Chapter 1 & Chapter 2 draft outline

Chapter 1

**[Objective]**

**[Function/feature] (teacher can upload question… student can solve the problem)**

**[Relevancy]/[Context]**

**[risks/mitigation]**

**[Define CS1 and CS2]**

Chapter 2 draft

**[Classical programming ways]**

Not quickly hit the key point ‘Parsons problem’. Readers only remember some other things like code tracing or Coached Program Planning, and they have no idea about Parsons problem. It may be better to explain these two methods when comparing with Parsons problem (‘Parsons problem integrates the respective advantages of both Coached Program Planning method and the code tracing method’ in the second paragraph).

Writing code, undoubtedly, is one of the core parts of the Science in Computing programme, and consequently, it is also the toughest part for learners, especially beginners. Students are not only required to have the ability to analyze the solving logic of problems, but also they need to be proficient in different coding syntaxes. In this case, students need to make many attempts to solve the programming problems, and they may easily give up during this period. To ameliorate this situation, two classical programming learning methods are introduced into the courses to improve students’ code-writing skills. Specifically, one of the methods is a preparation action for code writing, Coached Program Planning. This method guides students to analyze the problems and design the logical procedures with native-language style pseudocode for solving these problems [2, 3] so that the logical parts of codes can be partially separated from the syntax parts reducing the cognitive burden during coding to some extent. Another method is code tracing, which is to track the changes of variables by hand during the execution of codes [4]. Although this method does not have a direct effect on code writing and cannot be applied to every programming question because of its cumbersome procedure, it, through the accumulation of experience in code reading, still can provide some kind of auxiliary help in improving code writing skills.

**[Parsons problem structures]**

**[basic question model]**

Parsons problem is not a ‘new’ way since it was created in 2006. It is a little strange to write like this. After changing the previous paragraph, a more proper word needs to be chose to replace this one

Besides the above two programming learning methods, a new way, Parsons problem, was created to prepare students for writing code [5]. Instead of letting students directly write code, Parsons problem provides a set of code fragments – including the solutions to the questions and some distractors (some common errors) - for students to choose from and reorder by dragging and dropping [5]. During this period, students can get some instruction feedback for their reordered answers, and they need to repeat reordering until their answers are one hundred percent correct [5].

**[game-like design] [puzzle]**

**[distractor] + [connect the related distractors with the specific blocks of code]**

**[pre-scaffold]**

**[context]**

**[benefits]**

This method provides notable help in introductory programming study. To be more specific, the puzzle-like game-style Parson problem can improve students’ engagement and motivation in learning programming [5]. And with prepared code fragments and instant feedback, the levels of difficulty of the questions are reduced, and students are more likely to persist in programming instead of giving up halfway. Besides, Parsons problem can be of use to reduce cognitive load since students are only required to reorder the prepared code fragments instead of writing code directly [1]. And some context (fixed code) also can be provided to students to reduce cognitive load further [6]. In addition, Parsons problem integrates the respective advantages of both Coached Program Planning method and the code tracing method. Specifically, Parsons and Haden picked up an idea to include activity diagrams in the questions’ descriptions to help students to understand the solution logic of problems [5], which has a similar function to Coached Program Planning method. And since Parsons problem also requires students to read and understand the meaning of every code fragment, it also takes advantage of code reading just like the code tracing method. In other words, Parsons problem provides magnificent solution examples for students to learn from, giving them some reference material to think about solution steps when meeting some similar questions. Finally, Parsons problem also has an effect on helping students to cultivate good coding habits. For example, the distractors in Parsons problem can be used to show some improper variable names, which assists students to distinguish good names from bad names and train the habits of using meaningful and conforming naming rules names [5].

**[The benefits of using Parsons problem in Data Structures and Algorithms courses]**

Although Parsons problem makes great success in teaching programming, it is limited to only being used in introductory programming courses, and it has not been expanded in middle-level programming courses, for example, Data Structures and Algorithms courses. ~~From my~~ ~~perspective~~, Parsons problem still can demonstrate its superiority in Data Structures and Algorithms courses. Admittedly, students taking intermediate-level programming courses should be able to write code instead of just rearranging the order of provided answer blocks. However, because of the abstractness and universality of programming in this course, it also is a challenging task for students to write it directly (for example, recursion problems). Thus, it is of the essence to introduce Parsons problem to build a “bridge” for students to grow their capability to write code directly by themselves. But, since there are some differences between introductory programming courses with Data Structures and Algorithms courses, the previous Parsons problem in introductory programming courses does not fit the situation in Data Structures and Algorithms courses, and it is not suitable to apply the previous Parsons problem directly. Consequently, some new ideas should be introduced to Parsons problem. The detailed difference between the two courses and the limitation of the previous Parsons problem will be discussed in the following paragraphs.

Need some changes, briefly explain