Journal of Education and Learning; Vol. 5, No. 2; 2016
ISSN 1927-5250 E-ISSN 1927-5269
Published by Canadian Center of Science and Education

Correlates of Success in Introductory Programming: A Study with Middle School Students

Yizhou Qian[1] & James D. Lehman[1]

[1] College of Education, Purdue University, USA

Correspondence: Yizhou Qian, Beering Hall Room 3134, Purdue University, 100 N. University St., West Lafayette, IN, 47907, USA. Tel: 1-765-337-2667. E-mail: qian47@purdue.edu

Received: January 28, 2016 Accepted: February 23, 2016 Online Published: March 15, 2016

doi:10.5539/jel.v5n2p73 URL: http://dx.doi.org/10.5539/jel.v5n2p73

Abstract

The demand for computing professionals in the workplace has led to increased attention to computer science education, and introductory computer science courses have been introduced at different levels of education. This study investigated the relationship between gender, academic performance in non-programming subjects, and programming learning performance among middle school students with no prior programming experience who took an introductory programming course. We found that girls performed as well as or even better than boys in introductory programming among high-ability Chinese middle school students. However, we found that, instead of gender, students' performance differences in programming were better explained by their academic performance in non-programming subjects. Students' math ability was strongly related to their programming performance, and their English ability was the best predictor of their success in introductory programming for these Chinese students. Findings confirm previous studies that have shown a relationship between students' math ability and performance in learning to program, but the relationship between English ability and introductory programming was unexpected. While this relationship may be specific to students whose first language is not English, aspects of native language may pose hidden barriers that might affect all students' success in introductory programming.

Keywords: programming, gender differences, computer science education, natural language

1. Introduction

Although job prospects in computer science look very good in today's computing-intensive world, there is still a paucity of the workforce in computer and information science in the U.S. (Bureau of Labor Statistics, 2014-15). The demand for computing professionals in the workplace has led to increased attention to computer science education, and introductory computer science courses have been introduced at different levels of education. In the U.S., efforts, such as the National Science Foundation's CS10K initiative and the new CS Principles course of the College Board, are being made to broaden high school students' participation in computer science. However, computing and introductory programming may be introduced even before the secondary school level.

In the U.S., the Computer Science Teachers Association (CSTA) standards for computer science education address three grades bands: K-3, 3-6, and 7-12 (Seehorn et al., 2011), and to raise awareness of computing at the pre-secondary level the organization produced a special issue focusing on computer science in grades K-8 (Phillips, 2012). In the U.K., computer science is introduced to elementary school students (Brown, Sentance, Crick, & Humphreys, 2014). Even children as young as kindergarteners have been exposed to learning about robotics and programming (Sullivan & Bers, 2013).

However, regardless of the level at which computer science is introduced, introductory programming courses are notoriously difficult for learners (McCracken et al., 2001). Students' learning of computer science, like that of other subjects, can be understood through a lens of constructivist theory, which posits that students actively construct understanding by building recursively on the knowledge that they already possess (Ben-Ari, 2001). This construction of understanding can be particularly challenging in computer science because beginning students lack of a model of the computer (Ben-Ari, 2001).

Various instructional strategies are typically employed in beginning programming courses to help students construct their understanding. For example, the physical computing approach, which involves programming of electronic boards, can improve students' understanding of programming (Rubio, Romero-Zaliz, Mañoso, & Angel, 2015). Visualizing the steps of program execution can also help students to develop their understandings of how computer programs function (Sirkia & Sorva, 2012). For instance, Online Python Tutor is a well-known visualization tool that is widely used by instructors of introductory programming courses (Guo, 2013). Another commonly used instructional strategy in introductory programming is employing automated assessment learning systems (Douce, Livingstone, & Orwell, 2005). In these learning systems, students solve programming problems and submit their solutions. The systems can automatically assess students' solutions, provide immediate feedback, reduce instructors' workload, and significantly improve students' learning experience (De-La-Fuente-Valentín, Pardo, & Kloos, 2013; Wang, Su, Ma, Wang, & Wang, 2011).

Many factors, including prior knowledge, may influence students' success in introductory programming (Alvarado, Lee, & Gillespie, 2014; Bergin & Reilly, 2005; Rubio et al., 2015; Wilson & Shrock, 2001). If we are to engage more students into computing fields and improve our instruction in introductory programming, we need to better understand the factors that influence students' success.

Many studies on factors associated with success in introductory programming have been conducted over the past several decades. While a variety of factors have been investigated, including students' academic background, personal characteristics, and cognitive factors (Bergin & Reilly, 2005; Wilson & Shrock, 2001), three major factors have been studied across numerous studies: gender, previous computing experience, and academic performance in non-programming subjects.

1.1 Gender

Because of the low participation of women in computer science (Simard, Stephenson, & Kosaraju, 2010), the gender gap between females and males in computer science has been discussed by many researchers (Alvarado, Dodds, & Libeskind-Hadas, 2012; Guzdial, Ericson, Mcklin, & Engelman, 2014; Patitsas, Craig, & Easterbrook, 2014). Some previous studies have indicated that males have more positive attitudes towards computers than females (Alvarado et al., 2014; Beyer, Rynes, Perrault, Hay, & Haller, 2003; Dambrot, Watkins-Malek, Silling, Marshall, & Garver, 1985; Shashaani, 1997). For example, Shashaani and Khalili (2001), in a study of 375 Iranian undergraduates, found that even though female students strongly believed they had the same ability and competence in using computers as males, they showed less confidence in working with computers. Another study of 56 college students in a computer science course showed that while there were no gender differences in interests in computer science, females had much lower computer confidence (Beyer et al., 2003). However, other researchers have reported no significant gender differences in attitudes toward computers (Hattie & Fitzgerald, 1987). To determine an overall effect from multiple studies, Whitley Jr (1997) conducted a meta-analysis of studies on gender differences in computer-related attitudes and behavior and found that by comparison with females, males held more positive attitudes toward computers and believed they had higher computer-related competence (ES = .232, $p < .001$).

A few studies have indicated that male students show superior computer aptitude to females (Dambrot et al., 1985; Makrakis & Sawada, 1996). Based on the sample of 941 students in a psychology class, Dambrot et al. (1985) found that male students showed higher computer aptitude and math aptitude than female students, even though female students had higher high school and college GPAs. In a study of 773 ninth-grade students in Japan and Sweden, Makrakis and Sawada (1996) found that overall boys scored higher on computer aptitude and showed more positive attitudes toward computers, math, and science than girls. According to a recent study by Guzdial et al. (2014) of AP CS test takers in Georgia in 2013, male students showed much better performance than female students. However, other researchers have reported that females demonstrate the same performance as males in learning programming (Bruckman, Jensen, & DeBonte, 2002; Linn, 1985; Rubio et al., 2015).

1.2 Previous Computing Experience

Previous computing experience is another major factor that may contribute to success in introductory programming. According to a study (Kersteen, Linn, Clancy, & Hardyck, 1988) of freshmen and sophomores in an "Introduction to Computer Science" course, students' prior computing experience showed positive impact on their learning performance in this course. By studying 105 students in a CS1 introductory computer science course, Wilson and Shrock (2001) reported that students' grades were significantly correlated with their previous programming course experience. However, the positive impact of prior computing experience on learning to program may only be effective in the introductory programming course. In a study of students' performance in a sequence of programming courses, Holden and Weeden (2003) found that previous programming experience did help in the first programming course, but "it does not seem to have a significant impact on student performance in subsequent courses" (p. 45).

In addition, researchers have found a gender gap in previous computing experience. In general, males have more computer-related experience than females (Shashaani, 1997). Male students not only play more computer games than female students (Lockheed, 1985), but they also take more programming classes (Murphy et al., 2006). Interestingly, all types of prior computing experiences show a positive impact on females' learning performance in introductory programming; however, only certain types of previous experiences help male students (Taylor & Mounfield, 1994). For instance, too much game playing may lead to a negative effect on exam scores (Holden & Weeden, 2003).

1.3 Academic Performance

In addition to gender and prior computing experience, students' academic performance in other subjects may also influence success in learning to program. Previous research has revealed that students' academic performance in math has a positive relationship to their performance in introductory programming (Bennedsen & Caspersen, 2005; Bergin & Reilly, 2005; Wilson & Shrock, 2001). In a study of 235 college students in an object-first CS1 course, Bennedsen and Caspersen (2005) reported that students' math score from high school was the best predictor of their final grade in this course and explained over 15% of the variance in their predictive model. By investigating students' learning performance in the introductory programming course and their performance in the Irish Leaving Certificate (LC) examinations in mathematics and science subjects, Bergin and Reilly (2005) found that mathematics and science both significantly correlated with students' performance in programming. It is not surprising that the results of these studies suggest that ability in disciplines such as mathematics and science is an important predictor of students' learning performance in introductory programming courses.

1.4 Research Questions

Although there have been studies of factors that influence students' success in programming (Alvarado et al., 2014; Bergin & Reilly, 2005; Rubio et al., 2015; Wilson & Shrock, 2001), most of these studies have focused on college students. A few previous studies have discussed the gender effects in the learning of programming among middle school students (Bruckman et al., 2002; Linn, 1985), but these studies did not discuss other factors that might affect students' programming learning. More importantly, according to previous research, factors such as gender may or may not influence performance in learning to program depending on the age of the students (Bruckman et al., 2002; Guzdial et al., 2014; Sullivan & Bers, 2013). Hence, because introduction to computer science courses have spread from college to high school and now even into lower grades levels, researchers and educators should investigate factors contributing to programming success for pre-college learners. From an instructional perspective, it is also important to identify factors that influence success in learning to program. If educators know what factors are important for the success in computer science, they can better prepare their instruction to support and enhance students' learning.

The purpose of this study was to investigate the relationship between gender, academic performance in non-programming subjects, and programming learning performance among middle school students with no prior programming experience who took an introductory programming course. The following research questions guided the study:

1) Is there a difference in the performance of boys and girls in learning programming?

2) Are there any relationships between gender, academic performance in non-programming subjects, and programming learning performance?

2. Methods

2.1 Subjects

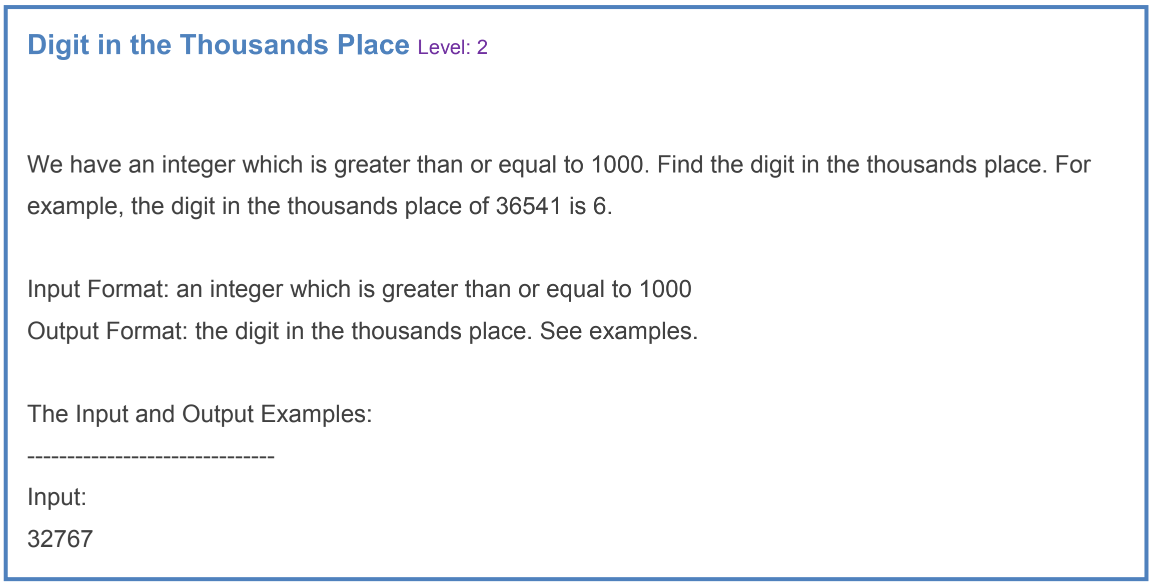
The research subjects in this exploratory study were two groups of students taking an introductory computer programming course in 7th grade from a middle school in an eastern city in China. Group A had 33 students, 17 boys and 16 girls. The average age of Group A was 12.3. Group B had 36 students, 23 boys and 13 girls. The average age of Group B was 13.2. Both groups of students were high-ability students who scored in the top 10% of the high-ability identification exam of this school. Normally, in this city, elementary school education is six years long and middle school education is three years long. Due to the students' high academic ability, the students in Group A skipped the 6th grade and started the three-year-long middle school education directly when they finished the coursework of the 5th grade. On the other hand, the students in Group B finished all six years of elementary school education, but they were scheduled to complete the three-year-long middle school in just two years. None of the students had taken any computer programming courses before. The core academic courses required for both groups of students were math, Chinese, and English.

2.2 Procedures

Students from both groups took an introductory Pascal programming course taught by the same instructor (the first author) during the same semester. Students took this course for 14 weeks, and during every week students participated in a 90-minute block. However, because of the holiday schedule of the school, students in Group B missed 2 blocks and so had less programming practice time than students of Group A.

This course employed an automated assessment system as part of the instruction in order to provide students opportunities of actively constructing new knowledge and enhance their experience of learning to program (De-La-Fuente-Valentín et al., 2013; Douce et al., 2005; Wang et al., 2011). The automated assessment learning system used in the instruction was the SangTian Programming Learning Environment (SPLE), which was designed by the course instructor. SPLE has a problem pool which contains dozens of programming problems for students to solve. All problems were presented in Chinese; a translated example problem is shown in Figure 1. Students started to use SPLE in the 5th week. Every class block, the instructor first introduced new course content such as the syntax of using for loops in Pascal. After about 10-minute lecture, the instructor would demonstrate how to use the new statements to solve programming problems. Examples used in the demonstration usually were similar to the problems in SPLE. After the lecture and demonstration, students had around 60 minutes to solve problems in SPLE individually. The instructor provided necessary support when a student had difficulties and asked for help. Students accessed SPLE only during the class time and did not have assignments after class.

To solve the problems, students were required to write short programs to produce the correct output. All the solutions students submitted to SPLE were graded by the system automatically. Immediate feedback was provided to inform the student whether the submitted solution was correct or not, and if syntax or logic errors existed. While solving a problem, the student accumulated experience points according to the difficulty of the problem. The difficulty of problems was categorized from Level 1 (low difficulty) to Level 10 (high difficulty). The more difficult the problem the student solved, the more experience points he or she could earn. Accruing experience points also raised the student's account level. If a student with a high level account solved a low difficulty-level problem, he or she could only earn half of the experience points available for the problem. Thus, students were encouraged to solve higher difficulty level problems as they gained experience. Once a student solved a certain problem, he or she could not solve it again. If a student did well in programming and solved more high difficulty level problems, he or she could earn more experience points. The total possible experience points students could earn in SPLE was around 3200, and the highest level of student account was 10. Based on the accumulative evaluation system, the final experience points students earned were used as a proxy measure of their learning performance in this course. At the end of the semester, students' learning performance data were extracted from the SPLE database and analyzed.



Digit in the Thousands Place Level: 2

We have an integer which is greater than or equal to 1000. Find the digit in the thousands place. For example, the digit in the thousands place of 36541 is 6.

Input Format: an integer which is greater than or equal to 1000
Output Format: the digit in the thousands place. See examples.

The Input and Output Examples:

```
--------------------------------
Input:
32767
```

Figure 1. A Translated Example Problem in SangTian Programming Learning Environment (SPLE)

In addition, students' academic records of their core courses were accessed to gather data about their performance in other subjects. We calculated the averages of the formal exam scores in each subject for each student. The maximum score on each of these exams was 100 points. These average scores were used as the measure of students' academic performance in each subject.

2.3 Analysis Performed

Various statistical techniques were employed for data analysis. We applied t-tests to compare boys' and girls' programming learning performance in both groups. Correlations were computed to analyze relationships between gender, academic performance in non-programming subjects, and programming learning performance. In addition, stepwise regression analyses were used to identify important factors that predicted students' programming learning performance. For the stepwise regression, we set significance levels of F to enter (FTE) of $\alpha = 0.15$ and F to delete (FTD) of $\alpha = 0.075$ consistent with recommendations of Derksen and Keselman (1992). As gender is a categorical variable, when computing correlations and regressions, we coded gender as male equals 0 and female equals 1.

3. Results

3.1 Gender Differences in Programming Learning Performance

Overall, in this study, girls performed better than boys in learning programming as measured by SPLE experience points (See Table 1). In Group A, girls gained more experience points in the SPLE than boys; although the difference ($t(31) = 1.88$, $p = .07$) is not statistically significant at the $\alpha = 0.05$ level, the trend suggests that girls performed better than boys. In Group B, girls gained significantly more experience points in SPLE than boys ($t(34) = 2.69$, p = . 01). Therefore, our results suggest that there was a difference in the performance of boys and girls in learning programming with girls showing better learning performance than boys.

Table 1. Gender differences in programming learning performance

Group	Males			Females				
	Mean	SD	n	Mean	SD	n	t	p-value
A	1827.8	601.5	17	2199.8	530.9	16	1.88	.07
B	1697.4	567.8	23	2190.2	442.7	13	2.69	.01

3.2 Relationships between Gender, Academic Performance, and Programming Learning Performance

Correlations between students' programming performance, as measured by experience points in SPLE, and their academic scores in non-programming subjects were computed (see Table 2). In Group A, students' programming performance as measured by experience points was positively correlated with their English score ($r = .57$, $p < .001$), math score ($r = .45$, $p < .01$), and Chinese score ($r = .36$, $p < .05$). In Group B, students' programming performance as measured by experience points was positively correlated with their English score ($r = .58$, $p < .001$) and math score ($r = .45$, $p < .01$). In both groups, our results showed that the relationship between students' English performance and programming learning performance was stronger than the relationship between students' math performance and programming learning performance.

Stepwise regression analyses were conducted to investigate whether the various factors studied were able to predict students' programming learning performance (See Table 3). Independent variables were gender, math score, Chinese score, and English score. In Group A, a stepwise regression method resulted in a significant model with only the English score as the predictor, $R^2 = .32$, $F(1, 31) = 14.67$, $p < .001$. Students' English score significantly predicted the programming learning performance, $\beta = 140.37$, $p < .001$, explaining 32% of the variance. In Group B, a stepwise regression method also found a significant model with only English score as the predictor, $R^2 = .34$, $F(1, 34) = 17.51$, $p < .001$. Students' English score significantly predicted the programming learning performance, $\beta = 90.67$, $p < .001$, explaining 34% of the variance. In the regression model, gender was no longer important in explaining the difference in programming learning performance when the English score was considered. Therefore, these results indicate that the observed gender differences in programming learning performance could be better explained by other factors such as students' English ability.

Table 2. Correlations between programming learning performance and academic performance

Group A (n = 33)

	Programming	Gender	Chinese	Math	English
Programming	-				
Gender	.32	-			
Chinese	.36*	.66***	-		
Math	.45**	-0.21	-0.10	-	
English	.57***	.27	.48**	.47**	-

Note. * *p* < .05. ** *p* < .01. *** *p* < .001

Group B (n = 36)

	Programming	Gender	Chinese	Math	English
Programming	-				
Gender	.42*	-			
Chinese	.18	.18	-		
Math	.45**	-0.12	.21	-	
English	.58***	.36*	.43**	.60***	-

Note. * *p* < .05. ** *p* < .01. *** *p* < .001

Table 3. Summary of stepwise selection

Group A

Step	Variable Entered	Variable Removed	Number Vars In	Partial R-Square	Model R-Square	C(p)	F Value	Pr > F
1	English		1	0.3212	0.3212	5.1807	14.67	0.0006

Group B

Step	Variable Entered	Variable Removed	Number Vars In	Partial R-Square	Model R-Square	C(p)	F Value	Pr > F
1	English		1	0.3400	0.3400	7.1916	17.51	0.0002
2	Gender		2	0.0508	0.3908	6.1747	2.75	0.1066
3		Gender	1	0.0508	0.3400	7.1916	2.75	0.1066

4. Discussion

4.1 Gender and Learning Performance in Programming

The result in this study showed that overall girls performed as well as or even better than boys in learning programming. In Group A, girls gained more experience points on average, but the difference was not significant. In Group B, girls performed significantly better than boys. This result is contrary to some studies that have shown males to demonstrate superior performance to females (Dambrot et al., 1985; Guzdial et al., 2014) but consistent with other previous studies of children or middle school students (Bruckman et al., 2002; Linn, 1985; Sullivan & Bers, 2013). For example, in one study of middle school students, Linn (1985) stated, "at several schools, females had higher means than males on this assessment [Final Programming Assessment], although the differences in mean scores were not statistically significant" (p. 235). In a study of kindergarteners, Sullivan and Bers (2013) indicated that no gender differences were found in robotics and programming achievement.

Previous studies that have indicated a gender difference that appears to favor males with respect to attitudes towards computers (Beyer et al., 2003; Shashaani, 1997), computer aptitudes (Makrakis & Sawada, 1996), and AP CS test performance (Guzdial et al., 2014) were primarily studies of high school and college students. There is good evidence that the gender gap in computer science does not exist among younger students (Gürer & Camp, 2002). As females' participation in computer science is still low (Simard et al., 2010), if we are to engage more people into computing fields, researchers and educators should consider exposing all students to computer-related courses before the gender gap arises. An introductory programming course for middle school students, such as the course that was the object of this study, might be a possible choice, and computing concepts can be introduced to students in even earlier grades (Phillips, 2012).

4.2 Math, Programming, and Gender Stereotypes

Our results indicated that even though girls performed as well as or even better than boys in learning to program, gender was not the primary factor in explaining the differences when we considered other factors, such as students' English and math ability. The influence of English and math ability on students' programming performance is not a surprise given that constructivist theory suggests that students' construction of new knowledge is built upon the foundation of existing knowledge (Ben-Ari, 2001). According to our results, students' academic performance in math showed a significant positive relationships with their programming performance. Many previous studies have revealed that math ability is an important predictor of students' success in programming (Bennedsen & Caspersen, 2005; Bergin & Reilly, 2005; Wilson & Shrock, 2001). This connection between math and programming may be an important consideration when addressing the gender gap in computer science. First of all, female students have been reported to be less confident than male students in their math abilities (Sax, 1994). In addition, although math abilities of young boys and girls are almost equivalent, male students start to show significant superiority in high level math skills by the end of middle school (Entwisle, Alexander, & Olson, 1994). Moreover, culture and social environments can affect people's perception of math (Guiso, Monte, Sapienza, & Zingales, 2008; Hyde, Lindberg, Linn, Ellis, & Williams, 2008). Hyde et al. (2008) stated "Stereotypes that girls and women lack mathematical ability persist and are widely held by parents and teachers" (p. 494). As a gender gap and stereotypes exist in math, girls may be enculturated to think computer science, which is highly related to math, is a masculine field as well.

Therefore, as computer science and math are closely linked together, both in the performance and perception, educators should work to counteract stereotypes about gender, math, and computer science among students as early as possible. We suggest that we should provide students with programming experiences at a young age and help them to develop non-stereotyped and gender-neutral perceptions of computer science.

4.3 English Ability, Natural Language, and Success in Programming

In this study, the results indicated that students' English ability showed the strongest relationship with programming learning performance and was the most important factor in explaining the differences in programming. In Group A, both programming learning performance and English scores of girls were higher than those of boys, although the differences were not statistically significant. In Group B, both programming learning performance and English scores of girls were significantly higher than those of boys. Thus, the superior performance of girls in English may have translated into superior performance in learning to program computers. Previous research has shown that math and science abilities are important factors that contributed to the success in programming (Bennedsen & Caspersen, 2005; Bergin & Reilly, 2005). However, these studies were based on native English speakers. It is possible that for Chinese students English ability is a more important factor that contributes to programming success, because the programming language used in this study was based on English.

For example, in Pascal, the reserved word "const", which is short for "constant", is used to declare constants. However, these students had not learned the word "constant" in their English class when they took the programming course. Hence, they may have had difficulties in understanding the Pascal reserved word "const." Students may have come to understand the purpose of this and other words in Pascal, but they were hampered by not knowing the meanings of the English words on which they were based. As another example, students had learned the math concept "real number", which in Chinese is called "shi shu", and also knew the English words "real" and "number", but they did not know "real number" means "shi shu". Thus it was hard for them to understand the data type "real" in Pascal.

Spelling was likely another obstacle these Chinese students faced in learning programming. For example, one student often misspelled the function “writeln” as “writeIn”, because he could not tell if there should be a lower case “l” or an upper case “I” in it. If he knew “writeln” was short for “write a line”, he might not have had the confusion. Therefore, it makes sense that the English ability of Chinese students may have an effect on their ability to learn to program.

Although the relationship we observed could be specific to students whose first language is not English, previous studies based on native English speakers have also indicated that students’ natural language can sometimes result in programming errors. According to Du Boulay (1986), English words in programming languages have specific meanings, but the student may not understand them as expected. For example, the word “and” is a Boolean operator in many programming languages, but it is a conjunction in everyday use. Du Boulay (1986) stated, “Exasperation with a programming system can occasionally be caused by the mismatch between the designer’s and the user’s understanding of what is implied by a particular name” (p. 62). Inappropriate use of natural language is another source of programming bugs of novices. After analyzing students’ programs, Bonar and Soloway (1985) concluded that novices often made misleading links between SSK (step-by-step natural language programming knowledge) and PK (Pascal programming knowledge). For example, students may interpret the word “then” in the if-then-else construct in a programming language as it is used in natural language. Then, they may add a “then” to the repeat-until construct, which is incorrect in Pascal syntax. To help address this issue, some natural-language-style programming languages such as Hypertalk have been developed (Bruckman & Edwards, 1999; Kelleher & Pausch, 2005). One such language, MOOSE, was deliberately designed to help children learn reading, writing, and computer programming (Bruckman & Edwards, 1999). The programming language MOOSE tries to keep balance between natural language and a regular syntax. After analyzing 2970 errors made by sixteen children using MOOSE, Bruckman and Edwards (1999) indicated that children were less likely to make natural language errors in a natural-language-like programming language such as MOOSE than in a formal language like Pascal.

Because students’ natural language may also affect their learning in programming, researchers and educators should pay attention to natural language issues. While girls may have better verbal skills than boys at this age (Guiso et al., 2008), whether this advantage may result in positive or negative effects in learning to program has not been researched. Obviously for these Chinese students who were just starting to learn English, better English ability provided benefits in learning to program. However, for native English speakers, further research is needed. Although previous research has investigated various factors that might contribute to success in introductory programming, including gender (Alvarado et al., 2014), prior computing experience (Bruckman et al., 2002), and academic performance (Bergin & Reilly, 2005), there may be more factors that are important but seldom researched, such as natural language ability. Therefore, to better understand the factors that influence success in learning to program, researchers need to consider the full range of variables that may contribute to these differences.

Finally, in our study the results showed that students’ English ability was more important than their math ability in learning to program, but this may be an artifact of the sample. First, all the students in this study were high-ability students with strong math abilities. Because all of the students had strong abilities in math, there may not have been sufficient variation to account for the differences in programming performance. Second, according to the data of this study, boys and girls did not show differences in their academic performance in math. Finally, the math skills required in this introductory programming course were less advanced than those required in high school or college level programming courses. Thus, the importance of math in explaining performance differences in programming may have been reduced in this study compared to previous studies with older students.

5. Conclusions

We found that girls performed as well as or even better than boys in introductory programming among high-ability Chinese middle school students. However, gender was not the primary factor in explaining the performance difference in programming. We found that, instead of gender, students’ performance differences in programming were better explained by their academic performance in non-programming subjects. Students’ math ability was strongly related to their programming performance, and their English ability was the best predictor of their success in introductory programming for these Chinese students. Our findings confirm previous studies that have shown a relationship between students’ math ability and performance in learning to program, but the relationship between English ability and introductory programming was unexpected.

In reviewing these results, several limitations of this study should be considered. For one, all of the students were high-ability and so may not be representative of the general population of students learning to program computers. Likewise, because all of these students were English language learners, the influence of English on programming performance of these students may be different than it would be for native English speakers. Finally, the use of experience points in the computer system as a proxy for programming learning performance might be influenced by other factors, such as student persistence, that could vary across genders and were not investigated in this study. However, despite these limitations, the results of this study provide information that may help to illuminate factors that influence the success in introductory programming of various populations.

According to the findings in the study, we suggest that introducing programming courses to students at the early age may help to encourage more females to enter the computer science field, because at least in the middle grades girls do not show any disadvantages relative to boys. In addition, as there is a connection between math and programming, to engage more students into computer science, we also need to confront stereotypes about gender, math, and computer science. Finally, there may be hidden barriers that could influence success in programming, such as native language. Further research is needed to investigate the role of language proficiency in learning computer programming and whether language may create difficulties for students learning computer science.

References

Alvarado, C., Dodds, Z., & Libeskind-Hadas, R. (2012). Increasing women's participation in computing at Harvey Mudd College. *ACM Inroads*, *3*(4), 55-64. http://dx.doi.org/10.1145/2381083.2381100

Alvarado, C., Lee, C. B., & Gillespie, G. (2014). New CS1 pedagogies and curriculum, the same success factors? In *Proceedings of the 45th ACM technical symposium on computer science education* (pp. 379-384). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/2538862.2538897

Ben-Ari, M. (2001). Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching*, *20*(1), 45-73. http://dx.doi.org/10.1145/274790.274308

Bennedsen, J., & Caspersen, M. E. (2005). An investigation of potential success factors for an introductory model-driven programming course. In *Proceedings of the first international workshop on computing education research* (pp. 155-163). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/1089786.1089801

Bergin, S., & Reilly, R. (2005). Programming: Factors that influence success. *ACM SIGCSE Bulletin*, *37*(1), 411-415. http://dx.doi.org/10.1145/1047124.1047480

Beyer, S., Rynes, K., Perrault, J., Hay, K., & Haller, S. (2003). Gender differences in computer science students. *ACM SIGCSE Bulletin*, *35*(1), 49-53. http://dx.doi.org/10.1145/792548.611930

Bonar, J., & Soloway, E. (1985). Preprogramming knowledge: A major source of misconceptions in novice programmers. *Human-Computer Interaction*, *1*(2). http://dx.doi.org/ 10.1207/s15327051hci0102_3

Brown, N. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The resurgence of computer science in UK schools. *ACM Transactions on Computing Education (TOCE)*, *14*(2), 1-22. http://dx.doi.org/10.1145/2602484

Bruckman, A., & Edwards, E. (1999). Should we leverage natural-language knowledge? An analysis of user errors in a natural-language-style programming language. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 207-214). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/302979.303040

Bruckman, A., Jensen, C., & DeBonte, A. (2002). Gender and programming achievement in a CSCL environment. In *Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community* (pp. 119-127). Hillsdale, NJ: Lawrence Erlbaum Associates. http://dx.doi.org/10.3115/1658616.1658634

Bureau of Labor Statistics, U.S. Department of Labor. (2014-15). *Occupational Outlook Handbook, 2014-15 Edition, Computer and Information Research Scientists*. Retrieved from http://www.bls.gov/ooh/computer-and-information-technology/computer-and-information-research-scientists.htm

Dambrot, F. H., Watkins-Malek, M. A., Silling, S. M., Marshall, R. S., & Garver, J. A. (1985). Correlates of sex differences in attitudes toward and involvement with computers. *Journal of Vocational Behavior*, *27*(1), 71-86. http://dx.doi.org/10.1016/0001-8791(85)90053-3

De-La-Fuente-Valentín, L., Pardo, A., & Kloos, C. D. (2013). Addressing drop-out and sustained effort issues with large practical groups using an automated delivery and assessment system. *Computers & Education*, *61*, 33-42. http://dx.doi.org/10.1016/j.compedu.2012.09.004

Derksen, S., & Keselman, H. J. (1992). Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of Mathematical and Statistical Psychology*, *45*(2), 265-282. http://dx.doi.org/10.1111/j.2044-8317.1992.tb00992.x

Douce, C., Livingstone, D., & Orwell, J. (2005). Automatic test-based assessment of programming: A review. *Journal on Educational Resources in Computing*, *5*(3), 1-13. http://dx.doi.org/10.1145/1163405.1163409

Du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research*, *2*(1), 57-73. http://dx.doi.org/10.2190/3LFX-9RRF-67T8-UVK9

Entwisle, D. R., Alexander, K. L., & Olson, L. S. (1994). The gender gap in math: Its possible origins in neighborhood effects. *American Sociological Review*, *59*(6), 822-838. http://dx.doi.org/10.2307/2096370

Guiso, L., Monte, F., Sapienza, P., & Zingales, L. (2008). Culture, gender, and math. *Science*, *320*(5880), 1164-1165. http://dx.doi.org/10.1126/science.1154094

Guo, P. J. (2013). Online python tutor: Embeddable web-based program visualization for CS education. In *Proceeding of the 44th ACM technical symposium on Computer science education* (pp. 579-584). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/2445196.2445368

Gürer, D., & Camp, T. (2002). An ACM-W literature review on women in computing. *ACM SIGCSE Bulletin*, *34*(2), 121-127. http://dx.doi.org/10.1145/543812.543844

Guzdial, M., Ericson, B., Mcklin, T., & Engelman, S. (2014). Georgia Computes! An intervention in a US state, with formal and informal education in a policy context. *ACM Transactions on Computing Education (TOCE)*, *14*(2), 1-29. http://dx.doi.org/10.1145/2602488

Hattie, J., & Fitzgerald, D. (1987). Sex differences in attitudes, achievement and use of computers. *Australian Journal of Education*, *31*(1), 3-26. http://dx.doi.org/10.1177/000494418703100101

Holden, E., & Weeden, E. (2003). The impact of prior experience in an information technology programming course sequence. In *Proceedings of the 4th conference on Information technology curriculum* (pp. 41-46). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/947121.947131

Hyde, J. S., Lindberg, S. M., Linn, M. C., Ellis, A. B., & Williams, C. C. (2008). Gender similarities characterize math performance. *Science*, *321*(5888), 494-495. http://dx.doi.org/10.1126/science.1160364

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys (CSUR)*, *37*(2), 83-137. http://dx.doi.org/10.1145/1089733.1089734

Kersteen, Z. A., Linn, M. C., Clancy, M., & Hardyck, C. (1988). Previous experience and the learning of computer programming: The computer helps those who help themselves. *Journal of Educational Computing Research*, *4*(3), 321-333. http://dx.doi.org/10.2190/9LE6-MBXA-JDPG-UG90

Linn, M. C. (1985). Fostering equitable consequences from computer learning environments. *Sex Roles*, *13*(3-4), 229-240. http://dx.doi.org/10.1007/BF00287913

Lockheed, M. E. (1985). Women, girls, and computers: A first look at the evidence. *Sex Roles*, *13*(3-4), 115-122. http://dx.doi.org/10.1007/BF00287904

Makrakis, V., & Sawada, T. (1996). Gender, computers and other school subjects among Japanese and Swedish students. *Computers & Education*, *26*(4), 225-231. http://dx.doi.org/10.1016/0360-1315(95)00085-2

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B. D., ... & Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ACM SIGCSE Bulletin*, *33*(4), 125-180. http://dx.doi.org/10.1145/572139.572181

Murphy, L., Richards, B., McCauley, R., Morrison, B. B., Westbrook, S., & Fossum, T. (2006). Women catch up: Gender differences in learning programming concepts. *ACM SIGCSE Bulletin*, *38*(1), 17-21. http://dx.doi.org/10.1145/1124706.1121350

Patitsas, E., Craig, M., & Easterbrook, S. (2014). A historical examination of the social factors affecting female participation in computing. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 111-116). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/2591708.2591731

Phillips, P. (Ed.). (2012). *K-8 Computer Science: Building a Solid Foundation*. Retrieved from http://www.csta.acm.org/Curriculum/sub/CurrFiles/CS_K-8_Building_a_Foundation.pdf

Rubio, M. A., Romero-Zaliz, R., Mañoso, C., & Angel, P. (2015). Closing the gender gap in an introductory programming course. *Computers & Education*, *82*, 409-420. http://dx.doi.org/10.1016/j.compedu.2014.12.003

Sax, L. J. (1994). Mathematical self-concept: How college reinforces the gender gap. *Research in Higher Education*, *35*(2), 141-166. http://dx.doi.org/10.1007/BF02496699

Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., ... Verno, A. (2011). *CSTA K-12 Computer Science Standards*. Retrieved from http://www.csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf

Simard, C., Stephenson, C., & Kosaraju, D. (2010). *Addressing Core Equity Issues in K-12 Computer Science Education: Identifying Barrier and Sharing Strategies*. The Anita Borg Institute and the Computer Science Teachers Association. Palo Alto, CA.

Sirkia, T., & Sorva, J. (2012). Exploring programming misconceptions. In *Proceedings of the 12th Koli Calling International Conference on Computing Education Research* (pp. 19-28). New York: Association for Computing Machinery. http://dx.doi.org/10.1145/2401796.2401799

Shashaani, L. (1997). Gender differences in computer attitudes and use among college students. *Journal of Educational Computing Research*, *16*(1), 37-52. http://dx.doi.org/10.2190/Y8U7-AMMA-WQUT-R512

Shashaani, L., & Khalili, A. (2001). Gender and computers: Similarities and differences in Iranian college students' attitudes toward computers. *Computers & Education*, *37*(3), 363-375. http://dx.doi.org/10.1016/S0360-1315(01)00059-8

Sullivan, A., & Bers, M. (2013). Gender differences in kindergarteners' robotics and programming achievement. *International Journal of Technology & Design Education*, *23*(3), 691-702. http://dx.doi.org/10.1007/s10798-012-9210-z

Taylor, H. G., & Mounfield, L. C. (1994). Exploration of the relationship between prior computing experience and gender on success in college computer science. *Journal of Educational Computing Research*, *11*(4), 291-306. http://dx.doi.org/10.2190/4U0A-36XP-EU5K-H4KV

Wang, T., Su, X., Ma, P., Wang, Y., & Wang, K. (2011). Ability-training-oriented automated assessment in introductory programming course. *Computers & Education*, *56*(1), 220-226. http://dx.doi.org/10.1016/j.compedu.2010.08.003

Whitley Jr, B. E. (1997). Gender differences in computer-related attitudes and behavior: A meta-analysis. *Computers in Human Behavior*, *13*(1), 1-22. http://dx.doi.org/10.1016/S0747-5632(96)00026-X

Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: A study of twelve factors. *ACM SIGCSE Bulletin*, *33*(1), 184-188. http://dx.doi.org/10.1145/366413.364581