ASM Sc. J., 14, Special Issue 1, 2021 for ICSTSS2018, 55-62

An Assessment of Learning Content Model for Introductory Programming in Higher Education

Sin-Ban Ho[1*], Swee-Ling Chean[2], Ian Chai[1] and Chuie-Hong Tan[3]

[1] *Faculty of Computing and Informatics, Multimedia University, 63100 Cyberjaya, Selangor, Malaysia*

[2] *Lee Kong Chian Faculty Engineering and Science, Universiti Tunku Abdul Rahman, 43000 Kajang, Selangor, Malaysia*

[3] *Faculty of Management, Multimedia University, 63100 Cyberjaya, Selangor, Malaysia*

Learning content plays an important role in introducing programming concepts to undergraduates. Nevertheless, among the key challenges in learning content adoption include the numerous undergraduates' background characteristics. Learning content in introducing a programming language may take various forms depending on the level of mastery required. There is a need to identify how a particular form of learning content suits certain learners' background characteristics. We propose the LEAP model, tracking the students' Language proficiency in English (speaking, listening, reading, and writing), Educational background, Achievement in previous school examinations, and Programming experience. Furthermore, a programming learning assessment (PLA) model is proposed with four dimensions: True/False Questions, Multiple Choice Questions (MCQ), Fill in the Blank, and Time taken to fulfill the given learning task. The results show that the percentage score of the PLA model correlates with the LEAP categorical data. This finding supports that learning content principles lead to benefits in the field of introductory programming in higher education. The models can be enhanced by adding more variables in assessing the learning content for any other programming languages.

Keywords: e-Learning; higher education; information technology; learning content; programming

I. INTRODUCTION

Building good learning content to introduce programming is still an important concern in higher education. Programming assessment approaches can be customised, reused, and repurposed for the study of learning content (Bey *et. al.*, 2018; Chen *et. al.*, 2017; Krusche & Seitz, 2018). Information Technology enables the use of e-learning educational systems to help managing and researching learning content. Such systems are known as learning content management systems (LCMS). An LCMS central object repository has the advantage of being searchable and adaptable in e-Learning courses. Besides sharing materials, it allows a multi-user environment.

We identified two main problems. First, despite the advances in LCMS in helping to enhance learning content, failure rates on the familiarity to master the relevant parts of a framework are still high. One of the key challenges in learning content adoption includes the complication of prefabricated parts in object-oriented frameworks (Chai, 2000; Ho, 2008). These parts can be assembled to form a product faster than building every piece from the beginning. Especially if a framework is a white-box framework, the developer would be required to understand that internal structure to use it. Due to this, frameworks often have a steep learning curve (Alzaid *et al.*, 2017). Second, there is no model that relates the numerous undergraduates' background characteristics to programming learning assessment thus far. Modelling how the different learning assessments impacted from certain learners' background characteristics are still lacking at the moment of this writing.

Some literature have proposed instructional techniques, which use some educational tools (Gonçalves *et. al.*, 2018;

*Corresponding author's e-mail: sbho@mmu.edu.my

Lacave *et. al.*, 2018; Marques *et al.*, 2018). There is a widespread interest in learner-centred techniques (Clark & Dickerson, 2018). Nevertheless, the gaps in this area include a lack of essential attributes for content learning, and the lack of correlation of the LEAP (Language, Education, Achievement, and Programming) model towards learner characteristics (Magana *et al.*, 2018). Table 1 summarises the contributions and analysis of limitations with future enhancements in recent literature.

Table 1. A summary of the contributions and analysis of limitations with future enhancements on recent literature

	Comparison	
Ref.	**Contribution**	**Limitation/Future Enhancement**
[5]	Introduced student-centered and active learning (Think-pair-share, minute paper and pair programming) in Introductory courses. Significantly higher exam scored were obtained.	The experiment only involved a single instructor, so more studies would be needed for stronger and general conclusion.
[7]	Introduced instructional feedback that provides explanations to students based on their interaction with Project Management tool. Feedback was given immediately and helped students to gain a better understanding.	• Students' language proficiency and background might be threats to validity of the experiment result. • Instructional feedback may be expanded to address various learning preferences.
[14]	The research measured students' perceptions of difficulties in learning programming recursion through a preliminary instrument. The paper presented that neither motivation nor students' previous knowledge is the factor that affects their learning.	The sample size is only around half of the recommended dimension, needs more data for validity and reliability.
[15]	The study compared passive, active and constructive learning approaches among graduates and undergraduates. The results show that active learning benefits instructor-independent learners than instructor-dependent learners. The students still perceive that lectures are more useful.	Lack of correlation of the LEAP model towards learner characteristics.
[16]	Introduced reflexive weekly monitoring (RMX) to enhance students' learning experience. The students in monitored teams were more effective and make a more accurate diagnosis of their projects.	

We attempt to address these gaps through three main research objectives in this exploratory study: Firstly, we intend to identify the essential attributes of a good LCMS framework for learning introductory programming. There may be difficulties in coming up with a constructive framework to develop LCMS for programming courses. If we know the expectations and needs from the students, we can prepare a better approach to improve the programming achievement of programming learners. Secondly, we examine the LEAP model of undergraduates using introductory programming as a testbed for LCMS. We build the experiment to test the philosophy considering the correlation of the LEAP model towards learner characteristics. Thirdly,

we determine whether the LCMS improves programming learners' achievement via the PLA model. This also involves determining the performance of the students as a result of using the LCMS.

II. METHODOLOGY

The e-Learning field is becoming more important with the establishment of its trustworthiness and security (Miguel *et al.*, 2017). The cost of learning is affected by attributes of factors called cost drivers that refer to learning content size, such as bytes, total sections, paragraphs, files and folders, use of certain learning content, and learning capabilities of the learners, which we considered via the LEAP model, tracking the students' Language proficiency in English, Educational background, Achievement in previous school examinations, and Programming experience.

We highlight more details on the LEAP model steps here to give a clearer picture. The methodology in determining the undergraduates' language proficiency in English includes six steps: First, we find out their first language, whether English, Malay, Chinese, Tamil or another language. Second, we ask their English language qualification, i.e. Cambridge English, MUET, or None. Third, they rate their comfort in talking with friends in English outside of class in Likert scale of 1 (low) to 5 (high). Fourth, they indicate their percentage on how much they understood when they listen to lecturers speaking in English. Fifth, we collect their perception on the percentage they understand when they read English textbooks. Sixth, they rate their English proficiency in speaking, listening, reading and writing in terms of beginner, intermediate, fluent, or expert.

The educational background consists of five steps. First, we determine the type of primary school that they attended, whether national, independent, or home school. Second, they specify the medium of instruction used in their primary school. Third was the type of secondary school that they attended. Fourth was the medium of instruction used in their secondary school. Fifth, we enquire of their major streams, i.e. Science, Engineering, Arts, and/or Commerce.

The achievement in previous school examinations is assessed via four steps: First, we identify their highest qualifications at entry to the university/college, which could be Foundation, A-Level, SPM (Malaysian Certificate of Education), or STPM (Malaysian Higher Certificate of Education). Second, they indicate the result obtained in the highest entry qualification. Third, we determine their grade obtained in English subject. Fourth, they record their score in advanced or common mathematics.

For programming experience, there are six steps: First, we ask them whether they have ever taken any programming course before. Next, they proceed to the second step in answering when the first time they studied the programming course. In the third step, they note their first programming language, whether C, C#, Java, Visual Basic etc. The fourth step records the time they took the programming course, whether in the foundation programme, the schools that they attended or Diploma. The fifth step determines what medium of instruction used to learn programming. Finally, we find out what programming languages they are comfortable with, such as C, C++, Java, Python, and/or Visual Basic.

The experimental subjects of this study are undergraduates in introductory programming courses. In this study, the number of e-Learning subjects reporting the same remarks can be used as a measure of the significance of the qualitative findings made available at the end of the exercise. Therefore, we can attach the priority to improve the learning content for the subsequent studies. Here, we highlight guidelines for some of the measures observed throughout this research on the programming learning assessment (PLA) model.

(a) *Completion time estimation*: The cost of a project of learning technology includes the time and money expended in picking up new skills. The time factor is widely used in the assessment of programming tasks (Pieterse & Liebenberg, 2017; Tek *et al.*, 2018). First, the more data is collected, the more accurate will the estimate be. The major difficulty is that there may be new variables impacting upon the current task which have not been considered in the past. Second, a way to estimate completion time is through mathematical models.

(b) *Comprehension*: A high understanding score contributes to good students' perceptions of the given formative assessment (Ogange *et. al.*, 2018; Pyper, 2018; Voinea, 2018), that is the ability to determine what a programming framework does and how it works, by going through the accompanying learning content. Comprehension has an inverse relation to complexity, as the complexity of a framework in question increases, the comprehension tends to

decrease. This measure is assessed via the dimensions of True/False Questions, and Multiple Choice Questions (MCQ).

(c) *Workings*: Parihar *et al.* (2017) proposed grading scores to obtain work repair feedback, which supports our usage of this dependent variable of workings. This measure is an external consideration since its achievement requires knowledge from the framework as well as external factors such as the learning process and effort involved. We assess this measure using Fill in the Blank questions.

To draw on undergraduates' recent experience with learning content, this paper empirically reveals how the percentage score of the PLA model relates to the LEAP categorical data. We use a Moodle platform (Miguel *et al.*, 2017), i.e. Web-Based Learning Environment (WBLE) to support this research in teaching introductory C++ programming. The steps involve are: First, we design a Google form to represent the LEAP model. Second, we develop the programming learning content that consists of introduction of terminologies, examples of case studies for problem solving, and practical exercises using presentation software, word processing software and multimedia software. In addition to learning content, scaffolding and needed support was given to the subjects. Third, we build the programming self-assessment exercise with three quizzes to gather the data required for our PLA model. The topics covered in the quizzes include fundamentals of programming, programming modules and routines and control structures. Fourth, we test both the LEAP and PLA; improve any vague areas, before uploading the final production version to the Moodle platform. Fifth, we conduct the experiment in laboratory sessions using the Moodle platform. Once we collect the data, we use SPSS (Statistical Package for Social Science) to key in the data and analyse them. In addition, we also choose a car service application as another basis for one of our studies using Visual Basic .net (VB.net) (Irvine & Gaddis, 2012; Mayo, 2010; Schneider, 2011; Shelly & Hoisington, 2011).

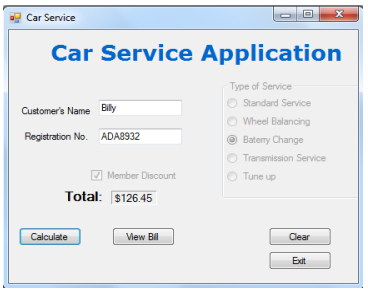
A. Experiment Details

Figure 1 shows the system sample code, which demonstrates the click event for the calculate button. The learning content is aimed to guide the undergraduates in completing the work task. We adopt a development approach for new ideas or pieces of information in transferring the knowledge so that the undergraduates can use them effectively. When they need to understand a piece of the program, the assimilation process enables them to use the framework. The undergraduates can follow the procedure in the learning content presented.

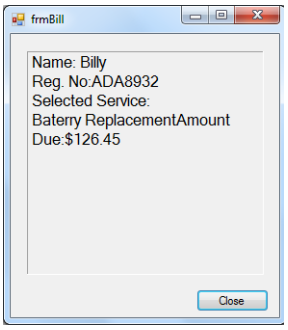
```
'Code for the click event
Private Sub cmdCalculate_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles cmdCalculate.Click
    Dim cost As Double
    gbTypeService.Enabled = False  'disable editing
    If rdbBattery.Checked = True Then
       cost = 140.5
    ElseIf rdbService.Checked = True Then
       cost = 66.0
    ElseIf rdbTransmission.Checked = True Then
       cost = 550.3
    ElseIf rdbTune.Checked = True Then
       cost = 110.0
'(code for wheel balancing option goes here)
'(code for dislaying msgbox goes here)
    End If
End Sub
```

Figure 1. Core part of the car service application

Figure 2 shows screenshots of main components of the application built by the participants.



(a)



(b)

Figure 2. Screenshots of the work task outcome: (a) Calculate Function, and (b) View Bill Function

III. RESULTS AND DISCUSSION

The central idea from the findings suggests that one can characterise the behaviours of undergraduates into four groups.

- *Excellent (E)* – The participants who work perfectly on the exercise variables and perform better than all the three average scores.
- *Good (G)* – This group often orally requests for help on the task at hand from time to time. They intend to perform the task that conforms to the learning content before the given time ends. They perform better than the average scores in two out of three variables.
- *Moderate (M)* – Although these participants do not score perfectly, they are able to record their achievement, albeit lower than the average scores. They perform better than the average scores in only one out of three variables.
- *Poor (P)* – These participants perform weaker than the average scores in all three variables, and hand in very rough answers.

We prioritise three major dependent variables for the PLA model. They are completion time, comprehension, and workings, which are summarised in the following description.

- *Completion time (COMPLTIME)* – The time taken for participants in finishing the given exercise.
- *Comprehension (COMPR.)* – The subjects are to answer True/False Questions and Multiple Choice Questions (MCQ), which test their knowledge on the introductory programming concepts.
- *Workings (WORK.)* – This evaluates how the undergraduates figure out the code fragment in filling in the blank of missing code, which consists of assigning default component settings.

Furthermore, we select three dependent variables so that we can use the hexagonal graph in Figure 3 to analyse them. With this, the participants can be grouped in the *E*, *G*, *M*, or *P* groups. When we navigate down the hexagonal graph, one of the variables, such as *x*, changes its negation, which is *x'* in Figure 3. In addition, *x* represents *COMPLTIME*, *y* represents *COMPR.*, and *z* represents *WORK*.

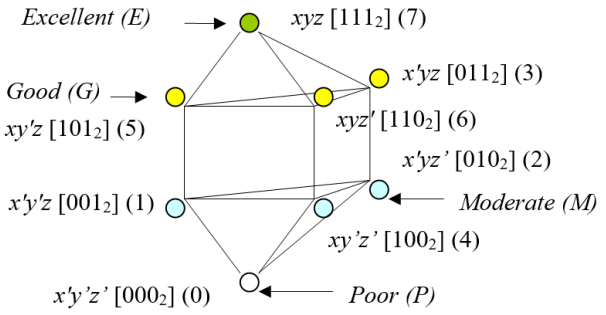


Figure 3. The learner model to transfer knowledge

The identification of people with performance in terms of abilities was the authors' idea inspired by the Karnaugh-Gray map (Ho *et al.*, 2011). It is relevant to the controlled experiment. Figure 3 does not have a cross edge, such as between node (3) and node (1). This is because each line represents one variable change only. For instance, the line connecting node (3) and node (2) in Figure 3 indicates the change of the *z* variable, i.e. *x'yz* to *x'yz'*. This conversion is similar to the Gray code cube (Rosen, 2007). In elaboration, *z* represents satisfactory Workings, while *z'* denotes a lower than the average Workings. The change of a variable to its prime version, e.g. *z* to *z'*, shows a lower achievement of *z*.

We represent the Completion time perspective as *x*, the Comprehension as *y*, and Workings perspective as *z*. For instance, the negation of completion time, *x'* denotes a lower than an average score. The variable without negation, *x*, stands for a participation that is better than the average. An empirical study (Magana *et. al.*, 2018; Juristo & Moreno, 2001; Wohlin *et al.*, 2000) could be viewed as a behaviour full of different characteristics of participants. By integrating Figure 3 and the Karnaugh map (Rosen, 2007), we obtain a two-dimensional view as depicted in Table 2. The Excellent group consists of *xyz*, which achieves all higher than average scores. The Good category has only one variable in negation, i.e. *xyz'*, *xy'z*, and *x'yz*. The Moderate group has two negation variables, i.e. *xy'z'*, *x'yz'*, and *x'y'z*. The Poor group performs worse than average scores in all the three variables, i.e. *x'y'z'*.

Table 2. A two dimensional view of the behaviours of learners

	yz	***yz'***	***y'z'***	***y'z***
x	Excellent (E)	Good (G)	Moderate (M)	Good (G)
x'	Good (G)	Moderate (M)	Poor (P)	Moderate (M)

Table 3 provides a summary of how to categorise the opportunistic model into four major groups. The sum of Rep7 is critically computed with value 7_{10}, which is equivalent to 111_2. A value of four is assigned to positive x when the participant's *COMPLTIME* is faster than the average of completion time ($\mu_{COMPLTIME}$). A value of two is for positive y when *COMPR.* is more than the average of comprehension ($\mu_{COMPR.}$), and value of one for positive z when the workings score (*WORK.*) is more than the average of workings ($\mu_{WORK.}$). The best performance group across the three critical dependent variables is identified as the *Excellent* group, with an accumulated value of seven. Meanwhile, the weakest group is called the *Poor* group, with a total value of zero.

Table 3. A summary of how to categorise the learner model into four major groups

Representation	***x (Value 4 when COMPLTIME <$\mu_{COMPLTIME}$)***	***y (Value 2 when COMPR. >$\mu_{COMPR.}$)***	***z (Value 1 when WORK. >$\mu_{WORK.}$)***	***Sum (x + y + z) => [Group]***
Rep7 (111_2) *xyz*	4	2	1	7 [*E*]
Rep6 (110_2) *xyz'*	4	2	0	6 [*G*]
Rep5 (101_2) *xy'z*	4	0	1	5 [*G*]
Rep3 (011_2) *x'yz*	0	2	1	3 [*G*]
Rep4 (100_2) *xy'z'*	4	0	0	4 [*M*]
Rep2 (010_2) *x'yz'*	0	2	0	2 [*M*]
Rep1 (001_2) *x'y'z*	0	0	1	1 [*M*]
Rep0 (000_2) *x'y'z'*	0	0	0	0 [*P*]

Note: Completion time perspective (x); Comprehension perspective (y); Workings perspective (z).

We highlight the contribution of the work here by elaborating the analysis of results collected from 65 respondents. The English score obtained from the LEAP model helps us to map the *Excellent* (*E*) and *Good* (*G*) groups to 32 fluent English speaking students. Meanwhile, the remaining 33 non-fluent students are mapped to the *M* and *P* groups. Among the dependent variables within COMPR from the PLA model analysed through a parametric test of MANOVA (Multivariate Analysis of Variance), we see that the treatments in *Y1True/False Question* has a strong significant difference at the 0.050 level (significance or p-value < 0.050). *Y3FillinTheBlank* has a marginal difference at the 0.100 level (p-value < 0.100). There is no significant difference between the groups in *Y2MCQ*, *COMPLTIME*, and *WORK*.

IV. OBSERVATION OF LEARNING CONTENT SUITABILITY

After we have carried out the few series of controlled experiments, we consider the following statements, which are proposed to be sound.

All object-oriented (OO) frameworks reuse components. Some OO frameworks are suitable for particular learning content, such as the Swing framework as observed in Ho (2008).

Therefore, some frameworks that reuse components are suitable to be documented with learning content.

We use first order logic formulas with typical connectors, i.e. $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (negation) and $\rightarrow$ (implication), plus the quantifiers of $\forall$ (universal), and $\exists$ (existential). The conclusion follows after the symbol of =>, which indicates the inference obtained from the premises.

$$\forall x\,[\,OO(x) \rightarrow RC(x)\,] \wedge \exists x\,[\,OO(x) \wedge LS(x)\,] \Rightarrow \exists x\,[\,RC(x) \wedge LS(x)\,] \quad (1)$$

The following denotation of predicates takes place:

OO: Object-oriented framework

RC: Reuse components

LS: Learning-content suitability

x: A particular framework variable to be tested, such as Swing, Visual Basic, C++, or Python

Inference Proof:

1. $\forall x\,[\,OO(x) \rightarrow RC(x)\,]$ P (Premise)
2. $\exists x\,[\,OO(x) \wedge LS(x)\,]$ P
3. $OO(c) \wedge PS(c)$ 2, Existential Instantiation (EI) to instantiate an element, *c*
4. $OO(c) \rightarrow RC(c)$ 1, Universal Instantiation (UI)
5. $OO(c)$ 3, Simplification
6. $RC(c)$ 4, 5, Modus Ponens
7. $LS(c)$ 3, Simplification
8. $RC(c) \wedge LS(c)$ 6, 7 Conjunction
9. $\exists x\,[\,RC(x) \wedge LS(x)\,]$ 8 Existential Generalization (EG)

q.e.d. (what was to be proven: *quat ermm demonstrantion*)

The sound intuition of number *9* from Existential Generalisation (EG) supports the sequence in seeking a proper way to find a good LCMS solution for a given framework, *x*. With the sound observation that some frameworks that reuse components are suitable for learning content, we explore further what frameworks (other than Swing framework), would benefit from the use of a particular learning content. This enforces the necessity in finding out the attributes of a good LCMS framework for learning introductory programming.

V. CONCLUSION

In this study, the measurement procedure can demonstrate a number of characteristics. This would give a more thorough discussion of the learning metrics, in terms of whether a learner is represented as excellent, good, moderate or poor (EGMP). We use ANOVA and MANOVA (Kent, 2015), and correlation (Flora, 2018) to assess the relationships between the LEAP and PLA model. All the subjects were undergraduates in a programming course in university. Thus, they were reasonably representative of undergraduates in higher education.

We had addressed our first research objective by identifying three essential attributes, i.e. completion time, comprehension, and workings to prepare a learner model to transfer programming knowledge. Next, our second objective is achieved via the methodology in the LEAP and PLA models as learning content assessment. We have shown the LEAP model steps in detail to distinguish our context from more than 90 acronyms for LEAP on the web. The overall presented methodology can be helpful to readers with detailed explanations for each measure.

The proposed LEAP, PLA and EGMP models can fill in the gap highlighted by Medeiros et al. (2019). Our findings further affirm that subjects with English proficiency performed significantly better than non-fluent subjects in True/False Question and Fill in the Blank questions. The PLA model also helped the fluent ones (similar to *E* and *G* groups) master the more advanced topics such those found in C++, as compared to another non-fluent group.

Finally, we achieve our third objective by examining the number of learners who fall into each of the four groups according to the learning content. We can divide the students into smaller groups of EGMP so that instructions can be tailored according to the learners' groups. This paves the way to measure the achievement in terms of the three crucial variables mentioned. When the instructor knows the learners' groups well, the instructor would be able to focus effort on the improvement of the moderate and poor groups so that they can perform better with the help of specific suitable learning content.

VI. ACKNOWLEDGMENTS

The authors would like to thank the financial supports given by the Fundamental Research Grant Scheme, FRGS/1/2019/SS06/MMU/02/4.

VII. REFERENCES

Alzaid, M, Trivedi, D & Hsiao, IH 2017, 'The effects of bite-size distributed practices for programming novices', in Proceedings of IEEE Frontiers in Education Conference (FIE 2017), Indianapolis, IN: IEEE, pp. 1-9.

Bey, A, Jermann, P & Dillenbourg, P 2018, 'A comparison between two automatic assessment approaches for programming: an empirical study on MOOCs', Educational Technology & Society, vol. 21, no. 2, pp. 259-272.

Chai, I 2000, 'Pedagogical framework documentation: how to document object-oriented frameworks: an empirical study', PhD thesis, University of Illinois, Urbana-Champaign, IL, USA.

Chen, LS, Chen, CC, Chang, SH & Yang, E 2017, 'An e-learning system for programming languages with semi-automatic grading', in Proceedings of 10th International Conference on Ubi-media Computing and Workshops, Pattaya, Thailand, pp. 356-361.

Clark, RM & Dickerson, SJ 2018, 'A case study of post-workshop use of simple active learning in introductory computing sequence', IEEE Transactions on Education, vol. 61, no. 3, pp. 167-176.

Flora, DB 2018, Satistical methods for the social & behavioural sciences: a model-based approach, SAGE Publications Ltd., London.

Gonçalves, RQ, von, Wangenheim, CAG, Hauck, JC, R & Zanella, A 2018, 'An instructional feedback technique for teaching project management tools aligned with PMBOK', IEEE Transactions on Education, vol. 61, no. 2, pp. 143-150.

Ho, SB 2008, 'Framework documentation with patterns: an empirical study', PhD thesis, Multimedia University, Cyberjaya, Malaysia.

Ho, SB, Chai, I & Tan, CH 2011, 'Evaluating documenting techniques on frameworks of object-oriented code', in Proceedings of IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE 2011), Parkroyal Penang Hotel, Penang, Malaysia.

Irvine, K & Gaddis, T 2012, Advanced visual basic ® 2010, 5th edn, Pearson Addison-Wesley, Boston, MA, USA.

Juristo, N & Moreno, AM 2001, Basics of software engineering experimentation, docdrecht, Kluwer Academic Publisher, the Netherlands.

Kent, R 2015, Analysing quantitative data: variable-based and case-based approaches to non-experimental datasets, SAGE Publications Ltd., London.

Krusche, S & Seitz, A 2018, 'ArTEMiS – An automatic assessment management system for interactive learning', in Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18), Baltimore, MD, USA.

Lacave, C, Molina, AI & Redondo, MA 2018, 'A preliminary instrument for measuring students' subjective perceptions of difficulties in learning recursion', IEEE Transactions on Education, vol. 61, no. 2, pp. 119-126.

Magana, AJ, Vieira, C & Boutin, M 2018, 'Characterising engineering learners' preferences for active and passive learning methods', IEEE Transactions on Education, vol. 61, no. 1, pp. 46-54.

Marques, M, Ochoa, SF, Bastarrica, MC & Cutierrez, FJ 2018, 'Enhancing the student learning experience in software engineering project courses', IEEE Transactions on Education, vol. 61, no. 1, pp. 63-73.

Mayo, J 2010, Microsoft ® visual studio ® 2010: A beginner's guide, McGraw Hill, New York, NY, USA.

Medeiros, RP, Ramalho, GL & Falcão, TP 2019, 'A systematic literature review on teaching and learning introductory programming in higher education', IEEE Transactions on Education, vol. 62, no. 2, pp. 77-90.

Miguel, J, Cabellé, S & Xhafa, F 2017, Intelligent data analysis for e-learning: enhancing security and trustworthiness in online learning system, Elsevier Inc., London.

Ogange, BO, Agak, JO, Okelo, KO & Kiprotich, P 2018, 'Student perceptions of the effectiveness of formative assessment in an online learning environment', Open Praxis, vol. 10, no. 1 , pp. 29-39.

Parihar, S, Dadachanji, Z, Singh, PK, Das, R, Karkare, A & Bhattacharya, A 2017, 'Automatic grading and feedback using program repair for introductory programming courses', in Proceedings of 22nd Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'17), Bologna, Italy.

Pieterse, V & Liebenberg, J 2017, 'Automatic vs manual assessment of programming task', in Proceedings of the 17th Koli Calling International Conference on Computing Education Research (Koli Calling '17), Koli, Finland.

Pyper, A 2018, 'Student perceptions of the implementation of formative assessment: a Royal St. George's College case study', The Young Researcher, vol. 2, no. 1, pp. 135-147.

Rosen, KH 2007, Discrete mathematics and its application, 6th edn, McGraw Hill, New York, NY, USA.

Schneider, DI 2011, An introduction to programming using visual basic ® 2010, 8th edn, Pearson Prentice Hall, Upper Saddle River, NJ, US.

Shelly, GB & Hoisington, C 2011, Microsoft ® visual basic ® for windows applications introductory, International Edition, pp 75–78, Central Virginia Community College, USA: Shelly Cashman Series, Course Technology, Cengage Learning.

Tek, FB, Benli, KS & Deveci, E 2018, 'Implicit theories and self-efficacy in an introductory programming course', IEEE Transactions on Education, vol. 61, no. 3, pp. 218-225.

Voinea, L 2018, 'Formative assessment as assessment for learning development', Journal of Pedagogy, vol. 1, pp. 7-23.

Wohlin, C, Runeson, P, Host, M, Ohlsson, MC, Regnell, B & Wesslen, A 2000, Experimentation in software engineering – an introduction, Kluwer Academic Publishers.