenenberg & L. Malmi (Eds.), *Proceedings of the 2017 ACM Conference on International Computing Education Research* (pp. 2–11). ACM. https://doi.org/10.1145/3105726.3106178

Ogata, H., Majumdar, R., Akçapinar, G., Hasnine, M. N., & Flanagan, B. (2018, November). Beyond learning analytics: Framework for technology-enhanced evidence-based education and learning. In Y.-T. Wu et al. (Eds.), *26th International Conference on Computers in Education Workshop Proceedings* (pp. 493–496). Asia-Pacific Society for Computers in Education.

Ogle, D. M. (1986). KWL: A teaching model that develops active reading of expository text. *The Reading Teacher*, *39*(6), 564–570. https://doi.org/10.1598/RT.39.6.11

Park, Y., & Jo, I. H. (2015). Development of the learning analytics dashboard to support students' learning performance. *Journal of Universal Computer Science*, *21*(1), 110–133. https://doi.org/10.3217/jucs-021-01-0110

Park, Y., & Jo, I. H. (2019). Factors that affect the success of learning analytics dashboards. *Educational Technology Research and Development*, *67*(6), 1547–1571. https://doi.org/10.1007/s11423-019-09693-0

Pulver, C. J. (2020). Active reading to understand a problem. In *Thinking rhetorically: Writing for professional and public audiences* (pp. 151–160). Roger Williams University Department of Writing Studies, Rhetoric, and Composition. https://rwu.pressbooks.pub/thinkingrhetorically/chapter/active-close-reading/

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two-year case study using "Scratch" in five schools. *Computers & Education*, *97*, 129–141. https://doi.org/10.1016/j.compedu.2016.03.003

Schwendimann, B. A., Rodriguez-Triana, M. J., Vozniuk, A., Prieto, L. P., Boroujeni, M. S., Holzer, A., Gillet, D., & Dillenbourg, P. (2016). Perceiving learning at a glance: A systematic literature review of learning dashboard research. *IEEE Transactions on Learning Technologies*, *10*(1), 30–41. https://doi.org/10.1109/TLT.2016.2599522

Sentance, S., & Waite, J. (2017, November). PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. In E. Barendsen & P. Hubwieser (Eds.), *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 113–114). ACM. https://doi.org/10.1145/3137065.3137084

Siemens, G., & Baker, R. S. D. (2012, April). Learning analytics and educational data mining: Towards communication and collaboration. In S. Dawson, C. Haythornthwaite, S. Buckingham Shum, D. Gasevic & R. Ferguson (Eds.), *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge* (pp. 252–254). ACM. https://doi.org/10.1145/2330601.2330661

Spivey, N. N. (1987). Construing constructivism: Reading research in the United States. *Poetics*, *16*(2), 169–192. https://doi.org/10.1016/0304-422X(87)90024-6

Toyokawa, Y., Majumdar, R., Kondo, T., Horikoshi, I., & Ogata, H. (2024). Active reading dashboard in a learning analytics enhanced language-learning environment: Effects on learning behavior and performance. *Journal of Computers in Education*, *11*, 495–522. https://doi.org/10.1007/s40692-023-00267-x

Verbert, K., Duval, E., Klerkx, J., Govaerts, S., & Santos, J. L. (2013). Learning analytics dashboard applications. *American Behavioral Scientist*, *57*(10), 1500–1509. https://doi.org/10.1177/0002764213479363

Wagner, S., & Wyrich, M. (2022). Code comprehension confounders: A study of intelligence and personality. *IEEE Transactions on Software Engineering*, *48*(12), 4789–4801. https://doi.org/10.1109/TSE.2021.3127131

Yager, R. E., & Lutz, M. V. (1994). Integrated science: The importance of "how" versus "what". *School Science and Mathematics*, *94*(7), 338–346. https://doi.org/10.1111/j.1949-8594.1994.tb15690.x

Publisher's Note