Computer programming skills: A cognitive perspective

Graafsma, Irene

DOI:
10.33612/diss.168003240

Chapter 2

Validating two short versions of the Second Computer Science 1 programming ability test[1]

[1] This chapter has been submitted as an article for publication.

2.1 INTRODUCTION

The use of digital technology in daily life has seen a large increase over the last two decades. As a result, both current and future generations need to be able to use digital technology (Rushkoff, 2012). In addition, a substantial proportion of the future workforce will need training to work in fields that develop these technologies (Bailey & Mitchell, 2006; Seegerer et al., 2019). Because of these changes, programming has gained major interest and importance in educational systems worldwide (Balanskat & Engelhardt, 2015). As we see more careers requiring programming skills at the tertiary level, more students choose to enrol in Computer Science degrees, and more students from other fields choose to undertake additional computer programming courses (European Commission, 2018; Marasco et al., 2017).

Although there has been an increase in demand for programming education, little is understood about how programming skills are best acquired and what characterises optimal training. To answer such questions, we need to combine knowledge and expertise from computer science education with other disciplines such as cognitive science and psychology (Guzdial, 2015). A prerequisite to researching the acquisition of programming skills is a validated instrument to measure programming skill that allows for comparisons between different research studies and that can measure change within a study (Parker et al., 2016). The present research aims to contribute to the development of such an instrument.

Aiming to measure programming skill in undergraduate novice programmers, Parker et al. (2016) developed the Second Computer Science 1 (SCS1) assessment. To date, two studies (Parker et al., 2016; Xie, Davidson et al., 2019) have investigated the difficulty and the reliability of the SCS1 test items. However, there are still limitations to the current SCS1 test and its validation. 1) The test takes an hour to administer, limiting its use in many research studies with time constraints. 2) It does not offer multiple versions to allow for repeated testing without introducing possible test-retest effects. This restricts the test's usefulness for research studies with an interest in measuring improvement over time. 3) No study to date has examined the SCS1's external validity. This means that we don't know how performance on the SCS1 relates to performance on other tests of programming ability, such as course exams. 4) The difficulty and reliability of the SCS1 were tested only

using American undergraduate students, and at only two institutions (Georgia Institute of Technology university and the University of Washington). As a result, it is still unknown whether the SCS1 performs similarly for other student populations, limiting its use for researchers at other institutes.

The aim of the current study is to address these four limitations by creating and validating two parallel short versions of the SCS1, the SCS1-Short Version 1 (SCS1-S1) and Version 2 (SCS1-S2). By doing so, we aim to address the first and second limitation by reducing testing time and allowing for repeated testing. We address the third limitation by testing the SCS1's external validity for the first time, by correlating scores on the short versions, and for individual items, with performance on programming assessments in a programming course. Lastly, we will assess the difficulty and internal-consistency of the short versions, thereby addressing the fourth limitation by generalizing knowledge on test and item quality to a new student population. In the sections of the Introduction that follow, we further elaborate on the characteristics of the SCS1 and on each of these four limitations.

2.1.1 Second Computer Science 1 (SCS1) as an assessment tool

The SCS1 (Parker et al., 2016) was developed as a parallel version of the Foundational Computer Science 1 (FCS1) Assessment (Parker et al., 2016; Tew, 2010; Tew & Guzdial, 2011). The FCS1 is no longer publicly available. However, Parker et al. (2016) developed a very similar test in the SCS1, consisting of the same topics and question types, but with different content.

The SCS1 consists of 27 multiple choice questions requiring students to read, evaluate and complete pieces of code in a pseudo-code programming language. Although it is impossible to develop a test that is fully independent from specific programming languages, by using a pseudo-language (Guzdial, 2019), Parker et al. (2016) aimed to make it as independent as possible from the main languages in undergraduate computer science. The SCS1 questions were designed to measure 9 different programming content areas (basics - i.e., applying simple mathematical formulae -, for-loops, while-loops, if-statements, logical operators, arrays, function parameters, function return values,

recursion) with three different question types (definitional, code tracing, code completion) each with five answer choices. In addition, students receive a two-page pseudo-code guide, which they were instructed to use as a reference when answering the questions.

2.1.2 Need for short versions

Behavioural research studies often require a number of different experimental measures. However, total testing time is usually constrained because of limits in both participant attention span and research funding. This means that using an assessment that is as long as the SCS1 is often not feasible. One way to deal with test length is to use a subset of items. Although Parker et al. (2018) did this in a later study, the subset used in that study was not validated separately. This means that the difficulty, validity and reliability of that subset are unknown. Consequently, there remains a need for a validated, short assessment of programming skill.

2.1.3 Need for parallel versions

Additionally, an important goal in any field of education is to be able to measure progress on a certain skill (Marston et al., 1986), for example, in the context of comparing different courses or interventions. One recent study by Nelson et al. (2017) attempted to measure change in programming skill by administering the SCS1 before and after a one-day course where participants were randomly assigned to two conditions with different course structures. This design aimed to compare improvement across conditions. However, because they used the same items twice, their conclusions are confounded with test-retest (practice) effects. This means that the true improvement as a result of change in underlying skill is likely lower than reported, as the participants could have benefitted from attempting the same questions for a second time. One way to minimise test-retest effects is through the use of two parallel versions of a (programming) test, allowing researchers to test students twice without using the same questions. Critically, parallel versions of a test should be equal in difficulty and measure the same knowledge.

2.1.4 Lack of evidence for external validity

It is essential that any test assesses what it is supposed to be measuring: the SCS1 should

reflect the programming skill of novice computing students. Moreover, the skills measured should reflect a broad scope of programming knowledge, covering all central topics. When Parker et al. (2016) validated the SCS1 with students from three different introductory programming courses, they did not report the correlations between the SCS1 and course grades, but did report these correlations for the FCS1, on which they based their SCS1 questions. Scores on the FCS1 showed a significant correlation of $r = .483$ ($p < .001$) with grades on a Python course for students with Computer Science majors ($n = 140$), but no significant correlations with grades on a MATLAB course ($r = .509$, $p = .076$) for a small sample of students with Engineering majors ($n = 13$), or grades of students undertaking a media computation course ($n = 30$) ($r = .298$, $p = .110$). It is possible that these correlations did not reach significance because of the small sample sizes. Parker et al. (2016) did find that the total scores on the SCS1 correlated with the total scores on the earlier FCS1 Assessment. However, this correlation was only moderate ($r = .57$), which was surprising, as the questions of the SCS1 were modelled on the questions of the FCS1.

Xie, Davidson et al. (2019) tested the relationship between the different questions of the SCS1 using confirmatory factor analysis. They showed that 24 of the 27 questions loaded on one latent variable, presumably programming skill. However, questions 20, 24 and 27 did not load on this variable and therefore seem to measure something else. Hence, while the majority of SCS1 questions seem to measure related concepts, it remains unclear to what extent those concepts are programming knowledge specifically. That is, the extent to which the test has external validity.

2.1.5 Reliability and difficulty

Both Parker et al. (2016) and Xie, Davidson et al. (2019) tested the internal-consistency and reliability of the SCS1 by calculating Cronbach's alpha. Parker et al. (2016) found moderate internal-consistency and reliability (Cronbach's alpha = .59), but this was higher in Xie, Davidson et al. (2019) (Cronbach's alpha = .70). This suggests that the internal-consistency and reliability of the test might vary across different student populations. To further determine this, the internal-consistency and reliability needs to be tested in more different courses.

In addition to being reliable, an ideal test of programming skill would also be of appropriate difficulty. This means that it should not be too easy or too difficult overall for the target group, and that it should have items of varying levels of difficulty to discriminate between learners with varying knowledge (Xie, Davidson et al., 2019). In their initial validation study, Parker et al. (2016) found that most questions were difficult for beginning programmers. For 22 of the 27 questions, less than half of the students answered correctly, and on the remaining five questions, 51-85% of students answered correctly. Although this showed that many questions were difficult, the chance level would lie at 20 percent correct for questions with five multiple choice answer options. It, therefore, seems that, even on these harder questions, students still performed above chance and that although the SCS1 seems to be rather difficult, it can be effectively used to distinguish between students of different programming ability.

Xie, Davidson et al. (2019) used item response theory for a more fine-grained analysis of the difference between item difficulty and learner knowledge. They tested students with a wider range of programming ability than Parker et al. (2016), by adding a sample of more proficient programmers, therefore generalizing to a population beyond beginner programmers. They found that only four items (Qs 5, 13, 15 and 18) were particularly difficult for their beginner programmers, with only 12-21% of students answering those questions correctly, thus performing at chance level. For the students with more programming experience two of these questions remained difficult (Q13 and 15), while performance on the other two questions (Q5 and 18) increased to 30 and 31% respectively. These results seem to replicate Parker et al.'s (2016) finding that the SCS1 is difficult for beginning programmers, but that most questions are still answered correctly at rates greater than chance. However, the results from Xie, Davidson et al. (2019) cannot be directly compared to Parker et al. (2016). Firstly, Xie, Davidson et al. (2019) excluded participants who attempted fewer than 10 of the 27 questions. It is therefore likely that the weaker programmers were not included in the analysis, thereby making the questions seem easier than if all students were included in the analysis. This might also have led to the higher internal-consistency and reliability. Secondly, Xie, Davidson et al. (2019) used the SCS1 as a pre-test at the start of a programming course, rather than as a post-test at the end. This means that performance was less dependent on the course content, and more

dependent on previous programming experience. The fact that they found differences in performance between beginners and more advanced programmers suggests that the test successfully discriminates between different levels of programming skill. However, it is possible that in the assessment of basic programming skill in true novices the SCS1 is not very useful because of the overall difficulty of the test: It might be better employed after some initial programming education. To further determine the difficulty of the tests and individual items they need to be administered to a wider variety of students with different backgrounds.

In sum, the current study aims to address the four limitations of the SCS1 that we have described. By developing and validating two short versions of the SCS1 in a new student population and by correlating performance with course grades we aim to achieve 1) a shorter testing time, 2) the possibility for repeated testing, 3) knowledge on the external validity of the tests and items, and 4) generalization of knowledge on test and item quality to a student population outside Georgia Institute of Technology. We will compare our findings to those of Parker et al. (2016). However, we will refrain from making a direct comparison to the Xie, Davidson et al.'s (2019) data, as they excluded participants who answered fewer than 10 questions and administered the SCS1 at the start rather than at the end of the course.

2.2 METHODS

2.2.1 Participants

As part of a mandatory tutorial, 668 students undertaking an undergraduate introduction to programming course at Macquarie University (link: https://osf.io/y9bkw/), completed the testing session at the end of their course. Of these participants, 415 (60%) consented for their data to be used for the current study. The students were enrolled in a wide variety of undergraduate majors, such as science, mathematics, business, and computing. As all our test instructions and many of the test items were in English, participants were excluded if they indicated that their level of English was lower than "Good", level 3, on a 5-point rating scale from 1 (minimal) to 5 (native) (16 participants excluded), or indicated that they

cheated or did not seriously attempt to answer the questions in a post-test probe question (48 participants). Thus, the results reported here were from the remaining 354 participants (68 female, 255 male, 31 other or unspecified; mean age 19.87 years, *SD* = 3.52). The study received ethical approval from the Macquarie University Human Research Ethics Committee (Reference number: 5201800224).

2.2.2 Materials

This study reports results that form part of data collected for a larger study, in the context of which we also collected data on other skills (e.g., tests of pattern recognition, logical reasoning and grammar learning skills). Participants were also administered two questionnaires: a sense of agency scale, measuring their feelings of control while programming, and behaviours and personality questionnaire that looks at autistic traits in the general population. Below we describe the two tests that were used for the study reported here: the two adapted SCS1 tests of programming skill and the tests completed by the students for their course grades.

Second Computer Science 1 (SCS1) test

The SCS1 (Parker et al., 2016) formed the basis of our assessment of programming skill. The SCS1 consists of the following content areas: basics (i.e., applying simple mathematical formulae), conditionals, for loops, indefinite/while loops, logical operators, arrays, recursion, function parameters, and function return values. We developed two short versions of the SCS1 by dividing the questions in such a way that both versions covered all content areas and all three question styles (definitional, code tracing, code completion). As the SCS1 has an uneven number of items, we removed one item to make two versions of equal length. Based on the first validation study by Parker et al. (2016) and our own pilot work, we dropped item 15, in the study by Parker et al. (2016) it was among the hardest items with poor discrimination, as less than 50 percent of students answered it correctly, and it showed a point-biserial correlation below 0.1. In our own pilot of 9 participants none answered this question correctly (Xie, Davidson et al., 2019 which was published after the start of our study found similar results for Q15). The division of items across versions resulted in combining the original items 1, 2, 5, 6, 9, 12, 13, 14, 17, 18, 22, 25, 27 into

Version 1, and items 3, 4, 7, 8, 10, 11, 16, 19, 20, 21, 23, 24, 26 into Version 2. Appendices 1 and 2 show the items assigned to each version with the level of difficulty as estimated according to the first validation of the SCS1 by Parker et al. (2016) and according to our pilot work. We kept the original pseudocode guide for participants to use as a guide while answering the questions. Each version was converted into an online format using Qualtrics (Qualtrics, Provo, UT), this was similar to the online format used by Xie, Davidson et al. (2019).

Student grades

These were the student's final course grades on the main course assessments of their university undergraduate programming course. These assessments consisted of six tests each comprising open questions where students were asked to answer conceptual questions or to solve small programming problems. Students could attempt each test three times throughout the course, but were assigned equivalent but different questions on each attempt. Their highest scores counted as their final grades and were used to compute the total grade for the current study. The tests could be completed throughout the semester as well as during the final exam testing session, which took place two weeks after the testing session with the SCS1-S. The six different tests addressed the topics of: fundamentals of computing, variables and conditionals, loops, functions, arrays and strings, program design and problem solving. For more information see the Unit Guide in the Cognition of Coding project on the Open Science Framework (https://osf.io/y9bkw/). We used the raw module scores from five of the six modules, averaged over the subtopics for our analysis and disregarded penalties for incomplete work or lack of attendance. We excluded grades from the module related to the fundamentals of computing, which focused mainly on the history of the field, and did not reflect programming skill as measured by the SCS1.

2.2.3 Procedure

During the last tutorial of the introductory programming course, students were given a link to the Qualtrics survey and were asked to complete the test individually. The Qualtrics survey started with information about the study and participants were asked for consent for their data to be used, and for their anonymised data to be made publicly available for

future research. Participants were then assigned SCS1-S1 or SCS1-S2 based on whether their student number was even or odd. They were informed that they had 30 minutes to complete the test as well as they could. During the test they could scroll back and forth through the questions and saw a clock with the remaining time in the corner of the screen. Skipping questions was allowed in this format. There was a button on the page that opened the pseudocode guide in a separate window, which they could use as a reference to help answer the questions. Students were informed that they were free to also make use of paper and pen to help them solve the problems. Students who missed their tutorials had the opportunity to complete the tests at home for partial attendance credit. 343 of the 354 participants used in analysis completed the experiment during the tutorial, with the remaining 11 completing it at home. Students completed their module tests that together made up their final course grades throughout the semester, with the final test opportunity during the scheduled exam time approximately two weeks after our experimental testing session. For all consenting students we obtained their grades on their module tests once these were finalised and had been released to the students.

2.2.4 Analysis

We ran four separate analyses to address the four aims of the current study. For all analyses we only included participants if they had answered at least one question on their SCS1-S version (167 participants for Version 1 and 170 participants for Version 2). To determine whether it was possible to develop reliable parallel short versions of the SCS1, the first analysis compared the difficulty of both versions by comparing accuracy on each version using a student's *t*-test and a Bayesian *t*-test. The Bayesian *t*-test allows examination of the strength of the evidence for the null hypothesis (there is no difference in the difficulty of the two versions) versus the alternative hypothesis (the two versions differ in difficulty). Bayes factors can be interpreted as a direct ratio and can therefore be interpreted without a cut-off criterion. However, typically the Bayes factors are interpreted so that Bayes factors between 0 and 0.33 show support for the null hypothesis, with lower values showing stronger support. Bayes factors between 0.33 and 3 are considered inconclusive. Bayes factors above 3 show support for the alternative hypothesis, with higher values showing stronger support (Rouder et al., 2009).

Secondly, to examine and compare the external content validity of each version we correlated accuracy on both versions with student's course grades on the programming course. We then compared the correlations across the two versions using Fisher's *z*-transformation.

To determine the reliability and difficulty of the versions, we analysed the properties of the individual items on both SCS1-S versions. We first used classical test theory to examine each item's difficulty by computing the percentage of students who answered each question correctly. Since most students were not able to complete all questions within the allotted 30 minutes, we used two different measures for item difficulty. Firstly, we considered accuracy per item when looking at all participants, for this we counted unanswered items as incorrect. Secondly, we looked at the accuracy per item whilst taking only participants who attempted that item into account, for this we counted unanswered items as missing values. We also considered discriminability of each item by computing point-biserial correlations between the item accuracy and the total score on their SCS1-S version, and with the course grade.

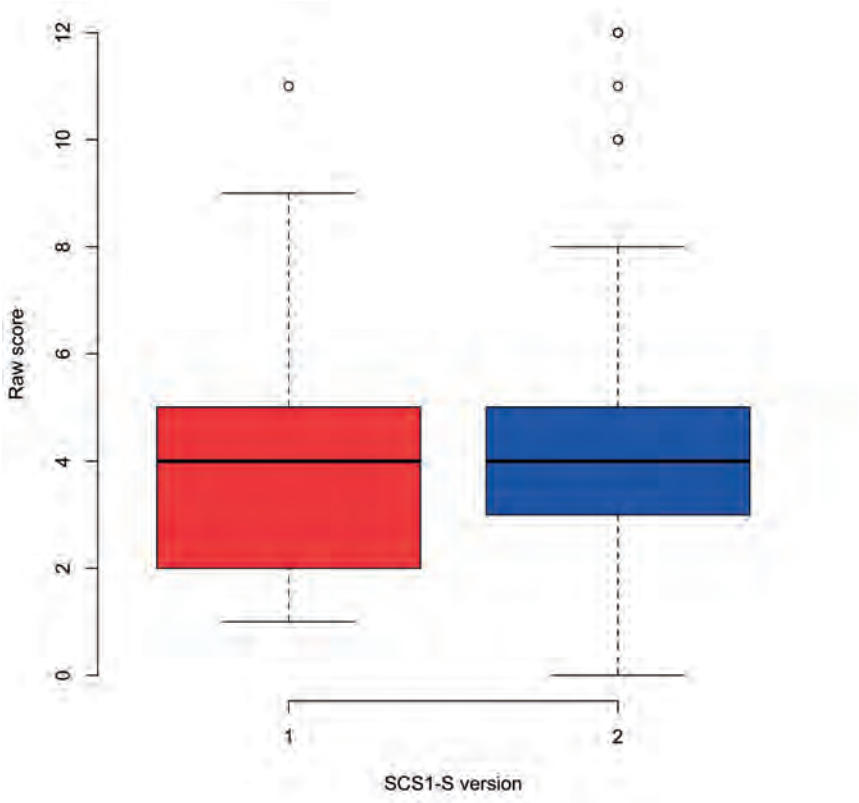
Finally, we used Item Response Theory (IRT) analysis to further examine the quality of the individual items. Mirroring the Xie, Davidson et al. (2019) analysis procedure, we started with a confirmatory factor analysis (CFA) to determine whether all questions loaded on the same factor, which would suggest that the test measured a single concept. Next, we computed Cronbach's alpha for each version separately to examine the internal-consistency and reliability. Finally, we conducted IRT analyses of the test items.

2.3 RESULTS

2.3.1 Test difficulty

On the two short versions of the SCS1-S, average accuracy was 29% correct for Version 1 (3.81 (*SD* = 1.88) out of 13), and 32% correct (4.18 (*SD* = 2.30) out of 13) for Version 2. The scores on the two versions did not differ significantly ($t(324.63) = -1.61$, $p = .11$; see Figure 2.1 for boxplots). However, a Bayesian t-test showed that the data favoured the alternative hypothesis that the scores on the two versions differ, but only to a modest (inconclusive) degree ($BF_{10} = 2.4$). In other words, although there was no significant difference between the scores on both versions, we cannot conclude that the versions were equal in difficulty for our cohort of participants.

Figure 2.1. Median raw scores and distributions per version of the SCS1-S.

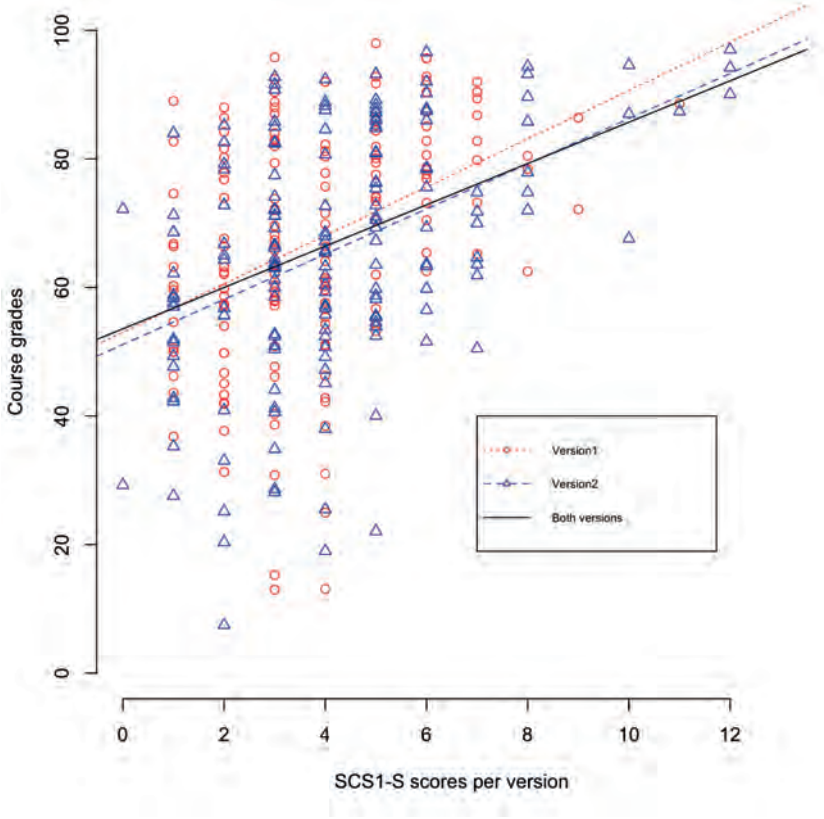


Note: Thirteen is the maximum score on both versions. The horizontal black lines indicate the median score.

2.3.2 External validity

To evaluate the external validity of each version of the SCS1-S, we computed individual correlations between the SCS1-S test scores and the exam grades. When correlating the scores on the SCS1-S with course grades across versions, results showed a moderate and significant correlation between SCS1-S scores and student exam marks ($r(329) = .41$, $p < .001$). For both versions separately we found modest and significant correlations between total scores on the version and exam grades (Version 1: $r(164) = .40$, $p < .001$; Version 2: $r(163) = .44$, $p < .001$). These correlations did not differ significantly between the versions ($z = -.47$, $p = .64$). Figure 2.2 depicts the scatter plots and lines of best fit for these data.

Figure 2.2. Scatter plot between raw SCS1-S scores and student grades on the course exam.



Note: Raw SCS1-S scores are plotted per version with lines of best fit per version and for both versions combined.

2.3.3 Internal validity: Quality of individual items

Below we report the analyses of the quality of the individual items using respectively CTT and IRT.

Classical test theory

Every question was completed by at least 83% of participants, with attempt rates being slightly lower for questions at the end of the versions because not all participants managed to complete the full test (see Table 2.1). We used a binomial model to calculate the critical accuracy rates that had to be reached for each item to conclude that participants performed significantly above chance at an alpha-level of .05 (one-tailed). The probability for the null binomial model was assumed to be .2, since each item had 5 response alternatives. The exact critical value differed for each item depending on the number of participants who answered the question, with the critical values ranging from 25.29% answered correctly for the items with a 100% response rate, to 25.71% answered correctly for Version 1 item 13 which had the lowest number of responses. For items 3, 7, 10 and 13 of Version 1, and items 2, 3 ,6 ,7 and 12 of Version 2 performance was poor and we cannot reject the possibility that participants were responding at or below chance. Performance on all other items was significantly above chance.

Table 2.1. Accuracy and response rates for the two versions of the SCS1-S.

	Version 1				Version 2			
Question number SCS1-S	Original question number on full SCS1	% of participants attempted	Accuracy (all)	Accuracy (attempted only)	Original question number on full SCS1	% of participants attempted	Accuracy (all)	Accuracy (attempted only)
1	1	100	27.54	27.54	3	100	51.76	51.76
2	2	95.21	35.33	37.11	4	100	20.00	20.00*
3	5	99.40	16.77	16.87*	7	95.29	21.76	22.84*
4	6	98.80	40.72	41.21	8	100	35.29	35.29
5	9	97.60	29.34	30.06	10	98.24	49.41	50.30
6	12	94.01	46.11	49.04	11	100	21.18	21.18*
7	13	94.01	20.36	21.66*	16	97.06	22.94	23.64
8	14	93.41	38.92	41.67	19	98.24	44.12	44.91
9	17	91.02	28.74	31.58	20	97.06	19.41	20.00*
10	18	86.23	13.77	15.97*	21	94.12	29.41	31.25
11	22	86.83	31.74	36.55	23	94.12	60.59	64.38
12	25	86.23	31.74	36.80	24	90.59	16.47	18.18*
13	27	83.83	19.76	23.57*	26	89.41	25.29	28.29

Note: For each version the first column shows the percentage of participants who attempted each question. The second column shows the percentages of all participants who answered a question correctly, with unanswered questions counted as incorrect. The third column shows the percentage of correct attempts per question, with unanswered questions treated as missing values. * indicates items where performance is not significantly above chance (Binomial test, $p > .05$).

We also examined the extent to which a participant's accuracy on each item was related to their accuracy on the entire test (see Table 2.2). Higher correlations indicate a stronger relationship between the item and the test total and indicate good internal validity for that item. In these analyses we counted unanswered items as incorrect. That is, all participants are included, regardless of whether or not they attempted the item. On Version 1 items 10 and 13 appeared problematic as they did not show significant correlations with the total score on the version. On Version 2 this was only the case for item 3.

For both versions we also tested the correlation of each individual item with the course grades (Table 2.2). On Version 1 seven questions correlated significantly with the scores on the unit exam (1,2,6,7,8,9 and 13). This was the case for eight questions on Version 2 (1,5,6,7,8,10,11 and 12).

Regarding the difficulty and discriminability of the individual items, Parker et al. (2016) did not report the exact values but rather categorised the items into three categories as seen in Table 2.3. In Table 2.3 we categorised the items using the same scheme as Parker et al. (2016) to allow for direct comparison. We used the original item numbers from the full SCS1 version to allow for easier comparison. Overall, the students in the current study seemed to perform more poorly than participants in the study by Parker et al. (2016). However, in the current study the items were generally more predictive of the final score on the SCS1-S version than in the study by Parker et al. (2016).

Item response theory

Below we first report the results of the verification of IRT assumptions, followed by the results of the IRT model. We used the same steps and models as Xie, Davidson et al. (2019) on our SCS1-S versions.

Table 2.2. Point biserial correlations of individual items with SCS1-S total score and course grade.

	Version 1					Version 2				
Question number SCS1-S	Original question number on full SCS1	Correlation r with SCS1-S total score	p-value	Correlation r with course grade	p-value	Original question number on full SCS1	Correlation r with SCS1-S total score	p-value	Correlation r with course grade	p-value
1	1	.40	<.001***	.28	<.001**	3	.44	<.001***	.37	<.001***
2	2	.35	<.001***	.18	.023*	4	.44	<.001***	.12	.116
3	5	.26	<.001***	.06	.471	7	.10	.211	-.00	.957
4	6	.27	<.001***	.02	.772	8	.40	<.001***	.12	.115
5	9	.33	<.001***	.15	.059	10	.42	<.001***	.19	.013*
6	12	.39	<.001***	.21	.007*	11	.48	<.001***	.25	.001**
7	13	.38	<.001***	.19	.016*	16	.47	<.001***	.20	.010*
8	14	.48	<.001***	.41	<.001**	19	.56	<.001***	.46	<.001***
9	17	.45	<.001***	.21	.006*	20	.27	<.001***	-.10	.199
10	18	.15	.050*	-.14	.065	21	.41	<.001***	.17	.029*
11	22	.32	<.001***	.13	.095	23	.39	<.001***	.23	.003**
12	25	.30	<.001***	.08	.286	24	.35	<.001***	.16	.037*
13	27	.05	.515	-.22	.004*	26	.36	<.001***	.03	.713

Note: *indicates *p*-value below .05, ** indicates *p*-value below .01, *** indicates *p*-value below .001.

Table 2.3. Crosstabulation of item difficulty and discriminability using the scheme from Parker et al. (2016).

		Difficulty		
		Parker et al.		
		< 50%	50 - 75%	> 75%
Present Study	< 50%	4,5,6,7,8,9,11,12,13,14,16, 17,18,20,21,22,24,25,26,27	1,2,19	-
	50 - 75%	10	3,23	-
	> 75%	-	-	-
		Discriminability		
		Parker et al.		
		Poor (< 0.1)	Fair (0.1 - 0.3)	Good (> 0.3)
Present study	Poor (< 0.1)	27	-	-
	Fair (0.1 - 0.3)	5,18,20	6,7,25	-
	Good (> 0.3)	8,24	4,9,10,11,12,13, 16,17,21,22,26	1,2,3,14,19

Note: Difficulty is defined as the percentage of correct attempts for each question. Discriminability is defined by the point biserial correlation between the score on each item (counting unanswered questions as incorrect) and total SCS1-S score. For easier comparison we used the full SCS1 item numbers. The corresponding SCS1-S item numbers can be found in Table 2.2 for conversion to item numbers per version.

Verifying IRT assumptions

To check whether each of the SCS1-S versions loaded onto one factor we performed a confirmatory factor analysis (CFA) with diagonally weighted least squares (WLSMV) as the estimation method, and the variance of the latent factor constrained to 1. Chi-square tests indicated good model fits for both versions (Version 1: $X2(65) = 66.75$, $p = .42$; Version 2: $X2(65) = 64.35$, $p = .50$). Other goodness of fit statistics also indicate that the two versions of the SCS1-S seem to be measuring one latent variable: Comparative fit index, Version 1 0.96, Version 2 1.0 (above 0.90 is acceptable); root mean square error of approximation, Version 1 0.013, Version 2 < .001 (below .1 is acceptable); standardised root mean square residual, Version 1 .067, Version 2 .066 (acceptable below .8).

To test for the internal-consistency and reliability of the versions we also computed Cronbach's α for each version. For Version 1 we found a low Cronbach's α of .29, and for

Version 2 a higher Cronbach's α of .55. Both of these values are below the recommended level of reliability of .70 that is recommended for early-stage research (Lance et al., 2006). Table 2.4 summarises the sensitivity of α to each item. For each item, we report the change in Cronbach's α if that item was to be removed from the analysis. For Version 1 removal of items 4, 10 and 13 led to improved (higher) Cronbach's α, and on Version 2 this was true only for the removal of item 3. Items for which deletion would lead to a positive change are items on which scores show low consistency with other items on the version.

Fitting the IRT model

Xie, Davidson at al. (2019) tested a Rasch/1Parameter Logistic (PL) model, a 2PL model and a 3PL model. They found that the 2PL model showed the best fit, the Rasch model had the second-best fit. Consequently, in the current study we compared Xie, Davidson et al.'s two best models, a Rasch model (1PL) and a 2PL model. As for Xie, Davidson et al. (2019) the 2PL model provided the best fit for both versions of the SCS1-S, the ANOVA showing a greater Bayesian information criterion (BIC) for the 2PL model than for the Rasch model (Version 1: 2PL *BIC* = 2629, 1 PL *BIC* = 2595; Version 2: 2PL *BIC* = 2629, IPL *BIC* = 2590). We therefore performed the IRT analysis with the 2PL model. In contrast to Xie, Davidson and colleagues (2019), who excluded items 20, 24 and 27 of the full SCS1 version after finding that Cronbach's α would improve by deleting the items, we decided to leave all items in for the IRT analysis. We did this because we found different items that would have to be removed from those excluded by Xie, Davidson and colleagues (2019), therefore removing those items would make comparison with other studies more difficult. Table 2.4 shows the results of this analysis with the difficulty scores and discrimination scores for each item on each of the versions. Generally, an item with good properties would have a difficulty rating below three, and a discriminability rating above zero (Xie, Davidson et al., 2019). This suggests that for Version 1 item 3 may be a bit too difficult, and items 10, 11 and 13 are problematic discriminators, as for those items participants who got the item correct tend to do badly overall and vice versa. For Version 2, item 3 is a problematic discriminator and item 9 (and probably 12) too difficult.

Table 2.4. Sensitivity of Cronbach's alpha, difficulty and discriminability for each item.

	Version 1					Version 2				
Question number SCS1-S	Original question number full SCS1	Cronbach's α after deleting item	Change in Cronbach's α	Difficulty	Discriminability	Original question number full SCS1	Cronbach's α after deleting item	Change in Cronbach's α	Difficulty	Discriminability
1	1	.24	-.05	2.42	0.42	3	.52	-.03	-.13	0.63
2	2	.27	-.02	1.17	0.55	4	.52	-.03	1.65	0.99
3	5	.28	-.01	5.49	0.30	7	.59	+.04	-4.03	-0.32
4	6	.31	+.02	1.28	0.30	8	.53	-.02	0.82	0.83
5	9	.27	-.02	1.84	0.51	10	.53	-.02	0.04	0.59
6	12	.25	-.04	0.21	0.83	11	.51	-.04	1.73	0.87
7	13	.24	-.05	1.58	1.03	16	.51	-.04	1.67	0.82
8	14	.20	-.09	0.49	1.15	19	.49	-.06	0.14	2.89
9	17	.21	-.08	1.33	0.77	20*	.55	0	6.48	0.22
10	18	.31	+.02	-5.92	-0.32	21	.53	-.02	1.16	0.87
11	22	.28	-.01	-65.85	-0.01	23	.54	-.01	-0.61	0.82
12	25	.29	0	2.71	0.29	24*	.53	-.02	3.18	0.54
13	27*	.36	+.07	-2.18	-0.71	26	.54	-.01	2.41	0.47

Note: Sensitivity of Cronbach's alpha to the presence of each item and difficulty and discriminability of items. Values were determined with the 2PL IRT model. *Items that in Xie, Davidson et al. (2019) removal resulted in higher Cronbach's alpha.

2.4 DISCUSSION

The current study aimed to split the SCS1 programming test (Parker et al., 2016) into two parallel short versions to allow for repeated testing and shorter testing times. By doing so, we aimed to improve on the original SCS1 in four ways, by enabling: 1) shorter testing time, 2) the possibility for repeated testing, 3) knowledge of the external validity of the tests and items, and 4) generalization of knowledge on test and item quality to a student population outside Georgia Institute of Technology, where the full SCS1 was primarily developed. Below, we discuss how well we succeeded in reaching each of our aims.

We created two short versions of the SCS1 - the SCS1-Short, Version 1 (SCS1-S1) and Version 2 (SCS1-S2) - balanced for content and difficulty based on the content of the test items, on the Parker et al. (2016) and our own pilot studies. We tested the two SCS1-S versions on a relatively large cohort (N = 354) of university undergraduate programming students. Although the two SCS1-S versions did not show a significant difference in difficulty, Bayesian analyses demonstrated that we also could not convincingly conclude that they were of equal difficulty. Although not significant, scores on Version 2 were numerically higher than on Version 1, and it correlated slightly more strongly with course content (but again not significantly different from Version 1). Moreover, Version 2 showed a higher internal-consistency and reliability than Version 1. In sum, we are unable to conclude that the versions are fully parallel, and Version 2 seems to be of better quality than Version 1.

On average, the SCS1-S scores in our study were lower than the SCS1 scores in the study by Parker et al. (2016) who found an average 35% correct (9.68/27) on the full version of the SCS1. One possibility is that this is due to a difference in course content that the students were exposed to. Both the original FCS1 and the full SCS1 (which was inspired by the FCS1) were developed by researchers from Georgia Institute of Technology, USA and tested primarily on their students. It could therefore be that the questions were more closely related to the contents of their curriculum than to those of other universities, in particular in other countries. That course content may affect performance on SCS1 items is also suggested by the different correlations found for the FCS1 with course scores on the three different programming courses from which Parker et al. (2016) recruited their participants: performance of students who had undertaken MATLAB and Python courses

correlated more strongly with the FCS1, than that of students in a media computation course. Given that the correlations vary considerably by course, it may be that the test is sensitive to the specific material taught. In order to confirm this hypothesis, the SCS1-S (or the whole SCS1) would need to be systematically tested across different programming courses, preferably in a wide variety of contexts.

With regard to external validity, performance on both SCS1-S versions showed a significant correlation with course grade and these correlations did not differ significantly. This suggests that the questions in both versions reflect knowledge that is similar to the knowledge assessed in the unit exam. In addition, the magnitude of these correlations is similar to the correlations of the FCS1 (after which the SCS1 was modelled) with the students undertaking MATLAB and Python courses, but higher than the FCS1 correlation with the students in a media computation course (Parker et al., 2016) (see Introduction for the exact correlations). This suggests that the SCS1-S might reflect programming knowledge to a similar extent as the FCS1.

The SCS1-S2 showed similar internal-consistency (Cronbach's alpha= .55) to the full SCS1 as reported in the post-course test by Parker et al. (2016) (Cronbach's alpha = .59), but lower than for the full SCS1 as reported in the pre-course test by Xie, Davidson et al. (2019) (Cronbach's alpha = .70). To allow us to investigate whether these differences were due to pre-course versus post-course use of the SCS1, B. Xie (personal communication, March 27, 2020) calculated the internal consistency of their SCS1 post-course data samples, which included the Parker et al. (2016) post-test data. Importantly, in these calculations, as for their SCS1 pre-test data (Xie, Davidson et al., 2019), Xie, Davidson et al. (2019) used an exclusion criterion that we and Parker et al. (2016) did not use. Namely, they excluded all participants who answered fewer than 10 questions. When applying their exclusion criterion to the Parker et al. (2016) part of their post-test sample, the internal consistency increased to the same level as for the SCS1 pre-test values reported in Xie, Davidson et al. (2019) (Cronbach's alpha = .74). These results suggest that the differences in internal consistency between our and the other two studies, are not due to the timing of testing, pre- or post- course, but more likely an effect of the applied exclusion criterion. SCS1's internal consistency appears to be higher when only including the students who complete more questions. There are several potential explanations for this effect. It could be that this

excludes the less motivated or less able students, thereby improving the quality of the answers and reducing the guessing rate. It is also possible that fewer missing values per student improves the statistical quality of the sample.

At the individual item level, we did not compare findings with the Xie, Davidson et al. (2019) study because of the differences in their sample (assessed before the start of the course; exclusion of students who answered fewer than 10 questions). Compared to Parker et al. (2016), the items were generally more difficult for participants in the current study. However, for our sample, many items had better levels of discrimination than they did for Parker at al.'s (2016) sample. A factor contributing to these differences could be that we allowed students to skip questions while Parker et al. (2016) did not, which may have reduced the guessing rate in our data. Of the 27 original items in the full SCS1, if one were to keep only the items that, in our study: 1) were answered correctly at levels greater than chance, 2) showed sufficient discriminability, and 3) contributed to stronger internal-consistency and reliability, 12 items would have to be excluded, and 14 items retained – retaining those with original SCS1 numbers 1-3,8-10, 12, 14, 17, 19, 21, 23, 25 and 26. These good quality items cover all topics and question types, except for the topics "function parameters" and "function return values". It is of note that function-use was not covered very extensively in the course undertaken by our sample. Therefore, it might be that those items showed poorer quality in the current study, because those topics received relatively less coverage than in the Georgia Institute of Technology courses. This again suggests that the SCS1 questions are sensitive to course content, and that these items may show better quality for students following a different course. Therefore, until this hypothesis has been proven to be incorrect, we recommend keeping items covering all topics in the tests to ensure content validity, even if this means potentially sacrificing some test quality.

2.4.1 Future research

Given that Versions 1 and 2 of the SCS1-S were not found to be parallel, there remains need for two parallel short versions. Future studies could address this in three ways. Firstly, it is possible that SCS1-S1 might show better qualities in a course with different content. Future studies could test this by administering both short versions to different student populations. Secondly, future studies could use the current and past findings on item

quality to attempt to create different subsets of the items to try and achieve better quality parallel versions, while ensuring that question content remains balanced. Lastly, future studies could attempt to create new items. For example, by selecting the best performing current items and creating items that test the same skill with a different question. The new items would then need to be validated with several student populations.

Additionally, combined with the results from previous studies, our results suggest that item and test quality are dependent on course content. Future studies could further investigate these effects by using the SCS1 or SCS1-S2 after a variety of different programming courses. Meanwhile, researchers who are looking to use the SCS1 or the SCS1-S2 in their studies need to be aware of these dependencies and are advised to look closely at how their course content relates to the SCS1 and SCS1-S programming items.

2.4.2 Conclusion

This study evaluated the effectiveness of two novel half-versions of an existing hour-long test of programming skill (the SCS1). The SCS1-S2 performed similarly to the full SCS1 in Parker et al. (2016), suggesting that it can be considered a validated short version of the SCS1, allowing researchers the option of a test that can be completed in 30 rather than 60 minutes. This will make the test more practical for use in a wider variety of studies and contexts, expanding the opportunities for use of this tool.