Failure Rates in Introductory Programming — 12 Years Later

By Jens Bennedsen, *Aarhus University* and Michael E. Caspersen, *It-Vest – Networking Universities*

For many years, it has been a common conception that CS1 is difficult to teach and learn, and that failure rates in CS1 are high. Until 2007, this claim was primarily anecdotal. In 2007 we published a paper on failure rates in CS1 and found an average failure rate of 33%. Now, more than ten years after, we have replicated our study with close to three times as many respondents. With an average failure rate in 2017 of 28%, it seems that the state of affairs is improving; and compared to 42-50% failure rates for college algebra in the US, an average CS1 failure rate of 28% does not seem alarmingly high.

In 2007, we published the paper *Failure Rates in Introductory Programming* [1] in which we present our investigation of the often expressed the belief—but until then only backed by anecdotal evidence—that introductory programming courses generally have high dropout and failure rates. In our study, we did not find the failure rates of introductory programming courses to be alarmingly high (we found an average failure rate of 33%), but we explicitly refrained from drawing hard conclusions, especially due to a relatively low number of respondents to our survey (80 out of a total population of 497 participated in our survey, i.e., 16%). In the conclusion, we recommended that relevant organisations, e.g., the ACM Education Council, engage in this to provide reliable and representative data from as many institutions as possible, but to our knowledge nothing like this has happened.

It is now more than ten years since we published our study. Our paper has had and still receives a high number of citations—more than half of the current citations are received within the past three years. The high number of citations strongly indicates that the topic is still relevant to many researchers. Most of the papers citing [1] use the article to argue that CS1 is challenged and has a certain kind of problem, and then go on to suggest some intervention to alleviate the perceived problem.

In the past 10+ years, we are only aware of one similar study [12]. In this paper, the authors answer the call for further substantial evidence on the CS1 failure rate phenomenon, by performing a systematic review of introductory programming literature, and statistical analysis on pass rate data extracted from relevant articles. They found an almost identical mean worldwide failure rate of 32.3%.

> Guzdial [6] commented on the study reported in [12]. Bennedsen and Caspersen answered that question with a big international survey. They recognised the limitations of their study—it was surveying on the SIGCSE member's list and similar email lists (i.e., to teachers biased toward being informed about the latest in computing education), and they got few responses. The ITiCSE 2014 best paper awardee [12] tried to measure failure rates again, by studying published accounts of pass rates. While they got a larger sample size this way, it's even more limited than the Bennedsen and Caspersen study.

Guzdial highlights three reasons why this is a more limited study.

1. Nobody publishes a paper saying, "Hey, we've had lousy retention rates for ten years running!"
2. The same class retention data appeared in several of the included papers (i.e., was counted more than once.
3. The authors do not explicitly cite the papers used in their meta-analysis, so it is not easy to see if the source of data is valid.

Guzdial concludes: "This paper is exploring an important question and does make a contribution. But it's a much more limited study than what has come before."

Many references to our previous paper indicate a widespread conception that teaching and learning programming is still considered hard [9]. In a recent PhD dissertation [7], the author writes:

> programming is a very difficult skill to learn, and even more difficult skill to master. After introductory courses, various students typically still have difficulties in reading the program code and writing simple programs. Moreover, the dropout rates in introductory programming courses are typically quite high.

We have only knowledge of one study trying to answer these questions in the last ten+ years ... So, we think it is relevant to re-ask the research questions we had 12 years ago to see the current status of things and see developments if there are any.

Thus, our more than ten years old study is heavily cited, recently more than ever, and there has only been one similar study which according to Guzdial is more limited than ours. It seems that time is ripe for a more thorough investigation of failure rates for introductory programming courses.

As the first step in a more thorough investigation of failure rates, we decided to simply replicate our analysis published in 2007. As a potential second step, we intend to look deeper into reasons for the observed failure rates; thus, as a complement to quantitative data collection and analysis, we intend to undertake a more thorough qualitative study to reveal potential patterns and regularities in reasons why students fail introductory programming courses. In times of "CS for All," our community must find ways to provide more effective programming instruction to truly make CS accessible for all. Furthermore, we look to find data for other courses to compare the failure rate of CS1 with other relevant courses (e.g., math and physics).

We replicate our previous study in the sense that we (1) use the same questionnaire, (2) collected data through the same channels (authors of five computing education conferences), and (3) have analysed and presented the data in a form like our 2007 paper. It's not an exact replication since the authors are different and therefore cover different institutions. However, the study does provide an updated state of the situation, which is what we set out to achieve.

We did consider making improvements to our previous study, particularly to the questionnaire (e.g., by clarifying questions and fixing potential ambiguities). These considerations lead to the realisation that we subsequently want to make a different and more qualitative study. Thus, to maximise comparison between the two studies, we decided to maintain the research design, i.e. replicate our previous study. Various more "radical" changes/improvements to the research design will be postponed for a potential second step—a more qualitative study—as mentioned previously.

RESEARCH

THE RESEARCH QUESTIONS

As we've noted, it continues to be the general view that there are high failure rates in introductory programming courses. However, to our knowledge, no worldwide statistics on failure rates, dropout rates, or pass rates for introductory programming courses at university level exist to back up this postulate. As described earlier, we tried to validate this claim in 2006/07 [1], where our research questions were: *What are the failure and pass rates for introductory programming courses at the university level? And: Is the failure rate high?* We have only knowledge of one study trying to answer these questions in the last ten+ years, namely the study [12]. So, we think it is relevant to re-ask the research questions we had 12 years ago to see the current status of things and see developments if there are any.

THE QUESTIONNAIRE

To answer the research question, we developed a short, web-based questionnaire in 2006/07 [1]. Since this is a replication study, we used that same method and tools as we did back then, that is, a questionnaire sent out to selected university persons.

In the questionnaire, four terms (same as the terms in [1]) were defined and the respondents were asked to give numbers for **abort** (the number of students aborting the course before the final exam), **skip** (the number of students not showing up for the final exam, but was allowed to), **fail** (the number of students who failed the course) and **pass** (the number of students who passed the course).

These categories aim to capture the worldwide differences in evaluation where students may decide to not attend or be prevented from attending the final exam at various points in time before the actual exam where they either pass or fail. Also, pass/fail may be decided based on an actual exam event, it may be based on grades earned throughout the course, or it may be based on a combination.

Apart from these numbers, we asked for the type of institution (university, college, etc.) and how the course was evaluated.

THE PARTICIPANTS

There exists no universal database of CS teachers, so we need to identify some group of teachers as a representative group. In 2006/07 we wrote to the authors of papers at five different conferences: four focusing on computer science education and one on learning technologies. Again, this is a replication study, so we decided to find the authors from the same conferences and address them via email. In 2006/07 there were 575 authors from the five conferences, ten years after there were 1068 authors! The five sources are the authors of articles for: Koli Calling 2017, the 17th Annual Finnish / Baltic Sea Conference on Computer Science Education [10]; 48th Technical Symposium on Computer Science Education [3]; 22nd Annual Conference on Innovation and Technology in Computer Science Education [4]; 17th International Conference on Advanced Learning Technologies [8]; and 19th Australasian Computing Education Conference [11].

Of the 1068 authors, we could not find email addresses for 48, leaving 1020 authors in total. Of those, 32 had more than one publication. Some of the authors were not teachers but representatives for different organisations, two of those offered contact information to relevant teachers thus adding ten respondents. Thus, we ended up with 998 authors whom we approached.[1] The distribution of the respondents was as follows: 64 from ICALT 2017, 62 from SIGCSE 2017, 36 from ITiCSE 2017, six from ACE 2017 and three from Koli Calling 2017.

Just like our last study, it is debatable whether the selected respondents (universities) are representative (see subsection *Threads to Validity*, in the **Discussion** section). Another problem is whether the persons responding to the questionnaire are representative of their university; for some of the responses we have had indications that a few respondents only gave data from "their own" introductory course, not the entire institution. Another problem we encountered was a respondent who wrote that it was forbidden for him to give the numbers when we needed the name of the institution. Lastly, two asked about the "final exam"—their course did not have a final exam. Our intention was "the point in time where the course has ended and it is officially decided who will pass and who will fail." Respondents may have interpreted this differently.

We do not claim that this study is representative for all universities with a computer science program, but it is useful as an indicator of (a lower boundary of) the state of affairs.

Requests for data were sent out to the 998 named respondents in early January 2018 using the survey system surveyXact (due to technical problems, the requests for data to the 23 authors of ACE 2017 were not distributed until March 2018). We send out two reminders to the respondents who had not answered (after one and three weeks respectively). Forty-one of the requests for participation were undeliverable, giving a population of 957.

Overall, 170 respondents answered the questionnaire in full, giving a response rate of 17.8% (161 gave partial information; these are not included in the following analysis). The 17.8% response rate is an increase from the 12.7% response rate we had in 2006/07; however, in absolute numbers we have an increase from 63 respondents in 2006/07 to 170 respondents in 2018, i.e. a 170% increase!

[1] We cannot know if the people used different emails, so theoretically there could have been fewer respondents.

The geographical distribution of responses is presented in Figure 1 (the observant reader will notice that the sum in Figure 1 is only 161. Nine respondents indicated that they were not teaching at university or college level; these are excluded from our analysis). In the following analysis, we have only used the responses from universities and colleges—22 from colleges, 138 from universities. In our last study, the ratio between colleges and universities was 1:4, now it is 1:6. The geographical distribution for this survey is better than our survey in 2006/07, where North America was more over-represented.

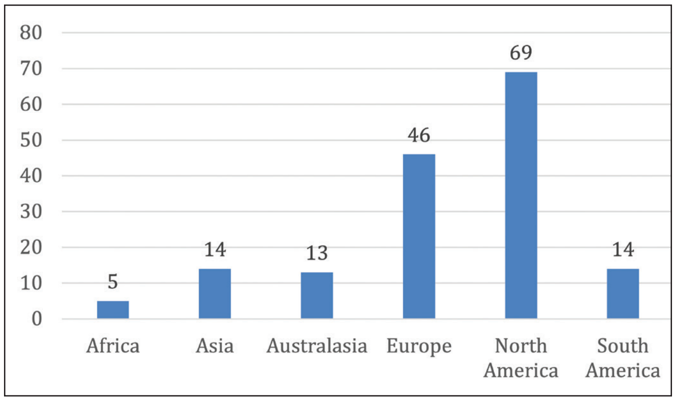

Figure 1: Number of respondents per continent.

RESULTS

PASS, FAIL, ABORT, AND SKIP RATES

Our main question twelve years back was: "What are the pass and failure rates?—is it true that CS1 is a particularly difficult course?" Our main question remains the same. Figure 3 shows that 72% of the students pass. The calculation is based on aggregate numbers, i.e., a course with more students counts more. If we have one course with ten students and a pass rate of 90% and another course with 100 students and a pass rate of 70%, the average pass rate will be $\frac{(0,7*100)+(0.9*10)}{(100+10)} = 71.8\%$. Figure 2 shows a histogram of the sizes of the courses.

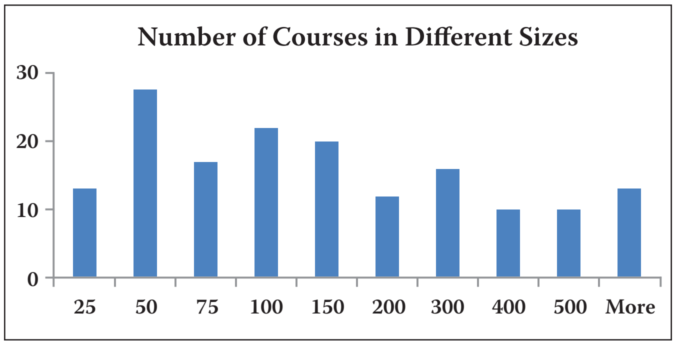

Figure 2: Histogram of course size.

The overall pass rate is 72%. In our 2006/07 study, the overall pass rate was 67%; thus, the pass rate has increased by five percentage points, see Figure 3. Comparing the pass rate from the 2006/07 study to the pass-rate of 2018 using a t-test, we conclude that there is a statistically significant higher pass-rate in 2018 than in the 2006/07 study. Consequently, our main conclusion remains the same as in [1]—it seems difficult to justify the often-postulated claim that introductory programming is very difficult and that many students fail. Further evidence for this conclusion is provided in subsection *Is the Failure Rate of CS1 High?* (found in the section **Discussion**) where we compare failure rates in introductory programming courses with those of college algebra in the US.

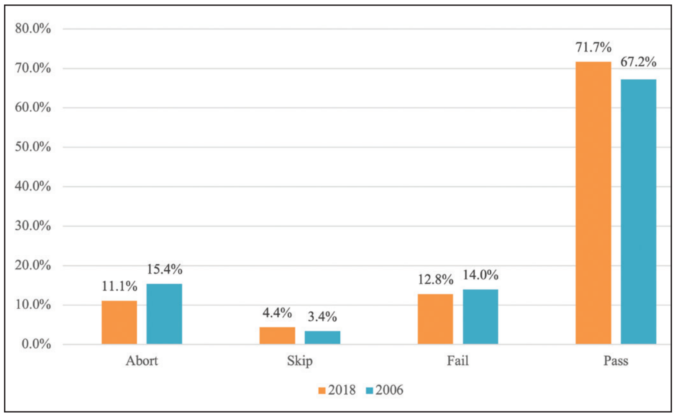

Figure 3: Pass, fail, abort and skip rates; aggregate.

There is a huge variation in the pass, fail, abort and skip rates found—from one course with only 9% of the students passing (7 out of 79) to a course with all students passing (100 out of 100 students), see Table 1.

Table 1: Mean, median and standard derivation.

	Abort	Skip	Fail	Pass
Mean	0.107	0.04	0.127	0.726
Median	0.076	0.001	0.1	0.771
Standard deviation	0.113	0.07	0.113	0.192

If we compare the pass-rates in the six continents, they are not the same. Using a pairwise t-test, only Europe and North America have a statistically significant difference (p=0.04), for all other pairs of continents, we cannot say if one has a higher pass-rate than the other. Table 2 presents the calculated the mean, median, and standard deviation for the continents.

Table 2: Mean, median and standard deviation by continent.

	Africa	Asia	Australasia	Europe	North America	South America
Mean	0.831	0.715	0.722	0.687	0.762	0.729
Median	0.837	0.824	0.723	0.689	0.814	0.750
Standard Deviation	0.077	0.248	0.144	0.193	0.187	0.203

The size of the courses also varies greatly, from the smallest with only 14 students to courses with more than 1200 students. The mean course size is 196 (a lot bigger than in 2006/07 where it was 116); only 8.8% of the courses in the current study have fewer than 30 students—a big drop from our previous study where the number was 23%. Figure 2 shows a histogram of course sizes for the current study.

In our 2006/07 study, we concluded that small classes (with less than 30 students) did better than the larger ones. For the present study, there are too few small classes to draw conclusions along these lines.

UNIVERSITIES AND COLLEGES

In general, between universities and colleges there is a little difference between the abort, skip, fail and pass rates. There is a tendency towards colleges doing a little better than universities as can be seen in Table 3 and there is a statistically significant difference (p-value 0.001 using a t-test: Two-Sample Assuming Unequal Variances).

Table 3: Comparison of Pass, fail, abort and skip rates.

Row Labels	Average of abort %	Average of skip %	Average of fail %	Average of pass %
College	6.5%	0.8%	9.7%	83.0%
University	11.3%	4.5%	13.2%	71.0%
Grand Total	**10.7%**	**4.0%**	**12.7%**	**72.6%**

In our 2006/07 study, we found no difference. However, the number of respondents from colleges in the current study is low (22 from colleges vs. 139 from universities).

DISCUSSION

IS THE FAILURE RATE OF CS1 HIGH?

The headline of this subsection is one of our two research questions. The obvious counter question is: what is "high?" When is a failure rate high? In our 2006/07 study we found an average failure rate of 33%; from a macroscopic analysis based on data from UNESCO, we concluded that 33% was not an especially high failure rate. However, we do acknowledge that the macroscopic analysis based on data from UNESCO was very uncertain.

To provide a frame of reference for our findings, we have looked for public data on failure rates for similar programs or courses, e.g., introductory math courses. These data are not easy to find, but we have managed to find information about introductory math courses in college and university in the US [5, p. 49]:

> Each year in colleges and universities across the United States approximately 1,000,000 students enroll in college algebra; and each year approximately 50% of these students fail to pass this course with a grade of C or better.

"Failure to pass with a grade of C or better" is not equivalent to fail; because we do not know how many students get a D (barely pass), we cannot conclude a failure rate of 50% for college algebra, but we can conclude a pass rate of at least 50%. However, later in the same report it says [5, p. 61]:

> Even though our pass rate (approx. 58%) is higher than the national average ...

i.e., the national average pass rate is between 50% and 58%. And conversely, the US national average college algebra failure rate is between 42% and 50%. Compared to 42–50% for college algebra, an average CS1 failure rate of 28% does not seem particularly high.

One of the anonymous reviewers noted that the interpretation of 'high' might vary by type of institution; e.g., that a failure rate of 28% at an elite university may be considered outrageously high and that 28% in a small university may be considered low.

Realizing that the average CS1 failure rate has decreased from 33% in 2006/07 to 28% in 2018, and that the average US failure rate in college algebra is 42–50%, we conclude that the CS1 failure rate is not alarmingly high.

It appears that introducing students to computing is still one of computing education's grand challenges and that we as a community have a huge challenge in developing more inclusive and effective learning environments and instructional methods for CS1. ... In a time where computing/informatics education is becoming general education for all and students don't choose to learn to programme out of personal interest, the challenge not only persists, but is reinforced.

THREADS TO VALIDITY

We identify three aspects where the quality of our data can be questioned: number of data points, the representativity of the respondents, and the clarity of the questionnaire.

A total of 161 universities provided data for this study (we had 170 respondents, but nine of these were not from higher education and were excluded from our analysis). In the UK alone, there are around 130 universities/colleges; in this light, data from 161 universities worldwide is a relatively low number. However, the number of data points is a factor of 2.6 larger than in our 2006/07 study.

The respondents are authors of papers from four CS education conferences and a conference on advanced learning technologies. People from these communities are likely to be more concerned about the quality of teaching and learning and proactive in improving their teaching than the average CS1 instructor (on the other hand, the respondents are not necessarily those teaching CS1 at their university). That said, it must be emphasized that we have used the same sources in this study as in our 2006/07 study and thus aimed at variable control for this aspect.

In a couple of cases, respondents have asked clarifying questions regarding the interpretation of terms used in the questionnaire. In general, respondents may have interpreted key terms in the questionnaire differently, thus provided data that are not strictly comparable. But again, to aim for variable control, we have used the same questionnaire as in our 2006/07 study.

Local culture might influence how the numbers are calculated. In Denmark, the number of students enrolled in a course is calculated one month after the course starts, implying that students aborting the course before that date are not calculated as aborts. In other contexts,—as one of the reviewers pointed out—it is standard practice… for an abort to count as a fail when calculating GPA … then there are almost no aborts, since an abort requires additional paperwork with no difference to GPA.

FUTURE WORK

As indicated in the introduction, it would be useful to conduct a more thorough investigation of failure rates for introductory programming courses, e.g., an international and multi-institutional qualitative study, in order to reach a deeper understanding of causes; this would be relevant for the community in order to identify specific challenges that could be addressed more directly to improve even further the teaching and learning of CS1.

CONCLUSION

We have replicated our 2006/07 study on failure rates in CS1. Based on data from 161 universities and colleges (63 in 2006/07), we found an average failure rate of 28% (33% in 2006/07). We have contrasted this finding with numbers of failure rates from college algebra in the US and concluded that computer science does not seem to have alarmingly high failure rates.

Of course, this does not imply that things cannot improve, and there are perceptions that improvement is necessary. In the report [9, p. 12], the authors write:

> Major concerns exist among the academic community internationally that when we set out to teach programming skills to students, we are less successful than we need to be and ought to be [. . .]. The particular concern is that, after more than forty [now fifty] years of teaching an essential aspect of our discipline to would-be professionals, we cannot do so reliably. Indeed, there are perceptions that the situation has become worse with time.

In a recent article in *ACM Inroads*, Kim Bruce writes about Five Big Open Questions in Computing Education [2]. In the opening paragraph, he writes: "Certainly, there is no shortage of problems in computer science education," and then goes on to discuss the five that he has nailed down as the most significant and challenging.

1. How can we deal with skyrocketing enrollment in CS undergraduate courses?
2. How can we make CS courses more inclusive?
3. How can we develop and support teachers for pre-college instruction in computing?
4. How can we get students to seriously address the ethical implication of computing?
5. How can we alleviate the high dropout rate in introductory CS classes?

It appears that introducing students to computing is still one of computing education's grand challenges and that we as a community have a huge challenge in developing more inclusive and effective learning environments and instructional methods for CS1.

In a time where computing/informatics education is becoming general education for all and students don't choose to learn to programme out of personal interest, the challenge not only persists, but is reinforced. ❖

Acknowledgements
We thank all respondents who took the time to answer the questionnaire. In particular, we thank those colleagues who have provided insightful comments and questions. We would also like to thank the anonymous reviewers for their insightful and constructive comments to an earlier version of this manuscript.

References

1. Bennedsen, J. and Caspersen, M. E. Failure rates in introductory programming. *SIGCSE Bulletin* 39, 2 (June 2007), 32–36; doi: https://doi.org/10.1145/1272848.1272879.
2. Bruce. K. Five Big Open Questions in Computing Education. *ACM Inroads*, 9, 4 (2018), 77–80.
3. Caspersen, M.E., Edwards, S.H., Barnes, T. and Garcia, D.D., Eds. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (SIGCSE 2017), Seattle, WA, USA, March 8–11.
4. Davoli, R., Godweber, M., Rössling and Polycarpou, I., Eds. (2017). *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education* (ITiCSE 2017), Bologna, Italy, July 3–5.
5. Ganter, S.L. and Haver, W.E. *Partner Discipline Recommendations for Introductory College Mathematics and the Implications for College Algebra*. The Mathematical Association of America, 2011.
6. Guzdial. M. A Biased Attempt at Measuring Failure Rates in Introductory Programming, *Computing Education Research Blog*, September 30, 2014; https://computinged.wordpress.com/2014/09/30/a-biased-attempt-at-measuring-failure-rates-in-introductory-programming/. Accessed 2019 April 2.
7. Kaila, E. *Utilizing Educational Technology in Computer Science and Programming Classes—Theory and Practice*, PhD Dissertation, Turku Centre for Computer Science, Department of Future Technologies, University of Turku, 2018.
8. Kinshuk, Sampson, D.G., Vasiu, R., Chang, M, Chen, N-S and Huang, R., Eds. *Proceedings of the 2017 IEEE 17th International Conference on Advanced Learning Technologies* (ICALT 2017), Timisoara, Romania, July 3–7.
9. McGettrick, A., Boyle, R., Ibbett, R., Lloyd, J., Lovegrove, G. and Mander, K. *Grand Challenges in Computing: Education*, (The British Computer Society, 2004).
10. Montero, C.S. and Joy, M., Eds. *Proceedings of the 17th Koli Calling Conference on Computing Education Research* (Koli 2017), Koli, Finland, November 16–19.
11. Teague, D. and Mason, R., Eds. *Proceedings of the 19th Australasian Computing Education Conference* (ACE 2017), Geelong, VIC, Australia, January 31–February 03.
12. Watson, C. and Li., F.W.B. Failure Rates in Introductory Programming Revisited. *Proceedings of the 19th annual Conference on Innovation and Technology in Computer Science Education* (ITiCSE 2014). (New York, NY, USA ACM, 2014), 39–44; doi: http://dx.doi.org/10.1145/2591708.2591749

Jens Bennedsen
ST Learning Lab
Aarhus University
Aarhus, Denmark
jbb@stll.au.dk

Michael E. Caspersen
It-Vest–Networking Universities
Aarhus, Denmark
mec@it-vest.dk

DOI: 10.1145/3324888