Available online at: https://journal.ikippgriptk.ac.id/index.php/saintek
Jurnal Pendidikan Informatika dan Sains
14(1), 2025, 109-120
E-ISSN: 2407-1536
P-ISSN: 2089-2802

THE IMPACT OF COMPUTATIONAL MODELING ON STUDENT SUCCESS IN ALGORITHM AND PROGRAMMING TRACK COURSES

Husnul Hakim*, Natalia, Cecilia Esti Nugraheni
Center for Data Science and Artificial Intelligence System, Program Studi Informatika, Fakultas Sains, Universitas Katolik Parahyangan, Jalan Ciumbuleuit No. 94, Bandung, Jawa Barat, Indonesia
*email: husnulhakim@unpar.ac.id

Received: 2025-04-18 Accepted: 2025-06-23 Published: 2025-06-28

Abstrak

Mata kuliah pemrograman, seperti Pemrograman Berorientasi Objek (PBO), Algoritma dan Struktur Data (ASD), dan Desain dan Analisis Algoritma (DAA) di Program Studi Informatika Universitas Katolik Parahyangan (IF UNPAR), menunjukkan tingkat kelulusan dan nilai rata-rata yang relatif rendah. Untuk meningkatkan kemampuan problem-solving mahasiswa, Kurikulum 2018 IF UNPAR menambahkan mata kuliah wajib Pemodelan untuk Komputasi (PUK). Penelitian ini bertujuan menganalisis pengaruh nilai PUK terhadap performa akademik mahasiswa pada mata kuliah pemrograman. Kami menggunakan pemodelan regresi linear dan klasifikasi Naïve Bayes untuk memprediksi nilai mahasiswa. Hasilnya, model regresi menghasilkan residual standard error antara 15,43 hingga 28,15, sedangkan model Naïve Bayes mencapai Root Mean Squared Error (RMSE) antara 1,29 hingga 1,85. Temuan ini mengindikasikan bahwa nilai PUK dapat menjadi indikator awal keberhasilan mahasiswa dalam mata kuliah pemrograman, sekaligus mendukung integrasi kemampuan pemodelan dan *problem solving* dalam kurikulum informatika.
Kata kunci: analisis data, regresi linier, naïve bayes, kurikulum informatika, pemrograman.

Abstract

Programming courses, such as Object-Oriented Programming (PBO), Algorithms and Data Structures (ASD), and Design and Analysis of Algorithms (DAA) at Parahyangan Catholic University's Informatics Study Program (IF UNPAR), have shown relatively low passing rates and average grades. To enhance students' problem-solving abilities, IF UNPAR's 2018 Curriculum introduced the compulsory course, Modeling for Computation (PUK). This study aims to analyze the influence of PUK grades on students' academic performance in programming courses. We used linear regression modeling and Naïve Bayes classification to predict student grades. The results show that the regression model yielded a residual standard error between 15.43 and 28.15, while the Naïve Bayes model achieved a Root Mean Squared Error (RMSE) between 1.29 and 1.85. These findings indicate that PUK grades can serve as an early indicator of student success in programming courses, simultaneously supporting the integration of modeling and problem solving capabilities into the informatics curriculum.
Keywords: *data analysis, linear regression, naïve bayes, informatics curriculum, programming.*

How to cite (in APA style): Hakim, H., Natalia, N., & Nugraheni, C. E. (2025). The impact of computational modeling on student success in algorithm and programming track courses. *Jurnal Pendidikan Informatika Dan Sains*, *14*(1), 109–120. https://doi.org/10.31571/saintek.v14i1.8875

DOI: 10.31571/saintek.v14i1.8875

INTRODUCTION

According to the Computing Curricula from the Association for Computing Machinery (ACM), computer science and informatics graduates must possess competencies across various domains, including algorithms and complexity, computer architecture and organization, graphics and visualization, human-computer interaction, software development fundamentals, and software engineering (Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science (2013)). To support the attainment of these competencies, especially those related to programming and software development, the Informatics Department at Universitas Katolik Parahyangan (UNPAR) offers a range of courses within its programming track. Despite a high level of satisfaction reported by employers regarding alumni competencies (40.38% rated excellent, 46.15% good), these educational efforts present significant challenges for students. High failure rates and low final grades in the Object-Oriented Programming (PBO) course, a foundational programming subject in the previous 2013 curriculum, underscore this issue. As Table 1 illustrates, PBO failure rates consistently exceeded 30%, peaking at 52.54% in Odd Semester 2016, with average grades often falling below passing standards. These figures indicate a fundamental hurdle in students' initial grasp and application of programming concepts, potentially impeding their progress in more advanced courses.

Tabel 1. Average final grades and failure percentage in the PBO course

No	Semester	Average	Failure Rates (%)
1	Odd 2013	59,93	37,03 %
2	Even 2013	51,20	44,12 %
3	Odd 2014	56,75	35,93 %
4	Even 2014	53,24	25,71 %
5	Odd 2015	60,89	33,33 %
6	Even 2015	49,10	40,91 %
7	Odd 2016	49,20	52,54 %
8	Even 2016	58,31	32,56 %
9	Odd 2017	53,60	38,50 %
10	Even 2017	47,11	50,00 %

Numerous studies consistently highlight the critical importance of problem-solving skills in programming education. Barlow-Jones & van der Westhuizen (2017) found logical, numerical, and verbal reasoning influence student success. Kožuh et al. (2018) assessed problem-solving skills in novice programmers, finding a positive effect on study success. Lawan et al. (2019) identified weak problem-solving skills as an obstacle, while Medeiros et al. (2019) stated it's both a needed skill and a major challenge. Veerasamy et al. (2019) also found a strong relationship, and Ubaidullah et al. (2021) emphasized discussion methods and problem-solving approaches. Alshaye et al. (2023) showed online problem-based learning significantly improved problem-solving and higher-order thinking. Banawi et al. (2024) demonstrated a significant correlation between study success and problem-solving skills, and Ranjeeth & Padayachee (2024) identified problem-solving and self-efficacy as predictors of programming skills. Interestingly, Harimurti et al. (2019) even found programming skills can improve problem-solving skills.

Other studies also highlight the importance of introductory programming courses as a foundation for further study. Martz et al. (2017) showed introductory programming is crucial for success in advanced courses, and Topalli & Cagiltay (2018) along with Bubnic et al. (2024) found complex problem-solving skills in these courses contributed to improved student performance. Köhler et al. (2023) further indicated that various background variables, including first-semester GPA, lecturer, and mathematical and language proficiency, can accurately predict student performance in introductory programming.

Various learning frameworks have been developed to improve programming and problem-solving skills. Kumar (2015) examined code-tracing's impact, while Lee et al. (2015) developed game-based and competition-based learning models. Loksa & Ko (2016) explored the relationship between self-regulation and novice success, and Bawamohiddin & Razali (2017) compiled a framework emphasizing problem-solving. Topalli & Cagiltay (2018) also highlighted technologies like game-based learning, problem-based learning, visual programming, and projects. Malik et al. (2019) developed the web-based and mobile PROBSOL framework using pseudocode. Schefer-Wenzl & Miladinovic (2019) designed a blended learning course for complex problem-solving. Erol & Çırak (2022) showed Scratch-based programming improved problem-solving abilities. Pinto & Terroso (2022) explored gamification in an introductory programming course. Chen & Huang (2024) examined Scratch within a metaverse virtual environment alongside the Jigsaw learning strategy, showing gains in programming, computational thinking, and motivation. Wong et al. (2024) investigated computational thinking development using a block-based programming curriculum, revealing significant improvements in algorithmic thinking and debugging skills.

To address the documented challenges in foundational programming and align with the extensive literature emphasizing problem-solving, the IF UNPAR revised its curriculum in 2018. This revision introduced the PUK course, specifically designed to cultivate problem-solving skills *before* students write program code. In PUK, students focus on logically and systematically designing solutions using pseudocode and flowcharts, making it a prerequisite for the DASPRO course. While existing literature extensively covers the importance of problem-solving in programming and various learning frameworks, a significant gap remains, particularly in the Indonesian context, regarding the explicit empirical evaluation of a dedicated problem-solving-focused course's impact on subsequent tiered programming courses at the tertiary level. Few studies explicitly analyze how a foundational *non-coding* problem-solving course directly contributes to student achievement across an entire progressive programming pathway. This study aims to bridge this identified research gap by empirically analyzing the effect of the PUK course on student performance in DASPRO, ASD, and DAA.

The analysis involves data exploration using correlation and hypothesis tests, and develops a predictive model for final grades and letter grades based on PUK course scores. The novelty of this study lies in its comprehensive empirical evaluation of the contribution of a dedicated problem-solving-based course (PUK) to staged achievements within the programming curriculum. By quantitatively validating PUK's role, this research provides concrete evidence for the importance of a strong problem-solving foundation in informatics curricula, while offering a predictive framework that can identify students requiring early support. These findings are expected to offer valuable insights for higher education institutions in designing more effective and responsive programming curricula. Specifically, this study addresses the following research question: Does the Modeling for Computation (PUK) course significantly influence student success in programming track courses such as DASPRO, ASD, and DAA?

METHOD

This study employed a quantitative research methodology to analyze the impact of the PUK course on student performance in subsequent programming courses. Figure 1 illustrates the overall research workflow. We collected a total of 447 student records for programming-related courses across two distinct curricula: the 2013 curriculum and the 2018 curriculum. The 2013 curriculum data included 227 students from the 2013, 2014, and 2015 cohorts, enrolled in PBO, ASD, and DAA. For the 2018 curriculum, we obtained data from 220 students from the 2016, 2017, and 2018 cohorts, covering PUK, DASPRO, ASD, and DAA. Table 2 provides a detailed description of the courses and student demographics for each curriculum.

After data collection, we performed a data preprocessing stage. During preprocessing, we specifically considered grades from the first attempt of course enrollment within the designated semester. For instance, if a student under the 2013 curriculum first enrolled in the ASD course in the third semester and received a grade, this grade was not directly included. Under the 2013 curriculum, ASD was scheduled for the second semester. Taking the course for the first time in the third semester implied the student failed ASD in its designated semester; therefore, their ASD grade for that designated second semester was recorded as 0 for the purpose of this study. This method ensured consistency in evaluating performance within the prescribed curriculum structure. Following preprocessing, we conducted data exploration to understand the characteristics of the student data, which revealed relatively similar student profiles across cohorts. To analyze the impact of the PUK course on student success, we performed various statistical analyses on student grades from the PUK, PBO, ASD, DAA, and DASPRO courses. Table 3 outlines the specific analyses conducted in this study.

Table 2. Data Description

No.	Course Name	Curriculum	Position in Curriculum	Average Score	Standard Deviation	Total Students
1	PBO	2013	Semester I	59.357	23.180	227
2	ASD	2013	Semester II	62.920	19.370	227
3	DAA	2013	Semester III	68.034	15.782	227
4	PUK	2018	Semester I	54.518	24.823	220
5	DASPRO	2018	Semester II	69.358	16.540	220
6	ASD	2018	Semester III	60.598	16.471	220
7	DAA	2018	Semester IV	70.440	13.180	220

Table 3. List of Performed Analysis

No.	Analysis	Method
1	Comparison of the Proportion of Final Letter Grades in ASD Before and After the Introduction of the PUK Course	Proportion Hypothesis Test
2	Comparison of the Proportion of Final Letter Grades in DAA Before and After the Introduction of the PUK Course	Proportion Hypothesis Test
3	Comparison of the Mean Final Numeric Grades in ASD Before and After the Introduction of the PUK Course	Mean Difference Hypothesis Test
4	Comparison of the Mean Final Numeric Grades in DAA Before and After the Introduction of the PUK Course	Mean Difference Hypothesis Test
5	Relationship Between Final Numeric Grades in PUK and Final Numeric Grades in DASPRO	Linear Regression
6	Relationship Between Final Numeric Grades in PUK and Final Numeric Grades in ASD	Linear Regression
7	Relationship Between Final Numeric Grades in PUK and Final Numeric Grades in DAA	Linear Regression
8	Prediction of Final Letter Grades in DASPRO Based on Letter Grades in PUK	Naïve Bayes
9	Prediction of Final Letter Grades in DASPRO Based on Letter Grades in ASD	Naïve Bayes
10	Prediction of Final Letter Grades in DASPRO Based on Letter Grades in DAA	Naïve Bayes

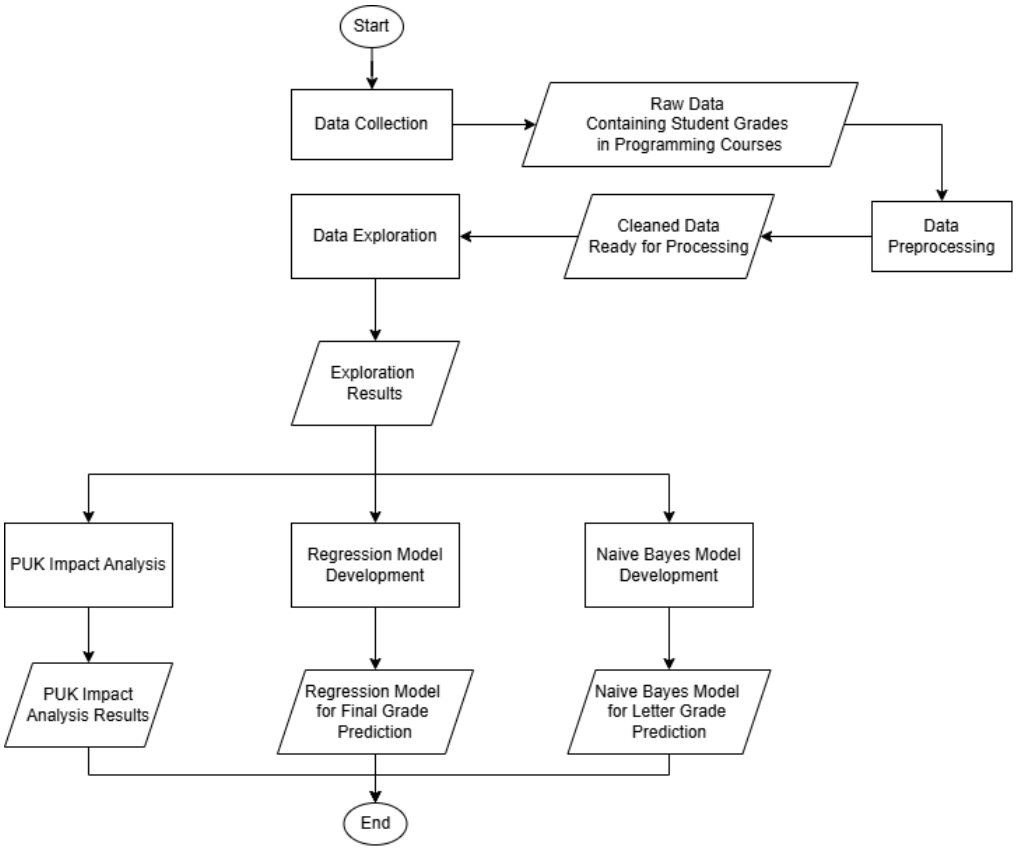
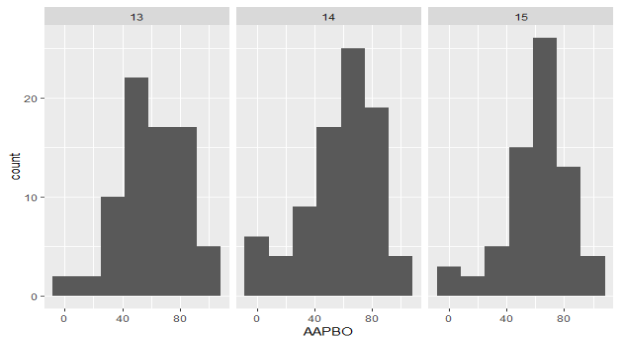


Figure 1. Research methodology

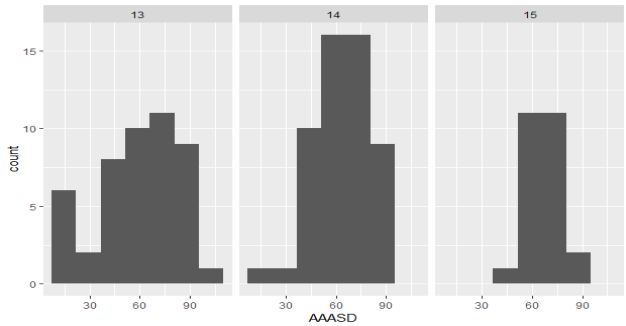
RESULTS AND DISCUSSION

To determine whether students enrolled in courses under the 2013 Curriculum and the 2018 Curriculum exhibit similar characteristics, various data distribution comparisons were conducted as follows: data exploration of PBO, ASD, and DAA grades in the 2013 curriculum; and data exploration of PUK, DASPRO, ASD, and DAA grades in the 2018 curriculum.

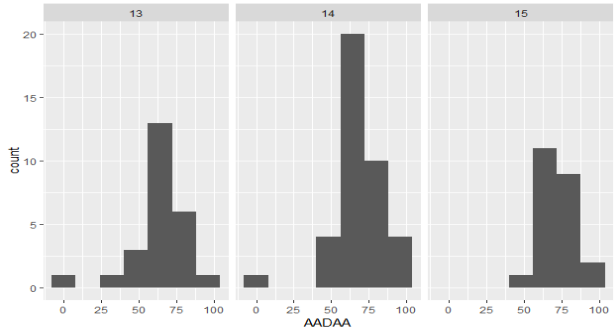
The comparison of the distribution of PBO, ASD, and DAA grades in the 2013 Curriculum is presented in Figure 2. As shown in Figure 2, the grade distribution of students from the 2013, 2014, and 2015 cohorts for the PBO, ASD, and DAA course in the 2013 Curriculum is relatively similar. Therefore, it can be concluded that the academic performance of students from these cohorts in the PBO, ASD, and DAA course does not differ significantly. Furthermore, the quality of course materials and instruction can also be considered relatively consistent across the three cohorts, indicating that the year of admission does not have a significant impact on the outcomes of the PBO, ASD, and DAA course.



(a)

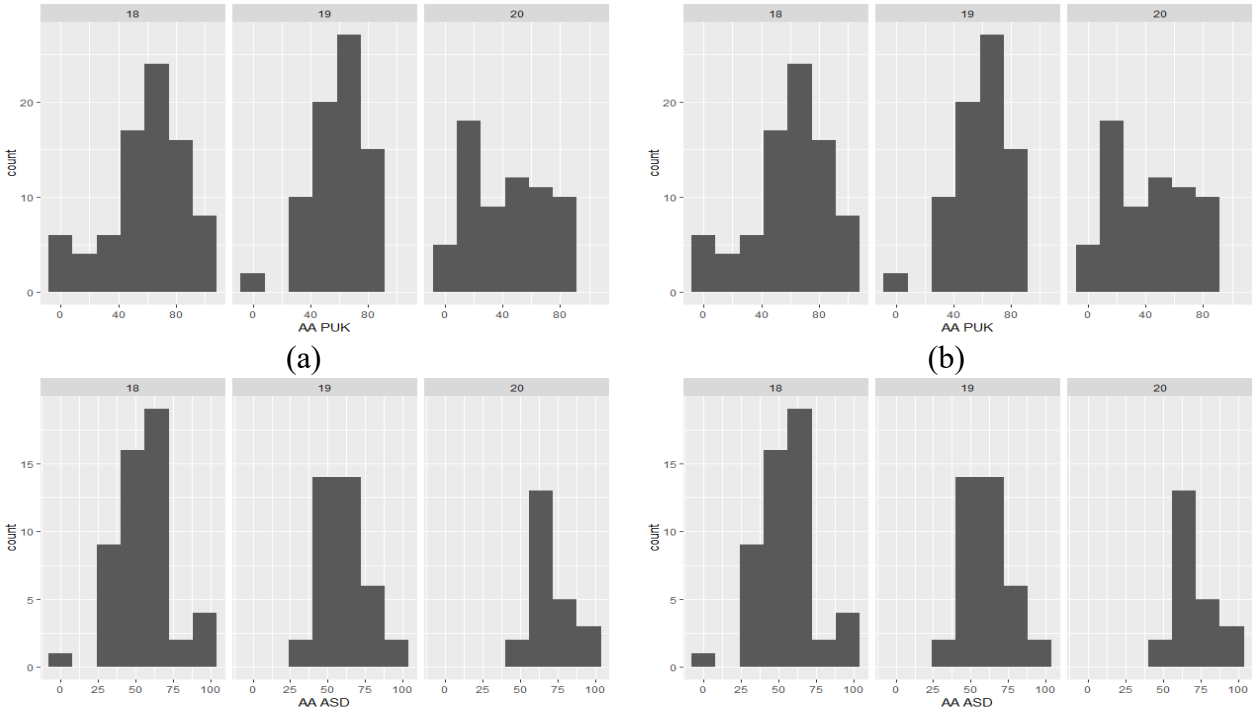


(b)



(c)

Figure 2. Grade distribution of students in the 2013 Curriculum: (a) PBO course, (b) ASD course, and (c) DAA course. Each subfigure represents the distribution for the 2013 cohort (left), 2014 cohort (middle), and 2015 cohort (right)



(a) (b)

(c) (d)

Figure 3. Grade distribution of students in the 2018 Curriculum: (a) PUK course, (b) PBO course, (c) ASD course, and (d) DAA course. Each subfigure represents the distribution for the 2018 cohort (left), 2019 cohort (middle), and 2020 cohort (right)

The comparison of grade distributions for PUK, DASPRO, ASD, and DAA courses in the 2018 Curriculum can be seen in Figure 3. Figure 3 shows that the grade distributions of students from the 2018, 2019, and 2020 cohorts for these four programming track courses in the 2018 Curriculum are not significantly different. Therefore, it can be concluded that the abilities of students from the 2018, 2019, and 2020 cohorts in these courses are comparable, as are the quality of the course materials and instruction.

To analyze the impact of PUK on programming track courses, data analysis was conducted on the ASD course before and after the inclusion of PUK in the curriculum, as well as on the DAA course before and after the inclusion of PUK in the curriculum. The impact on the DASPRO course was not analyzed, as DASPRO is only offered in the 2018 curriculum.

A proportion test revealed no statistically significant difference in the letter grade distribution of the ASD course between the 2013 and 2018 curricula (p-value = 0.056). Figure 4 presents a comparison of the proportion of ASD grades in the 2013 and 2018 curricula.

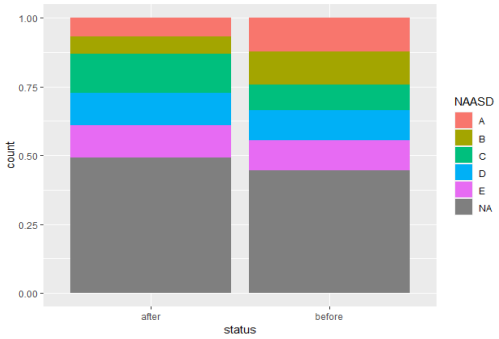


Figure 4. Comparison of the letter grade proportion in the ASD course of the 2018 curriculum (left) and the 2013 curriculum (right)

Although the mean difference in ASD scores is statistically non-significant (p-value = 0.284), this result warrants further contextual analysis. It may be attributed to several unaccounted factors, such as variations in instructional methods, differences in assessment strategies, or the diversity of students' academic backgrounds. Indeed, prior research highlights that various background variables, including first-semester GPA and mathematical and language proficiency, accurately predict student performance in introductory programming (Köhler et al., 2023). Further investigation is needed to understand the specific role of these variables in the observed outcome. Figure 5 presents a comparison of the final scores for the ASD course in the 2013 and 2018 curricula.

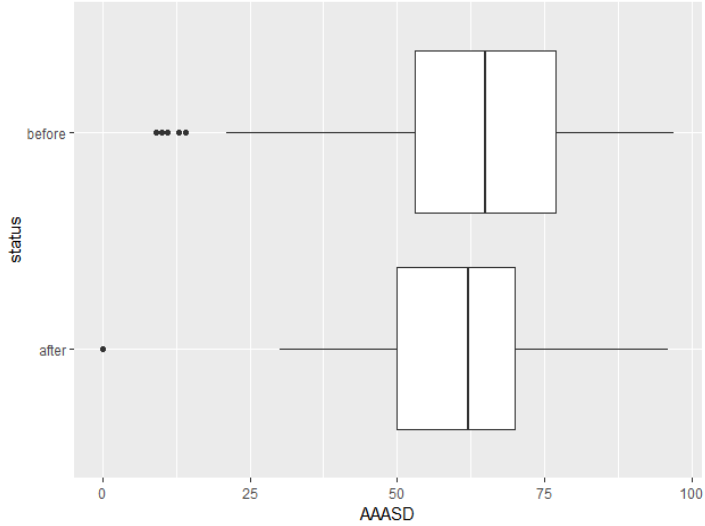


Figure 5. Comparison of final scores for the ASD course in the 2018 curriculum (bottom) and the 2013 curriculum (top)

Through a proportion test, the observed difference in letter grade distribution between the ASD course of the 2013 curriculum and the DAA course of the 2013 curriculum (p = 0.067) approached statistical significance. However, as it did not meet the conventional 0.05 threshold, this finding should be interpreted with caution.

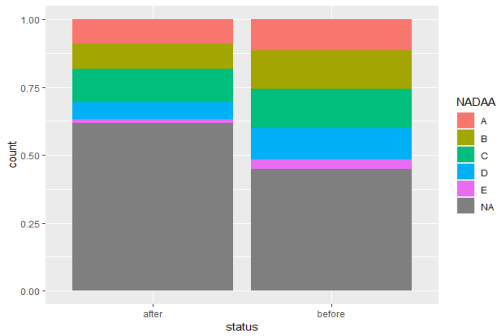


Figure 6. Comparison of the letter grade proportion in the DAA course of the 2018 curriculum (left) and the 2013 curriculum (right)

In terms of the final grade average, the average final grade for the DAA course in the 2013 curriculum is 66.664, while the average final grade for the ASD course in the 2018 curriculum is 70.440. Although there appears to be an increase in the average final grade, a t-test for mean difference yields a p-value of 0.067, indicating that PUK has no significant effect on the DAA course. Figure 6 presents a comparison of the final grades for the ASD course in the 2013 and 2018 curricula.

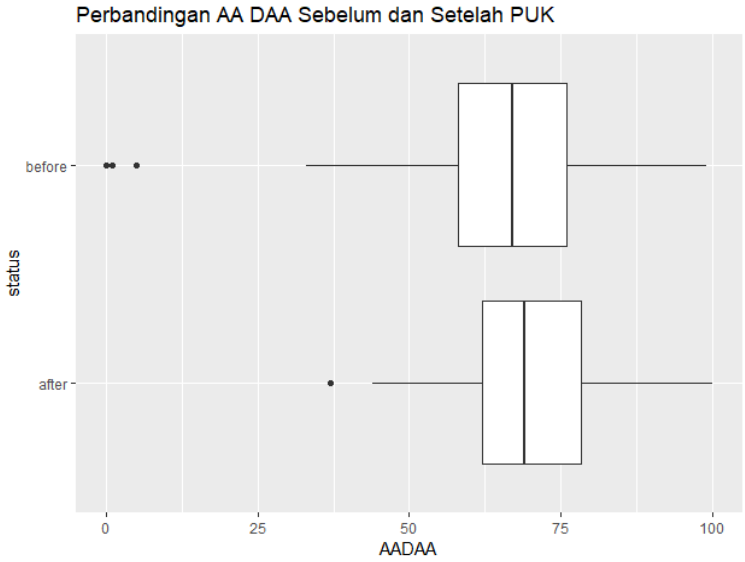


Figure 7. Comparison of final scores for the DAA course in the 2018 curriculum (bottom) and the 2013 curriculum (top).

To determine the relationship between PUK grades and DASPRO grades, three tests were conducted. The first was a correlation test to examine the relationship between the final numerical grades of PUK and DASPRO. The second was the development of a regression model to predict the final numerical grade of DASPRO based on the final numerical grade of PUK. The third involved building a Naïve Bayes model to predict the letter grade of DASPRO based on the letter grade of PUK.

The observed correlation of 0.607 between the final numerical grades of PUK and DASPRO indicates a moderately strong relationship between performance in a modeling-based, pre-programming course and success in a foundational programming course. This finding holds significant educational implications, suggesting that students who develop strong algorithmic thinking and structured problem-solving skills in courses like PUK are more likely to perform well when they begin writing program code. This linkage is crucial, as PUK specifically focuses on designing solutions logically and systematically using pseudocode and flowcharts *before* students are introduced to programming syntax. This result aligns with prior research that consistently highlights the importance of problem-solving abilities as a foundational skill for programming proficiency

(Alshaye et al., 2023; Banawi et al., 2024; Lawan et al., 2019; Medeiros et al., 2019; Ranjeeth & Padayachee, 2024; Veerasamy et al., 2019). Furthermore, it supports the notion that developing computational thinking and logical reasoning skills, as fostered in PUK, are highly transferable and predictive of success in subsequent programming challenges (Harimurti et al., 2019; Wong et al., 2024). The PUK course's emphasis on designing solutions before coding also reflects pedagogical approaches proven effective in computer science education, such as problem-based learning and the use of pseudocode in frameworks designed to improve problem-solving (Malik et al., 2019; Schefer-Wenzl & Miladinovic, 2019; Topalli & Cagiltay, 2018). Therefore, our findings underscore the value of explicitly integrating problem-solving instruction early in the computer science curriculum, providing a solid conceptual base before diving into coding implementation.

In addition to the correlation test, a regression model was developed to further analyze the relationship between the final numerical grades of PUK and DASPRO, as shown in Equation (1). In Equation (1), *daspro* represents the final numerical grade of DASPRO and *puk* represents the final numerical grade of PUK.

$$daspro = 0{,}9263 \times puk + 2{,}4162 \qquad \textbf{(1)}$$

Using the Naïve Bayes method, a prediction was performed to determine the final letter grade of DASPRO based on the final letter grade of PUK. To build this model, data from 146 students who had completed the PUK course were used. A total of 95 data points were used for training, while the remaining 41 were used for testing. The evaluation resulted in an RMSE of 1.288, meaning the prediction model deviated by approximately one letter grade. The RMSE was calculated by converting letter grades A, B, C, D, and E into numerical values of 4, 3, 2, 1, and 0, respectively. The classification accuracy was 0.41, with precision and recall values of 0.89 and 0.89 respectively.

To determine the relationship between PUK scores and ASD scores, three tests were conducted. The first was a correlation test to examine the relationship between the final numerical scores of PUK and ASD. The second was the development of a regression model to predict the final numerical score of ASD based on the final numerical score of PUK. The third involved constructing a Naïve Bayes model to predict the letter grade of ASD based on the letter grade of PUK.

Using the correlation test, the correlation between the final numerical scores of PUK and ASD was found to be 0.563, with a p-value of 10^{-12}. This indicates a significant correlation between the final numerical score of PUK and the final numerical score of ASD. In addition to the correlation test, a regression model was developed to determine the relationship between the final numerical scores of PUK and ASD, as shown in Equation 2. In Equation 2, *asd* represents the final numerical grade of ASD and *puk* represents the final numerical grade of PUK.

$$asd = 1{,}253 \times puk + 39{,}197 \qquad \textbf{(2)}$$

Using the Naïve Bayes method, a prediction was made to determine the final letter grade of the ASD course based on the final letter grade of the PUK course. This model was built using data from 146 students who had passed the PUK course. A total of 95 data points were used for training, while the remaining 41 were used for testing. The classification accuracy was 0.32, with precision and recall values of 0.44 and 0.80, respectively. Although the classification accuracy was relatively low, the model yielded an RMSE of 1.449, suggesting that on average, the predicted grades deviated by only about one letter grade.

To determine the relationship between PUK scores and DAA scores, three tests were conducted. The first was a correlation test to examine the relationship between the final numeric scores of PUK and DAA. The second involved building a regression model to predict the final numeric score of

DAA based on the final numeric score of PUK. The third was the development of a Naïve Bayes model to predict the final letter grade of DAA based on the final letter grade of PUK.

The observed correlation coefficient of 0.663 between the final numeric scores of PUK and DAA indicates a strong and statistically meaningful relationship. This finding carries significant pedagogical relevance, suggesting that early modeling-based instruction, such as that provided by the PUK course, effectively equips students with essential abstract thinking and algorithmic reasoning skills. These skills prove crucial for success in more advanced and conceptually demanding programming courses like Algorithm Design and Analysis. This result strongly aligns with educational theories emphasizing the importance of early cognitive scaffolding and the development of foundational computational thinking skills in computing disciplines, a point consistently highlighted by prior research on introductory programming's role as a vital foundation for advanced studies (Bubnic et al., 2024; Martz et al., 2017; Topalli & Cagiltay, 2018). Furthermore, the strong correlation underscores that the ability to design logical solutions and analyze algorithmic complexity, fostered in PUK, directly translates to success in advanced problem-solving contexts within DAA. This outcome is consistent with numerous studies demonstrating how well-developed problem-solving skills, structured approaches, and effective pedagogical frameworks contribute to improved student performance in complex programming scenarios (Alshaye et al., 2023; Banawi et al., 2024; Malik et al., 2019; Ranjeeth & Padayachee, 2024; Schefer-Wenzl & Miladinovic, 2019; Wong et al., 2024). Therefore, this correlation provides compelling evidence for the long-term impact and transferability of core problem-solving competencies, developed early in the curriculum, on higher-level analytical programming skills.

In addition to the correlation test, a regression model was developed to analyze the relationship between the final numeric scores of PUK and DAA, as shown in Equation (3). In Equation 3, *daa* represents the final numerical grade of DAA and *puk* represents the final numerical grade of PUK.

$$daa = 1{,}823 \times puk - 86{,}675 \quad \textbf{(3)}$$

Using the Naïve Bayes approach, a prediction was made to determine the final letter grade of the DAA course based on the final letter grade of the PUK course. To build this model, data from 146 students who had completed the PUK course were used. A total of 95 data points were utilized for training, while the remaining 41 were used for testing. Although the model showed limited performance in classification with an accuracy of 0.12, precision of 0.27, and recall of 0.67 it achieved an RMSE of 1.854, suggesting that its predictions were, on average, within one letter grade of the actual values.

CONCLUSION

This study aims to examine the effect of the Modeling for Computation (PUK) course in the 2018 curriculum on students' academic success in advanced programming courses, such as Programming Basics (DASPRO), Algorithms and Data Structures (ASD), and Algorithm Design and Analysis (DAA). Based on the results of data analysis, several main conclusions were drawn. First, although there was a small decrease in the average final grade for ASD courses in the 2018 curriculum, statistical analysis showed that the effect of PUK on final grades was not significant. A t-test on the difference in mean grades indicated that the curriculum change did not cause a significant change in students' academic results in DASPRO, ASD, and DAA courses. Second, the correlation test results showed a significant relationship between PUK final grades and final grades for advanced courses, such as DASPRO, ASD, and DAA. This finding suggests that academic achievement in PUK courses is closely related to student success in advanced programming courses, reinforcing the importance of effective teaching in programming foundation courses. Third, the regression and Naïve Bayes models built to predict final grades in advanced courses based on PUK scores showed fairly accurate results, with low RMSE (around one letter grade). This indicates that data-driven prediction

methods, such as linear regression and Naïve Bayes, can provide useful information for predicting students' academic performance in advanced programming courses. Furthermore, this study supports previous findings that modeling and problem-solving skills taught in introductory programming courses, such as PUK, are essential to help students prepare for more complex programming courses. Although the direct effect on academic outcomes was not significant, the results of the correlation analysis indicate that PUK remains an important factor in students' academic success in subsequent courses. Finally, this study suggests that a curriculum change introducing PUK as a prerequisite for advanced programming courses has the potential to positively help students develop the foundational skills needed to succeed in programming. However, the effectiveness of PUK instruction needs to be further evaluated by considering the teaching methods used and other curriculum support. Overall, this study confirms the importance of a curriculum that emphasizes a basic understanding of programming as a strong foundation for students to succeed in advanced programming courses. The results also suggest that data-driven predictive models can support academic decisions and improve teaching efficiency in the future.

REFERENCE

Alshaye, I. A., Tasir, Z., & Jumaat, N. F. (2023). The effectiveness of online problem-based learning tasks on Riyadh's secondary school students' problem-solving ability and programming skills. *Open Education Studies*, *5*(1). https://doi.org/10.1515/edu-2022-0208

Banawi, A., Rumasoreng, M. I., Hasanah, N., Rahawarin, D. A., & Basta, I. (2024). The relationship between problem-solving skills and student academic achievement: A meta-analysis in education. *Journal of Ecohumanism*, *3*(3), 1287–1299. https://doi.org/10.62754/joe.v3i3.3413

Barlow-Jones, G., & van der Westhuizen, D. (2017). Problem solving as a predictor of programming performance. *Communications in Computer and Information Science*, *730*, 209–216. https://doi.org/10.1007/978-3-319-69670-6_14

Bawamohiddin, A. B., & Razali, R. (2017). Problem-based learning for programming education. *International Journal on Advanced Science, Engineering and Information Technology*, *7*(6), 2035–2050. https://doi.org/10.18517/ijaseit.7.6.2232

Bubnic, B., Mernik, M., & Kosar, T. (2024). Exploring the predictive potential of complex problem-solving in computing education: A case study in the introductory programming course. *Mathematics*, *12*(11). https://doi.org/10.3390/math12111655

Chen, C.-M., & Huang, M.-Y. (2024). Enhancing programming learning performance through a Jigsaw collaborative learning method in a metaverse virtual space. *International Journal of STEM Education*, *11*(1), 36. https://doi.org/10.1186/s40594-024-00495-2

ACM, Inc. (2013). *Computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science*. https://doi.org/10.1145/2534860

Erol, O., & Çırak, N. S. (2022). The effect of a programming tool scratch on the problem-solving skills of middle school students. *Education and Information Technologies*, *27*(3), 4065–4086. https://doi.org/10.1007/s10639-021-10776-w

Harimurti, R., Ekohariadi, E., Munoto, M., Asto B, I. G. P., & Winanti, E. T. (2019). Analysis of programming skills concept in developing problem solving skills. *Jurnal Pendidikan Teknologi Dan Kejuruan*, *25*(1), 43–51. https://doi.org/10.21831/jptk.v25i1.22638

Köhler, J., Hidalgo, L., & Jara, J. L. (2023). Predicting students' outcome in an introductory programming course: Leveraging the student background. *Applied Sciences (Switzerland)*, *13*(21). https://doi.org/10.3390/app132111994

Kožuh, I., Krajnc, R., Hadjileontiadis, L. J., & Debevc, M. (2018). Assessment of problem solving ability in novice programmers. *PLoS ONE*, *13*(9). https://doi.org/10.1371/journal.pone.0201919

Kumar, A. N. (2015). Solving code-tracing problems and its effect on code-writing skills pertaining to program semantics. *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*, *2015-June*, 314–319. https://doi.org/10.1145/2729094.2742587

Lawan, A. A., Abdi, A. S., Abuhassan, A. A., & Khalid, M. S. (2019). What is difficult in learning programming language based on problem-solving skills? *2019 International Conference on Advanced Science and Engineering, ICOASE 2019*, 18–22. https://doi.org/10.1109/ICOASE.2019.8723740

Lee, Y. C., Ko, J. Y., & Kim, S. B. (2015, July). Improving Problem Solving and Programming Skills through Learning Games of an Inter-competition Type. In *2015 International Conference on Artificial Intelligence and Industrial Engineering* (pp. 290-293). Atlantis Press.

Loksa, D., & Ko, A. J. (2016). The role of self-regulation in programming problem solving process and success. *ICER 2016 - Proceedings of the 2016 ACM Conference on International Computing Education Research*, 83–91. https://doi.org/10.1145/2960310.2960334

Malik, S. I., Mathew, R., Al-Nuaimi, R., Al-Sideiri, A., & Coldwell-Neilson, J. (2019). Learning problem solving skills: Comparison of E-learning and M-learning in an introductory programming course. *Education and Information Technologies*, *24*(5), 2779–2796. https://doi.org/10.1007/s10639-019-09896-1

Martz, B., Hughes, J., & Braun, F. (2017). Creativity and problem-solving: Closing the skills gap. *Journal of Computer Information Systems*, *57*(1), 39–48. https://doi.org/10.1080/08874417.2016.1181492

Medeiros, R. P., Ramalho, G. L., & Falcao, T. P. (2019). A Systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education*, *62*(2), 77–90. https://doi.org/10.1109/TE.2018.2864133

Pinto, M., & Terroso, T. (2022). Learning computer programming: A gamified approach. *Open Access Series in Informatics*, *102*. https://doi.org/10.4230/OASIcs.ICPEC.2022.11

Ranjeeth, L., & Padayachee, I. (2024). Factors that influence computer programming proficiency in higher education: A case study of Information Technology students. *South African Computer Journal*, *36*(1), 40–75. https://doi.org/10.18489/SACJ.V36I1.18819

Schefer-Wenzl, S., & Miladinovic, I. (2019). Developing complex problem-solving skills: An engineering perspective. *International Journal of Advanced Corporate Learning (IJAC)*, *12*(3), 82. https://doi.org/10.3991/ijac.v12i3.11067

Topalli, D., & Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers and Education*, *120*, 64–74. https://doi.org/10.1016/j.compedu.2018.01.011

Ubaidullah, N. H., Mohamed, Z., Hamid, J., & Sulaiman, S. (2021). Discovering the role of problem-solving and discussion techniques in the teaching programming environment to improve students‟ computational thinking skills. *International Journal of Information and Education Technology*, *11*(12), 615–623. https://doi.org/10.18178/IJIET.2021.11.12.1572

Veerasamy, A. K., D'Souza, D., Lindén, R., & Laakso, M. J. (2019). Relationship between perceived problem-solving skills and academic performance of novice learners in introductory programming courses. *Journal of Computer Assisted Learning*, *35*(2), 246–255. https://doi.org/10.1111/jcal.12326

Wong, G. K. W., Jian, S., & Cheung, H. Y. (2024). Engaging children in developing algorithmic thinking and debugging skills in primary schools: A mixed-methods multiple case study. *Education and Information Technologies*. https://doi.org/10.1007/s10639-024-12448-x