Teaching Philosophy: How to Help My Students to Master the Substantial Core of Introduction to Computer Science (CS111)

Suiliang Ma

Rutgers University

I am a peer instructor teaching recitations for Introduction to Computer Science (CS111). Aside from being a peer instructor, I also tutor students taking other computer science classes. During my time as a peer instructor, I have noticed that there are high achieving students, who still struggle with the fundamental concepts that they should have mastered throughout the semester. The failure to learn the core concepts of CS111 become barriers for them in subsequent Computer Science Classes. Therefore, as a peer instructor of CS111, I strive to encourage and challenge students to help them master the substantial core concepts and ideas of CS111 so that they will really object oriented programming. Recently, collaborative learning practices emerged as a feasible solution for students to learn not just the concepts of programming, but the underlying logic behind the code.

The first strategy that I used aims at structuring a more efficient recitation based on the result of research on human memory and learning. The change of the degree of retention of students in a 40-minute learning episode is shown below:

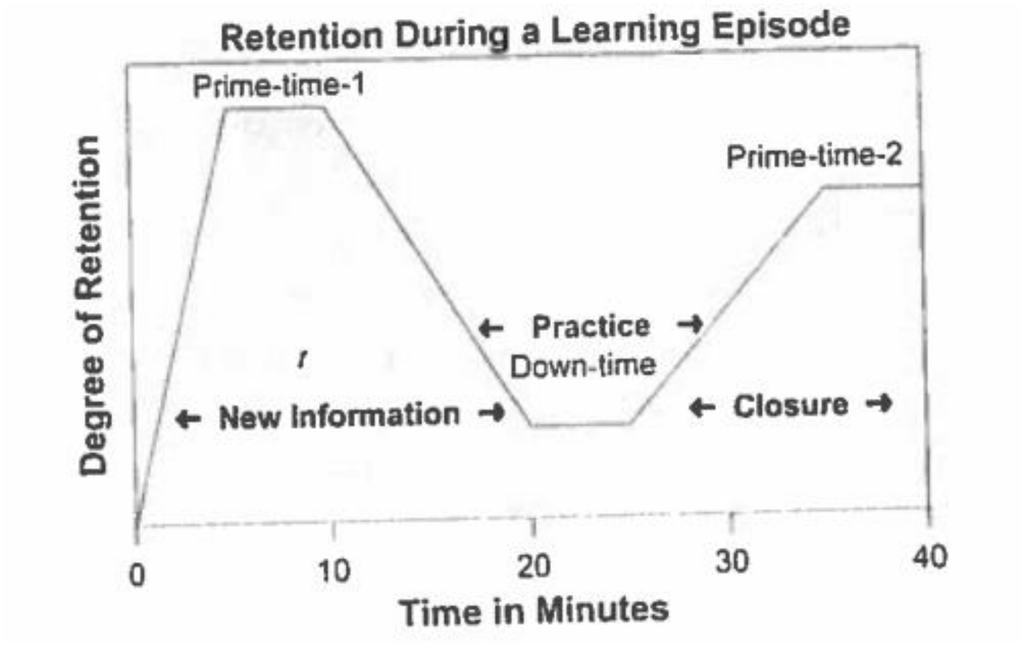

Figure I: Retention During a Learning Episode (Sousa, 96)

Therefore, to ease the teaching efficiency, an instructor is encouraged to introduce new information during the prime-time-1, and perform some practices or exercises during the down-time of the class, then end the lecture. (Sousa, 95, 96). While at Rutgers, the duration of CS111 recitation is 55 minutes instead of 40 minutes (figure I), the pattern of primacy-recency-effect would be similar. Rather than doing the review of the previous lectures in between, when students are at the lowest degree of retention, I put the review session at the beginning, which can help my recitation to be much more efficient. Therefore, I typically do a review of the knowledge that students learnt in the lectures, which takes in about 10 to 15 minutes, then ask the students to do some programming exercises for 20 to 25 minutes, then we go back to discuss about the exercises with some conclusions and reflections for the last 10 to 15 minutes. By doing this, I do the review when students maintain the top degree of retention; ask them to perform some programming tasks when their retention is relatively low; then make a conclusion and collect reflections of students at the end of the recitation so that they could learn something from the recitation.

Cooperative learning is another strategy that I use, based on the importance of cooperative learning and the hope that cooperative learning will benefit the students. Cooperative learning is very important for students learning computer science, because they will be very likely to work in groups for the careers. Also, cooperation tends to result in higher achievement, especially when people do cooperative learning in pairs (David W. Johnson & Roger T. Johnson & Smith, 19). Therefore, it is extremely necessary to practice their cooperative learning skills during recitations. Up to now, I have been focusing on promoting the individual accountability of students by giving individual programming exercises to each student, because given the current situation of students that they have not learned much about Java programming, it is better to hold them accountable first, then the cooperation with other students will be much effective. However, that doesn’t mean that I do not encourage the cooperative learning. I encourage students to communicate and learn from each other and communicate with one another. When the appropriate time comes, I will implement some programs to ask students to work on the assignment with a partner to finish tasks together, where they will need to learn how to maximize the learning of together and help each other by sharing resources. I will continue to emphasize the importance of collaboration and I hope that they realize the benefits of working in teams. The third strategy that I implement over the recitation is the emphasis of practices of programming. The practices of programming for CS111 are extremely important not only because practice makes people more comfortable understanding the fundamentals of Java programming, but to have a better understanding of the fundamental concepts of Java, students have to practice writing the code from beginning to end. However, learning the syntax of programming is not the focal point over the actual lecture as professors will spend the majority of time talking about theorems and concepts for the course with perhaps, given the best-case scenario, some codes demonstrating specific theorems and related concepts. Therefore, it is my job to demonstrate the importance of practices of programming over the recitations. Usually, I send the solution code to my students before the start of recitation. During recitation, I will spend up to 25minutes for students to work on their code. To finish the codes, students will need to understand the requirements of the code first, then try to apply what they have learned in the lectures, which belongs to the apply level of Bloom’s Taxonomy (Sousa, 257). If they can come up with different ideas for the same problem, they will also need to analyze on these different ideas, make a contract of them, and in the end, choose one that works the best to implement it. Here, analyze level and evaluate level of Bloom's Taxonomy are included (Sousa, 257). An example is included here:

```
public static int uniqueCharacters(String str){
        // your code here;
}
```

This method returns the number of unique Characters, which refer to the characters that happen exactly once in the string str. Let's assume that the input string str will only contain letters.

In order to finish this method, students will need to understand the method first. This is a static method that takes a string str as an input, and returns a variable of type int. The ability to understand the method header, or signature, is related to remember and understand level of Bloom Taxonomy. Then, students will need to recall all the knowledge they have learnt so far, and come up with some algorithms on solving this problem. This belongs to the apply level. Say, a student comes up with an algorithm that he can write a nested loop so that it can check every character against every other character, count the number of unique characters, and return the number. Another comes up with another algorithm that since we know there is a finite number of letters, namely, 52, if case sensitive here, he can map every single character that occurs in the input string to an element that represents this specific character. If there is more than one occurrence of the same character, he could just increase the number of that element to which the character maps. In the end, all he needs to do is to count the number of these elements that appear exactly once, and return the value. So right now, there are two different algorithms for solving the same problem. Students will need to analyze and contract them so that they can choose one to implement on their own based on the decisions of implementation style and the efficiency of the program. Therefore, apply and evaluate levels are involved here. As we can see here, even the implementation of a single method would require a high level of thinking behavior. Almost All the practices that I ask the students to do will contain the similar questions which aim to help students perform a divergent thinking. Therefore, the emphasis of practices of programming is the third method that I used to elicit a higher level of thinking.

As a teaching assistant, I really hope that students will be capable of coding on their own, which belongs to the create level of Bloom's Taxonomy (Sousa,257). Based on this goal, I use three strategies in the hope that they will be helpful for students to be able to code on their own.

References

Sousa,D.A. (2005) How the brain learns. *SAGE Publications*, 95-96

Sousa, D.A (2005) How the brain learns. *SAGE Publications*, 257-258

Johnson, D.W   Johnson, T.R   Smith, K. (2007) The State of Cooperative Learning in Postsecondary and Professional Settings. Educ Psychol Rev 19:15–29 DOI 10.1007/s10648-006-9038-8