Poster paper

Using Drag and Drop Puzzles to Teach Introductory Programming

Dale Parsons **Joy Gasson**
Department of Information Technology and Electrotechnology
Otago Polytechnic, Dunedin, NZ
dale@tekotago.ac.nz

Learning objects are becoming a common tool in the teaching of introductory programming. (Ford, 2004).

This was an attempt to create a learning tool for less able programming students. Over the years I have found students learning to program are slow at writing code, often frustrated by making trivial errors and inefficient in the structures that they create. They also make predictable programming errors.

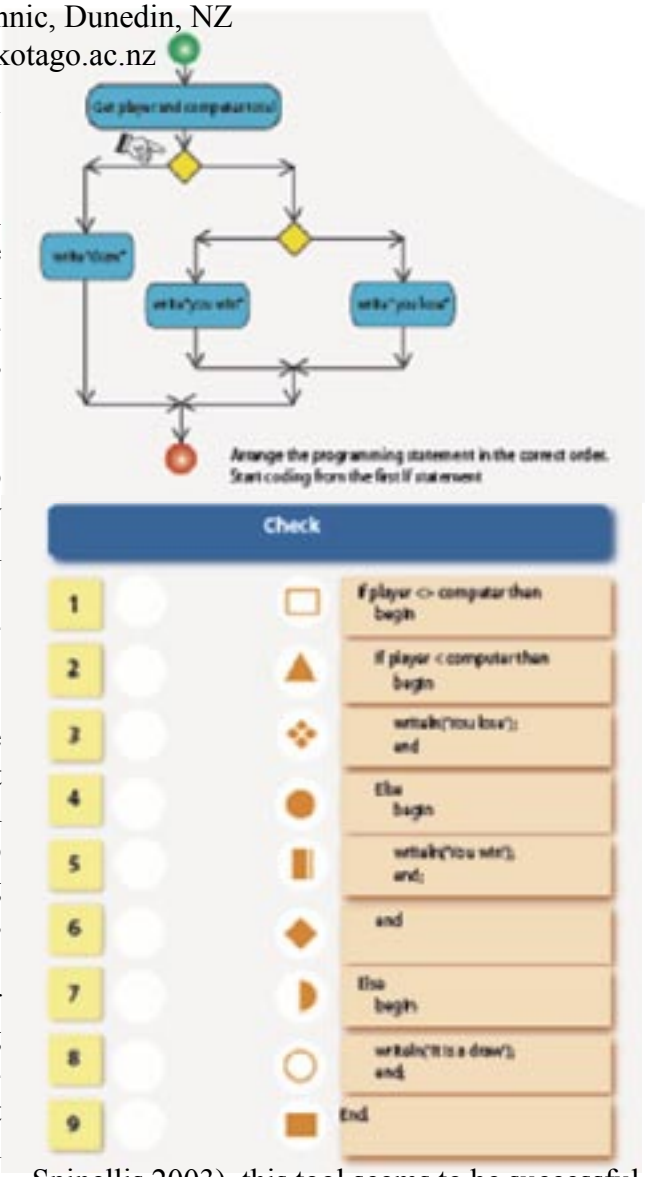
These drag and drop puzzles get students to pick and arrange correct lines of code so that they produce a "program" quickly. By showing them common errors alongside the correct line of code, I am trying to highlight the common misconception and direct them to the correct model.

This poster shows two of a series of puzzles. The Lab 5 puzzle uses an activity diagram of the problem. This makes the students use the most efficient structure, a Nested If. The diagram lacks the test conditions leading the students to interpret the diagram and work out the missing logical statements. This makes the student focus on the key issues.

The students get immediate feedback on their attempts and they keep dragging and dropping until the code is perfect. So they end up reading and creating good code. They can also print out the solution. These puzzles take less than 5 minutes to do whereas planning, coding, running and testing a problem like this would take a weaker student over an hour and they would not code it efficiently.

The second puzzle (Lab 12) on calling procedures and functions shows only the first line of each procedure and function declaration. This focuses the students on the key concept which is how to call a procedure or a function. With this quiz they can practise calling these procedures without having to be able to write them.

In conclusion, there is some debate in the literature about whether it is easier to read other people's code or write your own (Lister 2000, Spinellis 2003), this tool seems to be successful as an intermediary step where students assemble blocks of pre-written code.



The quizzes were created in Hot Potatoes http://web.uvic.ca/hrd/halfbaked/

Try them out at:
http://fred.tekotago.ac.nz/~dale/lab5.htm

References

Ford, L. (2003) Annual Joint Conference Integrating Technology into Computer Science Education. Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education Leeds, United Kingdom

Lister, R. (2000), On blooming first year programming, and its blooming assessment, in 'Proceedings of the Australasian Conference on Computing Education', ACM Press, pp. 158–162.

Spinellis, D. (2003), 'Reading, writing, and code', ACM Queue 1(7), 84–89.