Visible thinking to support online project-based learning

Education and Information Technologies (2024) 29:2329–2363
https://doi.org/10.1007/s10639-023-11896-1

Visible thinking to support online project-based learning: Narrowing the achievement gap between high- and low-achieving students

Jun Peng[1] · Meng Sun[2] · Bei Yuan[3] · Cher Ping Lim[4] · Jeroen J. G. van Merriënboer[5] · Minhong Wang[6,7]

Received: 28 September 2022 / Accepted: 9 May 2023 / Published online: 2 June 2023

Abstract

Project-based learning (PjBL) has been increasingly promoted and extended to online environments to enhance the quality of higher education. However, PjBL involves complex processes requiring higher-order thinking skills, which may pose challenges to many students especially in online settings with little prompt support from teachers. The problem may compromise the learning of low-achieving students, who often have inadequate higher-order thinking skills. Visible thinking approaches have the potential to make higher-order thinking processes accessible to students. This study was conducted with 72 university students who engaged in visible thinking supported online PjBL of computer programming. A one-group pretest-posttest design was adopted to compare the learning outcomes among high-, medium- and low-achieving students. The results showed that compared to high and medium achievers, low-achieving students made the most progress in product quality and thinking skills (in particular process design skills). They performed almost as well as medium and high achievers in product quality and process design skills at the end of the study. They also gained more knowledge from the project than high achievers did. Moreover, compared to medium achievers, low achievers perceived the approach as more valuable, made more effort on the study, and felt more competent in completing the project. The findings reveal the promising effects of visualizing higher-order thinking processes in narrowing the achievement gap between high and low achievers, offering all students an equal chance to engage in effective learning with projects.

Keywords Project-based learning · Online learning · Visible thinking · Computer programming

Extended author information available on the last page of the article

1 Introduction

It has been a constant challenge for higher education institutions to address the competency gaps between graduates' professional attributes and employees' expectations in areas such as problem-solving, reasoning, and decision making (Harvey, 2000; Jollands et al., 2012). To address the challenge, project-based learning (PjBL) has been widely promoted in higher education especially in the senior years of undergraduate studies, where students are encouraged to learn by completing real-world projects (Jollands et al., 2012).

PjBL is rooted in learning-by-doing theory which claims that students develop a meaningful understanding of knowledge by integrating knowing and doing instead of being passive recipients of knowledge (Blumenfeld et al., 2011; Thomas, 2000). The core idea of PjBL is to have students actively engaged in meaningful tasks and inquiry processes (organized around problems or projects in realistic contexts) to acquire a deeper understanding of the subject matter and develop practical skills for problem-solving, decision making, and communication (Blumenfeld et al., 2011; Chen & Yang, 2019). Moreover, PjBL features student-centered inquiry, which emphasizes learner autonomy and independence by encouraging students to take high levels of responsibility for their learning. Compared to another inquiry-based and student-centered pedagogy called problem-based learning (PBL), PjBL emphasizes the creation of tangible solutions and end products (or artifacts) closer to professional reality (Loyens & Rikers, 2016).

With its advantages in improving student motivation, the understanding of abstract knowledge, and the development of soft skills (in problem-solving, communication, and self-regulation), PjBL has been increasingly promoted in educational practice (Chen & Yang, 2019; Guo et al., 2020; Blumenfeld et al., 2011; Thomas, 2000). Students in PjBL are expected to engage in active learning experience, develop higher-order thinking skills by working with real-world projects, and enhance subject knowledge. Their learning outcomes involve cognitive aspects (e.g., the quality of artifacts or solutions to real-world problems, higher-order thinking skills, and subject knowledge) and affective aspects such as intrinsic motivation (Guo et al., 2020).

More recently, with the support of information and communication technology (ICT), PjBL has been extended from face-to-face to online learning environments to support student-centered flexible learning with projects. ICT helps students and teachers to access project information, enable students to perform online learning with a project, allow students to construct and manipulate their artifacts in flexible digital presentation, and help teachers to monitor project progress and provide feedback to students (Blumenfeld et al., 2011). In PjBL courses, asynchronous online discussions can be used to support group discussions during the project (Koh et al., 2010); wikis can provide effective support for PjBL by facilitating collaborative real-time editing of project output and tracking of revision history (Chu et al., 2017); and computer-based prompts can be applied to facilitate group collaboration and reflection throughout the project (Splichal et al., 2018).

Although technology can support PjBL in multiple aspects, it can't solve all problems. Online learning environments in most situations are not well designed to make the complex PjBL process accessible to students (English & Kitsantas, 2013; Leyer et al., 2023). While PjBL is extended from the classroom to online settings, students have difficulties receiving prompt support from teachers to complete complex problem-solving activities in PjBL. The challenge is more serious for low-achieving students who often have inadequate higher-order thinking skills to complex PjBL processes. Recent studies explored technology-supported visible thinking approaches, i.e., the use of visual representations (e.g., graphs, concept maps, mind maps, flowcharts, causal maps, system models) to externalize complex ideas for effective thinking and communication in learning with real-world problem-solving projects or tasks (Gijlers & de Jong, 2013; Slof et al., 2012; Wang et al., 2018). Visible thinking approaches have a potential to make complex processes accessible to low-achieving students to engage them in effective learning. Nevertheless, it remains unknown whether and how such visible thinking approaches may benefit students of different levels of academic achievement (e.g., low, medium, and high achievers), who may differ in their ability to complete the tasks requiring higher-order thinking skills (Zohar et al., 2001).

To address the gap, this study incorporated a visible thinking approach into online PjBL of computer programming. We investigated whether and how students of different levels of academic achievement might benefit differently from the visible thinking approach to PjBL of programming in an online environment.

1.1 Challenges in implementing PjBL

Recent reviews of empirical studies in PjBL reported the positive effects of PjBL on improving students' knowledge, skills, and motivation (Chen & Yang, 2019; Guo et al., 2020). These reviews also reveal some difficulties in implementing PjBL. Different from traditional education, PjBL involves a wide range of problem-solving activities and extensive hands-on practices. Students in PjBL need to pursue solutions to real-world problems by investigating problems, exploring solutions, drawing conclusions, and creating artifacts in a variety of forms such as writings, drawings, videos, and technology-based presentations (Blumenfeld et al., 2011; Guo et al., 2020). Many students have difficulties completing these complex activities without necessary support. For example, Stewart (2007) found that students who lack self-directed learning skills may not achieve prospective learning outcomes. Sánchez-García and Pavón-Vázquez (2021) reported that students complained about dealing with complicated information of a project. Meanwhile, teachers reported the challenges in designing PjBL curricula and in supporting students during PjBL (e.g., assessing students' progress, diagnosing their problems, and providing feedback to students) (Blumenfeld et al., 2011). As a result, PjBL is often not fully implemented in educational practices (Blumenfeld et al., 2011; Guo et al., 2020). Moreover, while students in PjBL are expected to achieve learning outcomes in multiple aspects such as product quality, thinking skills, subject knowledge, and affective experiences, many studies adopting PjBL failed to demonstrate these outcomes (Guo et al., 2020).

1.2 High- and low-achieving students in PjBL

Working with a real-world project often involves a complex process requiring higher-order thinking skills such as understanding problems, exploring and refining ideas, designing plans, analyzing data, drawing conclusions, and implementing solutions. In PjBL courses, many students, especially low achievers, have little knowledge about the complex process, which is difficult to predefine since there is no single algorithm for solving complex problems or tasks (Hmelo-Silver et al., 2007; Kirschner et al., 2006; Sasson et al., 2018; Wang et al., 2018). They have inadequate higher-order thinking skills to perform complex activities. As a result, many students may feel overwhelmed and unable to engage in effective learning and achieve desirable learning outcomes in PjBL contexts (Blumenfeld et al., 2011; Helle, Tynjälä, & Olkinuora, 2006; Thomas, 2000). This makes PjBL a challenge to many students, especially in online settings without prompt support from teachers. In particular, the problem may compromise the learning of low-achieving students, who often lack higher-order thinking skills.

Research shows that many students (especially low achievers) have difficulties completing a real-world project in PjBL courses (Jazayeri, 2015). Many teachers found it difficult to teach problem-, project-, or inquiry-based curricula (Blumenfeld et al., 2011; Pucher & Lehner, 2011). They claimed that the tasks involving higher-order thinking are appropriate mainly for high-achieving students. Such situations echo inconclusive findings on the outcomes of adopting PjBL (Blumenfeld et al., 2011; Guo et al., 2020). Students of different levels of academic achievement tend to differ in their ability to complete complex learning tasks; low-achieving students tend to have inadequate ability to perform such tasks (Zohar et al., 2001). However, all students should have an equal chance to engage in effective learning with real-world projects, which is important for the equality of opportunity in education (Noguera et al., 2015). Low-achieving students should not be deprived from learning tasks requiring higher-order thinking skills (Zohar et al., 2001). Educators are expected to explore approaches to improve students' c skills and narrow the gaps between high and low achievers (Mitani, 2021).

In PjBL of computer programming, students are encouraged to apply abstract programming knowledge (e.g., concepts, syntax, semantics) to real-world programming projects (Pucher & Lehner, 2011; Sun et al., 2022). However, low-achieving students often have difficulties acquiring abstract programming knowledge. Moreover, they have difficulties mastering the strategies and skills for applying programming knowledge to realistic projects because such strategies and skills are often implicit and hard to capture (Robins et al., 2003; Soloway, 1986; Xinogalos, 2016). In such contexts, low-achieving students may feel overwhelmed and unable to engage in effective learning with programming projects (Jazayeri, 2015; Helle et al., 2006; Thomas, 2000).

1.3 Visible thinking for complex learning with real-world projects

To help learners to accomplish complex learning with real-world projects or problem-solving tasks, researchers highlight the importance of providing students with necessary guidance and support to help them engage in higher-order thinking and effective learning (Hmelo-Silver et al., 2007; Kirschner et al., 2006). The commonly used approaches include teaching students about discipline-specific strategies for thinking and reasoning, structuring or decomposing complex tasks, using prompts or hints to facilitate the task process, and encouraging collaborative inquiry and problem-solving (Belland et al., 2016; Hmelo-Silver et al., 2007; Lazonder & Harmsen, 2016; Reiser, 2004; White & Frederikse, 1998).

Recent research explored the use of visible thinking approaches, i.e., the use of visual representations (e.g., graphs, concept maps, mind maps, flowcharts, causal maps, system models) to externalize complex ideas for effective thinking and communication, which has shown promising effects on improving students' knowledge and performance in learning with real-world problem-solving projects (Gijlers & de Jong, 2013; Slof et al., 2012; Wang et al., 2018). By representing abstract issues and complex processes in visible forms (with computer-based tools), visible thinking approaches can support meaningful understanding of complex knowledge and facilitate higher-order thinking and reasoning with complex problems or projects. For example, Hsu et al. (2015) incorporated a graph-oriented, computer-assisted application in a science classroom to support collaborative argumentation among students in PjBL, which showed a positive impact on student development of science knowledge and scientific argumentation skills. Moreover, the study of Chen et al. (2021) reveals that students' higher-order thinking skills reflected in a reasoning map can predict their problem-solving performance in scientific inquiry. In addition to academic achievements, visual thinking approaches can enhance students' motivation and emotional experience by increasing self-efficacy or confidence, reducing anxiety, and increasing enjoyment of learning with complex tasks (Hall & O'Donnell, 1996; Laight, 2004; Sung & Hwang, 2013; Wang et al., 2018; Yuan et al., 2020). With these affordances, visible thinking approaches can engage learners, foster higher-order thinking, and sustain motivation to learn with complex tasks, which are crucial to PjBL especially in online environments.

In programming education, visible thinking approaches such as diagrams, pictures, animations, and simulations have been used to engage learners, assist in their understanding of abstract concepts and complicated behavior of programs, and enhance their intrinsic motivation to learn complex programming (Eisenberg et al., 2014; Fanchamps et al., 2021; Hundhausen & Brown, 2007; Jamil & Isiaq, 2019; Jerez et al., 2012; Naps et al., 2003; Peng et al., 2019; Sorva et al., 2013). With respect to PjBL of computer programming, Peng et al. (2019) explored a visible thinking approach to externalize the complex process of completing a real-world programming project in an online learning environment and demonstrated its effects on improving students' project performance.

Although the literature reported promising advantages of visualization approaches in engaging programming learners, inconclusive findings were found with respect to the effects of these approaches on improving learners' programming performance (Rajala et al., 2008; Sorva et al., 2013). It remains unknown whether and how visible thinking approaches may benefit a wide range of students of different levels of academic achievement (e.g., high, medium, and low), who may differ in their ability to complete the tasks requiring higher-order thinking skills.

1.4 Affective experiences in PjBL

Affective experiences in educational contexts mainly consider motivational and emotional experiences. Regarding *motivation*, it mainly concerns beliefs and attitudes that drive students to engage in working toward their learning goals, with more attention to intrinsic motivation. Intrinsic motivation is an inner drive that propels a person to pursue an activity for its own sake and not because of external factors like reward or punishment (Cameron & Pierce, 1994). Intrinsic motivation can be measured in terms of interest/enjoyment, perceived competence, effort/importance, value/usefulness, pressure/tension, and perceived choices (McAuley et al., 1989). Among them, perceived competence appears to be particularly salient in student-centered learning contexts like PjBL since students' competence may influence the degree to which they persist in learning when facing challenges. With respect to *emotion*, it encompasses learners' positive and negative reactions to the learning experience. Enjoyment is a common positive emotion, whereas anxiety or tension is a typical negative emotion (Fredricks et al., 2004; Pekrun et al., 2011). Enjoyment and anxiety are also related to intrinsic motivation and are included in the intrinsic motivation inventory model proposed by McAuley et al. (1989) to measure intrinsic motivation in terms of interest/enjoyment, pressure/tension, perceived competence, effort/importance, and value/usefulness. In this study, we included these key elements to analyze students' affective experiences in PjBL.

Compared to traditional education, students are more likely to be motivated in PjBL as they are encouraged to apply abstract knowledge to contextualized real-world projects (Guo et al., 2020). However, many students experience difficulties in PjBL as mentioned above. The problems may make students feel discouraged and frustrated and may influence their motivation and learning outcomes (Blumenfeld et al., 2011). Prior research reveals that affective experiences are closely intertwined with cognitive experiences (Phelps 2006; Schutz & DeCuir 2002) and are significantly related to learning achievements (Pekrun et al., 2011). If a learning task is too complex, students may have difficulties engaging in effective thinking; further, they may feel frustrated and anxious, and have a lack of confidence and motivation to persist in learning. Such negative affective experiences can impede cognitive processes, whereas positive affect can foster thinking and learning (Pekrun et al., 2011). Although PjBL can promote student motivation, it is important to sustain students' intrinsic motivation when they face challenges in completing the complex process of PjBL. It is therefore important to analyze students' affective experiences as an important part of a whole picture of student learning in PjBL.

1.5 The present study

PjBL has been applied to educational practice across multiple levels (e.g., primary, secondary, and tertiary) and in various subject areas including engineering, science, mathematics, and social science (Chen & Yang, 2019; Pucher & Lehner, 2011; Ralph, 2015; Reis et al., 2017). This study focused on PjBL of computer programming, an important subject in engineering. PjBL is promoted in engineering education and regarded as among the most suitable means of developing students' professional competencies in response to society's demands on engineering professionals (De los Ríos et al., 2010). In programming courses, PjBL enables students to master abstract programming knowledge by applying it to real-world programming projects, mainly through developing computer programs (Pucher & Lehner, 2011).

PjBL involves complex processes requiring higher-order thinking skills, which pose challenges to many students, especially low achievers, who often have inadequate higher-order thinking skills. Visible thinking approaches have the potential to make complex thinking processes accessible to students to engage them in effective learning. This study incorporated a visible thinking approach into online PjBL of programming. Despite the effects of visualization approaches in programming education, the literature reports inconclusive findings on the effects of such approaches on improving learners' programming performance (Rajala et al., 2008; Sorva et al., 2013). Regarding the aforementioned visible thinking approach to support PjBL, there is little research investigating whether and how the approach may benefit a wide range of students of different levels of academic achievement (e.g., high, medium, and low), who may differ in their ability to complete PjBL tasks requiring higher-order thinking skills.

This study aimed to address the gap by answering the following research questions.

1. Do high-, medium-, and low-achieving students differ in their learning outcomes (reflected in product quality, thinking skills, and subject knowledge) acquired from visible thinking supported online learning with a programming project? If so, what are the differences?
2. Do high-, medium-, and low-achieving students differ in their affective experiences acquired from visible thinking supported online learning with a programming project? If so, what are the differences?

2 Method

To answer the research questions, a one-group pretest-posttest design was adopted to compare the learning outcomes between high-, medium- and low-achieving students. Based on the literature, the levels of academic achievement among students are often determined based on knowledge exam or test scores (Zohar et al., 2001). Students in this study were categorized into *high-, medium-, and low-achieving groups* according to their pre-study knowledge test scores.

2.1 Participants

The study was conducted in an ordinary university in southern China. The study received the ethical approval from the Human Research Ethics Committee of the researchers' university. The participants were 72 Year-3 students (46 males and 26 females) of the computer science program in this university. Their average age was

21.0 years. The students gave informed consent to participate in this study on PjBL of computer programming. The participants were offered a 6-week course aimed at helping senior-year students develop authentic programming skills required for programming related jobs. The course was designed by the research team and taught by a teacher at the university. It was not included in but closely aligned with the university's curriculum. ASP.NET was selected as the programming language since it is a popular programming language and a widely used web application framework for developing dynamic modern web applications and services. Before the study, the participants had studied relevant programming courses and obtained online learning experience. All the participants completed all the activities of this study.

2.2 Learning task

The participants were asked to complete an authentic programming project—membership management. They need to develop a computer program that can be used for member registration, password setting and resetting, user login, login validation, and update of member information. To complete the project, students need to go through a set of phases including problem understanding, solution planning (modular design), solution design (process design), solution implementation (coding), and evaluation and reflection. In each phase, students need to submit relevant learning artifacts such as problem statement, modular design diagram, program flowchart, and program code. In each phase, students are given heuristics (or rules of thumb) and tools that help them to accomplish each phase.

2.3 Online learning environment for visible thinking supported PjBL

In this study, students performed PjBL of computer programming in an online environment (see Fig. 1). A visible thinking approach developed in a prior study (Peng et al., 2017; 2019) was applied in the online learning environment to make the complex process of PjBL accessible to students. The visible thinking approach was designed based on the four-component instructional design (4C/ID) model, a framework for systematic learning with complex tasks (Van Merriënboer & Kirschner, 2017). In this model, learners are provided with a systematic approach to problem solving. The mechanism of the approach includes (a) specifying the phases a learner must go through to solve a complex problem or complete a realistic project, and (b) providing the rules-of-thumb or heuristics that may help the learner to accomplish each phase.

This study incorporated the visible thinking approach by (a) externalizing the complex process of completing a programming project as five main phases, namely *problem understanding*, *solution planning (modular design)*, *solution design (process design)*, *solution implementation (coding)*, and *evaluation and reflection*; and (b) providing relevant heuristics (e.g., formulating a problem statement by specifying the project requirements and goals) and tools (e.g., specifying the project requirements and goals in a semi-structured form) that may help learners to complete individual phases. The five phases and relevant heuristics were proposed based

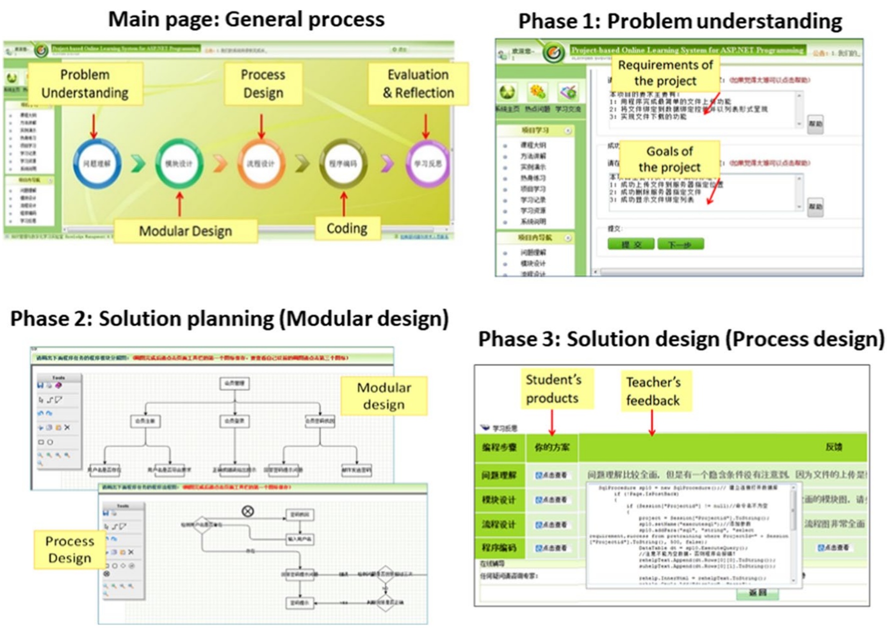


Fig. 1. Visible thinking approach implemented in the online learning environment

on the methods and principles for computer programming projects (Deek et al., 1999). They form a visible thinking approach for PjBL of programming by making the complex implicit aspects of the project visible or accessible to learners. Figure 1 depicts the implementation of the visible thinking approach in the online system, which is elaborated in the following text.

After logging in the system, students opened the main page, where they could view the general process of completing a programming project as five phases. After a click on the icon of each phase, students could enter the space of each phase to perform relevant activities in the system. The details of each phase are described below with illustrating examples shown in Fig. 2.

Problem understanding At the beginning of the project, students were asked to formulate a problem statement for the project. In this phase, students were presented with relevant heuristics that suggest specifying the project requirements and project goals in a semi-structured form. Figure 2.A shows an example of a student's statement of the requirements (e.g., Users can update their personal information at any time after registration) and goals (e.g., After a successful login, users will go to the main page) of the project on membership management.

Solution planning (modular design) Based on the understanding of the project requirements and goals, the students were expected to generate a solution plan. The learning system provided relevant heuristics or guidelines on proposing a set of functional modules and specifying the relationships between the modules. The

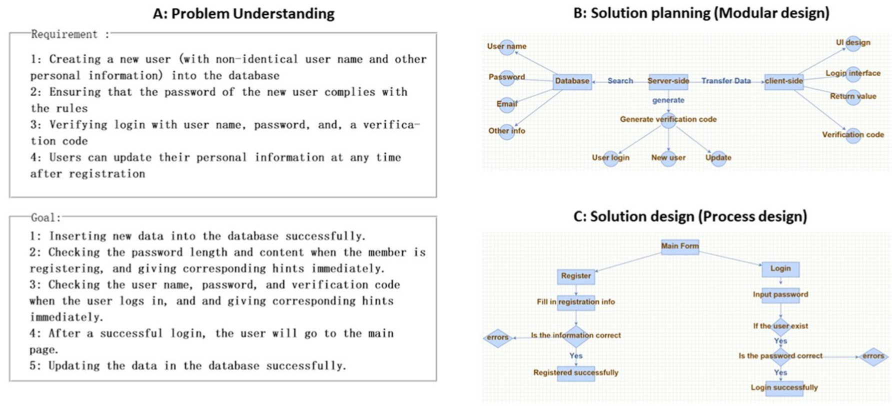


Fig. 2. Examples of student-generated artifacts during the project

learning system also provided a diagramming tool for students to outline the modular design in a modular block diagram. Figure 2.B shows an example of the solution plan, which was presented as a set of functional modules including client-side interfaces, verification code generation, and search for information in the database.

Solution design (process design) Based on the modular design, the students were asked to generate a detailed design of the solution by building a program flowchart. They were given relevant guidance on how to design a solution process by decomposing functional modules and organizing the process within and across the functional modules. The system provided a diagramming tool to build the flowchart. Figure 2.C demonstrates an example of the solution process related to member registration and member login; the process of member registration included filling in the information, validating information, and returning the outputs.

Solution implementation (coding) To implement the solution, students need to translate the modular design and process design into an executable program by writing source code in ASP.NET, a programming language. Students could submit their programs using an online coding tool in the system and modify their programs throughout the project.

Evaluation and reflection After coding, students need to evaluate their programs by testing and debugging their codes. They can also review and modify their artifacts (e.g., modular block diagram, flowchart) generated in previous phases. In the meantime, students will receive the teacher's comments and feedback to their artifacts in each phase as shown in Fig. 1.d. In this way, they could reflect on their strengths and weaknesses, and then refine their products until an acceptable solution to the project was achieved.

2.4 Procedure

The PjBL course lasted for six weeks, with the detailed activities shown in Table 1. In the first week, students signed the consent form and completed a questionnaire to collect their demographic information about age and gender. Then, students received a one-hour face-to-face instruction on how to use the online learning environment to complete a programming project. A sample project was provided for illustration. Afterwards, the students completed a knowledge test and a programming task to assess their programming knowledge and programming performance before the study.

From week 2 to week 5, students performed independent learning with a programming project—membership management. They were asked to learn at their own pace and spent at least 2 hours per day on the project. In the meantime, the teacher monitored students' progress and provided comments and feedback to their artifacts via the online system. Face-to-face consultation was arranged in week 3 and week 5, with one hour for each session.

In week 6, students completed the post-study knowledge test and programming task. In addition, a questionnaire survey was administered to collect students' affective perceptions and their responses to open-ended questions about their comments on the learning program.

2.5 Measures and instruments

In this study, student performance in the programming project was assessed based on the model proposed by Deek et al. (1999). This model is a classical programming model, which has been widely used in empirical studies on computer programming or engineering education (e.g., Gul et al., 2023; Felder & Brent, 2010; López-Pimentel et al., 2021; Norton et al., 2007). As claimed by Deek et al. (1999), problem-solving skills are an integral part of the knowledge that students should acquire from programming learning; the assessment of programming learning should go beyond subject knowledge from a language construct view (e.g., syntax) by paying more attention to higher-order thinking skills for problem solving, which may involve understanding or formulating the problem, planning the solution, designing the solution, and implementing the solution. Based on this model, the assessment of student programming performance considers two distinct categories: the programming process and the coding of the program. The former reflects higher-order thinking involved in the design and development of the program, while the latter represents the solution or product. Accordingly, students' programming performance in this study was examined in the following two aspects.

Product quality The quality of student-generated products (program codes) was assessed by a programming task before and after the study. The two tasks were designed to be practical and moderately difficult. The two tasks were different in content but at the same level of difficulty as determined by the experienced teacher

Table 1 Main activities in the PjBL course

Week	Learning activity	Data collection
Week 1	Face-to-face instruction to get familiar with the learning environment (1 hour)	• A questionnaire to collect demographic information • A pre-study knowledge test • A pre-study programming task to assess product quality and programming thinking skills
Weeks 2-5	Self-directed online learning to complete the project (at least two hours per day)	Nil
	Face-to-face consultation in Week 3 and Week 5 (one hour for each session)	Nil
Week 6	Nil	• A post-study knowledge test • A Post-study programming task to assess product quality and programming thinking skills • A questionnaire to collect students' affective perceptions and their comments on the course

and the industry expert. Based on the programming assessment model proposed by Deek et al. (1999), the program codes were assessed in terms of correctness, efficiency, reliability, and readability. The score ranged from 0 to 10 for each of the four subscales, and the maximum score for the product (code) quality was 40 points. The assessment applied the rubrics proposed in the study of Peng et al. (2019) with the details provided in Appendix 1 (at the end of the manuscript).

Programming thinking skills The assessment of programming thinking skills or programming process concerns (a) problem understanding, (b) solution planning (i.e., modular design), and (c) solution design (i.e., process design), which reflect higher-order thinking skills required to complete a programming task. In this study, in addition to program code, students were required to submit a problem statement, modular design diagram, and program flowchart to assess their programming thinking skills on the three aspects. The maximum score was 20 points for each of the three subscales. The assessment applied the rubrics proposed in the study of Peng et al. (2019) with the details provided in Appendix 2 (at the end of the manuscript).

Knowledge tests The students' programming knowledge was assessed before and after the study. The questions in both tests were adapted from relevant textbooks used by the university. The Cronbach alphas for pre- and post-test were .67 and .71, respectively. Their validity was confirmed by an experienced programming teacher and an industry expert in computer programming. The two tests used different questions, but at the same level of difficulty as determined by the domain experts. Each test included single-choice questions, fill-in-the-blank questions, and short program-writing questions. An example is provided in Appendix 3 at the end of the manuscript. The maximum score for each test was 100 points.

Affective experiences A questionnaire survey was designed based on the intrinsic motivation inventory model (McAuley et al., 1989) to assess students' affective experiences in terms of interest/enjoyment, perceived competence, effort/importance, value/usefulness, and pressure/tension at the end of the study. The questionnaire used a 5-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree). Examples of the items include "I enjoyed attending this project-based programming course"; "I felt confident during the learning of this course"; "I put a lot of effort into this course"; "I think this course is useful"; and "I got nervous while studying in this course." Cronbach's alpha values (.76 for interest/enjoyment, .77 for effort/importance, .75 for value/usefulness, .79 for perceived competence, and .86 for pressure/tension) on internal consistency confirm the reliability of the sub-scales.

Student comments Students' comments on the visible-thinking approach and project-based online course were collected for triangulation in data analysis. The participants were asked to give written responses to two open-ended questions: (1) What are your views on the advantages of the learning program? (2) What are your views on the weaknesses of the learning program?"

2.6 Data analysis

The following methods were used to analyze the collected data. *First*, two programming experts (an experienced teacher and an industry expert) graded the students' knowledge tests, programming thinking skills, and products (codes) blindly and independently based on the reference answers and solutions validated by the two experts. The inter-rater reliability measured using Cohen's Kappa ranged from 0.832 to 0.967 (see Table 2), suggesting a high level of agreement. The two raters' scores were averaged for analysis.

Second, after students were categorized into high-, medium-, and low-achieving groups according to pre-study knowledge test scores, a set of ANOVAs was conducted to evaluate the differences among students of three different academic levels in terms of knowledge test scores, product quality, thinking skills, gain scores (i.e., the difference between pre- and post-test scores), and affective experiences. The Levene's test was used to check the homogeneity of variances across groups. When the assumption was not violated, one-way ANOVA with Scheffe's post hoc test was conducted to compare the difference between each pair of the three levels. When the assumption was violated, Welch ANOVA with Games-Howell post hoc test was performed.

Fourth, a thematic content analysis was performed to probe common themes in students' responses to the open-ended question. The analysis followed an iterative process of code and theme generation in a bottom-up manner. The first author and a trained researcher coded 30% of the response data under each of the two questions. Student responses were categorized into a set of themes. Discrepancies between the two coders in the themes emerging from the responses were discussed between the two coders and reconciled by further consultation of the data. After consensus was reached, the two coders independently coded 20% of the dataset and the inter-coder agreement of the coding results was .99.

Table 2 Inter-rater reliability

Measures	Cohen's kappa coefficient	*p*
Pre-study knowledge tests	.911	< .001
Post-study knowledge tests	.927	< .001
Pre-study product quality	.884	< .001
Post-study product quality	.875	< .001
Pre-study thinking skills		
in Problem understanding	.967	< .001
in Modular design	.917	< .001
in Process design	.966	< .001
Post-study thinking skills		
in Problem understanding	.921	< .001
in Modular design	.832	< .001
in Process design	.892	< .001

After the differences in their coding results were discussed and resolved, all the response data were then coded by the first author based on the confirmed coding framework.

3 Results

The descriptive statistics of students' pre- and post-study knowledge test scores, product quality, thinking skills, and affective experiences are presented in Table 3. Skewness and kurtosis values of the observed variables were within acceptable ranges for being normally distributed (George & Mallery, 2010).

3.1 Differences in academic achievements among high, medium, and low achievers

Students were categorized into high-, medium-, or low-achieving groups according to their pre-study knowledge test scores. The 27 percent rule from the extreme group approach (Preacher, 2015; Preacher et al., 2005) was adopted to select the cut-off points. In particular, students with scores in the top 27% were assigned to the high-achieving group (n = 19); those in the bottom 27% to the low-achieving group (n = 19); and the rest to the medium-achieving group (n = 34).

Subject knowledge According to the ANOVA results shown in Table 4, there was a significant difference among high-, medium-, and low-achieving groups in their pre-test scores (Welch's F (2, 37.25) =122.13, p < .001) and post-test scores (F (2,

Table 3 Descriptive statistics of variables

Variable	Min	Max	Mean	*SD*	Skewness	Kurtosis
Pre-study knowledge	11	95	41.98	17.56	0.87	0.81
Pre-study problem understanding	5	20	12.9	3.78	0.24	-0.36
Pre-study modular design	5	20	12.85	3.88	0.25	-0.50
Pre-study process design	5	20	12.79	3.74	0.20	-0.56
Pre-study product quality	5	35	17.65	5.95	1.00	1.14
Post-study knowledge	23	100	63.42	18.66	-0.01	-0.64
Post-study problem understanding	10	20	18.17	2.37	-1.31	1.49
Post-study modular design	11	20	17.55	2.29	-0.50	-0.50
Post-study process design	11	20	17.77	2.5	-0.84	-0.26
Post-study product quality	15	38	29.77	4.9	-0.68	0.28
Interest/Enjoyment	3	5	4.32	0.61	-0.38	-0.88
Perceived competence	2.75	5	4.24	0.59	-0.36	-0.20
Effort/Importance	2	5	4.3	0.64	-0.45	-0.73
Pressure/Tension	1	5	2.69	0.99	0.56	-0.13
Value/Usefulness	3	5	4.4	0.51	-0.15	-0.81

Table 4 ANOVA of students' pre- and post-study knowledge test scores

Subject knowledge	Low-achieving group		Medium-achieving group		High-achieving group		*Levene's test (p)*	*F*	Post hoc test
	Mean	*SD*	Mean	*SD*	Mean	*SD*			
Pre-test	23.11	4.42	39.6	6.78	65.11	12.69	.003	122.13***	High>Medium; High>Low; Medium>Low
Post-test	49.68	12.13	61.76	16.47	80.11	15.17	.452	19.66***	High>Medium; High>Low; Medium>Low
Gain	26.58	11.95	22.17	18.07	15.00	12.71	.039	4.16*	Low>High

****p* < .001

**p* < .05

Table 5 ANOVA of students' pre- and post-study product quality

Product quality	Low-achieving group		Medium-achieving group		High-achieving group		*Levene's test (p)*	*F*	Post hoc test
	Mean	*SD*	Mean	*SD*	Mean	*SD*			
Pre-study	12.42	2.99	17.12	3.59	23.82	6.14	.004	30.05***	High>Medium; High>Low; Medium>Low
Post-study	28.53	6.1	29.53	4.32	31.45	4.3	.151	1.81	
Gain	16.11	5.48	12.41	4.57	7.63	7.25	.172	10.90***	Low>High; Medium>High

****p* < .001

69) =19.66, p < .001). Although the post hoc comparisons revealed that the gaps between the low-, medium-, and high-achieving groups were still significant at the end of study, the three groups differed in their knowledge gain (Welch's F (2, 43.25) = 4.16, p = .022); low achievers gained more knowledge from the learning program than high achievers did.

Product quality The ANOVA of students' pre- and post-study product quality (see Table 5) indicated that students with different levels of prior knowledge differed in their pre-study product quality (Welch's F (2, 36.66) =30.05, p < .001). However, there was no significant difference among the three groups in their post-study product quality (F (2, 69) =1.81, p > .05). The ANOVA result revealed that the three groups of students differed significantly in their gain in product quality (F (2, 69) = 10.90, p < .001). Post hoc comparison using Scheffe's test showed that both low- and medium-achieving groups made more progress than high-achieving ones in product quality.

Thinking skills Students' programming thinking skills were assessed in terms of problem understanding, modular design, and process design. The ANOVA results shown in Table 6 indicated that high-, medium-, and low-achieving students differed in all the three dimensions of thinking skills at the beginning of the study; however, the discrepancies narrowed at the end of the study. Specifically, the results indicated that at the end of the study the three groups of students differed in problem understanding (Welch's F (2, 39.23) =9.16, $p < .001$) and modular design (F (2, 69) =7.67, $p < .001$), but not in process design (Welch's F (2, 34.15) =1.67, $p = .204$).

Regarding *problem understanding*, the Games-Howell post hoc analysis showed no difference between medium- and low-achieving groups at the end of the study. The analysis of gain scores revealed a significant difference among the three groups (F (2, 69) = 9.20, $p < .001$), with low- and medium-achieving students making more progress than high-achieving ones in this dimension.

Regarding *modular design*, the Scheffe's post hoc test showed that the only difference among the three groups was that high-achieving students scored higher than low-achieving students. The analysis of gain scores showed that low-achieving students made more progress than medium- and high-achieving ones in this dimension.

With respect to *process design*, the Scheffe's post hoc test of gain scores showed that low-achieving students made the most progress, while high-achieving students make the least progress among the three groups in this dimension.

3.2 Differences in affective experiences among high, medium, and low achievers

The ANOVA results shown in Table 7 indicate that students of different achievement levels had similar affective experiences in terms of interest/enjoyment (F (2, 69) = 2.41, $p > .05$) and perceived pressure/tension (F (2, 69) =0.10, $p > .05$). However, there were some differences among the three groups of students in terms of perceived competence (F (2, 69) = 5.73, $p < .01$), effort/importance (F (2, 69) =3.23, $p < .05$), and value/usefulness (F (2, 69) = 4.00, $p < .05$). Based on Scheffe's post hoc test results, low-achieving students had higher scores than medium-achieving students in these three aspects.

3.3 Differences in comments among high, medium, and low achievers

The results of analysis of the students' responses to the survey question "What are your views on the advantages of the learning program?" are presented in Table 8, which demonstrates the themes, illustrative examples, and the frequency of each theme. Generally, they felt the learning program to be effective for self-directed learning (61%), improving problem-solving skills (54%), and supporting the motivation for learning programming (43%). On the other hand, it was interesting to note that most positive comments were from the low-achieving group. They appreciated the visible thinking approach provided in the system (63%); they found the course effective for knowledge acquisition (42%). In contrast, the high-achieving students

Table 6 ANOVA of students' pre- and post-study thinking skills

Thinking skills	Low-achieving group		Medium-achieving group		High-achieving group		*Levene's test (p)*	*F*	Post hoc test
	Mean	*SD*	Mean	*SD*	Mean	*SD*			
Pre-test									
Problem Understanding	9.68	2.34	12.56	2.72	16.74	3.28	.021	28.96***	High>Medium; High>Low; Medium>Low
Modular Design	9.00	2.13	12.53	2.33	17.26	2.92	.099	54.58***	High>Medium; High>Low; Medium>Low
Process Design	8.84	1.92	12.91	2.35	16.53	3.19	.058	44.87***	High>Medium; High>Low; Medium>Low
Post-test									
Problem Understanding	16.63	2.73	18.31	2.17	19.47	1.31	.013	9.16***	High>Low; High>Medium
Modular Design	16.20	2.46	17.56	2.06	18.87	1.76	.611	7.67**	High>Low
Process Design	16.84	2.93	18.24	2.03	17.87	2.70	.029	1.67	
Gain									
Problem Understanding	6.95	2.27	5.75	3.39	2.74	3.44	.248	9.20***	Low>High; Medium>High
Modular Design	7.20	2.37	5.03	2.61	1.60	3.40	.341	19.63***	Low>High; Low>Medium
Process Design	8.00	3.06	5.32	3.26	1.34	4.73	.327	16.06***	Low>High; Low>Medium; Medium>High

***$p < .001$

**$p < .01$

Table 7 ANOVA of students' affective experiences

Affective experiences	Low-achieving group		Medium-achieving group		High-achieving group		*Levene's test (p)*	*F*	Scheffe's post hoc test
	Mean	*SD*	Mean	*SD*	Mean	*SD*			
Interest/Enjoyment	4.56	0.58	4.19	0.63	4.32	0.61	.633	2.41	
Perceived Competence	4.61	0.42	4.07	0.62	4.17	0.57	.626	5.73**	Low>Medium
Effort/Importance	4.60	0.47	4.15	0.70	4.28	0.60	.275	3.23*	Low>Medium
Value/Usefulness	4.66	0.42	4.26	0.50	4.38	0.52	.722	4.00*	Low>Medium
Pressure/Tension	2.72	0.99	2.64	1.06	2.75	0.88	.798	0.10	

**$p < .01$

*$p < .05$

highlighted the advantages regarding professional skills of solution planning and design (47%) and knowledge-practice integration (32%). Compared with the other two groups, the medium-achieving group reported different aspects of benefits more evenly.

Regarding students' responses to the survey question ("What are your views on the weaknesses of the learning program?"), Table 9 presents the analysis results. Most students reported technical problems (40%) of the learning system. In addition, the low-achieving students mentioned the difficulty in learning, while the high- and medium-achieving students indicated the lack of teacher-student interaction and learning resources.

4 Discussion

Previous studies didn't investigate whether and how visible thinking approaches may benefit a wide range of students of different levels of academic achievement. The results of this study shed light on how students of varied academic levels perform differently in a visible thinking supported PjBL context. In this section, we discussed the findings on the differences among high-, medium-, and low-achieving students in their learning outcomes and affective experiences acquired from the visible thinking supported PjBL program.

4.1 Differences in academic achievements among high, medium, and low achievers

Subject knowledge It was found that the gaps in subject knowledge among low-, medium-, and high-achieving students still existed at the end of the study. However, the gain scores showed that low achievers gained more knowledge from the PjBL

Table 8 Student comments on the advantages of the learning program

Theme	Illustrative example	Frequency			
		Low-achieving group (N=19) K (%)	Medium-achieving group (N=34) K (%)	High- achieving group (N=19) K (%)	Total (N=72) K (%)
Self-directed learning	(This online course) is flexible. I can study at any time, any place, at my own pace.	13(68%)	19 (56%)	12 (63%)	44(61%)
Time management	I have improved my ability to control time in the process of programming.	5(26%)	7(21%)	5(26%)	17(24%)
Reflection	(This course) helps me to be aware of my problems in computer programming and make improvement accordingly.	6(32%)	8(24%)	5(26%)	19(26%)
Scaffolding for learning	(The scaffolding for learning) made me feel that there seemed a teacher around to help me.	12(63%)	7(21%)	4(21%)	23(32%)
Knowledge acquisition	It also enables me to learn more knowledge that was not taught in traditional classes.	8(42%)	5(15%)	5(26%)	18(25%)
Knowledge-practice integration	(This course) combines knowledge with practice, enabling us to master various skills effectively.	5(26%)	9(27%)	6(32%)	20(28%)
Problem-solving ability	My problem-solving ability in computer programming has been improved. I can propose solutions to problems by myself instead of passively waiting for teacher's instruction.	14(74%)	13(38%)	12(63%)	39(54%)
Solution planning and design	(This course) enables me to develop the habit of modular design when developing a computer program.	7(37%)	8(24%)	9(47%)	24(33%)
Motivation for learning	This way of learning is very good to elicit our enthusiasm for learning programming.	10(53%)	11(32%)	10(53%)	31(43%)

N = number of all students. K = number of students giving responses under each theme.

% = K/N

Table 9 Student comments on the weaknesses of the learning program

Theme	Illustrative example	Frequency			
		Low-achieving group (N=19) K (%)	Medium-achieving group (N=34) K (%)	High- achieving group (N=19) K (%)	Total (N=72) K (%)
Technical problems	There were some bugs occurring when I drew the flowchart.	9(47%)	15(44%)	5(26%)	29(40%)
Learning difficulty	It is a bit difficult for poor and average students in the early stage of the course.	8(42%)	7(21%)	3(16%)	18(25%)
Teacher-student interaction	It is not very convenient for students and teachers to communicate with each other during the course.	3(16%)	9(26%)	6(32%)	18(25%)
Learning resource	The learning resources were not adequate.	3(16%)	9(26%)	6(32%)	18(25%)
Supervision	I had to rely on myself due to inadequate supervision arranged by the teacher.	3(16%)	7(21%)	4(21%)	14(19%)

N = number of all students. K = number of students giving responses under each theme.

% = K/N

course than high achievers did. The result might be explained by the nature of PjBL that focuses on the application of subject knowledge to real-world projects. While knowledge-practice integration can help students to consolidate subject knowledge, it might not reduce the gap among low-, medium-, and high-achieving students in a short time. In this study, low-achieving students made more improvement in subject knowledge than high achievers did after the six-week PjBL program, but it may take more time for low achievers to perform as well as high achievers in the knowledge test. In addition, prior studies noted that learning outcomes in problem-solving contexts may not be directly reflected in traditional knowledge tests that lack sensitivity to learning in such contexts in their assessment criteria (Gijbels et al., 2005; Wu et al., 2016).

Product quality Our findings reveal that the gaps in the product quality among low-, medium-, and high-achieving students existed at the beginning, but almost disappeared at the end of the study. That is, low-achieving students performed almost as well as medium- and high-achieving students in the product quality after completing the PjBL course. The observed change is consistent with the gain scores in product quality, which show that low- and medium-achieving students made higher improvements in their product quality than high achievers did.

The findings on the quality of student products are consistent student progress in programming thinking skills and subject knowledge achieved from the proposed PjBL course. Completing a real-world project requires not only subject knowledge, but more importantly higher-order thinking skills for solving complex problems (Hmelo-Silver et al., 2007; Kirschner et al., 2006; Sasson et al., 2018; Wang et al., 2018). In programming education, researchers have also highlighted the importance of programming thinking skills in affecting students' programming performance or project quality (Blumenfeld et al., 2011; Deek et al., 1999; Gul et al., 2023; Pucher & Lehner, 2011). Compared to high achievers, low and medium achievers made more improvement in their programming thinking skills for *problem understanding*. In addition, low achievers made more progress than medium and high achievers in the programming thinking skills for *modular design* and *process design*. These differences support low achievers' biggest improvement in product quality. Besides, subject knowledge is an important foundation for completing a program product. As reported, low achievers made more improvement in subject knowledge than high achievers did, which is consistent with low achievers' great improvement in product quality.

Programming thinking skills It was found that the discrepancies among low-, medium-, and high-achieving students in programming thinking skills related to *process design* almost disappeared at the end of the study. There still existed discrepancies in *problem understanding* and *modular design* related skills among low-, medium-, and high-achieving students at the end of the study, but the discrepancies narrowed at the end of the study. The observed changes are consistent with the gain scores in programming thinking skills. The analysis of gain scores showed that low- and medium-achieving students made more progress than high-achieving ones in *problem understanding*; low-achieving students made more progress than medium- and high-achieving ones in *modular design* and *process design*.

On the contrary to the belief that weaker students are often not prepared for programming projects (Jazayeri, 2015), this study demonstrates that low-achieving students performed as well as high-achieving students in their task performance and thinking skills for *process design* in the visible thinking supported PjBL program. One possible explanation might be that high-achieving students have mastered high-level cognitive and metacognitive skills for handling complex tasks or projects, whereas low-achieving or weak students often lack such skills (Jazayeri, 2015; White & Frederiksen, 1998). Nevertheless, as long as effective tools are provided, students of all academic levels could engage in the tasks that involve higher order thinking (Zohar & Dori, 2003) and low achievers with inadequate skills for higher-order thinking tend to benefit more from the use of thinking tools (White & Frederiksen, 1998). The findings of our study indicate the importance of providing visible thinking tools to help low-achieving students to improve higher-order thinking skills for accomplishing complex projects. Regarding the findings on the thinking skills for *problem understanding* and *modular design*, educators and researchers may need to further investigate the needs of diverse students in PjBL. For example, besides the use of the visible thinking tool to externalize the complex process, there is a need to provide students with relevant strategies for completing project activities, e.g., how to formulate a problem statement by specifying project requirements and project goals, and how to generate a solution plan by proposing a set of interrelated functional modules.

4.2 Differences in affective experiences and comments among high, medium, and low achievers

The PjBL proposed in this study benefited students in their affective experiences, especially for low-achieving students. Compared to medium-achieving students, low-achieving students found the learning program more useful and were more motivated to work hard on it; they also felt more confident during the learning process.

Students' written comments support the findings. Low-achieving students' comments were the most positive among the three groups. Besides the advantages mentioned by most students (effective for self-directed learning, improving problem-solving skills, support the motivation for learning), low-achieving students highlighted that they appreciated the visible thinking approach provided in the learning system and found the course effective for knowledge acquisition. The high-achieving students highlighted the advantages regarding professional skills in solution planning and design and knowledge-practice integration.

Low-achieving students' more positive affective experiences are consistent with their larger improvement in product quality and thinking skills from the beginning to the end of the study, in comparison to high-achieving students. Results of the study echo the findings of previous research that visible thinking approaches can reduce students' anxiety and frustration while working with complex tasks (Corbalan et al., 2009; Sung & Hwang, 2013; Wang et al., 2018; Yuan et al., 2020). This study reveals that the advantages of visible thinking approaches can be more salient to low-achieving students, making them learn with more positive affect.

4.3 Limitations and future work

This study has several limitations. *First*, the study was based on a one-group pretest-posttest design, which may limit the generalizability of the findings to some extent. A quasi-experimental design can strengthen the conclusions of this study. *Second*, the participants of this study were from one university, which may constrain the generalization of the findings. Future studies could extend to students in other regions. In addition, the students participated in this study included more males than females, which is consistent with the gender gap in most engineering related fields. The moderating effect of gender in visible thinking supported PjBL can be investigated in further research. *Third*, students' engagement in their online learning process was not investigated, which could influence their learning outcomes. It is possible that students who engaged more in learning could achieve better performance in the post-study tests. Students' log data such as browsing times and frequency can be collected for further analysis. Previous studies also used questionnaires and reflection journals to analyze student engagement in PjBL (Cudney & Kanigolla, 2014; Fujimura, 2016). *Fourth*, some weaknesses of the proposed learning program mentioned in student comments could affect students' learning experience. Future studies will improve the design and implementation of the learning program via solving technical problems of the learning system, providing more learning resources, and enhancing teacher-student interactions.

5 Conclusions

Project-based learning (PjBL) has been increasingly promoted and extended to online environments to enhance student learning in higher education settings. However, there is concern that PjBL involves complex processes requiring higher-order thinking skills, which may pose challenges to many students especially in online settings without prompt support from teachers. The challenge may compromise the learning of low-achieving students, who often have inadequate higher-order thinking skills.

This study incorporated a visible thinking approach into online learning with a programming project to make the complex process of PjBL accessible to students. The approach features a visual representation of the problem-solving phases that one must go through to complete a project together with the rules-of-thumb or heuristics that help learners to accomplish each phase. We investigated whether and how students of different levels (low, medium, and high) of prior knowledge might benefit differently from the proposed approach. The results showed that low-achieving students made the most progress in their product quality and thinking skills (in particular process design skills). They performed almost as well as medium and high achievers in product quality and process design skills at the end of the study. They also gained more knowledge from the learning program than high achievers did. Compared to medium achievers, low achievers perceived the approach as more valuable, made more effort on the study, and felt more competent in completing the project.

The findings of the study may contribute to the literature on incorporating visible thinking approaches into online learning with real-world projects or problem-solving tasks. While previous studies reported that visible thinking approaches can improve students' performance in problem-solving tasks or projects (Gijlers & de Jong, 2013; Peng et al., 2019; Slof et al., 2012; Wang et al., 2018), the present study reveals that the advantages of such approaches can be more salient to low-achieving students, who often have inadequate higher-order thinking skills to accomplish a real-world project. In particular, visible thinking approaches can more effectively help low achievers to stimulate intrinsic motivation and positive emotions during the task, and significantly improve their higher-order thinking skills and task performance. As a result, such approaches narrow the achievement gap between high and low achievers and offer a variety of students an equal chance to engage in effective learning with real-world projects or problem-solving tasks.

Our findings have several implications for research and practice in online learning with real-world projects. *First*, when extending student-centered PjBL to online settings with little prompt support from teachers, students may face substantial challenges in accomplishing a real-world project. It is thus important to make the complex thinking process visible or accessible to students to foster effective thinking and self-sustained learning in online settings. *Second*, such kind of support is more crucial to low-achieving students, who often have inadequate higher-order thinking skills to accomplish a real-world project. Providing such kind of support has a potential to reduce the disadvantage of low-achieving students and make complex PjBL accomplishable by a wide range of learners. In this way, all students have an equal chance to accomplish complex learning with real-world projects, which is important for the equality of opportunity in education (Noguera et al., 2015). *Third*, although PjBL can promote students' motivation to learn, it is important to sustain students' motivation especially when they face challenges in completing the complex process of PjBL. Effective design of technology-supported learning environments can play a role in externalizing and facilitating higher-order thinking processes to empower students to persist through challenges and feel more competent in completing complex PjBL. *Finally*, while visualization approaches in programming education have focused on helping students to understand abstract concepts and complicated behavior of programs and increase their intrinsic motivation to learn complex programming, it is important to visualize the higher-order thinking process of completing a realistic programming project. The higher-order thinking process involves not only the project process (i.e., problem formulation, solution planning, solution design, and solution implementation), but also the strategies for completing project activities (e.g., how to formulate a problem statement by specifying project requirements and project goals, and how to generate a solution plan by proposing a set of interrelated functional modules). Visualizing the higher-order thinking process is critical to completing a programming project (Gómez-Albarrán, 2005; Peng et al., 2019), especially ill-defined realistic programming projects involving complex problem-solving processes.

Appendix 1.

Table 10 Assessment rubrics for product quality

Aspect	Description*	Score range
Correctness	10 – Correct solution specifications/program code and results consistent with problem requirements. 5 – Partial solution specifications/program code and/or some results. 0 – No solution specifications/program code, or results inconsistent with problem requirements.	0 to 10
Efficiency	10 – Most algorithms, data structures, control structures, and language constructs are appropriate. 5 – Program accomplishes its task, but lacks coherence in choice of either data and/or control structures. 0 – Program solution lacks coherence in choice of both data and control structures.	0 to 10
Reliability	10 – Program functions properly under all test cases. Works for and responds to all valid inputs. 5 – Program functions under limited test cases. Only works for valid inputs, but fails to respond to invalid inputs. 0 – Program fails under most test cases.	0 to 10
Readability	10 – Program code includes clear documentation (comments, meaningful identifiers, indentation to clarify logical structure) and user instructions. 5 – Program code lacks clear documentation and/or user instructions. 0 – Program code is totally incoherent.	0 to 10

*Scores of 1-4 and 6-9 were assigned when the quality of solution was assessed to be between the major units 0, 5, and 10.

Appendix 2.

Table 11 Assessment rubrics for programming thinking skills

Aspect		Description*	Score range
Problem understanding	Project requirements	10 – Project requirements are clearly and correctly stated. All elements are identified. 5 – Project requirements are partially stated. Some elements are not identified. Some statements are incorrect or irrelevant. 0 – No relevant project requirements are identified.	0 to 10
	Project goals	10 – Project goals are clearly and correctly stated. All elements are identified. 5 – Project goals are partially stated. Some elements are not identified. Some statements are incorrect or irrelevant. 0 – No relevant project goals are identified.	0 to 10
Modular design	Functional modules	10 – Detailed and clear planning of the solution, with complete and appropriate functional modules. 5 – Partially correct planning of the solution, with incomplete or inappropriate functional modules. 0 – No appropriate or relevant modules proposed for the solution plan.	0 to 10
	Relationships between functional modules	10 – Detailed and appropriate relationships between functional modules. 5 – Partially correct or incomplete relationships between functional modules. 0 – No or completely inappropriate relationships between functional modules.	0 to 10
Process design	Module decomposition	10 – Complete and appropriate decomposition of functional modules in the process design. 5 – Partially correct or incomplete decomposition of functional modules in the process design. 0 – No or completely inappropriate decomposition of functional modules in the process design.	0 to 10
	Process organization	10 – Complete and appropriate organization of the process. 5 – Partially correct or incomplete organization of the process. 0 – No or completely inappropriate organization of the process.	0 to 10

**Scores of 1-4 and 6-9 were assigned when the quality of solution was assessed to be between the major units 0, 5, and 10.

Appendix 3.

Programming knowledge pre-test

I. Single-choice questions

1. C# is what kind of language? ( )
A) Object-Oriented programming language
B) Machine language
C) Assembly language
D) Natural language

2. Which word below belongs to C# keyword? ( )
A) abstract
B) camel
C) Salary
D) Employ

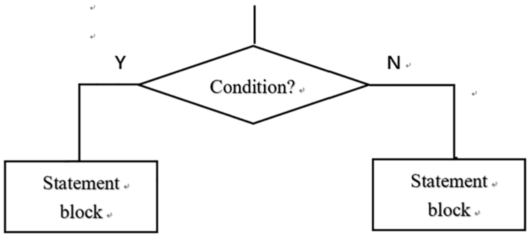
3. If int a=11, then the value of expression (a++*1/3) is ( ).
A) 0
B) 3
C) 4
D) 12

4. What is the value of "a" in following expression? a=3+3>5? 0:1 ( )
A) 6
B) 1
C) 0
D) true

5. Which description below about constructor is correct? ( )
A) The constructor must possess the same name of its containing class.
B) The constructor cannot be private.
C) The constructor cannot receive parameters.
D) The constructor can have return value.

6. Which description below about array is incorrect? ( )
A) The index of array starts from 0.
B) If the index < 0, or index> the length of array, the compiler will throw an IndexOutOfRangeException exception.
C) Array.Reverse() is used to reverse the contents of a one-dimensional array.
D) The index of array starts from 1 and ends at the length of array.

7. Which program structure does the following flowchart correspond? ( )



A) while
B) do … while …
C) if… else…
D) switch… case…

8. What is the output of the following program?

```
class Program
{
    static void Main(string[] args)
    {
        MyStruct s1 = new MyStruct(1, 2);
        s1.x = 2;
        s1.Sum();
        Console.ReadLine();
    }
}
struct MyStruct
{
    public int x;
    public int y;
    public MyStruct(int i, int j)
    {
        x = i;
        y = j;
    }
    public void Sum()
    {
        int sum = x + y;
        Console.WriteLine("the sum is {0}", sum);
    }
}
```

A) the sum is 4
B) the sum is 3
C) the sum is 2
D) the sum is 0

9. Which description below about method overloading is incorrect? ( )
A) Method overloading may extend the functionality of containing class.
B) The constructor cannot be overloaded.

C) ConsoleW(int value)is a method overloading to ConsoleW(string value).
D) Method overloading means "Same method name with different parameter lists".

10. If we want a method of a class to be modifiable in its sub-class, then the method should be defined as ( ).
A) sealed
B) public
C) virtual
D) override

II. Fill-in-blank questions (30 points):

1. ______ statement can stop the current loop cycle and continue with the next loop cycle.

2. Class fields can be accessed by get () and ______ property accessors.

3. Operator___is used to represent Logical NOT, while___is used to represent Logical AND.

4. If a=5, b=4, c=6, then value of a>b? (a>c? a:c):b is ______.

5. _______operator adds the value of an expression to the value of a variable and assigns the result to this variable

III. Short program-writing (30 pints):

1. If a=1, b=2, c=3, x=2, evaluate the value of y= ax2+bx+c and print it. Please write your answer within the Main method.

```
static void Main(string[] args)
{

}
```

2. Ask user to input seconds number then convert it to hours-minutes-seconds format and print it. For example, input 7278 then print 2-1-18. Please write your answer within the Main method.

```
static void Main(string[] args)
{

}
```

Appendix 4.

List of abbreviations:

- 4C/ID: Four-Component Instructional Design
- PBL: Problem-Based Learning
- PjBL: Project-Based Learning

Acknowledgements The authors would thank the students and teacher who participated in this study. The authors would also thank Professor Haijing Jiang for his valuable support for this study.

Authors' contributions JP: Data curation, Formal analysis, Software, and Writing - original draft; MS: Data curation, Formal analysis, and Writing - original draft; BY: Data curation, and Formal analysis; CPL: Validation, and Writing - review & editing; JJGM: Validation, and Writing - review & editing; MW: Supervision, Project administration, Validation, and Writing - review & editing. All authors read and approved the final manuscript.

Funding This research was supported by the Macau Foundation (No. MF2315), the General Research Fund from the Research Grants Council of the Hong Kong SAR Government (No.17201415), and the Eastern Scholar Chair Professorship Fund from Shanghai Municipal Education Commission of China (No. JZ2017005).

Data availability The datasets generated during the current study are not publicly available due to ethical requirements but are available from the corresponding author on reasonable request.

Declarations

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

References

Belland, B. R., Walker, A. E., Kim, N. J., & Lefler, M. (2016). Synthesizing results from empirical research on computer-based scaffolding in STEM education: A Meta-analysis. *Review of Educational Research, 87*(2), 309–344. https://doi.org/10.3102/0034654316670999

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & Palincsar, A. (2011). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist, 26*(3/4), 369–398. https://doi.org/10.1080/00461520.1991.9653139

Cameron, J., & Pierce, W. D. (1994). Reinforcement, reward, and intrinsic motivation: A meta-analysis. *Review of Educational Research, 64*(3), 363–423. https://doi.org/10.3102/00346543064003363

Chen, C. H., & Yang, Y. C. (2019). Revisiting the effects of project-based learning on students' academic achievement: A meta-analysis investigating moderators. *Educational Research Review, 26*, 71–81. https://doi.org/10.1016/j.edurev.2018.11.001

Chen, J., Wang, M., Dede, C., & Grotzer, T. A. (2021). Analyzing student thinking reflected in self-constructed cognitive maps and its influence on inquiry task performance. *Instructional Science, 49*, 287–312.

Chu, S. K. W., Zhang, Y., Chen, K., Chan, C. K., Lee, C. W. Y., Zou, E., & Lau, W. (2017). The effectiveness of wikis for project-based learning in different disciplines in higher education. *Internet and Higher Education, 33*, 49–60. https://doi.org/10.1016/j.iheduc.2017.01.005

Corbalan, G., Kester, L., & Van Merriënboer, J. J. G. (2009). Dynamic task selection: Effects of feedback and learner control on efficiency and motivation. *Learning and Instruction, 19*(6), 455–465. https://doi.org/10.1016/j.learninstruc.2008.07.002

Cudney, E., & Kanigolla, D. (2014). Measuring the impact of project-based learning in six sigma education. *Journal of Enterprise Transformation, 4*(3), 272–288. https://doi.org/10.1080/19488289.2014.930546
De los Ríos, I., Cazorla, A., Díaz-Puente, J., & Yagüe, J. (2010). Project–based learning in engineering higher education: Two decades of teaching competences in real environments. *Procedia Social and Behavioral Sciences, 2*(2), 1368–1378. https://doi.org/10.1016/j.sbspro.2010.03.202
Deek, F. P., Hiltz, S. R., Kimmel, H., & Rotter, N. (1999). Cognitive assessment of students' problem solving and program development skills. *Journal of Engineering Education, 88*(3), 317–326. https://doi.org/10.1002/j.2168-9830.1999.tb00453.x
Eisenberg, M., Basman, A., & Hsi, S. (2014). Math on a sphere: Making use of public displays in mathematics and programming education. *Knowledge Management & E-Learning*, *6*(2), 140–155. https://doi.org/10.34105/j.kmel.2014.06.010
English, M. C., & Kitsantas, A. (2013). Supporting student self-regulated learning in problem-and project-based learning. *Interdisciplinary journal of problem-based learning, 7*(2), 6. https://doi.org/10.7771/1541-5015.1339
Fanchamps, N. L. J. A., Slangen, L., Specht, M., & Hennissen, P. (2021). The impact of SRA programming on computational thinking in a visual oriented programming environment. *Education and Information Technologies, 26*(5), 6479–6498. https://doi.org/10.1007/s10639-021-10578-0
Felder, R. M., & Brent, R. (2010). The national effective teaching institute: Assessment of impact and implications for faculty development. *Journal of Engineer Education, 99*(2), 121–134. https://doi.org/10.1002/j.2168-9830.2010.tb01049.x
Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004). School engagement: Potential of the concept, state of the evidence. *Review of Educational Research, 74*(1), 59–109. https://doi.org/10.3102/00346543074001059
Fujimura, T. (2016). EFL students' learning through project work in a content-based course. *The Journal of Kanda University of International Studies, 28*, 105–124.
George, D., & Mallery, M. (2010). *SPSS for Windows Step by Step: A Simple Guide and Reference (10a Ed.)*. Boston: Pearson Education.
Gijbels, D., Dochy, F., Van den Bossche, P., & Segers, M. (2005). Effects of problem-based learning: A meta-analysis from the angle of assessment. *Review of Educational Research, 75*(1), 27–61. https://doi.org/10.3102/00346543075001027
Gijlers, H., & de Jong, T. (2013). Using concept maps to facilitate collaborative simulation-based inquiry learning. *Journal of the Learning Sciences, 22*(3), 340–374. https://doi.org/10.1080/10508406.2012.748664
Gómez-Albarrán, M. (2005). The teaching and learning of programming: A survey of supporting software tools. *The Computer Journal, 48*(2), 130–144. https://doi.org/10.1093/comjnl/bxh080
Gul, S., Asif, M., Nawaz, Z., Aziz, M. H., Khurram, S., Saleem, M. Q., Habib, E. O. A., Shafiq, M., Sheta, O. E. (2023). Sustainable Learning of Computer Programming Languages Using Mind Mapping. *Intelligent Automation & Soft Computing*, *36*(2), 1687-1697. https://doi.org/10.32604/iasc.2023.032494
Guo, P., Saab, N., Post, L. S., & Admiraal, W. (2020). A review of project-based learning in higher education: Student outcomes and measures. *International Journal of Educational Research, 102*, 101586. https://doi.org/10.1016/j.ijer.2020.101586
Hall, R. H., & O'Donnell, A. (1996). Cognitive and affective outcomes of learning from knowledge maps. *Contemporary educational psychology, 21*(1), 94–101. https://doi.org/10.1006/ceps.1996.0008
Harvey, L. (2000). New realities: The relationship between higher education and employment. *Tertiary Education and Management, 6*, 3–17. https://doi.org/10.1080/13583883.2000.9967007
Helle, L., Tynjälä, P., & Olkinuora, E. (2006). Project-based learning in post-secondary education–theory, practice and rubber sling shots. *Higher Education, 51*(2), 287–314. https://doi.org/10.1007/s10734-004-6386-5
Hmelo-Silver, C. E., Duncan, R. G., & Chinn, C. A. (2007). Scaffolding and achievement in problem-based and inquiry learning: A response to Kirschner, Sweller, and Clark (2006). *Educational Psychologist, 42*(2), 99–107. https://doi.org/10.1080/00461520701263368
Hsu, P. S., Van Dyke, M., Chen, Y., & Smith, T. J. (2015). The effect of a graph-oriented computer-assisted project-based learning environment on argumentation skills. *Journal of Computer Assisted Learning, 31*(1), 32–58. https://doi.org/10.1111/jcal.12080
Hundhausen, C. D., & Brown, J. L. (2007). What you see is what you code: A "live" algorithm development and visualization environment for novice learners. *Journal of Visual Languages & Computing, 18*(1), 22–47. https://doi.org/10.1016/j.jvlc.2006.03.002
Jamil, M. G., & Isiaq, S. O. (2019). Teaching technology with technology: Approaches to bridging learning and teaching gaps in simulation-based programming education. *International Journal of Educational Technology in Higher Education, 16*, 25. https://doi.org/10.1186/s41239-019-0159-9

Jazayeri, M. (2015). Combining mastery learning with project-based learning in a first programming course: An experience report. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (Vol. 2, pp. 315-318).

Jerez, J. M., Bueno, D., Molina, I., Urda, D., & Franco, L. (2012). Improving motivation in learning programming skills for engineering students. *International Journal of Engineering Education, 28*(1), 202–208.

Jollands, M., Jolly, L., & Molyneaux, T. (2012). Project-based learning as a contributing factor to graduates' work readiness. *European Journal of Engineering Education, 37*(2), 143–154. https://doi.org/10.1080/03043797.2012.665848

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist, 41*(2), 75–86. https://doi.org/10.1207/s15326985ep4102_1

Koh, J. H. L., Herring, S. C., & Hew, K. F. (2010). Project-based learning and student knowledge construction during asynchronous online discussion. *Internet and Higher Education, 13*(4), 284–291. https://doi.org/10.1016/j.iheduc.2010.09.003

Laight, D. W. (2004). Attitudes to concept maps as a teaching/learning activity in undergraduate health professional education: Influence of preferred learning style. *Medical Teacher, 26*(3), 229–233. https://doi.org/10.1080/0142159042000192064

Lazonder, A. W., & Harmsen, R. (2016). Meta-analysis of inquiry-based learning: Effects of guidance. *Review of Educational Research, 86*(3), 681–718. https://doi.org/10.3102/0034654315627366

Leyer, M., Yuan, B., Wang, M., & Moormann, J. (2023). Classroom or online learning? Impact of experiential learning in business process management education. *Knowledge Management & E-Learning*, *15*(2), 214–234. https://doi.org/10.34105/j.kmel.2023.15.012

López-Pimentel, J. C., Medina-Santiago, A., Alcaraz-Rivera, M., & Del-Valle-Soto, C. (2021). Sustainable Project-Based Learning Methodology Adaptable to Technological Advances for Web Programming. *Sustainability, 13*(15). https://doi.org/10.3390/su13158482

Loyens, S. M. M., & Rikers, R. M. J. P. (2016). Instruction based on inquiry. In R. E. Mayer & P. A. Alexander (Eds.), *Handbook of research on learning and instruction* (pp. 518–551). Taylor & Francis.

McAuley, E., Duncan, T., & Tammen, V. V. (1989). Psychometric properties of the Intrinsic Motivation Inventory in a competitive sport setting: A confirmatory factor analysis. *Research quarterly for exercise and sport, 60*(1), 48–58. https://doi.org/10.1080/02701367.1989.10607413

Mitani, H. (2021). Test Score Gaps in Higher Order Thinking Skills: Exploring Instructional Practices to Improve the Skills and Narrow the Gaps. *AERA Open, 7*(1), 1–23. https://doi.org/10.1177/23328584211016470

Naps, T. L., Rößling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., Rodger, S., & Velázquez-Iturbide, J. A. (2003). Exploring the role of visualization and engagement in computer science education. *ACM SIGCSE Bulletin, 35*(2), 131–152. https://doi.org/10.1145/960568.782998

Noguera, P., Darling-Hammond, L., & Friedlaender, D. (2015). *Equal Opportunity for Deeper Learning*. Students at the Center: Deeper Learning Research Series. Boston, MA: Jobs for the Future.

Norton, S. J., McRobbie, C. J., & Ginns, I. S. (2007). Problem solving in a middle school robotics design classroom. *Research in Science Education, 37*(3), 261–277. https://doi.org/10.1007/s11165-006-9025-6

Pekrun, R., Goetz, T., Frenzel, A. C., Barchfeld, P., & Perry, R. P. (2011). Measuring emotions in students' learning and performance: The Achievement Emotions Questionnaire (AEQ). *Contemporary Educational Psychology, 36*(1), 36–48. https://doi.org/10.1016/j.cedpsych.2010.10.002

Peng, J., Wang, M., & Sampson, D. (2017). Visualizing the complex process for deep learning with an authentic programming project. *Journal of Educational Technology & Society*, *20*(4), 275–287. Retrieved from https://www.jstor.org/stable/26229223

Peng, J., Wang, M., Sampson, D., & van Merrienboer, J. (2019). Using a visualization-based and progressive learning environment as a cognitive tool for learning computer programming. *Australasian Journal of Educational Technology, 35*(2), 52–68. https://doi.org/10.14742/ajet.4676

Phelps, E. A. (2006). Emotion and cognition: Insights from studies of the human amygdala. *Annual Review of Psychology, 57*, 27–53. https://doi.org/10.1146/annurev.psych.56.091103.070234

Preacher, K. J. (2015). Advances in mediation analysis: A survey and synthesis of new developments. *Annual Review of Psychology, 66*, 825–852. https://doi.org/10.1146/annurev-psych-010814-015258

Preacher, K. J., Rucker, D. D., MacCallum, R. C., & Nicewander, W. A. (2005). Use of the extreme groups approach: A critical reexamination and new recommendations. *Psychological Methods, 10*, 178–192. https://doi.org/10.1037/1082-989X.10.2.178

Pucher, R., & Lehner, M. (2011). Project based learning in computer science–A review of more than 500 projects. *Procedia Social and Behavioral Sciences, 29*, 1561–1566. https://doi.org/10.1016/j.sbspro.2011.11.398

Rajala, T., Laakso, M. J., Kaila, E., & Salakoski, T. (2008). Effectiveness of program visualization: A Case study with the ViLLE tool. *Journal of Information Technology Education, 7*, 15–32.

Ralph, R. A. (2015). Post secondary project-based learning in science, technology, engineering and mathematics. *Journal of Technology and Science Education, 6*(1), 26–35. https://doi.org/10.3926/jotse.155

Reis, A. C. B., Barbalho, S. C. M., & Zanette, A. C. D. (2017). A bibliometric and classification study of Project-based Learning in Engineering Education. *Production, 27*, e20162258. https://doi.org/10.1590/0103-6513.225816

Reiser, B. J. (2004). Scaffolding complex learning: The Mechanisms of structuring and problematizing student work. *Journal of the Learning Sciences, 13*(3), 273–304. https://doi.org/10.1207/s15327809jls1303_2

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A Review and discussion. *Computer Science Education, 13*(2), 137–172. https://doi.org/10.1076/csed.13.2.137.14200

Sánchez-García, R., & Pavón-Vázquez, V. (2021). Students' perceptions on the use of project-based learning in CLIL: Learning outputs and psycho-affective considerations. *Latin American Journal of Content & Language Integrated Learning, 14*(1), 69–98. https://doi.org/10.5294/laclil.2021.14.1.3

Sasson, I., Yehuda, I., & Malkinson, N. (2018). Fostering the skills of critical thinking and question-posing in a project-based learning environment. *Thinking Skills and Creativity, 29*, 203–212. https://doi.org/10.1016/j.tsc.2018.08.001

Schutz, P. A., & DeCuir, J. T. (2002). Inquiry on emotions in education. *Educational Psychologist, 37*(2), 125–134. https://doi.org/10.1207/S15326985EP3702_7

Slof, B., Erkens, G., Kirschner, P. A., Janssen, J., & Jaspers, J. G. M. (2012). Successfully carrying out complex learning-tasks through guiding teams' qualitative and quantitative reasoning. *Instructional Science, 40*(3), 623–643. https://doi.org/10.1007/s11251-011-9185-2

Stewart, R. A. (2007). Investigating the link between self directed learning readiness and project-based learning outcomes: The case of international Masters students in an engineering management course. *European Journal of Engineering Education, 32*(4), 453–465. https://doi.org/10.1080/03043790701337197

Soloway, E. (1986). Learning to program = learning to construct mechanisms and explanations. *Communications of the ACM, 29*(9), 850–858. https://doi.org/10.1145/6592.6594

Sorva, J., Karavirta, V., & Malmi, L. (2013). A review of generic program visualization systems for introductory programming education. *ACM Transactions on Computing Education, 13*(4), 15. https://doi.org/10.1145/2490822

Splichal, J. M., Oshima, J., & Oshima, R. (2018). Regulation of collaboration in project-based learning mediated by CSCL scripting reflection. *Computers & Education, 125*, 132–145. https://doi.org/10.1016/j.compedu.2018.06.003

Sun, L., Guo, Z., & Zhou, D. (2022). Developing K-12 students' programming ability: a systematic literature review. *Education and Information Technologies, 27*(5), 7059–7097. https://doi.org/10.1007/s10639-022-10891-2

Sung, H. Y., & Hwang, G. J. (2013). A Collaborative game-based learning approach to improving students' learning performance in science courses. *Computers & Education, 63*, 43–51. https://doi.org/10.1016/j.compedu.2012.11.019

Thomas, J. W. (2000). *A review of research on project-based learning*. Autodesk Foundation.

Van Merriënboer, J. J. G., & Kirschner, P. A. (2017). *Ten steps to complex learning: A systematic approach to four-component instructional design (3rd Rev. Ed.)*. https://doi.org/10.4324/9781315113210

Wang, M., Wu, B., Kirschner, P. A., & Spector, J. M. (2018). Using cognitive mapping to foster deeper learning with complex problems in a computer-based environment. *Computers in Human Behavior, 87*, 450–458. https://doi.org/10.1016/j.chb.2018.01.024

White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling and metacognition: Making science accessible to all students. *Cognition and Instruction, 16*(1), 3–118. https://doi.org/10.1207/s1532690xci1601_2

Wu, B., Wang, M., Grotzer, T. A., Liu, J., & Johnson, J. M. (2016). Visualizing Complex Processes Using a Cognitive-Mapping Tool to Support the Learning of Clinical Reasoning. *BMC Medical Education, 16*, 216. https://doi.org/10.1186/s12909-016-0734-x

Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies, 21*(3), 559–588. https://doi.org/10.1007/s10639-014-9341-9

Yuan, B., Wang, M., van Merrienboer, J., Tao, X., Kushniruk, A., & Peng, J. (2020). Investigating the Role of Cognitive Feedback in Practice-Oriented Learning for Clinical Diagnostics. *Vocations and Learning, 13*, 159–177. https://doi.org/10.1007/s12186-019-09234-z

Zohar, A., & Dori, Y. J. (2003). Higher order thinking skills and low-achieving students: Are they mutually exclusive? *The journal of the learning sciences, 12*(2), 145–181. https://doi.org/10.1207/S15327809JLS1202_1

Zohar, A., Degani, A., & Vaaknin, E. (2001). Teachers' beliefs about low achieving students and higher order thinking. *Teaching and Teachers' Education, 17*, 469–485. https://doi.org/10.1016/S0742-051X(01)00007-5

Authors and Affiliations

Jun Peng[1] · Meng Sun[2] · Bei Yuan[3] · Cher Ping Lim[4] · Jeroen J. G. van Merriënboer[5] · Minhong Wang[6,7]

✉ Minhong Wang
magwang@hku.hk

Jun Peng
4588775@163.com

Meng Sun
msun@bnu.edu.cn

Bei Yuan
yuanbeivicki@163.com

Cher Ping Lim
clim@eduhk.hk

Jeroen J. G. van Merriënboer
j.vanmerrienboer@maastrichtuniversity.nl

1 School of Education, City University of Macau, Macao, China

2 College of Education for the Future, Beijing Normal University, Beijing, China

3 Department of Educational Monitoring and Quality Assessment, Zhongshan Teacher Development Center, Zhongshan, China

4 Faculty of Education and Human Development, The Education University of Hong Kong, Hong Kong, China

5 School of Health Professions Education, Maastricht University, Maastricht, The Netherlands

6 Faculty of Education, The University of Hong Kong, Pokfulam Road, Hong Kong, China

7 Department of Educational Information Technology, East China Normal University, Shanghai, China