Chang, M. et al. (Eds.) (2019). Proceedings of the 27th International Conference on Computers in Education. Taiwan: Asia-Pacific Society for Computers in Education

Research on College Students' Practice Behavior Model after Class of Programming Course

Fang-Jing NING[a], Bao-Ping Li[a,b*]
[a] *Faculty of Education, Beijing Normal University, China*
[b] *Beijing Advanced Innovation Center for Future Education, Beijing Normal University, China,*
*libp@bnu.edu.cn

Abstract: In this paper, we develop the learning system-DQuiz used in the C language programming course of college students, which can support students to use the mobile terminal to carry out programming exercises at any time. The study selected 74 undergraduates from a university and used DQuiz to conduct a one-semester after-class study. During this period, this study collected the learning behaviors of students in the after-class exercises, including doing new questions, view answers, view explanation, collect question, review, comment and like. Through statistical analysis and lag sequence analysis, it is found that the high-score group students tend to comment more frequently, and the behavior probability of reviewing after collecting questions and viewing the answer after a review is higher, while the low-score group students tend to do new questions after reviewing. Based on this, the study suggested that in the after-class exercises:1. Students should be guided to participate in peer learning and mutual help behaviors. 2. Students should be guided to actively carry out knowledge review, and the teaching platform should push practice explanation to help students improve learn programming effects.

Keywords: Learning behavior, Programming learning

1. Introduction

With the development of smart-phones, tablets, and wireless mobile communication terminals, mobile learning has increasingly become a free and convenient way to learn. In traditional programming learning, debug is proved to be an effective way to improve students' programming skills (Hwang, Shadiev, Wang, & Huang, 2012). However, in the mobile learning environment, it is not suitable for students to debug on tables.

The data of learning behavior is the data generated by the learner during the learning process, such as the number of clicks, length of study, the progress of learning, and activity (Ji & Han, 2019), which can reflect the student's learning and learning characteristics (Zheng, Jiang, Yue, Pu, & Li, 2019). The method of analyzing these behavioral data is called learning behavior analysis that focuses on collecting traces that learners leave behind and using those traces to improve learning (Verbert & Duval, 2012). It can help teachers to understand students' learning situation and learning style, judge their enthusiasm for learning, position their roles, and provide important teaching intervention methods (Ji & Han, 2019). Studies have shown that in programming learning, learners' behavior in programming is often considered as a key factor for assessing how effective a methodology is for learning programming skills (Del Fatto, Dodero, & Gennari, 2016).

The mobile learning environment has a large amount of unstructured data that can store student learning behavior in the form of data records on the platform. Based on this, this study aims to collect college students' programming learning behaviors by using mobile practice Application, and to analyze college students' after-class practice strategies by analyzing the differences in behavior patterns of students with different learning levels, and to provide intervention strategies for teachers' after-class practice guidance.

2. Literature Review

2.1 Programming learning

At present, there are many learning platforms for assisting programming learning to enhance learning effects. Zingaro, Cherenkova, Karpova, and Petersen (2013) used the Python classroom response system, which teachers can use to give students questions during class. The types of questions include multiple-choice questions and code-writing questions. When the student submits the answer, the system will automatically provide immediate feedback on the type of error of the student according to a set of rules. The teacher can view the answers and feedback submitted by the student in real-time, so as to adjust his teaching plan in time. Students can gain the key skills of code writing through problem-solving skills and the ability to write code and debug code. Students and teachers can also communicate and discuss certain issues in the classroom. Hoffman, Lu, and Pelton (2011) used a web-based quiz system c-doku to provide students with a large number of code snippets, allowing students to fill in the input and output by reading the code. The system automatically checks the solution to provide feedback, which improves the student's code reading ability. The learning platform built by Basu, Biswas, and Kinnebrew (2017) provided students with programming blocks (e.g., conditions and loops) and hypertext resources (e.g., science books and programming guides). The platform let students practice conceptual concepts such as abstraction and decomposition by constructing conceptual models and computational models. The system also provides multiple-choice exercises, instructor guidance and suggested resource pages.

Most of these platforms provide students with questions, answers, communication, and other learning resources. At the same time, from the application scenarios of these teaching platforms, researchers pay more attention to learning or phased tests of the course, and rarely apply them in after-class exercises. In fact, after-class exercises are an important part of programming learning. After-class exercises can improve learning effects through test effects. Test effect means that taking the same time to test the learning materials that have already been learned, instead of repeating the learning, the learning effect of the material will be better.

In most universities in China, the programming course hours for non-computer majors are usually 2 to 4 hours per week. The first two hours are taught by the teachers, and the last two hours are programming exercises by students. However, it is not enough to practice only by the time of the class. Students not only have to practice new knowledge but also review and consolidate existing knowledge. This requires students to spend more time and practice a lot under the class. Therefore, the study of students' after-class programming practice strategies is of great value in improving students' programming ability.

2.2 Learning behavior analyze

learning behavior analysis focuses on collecting traces that learners leave behind and using those traces to improve learning (Verbert & Duval, 2012). The learning behavior analysis method has been initially applied to programming learning and has achieved some results. For example, by recording the information search behavior of the programming learners on the online discussion forum, it is found that the general programming learners indeed seek for information from discussion forums by actively searching and reading progressively according to course schedule topics. But compared to novices, advanced students consistently perform query refinements, examine search results and commit to reading (Lu & Hsiao, 2017). By recording the six types of learning behaviors in collaborative programming learning activities including completely independent, self-improving using, performing poor without, confident after, imitating, plagiarizing, it is found that learning behavior has a certain relationship with academic achievement(Hwang, Shadiev, Wang, & Huang, 2012).

A major obstacle to applying learning behavior analysis methods in the programming domain is the lack of meaningful observable actions for describing the students' learning process (Lazar, Sadikov, & Bratko, 2017). The learning behaviors associated with programming learning found in current research usually include the total amount of learners' exercises, the number of correct and wrong attempts, and the scores of exercises. For example, Spacco, Denny, Richards, Babcock, Hovemeyer, Moscola, and Duvall (2015) collected the number and percentage of students’ try and practice, the

number and the percentage of submission exercises from the platform of the CloudCoder novice programming exercise. It was found that the more times students try to practice, the better the ability of students' programming. Norris, Barry, Fenwick, Reid, and Rountree (2008) used the ClockIT toolkit to collect the event of students' project open or close events, package open or close events, compilation success, compilation error, compilation warning, invocation, file change or delete. It was found that the percentage of compilation errors and the type of compilation errors appear to relate to the student s performance on the project. Edwards, Snyder, Pérez-Quiñones, Allevato, Kim, and Tretola (2009) used the web-CAT automated scoring programming exercise platform to collect students' number of Submissions, time of the first submission, time of last submission, total elapsed time, and amount of code written. It is found that the first and last submissions of the high score students are earlier than the low score students.

Combined with the common functions of the programming learning platform, it can be found that there are more researches in programming learning itself, but other factors such as feedback, monitoring, and socialization in the programming learning process will also have an impact on learning. Therefore, reviewing answer, viewing explanation and other helpful feedback behaviors, collecting, review and other self-monitoring behaviors and comments, likes, and other social behaviors in the process of doing the new questions, are worth further exploration. Analyzing these behaviors will help to improve the student's programming learning with the exercises, especially to determine the sequence relationship between these behaviors, and to provide appropriate support strategies at the right time.

3. Research design

3.1 Research Platform

The study selected the mobile application DQuiz (Daily Quiz) for programming learning (Zhang, Li, Zhou, & Chen, 2019). The application supports students to conduct multiple-choice exercises after class and collects various types of programming learning behaviors. The core functions of the system are daily exercises, viewing answers, viewing explanation, collecting question, comment and like. After the student submits the answer, the system will immediately give correct and incorrect feedback. Students can only continue to choose whether to view the correct answer or view the explanation after submitting an answer. Students can also collect the title, and the collected questions will be automatically generated into the collection area. Students can view the questions in the collection area at any time. The collection question is a way of marking the question, which is convenient for the student to view the question next time.

3.2 Research Data and Method

In order to gain a deeper understanding of how students learn, especially if there is a difference in learning patterns. This study used lag sequence analysis to analyze students' learning behavior. Lag sequence analysis was proposed by Sackett (1978) to explore human behavior patterns by analyzing the significance of a behavior occurring after another behavior. This study encodes students' behaviors during the practice process, including seven behaviors, namely, doing new questions, reviewing, reviewing answers, viewing answer analysis, collecting questions, comment and like. The specific behavioral descriptions are shown in Table 1.

Table 1

Post-class practice behavior description

Behavior	Encode	Description
New Questions	NP	Students click the <Submit> tab to practice a new question.
View Answers	VA	Students click the <View Answers> tab to view a question's answer.
View Explanation	VE	Students click the <View Answers> tab to view

		an explanation of a question.
Collect Question	CQ	Students click the <Collect Question> tab to collects a certain topic.
Review	RE	Students click the <Submit> tab to practice the question again.
Comment	CM	Students click the <Comment> tab to ask questions or comments on a topic.
Like	LK	Students click the <Like> tab to like a comment.

3.3 Teaching Experiment

The study selected 74 freshmen from a university participating in the C language programming course, including 27 boys and 47 girls. These students are non-computer majors, which have no experience in programming. All students receive the same teacher's teaching at the same time and complete the same required after-class assignments. All students complete their studies in a natural state.

4. Result

4.1 Basic Situation

According to the final grades, the study divided the first 27% and the latter 27% of the students into high-score group and low-score group to study the differences in the behavioral patterns of the two groups of students. There were no significant differences between the two groups in terms of the number of new questions, online learning time, and programming basics (5-point scale).

Table 2

Student basic situation table

	M		T
	Low-score group	High-score group	
Pre-test (programming basics)	1.64	2.27	1.761
Number of new questions	223.25	224.70	1.237
Online learning time	10.71h	15.81h	1.640
Final exam score	59.05	94.51	14.018***
Number of exercises	22.15	30.06	1.646

4.2 Programming Learning Behavior Frequency

From the frequency of the behavior, high-score group students and low-score group students have no significant difference in the frequency of reviewing, viewing answers, viewing explanation and collection. According to the frequency of comments, the high-score group students were significantly higher than the low-score group students ($p>0.05$). Therefore, simply commenting and liking in the multiple-choice questions can help improve the learning effect. The frequency of other learning behaviors is not obvious, even on viewing answers and viewing the analysis of answers, low-score group students are higher than high-score group students.

Table 3

Independent sample t-test analysis of learning behavior frequency by knowledge score

Behavior frequency	M		Final exam score
	Low-score group	High-score group	T
New Question	65.3	70	0.193
View Answer	12.05	26.15	2.187*
View Explanation	61.7	41.8	-1.632
Collect Question	78.25	56.1	-1.757
Review	20.6	28.3	1.394
Comment	65.3	70	0.193
Like	12.05	26.15	2.187*

4.3 Programming Learning Behavior Pattern

The study defines a behavioral transformation into another behavioral sequence. After analyzing and summarizing 49 pairs of behavioral sequences, GSEQ (Generalized Sequential Querier) is used to statistically predict the probability of occurrence of student behavioral sequences (Z-score). And the study select the significant behavior (z>1.96) to draw the behavioral sequence conversion diagram of the two groups of students, more intuitively understand the difference in learning behavior patterns between the two groups of students as shown in Figure 1. The nodes in the graph represent behavior. The direction of the arrow indicates the direction of the behavior transition. And the number on the arrow indicates the Z-score of each behavior sequence.

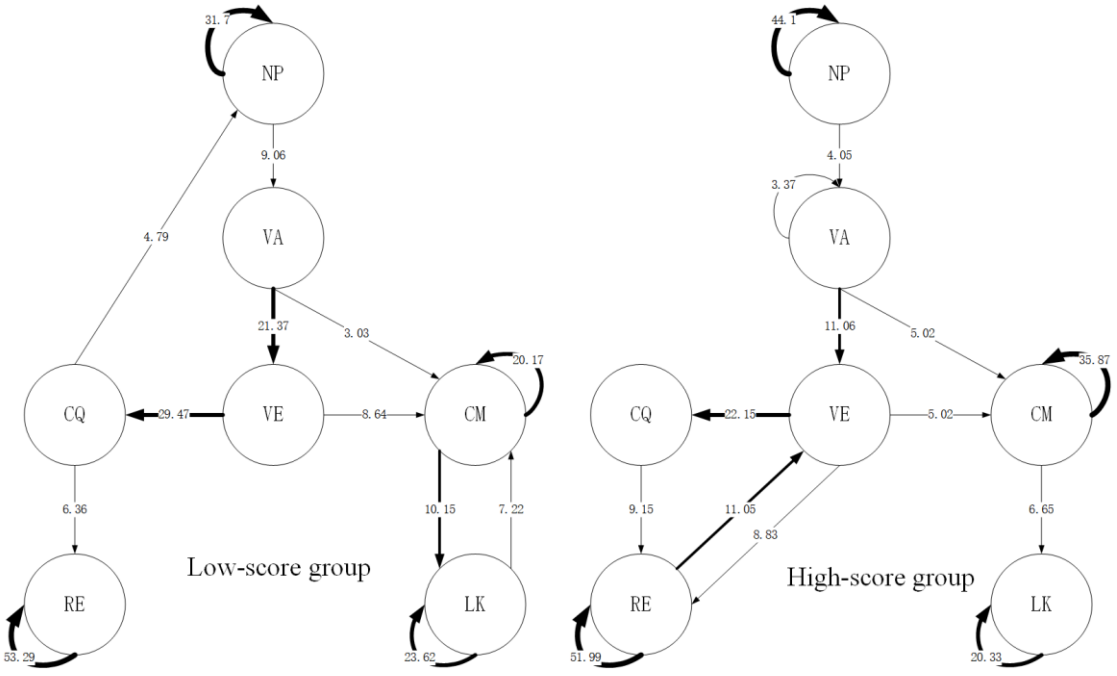

Figure 1. Low-score group (left) and high-score group (right) learning behavior transition map

It can be seen from the figure that the high-score group and the low-score group are basically similar in the behavior sequence, and can be summarized into several main behavior patterns.

(1) Doing new questions in a row: Do new question (NP)-do new question (NP). It complete new problems in a row.

(2) Do new questions and understand them: There are two paths: do new question (NP) - view answer (VA) - view explanation (VE)- comments (CM) - like (LK) and do new question (NP) - view answer (VA) - view Explanation (VE) - collect question (CQ). The subjective efforts of the former are more obvious, and the latter relies more on the automatic solution of the system.
(3) Review Questions: There are two paths: a continuous review (RE-RE) and review (RE) - view explanation (VE) - collect question (CQ) - review (RE).

However, in the above three aspects of behavior patterns, high-score group and low-score group are slightly different:
(1) Doing new questions in a row. The probability of high-score group students' behavior is higher than that of low-score group. It can be inferred that the high-score group students' first-time correct rate is higher so that the behavior sequence of doing the new question is less interfered by the error feedback.
(2) Do new questions and understand them. High-score group and low-score group behave differently after collect question (CQ) and comment (CM). Low-score group students will do new question (NP) after collecting question (CQ), while high-group students are more likely to enter the review (RE) after collecting question (CQ). Because in daily teaching, students often collect the wrong topics and review them later. The learning behavior pattern of collecting question and then doing new questions reflect that the students in the low-score group are more likely to wait for the teacher to explain in the class. On the contrary, the high-score group students tend to review and deepen the impression. Low-score group are significantly more interactive between comments and likes than highs. Because the practice frequency of low-score group is less than the high-score group, it means that students are more inclined to do many exercises in one time. Relatively speaking, the high-score group is a kind of decentralized learning and more frequent practice questions, which is also in line with the testing effect.
(3) Review Questions: High-score group students are more inclined to review (RE)-view explanation (VE)-collection question (CQ)-review (RE). During the review, the high-score group will often view the explanation (VE), forming a closed-loop of review. Answer explanation often explains the error in more detail and describes the knowledge of other option designs. This behavior pattern also reflects the behavior of high-score group students deep understanding exercises.

5. Research Conclusions and Recommendation

This study analyzes the frequency and sequence of high-score group students and low-score group students' learning behavior in one-semester practice. From the frequency of learning behavior, the more comments and likes, the better the learning effect, but not in reviewing, viewing answers, viewing explanation and collecting question, even low-score group students view answers and view explanation more frequently than high-score group students. This shows that participation in social learning can effect positively, but complete personal learning has limited effect on programming learning even frequently viewing answers effect negatively. From the perspective of the learning behavior sequence, the high-score group students tend to review after collecting the questions compared to the low-score group. The collected question is generally a wrong question or a difficult problem. After the collection, more review has been carried out, reflecting the timely consolidation and exploration of the problem by the high-score group. It's also important to look at when students view the explanation. Although the low-score group tend more frequently view explanation than the high-score group, the high-score group is more significant in behavior from review to view explanation and from view an explanation to review, indicating the importance of view explanation when review. Answer explanation is a more deeply explanation and combing than the answer, which can prevent students from being guess question correctly or misunderstood. For procedural knowledge such as c language, the learners tended to review the explanation of the question and re-answer it.

Based on the above conclusions, this study proposes the following two suggestions for the after-class exercises of programming learning.
(1) When practicing after class, it is very important that teachers and students, students and students communicate around the exercises. With the development of smart systems, the general teaching platform provides diagnostic feedback to help students answer relevant exercises, but this is not a substitute for communication between people. In the discussion area of the question, students can express their doubts, and teachers and students can give targeted explanations. Some of the possible reasons for doing question wrong are that knowledge is not mastered or the other details are not careful.

This question cannot be resolved from viewing the answer or viewing answer explanation. Free speech around the question enables students to achieve a transition from passive acceptance to active learning. (2) When practicing after class, the teacher should grasp the timing of the presentation of the answer explanation, and students should be encouraged to view explanation when reviewing. In normal teaching, there always exits situation that the teacher thinks that the student is mastered and the student is embarrassed to say no, or that the student does not realize that he or she does not master. Therefore, on the one hand, when the students review the question in the teaching system, the teaching system can actively present explanation. This will not make the students lose the opportunity to answer again because of viewing explanation, but also allow the students to prove their ideas when they are right. On the other hand, if there is no teaching platform, the teachers can also analyze the detailed answers explanation to the students by mail, group, etc.

Acknowledgements

The study is supported by Open Funding Project of the Key Laboratory of Modern Teaching Technology, MOE of PRC (Grant No. SYSK201802).

References

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. User Modeling and User-Adapted Interaction, 27(1), 5-53.

Del Fatto, V., Dodero, G., & Gennari, R. (2016). How measuring student performances allows for measuring blended extreme apprenticeship for learning Bash programming. Computers in Human Behavior, 55, 1231-1240.

Edwards, S. H., Snyder, J., Pérez-Quiñones, M. A., Allevato, A., Kim, D., & Tretola, B. (2009). Comparing effective and ineffective behaviors of student programmers. In Proceedings of the fifth international workshop on Computing education research workshop (pp. 3-14). Berkeley, USA: ACM Press.

Hwang, W. Y., Shadiev, R., Wang, C. Y., & Huang, Z. H. (2012). A pilot study of cooperative programming learning behavior and its relationship with students' learning performance. Computers & education, 58(4), 1267-1281.

Hoffman, D. M., Lu, M., & Pelton, T. (2011). A web-based generation and delivery system for active code reading. In Proceedings of the 42nd ACM technical symposium on Computer science education (pp. 483-488). Dallas, USA: ACM Press.

Ji, Y., & Han, Y. (2019). Monitoring Indicators of the Flipped Classroom Learning Process based on Data Mining-Taking the Course of" Virtual Reality Technology" as an Example. International Journal of Emerging Technologies in Learning, 14(3), 166-176.

Lazar, T., Sadikov, A., & Bratko, I. (2017). Rewrite Rules for Debugging Student Programs in Programming Tutors. IEEE Transactions on Learning Technologies, 11(4), 429-440.

Lu, Y., & Hsiao, I. H. (2017). Personalized Information Seeking Assistant (PiSA): from programming information seeking to learning. Information Retrieval Journal, 20(5), 433-455.

Norris, C., Barry, F., Fenwick Jr, J. B., Reid, K., & Rountree, J. (2008). ClockIt: collecting quantitative data on how beginning software developers really work. In Proceedings of the 13th annual conference on Innovation and technology in computer science education (pp. 37-41). Madrid, Spain: ACM Press.

Sackett, G. P. (1978). Observing behavior: I. Theory and applications in mental retardation. Oxford, England: University Park.

Spacco, J., Denny, P., Richards, B., Babcock, D., Hovemeyer, D., Moscola, J., & Duvall, R. C. (2015). Analyzing Student Work Patterns Using Programming Exercise Data. In Proceedings of the 46th ACM Technical Symposium on Computer Science Education (pp. 18-23). Kansas City, USA: ACM Press.

Verbert, K., & Duval, E. (2012). Learning analytics. Learning and Education, 1(8).

Zheng, S. Y., Jiang, S. P., Yue, X. G., Pu, R., & Li, B. Q. (2019). Application Research of an Innovative Online Education Model in Big Data Environment. International Journal of Emerging Technologies in Learning, 14(8), 125-138.

Zhang L., Li B., Zhou Y., Chen L. (2019) Can Fragmentation Learning Promote Students' Deep Learning in C Programming?. In: Chang M. et al. (eds) Foundations and Trends in Smart Learning. Lecture Notes in Educational Technology. Springer, Singapore

Zingaro, D., Cherenkova, Y., Karpova, O., & Petersen, A. (2013). Facilitating code-writing in PI classes. In Proceeding of the 44th ACM technical symposium on Computer science education (pp. 585-590). Denver, USA: ACM Press.