The Impact of Peer Assessment and Feedback Strategy in Learning Computer Programming in Higher Education

Wing-Shui NG
The Hong Kong Institute of Education, Hong Kong

ngws@ied.edu.hk

Abstract

This paper describes the process of using peer assessment and feedback strategy to enhance the effectiveness on learning computer programming. Students worked in pairs to complete learning tasks collaboratively on computer programming. Data were collected using questionnaires and quizzes for further analysis. The findings show that the students were satisfied with the peer assessment and feedback strategy in learning computer programming. Moreover, their actual performance was also better when compared with that achieved using traditional teaching method.

Keywords: peer assessment, assessment for learning, computer programming, common programming error, higher education

Introduction

Increasing emphasis has been placed in recent years on the significance of assessment for learning. One of the significant contributions in this area is that of Black and Wiliam (1998) who conducted an extensive review of related research and confirmed broad evidence for the value of using assessment for learning to raise standards. Their literature review includes studies reporting learning gains related to the use of assessment for learning strategies, applicable to a diverse range of targets and in various disciplines. Numerous studies (Brown et al., 2009; Black & Wiliam, 2009; Elwood, 2006; Carless, 2005) of effective strategies for assessment for learning have been reported in academic journals in recent years. To advance our knowledge in this area further, this study explores the effectiveness of strategies related to assessment for learning in teaching an undergraduate computer programming course. The following sections introduce the theoretical framework for this research and then describe the research setting, design, and rationale, as well as ethical concerns. In the final section, the data collection methods are set out, and the results analyzed and discussed.

Theoretical Framework

Difficulties with Computer Programming

Programming can be regarded as a very useful skill. Introductory programming courses are popular in the higher education sector as part of the foundations of an information technology-related curriculum (Robins et al., 2003). However, programming is a complex intellectual activity and few students find it easy to learn. Programming courses are generally regarded as difficult and often have high dropout rates (Ahoniemi et al., 2007). It is an eminent problem that has motivated many researchers to propose methodologies and tools to help students learn computer programming (Robins et al., 2003; Gomes & Mendes, 2007; Jenkins, 2002).

The difficulties involved in learning how to program have various aspects, among which the linguistic intricacies of computer programming languages have been addressed by many researchers (Hristova et al., 2003; Jenkins, 2002; Gomes & Mendes, 2007, Truong et al., 2004). As mentioned by Gomes and Mendes (2007), the syntax of programming languages is very complex. Computer programming languages were developed for professional use with many complex syntactic details to be memorized, and are not suitable for novices. It is common for students to find it difficult to detect simple syntactical and logical programming errors. In this connection, efforts have been made by academics to address common programming errors made by students (Truong et al., 2004). However, despite extensive coverage of such mistakes in textbooks and lectures, they tend to persist when students actually write programs (Hristova et al., 2003). To enhance the accuracy on writing computer programs so as to further improve students' overall programming abilities, it is still worth making an effort to explore effective strategies for tackling students' common programming errors.

Assessment for Learning

Black and Wiliam (1998b) defined assessment broadly to include all activities undertaken by teachers and students to obtain information that can be used diagnostically to alter teaching and learning. In the literature, few studies investigate the use of such assessment for learning strategies in the field of computer programming, particularly in identifying common programming errors. The purpose of assessment for learning, also known as formative assessment, includes identifying students' strengths and weaknesses, assisting educators in the planning of subsequent instruction, helping students to guide their own learning, and fostering increased autonomy and responsibility for one's own learning (Cizek, 2010).

According to Cizek (2010), teachers' work in assessing students is intended to collect evidence on their learning so that both parties can further develop specific strategies to improve its learning effectiveness. In this connection, Black and Wiliam (2009) conceptualized the following five key strategies to implement assessment for learning:

1. *Clarify and share learning intentions and criteria for success;*
2. *Engineer effective classroom discussions and other learning tasks that elicit evidence of student understanding;*
3. *Provide feedback that moves learners forward;*
4. *Activate students as instructional resources for one another; and*
5. *Activate students as the owners of their own learning.*

(Black & Wiliam, 2009, p. 8)

Peer Assessment and Feedback

Within the concept of assessment *for* learning, enhancing the effectiveness of learning is the main purpose of peer assessment. Peer assessment using constructive feedback can serve the purpose of activating students as instructional resources for one another and as the owners of their own learning (Leahy et al., 2005).

A number of studies indicate that peer assessment and feedback strategies can improve students' learning. Students can learn from one another's performance, both excellent and poor, if they are engaged to critically analyze others' writing styles, techniques, ideas, and abilities (Race, 1998). Topping and colleagues (2000) point out that students can develop a better understanding of the objectives and purposes of the assessment task and the course as a result. Moreover, by being assigned the role of assessor, students will learn more about why teachers find it difficult to assign marks (Billington, 1997; Hanrahan & Isaacs, 2001). Students can thereby develop awareness of the importance of presenting assignments in a clear and logical format as well as appreciating why and how marks are awarded (Brindley & Scoffield, 1998; Race, 1998). They can also come to understand the standards required and teachers' expectations of them when performing assessment tasks (Falchikov, 1995; Hanrahan & Isaacs, 2001; Race, 1998). Furthermore, when they are forced to confront and reflect on their performance in assessment tasks, students must pay closer attention to what factors lead to a good or poor assignment. As a consequence, they will develop more understanding, improve their confidence, and achieve better performance in subsequent tasks (Dochy et al., 1999; Mowl & Pain, 1995; Searby & Ewers, 1997; Topping et al., 2000).

In view of the positive impact of an instructional strategy incorporating peer assessment and feedback, the present study attempted to explore the use of this approach in enhancing the learning effectiveness of an undergraduate computer programming course.

Research Setting

The author is a teacher trainer in The Hong Kong Institute of Education and is involved in training and assessing pre- and in-service teachers. A course entitled *Introduction to Programming and Problem Solving*, taught by the researcher, was offered to a combined class of first-year undergraduates. The course aimed to provide participants with the concepts, knowledge, skills, and techniques of programming together with the opportunity to gain first-hand experience by developing and debugging their own programs. The aim was to produce teachers who would be able to help their students develop problem-solving abilities. All the students were in-service technical staff in primary and secondary schools; 23 were male and 1 female. The course was offered from June to August 2011. Upon graduation, students were expected to work as primary or secondary school teachers in Hong Kong.

Research Design

A computer program is a set of instructions for completing specific tasks. The simplest program structure involves all instructions being listed sequentially and the instructions executed line by line. This kind of simple program requires relatively little cognitive effort to write. However, most programs have a more complex structure involving the construction of control flow statements, in which instructions are no longer executed sequentially. Control statements control which parts of instructions will be executed at specific times according to different situations arising during the execution of the program. Due to the leaping nature of control flow, students normally start to encounter difficulties at this point, and will make a number of common programming errors.

With a view to tackling the challenge of learning about control flow structure, the researcher specially designed a series of bespoke activities for students after related topics had been taught. Drawing on the work of Deitel and Deitel (2010, pp. 75-166), 18 common programming errors concerning control flow statements were identified. Activities were designed around these common programming errors which aimed at helping students to avoid committing them. These activities lasted for two weeks as described below.

Week 1 Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy

The researcher specially designed an assignment consisting of a series of short computer programs into which 10 common programming errors had intentionally been inserted. The objective was to learn about common programming errors by identifying these mistakes. At the beginning of week 1, students were required to form pairs. Since they were expected to support each other during the learning activities, they were free to choose whoever they wanted to work with. The assignments were then given to each pair. In the first phase, each student was responsible for studying half the programs and then identifying the error in each, suggesting how it could be corrected, and then carrying out the remedial work and submitting corrected programs. The first phase lasted for two days. After this, students proceeded to the second phase, in which they had to exchange their work to obtain peer assessment and feedback. Each student was required to review the classmate's work and provide constructive feedback. Students were also expected to revise the computer programs further to improve their overall quality. At the end of the week, the students had to submit all their work, including the corrected programs, to the researcher.

Immediately after the students submitted their assignments, the researcher explained the common programming errors which had been covered. Afterwards, students were given a short quiz to test their understanding of related errors. The quiz comprised four short programs into which five common programming errors had been intentionally inserted. Programs may be completely correct (that is, error free) or may include more than one error. They were required to locate the errors, identify which category they fell into, and provide suggested corrections.

Week 2 Direct Learning Common Programming Errors by Examples Strategy

The researcher prepared teaching notes which clearly listed eight common programming errors with illustrated examples. These were given to students at the beginning of Week 2 for self-study purposes.

At the end of the week, the researcher explained related common programming errors to students in more detail and a short quiz was given to test their understanding. The format of the quiz was the same as the quiz in Week 1 and the level of difficulty was also similar.

Rationale of the Research Design

The main purpose of this research was to explore the effectiveness of using peer assessment and feedback strategy to learn computer programming by means of identifying common programming errors. The learning activities in Week 1 were designed for this purpose. On the other hand, the activities in Week 2, which required students to learn common programming errors by self-study of suitable examples, epitomized traditional instructional methods used by teachers. Students' performance and attitudes were compared across Weeks 1 and 2. In addition, the quizzes at the end of each week aimed at analyzing students' actual understanding of related knowledge. This also served as an indicator of the effectiveness of using the peer assessment and feedback strategy on learning computer programming.

Ethical Concerns

In order to obtain their consent to the collection of data related to their learning performance, a consent form was given to students in the early stages of the course. This form clearly stated that the data were being collected solely for academic use and to improve the effectiveness of teaching. Students were asked to participate in lesson activities as normal, and told that information about individuals would never be disclosed. They were also advised that their assessment results would not be affected by any data analysis procedure, and that they could withdraw without prejudice at any stage of the research. The students were invited to sign the consent form to indicate their agreement and all eventually did so.

Data Collection and Analysis

Questionnaires

According to one study (Lin et al., 2001) of attitudes and achievements in web-based learning, students' attitudes toward peer assessment are significantly related to overall performance. Students with positive attitudes outperform those who are negative. In line with this finding, a questionnaire adapted from Xiao & Lucking (2008) was used to analyze students' attitude toward the peer assessment and feedback strategy used in this research. If students were satisfied with the approach, it is reasonable to anticipate that their learning performance would also be better. The questionnaires were given to students at the end of Week 2 in the lesson. The items included in the questionnaire are listed in the Appendix.

Descriptive statistical analysis was used to analyze students' overall attitude towards the strategy. In addition, since the same group of participants had experienced two different instructional strategies, a t-Test of correlated means was carried out for three pairs of questions (1 and 2, 3 and 4, and 7 and 8) to compare their opinions on the two strategies.

Quiz

Data collected in the quizzes administered in Weeks 1 and 2 were used to check the students' actual comprehension of common programming errors. Since the same group of participants had been involved, a t-Test of correlated means was used to analyze their performance under the different instructional strategies. In addition, descriptive statistics were also used for analysis.

Results

Descriptive Statistics of Questionnaire Items

The descriptive statistics are shown in Table 1. Except for questions 2, 4, and 8, which covered students' opinions on the learning by examples strategy, and question 5, which covered whether or not the peer assessment and feedback strategy had been too demanding, all the mean scores ranged from 5.88-6.47 which is above the mid-point of the range (5.5). This indicates that the students generally agreed on the different aspects of the peer assessment and feedback strategy, and further that they were satisfied with its application to their learning about computer programming.

However, it is interesting to note that the mean score for question 5 is 5.71, which is only a little higher than the middle value. This means that although students generally agreed that the peer assessment and feedback strategy had been good for them, they felt that completing the learning activities had been somewhat demanding.

Table 1: Descriptive statistics for each questionnaire item

	Mean	Median	Mode	Std. Deviation	Range	Minimum	Maximum
Q1	6.18	6	6	1.944	7	3	10
Q2	5.88	6	4[a]	1.933	7	3	10
Q3	6.24	7	7	1.985	7	2	9
Q4	6.41	7	7	1.661	6	3	9
Q5	5.71	6	5[a]	1.49	5	3	8
Q6	6.24	6	5[a]	1.64	6	3	9
Q7	6.35	6	6	1.539	6	3	9
Q8	6.29	6	5[a]	1.759	6	3	9
Q9	5.88	6	6	2.058	6	3	9
Q10	6.35	6	5	2.06	7	3	10
Q11	6.41	6	4	2.033	6	4	10
Q12	6.47	6	5	2.004	6	4	10
Q13	6.47	7	7	1.972	7	3	10
Q14	6.41	6	5[a]	2.063	7	3	10
Q15	6.29	6	8	2.114	8	2	10
Q16	6.47	7	5[a]	1.625	5	4	9
Q17	6.29	6	6	1.759	6	4	10
Q18	6.29	6	6	1.611	5	4	9

a. Multiple modes exist. The smallest value is shown

t-Test of Correlated Means of Questionnaire Items

Since the same group of students had experienced two different instructional strategies, a t-Test of correlated means was applied to Questions 1 and 2, 3 and 4, and 7 and 8 to compare students' attitudes towards them. The results are shown in Table 2. It can be seen that there is no significant difference in attitudes toward the two instructional strategies. If we look at the mean scores of the related questions in Table 1, all range from 5.88-6.41. This indicates that students expressed satisfaction with both learning strategies. It can also be interpreted as suggesting that the students generally appreciated their teachers' efforts to assist with their learning using both instructional strategies.

Table 2: Results of the t-Test of correlated means comparing students' attitudes towards the two instructional strategies

		Paired Differences					t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	**Q1 - Q2**	.294	2.201	.534	-.838	1.426	.551	16	.589
Pair 2	**Q3 - Q4**	-.176	2.325	.564	-1.372	1.019	-.313	16	.758
Pair 3	**Q7 - Q8**	.059	1.478	.358	-.701	.819	.164	16	.872

Descriptive Statistics of Quizzes

The descriptive statistics for the two quizzes are shown in Table 3. Items E1 to E5 refer to the five common programming errors which had been deliberately inserted. The maximum mark for each quiz was 10. The score for each error identified was 2, with 0.5 marks awarded for identifying the error, 0.5 marks for categorizing it, and 1 mark for correcting it.

As shown in Table 3, students' mean total marks were 5.053 and 4.441 for Quizzes 1 and 2 respectively. Their performance in both can therefore be regarded as average. In addition, it can be seen that students' performance in Quiz 1, administered after experiencing the peer assessment and feedback strategy, was better than in Quiz 2 in which the direct learning by examples strategy had been adopted.

Table 3: Descriptive statistics for the quiz scores

	Quiz 1						**Quiz 2**					
	E1	*E2*	*E3*	*E4*	*E5*	*Total Mark*	*E1*	*E2*	*E3*	*E4*	*E5*	*Total Mark*
Mean	1.21	1.289	0.684	0.789	1.07	5.053	1.794	0.971	0.206	0.676	0.794	4.441
Mode	2	2	0	0	2	0	2	0	0	0	0	5.5
Median	2	1.5	0	0	1.5	4.5	2	1	0	0	0.5	5
S.D.	0.976	0.8711	0.8694	0.9327	0.8861	3.2528	0.5321	0.9758	0.5879	0.8467	0.8303	1.9596

t-Test of Correlated Mean of Quizzes

Since the same group of students had taken both quizzes, a t-Test of correlated means was carried out to identify any difference in performance. The results shown in Table 4 indicate that no significant difference was found.

Table 4: Results of the t-Test of correlated means comparing students' scores in Quizzes 1 and 2

		Paired Differences						
				95% Confidence Interval of the Difference				
	Mean	Std. Deviation	Std. Error Mean	Lower	Upper	t	df	Sig. (2-tailed)
Pair 1 Quiz1 - Quiz2	.5333	2.1586	.5573	-.6621	1.7287	.957	14	.355

Discussion

The results of this study suggest that students were generally satisfied with the peer assessment and feedback strategy for learning computer programming. The effectiveness of their learning was also higher when compared with that achieved using a traditional teaching strategy employing self-study of examples. However, this evidence is not enough to show that the peer assessment and feedback strategy had a significant effect on learning computer programming. The study design has certain limitations which may have led to this unsatisfactory result. One major difficulty is the relatively small sample size. The valid sample size is only 16. One of the assumptions for the t-Test is that the data are normally distributed, which is inevitably weakened with such a small sample. Moreover, since all the students in the class were part-time students, they all had day jobs and hence limited time could be spent on study. As a consequence, it is reasonable to suggest that their learning curves would be relatively gentle, with more time being required to master particular areas of knowledge and skills. As this study lasted for only two weeks, it is perhaps unrealistic to expect that the participants would demonstrate significant improvement. Based on these analyses, future research utilizing larger class sizes and of longer duration would be appropriate in order to obtain sufficient evidence for the impact of using peer assessment and feedback strategy on learning computer programming. In addition, future studies could also look into the impact of particular factors within peer assessment and feedback strategy on learning computer programming.

Conclusion

This study addressed the growing current concern about the effectiveness of the assessment for learning strategy. The findings showed that students were satisfied with the use of peer assessment and feedback strategy in learning computer programming. Students' actual performance was also better when compared with that obtained after using a traditional teaching method. By sharing the results of this study, it is hoped that effective strategies for implementing assessment for learning can be progressively unveiled in future.

References

Ahoniemi, T., Lahtinen, E., & Erkkola, T. (2007). Fighting the student dropout rate with an incremental programming assignment. In Lister et al. (Eds.), *Seventh Baltic Sea Conference on Computing Education Research (Koli Calling 2007)* (pp. 163-166). Koli National Park, Finland.

Billington, H. L. (1997). Poster presentations and peer assessment: Novel forms of evaluation and assessment. *Journal of Biological Education, 31*(3), 218-220.

Black, P., & Wiliam, D. (1998a). Assessment and classroom learning. *Assessment in Education, 5*(1), 7-74.

Black, P., & Wiliam, D. (1998b). *Inside the black box: Raising standards through classroom assessment.* London, UK: GL Assessment.

Black, P., & Wiliam, D. (2009). Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability, 21*, 5-31.

Brindley, C., & Scoffield, S. (1998). Peer assessment in undergraduate programs. *Teaching in Higher Education, 3*(1), 79-89.

Brown, G. T., Kennedy, K. J., Fok, P. K., Chan, J. K., & Yu, W. M. (2009). Assessment for student improvement: Understanding Hong Kong teachers' conceptions and practices of assessment. *Assessment in Education: Principles, Policy & Practice, 16*(3), 347-363.

Carless, D. (2005). Prospects for the implementation of assessment for learning. *Assessment in Education, 12*(1), 39-54.

Cizek, G. J. (2010). An introduction to formative assessment: History, characteristics, and challenges. In H. L. Andrade & G. J. Cizek (Eds.), *Handbook of formative assessment* (pp. 3-17). London, UK: Routledge.

Deitel, P., & Deitel, H. (2010). *Java: How to program: Late objects version.* Hong Kong: Pearson Education.

Dochy, F., Segers, M., & Sluijsmans, D. (1999). The use of self-, peer and co-assessment in higher education: A review. *Studies in Higher Education, 24*(3), 331-350.

Elwood, J. (2006). Formative assessment: Possibilities, boundaries and limitations. *Assessment in Education, 13*(2), 215-232.

Falchikov, N. (1995). Peer feedback marking: Developing peer assessment. *Innovations in Education and Training International, 32*, 175-187.

Gomes, A., & Mendes, A. J. (2007). Learning to program - difficulties and solutions. *International Conference on Engineering Education - ICEE 2007.* Coimbra, Portugal.

Hanrahan, S. J., & Isaacs, G. (2001). Assessing self- and peer-assessment: The students' views. *Higher Education Research & Development, 20*(1), 53-70.

Hristova, M., Misra, A., Rutter, M., & Mercuri, R. (2003). Identifying and correcting Java programming errors for introductory computer science students. *SIGCSE '03.* Nevada, USA.

Jenkins, T. (2002). On the difficulty of learning to program. *3rd Annual LTSN-ICS Conference* (pp. 53-58). Loughborough, UK: LTSN Centre for Information and Computer Sciences.

Leahy, S., Lyon, C., Thompson, M., & Wiliam, D. (2005). Classroom assessment minute by minute, day by day. *Educational Leadership, 63*(3), 18-24.

Lin, S. S., Liu, E. Z., & Yuan, S. (2001). Web based peer assessment: Attitude and achievement. *IEEE Transactions on Education, 44*(2).

Mowl, G., & Pain, R. (1995). Using self and peer assessment to improve students' essay writing: A case study from geography. *Innovation in Education and Training International, 32*(4), 324-335.

Race, P. (1998). Practical pointers on peer-assessment. In S. Brown (Ed.), *Peer assessment in practice* (p. 102). Birmingham, UK: SEDA.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer Science Education, 13*(2), 137-172.

Searby, M., & Ewers, T. (1997). An evaluation of the use of peer assessment in higher education: A case study in the School of Music, Kingston University. *Assessment and Evaluation in Higher Education, 22*(4), 371-383.

Topping, K. J., Smith, E. F., Swanson, I., & Elliot, A. (2000). Formative peer assessment of academic writing between postgraduate students. *Assessment and Evaluation in Higher Education, 25*(2), 146-169.

Truong, N., Roe, P., & Bancroft, P. (2004). Static analysis of students' Java programs. *The Sixth Australian Computing Education Conference (ACE2004).* Dunedin, New Zealand.

Appendix

Questionnaire

The questionnaire adapted from Xiao and Lucking (2008) consisted of the following questions.

1. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” enhanced my motivation to learn about common programming errors.
2. The “Direct Learning Common Programming Errors by Examples Strategy” enhanced my motivation to learn about common programming errors.
3. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” is an effective strategy for learning about common programming errors.
4. The “Direct Learning Common Programming Errors by Examples Strategy” is an effective strategy for learning about common programming errors.
5. The requirements of the activities in the “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” are too demanding.
6. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” provided opportunities for peer assessment and discussion which created a good learning environment.
7. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” with its peer assessment and discussion elements made me feel responsible for my own learning.
8. The “Direct Learning Common Programming Errors by Examples Strategy” with its requirement for students to carry out self-study made me feel responsible for my own learning.
9. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” with its peer assessment and discussion elements made me feel responsible for others’ learning.
10. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” involved peer assessment and discussion elements. I enjoyed giving peer feedback.
11. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” involved peer assessment and discussion elements. I enjoyed receiving peer feedback.
12. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” involved peer assessment and discussion elements. I was satisfied with the overall quality of the feedback that I gave.
13. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” involved peer assessment and discussion elements. I was satisfied with the overall quality of the feedback that I received.

14. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” involved peer assessment and discussion elements. The feedback I received helped me to identify common programming errors.
15. The “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” with its peer assessment and discussion elements benefited my learning.
16. In the “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy”, the instructor provided feedback immediately after we submitted our assignments. This effectively enhanced my understanding of common programming errors.
17. Quizzes were incorporated in both the “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” and the “Direct Learning Common Programming Errors by Examples Strategy.” This enabled me to understand my own level of comprehension of common programming errors.
18. Quizzes were incorporated in both the “Learning by Finding Common Programming Errors with Peer Assessment and Feedback Strategy” and the “Direct Learning Common Programming Errors by Examples Strategy.” This made me feel responsible for my own learning.

Students were instructed to respond to each statement by circling a number from 1-10 on a scale, with 1 representing “strongly disagree” and 10 “strongly agree.”

Biography

Wing-Shui NG is a teaching fellow in The Hong Kong Institute of Education for training and assessing pre-service and in-service teachers as well as conducting educational research. He taught Computer Subjects and served as the coordinator in secondary schools for many years. He was the person in charge of Information Technology in Education Committee and a member of School Administrative Council. Moreover, he has the experience of being seconded to Technology Education Section of Curriculum Development Institute of Education Bureau to develop New Senior Secondary Information and Communication Technology Curriculum, participate in school-based curriculum development, organize teacher training programmes and be invited as a speaker in educational seminars. He also contributed his efforts to serve as a reviewer of Computer Education Textbook Review Panel and a setter as well as marker of public examination papers for Hong Kong Examinations and Assessment Authority. In addition, he was appointed as the School Development Officer in Education Bureau. His research areas include Information Technology in Education and Assessment for Learning.