Relationship between SPM Results and C++ Programming Achievement among First-Year Students

Yahya Ibrahim., Nuridawati Mustafa., Kurk Wei Yi., Massila Kamalrudin

Fakulti Teknologi Maklumat dan Komunikasi Universiti Teknikal Malaysia Melaka (UTeM), Durian Tunggal, Melaka, 76100, Malaysia

DOI: https://doi.org/10.47772/IJRISS.2026.101000308

Received: 21 October 2026; Accepted: 26 October 2026; Published: 02 November 2026

ABSTRACT

This study examines whether prior academic achievement in Sijil Pelajaran Malaysia (SPM) is associated with performance in an introductory C++ programming course. Introductory programming is often difficult for new learners, and lecturers frequently rely on SPM results to anticipate which students may struggle, yet subject-level evidence from Malaysian cohorts remains limited. Data from 35 students in one class group were analysed, comprising SPM grades in Additional Mathematics, Mathematics, Physics, Chemistry, Computer Science and English, together with final examination marks Part A (program understanding) with 60%; Part B (problem solving) with Question 1, 15% and Question 2, 25% and the final course grade. SPM grades were converted to an ordinal scale, and descriptive statistics, Spearman rank correlations, Kruskal–Wallis and Mann–Whitney tests and linear regression were applied. The mean final examination mark was 74.9 out of 100 (SD = 11.9), and 57.1% of students obtained grade A. Additional Mathematics was the only individual SPM subject significantly correlated with the final examination mark ($\rho = .37$, $p = .033$); Mathematics, Physics, Chemistry, Computer Science and English were not, partly because of limited variation in grades and small numbers. A composite SPM score explained 17.4% of the variance in the examination mark, although the rank-based association was not significant. Students with SPM Computer Science did not differ significantly from those without it. Part B Question 2, the problem-solving task, was the most difficult component, with a mean of 59.9% of the maximum mark. The findings suggest that SPM results are modest predictors of C++ achievement, with Additional Mathematics the most informative, and should not be used alone to identify at-risk students**.** The small, single-group sample limits generalisation, and the results are exploratory.

Keywords: C++ programming, SPM, academic achievement, Additional Mathematics, introductory programming, student performance, correlation analysis

INTRODUCTION

Introductory programming remains a persistent challenge in higher education, with CS1 often functioning as a bottleneck course and failure rates commonly exceeding 30% [1], [2], [3]. This difficulty has intensified as programming is now taught not only to computer science majors but also to students from engineering and other disciplines, making it increasingly important to understand which students are likely to struggle and why [4], [5], [6]. Recent research shows that success in introductory programming is shaped by a broad set of interacting factors, including prior academic preparation, prior programming exposure, cognitive skills, motivation, self-efficacy, and learning behaviours, rather than by any single characteristic alone [7], [8].

Among background variables, mathematics preparation appears repeatedly as a useful but incomplete predictor of programming performance [9], [10]. Studies from Botswana, the United States, and Southeast Asia report that weaker pre-college mathematics preparation is associated with poorer CS1 outcomes, while stronger mathematics competency tends to align with better programming performance [10], [11]. In the Malaysian context, subject-level evidence remains limited, but available findings suggest that school results offer only modest predictive value, with Additional Mathematics carrying more signal than most other SPM subjects. Earlier Malaysian evidence also indicates that good Mathematics and English results, together with prior

exposure to programming, can be linked to stronger programming grades, although those findings come from a different cohort and analytic approach [12].

At the same time, recent work suggests that prior programming experience is not a uniformly reliable predictor of success [4], [13]. Some studies report a persistent grade advantage for students with prior experience, while others find that prior exposure raises confidence and expectations without clearly improving final grades [14], [15]. Non-cognitive factors appear at least as important: self-efficacy is repeatedly associated with better performance, perceived difficulty and fear of failure predict poorer outcomes, and lower self-efficacy is linked to increased risk of failing or withdrawing [5], [16]. Broader reviews also note that the field still lacks consensus on a definitive predictor set and that many studies have focused more on student background than on the learning processes through which success or failure develops [6], [8].

Because of this, current research increasingly favours early, staged prediction rather than reliance on school results alone [4]. Models built only from background variables can identify risk moderately well, but prediction improves substantially when early course data such as first assessments, submission timing, and number of attempts are added [17], [18], [19]. Early-semester performance, especially on the first major assessment, appears to be one of the strongest signals of later failure, supporting the use of early-warning systems and timely intervention in CS1 [17], [18], [19]. Against this background, the present study examines whether prior academic achievement, particularly SPM subject performance, is associated with achievement in an introductory C++ course, while recognizing that school results are likely to provide only one part of a broader explanation of programming success [18].

LITERATURE REVIEW

Recent CS1 research supports the claim that introductory programming performance is multifactorial, with modest contributions from background variables and stronger prediction once self-beliefs and early course behaviour are included [8], [18]. The evidence also shows that no single preparation measure is sufficient across contexts, so the most defensible literature review emphasizes interacting academic, psychological, and behavioural factors rather than one dominant cause [2], [8], [19].

Mathematics preparation is one of the most consistent background correlates of CS1 performance, but the effect is usually context dependent rather than determinative [9], [10], [20]. In Botswana, high school mathematics correlated significantly with introductory programming performance, while high school computer studies did not [9]. A U.S. study found that students ready for Calculus I were more likely to pass CS1, suggesting that a stronger math prerequisite may improve pass rates [21].

More recent work complicates a simple math-only account. In one aptitude-testing study, prior programming experience and previous computer science study were not significant predictors, whereas post-school math study remained useful in the model [20]. Another study found that complex problem-solving explained 64% of programming-performance variance and did not confirm a direct effect of high school mathematics, suggesting that reasoning ability may mediate part of the math signal [13].

Evidence on prior programming experience is mixed because its effects differ across grades, confidence, and retention. Some studies report a clear grade advantage: students with prior experience outscored peers by more than 6% on exams and 10% on programming quizzes in CS1, although that gap narrowed by CS2 [22]. A newer study similarly found a persistent one-letter-grade gap between students with no prior experience and those with any experience across CS1 and CS2 [13].

Other studies find that prior experience improves self-efficacy and expectations more reliably than final grades. In Finland, prior experience raised initial self-efficacy and grade expectations and reduced time on tasks but did not improve grades [14]. Recent path-analysis work also suggests that prior experience is the dominant factor shaping self-efficacy, while gender has a negligible direct effect in that relationship [23].

Psychological variables appear to be at least as important as background factors because they predict both performance and persistence. Older factor-based work identified comfort level and math background as

positive predictors, while attributing success to luck predicted worse outcomes [24]. In an objects-first CS1, student effort and comfort level were stronger predictors than SAT scores or critical-thinking measures [25].

Negative self-assessment also matters. Across three universities, students frequently judged themselves harshly during normal programming moments such as planning or encountering syntax errors, and more frequent negative self-assessment was associated with lower self-efficacy [26].

Recent predictive work shows that early course behaviour improves risk prediction beyond background-only models. In a cohort of 994 students, pre-university variables alone gave acceptable discrimination, but adding the first major exam raised AUC from about 0.77 to about 0.92 [18]. Background-only models can still be useful, but they are weaker and vary by specialization, with one Chilean study reaching 68% accuracy and finding that important predictors changed across programs [8].

Behavioural trace data are especially informative early in the semester. Week-3 models using grades, delivery time, and attempts reached an F1 of 86% [17]. Submission-only frameworks have also predicted pass/fail accurately early across different course structures, and reproduction studies show that timely submission and iterative problem solving are robust signals of persistence [19], [27].

TABLE I Representative Findings Across Preparation, Experience, Beliefs, And Behaviour

Study Focus	Main Finding	Why It Matters
Math preparedness	Calculus-ready students passed CS1 more often [21].	Readiness matters before CS1
Prior experience	Experience improved CS1 scores, but the gap narrowed by CS2 [22].	Early advantage is not fixed
Self-beliefs	Negative self-assessment tracked lower self-efficacy [26].	Affective processes shape persistence
Early analytics	Week-3 behavioural models predicted performance well [17]	Intervention can start early
Submission metadata	Early pass/fail prediction worked across course variants [27].	Models can generalize across settings

CS1 performance is therefore best framed as the product of preparation, self-beliefs, and early engagement rather than any single background trait. A strong literature review should end by arguing that mathematics and prior experience provide useful but incomplete signals, while the most actionable evidence comes from combining them with psychological measures and early-semester behaviour.

METHODOLOGY

Participants and Data

The data came from one class group of 35 first-year students enrolled in a C++ programming course. For each student, the records contained SPM grades in six subjects, final examination marks and the final course grade. Student names and matriculation numbers were removed before analysis, and results are reported only in aggregate. Not every student sat every SPM subject: Additional Mathematics was recorded for 34 students, Physics and Chemistry for 32, and Computer Science for 12. Entries marked N/A were treated as missing and each analysis used all available cases.

Measures

The final examination was marked out of 100 and consisted of Part A (program understanding) with 60 marks, Part B (problem solving) with Question 1 (15 marks) and Question 2 (25 marks). The total of these three components equalled the recorded examination mark for all 35 students. The recorded final course grade (A, A−, B+ or B) was taken as the overall course outcome; because the data do not state how it was derived, it may include components other than the final examination. SPM grades were converted to an ordinal scale: A+ = 9, A = 8, A− = 7, B+ = 6, B = 5, C+ = 4, C = 3, D = 2 and E = 1. A composite SPM score was computed as the mean of the available grades in Additional Mathematics, Mathematics, Physics, Chemistry and English. Computer Science was excluded from the composite because most students did not take it.

Analysis

Because SPM grades are ordinal and several subjects show little variation, Spearman's rank correlation (ρ) was the main association measure. Interpretation followed the conventional benchmarks of .10, .30 and .50 for small, medium and large effects [7]. Differences among final grade groups were tested with the Kruskal–Wallis test and differences between two groups with the Mann–Whitney U test and Welch's t-test. Linear regression was used to estimate how much variance in the examination mark the SPM results explained. A significance level of .05 was used throughout. No correction was made for multiple comparisons, so individual findings should be treated as exploratory.

RESULTS

A. Performance in the C++ Final Examination

Table I summarises the examination results. The mean total mark was 74.9 (SD = 11.9, range 55–98), and the distribution did not depart significantly from normality (Shapiro–Wilk W = .960, p = .234). Students scored highest on Part B Question 1 (95.2% of the maximum), followed by Part A (76.1%) and Part B Question 2 (59.9%). Question 2 also had by far the widest spread (SD = 8.3 on a 25-mark scale), and four students scored zero on it. The data cannot show whether these zeros reflect non-attempts or an inability to produce a working solution.

Table II Descriptive Statistics of the C++ Final Examination

Component	Mean	SD	Range	% of max
Part A (60)	45.69	7.92	28–58	76.1
Part B Q1 (15)	14.29	1.20	10–15	95.2
Part B Q2 (25)	14.97	8.30	0–25	59.9
Total (100)	74.94	11.94	55–98	74.9

Figure 1 shows the final course grades: 20 students (57.1%) obtained A, 8 (22.9%) A−, 4 (11.4%) B+ and 3 (8.6%) B, and none obtained a grade below B. Mean examination marks decreased steadily across grades (A = 81.5, A− = 69.9, B+ = 64.8, B = 58.3), and the difference among the four groups was significant (Kruskal–Wallis H(3) = 15.65, p = .001). The examination mark and the final grade were strongly correlated (ρ = .68, p < .001), which supports using the examination mark as the main outcome.

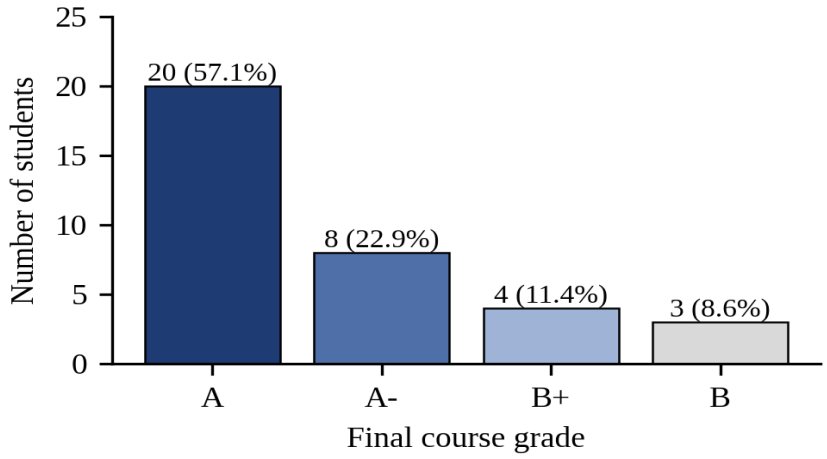


Fig. 1 Distribution of final C++ course grades (N = 35)

B. SPM Profile of the Students

Table II shows that the group was strong in SPM Mathematics, where all 35 students obtained A or A+, and in English, where 33 of 35 obtained A− or better. Grades were more varied in Additional Mathematics, Physics and Chemistry, where most students obtained B+ or lower. The lack of variation in Mathematics and English limits how far these subjects can discriminate between students.

Table III SPM Grade Distribution by Subject

Subject	**n**	**A+**	**A**	**A−**	**B+ or lower**
Add. Maths	34	1	6	5	22
Mathematics	35	17	18	0	0
Physics	32	1	5	9	17
Chemistry	32	2	2	6	22
Comp. Science	12	1	7	3	1
English	35	11	18	4	2

C. SPM Subjects and Examination Performance

Table III reports the correlation between each SPM subject and both the final examination mark and the final grade. Additional Mathematics was the only subject with a statistically significant correlation with the examination mark ($\rho = .37$, $p = .033$), a small-to-medium effect, and it was also significantly correlated with the final grade ($\rho = .38$, $p = .027$). Physics showed a weaker positive association that did not reach significance. Mathematics, Chemistry, Computer Science and English showed no significant association; the English coefficient was close to zero and slightly negative (Fig. 2). The Computer Science results rest on only 12 students and should be read with caution.

Table IV Spearman Correlations Between SPM Grades and C++ Outcomes

Subject	**n**	**ρ (exam)**	**p**	**ρ (grade)**	**p**
Add. Maths	34	.37*	.033	.38*	.027

Mathematics	35	.15	.389	.18	.309
Physics	32	.27	.136	.31	.082
Chemistry	32	.15	.428	.07	.716
Comp. Science	12	.24	.450	.42	.179
English	35	−.11	.538	.04	.830

** p < .05. "Exam" is the final examination mark out of 100; "grade" is the final course grade.*

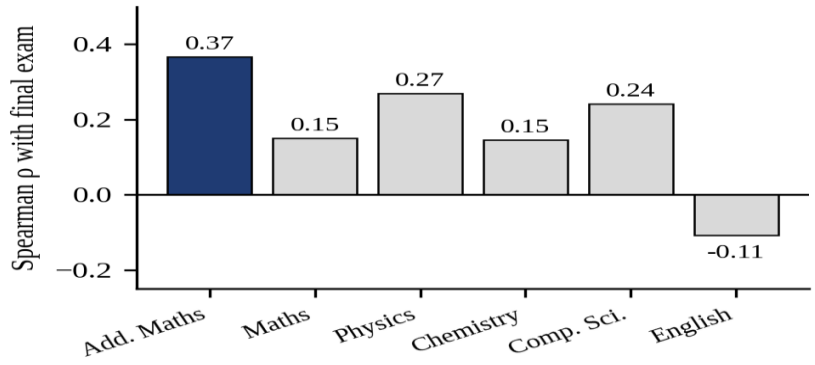


Fig. 2 Spearman correlation between each SPM subject and the final examination mark (dark bars: $p < .05$)

At component level, the clearest result was a negative correlation between Chemistry and Part B Question 1 ($\rho = -.41$, $p = .019$). Because that question had a near-ceiling mean of 14.3 out of 15 and limited variation, and because many correlations were tested without correction, this result is more likely to reflect chance or a ceiling effect than a real relationship and is not interpreted further.

D. Regression Analysis

A simple regression of the examination mark on the composite SPM score was significant ($b = 6.00$, $SE = 2.27$, $R^2 = .174$, $p = .013$, $n = 35$); each one-step increase in the average SPM grade was associated with about six additional marks (Fig. 3). The same relationship was weaker on a rank basis (Spearman $\rho = .28$, $p = .103$), so the evidence is suggestive rather than firm. A regression on Additional Mathematics alone gave $b = 2.92$ marks per grade step ($R^2 = .154$, $p = .022$, $n = 34$). When Additional Mathematics, Mathematics and English were entered together, the model was not significant overall ($F(3, 30) = 2.20$, $p = .109$, adjusted $R^2 = .098$), and only Additional Mathematics retained a significant coefficient ($b = 2.83$, $p = .033$). Taken together, SPM results explained at most about one-sixth of the variation in examination marks.

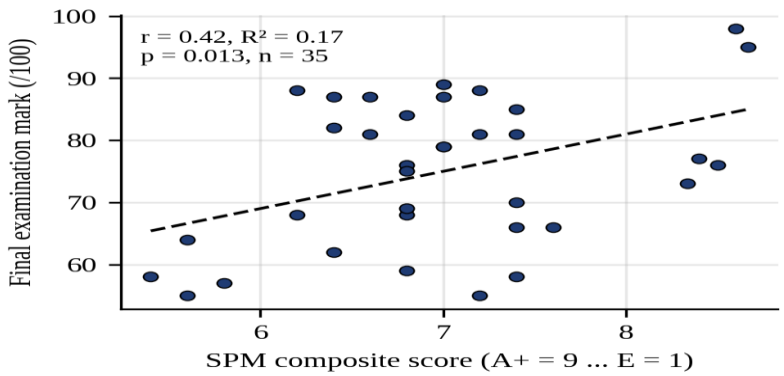


Fig. 3 Final examination mark against the SPM composite score, with fitted regression line

E. SPM Computer Science

The 12 students with SPM Computer Science had a higher mean examination mark (M = 77.2, SD = 12.0) than the 23 students without it (M = 73.8, SD = 12.0), but the difference was not significant (Welch $t(22.4) = 0.79$, $p = .437$; Mann–Whitney $p = .434$). Grade A was obtained by 66.7% of the first group and 52.2% of the second. With such small groups the study has little power to detect a modest advantage.

DISCUSSION

The main finding is that Additional Mathematics is the one SPM subject that shows a reliable association with C++ performance in this group, while the composite SPM score explains only a limited share of the variance. This is in line with studies that point to mathematical background as one of several relevant factors [24], [28], [29], and a plausible reason is that Additional Mathematics emphasises multi-step reasoning, functions and algebraic manipulation, which resemble the logic of writing and tracing programs. The absence of a Mathematics effect should not be read as evidence that it is unimportant: every student obtained A or A+, so there was almost no variation to detect. The same applies, to a lesser extent, to English.

The component results suggest where students struggle. Part B Question 1 was answered well, whereas Question 2 had a much lower mean and a widespread, which suggests that tasks requiring students to design and write their own solution are harder than more guided tasks. This is consistent with the general view that programming difficulty lies in problem solving as much as in syntax [2]. Instructors may wish to give more practice on open-ended problems, particularly to students with weaker Additional Mathematics grades.

Because the associations are modest, SPM results alone should not be used to label students as likely to succeed or fail in C++. At most they can serve as one indicator alongside early coursework, attendance, and self-reported prior programming experience [2], [29].

CONCLUSION

In this group of 35 first-year students, SPM Additional Mathematics was the only individual subject significantly associated with performance in the C++ final examination ($p = .37$, $p = .033$). The composite SPM score explained approximately 17.4% of the variance in examination marks, although its rank-based association was not statistically significant. Mathematics, Physics, Chemistry, Computer Science and English did not show statistically significant associations with the examination outcome. These findings suggest that prior school achievement may provide some indication of readiness for introductory programming, but its predictive value is limited and varies across subjects. The results should therefore not be interpreted as evidence that SPM grades alone can reliably identify students who will succeed or struggle in C++.

The examination component results also highlight the importance of problem-solving practice in introductory programming. Part B Question 2, which required students to develop a solution, had the lowest mean percentage score (59.9% of the maximum) and the widest spread of marks. This pattern suggests that independently planning and constructing a solution may be more challenging than completing a more guided task. Instructors may therefore consider providing additional opportunities for open-ended problem solving, algorithm design and code development, while using students' early coursework and assessment performance alongside their academic background to identify those who may benefit from timely support.

These findings should be interpreted in light of several limitations. The analysis involved only 35 students from one class group at a single institution, and all students obtained a final course grade of B or above. This restricted range may have weakened the observed associations and limits the generalisability of the results to other student groups and institutions. Missing SPM records reduced the number of cases available for some subject-level analyses, particularly Computer Science, which was taken by only 12 students. In addition, converting SPM grades to an ordinal numerical scale and constructing a composite score involved analytical assumptions. Motivation, study time, prior programming experience, attendance and teaching-related factors were not measured, while multiple statistical tests were conducted without adjustment. Finally, the

correlational design does not support causal conclusions. The results should consequently be regarded as exploratory evidence from this cohort rather than a definitive model of programming achievement.

Future research should examine larger samples drawn from multiple classes and institutions to determine whether the observed relationship between Additional Mathematics and C++ achievement is replicated across different student populations. Further studies should also include measures of prior coding experience, motivation, programming self-efficacy, study behaviour and early-semester assessment performance to evaluate whether these variables improve prediction beyond SPM results. Analyses that account directly for the ordinal nature of school grades, address missing data and control for multiple comparisons would strengthen the robustness of the findings. Longitudinal research could additionally investigate whether early indicators of difficulty, particularly performance on problem-solving tasks, can support timely interventions. Overall, SPM results, especially Additional Mathematics may be useful as one source of information, but they should be combined with evidence from students' ongoing learning and performance when planning support for introductory programming.

ACKNOWLEDGMENT

The authors would like to express gratitude to Fakulti Teknologi Maklumat dan Komunikasi (FTMK), Universiti Teknikal Malaysia Melaka (UTeM) for their invaluable support and resources provided throughout this research.

REFERENCES

1. W. Jin, H. Park, X. Xu, E. Brannock, T. Im, and T. Leader, The Synergy Approach: Technology-Enhanced Learning Across Key Stages in CS1. 2025. doi: 10.1145/3769694.3771109.
2. K. Quille and S. Bergin, "CS1: how will they do? How can we help? A decade of research and practice," Computer Science Education, vol. 29, pp. 254–282, 2019, doi: 10.1080/08993408.2019.1612679.
3. M. Rahimi, L. Margulieux, D. Towell, J. Calver, D. Loksa, and J. Prather, "The Impact of Students' Views of Failure on Performance in Introductory Programming Courses," in Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 1, 2025, p. doi: 10.1145/3724363.3729112.
4. B. Bubnic, M. Mernik, and T. Kosar, "Exploring the Predictive Potential of Complex Problem-Solving in Computing Education: A Case Study in the Introductory Programming Course," Mathematics, p., 2024, doi: 10.3390/math12111655.
5. G. Pitts and A. Aggarwal, "Engineering Students' Self-Efficacy, Perceptions, and Performance in a Flipped CS1 Course," ArXiv, vol. abs/2606.01471, p., 2026, doi: 10.48550/arxiv.2606.01471.
6. A. Salguero, Q. Cutts, and L. Porter, "Applying CS0/CS1 Student Success Factors and Outcomes to Biggs' 3P Educational Model," in Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1, 2024, p. doi: 10.1145/3626252.3630781.
7. S. Bergin and R. Reilly, "Predicting introductory programming performance: A multi-institutional multivariate study," Computer Science Education, vol. 16, pp. 303–323, 2006, doi: 10.1080/08993400600997096.
8. J. Köhler, L. Hidalgo, and J. Jara, "Predicting Students' Outcome in an Introductory Programming Course: Leveraging the Student Background," Applied Sciences, p., 2023, doi: 10.3390/app132111994.
9. Y. Ayalew, E. Tshukudu, and M. Lefoane, "Factors Affecting Programming Performance of First Year Students at a University in Botswana," African Journal of Research in Mathematics, Science and Technology Education, vol. 22, pp. 363–373, 2018, doi: 10.1080/18117295.2018.1540169.
10. F. Morales, J. Wong-Ma, and S. Gago-Masague, "Impacts of Academic Preparedness on CS1 Performance," in Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 2, 2024, p. doi: 10.1145/3626253.3635614.
11. E. G. Calaluan Hazel, "The Relationship between Mathematics Competency and Programming Performance of Computer Science 1st Year Students," International Journal of Advanced Research in Science Communication and Technology, p., 2026, doi: 10.48175/ijarsct-36442.

12. M. Mohsin, M. Wahab, M. Zaiyadi, and C. Hibadullah, "An Investigation into Influence Factor of Student Programming Grade Using Association Rule Mining," Adv. Inf. Sci. Serv. Sci., vol. 2, pp. 19–27, 2010, doi: 10.4156/aiss.vol2.issue2.3.
13. G. Bui, N. Sibia, A. Z. Bernuy, M. Liut, and A. Petersen, "Prior Programming Experience: A Persistent Performance Gap in CS1 and CS2," in Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1, 2023, p. doi: 10.1145/3545945.3569752.
14. A.-J. Lakanen and V. Isomöttönen, "CS1: Intrinsic Motivation, Self-Efficacy, and Effort," Informatics Educ., vol. 22, pp. 651–670, 2023, doi: 10.15388/infedu.2023.26.
15. Y.-T. Kuo and Y.-C. Kuo, "African American Students' Academic and Web Programming Self-Efficacy, Learning Performance, and Perceptions towards Computer Programming in Web Design Courses," Educ. Sci. (Basel)., p., 2023, doi: 10.3390/educsci13121236.
16. J. Pires, A. Gomes, A. Borges, and F. Correia, Predicting Higher Education Student's Aptitude to Learn to Program: A Systematic Literature Review. 2024. doi: 10.1145/3674912.3674949.
17. J. Llanos, V. Bucheli, and F. Restrepo-Calle, "Early prediction of student performance in CS1 programming courses," PeerJ Comput. Sci., vol. 9, p., 2023, doi: 10.7717/peerj-cs.1655.
18. R. Gutiérrez-Benítez, A. Vásquez-Guerra, and J. Carrasco-Sáez, "Who Fails and Why: Student Trajectories and Early Prediction of Performance in an Introductory Programming Course," Applied Sciences, p., 2026, doi: 10.3390/app16115644.
19. D. Zhidkikh et al., "Reproducing Predictive Learning Analytics in CS1: Toward Generalizable and Explainable Models for Enhancing Student Retention," J. Learn. Anal., vol. 11, pp. 132–150, 2024, doi: 10.18608/jla.2024.7979.
20. O. Kerr, L. Ball, and N. Danino, "Harnessing Pre-Course Aptitude Tests to Predict Performance on an Introductory Programming Assessment in Higher Education," ACM Transactions on Computing Education, p., 2026, doi: 10.1145/3806059.
21. E. Tomai and C. Reilly, "The impact of math preparedness on introductory programming (CS1) success (abstract only)," in Proceedings of the 45th ACM technical symposium on Computer science education, 2014, p. doi: 10.1145/2538862.2544292.
22. C. Wilcox and A. Lionelle, "Quantifying the Benefits of Prior Programming Experience in an Introductory Computer Science Course," in Proceedings of the 49th ACM Technical Symposium on Computer Science Education, 2018, p. doi: 10.1145/3159450.3159480.
23. K. Malik, A. Richardson, M. Craig, and A. Petersen, Interactive Effects of Prior Experience and Gender on Self-Efficacy and Achievement in CS1. 2025. doi: 10.1145/3702652.3744216.
24. B. Wilson and S. Shrock, "Contributing to success in an introductory computer science course: a study of twelve factors," pp. 184–188, 2001, doi: 10.1145/364447.364581.
25. P. Ventura, "Identifying predictors of success for an objects-first CS1," Computer Science Education, vol. 15, pp. 223–243, 2005, doi: 10.1080/08993400500224419.
26. J. Gorson and E. O'Rourke, Why do CS1 Students Think They're Bad at Programming?: Investigating Self-efficacy and Self-assessments at Three Universities. 2020. doi: 10.1145/3372782.3406273.
27. C. Van Petegem et al., "Pass/Fail Prediction in Programming Courses," Journal of Educational Computing Research, vol. 61, pp. 68–95, 2022, doi: 10.1177/07356331221085595.
28. P. Byrne and G. Lyons, "The effect of student attributes on success in programming," in Proceedings of the 6th annual conference on Innovation and technology in computer science education, New York, NY, USA: ACM, Jun. 2001, pp. 49–52. doi: 10.1145/377435.377467.
29. S. Bergin and R. Reilly, "Programming," ACM SIGCSE Bulletin, vol. 37, no. 1, pp. 411–415, Feb. 2005, doi: 10.1145/1047124.1047480.