THE PRIMM METHOD FOR TEACHING PROGRAMMING: EXPERIMENTATION AND VALIDATION

G. Peserico[1], M. Serafini[1], F. Voltolini[1], F. Picasso[2], D. Agostini[2], F. Fiore[2], A. Serbati[2], A. Montresor[2]

[1]Liceo Scientifico "Leonardo Da Vinci" of Trento (ITALY)
[2]University of Trento (ITALY)

Abstract

The project "Algorithmically: from problem-solving to computer science" is a research-action project funded and carried out in the academic year 2022-23 by the Scientific High School “Leonardo da Vinci” (Trento, Italy), in collaboration with the University of Trento. The goal of the project was to validate the PRIMM (Predict, Run, Investigate, Modify, Make) methodology for teaching programming, which subverts the traditional approach based on early program writing. Instead, students are encouraged to read and understand segments of code before writing their own, emphasising a "reading before writing" approach. The teaching experimentation was applied in 5 second classes of the high school, with 4 control group classrooms (190 students and 9 classes). To understand the effectiveness of this new methodology, the design was a Pre-Test and Post-Test, with a test aiming at understanding skills and knowledge and another based on the Intrinsic Motivation Inventory (IMI). The first iteration of this experimental teaching approach provides promising results but also indicates the need for refining the sample and some methods and procedures. However, there are significant findings:

1 PRIMM methodology proved to be slightly more effective than traditional teaching methods in terms of learning.

2 PRIMM methodology promotes Competence Perception, and its relationship with Competence Perception is confirmed, positive, and significant.

3 Competence Perception correlates with better computer science grades, so PRIMM might bring higher marks in Computer Sciences.

The PRIMM approach’s integration into computer science teaching thus appears promising, but additional research and iterations are needed to optimise its effectiveness.

Keywords: Stem education, teaching, learning, coding, computer science, programming education, secondary education.

1 INTRODUCTION

The project "Algorithmically: from problem-solving to computer science" carried out an action research activity in the academic year 2022-23 with the aim of experimenting with the PRIMM methodology [1,2] for teaching programming in the IT discipline of the "Applied Sciences" option of the Scientific High School. The project was proposed by the “Leonardo da Vinci” High School of Trento, in collaboration with the University of Trento and the Glow Cultural Association, and was financed by the Caritro Foundation.

The basic idea of the PRIMM methodology (Predict, Run, Investigate, Modify, Make) starts from the observation that "traditional" programming teaching requires the student to write code right from the start. This differs from the methodologies used in language learning (including mother tongue), where the production activity is preceded by reading and understanding the text.

Starting from the observation that the cognitive load necessary to write code is very high, the PRIMM methodology reverses the traditional sequence, starting first from understanding the text and only subsequently arriving at writing real code.

In particular, the proposed approach is divided into five activities:

- **Predict**: Students read a segment of code written in Python and make predictions about what the code will do when it is executed;

- **Run**: students execute the code proposed in the previous activity, comparing the actual behaviour with the prediction;
- **Investigate**: students are asked to analyse the code or its variants in more depth, using various types of exercises, such as bug fixes, code annotation, the use of Parson Puzzles, exploratory questions, etc;
- **Modify**: students are asked to modify the code, starting from very simple variations and then with increasingly complex modifications;
- **Make**: Finally, students create a completely new program inspired by the code seen previously but create new functionality or solve a different problem.

PRIMM can be viewed as sitting at the crossroads between a structured approach, which relies on guided discovery and direct instruction, and a constructionist approach [3], which emphasises pure discovery and open-ended challenges. Grover et al. recommend a five-phase process that progresses from highly guided activities to completely autonomous ones, with the level of scaffolding being gradually decreased [4].

As far as possible, many of the activities are carried out encouraging group work (in pairs) and exchange between peers, following the idea that the linguistic expression of solutions generally promotes learning, particularly in the field of programming where the linguistic component is fundamental.

In the rest of the article, we will describe the context in which we operated, the activities carried out and the results of the experimentation. We will conclude with a reflection on the validity of the approach and possible improvements to be applied in subsequent years.

2 METHODOLOGY

This study was conceived to understand the feasibility and effectiveness of the PRIMM methodology for teaching programming at the high school level. In an action-research approach, the idea is to refine the method and reiterate the experimentation based on the feedback gathered from all the participants.

2.1 Context

At "Leonardo Da Vinci" Scientific High School, there are 75 classes comprising a total of 1,530 students. Among these, 42 classes are dedicated to applied sciences, while 33 classes cover the regular curriculum, with an average of 20 students per class. In the Applied Science stream, computer science is taught for two hours per week from first to fifth grade. Starting from the second grade, students delve into actual programming using the Python language.

The teaching experiment was applied in five second-year high school classes (102 students, see Table 1), out of a total of nine, under the guidance of teachers Giulia Peserico, Maria Serafini and Francesca Voltolini. In the other four second-year classes (86 students), which act as control classes, traditional teaching methodologies were adopted. Table 1 contains the list of involved and control classes, the responsible teacher, and the number of students with gender breakdown.

For four of the classes involved, each activity lasted 100 minutes, while, for the fifth class, the activities took place in two sessions of 50 minutes each.

Table 1. List of classes involved, with teachers and size.

Class Name	*Teacher*	*Hours*	*No. of Students*	*Male students*	*Female students*
Primm1	Teacher1	100’	23	13	10
Primm2	Teacher1	100’	20	8	12
Primm3	Teacher1	100’	25	14	11
Primm4	Teacher2	100’	19	15	4
Primm5	Teacher3	50’+50’	18	13	5
Controllo1	Teacher4	50’+50’	21	16	5
Controllo2	Teacher4	50’+50’	20	11	9
Controllo3	Docente5	50’+50’	21	13	8
Controllo4	Teacher6	50’+50’	22	13	9

2.2 PRIMM Application

For the five classes involved in the experimentation, teaching cards were created to introduce the basic concepts of programming in the Python language (variables, types, Boolean operators, selection, iteration, lists, strings, and use of the Turtle library).

Most of the proposed activities were structured in pair work and included:

- Three or four *Predict* proposals*,* which proposed a new concept whose functioning was asked to be predicted, each followed by a phase *Run,* in which students copied and ran the code to verify their prediction;
- some exploratory questions, in which they were asked to explain in words what they understood about the new concept, also comparing it with previously learned concepts;
- a code within which to find some errors (syntax, run, semantics);
- a Parson's Puzzle;
- some code editing exercises (*Modify*);
- some code writing exercises (*Make*).

All classes were conducted in the computer lab, where each student was provided with a PC equipped with the Thonny interpreter [5]. At the end of each session, homework was assigned to reinforce the learned concepts, and *Modify* and *Make* activities were completed. As the course progressed, the *Predict-Run* part was reduced, Parson's Puzzles were eliminated, and *Modify* and *Make* activities were emphasised during partner exercises. Additionally, a brief introduction to new concepts was included at the start of each lesson, followed by a group review session. This was implemented as it was observed that students were not always paying sufficient attention when studying at home. Throughout the year, two types of evaluations were conducted: structured assessments featuring multiple-choice and open-ended questions similar to the *Predict* activities and error detection, and practical assessments requiring students to write 3-4 programs to solve various problems.

2.3 Instruments and Data Analysis

To verify the effectiveness of the PRIMM method, three types of data were collected on the five experimental classes and the four control classes:

- The first type of data concerns the student's motivation to carry out this type of activity. The tool used to collect this data was the Intrinsic Motivation Inventory (IMI) [6], translated and adapted, using only the modules that were deemed useful for this experimentation, namely: Interest/ satisfaction, Effort/importance, Perception of competence, Value/Utility, Pressure/Tension.
- The second type of data concerns the learning of knowledge and skills regarding programming and was detected through a carefully prepared test.
- The third type of data includes the class to which one belongs, gender and final grades in the various school subjects.

The first two types of data were collected according to a Pre-Post quasi-experimental design, that is, they were collected before the start of the experiment and after the end to detect any effects and differences between the experimental and control groups.

The Intrinsic Motivation Inventory (IMI) is a multidimensional measurement tool widely employed to assess participants' subjective experiences related to intrinsic motivation and self-regulation within experimental settings. It is broadly utilised across various domains, including education, to assess inherent motivational factors within individuals in relation to specific activities or tasks.

The IMI consists of several dimensions, each providing insight into different aspects of intrinsic motivation, contributing to a comprehensive understanding of participants' motivational states, but in this study, only the relevant ones were selected:

1 Interest/Enjoyment: this is considered the self-reported level of interest and enjoyment, often regarded as the most direct measure of intrinsic motivation. It focuses on the inherent satisfaction and pleasure derived from performing the activity, independent of external reinforcements or rewards.

2 Perceived Competence: this dimension reflects the individual's feelings of efficacy and competence related to the activity. It corresponds to the extent to which a participant feels capable and effective in performing the task.

3 Pressure/Tension: this dimension is indicative of the degree to which individuals experience stress, pressure, or tension in relation to the task. Lower levels of pressure and tension are usually associated with higher levels of intrinsic motivation.

4 Value/Usefulness: some versions of the IMI include this dimension to gauge the extent to which participants perceive the activity as valuable or useful, influencing the level of intrinsic motivation towards the task.

5 Effort/Importance: this dimension evaluates the amount of effort that participants are willing to invest and how important they perceive the activity, providing additional context to their motivational state.

Each of these dimensions is assessed through a series of statements related to the activity, to which respondents indicate their level of agreement on a Likert scale. The composite results of the IMI provide researchers and practitioners with insights into the multifaceted nature of intrinsic motivation, facilitating nuanced understanding and informed intervention development aimed at enhancing motivation and engagement.

The data analysis was initiated by normalising and cleaning the dataset. Also, variables were created to facilitate data processing, such as assigning a class group variable to each student and creating "_Diff" variables representing the difference between pre and post-test scores, as well as IMI questionnaire results.

From the descriptive statistics, it was apparent that one control group had inconsistent data, with the lowest pre-test scores and the highest post-test scores, which did not correspond to the final computer science evaluations by the teacher. Hence, it was removed from the data, unfortunately reducing the control elements for analysis and causing the sample to be slightly imbalanced. In the "Experimental" group, there were 100 students, while the control group contained 56 students (due to the group we had to eliminate). Additionally, there were 97 students of 'M' gender and 59 of 'F' gender. This imbalance could influence statistical analyses.

The IMI test's internal consistency coefficient, Cronbach's alpha, was calculated for IMI1 (Pre) and IMI2 (Post). For IMI1, Cronbach's alpha was 1.03, while for IMI2 it was 1.08. This indicates excellent consistency.

A linear regression analysis with robust standard errors was used to take into account the heterogeneity of the groups

3 RESULTS

The results of the analysis, for which a linear regression with robust standard errors was used to take into account the heterogeneity of the groups, highlight the following results:

- There is no statistically significant association between "belonging to the experimental group" and an "increase in test scores" ($p=0.455$). Despite this, the increases in test scores of the experimental group are slightly higher than those of the control group.
- There is a statistically significant association between "belonging to the experimental group" and an "increase in the IMI Effort/Importance" of the activity score ($p=0.047$), with the experimental group membership being a significant predictor. This suggests that the PRIMM methodology could enhance student engagement in activities and their perceptions of the activities' importance.
- There is a highly significant association between "belonging to the experimental group" and an increase in the IMI score regarding the perception of competence ($p<0.001$), with the experimental group membership being a significant predictor. The PRIMM methodology could therefore favor students' perception of their own competence and confidence in programming tasks. Students may also be able to self-assess more accurately.
- Students with the greatest improvement in their final test scores, compared to their initial test scores, also achieved higher grades in computer science.
- Belonging to the experimental group and, therefore, the PRIMM methodology, is associated with a positive impact on computer science grades. The result of the analysis comes very close to significance ($p=0.051$), with the experimental group membership being a quasi-significant predictor.

- The gender difference in outcomes was also explored. Although there are no conclusive results, the analysis detects, in the control group, an imbalance in interest in activities on the part of male students compared to female students (p=0.08). This indicates that the PRIMM methodology could be more inclusive than the traditional one.

4 CONCLUSIONS

The initial phase of this action research yielded encouraging outcomes, albeit highlighting the requirement to fine-tune the sample and certain methodologies and procedures. To elaborate, certain examination items necessitate further testing as they do not produce consistent results alongside other indicators. Moreover, multiple exercises were conducted with groups by students' proficiency levels, which could have impacted the fact that pupils who exhibited greater progress in test scores also attained superior grades in computer science.

Moving forward, the groups will be strategically divided to facilitate peer tutoring [7]. Overall, the implementation of the PRIMM method shows great promise and has already demonstrated a significant and positive effect on students' programming approach. Specifically, it has boosted their sense of competence, commitment, and perceived importance of the activities. However, in order to optimise its effectiveness, more research and replication is necessary.

At the end of the year, we asked the students for comments on the way they worked with the PRIMM methodology. A large group of students complained about the lack of usual face-to-face lessons with an explanation from the teacher before tackling laboratory activities; we can hypothesise that this depends on an (Italian) school path that is very focused on frontal lessons, so that students are not familiar with innovative approaches. Furthermore, some students stated that working in same-level pairs was not productive, and they would have rather been in pairs with companions who would help them more in their work.

A significant number of students have expressed their satisfaction with the PRIMM methodology, which has enabled them to work at their own pace. They complete the assigned tasks independently, without the pressure of a workgroup, and later, have the flexibility to elaborate on them at home. Overall, the positive feedback from students indicates that the PRIMM approach will be recommended again in the future.

ACKNOWLEDGEMENTS

Gratitude is extended to the students and teachers of Liceo “Da Vinci” in Trento for their invaluable participation in this research endeavour. Appreciation is also directed to the school's director for their support and to the CARITRO Foundation for their generous funding of this project.

REFERENCES

[1] S. Sentance, J. Waite, and M. Kallia, “Teachers’ experiences of using PRIMM to teach programming in school,” in *The 50th ACM Technical Symposium on Computing Science Education: SIGCSE 2019*, Minnesota, 2019.

[2] S. Sentance, J. Waite, and M. Kallia, "Teaching computer programming with PRIMM: a sociocultural perspective," *Computer Science Education*, vol. 29, no. 2-3, pp. 136-176, 2019.

[3] S. Papert, *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books, Inc, 1980.

[4] S. Grover, R. Pea, and S. Cooper, “Designing for deeper learning in a blended computer science course for middle school students,” *Computer Science Education*, vol. 25, no. 2, pp. 199–237, 2015.

[5] A. Annamaa, “Thonny: A Python IDE for learning programming,” in *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE ’15*, pp. 343-, 2015.

[6] E. McAuley, T. Duncan, and V. V. Tammen, “Psychometric properties of the Intrinsic Motivation Inventory in a competitive sport setting: A confirmatory factor analysis,” *Research Quarterly for Exercise and Sport*, vol. 60, no. 1, pp. 48-58, 1989.

[7] F. J. Alegre Ansuategui and L. Moliner Miravet, “Emotional and cognitive effects of peer tutoring among secondary school mathematics students,” *International Journal of Mathematical Education in Science and Technology*, vol. 48, no. 8, pp. 1185–1205, 2017.