Investigating the impact of applying natural language learning tools into the PRIMM programming framework

Cambridge Computing Education Research Symposium 2020, UK

Author: Alex Parry
alex.parry@raspberrypi.org

Background

The role of language in the field of computer science education is seen by some researchers as a particularly important aspect that is often overlooked when designing and implementing introductory programming courses (Lister et al., 2009; Portnoff, 2018).

This study investigates the relationship between learning a programming language and learning a first or second natural language, and how this may influence the way in which text-based programming is taught in schools. It was conducted in a school in England with 12-13 year old students, and builds upon the PRIMM (Predict, Run, Investigate, Modify, Make) framework. PRIMM aims to counter the problems that novices encounter as they attempt to write programs before they are able to read them (Sentance and Waite, 2017).

Method

One aim was to investigate the impact of integrating **methods for learning natural languages** on students' learning of a **programming language**. Material used in the initial large-scale PRIMM study (Sentance et al., 2019) was modified to include extra teaching tools, such as **colour coding words, cloze exercises,** and **fading worked examples**. Some of these tools were combined further, for example **fading cloze exercises**. An experimental group was taught with the new material and the control group was taught using the original resources. Each group took a baseline test and a posttest to compare progress.

The second part of this study explored **how students perceive learning a programming language compared to a natural language**. A focus group of 5 students from the experimental group explored students' views on this, which was categorised into themes.

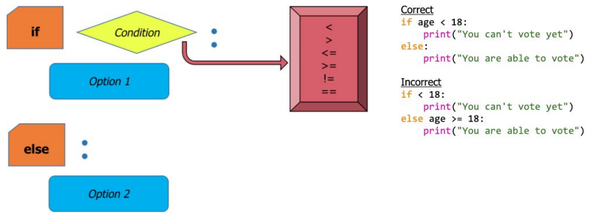

Figure 1. A colour coded `if` / `else` statement with a correct and incorrect example.

Task	Expression
Change the program so that it asks the user (the person using your program) what their name is at the beginning of the program.	`name = input("What is your name?")`
Change the program so that it lets you type in your answer on the same line as the question.	`answer = _____("Do you like programming" + name + "?")`

Ask the user what they had for breakfast and output a suitable response.	`breakfast = _____("What did you have for breakfast?")`
	
	`print("I really like " _____ + "too!")`

Ask the user what their favourite colour is and output a suitable response.	`colour _____(_____)`
	
	`_____(_____ " is great!")`

Figure 2. A series of fading cloze exercises to support code writing.

Findings

Baseline and posttest

To examine whether the results showed any significant difference between the control group and the experimental group before the intervention began, the Mann-Whitney U test was used on the data with the null hypothesis: "there is no significant difference between the scores of the experimental group and the control group in the baseline test". Table 1 displays the results, revealing that the p value (0.191) is over the common alpha risk value of 0.05. Therefore, **the performance of students on the baseline test was not significantly different between the two groups** so the null hypothesis **could not** be rejected.

	Baseline test	Posttest
Mann-Whitney U	313.500	198.000
Wilcoxon W	691.500	576.000
Z	-1.307	-3.237
Asymp. Sig. (2-tailed)	.191	.001

Table 1. Baseline test and posttest analysis.

The data from the posttests was used to analyse if there was any differences between students' learning in the intervention lessons compared to the control group once all of the lessons had ended. Table 1 shows the results of the Mann-Whitney U test against the null hypothesis: "there is no significant difference between the scores of the experimental group and the control group in the posttest". Considering that the p value (0.001) is under the alpha risk value of 0.05, this means that **there is a statistically significant difference between the two groups** and the null hypothesis **can** be rejected. Judging by the results of the Mann-Whitney U test, **the experimental group achieved a significantly higher score on the posttest than the control group**.

Focus group

Four themes emerged whilst coding and categorising the comments from the transcription: programming and natural language similarities, range and scope of vocabulary, usefulness of the material, and problems with syntax. Examples of these comments are shown in Table 2. Analysis of the focus group conveyed that most students made similar connections between both types of languages, however they were not always considered to be on the same level of difficulty. Some students thought programming was more challenging due to the syntax, whilst others found it easier because of the limited, yet more versatile vocabulary.

Theme	Example
Programming and natural language similarities	I think that's why I like coding because I like learning languages. (Natty)
Range and scope of vocabulary	...the reason why I find programming so interesting is that I find that so many different types of commands are so versatile and just really convenient because you don't have to learn a bunch of things rather than in other languages (Sam)
Usefulness of the material	Something that was useful and helped us understand... is that fact that we have the structure of the sentences... and it showed how you could go and change some of the words so it made it easier to code and easier to understand. (Sam)
Problems with syntax	I feel like what part of it is confusing for me - it's having all these different symbols…so you have just random letters in the middle of symbols and it's a bit weird. It's confusing. (Amy)

Table 2. Themes and example comments from the focus group.

Conclusions

The results of the two tests found that the performance of students on the baseline test was not significantly different between the two groups, yet the experimental group achieved a significantly higher score on the posttest than the control group. This indicates that the inclusion of language tools within the PRIMM materials made a positive impact on students' learning.

Some of the student in the focus group also commented on these tools, saying that they found the colour coded program statements helpful for understanding the grammatical structure of a new concept and that the fading cloze exercises were useful in the later stages of writing code.

Findings from the focus group indicate that a positive perspective of foreign or native languages may benefit students' belief in their ability to learn a text-based programming language, whilst negative preconceptions can discourage students. This corresponds with the view that to be good at programming, you must be highly skilled in your native language (Dijkstra, 1975).

One of the areas that was highlighted as a determinant regarding the self-efficacy of learning to program was encountering syntax errors. Further research is required to understand how natural language skills translate to learning a text-based programming language. This could include using more tools from natural languages that focus on supporting novices with vocabulary and syntax.

References:

- Dijkstra, E. W. (1975) *How do we tell truths that might hurt?* ACM SIGPLAN Notices. 17 (5), 13–15.
- Lister, R. et al. (2009) *'Further evidence of a relationship between explaining, tracing and writing skills in introductory programming'*, in Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education. ITiCSE '09. 2009 New York, NY, USA: ACM. pp. 161–165.
- Portnoff, S. R. (2018) *The introductory computer programming course is first and foremost a language course.* ACM Inroads. 9 (2), 34–52.
- Sentance, S. et al. (2019) *'Teachers' Experiences of using PRIMM to Teach Programming in School'*, in Proceedings of the 50th ACM Technical Symposium on Computer Science Education. SIGCSE '19. 2019 New York, NY, USA: ACM. pp. 476–482.
- Sentance, S. & Waite, J. (2017) *'PRIMM: Exploring Pedagogical Approaches for Teaching Text-based Programming in School'*, in Proceedings of the 12th Workshop on Primary and Secondary Computing Education. WiPSCE '17. 2017 New York, NY, USA: ACM. pp. 113–114.