Journal of Engineering Education Research
Vol. 17, No. 4, pp. 58~61, July 2014

A Web-based Tool for Teaching Computer Programming

Sehyeong Cho*,†
*Myongji University

ABSTRACT

This paper introduces a tool for effective teaching of introductory computer programming. In order for the class to be effective, we try to attain attention, relevance, confidence, and satisfaction based on Keller's ARCS model. A web-based tool is developed to help both the students and the instructors.

Keywords: computer programming, engineering education

I. Introduction

Computer programming languages are extremely difficult to learn, especially for students who have no background or experience in computer programming. One big problem is the lack of motivations. Motivations can be attained from various sources, either outside the course or inside the course. Keller's ARCS model[1] presents strategies for designing a class for attaining maximal motivations, based on attention, relevance, confidence, and satisfaction. We built a web-based tool so that students are better motived inside the course. Attention is of utmost importance, since each lab builds on the previous one, where losing attention for a few minutes may result in losing the whole class hour. Relevance is important because what they do should mean something to them; they can hardly be interested otherwise. Confidence is also important, since students need to make sure they are doing fine. Satisfaction gives positive feedback, resulting in self-propelled study activities.

We designed and implemented a web-based tool for teaching and learning C/C++ language programming with Keller's ARCS criteria and strategies in mind.

II. Keller's ARCS Model

Keller suggested a so called "ARCS" model, for attention, relevance, confidence, and satisfaction. ARCS model represents a set of strategies for instruction design for promoting maximal student motivation, and for sustaining it. Attention is attained at least in two ways. One is by perceptual arousal, and the other is by inquiry arousal. In order to stimulate perceptual arousal, the students need be surprised or get unexpected, interesting events. Inquiry arousal means stimulating curiosity by way of challenging questions or problems to be solved. In other words, we stimulate the learner's curiosity and therefore their attitude toward new problems and solutions.

Relevance is required to increase motivation by making them think the topic of study is relevant, familiar, and useful. By resorting to experience, learners can see how the new learning will use their existing skills. We can also try to make them think the subject matter has importance for them today, or maybe tomorrow. Another strategy would be to allow the learners to use different methods to pursue their work or allowing choice in how they organize it.

Confidence is important, because the learners should not be frustrated but succeed in some respect. The learners should be presented a degree of challenge that provides meaningful success. Strategies include 1) build up small successes gradually based on previous ones, 2) help the learners aware of precise performance requirements and evaluation criteria, 3) provide feedback as reward to internal attributions for success, and 4) make learners feel some degree of control over their learning and assessment, i.e., they should believe that their success is

Received 30 June, 2014; Revised 30 June, 2014
Accepted 30 July, 2014
† Corresponding Author: shcho@mju.ac.kr

a direct result of their effort they made

Satisfaction is based upon motivation, and comes by having the feeling of achievement and accomplishment. Therefore the learners should be provided with opportunities to use newly acquired knowledge or skill in a certain way and be satisfied. Appropriate feedback and reinforcements will help sustain the desired behavior. If learners feel good about their learning results, they will be motivated to learn. However, it is dangerous to reward overly, because too good compliments for bad performance may lead to unexpectedly bad behavior. Satisfaction is also closely related to confidence.

III. Strategies and Tools for attaining ARCS in Learning Programming

1. Attention

In a programming course, it is extremely difficult to maintain the learner's attention for an hour, if only lecturing is given throughout the class. One of the reasons, obviously, is that learners are passively engaged in the class during lecturing. More important reason, however, is that learning a language is not simply absorbing independent pieces of knowledge. Unlike knowledge-oriented subjects, each topic in a programming course builds upon the previously learned topic. Moreover, each step requires certain level of competence in order for the next topic be meaningfully learned. This is why we need reinforcement through actual laboratory exercises. In the past, one hour of lecture is followed by one hour of laboratory exercises plus homework assignment. This resulted in extreme inefficiency and ineffectiveness in both the lecturing and the laboratory. For this reason, we divided the lecture into small chunks each of which is followed by small exercises that will reinforce their understanding. The difficulty lies in how to efficiently mix lecturing and laboratory exercises, since too much time is wasted in transitioning from lecture to lab and back to lecture. In order to help this out, we designed a tool for easily create a problem, do and monitor exercises, automatically grade, and collect the result.

There are two types of web-based laboratory exercises. The first type is a simple question answering. Fig. 1 shows the interface for the instructor used to create a simple quiz. Usually they are multiple choice questions or short-answer questions, where the answer is relatively obvious. However, if wrong answer is selected, then the student gets a penalty. All that is required to correctly answer such a question is simply to pay attention.

The second type is fill-in-the-blank programming. In this type of exercises, students are given a simple C language program with a few blanks to fill in. When the blanks are filled in and the form is submitted, it is compiled and executed immediately.(See Fig. 3) This type of problem is usually intended to be very easy, and

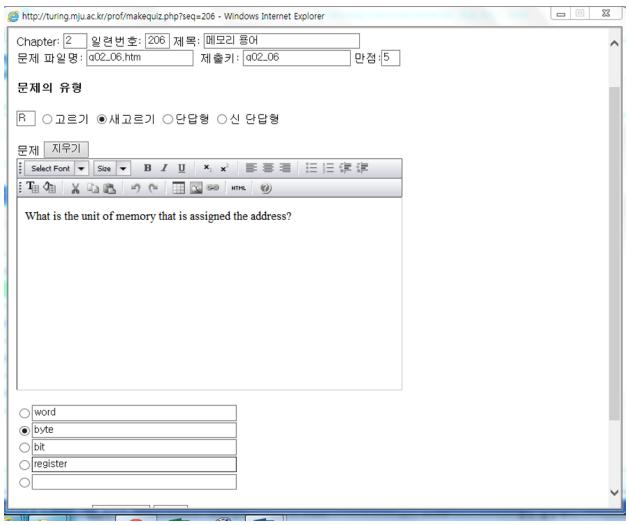

Fig. 1 The instructor's interface for creating a new question

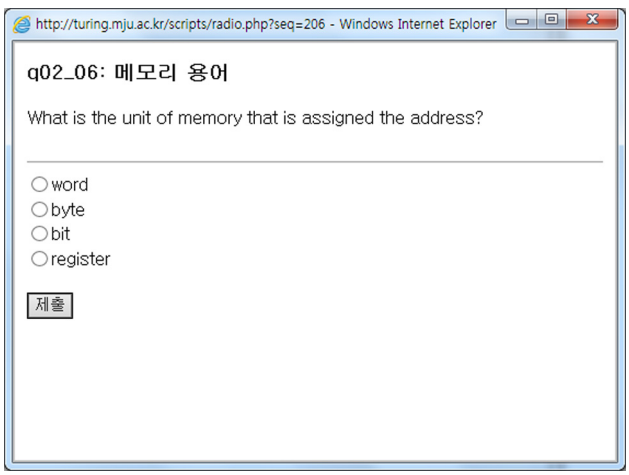

Fig. 2 The student interface for a multiple choice question

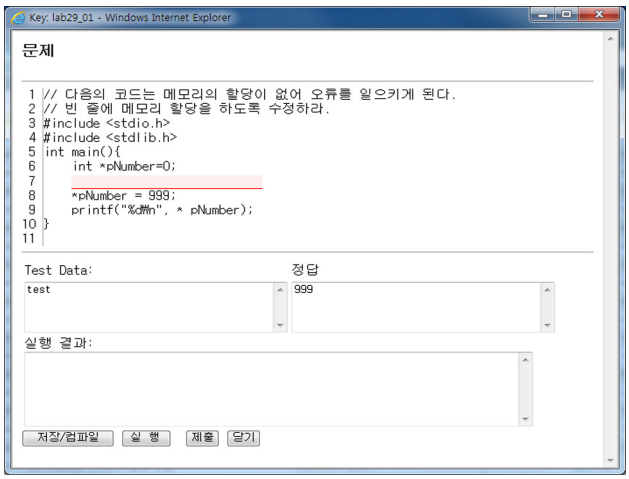

Fig. 3 Fill-in-the-blanks programming tool. Buttons at the bottom are: compile, run, submit, and close.

is not designed to do any serious problem solving. Its major role is to make sure the student keep his/her attention. The example shown in Fig. 3 is an exercise of memory allocation by "malloc" function.

On the instructor's part, all the results are gathered automatically and stored in a database for easier course management.

2. Relevance

As introduced in the previous section, one strategy for attaining relevance is to build on the previously acquired knowledge or skill of the student. Also, the learner should be able to appreciate what their program is doing for them. In order to resolve this problem, we created an executable flow-chart language called CFL, which stands for "C-like Flowchart Language". Instead of actually writing a C program in text, students in the beginning of the course use GUI-based flowchart language system(Fig. 4). Unlike traditional flowchart, CFL can actually be executed, either batch or step-by-step. They can watch the flow of control and see the variables change their values. Instructions provided are ordinary numerical expressions, Boolean expressions, assignments, decision(if-else), repetition loop, and function definition and invocation. We observed that students who first studied with CFL are slower in the beginning of the course, but they quickly catch up and do better than other students[2].

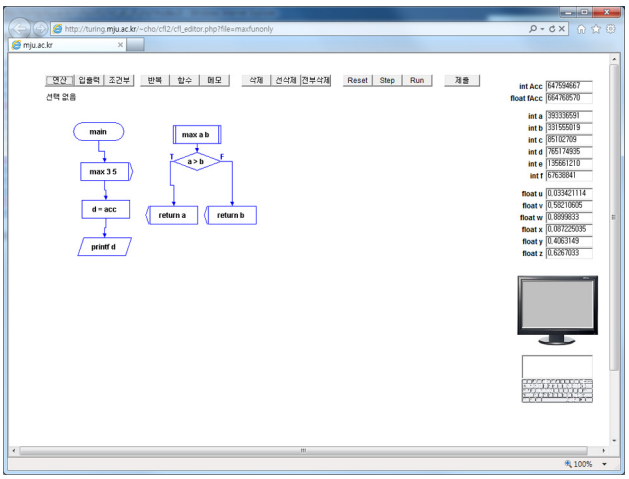

Fig. 4 Flowchart programming system for enhancing relevance

3. Confidence

Confidence is necessary for students in order to have interest in that topic. Lack of confidence leads to lack of actions. The problem is how to make them quickly "succeed" in their undertakings. If a task is too difficult, it is hard to succeed; if too easy, it does not help build strong confidence. We believe that the most important factor here is now to get rapid feedback when students make success in their tasks. The proposed web-based tool provides rapid feedback by using first an automatic grading tool, and then the actual grading system run by the instructor or the teaching assistant.

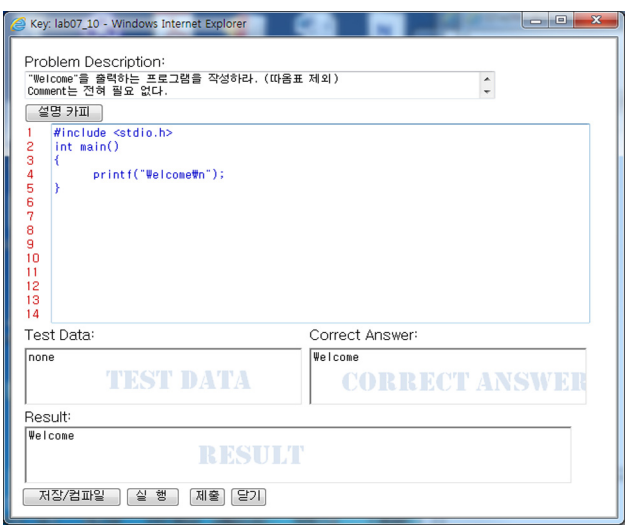

Fig. 5 Web based programming interface

One advantage of this web-based tools is that statistics is automatically obtained. Therefore the instructor is informed of the number of students who wrote the correct program, how long they took on the average, and the deviation in completion times.

4. Satisfaction

Satisfaction has to do with reward. Since we cannot give out any material reward, we decided to give out their performance in real-time, as shown in Fig 6. The instructor can choose either to show the detail or show only individual score plus the average and the deviation of the scores of the class.

The CFL programming also helps the students get the feeling of confidence and satisfaction because it is much easier to write programs.

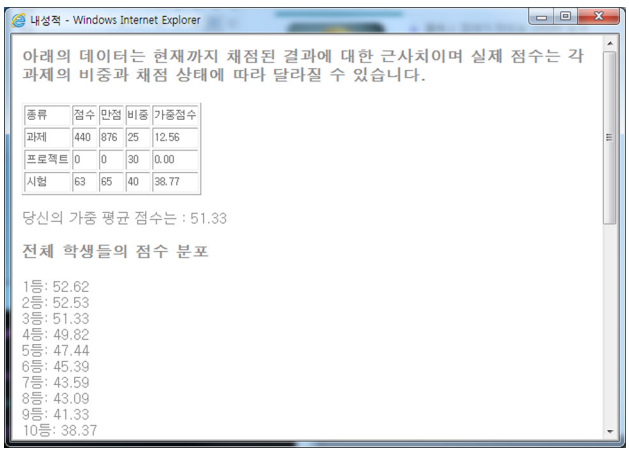

Fig 6. Realtime scoreboard for satisfaction

IV. Conclusion

A web-based tool presented here enables the instructors to easily assign homework and laboratory exercises to students, as well as collect submissions, test programs, record grade, and process class statistics. It also enables the students to approach the tasks with ease, and makes the task of submitting assigned work simple, and also enables the students to test their program as they submit it. It is also easy to see his or her status. More importantly, they help keep the attention of students, make them feel the topics and tasks are relevant to them, thereby acquiring confidence and get rewarded. As a result, the number of exercises has grown from 15 on the average to more than 200 small exercises. The failure rate dropped from around 30% to less than 20%.

References

1. John M. Keller, “Development and use of the ARCS model of instructional design,” Journal of instructional development, September 1987, Volume 10, Issue 3, pp 2-10
2. Sehyeong Cho, Yunseung Ryu, and Sang-Kyun Kim, “Learning C programming by using executable flowchart language,” in Proc. 2014 ASEE annual conference, Indianapolis 2014

Sehyeong Cho
Professor, Computer Engineering, Myongji University. Received BS (1981) and MS (1983) from Seoul anational University. Received Ph.D. (1992) from Pennsylvania State University. Worked as a senior research scientist at ETRI (1992~2000). Current research focuses on Computer Programming Education
Phone: 031-330-6779
E-mail: shcho@mju.ac.kr