



Delivering on Computer Science for All

By Ariel Rabkin

November 2016

Key Points

- Several major urban school districts are exploring some sort of universal computer science education; in New York, computer science will be available to all students, and in Chicago and San Francisco, it will be mandatory for all.
- High school graduation requirements and college application guidelines do not give sufficient credit for computer science work. Therefore, schools and students will have incentives to avoid computer science and favor other topics where achievement will be better recognized.
- While urban school districts will have an unusually easy time raising money, recruiting teachers, and engaging with innovative curriculum developers, smaller districts should be aware that adding computer science to the curriculum will pose considerable challenges.

There has been considerable political pressure in the past few years to incorporate more computer science education into American secondary schools. In his 2016 State of the Union message, President Barack Obama announced the goal of “offering every student the hands-on computer science and math classes that make them job-ready on day one.”¹ To achieve this goal, the president has requested several billion dollars in grants to schools for technology education and has channeled federal research money into improving computer science education.² Considerable amounts of money are going to be spent, and considerable curricular changes are being considered. The public, the policy community, and other parts of government should pay attention. We have reasonable prospects for a major improvement to American education—but also the opportunity to waste an immense amount of time, money, and opportunity.

Education, of course, is primarily controlled at the state and school-district level. We must look there to understand what concrete policies are likely to follow from grand national rhetoric.

Three major cities, New York, Chicago, and San Francisco, have separately announced some sort of “universal” computer science initiative. This paper will study these initiatives in detail. They are large, well-documented initiatives and are likely to set influential examples, whether they succeed or fail.

We begin by presenting the goals of these initiatives, the content they teach, and how they will be integrated into the school curriculum. Then the paper considers three related questions: Can these initiatives achieve their goals? Are they sustainable? Can they be replicated elsewhere? It closes with suggestions for policymakers seeking to emphasize computer science going forward.

There are grounds for cautious optimism. In the short to medium term, these initiatives have a good chance of getting many more students, particularly female students, into computer science. However, there will be difficulties in sustaining these programs and exporting them to other cities, unless high school graduation and college admissions standards are reformed.

Overview of the Initiatives

In December 2013, Chicago became the first major American school district to require computer science with the Computer Science for All initiative.³ New York and San Francisco followed two years later, announcing their own initiatives in the fall of 2015. All three programs have similar goals, rely on public-private partnerships for funding, make use of independently developed curricula, and have the ambition of integrating computer science comprehensively throughout the curriculum. As we will see, however, there are substantial differences in scope, ambition, and vision.

Goals. The backers of all three initiatives emphasize the economic opportunities that await students. The resolution introducing the computer science requirement in San Francisco has a lengthy statement of guiding principles. The first one was that “Understanding how computers work and how to creatively use this knowledge to solve problems will be increasingly important for all students as they prepare for college and their future careers.”⁴ Similarly in Chicago, the Computer Science for All explanation starts by noting that “computing skills lead to some of the highest paying careers.”⁵ Likewise, New York City notes that its “initiative responds directly to the needs of New York City’s local economy” for technology workers.⁶

All three cities explain that principles of equity require universal computing education (or at least universal access). San Francisco is especially emphatic on the point, making it an explicit goal that “students’ access to and achievement in computer science must not be predictable on the basis of race, ethnicity, . . . cultural affiliation or special needs.” Chicago and New York do not go as far, but they do justify their initiatives based on the goal of improved equity. Chicago notes that “minorities, women and children from low income families rarely have early exposure necessary” for high-paid technology jobs. And New York mentions the “diversity problem in tech across the nation.”⁷

Structure and Content. While these share similar rhetoric and goals, the actual policies vary considerably. Of the three initiatives, San Francisco’s is the most comprehensive and ambitious, attempting to reach every student in every grade within the

next few years, starting with a quarter-long course in middle school.⁸ New York’s is the least radical with its announced goal that every school should offer computer science by 2025—but enrollment is optional.⁹ Chicago’s falls in the middle by requiring one computing course before high school graduation.¹⁰

San Francisco is generating its own curriculum, but details have not yet been released. Chicago is relying on two externally developed curricula: Exploring Computer Science (ECS) and the more rigorous Advanced Placement (AP) course in computer science principles (CSP). New York is experimenting with a range of options, including ECS and AP CSP.

These two curricula differ considerably from past introductory courses in computer science and are therefore worth describing. Previous introduction to computer science courses, notably the previous AP computer science examinations, were largely about programming—students were expected to learn the practical skill of expressing algorithms in a particular programming language. The new curricula are considerably broader. Programming has been de-emphasized. Instead, these courses try to convey what programming (and computers in general) is for—to show students in detail how computing can help with a wide range of problems.

ECS is the better established of the two. Researchers primarily at the University of California, Los Angeles, developed it in an effort to build a high school curriculum more welcoming to female and ethnic minority students. It is organized around a series of activities: students spend six weeks developing web pages, six weeks building robots, six weeks collecting and analyzing data, and so forth. It has generated substantial student interest and has been adopted extensively within the Los Angeles school system. It has been offered at ever-larger scales since 2008. In the 2015–16 academic year, it was offered to 3,355 students at 43 schools.

AP CSP is a new course still in development. It is intended to supplement the existing AP computer science A exam, which focuses on programming and has been relatively unpopular compared with other AP examinations. The 2016–17 academic year is the first time the AP CSP course is being offered. However, it is based on numerous existing initiatives, at both the high school and college level, so there is a track record on which to evaluate the curriculum.

Students are expected to understand the ways that computers help solve practical problems, understand the general principles behind their operation (such as binary representation and digital logic), and engage with some of computing's social consequences.

The two curricula have radically different educational philosophies. AP students are supposed to *learn* and *do* more than ECS students. AP CSP, like every other AP course, has a fairly lengthy document specifying what students are expected to know. To receive credit for the course, students must pass an examination about the material. For the project component of the course, students must turn in a substantial computer program and a separate presentation and essay about the social consequences of some computing innovation. ECS, on the other hand, does not come with any standardized assessment. ECS emphasizes participation rather than a well-specified body of knowledge.

Funding. All three cities directly or indirectly rely on private philanthropy. New York and San Francisco receive direct private donations. The New York City Foundation for Computer Science Education is bearing half the costs of the city's Computer Science for All initiative, amounting to \$40 million in private money over the next 10 years.¹¹ San Francisco will fund its program with corporate donations from Salesforce, including \$6 million this year.¹² Only Chicago has not received large private donations so far. But even without direct donations, there is a considerable degree of indirect support for computer science education that helps all three cities. There are well-funded nonprofits operating at a national level that subsidize curriculum development, teacher training, and so forth. The organization Code.org stands out with its substantial roster of industrial donors.

How Universal Computer Science Could Succeed—or Fail

Now that I have described the three computer science education initiatives, it is time to analyze how well they might work if they are deployed in the ways their backers envision. Can these initiatives, or any other initiatives similarly constituted, achieve their backers' goals?

The Path to a Technology Job Goes Through College. All three initiatives' goals are to help students into technology jobs, particularly students from populations that are currently underrepresented in the industry. In the current state of the world, the pathway to a technology job typically goes through college. According to the Bureau of Labor Statistics, a bachelor's degree is the usual path to becoming a software developer, and an associate's degree is sometimes acceptable.¹³ This applies to virtually all information technology jobs.¹⁴ This means that helping underrepresented students into the software industry requires supporting their graduation from high school, encouraging them to apply to college, facilitating their transition to college, and attracting them to the relevant technical majors.

If computer science education can be offered without impairing graduation rates, that alone will likely increase the number of students entering technical jobs and reduce the gender disparity in computer science.

Evidence shows that offering computer science in high school does help with the last step of this path. College students who encountered computer science in high school—whether as an elective or as a requirement—are considerably more likely to become computer science majors.¹⁵ This effect is particularly strong for women. If computer science education can be offered without impairing graduation rates, that alone will likely increase the number of students entering technical jobs and reduce the gender disparity in computer science.

Limited Resources. However, there are risks to devoting resources to computer science that might make the first three steps—high school graduation, college acceptance, and college enrollment—harder. Schools have limited resources for hiring teachers and buying materials, and

computer science is an especially resource-intensive topic. Students have limited time and energy, so making room for a new topic means reducing other parts of their academic preparation.

Computer technology changes quickly, which means curricula will need regular updates.

The unhappy experience with the Algebra for All initiative shows the danger of overstuffing the curriculum. Starting in 1999 and continuing until 2010, California made a systematic attempt to intensify the middle school mathematics curriculum to ensure every eighth grader takes and passes algebra. Research suggests this effort was ineffective and counterproductive. Increased eighth-grade algebra enrollment is statistically associated with significantly decreased 10th-grade math scores, particularly in larger districts.¹⁶ Similarly, an effort in Chicago to increase the rigor of the ninth-grade mathematics and English curricula largely failed. As one study put it, “Although more students completed ninth grade with credits in algebra and English I, failure rates increased, grades slightly declined, test scores did not improve, and students were no more likely to enter college.”¹⁷ Schools and students have only finite capacity, so well-meaning but overly ambitious administrators can easily overfill the school day to the point where overall learning diminishes. A similar negative pattern could easily happen with universal computer science.

There are two particular reasons why computer science will be more resource-intensive for schools to offer than other subjects. First, computer technology changes quickly, which means curricula will need regular updates. ECS, for instance, has units on web design and data analysis. The appropriate software tools for these topics will change over time, requiring changes to lesson plans and instructional materials. Moreover, these topics may be much less central to computing in 10 years’ time than they are today. If the curriculum is not kept up-to-date, it may resemble a mathematics class of the 1990s emphasizing the use of the slide rule instead of the pocket calculator.

Second, computer science teachers will likely be more expensive to hire than other subject specialists. Because computer science and algorithmic thinking are richly rewarded commercially, computer science teachers will have better alternative employment options than most other teachers. Therefore, for the same reasons that schools want to teach computer science, they may have considerable difficulty and expense in doing so.

School officials point to resource limitations when explaining their decision not to offer computer science. A Gallup survey in 2014, conducted on behalf of Google, interviewed more than 10,000 school principals and district administrators. Those without computer science classes in their schools or districts predominately cited the need to prioritize subjects that have high-stakes tests or requirements. The second and third most-cited reasons were the lack of qualified teachers and the expense of hiring them.¹⁸

Students May Not Be Rewarded for Computer Science. In addition to the computer science education costs (a supply-side problem), sustaining student interest (a demand problem) may be a challenge. This particularly applies in New York, where enrollment is optional. But even in cities with mandatory computer science courses, students will put more time into courses they think have greater importance. To the extent that computer science grading is nonrigorous, students may be tempted to slack off and put in only minimal effort for mandatory computer science classes.

The main barrier here is likely to be college admissions requirements. Selective colleges typically require applicants to have taken sufficiently many rigorous courses in high school. These requirements do not typically mention computer science. Consider just the major public universities in the relevant states. The University of California (all campuses) expects a minimum of three years of mathematics but prefers four. Computer science does not count for either math or science.¹⁹ Likewise, the University of Illinois requires math and science and specifies that “computer courses are not acceptable.”²⁰ In New York, different state schools set their own requirements, but “three or four years of mathematics” is typical.²¹ We might hope that as computer science education spreads, universities will change their admissions practices, but AP computer science is

decades old, and admissions policies have yet to incorporate it.

Are There Helpful Synergies? If every hour of computer science instruction simply subtracts an hour from other subjects, students may experience limited overall benefit. Computer science instruction will far more likely achieve its aims if the time and energy spent on computer science has some spillover benefit to student achievement in other topics.

Synergies of this sort are surprisingly hard to find. Good studies of primary- and secondary-level computer science are rare, and the results we do have are equivocal.²² A study on elementary school students tested 12 distinct combinations of computer science instruction and other-subject gains. For two of the combinations tested, there were gains, and for one, a loss; the rest showed no significant effect. A preliminary College Board study showed that taking the AP computer science examination was correlated with higher scores on the SAT mathematics test and other mathematical AP examinations, even after controlling for prior achievement and demographics. However, the researchers caution that “these results cannot be interpreted as causal. . . . [It is] highly likely that there are additional confounding factors omitted from the model.”²³ This is a more encouraging result than if there were no correlation between computer science education and mathematics achievement. But a study that did not even convince its authors is a weak basis for a major education reform.

New York is the city most likely to have beneficial crossover between computer science and mathematics. New York is piloting several different curricula to prepare students for the AP CSP examination. One curriculum being piloted is Bootstrap,²⁴ a specialized programming language (based on Scheme) designed to harmonize well with mathematics instruction. In most programming languages, variables and functions behave somewhat differently than the concepts of the same name in mathematics. In Bootstrap, these terminology mismatches have been carefully minimized. The Bootstrap development team has shown that their curriculum considerably improves student performance on some math questions as compared with a control group.²⁵ No other computer science initiatives have this sort of evidence.

Sustaining Universal Computer Science

Even if universal computer science is a success in the near-term, it may wither over time. This section describes and evaluates the institutional mechanisms and incentives that will either underpin or undermine computer science education going forward. Computer science will be a resource-intensive subject for the foreseeable future; the major question is whether there are mechanisms to ensure that adequate resources are devoted to it.

There are several mechanisms that encourage schools to fund particular programs. One is parental (or generally, voter) demand. Another is performance on standardized examinations (particularly state examinations) and university entrance requirements. While computer science is likely to be popular with parents, state curriculum policy will actually work against it.

The relevant states do not require computer science for high school graduation; indeed, they do not even give credit for it. California has a state high school exit exam to test mathematics and English proficiency; computer science is not included. Illinois has coursework requirements for high school graduation. AP computer science is officially recognized as a mathematics course, but the state is silent about other computing courses.²⁶ It is therefore unclear whether the ECS option in Chicago would fulfill the state requirement. High school graduation requirements in New York are controlled by the state board of regents. A basic high school diploma requires three years of mathematics, of any kind. (The state is silent about whether computer science would count.) For the more prestigious advanced regents diploma, the student must take three years of the state-approved math curriculum, backed by the state mathematics tests.²⁷ Computer science is not part of the regents curriculum, and so an ambitious student would risk falling behind if they took computer science instead of the regular mathematics course. A school seeking to boost the fraction of advanced-diploma graduates would have an incentive to steer students away from computer science.

While state requirements will not encourage schools to supply computer science education, student and parent demand may do so. The 2014 Gallup survey²⁸ found that 76 percent of parents making less than \$54,000 per year thought that

most students should be required to learn computing; for higher-income households, 60 percent felt that way. And 85 percent of all parents thought computing was as important as or more important than traditional required courses such as mathematics and science. This suggests deep public support for these programs.

Seventy-six percent of parents making less than \$54,000 per year thought that most students should be required to learn computing; for higher-income households, 60 percent felt that way.

Parents will have some ability to compare the quality of computer science education in different schools. At present, the only standardized exams for computer science are the AP and International Baccalaureate exams. These examinations, particularly the AP exam, are highly suited to assessing educational effectiveness of the students who take them. Average AP exam scores are reported at the school and district level, and the College Board uses elaborate statistical techniques so that scores can be compared from year to year. As a result, it will be possible to measure progress over time and to assess the relative success of different schools in New York and Chicago. San Francisco, with its homegrown curriculum, does not have a calibrated examination, so this aspect will be lacking.

If the universal computer science education programs succeed in the short term, and if parents see that these programs really do help students attend college and find jobs, then they may build up considerable momentum. New York and Chicago have an advantage because a high school requirement will be visible more quickly than the middle school initiative that San Francisco is deploying. If San Francisco extends its initiative to high school (as planned), this gap should close.

Exporting Universal Computer Science

Even if universal computer science works in the cities that are now introducing it, there will be barriers to spreading it more widely. Certainly, magnet schools and affluent districts routinely offer computer science today, and these schools could make it mandatory. But precisely what is interesting and ambitious about the initiatives discussed in this paper is that they are *not* in affluent suburban districts; they are in urban districts where getting students to graduate at all is a challenge.

The districts discussed in this paper have many challenges, but they also have advantages that most American school districts do not. They are all in large cities with reasonably strong economies. New York and San Francisco have two additional advantages: a major concentration of technology companies and a tradition of private donation to support civic functions. San Francisco can raise millions of dollars per year in donations to fund teacher training and can bring in volunteers from Silicon Valley companies. New York has similar access to money and volunteers, and even Chicago has a considerable technology industry. Most American cities cannot raise money or volunteers close to the same scale. In addition, when a large city announces a novel initiative, education researchers and nonprofits notice and offer their services; when a small undistinguished city tries to follow their example, comparable support may not be forthcoming.

There is also a structural difference between the cities discussed here and most others. In New York and Chicago, ambitious and powerful mayors have pushed for the computer science initiatives. This propulsive force is unlikely to be as strong elsewhere. New York and Chicago are unusual in giving their mayors control over the school system;²⁹ most school districts are run by boards. Having a powerful mayor backing an education reform may help resolve problems that would thwart an appointed chancellor who answers to a risk-averse school board.

Conclusions

Several of America's biggest cities are conducting a large and ambitious experiment, adding a new major subject to the standard educational track. There are reasons for optimism, particularly for

New York and Chicago, which have the benefit of credible independent curricula and economies of scale. Even San Francisco, while hampered by its exceptionally ambitious goals and go-it-alone curriculum, might succeed, bolstered by the technology industry's wealth and human capital. These initiatives have reasonable odds of putting numerous students on the path to a good technology job who otherwise would not be on that path. They also have reasonable odds of reducing the gender gap in computing, which is a major announced goal of the cities involved.

There are grounds for pessimism about whether these programs can be institutionalized and maintained. The relevant states have not incorporated computer science into their graduation requirements. Of the states that have high-stakes, standardized tests for high school students, none include computing. As a result, schools will be constantly under pressure to shortchange computer science on behalf of subjects that are more intensively monitored.

There are a few public policy changes that would strengthen universal computer science programs. It would be helpful if computer science were considered equivalent to a science or math course for high school graduation and college entrance purposes. This requires not only making a change but also making it public—student behavior is influenced not just by what colleges do, but by what students, guidance counselors, and other mentors expect the colleges to do. State governments could require, or at least urge, public colleges to make these changes; we might reasonably expect private colleges to follow suit once students and schools shift to a new equilibrium.

There are also actions that the wider policy community, and particularly the boosters of these programs, should take. We should be honest that comprehensive computer science comes with costs, both financial and in terms of what other subjects will need to be abbreviated. These costs may be worth paying, but they are seldom acknowledged today, even though they should be. We should watch for ways to minimize these costs. To the extent that clever curriculum design can teach computing and other subjects simultaneously, this should be a high priority.

About the Author

Ariel Rabkin (asrabkin@gmail.com) is a visiting fellow with AEI's Center for Internet, Communications, and Technology Policy and a professional software engineer. He has published more than a dozen technical papers on topics including system monitoring, configuration management, and efficient analytics.

Notes

1. Mentioned in the 2016 State of the Union Address. See White House Office of the Press Secretary, "Remarks of President Barack Obama — State of the Union Address As Delivered," January 13, 2016, <https://www.whitehouse.gov/the-press-office/2016/01/12/remarks-president-barack-obama-prepared-delivery-state-union-address>.
2. Megan Smith, "Computer Science for All," the White House, January 30, 2016, <https://www.whitehouse.gov/blog/2016/01/30/computer-science-all>.
3. Choose Your Future, "What We Are Doing," Chicago Public Schools, 2016, <https://chooseyourfuture.cps.edu/computer-science-for-all/what-we-are-doing/>.
4. San Francisco Unified School District Board of Education, Resolution 155-26A2, June 9, 2015, <http://www.sfusd.edu/en/assets/sfusd-staff/about-SFUSD/files/board-agendas/Agenda%206%209%202015.pdf>.
5. Choose Your Future, "What Is CS4All?," Chicago Public Schools, 2016, <https://chooseyourfuture.cps.edu/computer-science-for-all/what-is-cs4all/>.
6. City of New York, "Computer Science for All: Overview," 2016, <http://www1.nyc.gov/office-of-the-mayor/education-vision-2015-facts.page>.
7. Ibid.
8. Liana Heitin, "San Francisco's Cutting-Edge Plan: Bring Computer Science to All Pre-K–12 Students," Education Week Curriculum Matters, August 12, 2016, http://blogs.edweek.org/edweek/curriculum/2015/08/san_francisco_to_bring_computer_science_to_all_prek_12_students.html.
9. Kate Taylor and Claire Cain Miller, "De Blasio to Announce 10-Year Deadline to Offer Computer Science to All Students," *New York Times*, September 15, 2015, <http://www.nytimes.com/2015/09/16/nyregion/de-blasio-to-announce-10-year-deadline-to-offer-computer-science-to-all-students.html?>
10. Chicago Public Schools, "2016–2017 High School Graduation Requirements," 2016, http://cps.edu/SiteCollectionDocuments/HSGraduationReq_English.pdf.
11. New York City Foundation for Computer Science Education, "Computer Science for All," 2015, <http://www.csnyc.org/computer-science-all>.

12. Salesforce.com, “Mayor Lee, San Francisco Unified School District and the Salesforce Foundation Announce Third Year of Partnership to Expand Computer Science Opportunities in San Francisco Schools,” press release, August 24, 2015, <http://investor.salesforce.com/about-us/investor/investor-news/investor-news-details/2015/Mayor-Lee-San-Francisco-Unified-School-District-and-the-Salesforce-Foundation-Announce-Third-Year-of-Partnership-to-Expand-Computer-Science-Opportunities-in-San-Francisco-Schools/default.aspx>.
13. Bureau of Labor Statistics, “Software Developers,” Occupational Outlook Handbook, December 17, 2015, <http://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm>.
14. Bureau of Labor Statistics, “Computer and Information Technology Operations,” Occupational Outlook Handbook, December 17, 2015, <http://www.bls.gov/ooh/computer-and-information-technology/home.htm>.
15. Jennifer Wang et al., “Gender Differences in Factors Influencing Pursuit of Computer Science and Related Fields,” 20th Annual Conference on Innovation and Technology in Computer Science Education, June 2015, <http://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43820.pdf>.
16. Thurston Domina et al., “Aiming High and Falling Short: California’s 8th Grade Algebra-For-All Effort,” *Educational Evaluation and Policy Analysis* 37, no. 3 (July 20, 2015), <http://emilykpenner.com/epkwp/wp-content/uploads/2015/08/Domina+McEachin+Penner+Penner+2014+EEPA+Aiming+high+and+falling+short.pdf>.
17. Elaine Allensworth, Takako Nomi Nicholas Montgomery, and Valerie E. Lee, “College Preparatory Curriculum for All: Academic Consequences of Requiring Algebra and English I for Ninth Graders in Chicago,” *Educational Evaluation and Policy Analysis* 31, no. 4 (2009): 367–91.
18. Google, *Searching for Computer Science: Access and Barriers in US K–12 Education*, fig. 14, https://services.google.com/fh/files/misc/searching-for-computer-science_report.pdf.
19. University of California, “A–G Courses,” 2014, <http://admission.universityofcalifornia.edu/freshman/requirements/a-g-requirements/index.html#math>.
20. University of Illinois, “Acceptable High School Courses,” 2016, <https://admissions.illinois.edu/Apply/Freshman/hs-courses>.
21. Of major universities in New York, Stony Brook, Cornell, and Columbia expect three years of mathematics for all students and four for technical students. Computer science is mentioned by none of these as an acceptable alternative.
22. Code.org, “June 2014–June 2015 Program Report,” 2015, <https://code.org/about/evaluation/learning>.
23. Jack Buckley, “Preliminary Results of AP Computer Science Analyses,” College Board, July 2015, <https://code.org/files/CollegeBoardPreliminaryCSMemo.pdf>.
24. Michael Preston and Leigh Ann DeLyster, “CSP Joins New York City’s Computer Science Portfolio,” *ACM Inroads* 6, no. 4 (December 2015): 67–70.
25. Emmanuel Schanzer et al., “Transferring Skills at Solving Word Problems from Computing to Algebra Through Bootstrap,” Proceedings of the Special Interest Group on Computer Science Education Conference, 2015, <http://www.ccs.neu.edu/racket/pubs/sigcse-sfkf.pdf>.
26. Illinois State Board of Education, “State Graduation Requirements,” February 2016, http://www.isbe.net/news/pdf/grad_require.pdf.
27. New York State Higher Education Services Corporation, “Regents Requirements,” <https://www.hesc.ny.gov/prepare-for-college/your-high-school-path-to-college/regents-requirements.html>.
28. Google, *Searching for Computer Science*, fig. 10.
29. Kenneth K. Wong and Francis X. Shen, “Mayoral Governance and Student Achievement: How Mayor-Led Districts Are Improving School and Student Performance,” Center for American Progress, March 2013, <https://cdn.americanprogress.org/wp-content/uploads/2013/03/MayoralControl-6.pdf>.

© 2016 by the American Enterprise Institute. All rights reserved.

The American Enterprise Institute is a nonpartisan, nonprofit, 501(c)(3) educational organization and does not take institutional positions on any issues. The views expressed here are those of the author(s).