Good afternoon! My name is Qitian. Today I would like to introduce Computer Based Testing. When we first hear the term, cbt, there might be several misconceptions. It does not mean that we have all multiple-choice questions, or the students take exams at home. It also does not imply 100% machine grading of exams, or webcam based remote proctoring. //

The most important concepts of cbt are question generators and randomization. For example, if we want to test the concept of one-digit addition, we can let the computer randomly generate the two adders, and students will encounter different variants of the problem every time. Furthermore, virtually every component of an exam can be randomized, including the order of questions and the question formats. The questions can be auto-gradable, but they can also be manually graded if we want to inspect more closely into the students’ answers.

We believe cbt alleviates the workload of instructors as well as pressure of students. Of course, the instructors need to invest effort into creating the question generators. But once the generators are created, they can be reused countless times. As for the students, the problem of cheating would be significantly reduced because every two students could be taking completely different exams even if they are sitting side by side. Additionally, the students can use the same question generators to practice before taking a summative assessment, so they are benefited from taking many medium-stakes assessments instead of few high-stakes ones. All of these amazing results are made possible by the inherent randomness of the generators. CBT also greatly enhances academic resiliency. Specifically, there will be no need to find a single room to accommodate hundreds of students at once, which is a problem most universities encounter constantly. This is a photo taken at Berkeley, where a class had to clear an entire basketball court to make room for students to take exams. Besides, many DSP accommodations and rescheduling whole exam or individual assessments also become easier.

Cbt has the potential to revolutionize the entire education system. Today the way we give assessments is in a large high-stakes environment. The model is a fixed time, a term, a semester or a quarter, and variable learnings. Students get A’s, and B’s, and C’s but they also get D’s and F’s. It is broken because many students do not actually master the material even though they are able to pass the course. The model we propose is mastery learning. It is fixed learning and variable time. You could imagine stretching the semester beyond the actual term limit if the student has not shown mastery of the work. Or if students already master the material, they can take the final in the middle of the semester. This is totally up to them. And in the end, all students would get an A when they have mastered the material. This big idea can only be realized with the help of cbt, because if the students are taking exams at different times, it is impossible to write different paper exams in order to accommodate every student. But this is exactly what question generators are able to do. Namely, they can easily create countless numbers of different exams. Therefore, cbt is the future of assessments, and once you go computer-based assessment, you would never want to go back to writing paper exams.

Currently, our cbt team at Berkeley is working on implementing question generators using PrairieLearn. We categorized the questions into two sections, snap! and Python. If you are interested in how we are implementing Python questions, feel free to contact me! We are currently implementing dead python questions. As shown in the image here, students can write code freely in the blank space. Later on, we are going to develop live python. This gives the same experience as coding in a python IDE, where students can actually run their code before submitting. We are also going to explore the possibility of creating live Snap!, which gives the students flexibility to play with their code.

Since this is Snap!con, we will cover more about implementation of Snap! questions. Before we get to that, I will pass on to Shannon to talk about our amazing concept map, which is essential when we create new question generators.