JEE-Tutor: An Intelligent Tutoring System For Java Expressions Evaluation

Samy S. Abu Naser
Faculty of Engineering and Information Technology, Al-Azhar University, Gaza, Palestine

Abstract: This study presents an Intelligent Tutoring System called JEE-Tutor for teaching Java operator precedence, associativity and expressions evaluation to freshman students in the Faculty of Engineering and Information Technology in Al-Azhar University. An overview of the JEE-Tutor architectural design and user interface will be discussed. According to the success of other similar Intelligent Tutoring Systems, it is also hypothesized that students will be able to learn operator precedence, associativity, expressions evaluation and gain knowledge more quickly and effectively than students using traditional methods of teaching.

Key words: Artificial intelligence, tutor, intelligent tutoring system, E-learning, java programming

INTRODUCTION

An Intelligent Tutoring System (ITS) provides an individualized computer-based instruction to students (Abu Naser, 2006, 2001; Brusilovsky et al., 1996). These systems emerged from application of artificial intelligence techniques to the Computer Aided Instruction (CAI) systems (Abu Naser and Sulisel, 2000). The difference is that an ITS usually compares the student's work with expert solutions or strategies, models the student's probably knowledge of a domain and provides coaching or advice, taking into account what the student's knowledge state and preferred learning style. Depending on Artificial Intelligence (AI) and cognitive science, ITS became very popular and effective domain in Education for many reasons: Better student performance, student learn in less time and student is in the driver seat (Anderson et al., 1995; Wooff et al., 2001; Graesser and Person, 2001). For many years, there is a continuous development and evaluation of ITS (Shute, 1995; Shute and Glaser, 1990) for tutoring and monitoring programming in the field of Artificial Intelligence in Education. Programming requires a group of problem-solving and diagnostic strategies. The behavior in which a student writes code provides great insight into the way of his thinking. As a result, programming provides attractive area to study learning and cognitive processes (Koedinger, 2001; Scott et al., 1991). Among the objectives of this research is to gather the developments in the ITS, Cognitive Science and AI to make a useful intelligent tutor to help students understand Java programming expressions evaluation.

Usually textbooks do not present the steps required to solve a problem, but using visualizations (Abu Naser, 2008) and intelligent tutoring systems, student can learn and solve problems comfortably.

Architecture and design of JEE-tutor: We present the model and architecture for the Java expressions evaluation intelligent tutoring system, the knowledge base design, expert module design, feedback module design and the user interface design.

The JEE-Tutor could be part of a courseware on teaching Java programming language to undergraduate freshman students at the university-level.

JEE-Tutor knowledge base design: Here, we describe the Knowledge Base architectural model for JEE-Tutor. In this study, we have concentrated on the syntax of expressions, operator precedence, associativity and evaluation of Java expressions.

Java has well-defined rules for specifying the order in which the operators in an expression are evaluated when the expression has several operators (Liang, 2008; Thomas, 2006; Horstmann, 2006).

Precedence order: When two operators share an operand the operator with the higher precedence goes first. For example, 1+2*3 is treated as 1+(2*3), whereas 1*2+3 is treated as (1*2)+3 since multiplication has a higher precedence than addition.

Associativity: When two operators with the same precedence, the expression is evaluated according to its associativity. For example x = y = z = 13 is treated as x = (y = (z = 13)), leaving all three variables with the value 13, since the - operator has right-to-left associativity (and an assignment statement evaluates to the value on the right hand side). On the other hand, 60/2/3 is treated as (60/2)/3 since the operator has left-to-right associativity.

We have stored the operators of Java, its precedence and associativity in the knowledge base (Fig. 1).
Fig. 1: Operator precedence and its associativity

For every problem in the knowledge base, we have stored the possible solution, possible errors for specific categories and possible hints for each error. Figure 2 shows a problem example with a solution and some errors.

The problem stored in the knowledge base can be classified into 5 levels of difficulties. Difficulties mean more operators, mixed associativity types and different level of precedence, are involved in the problem. The Expert model determine which problem level should be given to the student.

In order for the student to solve a problem, he must first determine operator order of evaluation by filling the table as shown above. Second, the student should fill the second table by evaluating each operator at a time and third, put the output of the expression in the last line of the second table as shown in Fig. 2.

**JEE-Tutor expert module:** JEE-Tutor expert module is an expert system that is used to determine the student level of understanding from the problem statement, the problem specification and student's answers. When the student reaches a certain score answering at the current level of difficulty, for example 80% or more, the expert system increases the level of difficulties of problems to be given for the student. On the other hand, if the student did not reach a minimum score at certain level say 50%, the expert system branches the student to the tutorial on the subject involved in the problem. When the student finishes the tutorial, the expert system permits the student to go back to the questions mode. The score of the student, the level of difficulties and his name are all stored in a database for further analysis in the future when the student comes back and use the JEE-Tutor. So, the database reflects the actual level of every student as a result of all previous sessions.

**JEE-Tutor feedback module:** Effectiveness of the system depends heavily upon its feedback timing and style. Timing refers to when the student is given a response to the solution. When the feedback is presented to the

Student solution

<table>
<thead>
<tr>
<th>Operator</th>
<th>=</th>
<th>%</th>
<th>*</th>
<th>++</th>
<th>-</th>
<th>--</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation order</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

JEE-Tutor: "There is an associativity problem with ++ and --. The order of evaluation is Right to Left. Do you remember that rule?"
Student: Yes.
JEE-Tutor makes the correction and proceeds with rest of the operators table.

<table>
<thead>
<tr>
<th>Operator</th>
<th>=</th>
<th>%</th>
<th>*</th>
<th>++</th>
<th>-</th>
<th>--</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation order</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

JEE-Tutor: "There is an associativity problem with % and *. The order of evaluation is Left to Right. Do you remember that rule?"
Student: Yes.
JEE-Tutor makes the correction and proceeds with rest of the operators table.

<table>
<thead>
<tr>
<th>Operator</th>
<th>=</th>
<th>%</th>
<th>*</th>
<th>++</th>
<th>-</th>
<th>--</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluation order</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

JEE-Tutor: "There is an associativity problem with - and *. The precedence of * is higher than -. Do you remember that rule?"
Student: Yes.
JEE-Tutor makes the correction and student continues with the solution of the problem by filling the second table.

Fig. 3: A dialogue between JEE-Tutor and the student.

JEE-Tutor user interface: The interface of intelligent tutoring systems is a very important factor that we gave it a careful consideration during the design of JEE-Tutor. The user interface is based on a presentation format implemented in many popular Integrated Development Environments used by professional programmers (Conlan et al., 2002). Upon connecting to JEE-Tutor website, the student's browser displays the working environment for JEE-Tutor. An appropriate skill-level problem is selected or the problem that last attempted is presented to the student.

Fig. 4: JEE-Tutor user interface
The student solves the problem in the Student Solution window. Once the student fills the operators evaluation order table and he press the Check button, the expert module will determine the appropriate response based on the diagnosis of the student solution. A dialogue will be initiated between the JEE-Tutor and the student in case of errors. The expert system module informs the user about each error and ask the user whether he remember the rule of precedence or associativity for that operator. If the student answers indicates that he does not remember that rule, the expert system using pattern matching template analyze the student response and display the operator precedence table. Before the expert module proceeds with the next error, it corrects the previous error for the student. Once all errors are corrected, the user will be given control to fill the table of expression evaluation to find its final answer.

The intelligent feedback of the expert module is sent to the student’s output window. The student, at any time, may explicitly request from JEE-Tutor to view the solution. Exit from the current problem and ask for a new one; furthermore, the student can view his performance based on statistics including problems attempted, problems solved, number of attempts on a problem and problem difficulty. The JEE-Tutor user interface is shown in Fig. 4.

CONCLUSIONS

In this study, we have presented recent developments related to the JEE-Tutor Intelligent Tutoring System for teaching students Java Expressions Evaluation. JEE-Tutor is based on sound theories, pattern recognition techniques, error detection and correction strategies as in (Sykes and Franek, 2003; Aleven and Koedinger, 2002). Even though an evaluation of the JEE-Tutor has not been done yet, the implemented examples discussed clearly demonstrates the potential of JEE-Tutor. This research is significant since it has the potential to be applied to many programming courses at the university-level.

FUTURE WORK

JEE-Tutor has not yet been evaluated as to its effectiveness as a tutoring tool. A full evaluation is planned during 2008, to be taken with an introductory Java programming class. Additional intelligent tutoring systems for Java programming should be invested to enable students overcome their difficulties that are faced in first programming languages.

REFERENCES