How Introduction of Programming to the K-12 Curriculum Influences Prior Knowledge for CS1

Sondre S. Bolland
sondre.bolland@uib.no
University of Bergen
Bergen, Norway

Torstein J. F. Strømme
torstein.stromme@uib.no
University of Bergen
Bergen, Norway

Abstract

In 2020, the Norwegian curriculum in K-12 education (ISCED 1-3) was reformed to include computational thinking as a core element, and programming has since been integrated in subjects such as mathematics, science and arts. Starting with those who graduated from upper secondary school in 2023, Norwegian university freshmen will now have been exposed to programming with Python in their mathematics courses throughout upper secondary school. This newfound prior knowledge of programming in the student body could potentially have a significant effect on how introduction to programming in higher education (CS1) should be taught.

We analyze results of the *Prior Informatics Knowledge Assessment* (PIKA), which was administered to $n = 3,038$ students that were about to begin their CS1 studies in one of eight different higher education institutions in Norway. We find that students who graduated from upper secondary school in 2023 or 2024 (and thus experienced the new curriculum) significantly outperform those who graduated earlier. Using Hodges-Lehmann estimator for central tendency we observe a pseudo-median difference of 23.9 out of 100 points.

We find that some groups have benefited more from the curriculum change than others. In particular, students that opt for the most advanced mathematics courses gain more from the new curriculum, meaning that the prior knowledge gap has increased with respect to mathematics background. Women also gained more than men, meaning that the gap is reduced with respect to gender.

CCS Concepts

• **Social and professional topics** → **Student assessment**; **K-12 education**; **CS1**.

Keywords

K-12, curriculum reform, CS1, prior knowledge

ACM Reference Format:
Sondre S. Bolland and Torstein J. F. Strømme. 2025. How Introduction of Programming to the K-12 Curriculum Influences Prior Knowledge for CS1. In *Proceedings of the ACM Conference on Global Computing Education (CompEd 2025).* ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

CompEd 2025, October 23–25, 2025, Gaborone, Botswana

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

The integration of programming and computational thinking (CT) into K-12 education (ISCED 1-3) has become increasingly widespread, with many countries incorporating these elements into their national curricula [2, 3, 8, 10, 15, 17]. In Norway, the updated curriculum, "Kunnskapsløftet 2020" (LK20) [10], introduced programming and CT as explicit objectives in various subjects, including mathematics, science, physics, music, and arts and crafts. The emphasis is particularly strong in mathematics, where students already in second grade begin learning CT. These changes will likely influence the prior knowledge of students entering higher education, potentially requiring adjustments to the introductory programming course (CS1) to focus on more advanced material.

To probe how far university freshmen have already progressed into the CS1 curriculum before the semester starts, the *Prior Informatics Knowledge Assessment*[1] (PIKA) was created [5]. This test assesses students' familiarity with CS1 content before the semester starts, using a series of programming tasks that address the course's core concepts [6]. In August 2024, the test was administered to $n = 3,038$ students from eight higher education institutions in Norway. Preliminary results suggest that the reform has positively impacted students' programming knowledge. Graduates under the revised curriculum (2023 and 2024 cohorts) achieved a mean score of 50 out of 100 points, compared to a mean score of 31.2 among earlier graduates.

However, these averages tell only part of the story. A more detailed analysis reveals substantial variation in outcomes based on students' educational backgrounds and other contextual factors. This study elaborates on these details, exploring how the reform has shaped students' prior programming knowledge. We aim to answer the following questions:

- **RQ1:** To what extent has the introduction of computational thinking and programming in the national K-12 curriculum influenced students' prior programming knowledge upon entering CS1 in higher education?
- **RQ2:** Are there specific groups (based on educational background and gender) for whom the curriculum change has had a greater impact?
- **RQ3:** How does the effect of LK20 compare to the effect of completing a typical CS1 course in higher education?

2 Background

2.1 Evaluating K-12 curriculum reforms for CT

Many countries worldwide have integrated CT and programming into their K-12 education systems, prompting research aimed at

[1] PIKA was previously named the *Nordic/National Prior Knowledge in Programming*.

assessing students' knowledge and skills in these areas. However, we have not found much literature that rigorously evaluates which effects such curriculum changes has on incoming university freshmen.

The only example we found is from Taiwan, where an analysis of the *Chippy Challenge* showed that a revised curriculum introduced in 2019 had a positive effect on students' programming skills [9]. However, this study focused on students entering upper secondary school, assessing programming knowledge gained in grades 7–9 rather than after completing K-12.

2.2 Instruments for evaluating CT at the end of K-12 or prior to university studies

Several tests have been developed that might be suitable to gauge CT ability of incoming university freshmen before they undertake CS1. Some such tests avoid text-based programming entirely, and rely on block-based structures or logic puzzles in the spirit of "CS unplugged" [1, 7, 14, 19], whilst others utilize a mixture of text-based coding and block-based constructs [18] or pseudo-code [12]. The "Foundational CS1 assessment" of Tew and Guzdial [16] and its successor, the "Second CS1 assessment" (SCS1) of Parker et al. [11] are validated instruments that would be natural to consider; but Bockmon and Bourke [4] found them too difficult for incoming university freshmen, and so combined SCS1 with the the "Computational Thinking Concepts and Skills Test" of Peteranetz et al. [13] in order to use it as a placement test for CS0 and CS1.

In the current paper, we analyze the results of the *Prior Informatics Knowledge Assessment* (PIKA) [5]. The current version of the test is created by a consortium of CS1 educators in Norway and Sweden, and development is still ongoing [6]. Rather than emphasizing computational thinking in an abstract sense, this test focus on programming knowledge in Python, the *de facto* primary programming language being used in upper secondary mathematics courses in Norway. The test consists of code tracing tasks divided into areas of data types, operations, variables, boolean expressions, conditionals, loops, lists and functions. The test is intended to be simple enough to meaningfully distinguish between students that have progressed fairly short into the content traditionally found in a CS1 course.

2.3 Programming in Norwegian K-12 Education

The Norwegian educational reform LK20 integrated programming and CT into various K-12 subjects. On the upper secondary level (grades 11–13) the curriculum change was gradually introduced, starting with students that began their upper secondary school in fall 2020. This class of students graduated in the summer of 2023.

In elementary and lower secondary school (grades 1–10), the curriculum change happened instantaneously for all grades in 2020. Hence, students that graduated K-12 in 2024 would have experienced *four* years of the new curriculum. Every year going forward (until 2032), the graduating students will have had one more year of experience with CT in the curriculum compared to the year before them.

The LK20 reform does not have a separate course for programming or CT in any grade (except electives), but primarily integrates it in mathematics. Text-based programming in Python is used for modeling and problem-solving alongside traditional mathematical concepts.

All programs in Norwegian upper secondary school has at least two years of mathematics, and it is also possible to chose an advanced course in the third year. Figure 1 shows the progression of these courses, ranging from foundational to advanced levels.

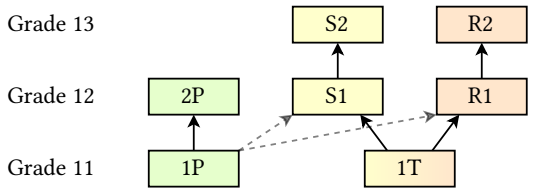

Figure 1: Norwegian upper secondary mathematics courses. Full lines are the recommended course progressions. Dotted lines are possible progressions in special circumstances. The courses from least to most advanced are: practical (P), social science (S) and natural science (R) mathematics. With respect to higher education admission requirements, R1 and S1+S2 are usually considered equivalent.

For enrollment in higher education it is typical that programs in engineering, architecture and natural sciences require their students to do the most advanced course (R2), especially the prestigious ones. Students' choice of mathematics course can hence be seen as a proxy for ability and ambition.

In addition to mathematics, students in upper secondary may have access to elective courses focused on computing, provided their school offers them. These courses align with the content typically taught in CS1 and related first-year university courses:

- *IT1*: Covers programming, data representation, web development, and the societal role of technology.
- *IT2*: Focuses on program design, debugging, object-oriented programming, and data analysis.
- *Programming and Modeling X* (PMX): Explores mathematical and scientific modeling of dynamic systems.

IT1, IT2 and PMX were available before the reform, so earlier graduates may also have programming experience through these courses.

3 Methodology

3.1 Study Design

To evaluate the impact of the curricular reform LK20, we analyzed data from PIKA [5]. This dataset included student responses to a variety of programming tasks and information about their educational and personal backgrounds.

3.1.1 Test Administration. The test was administered in August 2024 to $n = 3,038$ students that were about to begin their CS1 studies in one of eight different higher education institutions in Norway. The method of administration varied slightly between institutions, but was in every case administrated in the first week of the CS1 course or before the lectures had started. In some institutions the test was administrated in a supervised setting for all students, whereas in others it was shared as a voluntary activity to do at home.

3.1.2 Statistical Analysis. Since test scores are not normally distributed, we use Mann-Whitney U tests to see whether the scores of two student groups are significantly different. This ignores the point score as a numeric value, and instead considers the *ranking* of students according to their score.

When reporting the results from the Mann-Whitney U test, a measure of central tendency is provided for each group. We use the Hodges-Lehmann estimator for this purpose. This value is also called the pseudo-median, and is defined as the median of all pairwise averages.

We report the effect sizes of Mann-Whitney U using rank-biserial correlation (RBC). Given a reference group X and a treatment group Y, this value describes the proportion of pairwise contests that is won by the treatment group, normalized to a number between -1.0 and 1.0 (respectively representing that every member in X won over every member in Y and vice versa). A value of 0.0 indicates that there are equally many victories to members of X as there are to members of Y. Perhaps a more intuitive interpretation of the RBC is the following: add 1 to the RBC and then divide by 2; the resulting value indicates the probability that a randomly selected member of group Y does better than a randomly selected member of group X. Confidence intervals for the RBC is found by applying bootstrapping with 1000 repetitions.

To maintain statistical validity and reliability, analyses were restricted to student groups with sample sizes of $n > 30$.

3.2 RQ1: Difference after the Curricular Reform

To investigate the effect LK20 has had on students' prior knowledge for CS1 we compare two main groups:

- *new* students—those who graduated upper secondary school in 2023 or 2024 and were thus subject to the LK20 curriculum; and
- *old* students—those who graduated prior to 2023 and followed the previous curriculum.

A higher score among the new students would suggest a positive effect of LK20.

3.3 RQ2: Difference for Specific Student Groups

To investigate closer what effect the curricular reform has had we analyzed specific subsets of our dataset based on the students' educational background and gender. Educational background includes:

(1) mathematics courses completed,
(2) elective computer science courses taken, and
(3) programming experience outside formal education.

We remark that educational background and programming experience outside formal education are post-treatment variables, meaning they could potentially be influenced by the curriculum reform itself. We must therefore be very careful when we interpret findings for groups that are restricted by these variables, since the group composition itself could be influenced by the very factor we attempt to investigate the effect of.

3.4 RQ3: Difference Before and After CS1

To contextualize the observed effect sizes of LK20, we also administrated the test to CS1 students at University of Bergen one week before their final exam ($n = 85$). Participation was voluntary, and the only incentive offered was that the test was part of a "crash course" aimed at helping students prepare for the exam.

By providing a measure of central tendency of this group, we can make a rough comparison between the effects of the LK20 curriculum and completing a typical CS1 course in higher education.

3.5 Demographics

Our initial sample of test responses included 3,038 CS1 students but was narrowed down to 2,661 students who had no university-level programming experience.

Students were asked to provide information about their educational background and gender. Note that response rates varied across questions, leading to slight differences in the sample sizes reported in this section.

3.5.1 Graduation Years. New students made up 50.4% of our dataset ($n = 2,437$), while the remaining 49.6% followed the old curriculum.

3.5.2 Mathematics Courses. Figure 2 shows the distribution of students' most advanced mathematics courses, with the vast majority completing R2. This is likely because many study programs that include CS1 require advanced mathematics for admission.

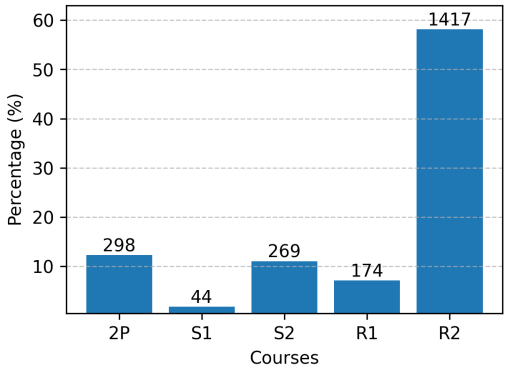

Figure 2: Distribution of students' most advanced mathematics courses ($n = 2,202$).

3.5.3 Elective IT Courses. Most CS1 students do not take IT electives in secondary school, however the proportion that completes IT1 and IT2 is not negligible. Figure 3 presents the distribution of students who took these three electives.

3.5.4 Programming Experience Outside School. Students self-reported their hours of experience with text-based programming languages, such as Python, Java, and C++, acquired outside formal education. A total of 20.9% of students ($n = 2,653$) reported having at least 30 hours of such experience.

3.5.5 Gender. The test was taken predominantly by male students: 58.1% (n=2,661). Women constituted 36.0% of the dataset, while 5.9% identified outside of these genders or did not answer.

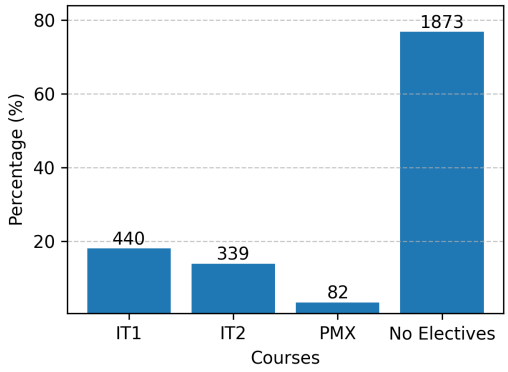

Figure 3: Distribution of students completing elective IT courses ($n = 2,661$).

4 Results

4.1 RQ1: The Effect of LK20

We initially look at the entire student body (excluding students that already has completed a university-level course in programming) and group them by which curriculum they experienced in upper secondary school. The new students had a pseudo-median score of 49.8 ($n = 1,228$), while the old had a score of 25.9 ($n = 1,209$). A Mann-Whitney U test revealed a statistically significant difference in the distribution of scores between the two groups ($p = 6.0 \cdot 10^{-22}$). The effect size, which we report as a 95% confidence interval for the rank-biserial correlation, is $[0.39, 0.47]$. This indicates that a randomly selected student exposed to LK20 has a 70.5% to 73.5% chance of doing better than a randomly selected student exposed to the old curriculum. Figure 4 displays the difference in score distributions. We also find this result in the first row of Table 1.

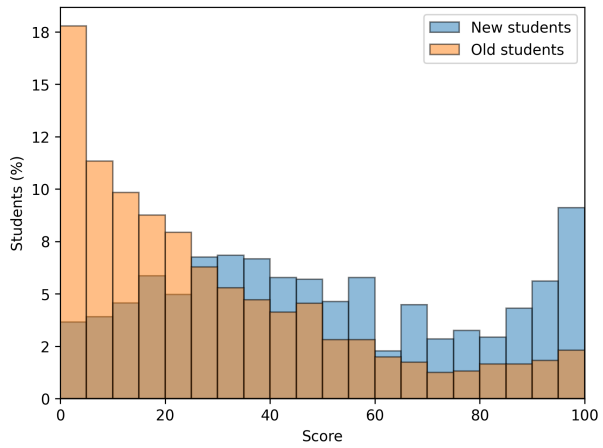

Figure 4: Distribution test scores of old ($n = 1,209$) and new ($n = 1,228$) students.

4.2 Gap Year Effects

Students who take gap years—delaying their entry into higher education after completing upper secondary school, for example due to compulsory military service or retaking exams to improve their grades—are a different group of students than those who enroll in university directly after completing upper secondary school. It might be that the difference we see between old and new students is due to the effects of gap years rather than due to LK20.

Ideally we should have administered an identical test in fall 2022, before the LK20 reform had taken effect, and done a comparison with those results. Unfortunately time travel is not possible. However, that should not dissuade us from doing the best we can with the data we have; the data is seldom perfect when analyzing historical events.

In order to exclude the effect of gap years, we turn to data we have collected for an older (and sligthly different) version of PIKA, which was administrated in fall 2023 and in fall 2024 at the University of Bergen. We analyze the results of this test with these specific groups in mind:

- *new* gap students ($n = 78$)—students who graduated from upper secondary school in 2023 and took the test in 2024, reflecting one gap year under the LK20 curriculum; and
- *old* gap students ($n = 156$)—students who graduated from upper secondary school in 2022 and took the test in 2023, reflecting one gap year under the old curriculum.

If the new gap students perform better than the old gap students, this would suggest a positive effect of the LK20 curriculum, independent of the gap year effects.

A comparison between these groups with a Mann-Whitney U test revealed a significant difference in their score distributions ($p = 1.6 \cdot 10^{-5}$). New gap students achieved a pseudo-median score of 46.9, whereas old gap students scored 34.1. The 95% confidence interval for the rank-biserial correlation was $[0.19, 0.48]$. Even though the scores in this test is not comparable to that of our main analysis (the questions on the tests are different), the effect size *is* comparable; assuming that both tests are ranking students according to their programming abilities.

We notice that the effect size of LK20 (as given by the RBC 95% confidence interval) for one-year gap students tends to be slightly on the lower side compared to the effect size we find in the primary analysis, despite their confidence intervals fully overlapping (recall that the effect size we found there was $[0.39, 0.47]$). It may therefore be that part of the effect we see in our primary analysis is due to gap years. Even so, we find that LK20 has *some* effect independent of gap years.

4.3 RQ2: A Finer Look

The impact of the curricular reform varied depending on the students' backgrounds. Table 1 presents the pseudo-median scores and effect size confidence intervals for different groups categorized by educational background, programming experience outside formal education and gender. The groups we display in the table are selected in order that we can compare how LK20 works differently for

- students who had different mathematics courses,

Table 1: Comparing old and new students on their test performance, restricted to those with the same background in mathematics, IT electives, experience with programming outside school, and gender. "Score" denotes the pseudo-median of the test scores for each group. The p-value is from the Mann-Whitney U test, and its effect size is found under "RBC CI", which represents the 95% confidence interval for rank-biserial correlation. An asterisk (∗) indicates that all values of this attribute are included, and a hyphen (−) indicates that only those with a blank value on this attribute are included.

Group				Old		New			RBC CI	
Math	Electives	Experience	Gender	Score	n	Score	n	p-value	Lower	Upper
∗	∗	∗	∗	25.9	1209	49.8	1228	$6.0 \cdot 10^{-22}$	0.39	0.47
2P	-	No	∗	14.0	149	11.6	75	$2.5 \cdot 10^{-1}$	-0.18	0.11
S2	-	No	∗	15.5	100	26.4	71	$1.1 \cdot 10^{-5}$	0.23	0.54
R1	-	No	∗	19.2	80	31.3	40	$5.9 \cdot 10^{-4}$	0.19	0.57
R2	-	No	∗	19.0	339	42.4	515	$3.2 \cdot 10^{-45}$	0.50	0.64
R2	IT1	No	∗	35.2	74	71.4	132	$3.7 \cdot 10^{-15}$	0.54	0.77
R2	IT2	No	∗	42.1	53	80.5	94	$1.0 \cdot 10^{-11}$	0.54	0.80
R2	-	Yes	∗	56.7	59	72.0	84	$1.7 \cdot 10^{-3}$	0.12	0.49
R2	IT1	Yes	∗	76.0	32	91.6	79	$6.0 \cdot 10^{-4}$	0.19	0.62
∗	∗	∗	m	33.0	690	56.4	713	$6.7 \cdot 10^{-39}$	0.35	0.46
∗	∗	∗	f	15.3	452	37.7	449	$1.6 \cdot 10^{-41}$	0.45	0.58
R2	-	No	m	24.6	169	44.8	270	$9.3 \cdot 10^{-18}$	0.39	0.59
R2	-	No	f	12.6	156	38.9	219	$1.7 \cdot 10^{-30}$	0.60	0.78

- students that did electives in IT and/or have programming experience outside of formal education, and
- students of different genders.

Note that certain groups are not included in our table due to their insufficient sample sizes ($n < 30$).

In summary, the reform appears to have had highest effect on students who completed advanced math courses, while those with the least advanced mathematics (2P) showed no improvement. Students that choose IT electives seem to benefit a lot under the new curriculum; perhaps the introduction of programming in their regular mathematics classes yields synergy effects. Regarding gender, the reform had a greater impact on women than men, particularly among women who took R2 mathematics.

4.4 RQ3: Programming Knowledge Before and After CS1

To assess programming knowledge after completing the CS1 course and compare it with K-12 graduates, we analyzed test scores from a group of $n = 44$ students that did not experience LK20, and who took the test both before the semester began (pre-scores) and one week before the final exam (post-scores). Students received no feedback on their initial performance and had no access to the test questions or answers between administrations.

The pre-score pseudo-median was 33.3 and the post-score pseudo-median was 83.7, which (unsurprisingly) is a significantly higher score (Mann-Whitney U test, $p = 1.5 \cdot 10^{-13}$). The effect size given as a 95% confidence interval for the rank-biserial correlation is $[0.79, 0.99]$, indicating a very strong effect of CS1 on student performance. We remark that this effect size is strictly stronger than the effect of LK20 on every group we observe; and the pseudo-median score is also higher than every group except for those with programming experience outside school that also did advanced mathematics and had IT electives under the new curriculum.

5 Discussion

5.1 RQ1: The Effect of the Curricular Reform

Our analysis indicates that students who completed their education under the new curriculum (LK20) perform better on programming tasks compared to those who graduated before its implementation. The new students had a pseudo-median score of 49.8, while the old had 25.9.

While the presence of gap years *may* have influenced the results, our analysis suggests that this at least does not explain the entire effect. Our analysis does not rule out the possibility that the effect of gap years is negligible. The improvement in scores remains substantial even when comparing students with similar gap years.

Our results suggests that the curricular reform has had a positive impact on the programming knowledge of incoming higher education students studying CS1 in Norway.

5.2 RQ2: Has the Reform Affected Students Differently?

The curricular reform had varying effects across student groups, with most showing statistically significant improvements. However, some groups saw no gains, highlighting disparities in its impact.

The most significant improvement was observed among students who completed the most advanced mathematics (R2) and IT2, with their pseudo-median scores increasing from 42.1 to 80.5, with an effect size of $[0.54, 0.80]$.

Notably, students with IT electives — who had also studied programming under the previous curriculum — showed a substantial effect size. This suggests a possible synergy when programming is integrated into both IT courses and regular mathematics instruction.

It is worth noting, however, that students who choose to take IT2 often demonstrate a strong interest in programming. In contrast, the CS1 courses analyzed in this study includes a more diverse group of students, many of whom are from non-CS disciplines where CS1 is a compulsory requirement. This probably contributes to the observed disparities in performance outcomes.

The group that benefited the least from the reform comprised students enrolled in the most basic mathematics course, 2P. No improvements were observed for this group. This indicates that the gap in prior knowledge for CS1 *widens* with respect to mathematics background under LK20.

From a gender perspective, the reform appears to have been more beneficial for women, helping to reduce the traditionally observed gender gap in computer science education. The observed effect size for women $[0.45, 0.58]$ was higher than that for men $[0.35, 0.46]$, indicating a stronger positive effect of the reform for women. Among women enrolled in advanced mathematics (R2), the effect was even more pronounced. Thus, LK20 *narrows* the gap in prior knowledge for CS1 with respect to gender. Men still scored higher than women, suggesting that while the reform narrowed the gender gap, it did not eliminate it.

5.3 RQ3: Difference Before and After CS1

While the pseudo-median scores of most student groups have increased under LK20, they still fall short of the post-CS1 score of 83.7. The only exception is students who completed both R2 and IT2 under the new curriculum or those with outside experience combined with R2 and IT1, who achieved similar scores. This suggests comparable learning outcomes within these groups for the topics covered in our assessment.

That being said, the intended learning outcomes in CS1 extend beyond the tasks assessed in PIKA. Even a perfect score of 100 points is no proof that the CS1 curriculum is fully mastered. While some students from the highest performing groups may be prepared to skip CS1 and move directly to CS2, we believe that most would still benefit from further exposure to CS1 material to reinforce their foundational skills.

Our interpretation is that the majority of students still require CS1 instruction before venturing further into more advanced coursework. However, it appears that a traditional CS1 course—starting with basic concepts of variable assignment, data types and conditionals—is no longer the most suitable starting point for many students; for mid- to high-performing students that have been exposed to the advanced mathematics course (R2) under the new curriculum, it would likely be more beneficial to start with more advanced material or at least progress faster through the initial basic concepts.

In light of the increased programming knowledge among incoming students, Norwegian CS1 instructors are considering adjustments to the curriculum to better align with students' prior knowledge. The findings of this study provide valuable insights for optimizing student placement. By considering students' secondary school course selections—such as advanced mathematics and IT electives—institutions can tailor their CS1 offerings to build more effectively on students' existing knowledge, creating more personalized and effective learning paths.

6 Conclusion

Our analysis of PIKA shows that the curricular reform LK20 has significantly enhanced the programming knowledge of incoming higher education students. While there is clear overall progress, the reform's impact has varied across different student groups. Furthermore, the increase in programming knowledge, while meaningful, remains moderate, with many students showing only a partial grasp of the CS1 fundamentals.

Despite these limitations, the results are promising for the future of Norwegian CS1 education. The 2023 and 2024 cohorts have had exposure to the new curriculum for just three and four years, respectively. As time progresses, this exposure to programming and computational thinking will only increase, reaching its peak in 2032, when students will have benefited from programming instruction throughout their entire K-12 education. Additionally, as teachers gain more experience and refine their methods for delivering computer science instruction, the overall quality of K-12 programming education is expected to improve.

We anticipate that these factors will further enhance the programming knowledge of future cohorts, leading to higher test scores and better preparation for CS1 courses. To monitor these trends and track developments in programming knowledge, PIKA will continue to be administered annually.

References

[1] Ana Liz Souto O. Araujo, Jucelio S. Santos, Wilkerson L. Andrade, Dalton D. Serey Guerrero, and Valentina Dagienė. 2017. Exploring computational thinking assessment in introductory programming courses. In *2017 IEEE Frontiers in Education Conference (FIE)*. 1–9. https://doi.org/10.1109/FIE.2017.8190652

[2] Tim Bell, Peter Andreae, and Anthony Robins. 2014. A Case Study of the Introduction of Computer Science in NZ Schools. *ACM Trans. Comput. Educ.* 14, 2, Article 10 (June 2014), 31 pages. https://doi.org/10.1145/2602485

[3] Stafania Boccani, Augusto Chioccariello, and Jeffrey Earp. 2018. *The Nordic approach to introducing Computational Thinking and programming in compulsory education.* Report prepared for the Nordic@BETT2018 Steering Group. https://doi.org/10.17471/54007

[4] Ryan Bockmon and Chris Bourke. 2023. Validation of the Placement Skill Inventory: A CS0/CS1 Placement Exam. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 39–45. https://doi.org/10.1145/3545945.3569738

[5] Sondre Bolland, Andreas Haraldsrud, Siri Jensen, Filip Strömbäck, Arne Styve, Erlend Tøssebro, Eirik Valseth, and Torstein J. F. Strømme. 2024. The Nordic Prior Knowledge Test in Programming: Motivation, Development and Preliminary Results. *Norsk IKT-konferanse for forskning og utdanning* 36, 4 (Nov. 2024). https://doi.org/10.5324/nikt.6216

[6] Sondre Bolland, Aleksandr Popov, Tyra Eide, Robert Kordts, and Torstein J. F. Strømme. 2024. Fundamentals of Norwegian CS1. *Norsk IKT-konferanse for forskning og utdanning* 4 (Nov. 2024). https://doi.org/10.5324/nikt.6217

[7] Lindsey Gouws, Karen Bradshaw, and Peter Wentworth. 2013. First year student performance in a test for computational thinking. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference* (East London, South Africa) *(SAICSIT '13)*. Association for Computing Machinery, New York, NY, USA, 271–277. https://doi.org/10.1145/2513456.2513484

[8] Peter Hubwieser. 2012. Computer Science Education in Secondary Schools – The Introduction of a New Compulsory Subject. *ACM Trans. Comput. Educ.* 12, 4, Article 16 (nov 2012), 41 pages. https://doi.org/10.1145/2382564.2382568

[9] Greg Lee, Yi-Ling Wu, and Jia-Yi Chen. 2024. Effect of New Computing Curriculum: Results from Incoming High School Freshmen. In *Informatics in Schools. Innovative Approaches to Computer Science Teaching and Learning*, Zsuzsa Pluhár

and Bence Gaál (Eds.). Springer Nature Switzerland, Cham, 18–29. https://doi.org/10.1007/978-3-031-73474-8_2

[10] Norwegian Directorate for Education and Training. 2021. Kunnskapsløftet 2020 – hvorfor har vi fått nye læreplaner? https://www.udir.no/laring-og-trivsel/lareplanverket/fagfornyelsen/hvorfor-nye-lareplaner Accessed: 2024-09-04.

[11] Miranda C. Parker, Mark Guzdial, and Shelly Engleman. 2016. Replication, Validation, and Use of a Language Independent CS1 Knowledge Assessment. In *Proceedings of the 2016 ACM Conference on International Computing Education Research* (Melbourne, VIC, Australia) *(ICER '16)*. Association for Computing Machinery, New York, NY, USA, 93–101. https://doi.org/10.1145/2960310.2960316

[12] Markeya S. Peteranetz and Anthony D. Albano. 2020. Development and evaluation of the Nebraska Assessment of Computing Knowledge. *Frontiers in Computer Science* 2 (2020), 11. https://doi.org/10.3389/fcomp.2020.00011

[13] Markeya S. Peteranetz, Patrick M. Morrow, and Leen-Kiat Soh. 2020. Development and Validation of the Computational Thinking Concepts and Skills Test. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 926–932. https://doi.org/10.1145/3328778.3366813

[14] Marcos Román-González, Juan-Carlos Pérez-González, and Carmen Jiménez-Fernández. 2017. Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior* 72 (2017), 678–691. https://doi.org/10.1016/j.chb.2016.08.047

[15] Swedish National Agency for Education. 2018. Curriculum for the compulsory school, preschool class and school-age educare. Retrieved December 10, 2024 from https://www.skolverket.se/publikationer?id=13128

[16] Allison Elliott Tew and Mark Guzdial. 2011. The FCS1: a language independent assessment of CS1 knowledge. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (Dallas, TX, USA) *(SIGCSE '11)*. Association for Computing Machinery, New York, NY, USA, 111–116. https://doi.org/10.1145/1953163.1953200

[17] UK Department for Education. 2014. The national curriculum in England: framework document. https://web.archive.org/web/20240104161617/https://assets.publishing.service.gov.uk/media/5a7db9e9e5274a5eaea65f58/Master_final_national_curriculum_28_Nov.pdf

[18] David Weintrop and Uri Wilensky. 2015. Using Commutative Assessments to Compare Conceptual Understanding in Blocks-based and Text-based Programs. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research* (Omaha, Nebraska, USA) *(ICER '15)*. Association for Computing Machinery, New York, NY, USA, 101–110. https://doi.org/10.1145/2787622.2787721

[19] Linda Werner, Jill Denner, Shannon Campe, and Damon Chizuru Kawamoto. 2012. The fairy performance assessment: measuring computational thinking in middle school. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (Raleigh, North Carolina, USA) *(SIGCSE '12)*. Association for Computing Machinery, New York, NY, USA, 215–220. https://doi.org/10.1145/2157136.2157200

Received 01 January 2099; revised 01 January 2099; accepted 01 January 2099