codio

2019 SURVEY OF COMPUTER SCIENCE TEACHING

Challenges and Resources in U.S. Colleges and Universities

CONTRIBUTORS:
ELISE DEITRICK, PHD
JOSHUA STOWELL

SUGGESTED CITATION:
DEITRICK, E., & STOWELL, J. (2019). *2019 SURVEY OF COMPUTER SCIENCE TEACHING: CHALLENGES AND RESOURCES IN U.S. COLLEGES AND UNIVERSITIES. CODIO INC.* RETRIEVED FROM **WWW.CODIO.COM/RESEARCH/2019-COMPUTER-SCIENCE-TEACHING-SURVEY**

Forward

Computing education (CEd) is important, everyone agrees. President Obama committed hundreds of millions of dollars to "Computer Science for All" (White House, 2016); governments have developed curricula and made computing a school subject across the world; online providers compete to teach coding (such as Code Academy, code.org, the Hour of Code, Khan Academy and Coursera); and tech giants put money into supporting CEd projects (CISCO supports the BlueJ and Scratch initial programming environments and Google funds substantial professional development programs and has produced a series of CEd research reports...).

With all of the effort and resources going into CEd, it would be comforting to think that we know what we are doing—that the problems of teaching and learning computing topics are well understood, that the solutions are known, and that best practices are widely shared. But this ideal picture is very much a work in progress. We still don't know enough about how students learn computing subjects, what effects different teaching approaches have, or how to equally engage people of all races and genders in the field. CEd research (CEdR) is how we work to understand and improve this.

– SALLY FINCHER AND ANTHONY ROBINS (2019, P.1)

Contents

Executive Summary

With increasing student-to-teacher ratios and adoption trends shifting from traditional textbooks to technology platforms, we conducted a survey of computer science instructors at US higher education institutions to find out the challenges they face, understand what they need to resolve those challenges, and identify solutions we should be building. Participants identified their most significant challenges and needs in terms of feedback and grading, curriculum creation and setup, course administration, and flexible course materials. Respondents ranged from graduate students to department chairs, and they taught classes ranging in size from 7 to 2,000 students, with 16.7% of class rosters reaching into the triple digits. Languages being taught were dominated by the C family (C/C++/C#), Java, and Python. Student enrollment numbers were highest (29.6%) in CS1 with business courses close behind (22.2%)—more than double those of CS2: Data Structures (10.6%).

In terms of student feedback, manual grading was the highest priority pain point (72%)—ranked higher by those teaching at a smaller scale due to the lack of TA resources. In terms of curriculum, teachers of small classes struggled with technology-related challenges—such as acting as technical support—where large scale courses cited pedagogical concerns such as low student engagement as their top concern. This concern could be a result of resource distribution—similar to the student feedback findings. For course admin, the majority of teachers (56%) were primarily concerned with increasing student-to-teacher ratios followed by inappropriate classroom or lab space. Over 75% reported physical lab resources. However, 58% indicated virtual labs, student VMs, and/or cloud resources—a possible indication of a shift towards online learning in higher education. This is complicated by the fact that while 26% of physical lab costs are translated to the students where the resources exist, 40% of instructors do not know who is responsible for paying for virtual lab technology.

A shockingly large 61.3% reported Open Educational Resource (OER) usage—with only 9.5% of teachings relying on purely paid resources. Small courses were over twice as likely to use paid only than larger courses (13.6% in small to 5.3% in large). Most who used paid resources used them in conjunction with OER—47.6%—whereas 42.9% used only OERs. This is troubling as it might indicate a lack of appropriate paid materials—but students are by-and-large being asked to pay for these inadequate resources. 95% of students are responsible for textbooks, and half of the students using paid platforms are paying out of pocket. 58% of classes have students paying for resources that the teacher then supplements with OERs to meet their needs.

Many instructors were interested in curricular assessment resources, favoring editable, auto-graded questions over questions in text or a simple bank of questions. Similarly, instructors ranked editable text over a simple PDF or digital text. This reiterates educators' increasing need to customize content for their context.

When looking at what would cause a teacher to shift platforms, a downloadable PDF and the student dashboard dropped noticeably. A digital textbook that allowed students to write and run code jumped from 6th to 3rd on teachers' list of priorities.

Introduction

Computer Science Education is a relatively new, quickly evolving field. To build a responsive product, Codio keeps up with Computing Education Research (CER) and Computer Science teaching practices by regularly reviewing research articles, news, and trends, as well as conducting our own investigations.

Many computing education researchers and projects aim to improve how computing is taught and learned. Denny et al (2019) found that 78% of the 118 surveyed computing educators "would make a direct change to their teaching practice if at least one of their [research] questions was answered" (Denny, 2019, p. 264). This report synthesizes recently published community knowledge Codio has been reading and results from our own investigations in an effort to support communication between computing education researchers and practitioners to enable better computer science education for everyone.

As we have seen in the past year, traditional textbooks are on the decline (Johnson, 2019) and being replaced or supplemented by different OER materials (Seaman, & Seaman, 2017; Pitt, Farrow, Jordan, de los Arcos, Weller, Kernohan, and Rolfe, 2019) and technology platforms (Top Hat, 2018).

Additionally, computer science learners are slowly moving to less traditional spaces (Arbeit, Bentz, Cataldi, and Sanders, 2019)—many of which are online (Online Higher Education. Hyper Fragmentation and the US Post Secondary Market, 2019). These drastic changes, particularly in higher education institutions, are causing an increased need for technology in CS teaching. As teachers and instructors increasingly adopt these new edTech platforms, it is important to know what features are most valuable and how these tools are being adopted and combined specifically within a CS education context. Additionally, to better understand the motivations behind these shifts, it is worth identifying the pain points that are at the top of mind and possibly not being fully addressed by current tools.

> **"The $300 textbook is dead, we have to reinvent and make a future for this company around $40 to rent an ebook, $80 for a completely integrated package that provides much more personalization, adaptive assessment capabilities that support faculty."**
>
> — JOHN FALLON, CEO, PEARSON (JOHNSON, 2019)

2019 Survey of Computer Science Teaching

This past summer, Codio conducted a survey to better understand the state of computer science education in higher education institutions in the United States.

METHODS

US-based computer science and engineering school lecturers and professors from a proprietary database of 4,419 were emailed a URL to the survey, which was hosted on Qualtrics.

The survey consisted of 20 questions covering topics including courses taught, number of students, course administration, grading/feedback for students, curriculum creation and development, as well as information about their institutions' technology setup. Participants were also asked about the types of resources they use with the courses they teach.

Qualtrics software estimated that the survey would take the average respondent 12 minutes to complete.

82 CS and STEM instructors (professors and lecturers) participated in the survey. Respondents were not required to complete any of the questions, resulting in partial data for 33 participants and a full data set for 49 participants. In the analyses below, any respondent with the required questions answered was included, even if they only provided partial data. For example, below, we analyze class size vs. GA/TA resources. Participants who completed both of those fields were included even if they did not complete other demographics or background questions.

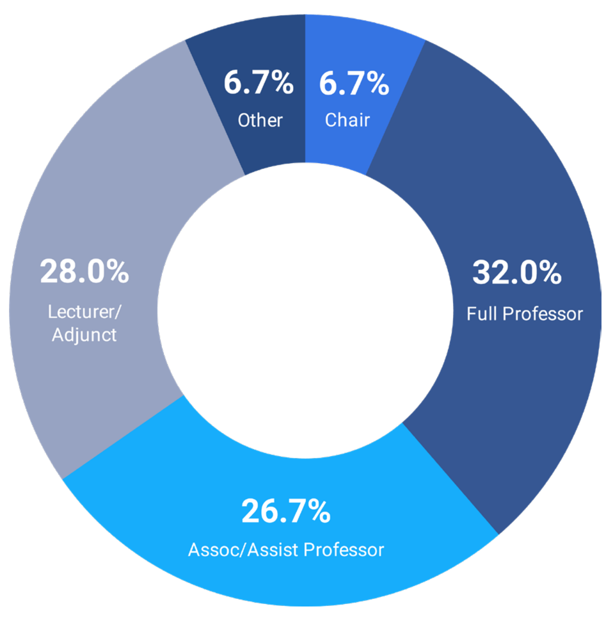
This way of recruiting participants and analyzing data has some clear limitations. Given the total number of CS Educators, the number of respondents is quite small. This is compounded by the recruitment being based on a database built in a proprietary, non-randomized or representative way and respondents being limited to the US. In terms of analysis, the use of incomplete data as described above could cause slight misrepresentations between analyses since the participant sets in each analysis could shift slightly.

RESPONDENT DEMOGRAPHICS/CONTEXT

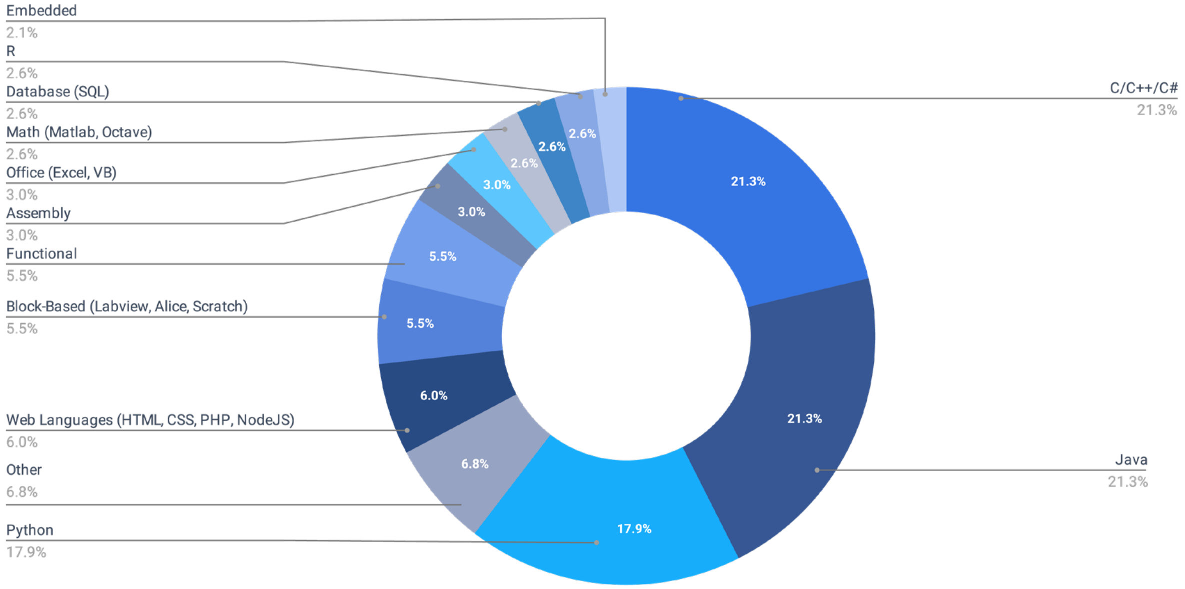
A quick categorization of submitted position titles reveals a wide range of respondents' titles from graduate students to department chairs. The majority of respondents were in teaching, tenure-track, or tenured faculty positions. Based on job title, 65.3% appeared to be tenure track, compared to the national average of approximately 82% reported by the National Academies of Sciences, Engineering, and Medicine (2018). This could be due to the increasing growth of teaching faculty at universities and colleges. Teaching positions grew 67% versus the 22% growth in tenure track positions (Ibid).

Categorizing all listed programming languages being taught in reported courses (with some grouping for those with smaller counts), we see that Java, Python and C/C++/C# account for 60% of the programming languages students are learning. This mirrors what we see in the top spots of the TIOBE index whose rankings as of September 2019 are Java, C, Python, C++, and C# (TIOBE Index for September 2019, 2019).

Titles of Respondents



Programming Languages Being Taught



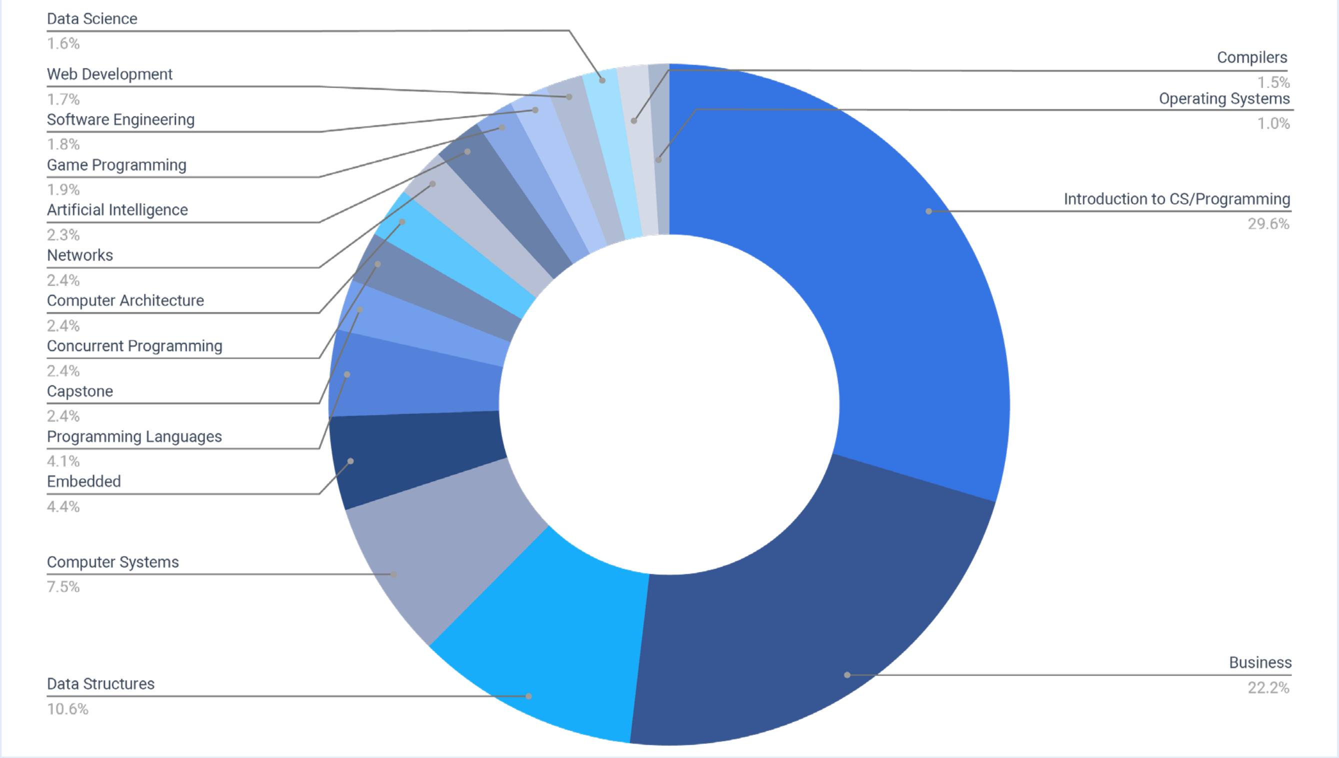
Looking at the wide variety of courses being taught by respondents, it is notable that 39% of courses reported were either a CS1 (Introduction to CS/Programming) or CS2 (Data Structures) course. Below illustrates the distribution of students between courses with over 100 students. The students below represent about 85% of the reported student count—the rest were spread out of a variety of smaller courses such as ethics, math and statistics courses, and IT courses. We see that there is a surprisingly large amount of students in business courses that teach flavors of programming (often Visual Basic within Excel). This could be the result of an increase in non-CS majors enrolled in CS courses—a rising trend in higher education, with some institutions introducing special mandates for programming courses for certain non-CS majors and others requiring that all non-CS majors take at least one programming course in order to meet the requirements for graduation (Chilana et al, 2015).

Looking at the average class size for these course titles, the largest five courses on average were: Business (383 students), Computer Systems (156 students), Computer Architecture (83 students), Compilers (78 students), and Capstone (62 students). The smallest classes averaging enrollments of less than 20 were: CS for non-majors (17 students), Algorithms (16 students), Ethics (15 students), and Databases (13 students). There did not appear to be any pattern to average class size based on course such as the order in CS course trajectory or perceived difficulty of content. In contrast, courses typically required for a CS major had higher counts than those often offered as electives. For example, following CS1 and CS2 are Embedded Programming (often an Assembly course), and Programming Languages with eleven each compared to Data Science which has a count of four.

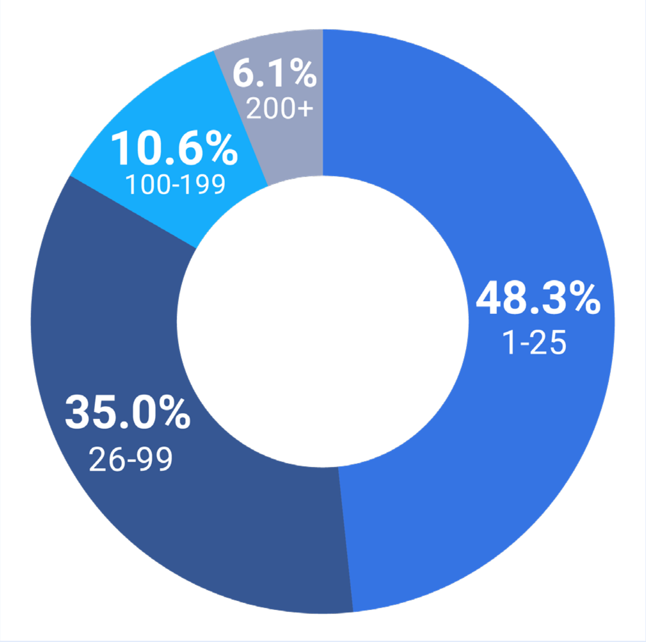
CS is taught in a range of classroom formats, from one-on-one tutoring to traditional university lectures to Massive Online Open Courses (MOOCs). In total, respondents taught 12,620 students over the 2018/2019 academic year. The average class size for respondents is 70 students, with the reported class size ranging from 7 to 2,000. Each instructor reported teaching an average of 2.6 classes.

Just under half (47%) of participating instructors taught at least one class of 50+ students, while 53% of respondents taught courses of only 49 or less during the 2018-19 academic year. We consider a class of 50 or more “large-scale” and teachers of those classes as “teaching-at-scale.”

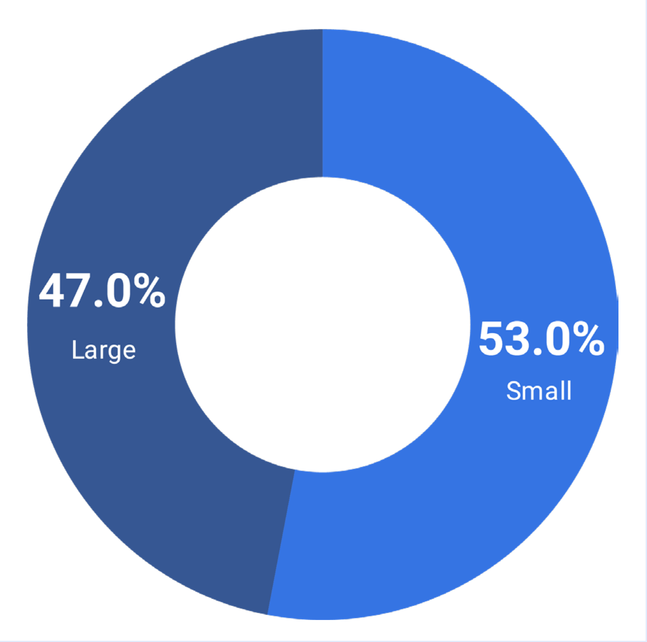
Students By Normalized Course Titles



Distribution of Class Size



Scale of Teaching



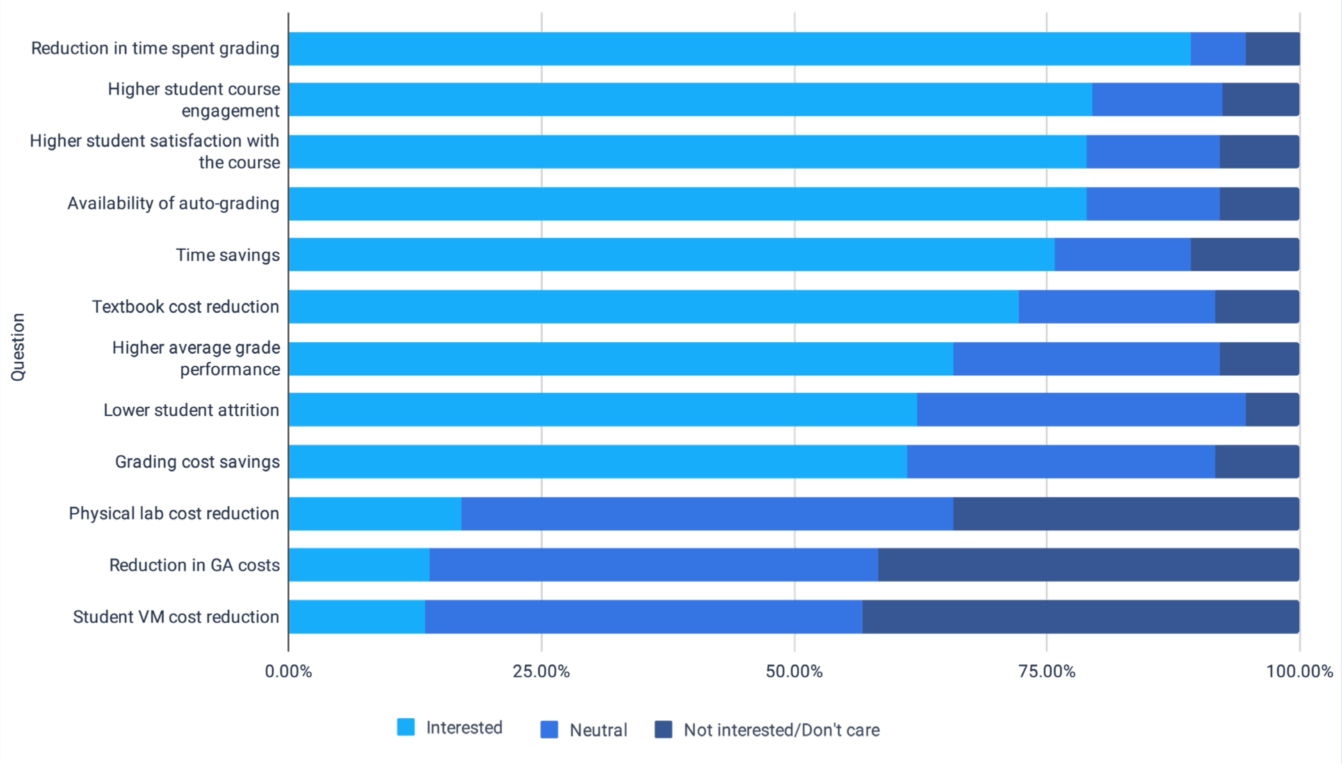
WHAT TECHNOLOGY BENEFITS ARE TEACHERS INTERESTED IN?

While shifting trends in CS education are causing higher adoption rates for technology, there is a non-trivial cost to this adoption process. Given this barrier to change, it is important to know which benefits (or the addressing of which pain points) make this effort worthwhile for a teacher.

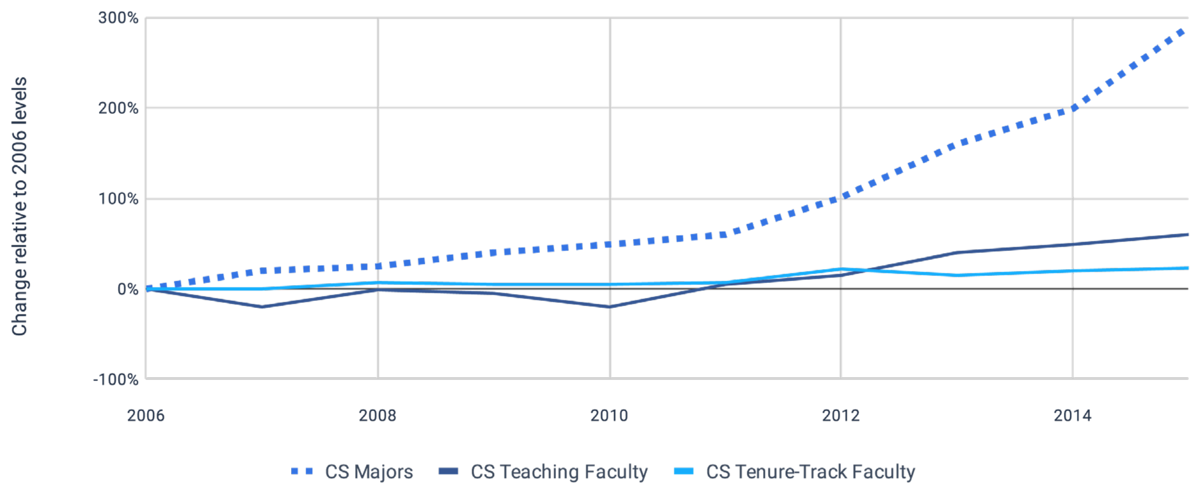
We asked teachers to indicate their interest level on 12 potential platform benefits. 89.2% were interested in reducing grading time. Similarly, 78.9% were interested in the availability of auto-grading. The second and third most interesting potential features were higher student engagement (79.4%) and higher student satisfaction with the course (78.9%). In the next section, we discuss the corresponding pain points in order of magnitude based on these results (grading/feedback, curricular challenges, and course administration).

While most instructors were rather disinterested in reducing the cost of student VMs (43% not interested), GAs (42% not interested), and physical labs (34% not interested), they did note interest in reducing the cost of textbooks (72% interested). In later sections, we explore who is paying for these resources to shed some light on this divide of interests.

Platform Benefits



Number of CS Majors, Teaching Faculty, and Tenure-Track Faculty Since 2006



Source: National Academics of Sciences, Engineering, and Medicine, 2018

EDUCATOR CHALLENGES AND NEEDS

For years, soaring CS enrollment has been widely reported as causing increasing demand for CS courses across the board. The current enrollment "boom" is causing a unique set of challenges for CS educators. Even with the movement to hire more teaching faculty, the large quantity of unfilled posted positions due to lack of supply is indicative of the exponentially growing gap between teachers and students. This is understandable when "approximately 57 percent of all new [Computer Science] Ph.D.s in North America take a position in industry" (National Academies of Sciences, Engineering, and Medicine, 2018).

To help clarify the challenges of this growing disparity between demand and resources, we asked participating instructors to rank their "pain points" or challenges in terms of feedback/grading, curriculum creation and setup, and course admin.

> **"Classroom space shortages, insufficient numbers of faculty and instructors to teach courses, and increased faculty workloads are among the top problems... [More] than one-third of the units reported concerns that student performance is declining... Many [schools] are clearly struggling to cope with their current enrollments in computer science courses. ... Current pressures on computer science [departments] are extremely difficult to manage and will also intensify if enrollments continue to grow."**
>
> — COMPUTING RESEARCH ASSOCIATION (2017)

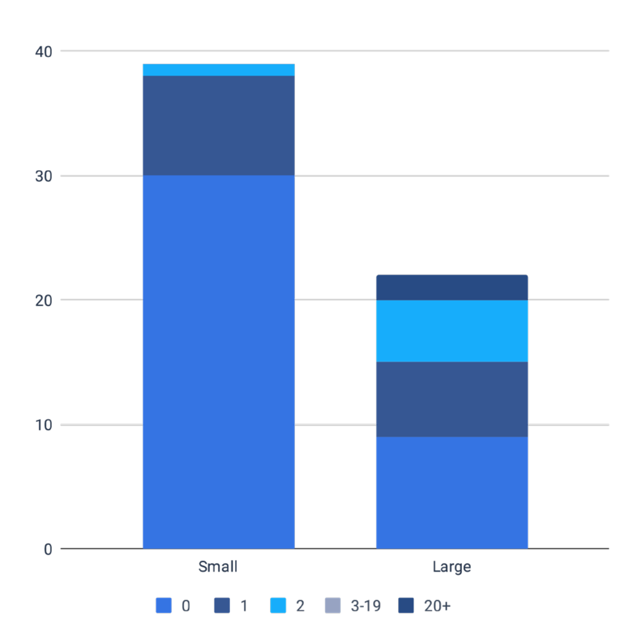
Grading Challenges

There are many challenges around grading: inadequate GA/TA resources, providing timely feedback to students, and notably in CS, the incompatibility of a student's development environment and the grading environment (resulting in the dreaded "but it worked on my machine!"). When asked to rank grading challenges, manual grading time was overwhelmingly ranked as the greatest pain point, with 72% of respondents placing it first.

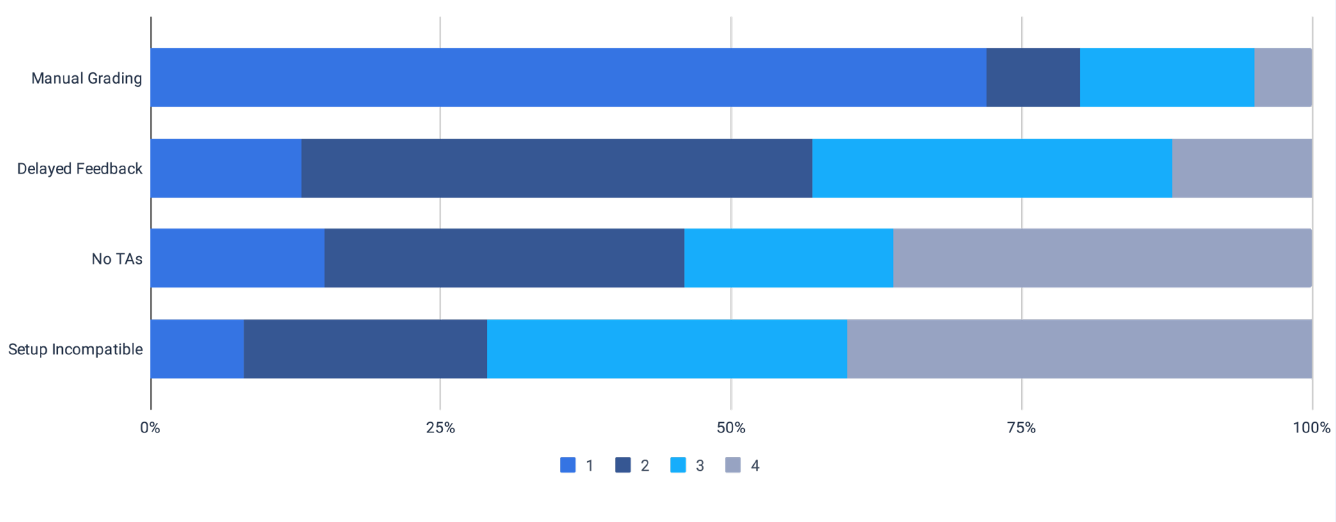
Manual grading is a bigger concern for those with small classes (< 50 students), with 79% saying it was their greatest challenge in terms of grading/feedback. 63% of instructors with large classes ranked manual grading time as their highest-priority pain point.

The majority of respondents (63.9%) lacked access to grading or teaching assistants. Of those who reported having GA/TA support in their largest class, 36.4% had two or more for the 2018/2019 academic year. The distribution of grading support looked strikingly different for teachers with under 100 students (small-scale) versus those with over 100 students (large-scale).

TA Resources for Large vs. Small Class Sizes



Ranking of Grading Challenges

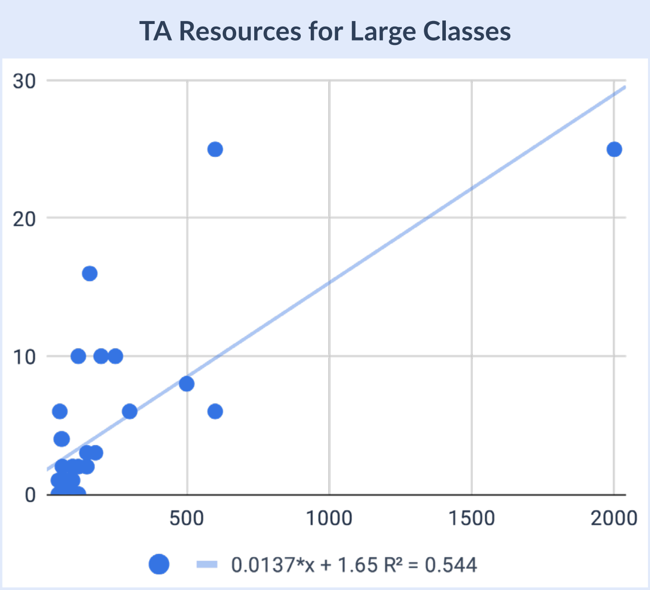


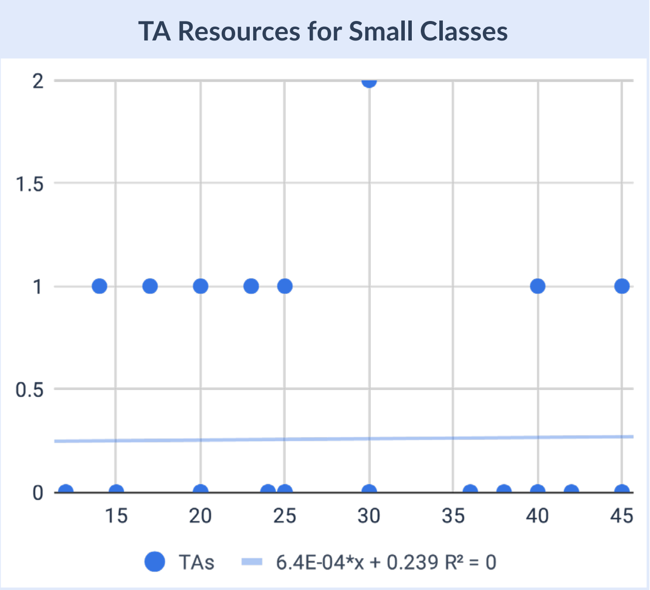
For respondents teaching small classes, there was no discernible correlation between the total number of students and the total number of TAs per professor. The majority of teachers in this group do not receive any TA support. This probably contributes to the higher concern amongst this segment than instructors in larger classes which often have grading help.

A weak correlation between total students and TAs for a professor was found for those teaching at scale. The model equation has a coefficient equivalent to a 1:73 TA to student ratio. Looking at the chart reveals a handful of teachers with courses well below that daunting ratio. This indicates that many teachers are overburdened and lack the necessary support, indicating a need for scalable technology as student-to-teacher ratios increase.

Anecdotally, we know that many departments have either homebrew submission systems or resort to manual grading. Some use built-in LMS features like rubrics, or existing free tools like codeworkout and web-CAT for auto-grading. Others use paid services like Gradescope (which was recently acquired by Turnitin, a plagiarism tool) which allows both manual and auto-grading. It is clear that there is a demand for an easy-to-use, auto-grading product that gives meaningful feedback to students.

> **“It is clear that there is a demand for an easy-to-use, auto-grading product that gives meaningful feedback to students.”**





Curriculum Challenges

The second and third most potentially interesting platform benefits—higher student engagement and satisfaction—are related to curriculum challenges. These challenges include not only low engagement but also high attrition, acting as technical support and configuring machines. When asked to rank these pain points, 27% of respondents indicated that low student engagement was the highest-priority pain point. When averaging the ranks, acting as technical support topped the list of challenges. While at first, these results might seem muddled, clarity arises when the data is split by scale.

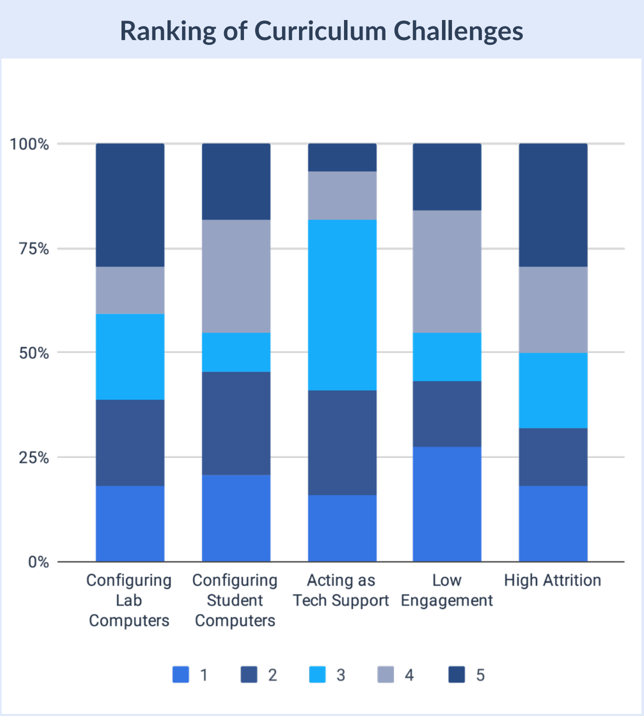
Looking at smaller scale teaching, configuring software on student machines placed in one of the top 2 spots for 50% tied with acting as technical support with 50% prioritizing it in one of the top 2 spots. These pain points are primarily technological in nature.

In contrast, those in larger-scale teaching contexts were primarily concerned about low student engagement/ satisfaction with 53% of respondents ranking it as the top or second challenge. This priority is primarily pedagogical in nature.

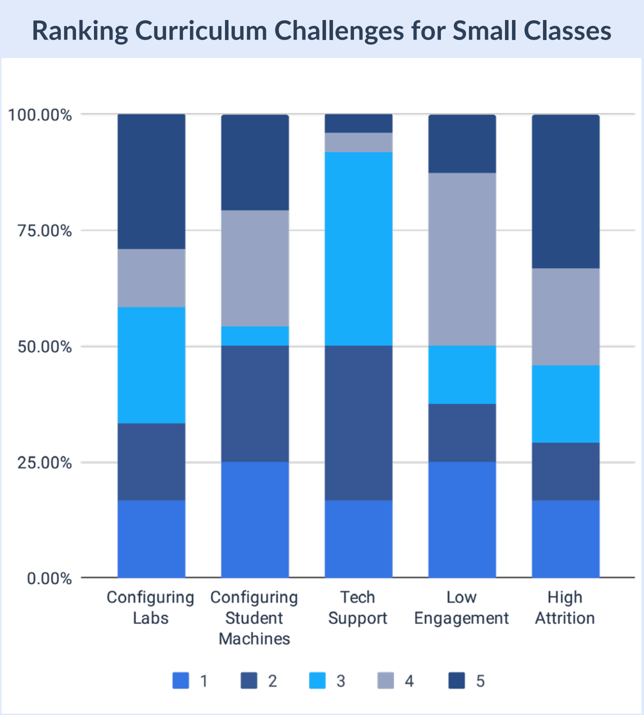
Slicing this data another way, if we consider configuring machines and acting as technical support to be part of "Technology Concerns" and low student engagement/ satisfaction and high attrition to be part of "Pedagogical Concerns" the difference is quite clear:

Large-scale teachers are more concerned about pedagogical challenges where small-scale teachers are more concerned with technological challenges. This may be due to the difference in resources available. As we saw with TA resources above, large-scale teachers often have more support. This external support could free the large-scale teachers of other concerns and allow them to focus on pedagogical concerns. As the distribution of these types of resources is often outside of the teacher's purview, this leads us to the third type of challenge—course administration challenges.

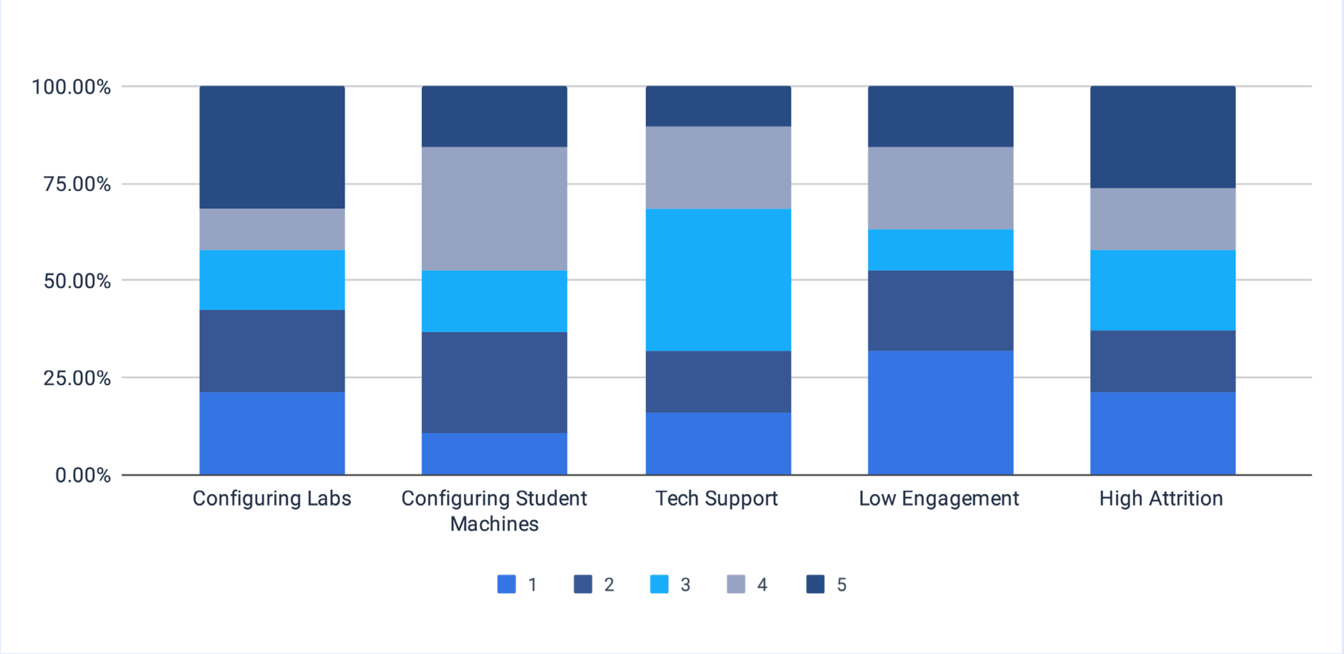
Ranking of Curriculum Challenges



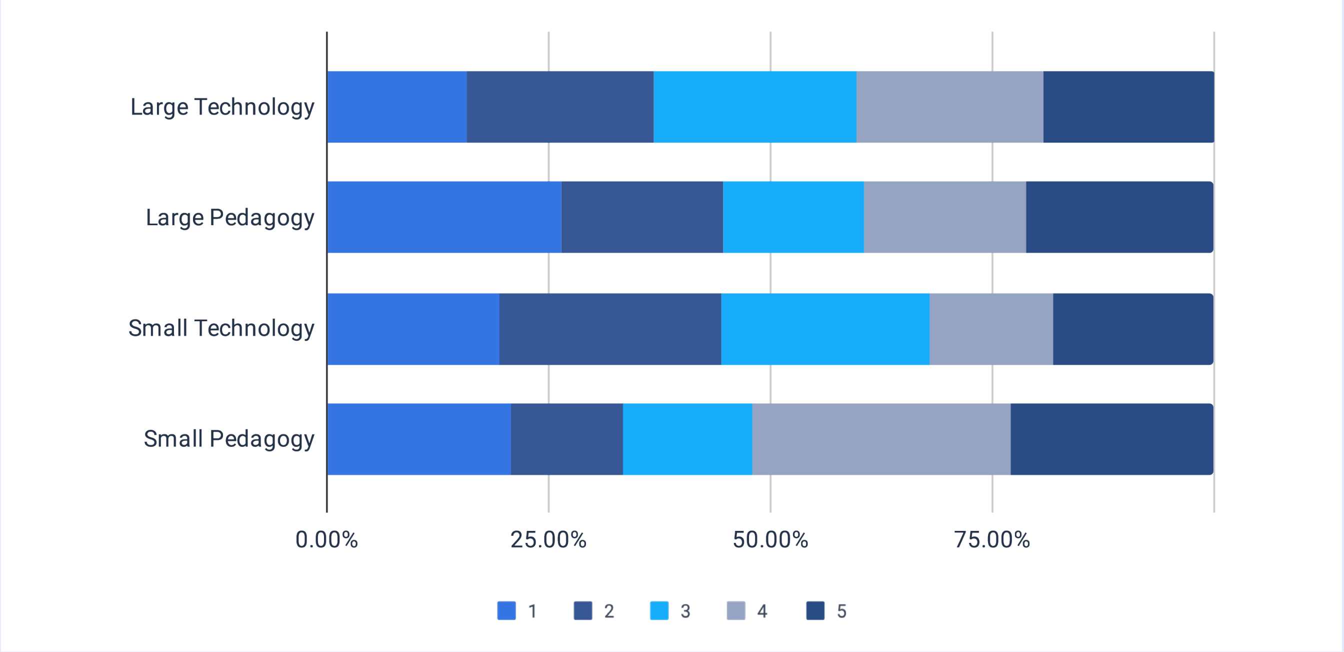
Ranking Curriculum Challenges for Small Classes



Ranking Curriculum Challenges for Large Classes



Type of Curriculum Challenge by Scale

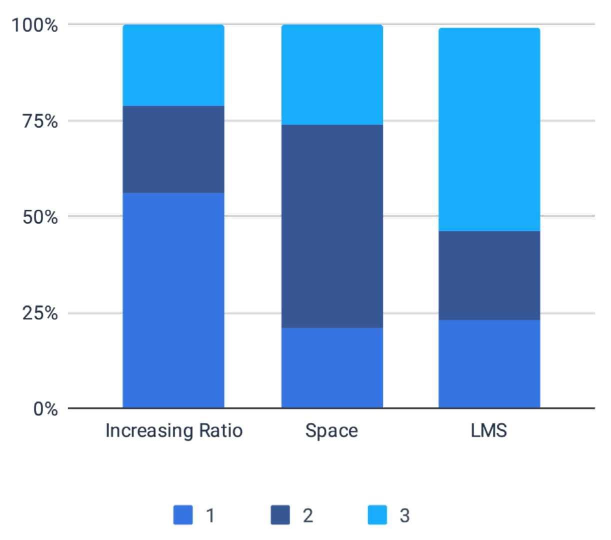


Course Administration Challenges

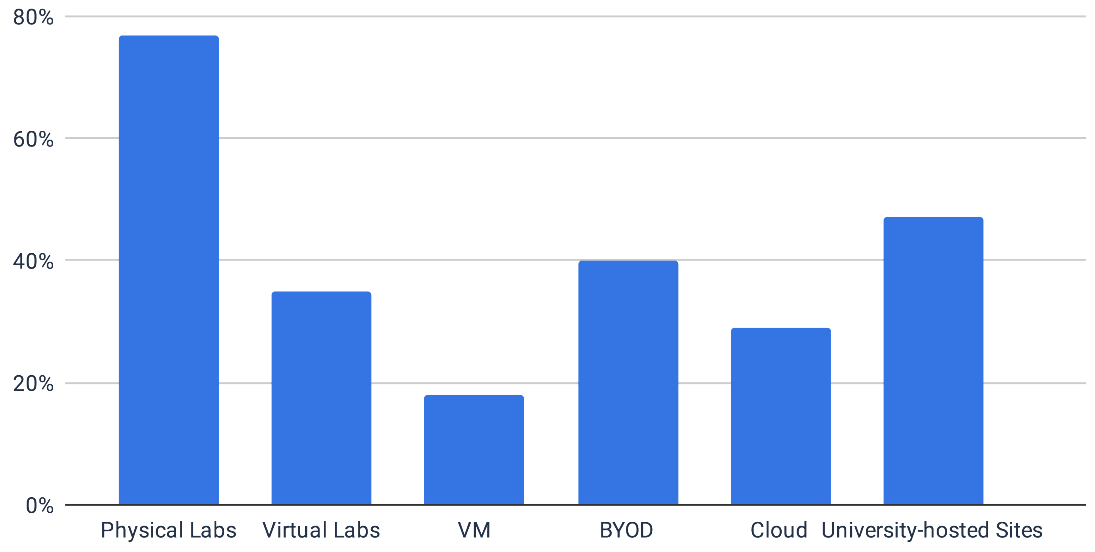
With 76% of teachers interested in “time savings” from technology, some of that pressure is probably due to course administration. Respondents were asked to rank increasing student-to-teacher ratios, mandatory LMS usage (Canvas, Blackboard, Moodle), and inappropriate classroom/lab space. 56% ranked increasing student-to-teacher ratios as their highest priority pain point. The 53% of respondents who indicated inappropriate space as their second concern have a 78% overlap of those with the first concern being increasing ratios—making these concerns related.

Physical labs are wide-spread, with 77% of respondents having them. A variety of digital options have emerged—with 58% reporting some combination of virtual labs, student VMs, or cloud options.

Ranking of Course Admin Challenges

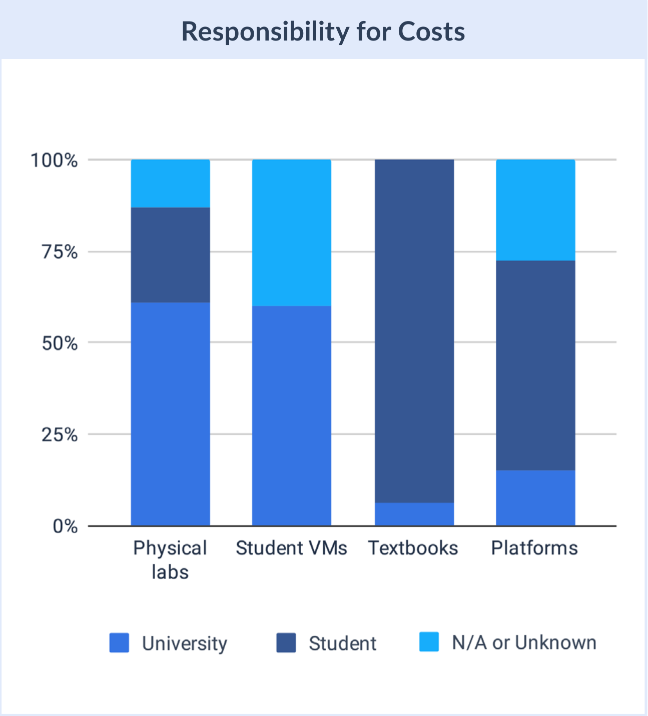


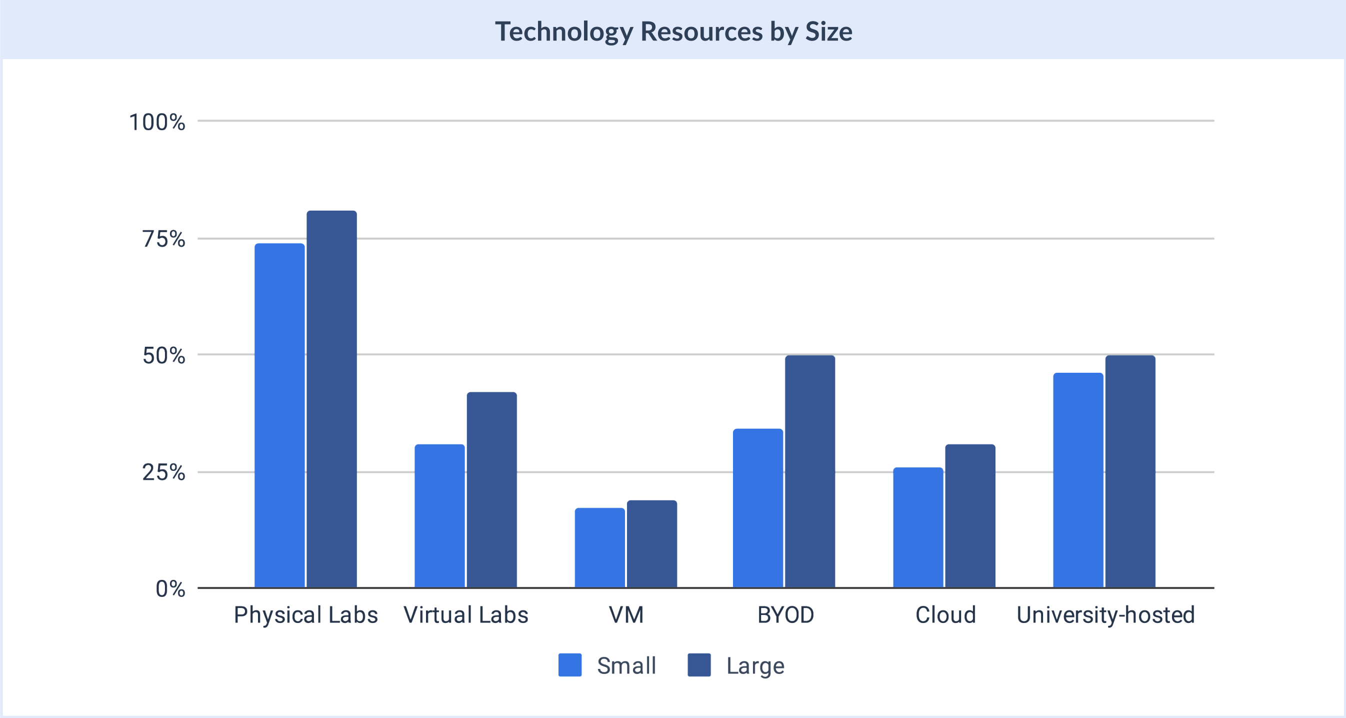
Reported Technology Resources

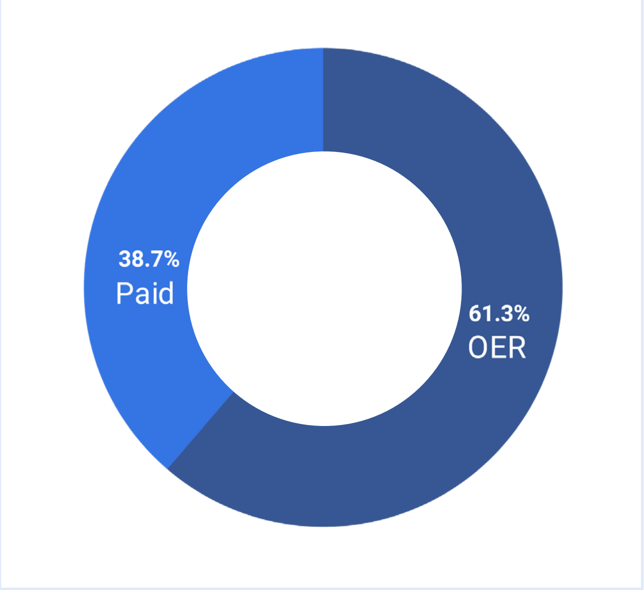


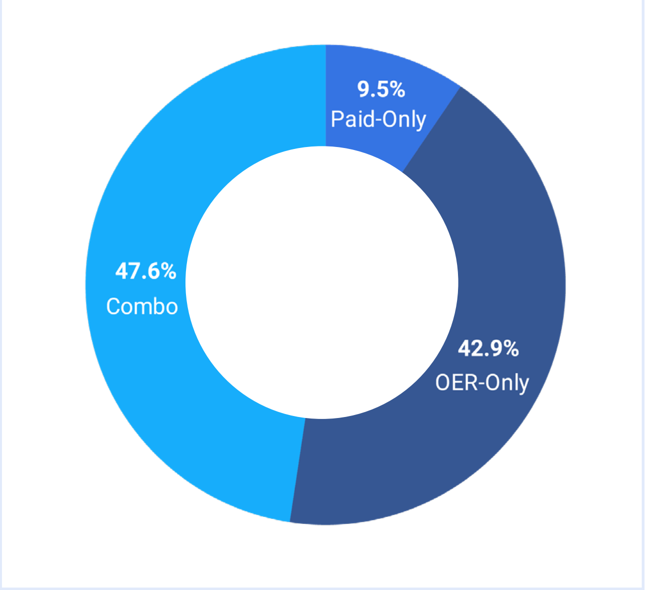
Looking at the distribution of these resources, 81% of those teaching-at-scale indicated their institution uses physical lab space. 74% of teachers with small classes indicated that their institution uses physical lab space. As we have seen—it appears that, across the board, at-scale teaching has access to more technological resources.

Overall, the majority of respondents' institutions cover the cost of the physical labs. 61% of respondents' institutions cover the cost of labs, while 26% say their students are responsible. Taking a look at student VMs, there were many who were unsure how the VMs were being paid for (40%), but a similar proportion of universities cover these costs (60%) compared to physical labs.






Resources Used


Combination of Resources Used

Summary of Challenges

Looking across the pain points above, we notice consistent differences in resources for those teaching at scale versus instructing smaller classes. This leads to those in smaller courses having other concerns over manual grading, struggling with technological challenges in the curriculum instead of pedagogy, and feeling disproportionately burdened with the increasing student-to-teacher ratios.

Curricular Resources Usage and Needs

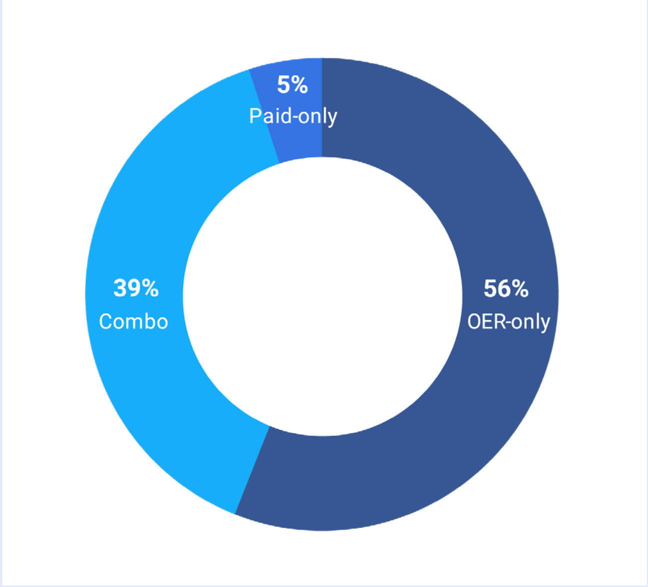
Slowly textbooks have lost market share as teachers create, customize, and adopt Open Educational Resources (OERs). Others have turned to paid platforms such as Zybooks or even digital tools created by those same textbook publishers. Looking at the distribution of reported resource usage, OERs were used to deliver course materials by a significantly higher proportion of respondents than paid resources.

9.5% of respondents reported using paid resources alone, while 43% reported using only OERs. 47.6% of respondents indicated that they utilized both paid and open educational resources to support their course-delivery. This large subset could be indicative of teachers' desire to customize content—52% factor in "ability to customize course materials" into their tech decisions (Top Hat, 2018). Most paid services which make money on the content itself do not allow editing, making OERs the more flexible option.

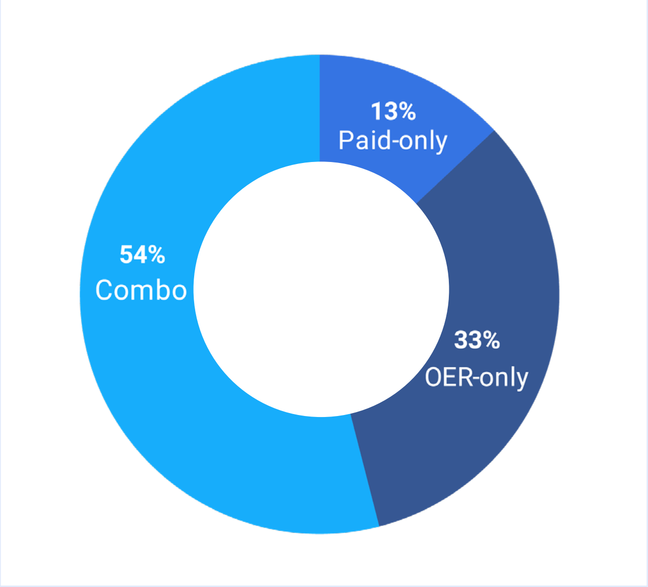
The majority of instructors teaching-at-scale used only OER materials. 95% reported using OERs alone or with paid resources. According to the survey results, teachers with smaller class sizes are more than twice as likely to use paid resources only.

Strikingly, of small class teachers, 33% reported using only OERs. The higher percent of paid resources (67% compared to 44%) could be due to several factors. Those teaching smaller classes might be at smaller institutions meaning that instead of teaching multiple sections of one course, each course they teach could need individual prep. This would exponentially increase the amount of time these teachers would need to curate and customize OERs, making off-the-shelf content appealing. Alternatively, these small classes could be highly specialized—making the availability of OERs much lower for those topics.

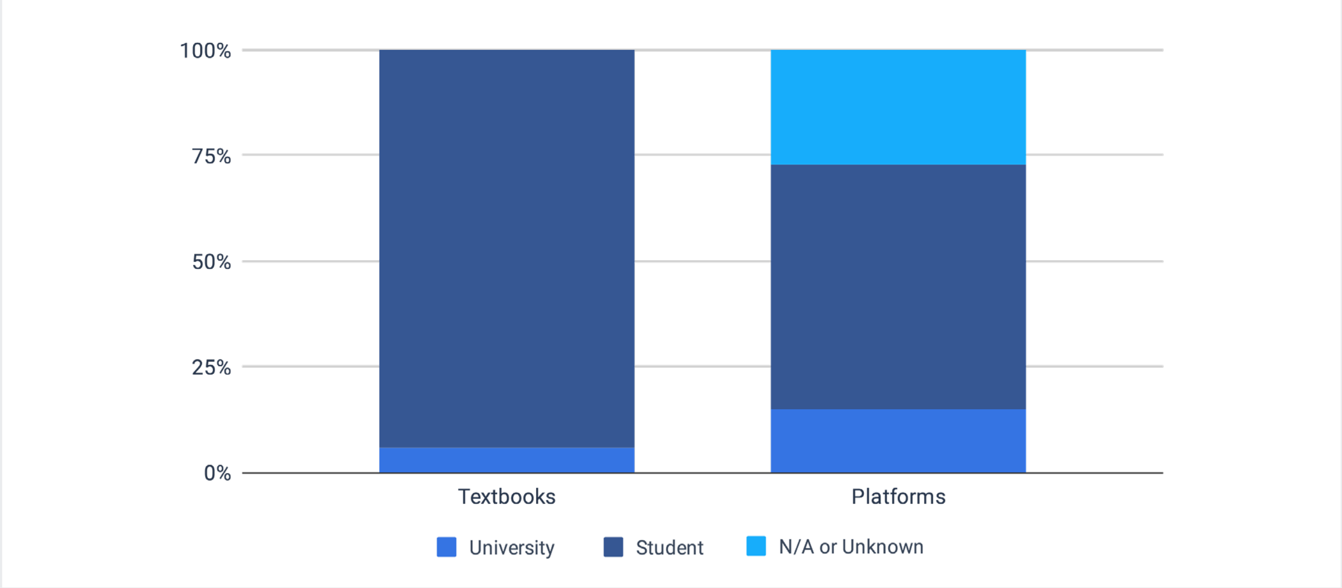
Combination of Resources Used by Large Classes



Combination of Resources Used by Small Classes



Purchaser of Curricular Resources

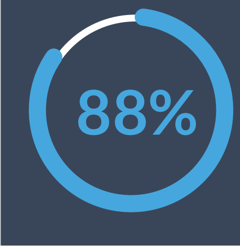


Textbooks remain widely used. However, they are expensive, and the high cost can have a detrimental impact on student satisfaction. 94% of students are responsible for paying for their own textbooks. No teachers-at-scale reported that their institution covered the cost of their students' textbooks. 58% of all respondents who use paid resources (whether in combination with OERs or not) said their students were responsible for covering the cost of "platforms that offer interactive textbooks."

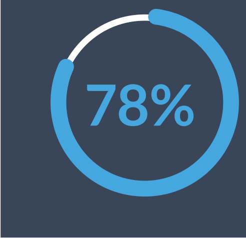
Technology Platform Needs

We now have an interesting conundrum—what platform benefits are professor's finding so useful they are willing to take on the cost of the platform? We asked teachers to denote each of the following features as Very Useful, Neutral, or Least Useful.

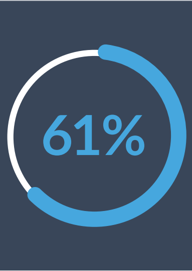
FEATURE



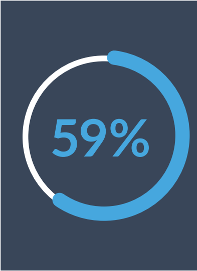
Editable assessments to enable immediate (auto-graded) feedback to the student on submission



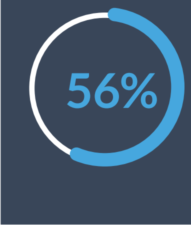
Assessments and exercises in the digital textbook that are set up to be auto-graded



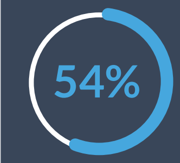
A dashboard of "progress" that allows you to track and review student progress and assessment completion



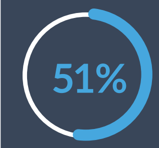
Editable textbooks with materials or exercises and assessments



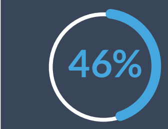
The ability for students to download and retain a full copy of the textbook in PDF form



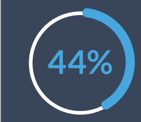
Textbook in digital form with the ability for students to edit, compile and run code examples within the resource



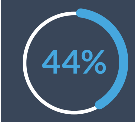
Use of an assessments library independent of the textbook originals that can be substituted or used to supplement the textbook (note that they can create editable copies)



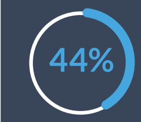
Textbook in digital form



Freeform "workspaces" (configured programming environments) for students to undertake longer form work independent of the textbook exercises

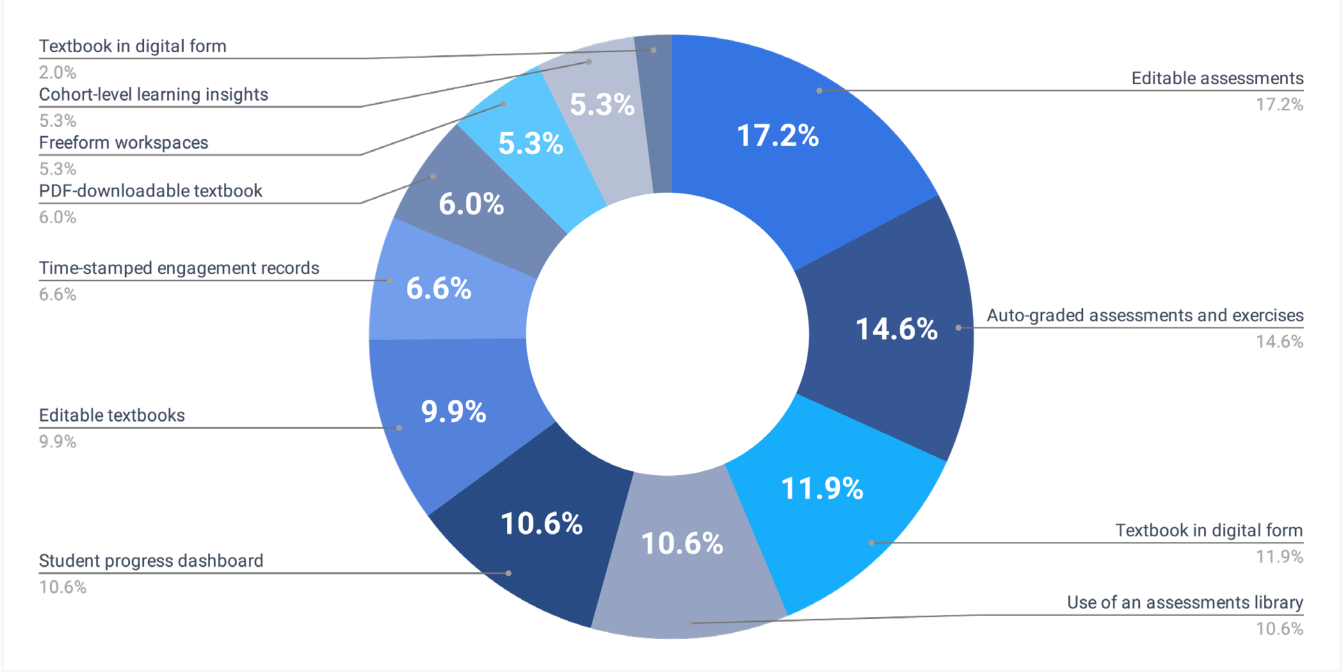


Cohort-level learning insights, e.g., most engaged, least engaged students, highest-achieving, lowest-achieving



A feature that provides a timestamp of engagement with each element (chapter, unit) of the teaching resources

Most Appealing Features



The majority of instructors were interested in curricular assessment resources, favoring editable, auto-graded questions over questions in text or a simple bank of questions. Similarly, instructors ranked editable text over a simple PDF or digital text. This reiterates teachers' increasing need to customize content for their context. It also emphasizes teachers' desire for technology to take on some of the grading burden—with much less interest in the analysis of those grades at any level larger than the individual student.

Finally, having considered the challenges, current resources, and the potential benefits in CS Higher Ed, we are left with the question: What would make the cost of switching worth it? With only 65% of teachers actively looking to or open to switching within the next 12 months, it is a high bar. We asked which of the above features would cause teachers to consider switching.

Many of the results were similar—notably, a downloadable PDF and the student dashboard dropped on teacher's priority lists meaning they are more "nice to have" rather than "need to have." A digital textbook that allowed students to write and run code jumped from 6th to 3rd on teachers' priorities list. This feature is interestingly specific to CS; thus, the desire might be due to many more general education platforms not currently supporting this feature.

Conclusion

There are several interesting trends revealed throughout this report. Teachers are feeling the increasing student-to-teacher ratio reported in the news. These pressures are felt most acutely in the increased grading burden, particularly in contexts without TA support. Similarly, teachers are feeling the strain of technology setup for small classes and pedagogically in large classes where attrition is increasing, and student engagement is falling.

The new wave of content platforms should allow teachers to modify content to their context and fill in perceived holes as indicated by the increasing use of supplemental materials being used in conjunction with paid resources. These tools should provide grading (notably auto-grading) support—not just to relieve the grading burden but also to give students rich and instant feedback on their understanding. Unique to our field, there is a willingness to switch products for a platform that allows students ample coding practice alongside the text—with the ability to create, edit, and run code.

Interestingly, there is a disparity in resources between institutions. This disparity appears along the dimension of class size, which probably predicts department and institution size. This means that price discounts by volume (a common practice) might exacerbate this inequity of access. This issue becomes more important given the shift of who is paying for these resources in some institutions. As more universities foot the bill for platforms and receive volume discounts, this puts smaller colleges at a disadvantage, unlike textbooks which the vast majority of institutions rely on students to purchase individually.

Help us continue this work.

Sign up to get emails about our next annual survey and report at

codio.com/survey

References

Arbeit, C. A., Bentz, A., Cataldi, E. F., and Sanders, H. (2019). Alternative and Independent: The Universe of T echnology-Related "Bootcamps." RTI Press Publication No. RR-0033-1902. Research Triangle Park, NC: RTI Press. **https://doi.org/10.3768/rtipress.2018.rr.0033.1902**

Chilana, Parmit K., et al. "Perceptions of non-CS majors in intro programming: The rise of the conversational programmer." 2015 *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). IEEE, 2015.*

Computing Research Association (2017). Generation CS: Computer Science Undergraduate Enrollments Surge Since 2006. **http://cra.org/data/Generation-CS/**

Denny, P., Becker, B. A., Craig, M., Wilson, G., & Banaszkiewicz, P. (2019, July). Research This! Questions that Computing Educators Most Want Computing Education Researchers to Answer. *In Proceedings of the 2019 ACM Conference on International Computing Education Research* (pp. 259-267). ACM.

Fincher, S. A., & Robins, A. V. (Eds.). (2019). *The Cambridge Handbook of Computing Education Research.* Cambridge University Press.

Johnson, E. (2019, August 2). "The $300 textbook is dead," says the CEO of textbook maker Pearson. Retrieved from **https://www.vox.com/recode/2019/8/2/20750863/john-fallon-pearson-education-textbook-digital-aida-teachers-kara-swisher-recode-decode-podcast**

National Academies of Sciences, Engineering, and Medicine. (2018). *Assessing and responding to the growth of computer science undergraduate enrollments.* National Academies Press.

Online Higher Education. Hyper Fragmentation and the US Post Secondary Market. (2019, August 1). Retrieved from **https://www.holoniq.com/notes/online-higher-ed.-hyper-fragmentation-and-the-us-post-secondary-market/?mc_cid=9a4eaaa11b&mc_eid=fa821c03c9**

Pitt, Rebecca; Farrow, Robert; Jordan, Katy; de los Arcos, Beatriz; Weller, Martin; Kernohan, D and Rolfe, V (2019). The UK Open Textbooks Report. Open Education Research Hub.

Seaman, J. E., & Seaman, J. (2017). Opening the Textbook: Educational Resources in US Higher Education, 2017. *Babson Survey Research Group.*

TIOBE Index for September 2019. (2019, September). Retrieved from **https://www.tiobe.com/tiobe-index/**

Top Hat. (2018). *Professor Pulse 2018*. Professor Pulse 2018. Retrieved from **https://tophat.com/wp-content/uploads/Professor-Pulse-Full-Report-Top-Hat.pdf**

codio

codio