Google

Ramaiah IT uses Applied CS to diversify mobile programming curriculum

RAMAIAH
Institute of Technology

At a Glance

What they wanted to do

- Capitalize on existing enthusiasm around Applied CS on campus with students by bringing it into a formal, for-credit setting
- Create an industry-inspired curriculum that introduces application-oriented courses into traditional CS syllabi

What they did

- Created a project-based learning course using the Applied CS content that was part of the regular CS course curriculum
- Provided code labs with detailed guidelines for every task required by the Applied CS course that kept students motivated
- Developed evaluation methods to help students monitor their performance and succeed

What they accomplished

- Offered a for-credit Applied CS course for nearly 220 students
- The Applied CS work has led to a number of students getting internships at Google and relevant tech companies and startups

Challenge

Ramaiah Institute of Technology (RIT) in Bangalore, India, is ranked as one of the top engineering institutes among more than 170 colleges that are part of Visvesvaraya Technology University. CS courses are among the most popular across campuses, but focus primarily on teaching theory and the fundamentals. Professors don't always have the flexibility in curricula and the right set of resources to offer practical programming assignments and practice for a range of students at different skill levels. The result is students often graduate without the applied coding experience they need to be successful in the job market.

Solution

RIT got its first taste of Applied CS with Android (now Applied CS Skills) in August 2015, when RIT students and former Google Student Ambassadors led four extracurricular workshops over one month. With the Android workshops gaining even more popularity and interest through student facilitators in 2016, RIT faculty decided to create a formal Applied CS course that was part of the CS curriculum during Spring semester 2017. The for-credit course was based on the curriculum created by Google and taught by three faculty members with the aid of teaching assistants.

Faculty added a few sessions on Android Studio installation and basic Android at the beginning and picked the units that they thought were most relevant. Taking advantage of the flexibility of the program content being organized in units, the faculty built upon the base curriculum from each unit selected to add more practice using a variety of real world problems as applications for the unit's data structure. For example, for the Anagrams unit (covering the HashMaps data structure), faculty added a module to implement a spell check program that used HashMaps in its solution as well. They used a "flipped classroom" model, where students completed much of their reading and review of basics at home and spent classroom time "pair programming," working on problems in small groups. The units focused less on the intricacies of a specific programming language and more on broad concepts and their real-world application.

For grading, faculty broke up the course into four key components: 10 percent submitting code samples for each unit completed; 20 percent for short programming tests covering the CS concepts; 10 percent for a midterm quiz; and 60 percent for the final project, developed as a part of the Applied CS Code Sprint unit. About 40 teams of students developed apps in groups of 2-3 for the Code Sprint, which involved a range of CS concepts and technologies from the program, including Firebase and Google Maps.

"Because the Applied CS course had so much collaboration, the college is open to a more modern syllabus that integrates project-based learning. The course made me and my colleagues interested in the idea of flipped classroom, as well as a course delivery method, something we had been thinking about for a while but had never taken the chance to implement."
—Dr. Mydhili Nair, Computer Science Professor, RIT

Benefits

Using CS concepts to visualize functionality
To learn how and when to apply certain CS principles, students reviewed the programming concepts and completed the Applied CS modules every week. They approached the applications of theoretical CS in a way that allowed them to visualize the functionality they were creating by seeing the outcome of their coding efforts in real time – something typically missing in traditional programming theory courses. When students bugs prevented the code from running, they would be able to dissect the code line by line to understand what went wrong, make changes and try it again.

Evaluations and customized support keep students motivated
Because the content was already provided by Google, the faculty could instead focus their time and energy on developing tools and systems to ensure students were supported and evaluated effectively. To accommodate the broad range of student skills, the faculty established three clearly-defined levels that students could reach based on their performance on a series of practical and theoretical tests. In addition, the faculty and teaching assistants built an automated grading system – part written subjective and part computer-based Android programming – to gauge student progress throughout the course and help identify tailored areas for development for each student. If students had areas of struggle, they could join instructor-led code labs that had more detailed step-by-step guidelines, including code snippets, for building Applied CS activities.

Future

The specialized learning and coding projects completed as part of the Applied CS program helped a number of students get internships at technology companies. "The students benefited from the structure of a curriculum that came from an industry expert, Google, and focused on leading mobile technology, Android," says Arjun Rao, an Applied CS teaching assistant.

Applied CS has influenced the type of classes that are being offered in CS in general, using project-based learning to reinforce theory. In the future, faculty will be introducing a new course called "Scripting Languages," that will cover Python and JavaScript similarly using a practical, project-based approach.

Tailored learning to simulate real-world challenges

The Applied CS course was designed with a broad set of problems for students to tackle in order to simulate coding environments and challenges that engineers face in tech jobs, but also is customizable to meet specific faculty interests and student needs.

Faculty and teaching assistants at RIT created adaptations and additional challenges for their students to reinforce learnings in each unit. For example, in Anagrams using HashMaps, students who were struggling with understanding or building the word game were asked to use a shopping metaphor instead, where they could visualize pictures of clothing items for sale and prices instead of Anagrams/word possibilities. Additionally, in the Ghost unit (also word game) using the Trie data structure, faculty challenged students to create a "fast phone book" that required the use of Tries to retrieve phone numbers that start with a particular sequence.