5-2012

Student usage patterns and perceptions for differentiated lab exercises in an undergraduate programming course

Heng Ngee MOK
Singapore Management University, hnmok@smu.edu.sg

Citation

MOK, Heng Ngee. Student usage patterns and perceptions for differentiated lab exercises in an undergraduate programming course. (2012). *IEEE Transactions on Education*. 55, (2), 213-217.
Available at: https://ink.library.smu.edu.sg/sis_research/1380

Student Usage Patterns and Perceptions for Differentiated Lab Exercises in an Undergraduate Programming Course

Heng Ngee Mok, *Member, IEEE*

***Abstract*—Differentiated instruction in the form of tiered take-home lab exercises was implemented for students of an undergraduate-level programming course. This paper attempts to uncover the perceptions and usage patterns of students toward these new lab exercises using a comprehensive survey. Findings reveal that these tiered exercises are generally very well received and preferred over their traditional "one size fits all" counterparts. Although the study does not show that tiered exercises have improved proficiency or scores, it does seem to indicate higher student engagement and motivation levels. Based on the survey results, a list of recommendations is put forth for the structure and format of tiered exercises that can be applied to future offerings of this programming course as well as to other similar courses.**

***Index Terms*—Computer science education, differentiated instruction (DI), differentiated learning, educational activities, higher education, student engagement and satisfaction, teaching/learning strategies.**

I. Background

THE WORK reported here was performed in an information systems (IS) school at a university in Singapore. During the first term of their freshman year, undergraduates majoring in IS need to complete a software programming course that is taught using the Java programming language. A traditional problem encountered by the teaching team is the huge variation in the initial programming skills of the students. The cohort of IS freshmen comprises "A-level" graduates, polytechnic graduates, and international students with a myriad of academic qualifications. Most of the "A-level" graduates have never done programming prior to matriculation, while most of the polytechnic graduates with IT/engineering-related diplomas have nontrivial experience with programming.

This bipolar readiness pattern results in two clusters of students who are at risk of becoming disengaged: the experienced programmers who find the course content repetitive, unchallenging, and hence boring, and the novices who find the course content novel, overly challenging, and a confidence damper. Given that the teaching team needs to "aim at the middle" when charting the learning objectives, syllabus, and content for this course, the ensuing challenge is how to engage the whole cohort of students using a standardized syllabus. Ideally, individual students should be allowed to learn at their own pace; proficient students should remain continually engaged with new challenges and content so that their programming skills will not stagnate, and less proficient students or novices should be given an encouraging climate to build up confidence and skills. It is also important that students in the latter group do not feel disadvantaged because they are learning with classmates with more experience in the subject matter. This problem with student diversity is not novel in Computer Science degree programs. Similar issues—especially in programming courses—have been widely reported [1]–[3].

A. Related Work

It is a trite belief that optimal learning is achieved when students are engaged. Barkley defines student engagement as "a process and a product that is experienced on a continuum and results from the synergistic interaction between motivation and active learning." She believes that teachers can create synergy between motivation and active learning by helping students work at their optimal level of challenge [4]. McKeachie also thinks that tasks given to students must be sufficiently difficult to pose a challenge, but not so difficult as to destroy the willingness to try [5]. Vygotsky's theory of "zone of proximal development" [6] suggests that productive learning results from learners operating in a situation that exposes them to concepts just slightly above their current level of development. Conversely, anxiety and a mismatch of task to skill threaten the "flow" potential that characterizes deep engagement [7].

These theories about motivation and engagement are related to a pedagogical approach adopted in heterogeneous classrooms called differentiated learning, differentiated teaching, or differentiated instruction (DI). DI is described as "a process to teaching and learning for students of differing abilities in the same class . . . (with the intention to) maximize each student's growth and individual success" [8], and has long been recognized as an effective approach in elementary and secondary schools to meet the learning needs of diverse student populations [8]–[11]. DI does not mean giving the weaker or stronger students more work to do; the focus is on providing each group of students in the same classroom with different kinds of work suitable for their individual learning needs. Differentiation can be applied in three areas: the content (curriculum and teaching materials), the process (the process of teaching or lesson delivery), or the product (the mode of assessment) [12]. A simple and successful case of differentiating content using tiered assignments has been documented by Suarez, who prepared three sets of exercises differing in difficulty for his high school math

Manuscript received June 08, 2011; accepted July 01, 2011.
The author is with the School of Information Systems, Singapore Management University, Singapore 187902, Singapore (e-mail: mok@ieee.org).
Digital Object Identifier 10.1109/TE.2011.2162070

lessons [13]. After his lecture session, students selected one of the three sets to work on. Quoting Glasser's choice theory [14], Suarez emphasized that the freedom to choose from the differentiated tasks empowered his students, increased their levels of motivation and enthusiasm, and resulted in higher achievement.

A recent survey of university faculty members teaching students' first course in programming seems to indicate that "differentiated teaching" is one of the five approaches employed to tackle student diversity, especially for the overachieving group [1]. However, compared to elementary and secondary schools, adoption of DI techniques is still at a relatively low level and considered to be slow for post-secondary education [15]. To demonstrate that DI can also be successfully realized in a university setting, Santangelo and Tomlinson conducted an educational psychology course for graduate students using DI techniques, and they reported optimized learning experiences [15]. For this programming course, face-to-face seminars were complemented with take-home programming lab exercises. The teaching team in the work reported here decided to apply a differentiated content approach using tiered lab exercises with the hope that this trial would meet with similar success.

B. Tiered Assignments

Although students are told to attempt these take-home lab exercises, they are not graded, and they do not even have to be submitted. In previous course offerings, every student had received the same set of exercises, each consisting of six to eight questions. It was decided to restructure them into the following formats.

1) Guided/unguided (G/UG) format: Each exercise comes in two documents released simultaneously to the students at the learning portal. The first (UG) document contains the questions, and the second (G) document contains the same questions with explanatory guides on how to arrive at the model solutions. It has to be emphasized that the G version does not merely show the model solutions, but the steps taken in order to reach them with relevant explanatory notes. Students are instructed to start working on the UG version and refer to the G version when they are unable to proceed.
2) Three-tiered format: Each exercise comes in four documents released simultaneously. The first is a diagnostic test comprising one question. The remaining three documents are the level-1 (L1), level-2 (L2), and level-3 (L3) exercises—all of which contain different questions. L1 questions are simple and guided; each question guides students through a programming example in a step-by-step manner to explain a single concept. L2 questions are the kind of questions students should expect to see in tests and examinations, and they should be attempted in order to meet the knowledge requirements for this course. L3 questions are optional and beyond the scope of the course syllabus. They include directed exploratory questions that introduce new concepts, with links to relevant Internet sites being provided for self-study. The model solutions (without guides) for L2 questions are also provided to students at a later date. Students are instructed to start with the diagnostic test. If they are unable to complete it, they are to start working on L1 and progress to L2. Otherwise, they should skip L1 and start working on L2. It was made clear that they would not be assessed on L3 content.

The G/UG exercises are designed for two tiers of students: The novices fall back on the G version, while proficient students can work on the UG version. The three-tiered exercises are designed for three tiers of students: The novices start working on simpler L1 questions, and after having acquired confidence and foundational skills, move on to the main set of (L2) questions. Proficient and motivated students can try the L3 questions after they have completed L2. These levels can be viewed as consecutive "zones of proximal development," and each student can choose to work in the most appropriate zone for optimal learning. Also, taking reference from [13], it is hoped that the empowerment given to students to self-rate and select their exercises will increase their motivation to complete them.

Eight lab exercises were given to all students for the Fall term of 2010: three presented in G/UG format, four in the three-tiered format, and one in the traditional format of a single set of questions.

II. Methods

This research project has two main research questions: 1) What are the usage patterns of the G/UG and three-tiered lab exercises? 2) How do students perceive them? A secondary objective is to discover noteworthy trends in how students attempt lab exercises in general. The results from this study can have broad applications in similar courses such as mathematics or science.

This study employed mixed methods and was divided into two phases. In phase one, four focus-group interviews were conducted with student volunteers the week after their final examination. The interviews were unstructured and aimed to elicit new themes that could be studied in the next phase. Interviewees were asked to "just talk" about their experiences with the lab exercises. The output from phase one was used to construct a set of 50 questions for phase two, in which 267 students who were given a grade for the course (including a "fail" grade) were invited to participate in an online questionnaire during December 2010. Respondents were required to identify themselves using their student e-mail addresses at the beginning of the survey so that their responses could be correlated to their course grades.

Because this study uses survey results as the source of primary data, the validity of the data depends largely on the accuracy of the respondents' input. In order to get candid responses, respondents were assured of the confidentiality of their responses even though the survey was not anonymous. The fact that the survey was done three weeks after the students had received their course grades could have influenced their responses to some of the questions, especially their self-perceived interest, enjoyment of programming, and importance of a good course grade.

III. Results

Only fully completed responses for the online questionnaire were taken into consideration. The response rate for this survey was relatively high at 43%.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

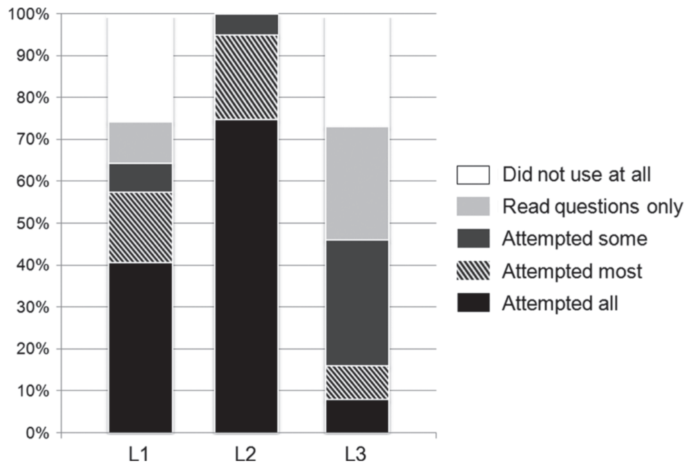

Fig. 1. Usage patterns of respondents for the three-tiered exercises. This chart shows the percentages of students who claimed not to have used the questions; read through the questions but did not attempt any; and attempted some, most, or all of the questions in each of the three tiers. All respondents attempted at least some of the L2 exercises.

A. Use of G/UG Exercises

When attempting their G/UG lab exercises, 66% of the respondents used both the G and UG versions, 30% used only the UG version, and 4% used only the G version. Most of the respondents who used both versions followed the prescribed instructions and started with the UG version, falling back to the G version only when necessary. However, a small fraction of students who were daunted by the UG version started working on the G version right away in order to build up confidence. Far from being a "misuse" of these exercises, this probably shows that the G version had helped this small group of students to get started. Several respondents who had used both versions had used the G version to check that they had done the exercises correctly. Respondents who preferred this mode of differentiation over the three-tiered exercises cite the usefulness of the G version when they were unable to proceed with the UG version. On the other hand, a few students suggested that the G version should be released one week later than the UG version in order to encourage students to try the questions without "assistance" first. There was also a comment to the effect that some students could be misled into thinking that the G version implies that there is only one correct way to solve the problem, which is definitely not the case for programming questions.

B. Use of Three-Tiered Exercises

Of the respondents, 58% attempted all or most of the L1 questions, 94% attempted all or most of the L2 questions, and only 16% attempted all or most of the L3 questions (Fig. 1). A total of 80% of the students followed the prescribed instructions by starting with the diagnostic test. Those who did not use the diagnostic test found it too easy or had already decided to start at either L1 or L2, regardless of the results. An interesting discovery is that two-thirds of the respondents who were able to complete the diagnostic test successfully went on to try at least some of the L1 questions, even though they could have skipped them. Respondents who did this explained that they wanted to validate their fundamental knowledge, to "warm up," to gain confidence, or get more practice. This seems to have reduced the usefulness of the diagnostic test as a triage instrument.

Of the respondents who attempted both L1 and L2 exercises, 73% agreed that the L1 exercises had helped them start the L2 exercises, implying that one of the primary objectives of the L1 exercises had been achieved. Respondents who disagreed complained that there was a gap between the L1 and L2 exercises, and that there should be a smoother transition in level of difficulty between the last questions from L1 and the first questions in L2.

When respondents who had not attempted the L3 exercises were asked to offer reasons for not doing so, the most common explanations were that they were "not on the syllabus" and "optional." High scorers (respondents who scored at least 80% for this course) who considered themselves interested in programming were "too busy" to attempt L3, and non-high scorers found them "too difficult." There was at least one respondent who attempted the L3 questions for the first few labs, but did not do so for the later labs in the course because the academic workload increased as the term progressed. Some suggestions to increase the number of students attempting L3 questions include: bundling L2 and L3 questions in the same document, providing more hints for L3 questions, providing model solutions for them, and providing more interesting questions that can be seen as relevant to their real-world context. Conversely, L3-attempters explained that they did the exercises "to challenge limits," "for fun," for a "sense of satisfaction/achievement," "curiosity," "to kill time," "extra knowledge," and "interest," with the term "challenge" appearing most frequently.

C. Preference for Differentiated Exercises

Respondents were asked to rank the three formats in order of preference: 1) traditional "one size fits all"; 2) differentiated in the form of G/UG; and 3) differentiated in the form of L1/L2/L3. It was quite clear that respondents prefer differentiated exercises to traditional ones: 11% ranked traditional exercises first, while the G/UG and three-tiered exercises garnered 41% and 47%, respectively. Comments in support of differentiated exercises include: "I have been able to think harder and learn more with the tiered set of questions"; "I really like the guided and unguided version of the lab as I become more independent as a learner . . . because . . . I may feel shy to ask very basic questions"; "We won't (need to) spend too much time on doing lab exercises as we can skip L1 questions and go straight to our level"; "Tiered (exercises) can cater to students of different capabilities"; and "I like that I can choose which level to do." Comments from L3-attempters include: "If it weren't for the tiered system, I'd be very bored . . . because of a lack of challenge given my prior background (in programming)"; and "I thought the tiered questions were very useful . . . L3 questions gave students like me a way of challenging myself and learning new things, which ultimately kept me very interested in doing the questions."

D. Inadequate Verification of Solutions

Another interesting finding is that when attempting their lab exercises, only half of the respondents whose code produced a "correct" output when executed went ahead to verify the correctness of their code. Most of those who verified their answers did so by showing their solutions to peers or by checking against model/guided solutions. The other half assumed that their code was correct because of the "correct" output. It is important that students be aware that poorly structured, inefficient, unnecessarily complicated code, or code not written in accordance with standard conventions, can still produce the "correct" output. Hence, students should be encouraged to verify their answers even if they seem to be correct.

E. Correlation Patterns

Respondents were asked to rate themselves on their interest in programming, their enjoyment of programming, and the importance they attached to achieving a good grade for this course. As expected, there is very strong positive correlation between self-perceived interest in programming and enjoyment in programming (Pearson's $\mathrm{r} = 0.9$). There is some correlation between enjoyment and perceived importance of the course grade ($\mathrm{r} = 0.51$), between interest and perceived importance of the course grade ($\mathrm{r} = 0.45$), between interest and the actual grades received ($\mathrm{r} = 0.42$), and between enjoyment and grades received ($\mathrm{r} = 0.45$). Surprisingly, the correlation between perceived importance of course grade and actual grade received is weak ($\mathrm{r} = 0.28$). Other interesting finds include weak correlations between interest and the rate of attempting L3 ($\mathrm{r} = 0.29$), as well as enjoyment and attempt rate of L3 ($\mathrm{r} = 0.28$). This probably means that besides interest and enjoyment, other factors (such as available time, confidence to complete L3) were at play in affecting whether a student attempts them. Also, students who obtained a better course grade tended to be those who attempted fewer L1 questions and more L3 questions. Nevertheless, it is impossible to derive any causal relationships among the factors studied.

F. Other Findings

The survey tried to uncover what avenues respondents took to find help when they met problems with their exercises, and whether they attempted the exercises individually or in study groups. Of the respondents, 96% attempted them individually. However, the importance of their peer networks becomes apparent when they met difficulties: 83% of the respondents approached peers for help, and 65% used the Internet to find solutions. Less than half of the respondents sought help from instructors, teaching assistants, and seniors (Fig. 2).

A happy finding is that most respondents agreed that they enjoyed and were interested in programming, although this finding may not be generalized to other cohorts or institutions. When respondents who did not attempt the L3 questions were asked to describe their classmates who did, common adjectives used were positive phrases that describe their motivation to learn ("self-motivated, interested, loved to be challenged, gung-ho, achievers, enthusiastic, hardworking") or emphasized their proficiency ("beyond the curve, intelligent, brilliant, above average, talented, experienced/good at Java, confident"). Other adjectives submitted include: "students who have time," "great time managers." Only four respondents used the word "kiasu" (a slightly derogatory adjective in a local dialect that literally translates to "afraid to lose out").

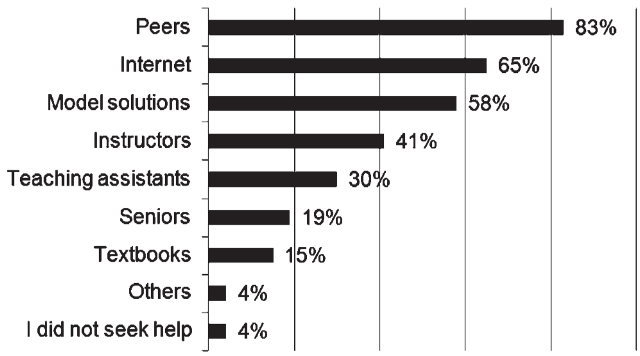

Fig. 2. Resources used by respondents to get help when they met problems with their lab exercises (respondents were allowed to select more than one choice). Considering that students were not formally placed into study groups or required to work together for this course, the results underscore the importance of the students' peer networks.

Besides the usual calls for the questions to be phrased more clearly, other miscellaneous feedback included a preference for more questions in each lab, arranging the questions in increasing order of difficulty, identifying the more challenging questions in L2 with an asterisk, and indicating the estimated completion time for each question in L2.

IV. Conclusion and Recommendations

On the whole, students prefer differentiated lab exercises to traditional "one size fits all" ones. The main strength of the G/UG exercises is the availability of "guides" that clearly explain how the model solutions are derived. The three-tiered exercises provide three distinct benefits: 1) for students who are not confident or require more practice, the L1 guided exercises provide a scaffold; 2) students who are already proficient do not have to "waste" time on L1; and 3) students who love challenges and have spare time can be engaged with L3. One obvious disadvantage with differentiated exercises is that much more effort has to be put in to prepare them (especially the "guides"). In order to combine the strengths of the G/UG and the three-tiered formats, it is recommended that future lab exercises be presented in three documents:

- L1 exercise: contains two to four guided questions.
- L2 exercise: contains 10–15 questions in increasing order of difficulty. There can be five to seven "normal" questions, one or two challenging questions (marked with *), followed by four to six optional questions (marked with **). L3 questions are rebranded as optional questions.
- Guided solutions for L2 questions, which may be released a week later.

The rebranding of the L3 questions as "optional" in L2 narrows the psychological gap between these two tiers and is expected to increase the attempt rate for this tier of questions among its target audience. The guided solutions for L2 questions follow the style of the G version of the G/UG exercises. Unlike model solutions, guided solutions are easier to follow and understand, and they are the primary reason for the popularity of the G/UG format. The first question in L2 serves as the diagnostic test to help students determine where they should start.

Hence, this question needs to be carefully designed to include the concepts that are covered in all the L1 questions. The elimination of a separate diagnostic test and a separate L3 document results in a simpler structure.

Other recommendations include the need to emphasize to students that there may be several acceptable ways to solve programming problems, and that they should always verify their answers even if they seem to produce the "correct" output when executed. The expected amount of time required to complete each question could be listed in the L2 exercises. If possible, the gap between L1 and L2 could be lessened by ensuring more continuity between the two tiers. As an example, an L1 exercise could have three questions, with each question covering one basic concept. Then, the first question in L2 would be a three-part "conglomerate" question, with each part corresponding to one of the L1 questions. The problem scenarios may be different, but the concepts tested are the same.

While this study cannot demonstrate that the differentiated lab exercises had a direct positive effect on the students' proficiency, the feedback collected seems to indicate higher engagement and motivation levels. Besides providing information on usage patterns and general trends on how students attempt their lab exercises, this study has identified useful recommendations for future differentiated lab exercises for this course—ideas which could be applied to other courses as well.

Acknowledgment

The author would like to acknowledge all students who responded to the survey, as well as the following colleagues who provided important suggestions: Dr. S. Miller, Dr. V. Shankararaman, Dr. J. Jiang, Dr. J. Shen, Y. L. Lee, V. R. Rao, J. Ducrot, F. Lee, and Dr. D. Lo. Dr. B. Lingard and the reviewers provided invaluable comments on this manuscript as well.

References

[1] J. Carter, S. White, K. Fraser, S. Kukovsky, C. McCreesh, and M. Wieck, "Motivating our top students," presented at the ITiCSE-WGR, Bilkent, Turkey, Jun. 26–30, 2010.

[2] H. C. Davis, L. A. Carr, E. C. Cooke, and S. A. White, "Managing diversity: Experiences teaching programming principles," presented at the 2nd LTSN-ICS Annu. Conf., London, U.K., 2001.

[3] T. Jenkins and J. Davy, "Dealing with diversity in introductory programming," in *Proc. 1st Annu. LTSN-ICS*, Edinburgh, U.K., 2000, pp. 81–87.

[4] E. F. Barkley, *Student Engagement Techniques: A Handbook for College Faculty*. San Francisco, CA: Jossey-Bass, 2009.

[5] W. J. McKeachie, *Teaching Tips: Strategies, Research, and Theory for College and University Teachers*, 9th ed. Lexington, MA: Heath, 1994.

[6] L. S. Vygotsky, *Mind in Society: The Development of Higher Psychological Processes*. Cambridge, MA: Harvard Univ. Press, 1978.

[7] J. E. Brophy, *Motivating Students to Learn*. Mahway, NJ: Erlbaum, 2004.

[8] T. Hall, N. Strangman, and A. Meyer, "Differentiated instruction and implications for UDL implementation," Nat. Center Access. Gen. Curriculum, Wakefield, MA, 2003.

[9] C. A. Tomlinson, K. Brimijoin, and L. Narvaez, "The differentiated school: Making revolutionary changes in teaching and learning," Assoc. Supervision Curriculum Dev., Alexandria, VA, 2008.

[10] D. Lawrence-Brown, "Differentiated instruction: Inclusive strategies for standards-based learning that benefit the whole class," *Amer. Secondary Educ.*, vol. 32, no. 3, pp. 34–62, 2004.

[11] A. Piggot, "Putting differentiation into practice in secondary science lessons," *School Sci. Rev.*, vol. 305, pp. 65–73, 2002.

[12] C. A. Tomlinson, *How to Differentiate Instruction in Mixed-Ability Classrooms*, 2nd ed. Alexandria, VA: Assoc. Supervision Curriculum Dev., 2001.

[13] D. Suarez, "When students choose the challenge," *Educ. Leadership*, vol. 65, no. 3, pp. 60–65, Nov. 2007, 2007.

[14] W. Glasser, *Choice Theory in the Classroom*. New York: Harper & Row, 1986.

[15] T. Santangelo and C. A. Tomlinson, "The application of differentiated instruction in postsecondary environments: Benefits, challenges, and future directions," *Int. J. Teaching Learning Higher Educ.*, vol. 20, no. 3, pp. 307–323, 2009.

Heng Ngee Mok (M'99) received the B.A.Sc. degree in computer engineering and M.Sc. degree in communication software and networks from Nanyang Technological University, Singapore, in 1999 and 2004, respectively.

He currently teaches software development and architecture at the School of Information Systems, Singapore Management University, Singapore.